Rheza's Notes Documentation

Release 2019

Rheza Budiono

Apr 04, 2020

Notes

CHAPTER 1

Logistic Regression

In this section we'll learn our first classification model class: logistic regression. Our approach will be to "invent" logistic regression for ourselves.

Here's how we'll do that:

- Introduce 0/1 classification loss and the risk expression under this loss
- See that we minimise risk under 0/1 loss by choosing the class with largest posterior probability
- Derive the logistic function as a way to model the log probability ratio between two classes
- Derive expressions for log likelihood, surrogate loss, and the gradient-based learning rule
- Generalize logistic regression to multi-class scenarios using the softmax function

1.1 0/1 Loss & Risk

0/1 loss is exactly what it sounds like: if we predict the correct class, we suffer no loss, and if we're wrong, we suffer loss equal 1. Formally:

$$l(h(x), y) = 0 \quad \text{if } y = h(x)$$
$$l(h(x), y) = 1 \quad \text{if } y \neq h(x)$$

Notation

- l refers to loss
- h refers to our decision rule, it is a function of observation x
- y refers to the true label for observation x

The expression for the risk of 0/1 loss given a joint probability distribution and a classifier h(x) is:

$$\begin{aligned} R(h) &= \int_x \sum_{c=1}^{C} l(h(x), c) p(x, y = c) dx \\ &= \int_x \sum_{c=1}^{C} l(h(x), c) p(y = c|x) p(x) dx \\ &= \int_x R(h|x) p(x) dx \end{aligned}$$

Above we see that it in order to choose h to minimise R(h), it suffices to choose h which minimises R(h|x) for each x. For a given x:

$$\begin{aligned} R(h|x) &= \sum_{c=1}^{C} l(h(x), c) p(y = c|x) \\ &= 0 \cdot p(y = h(x)|x) + \sum_{c' \neq h(x)} 1 \cdot p(y = c'|x) \\ &= 1 - p(y = h(x)|x) \end{aligned}$$

In words, the conditional risk of 0/1 loss is equal to the probability that your prediction h(x) is wrong. It's now clear that the way to minimise R(h|x), and thus R(h), is by predicting for each x the class c with the largest posterior p(y=c|x). Formally, our optimal classifier h^* is given by:

$$h^*(x) = \text{argmax}_c p(y = c|x)$$

We can express h^* as a "decision rule" in terms of probability ratios:

$$h^*(x) = c^* \quad \text{iff} \quad \forall c : log\Big(\frac{p(y = c^*|x)}{p(y = c|x)}\Big) \geq 0$$

1.2 Logistic Regression as modelling posteriors

Let us now consider the binary classification scenario. Our work above tells us that the optimal classifier is described as follows:

$$h^*(x) = 1 \quad \text{iff} \quad log\Big(\frac{p(y = 1|x)}{p(y = 0|x)}\Big) \geq 0$$

What we want now is to model the log probability ratio above using a linear function of x:

$$w \cdot x = log\Big(\frac{p(y = 1|x)}{p(y = 0|x)}\Big)$$

We can rearrange the equation to get an expression for the posterior:

$$p(y = 1|x) = \frac{1}{1 + exp[-w \cdot x]}$$

Notice that, as we'd expect, if $w \cdot x = 0$, then $p(y = 1|x) = \frac{1}{2}$. That is, if we're on the decision boundary, then both classes are equally likely.

1.3 Negative Log Likelihood Loss

We now turn to the problem of finding "good" w. Since the true joint probability distribution is not known, we cannot directly minimise risk. Instead, we do the next best thing: empricial risk minimisation - that is, we minimise the loss on some training dataset. Suppose we have been given a labeled dataset with N examples.

Notation

- X is the N-by-d matrix of training observations
- y is the N-dim vector of training labels
- w is the d-dim vector of weights
- $\sigma(z) = \frac{1}{1+exp[-z]}$ denotes the logistic function

---

It is computationally infeasible to directly minimise 0/1 loss so we instead employ a surrogate loss: negative log likelihood loss (i.e., we want to maximise the likelihood of the training examples). Negative log likelihood loss is given by:

$$L(w) = -logp(y|X) = -\sum_{i=1}^{N} y_i log(p(y_i = 1|x_i)) + (1 - y_i) log(p(y_i = 0|x_i)$$
$$= -\sum_{i=1}^{N} y_i log(\sigma(w \cdot x_i)) + (1 - y_i) log(1 - \sigma(w \cdot x_i))$$

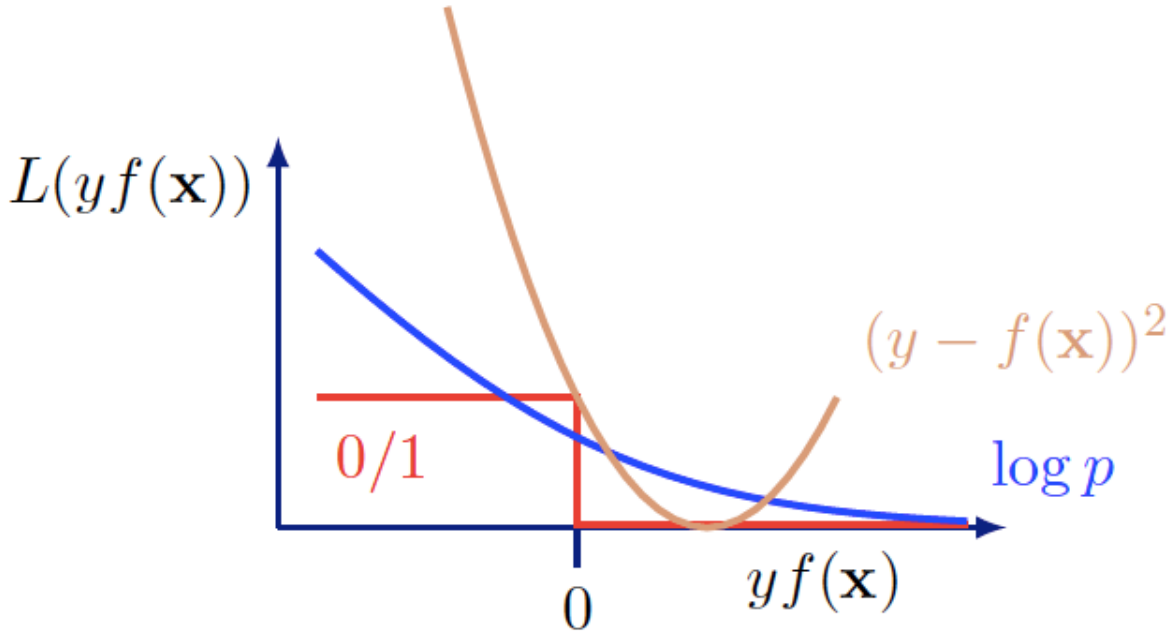


Fig. 1: Comparison of the surrogate loss, the original loss, and square error loss (Shakhnarovich, Slide 7, 2018)

Notice that although the expression looks complicated, it's actually quite simple. For each training example, we want to add the log likelihood of the label y_i given the observation x_i. The expressions y_i and $(1 - y_i)$ effectively act as if statements that respectively say "if $y_i = 1$, add $logp(y_i = 1|x_i)$" and "if $y_i = 0$, add $logp(y_i = 0|x)$."

Now that we have this surrogate loss in place, we can derive a gradient-based learning rule for w. (After first doing the scalar partial derivative,) we get (the update is for a single training example):

$$-\nabla_w L(w; x_i, y_i) = [y_i - \sigma(w \cdot x_i)]x_i$$
$$w^{t+1} = w^t - \eta \nabla_w L(w; x_i, y_i)$$

This learning rule has a nice, intuitive, geometric interpretation. Notice that geometrically, w represents the normal vector of the decision boundary (in the direction of training examples with $y_i = 1$).

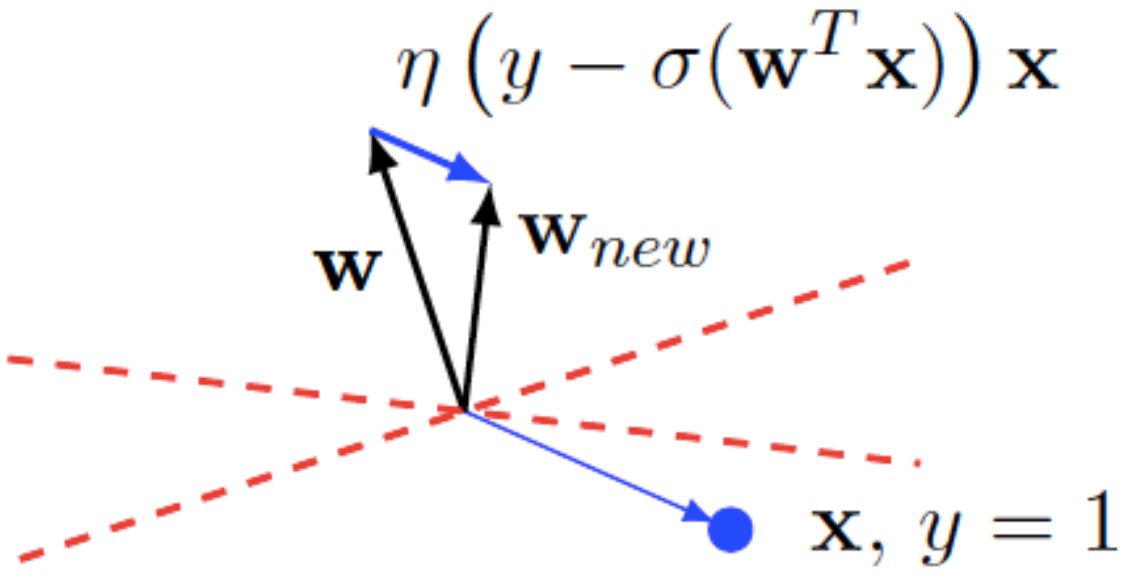


Fig. 2: (Shakhnarovich, Slide 7, 2018)

Suppose we're doing an update based on a single training example x_i with $y_i = 1$. So the update equation is given by $w^{t+1} = w^t + [1 - \sigma(w \cdot x_i)]x_i$. Since $0 < \sigma(w \cdot x_i) < 1$, the above update always adds a scaled version of x_i to w. This has the effect of "pulling" w towards x_i by some force. The magnitude of the force depends on how well the classifier currently classifies x_i. Consider the two extremes. If our classifier is doing very well, that is $\sigma(w \cdot x_i)$ is close to 1, x_i hardly pulls on w. If our classifier is doing very poorly, that is $\sigma(w \cdot x_i)$ is close to 0, x_i pulls very hard on w. Updates based on training examples with with $y_i = 0$ "push" w instead of pulling it. Quick sanity check: how will the magnitude of the push depend on $\sigma(w \cdot x_i)$ when $y_i = 0$?

The net effect of all this pushing and pulling is that w will roughly point in the direction towards all the training examples with label $y_i = 1$ and away from all the training examples with label $y_i = 0$.

1.4 Generalizing to multi-class with softmax

Notice that logistic regression only works in the binary classification case. But we can easily generalize to the multi-class scenario using the softmax distribution.

$$p(y = c|x) = \frac{exp[w_c \cdot x]}{\sum_{i=1}^{C} exp[w_i \cdot x]}$$

Again, we use the negative log likelihood loss to get a learning rule.

Notation

$\delta_{cy_i} = 1$ if $c = y_i$, else $\delta_{cy_i} = 0$ (Kronecker-Delta)

$$-\nabla_{w_c} L(w; x_i, y_i) = [\delta_{cy_i} - p(y = c|x_i)]x_i$$
$$w^{t+1} = w^t - \eta \nabla_w L(w; x_i, y_i)$$

The geometric intuition we have for the binary class case still applies here in the multi-class case; w_c is "pulled" in by training examples with $y_i = c$ and "pushed" away by all other training examples.

CHAPTER 2

Information Theory

I was introduced to information theory in David McAllester's Fundamentals of Deep Learning class (which I loved). The slides are good (but terse - you'll need to put in some work). The problems are excellent (in my humble opinion).

Chirs Olah's Visual Information Theory is what made information theory "click" for me. The optimal coding interpretation of information theory, and the corresponding visualizations, were extremely helpful to my understanding. The following is a cheat sheet of useful intuitions about definitions in information theory.

Entropy. This is the average code length of the optimal code for a given probability distribution.

$$H(p) = E_{x\sim p(x)} - \ln p(x) = \sum_x p(x) \cdot - \ln p(x)$$

Cross Entropy. This is the average code length if we sample from distribution p and communicate using the optimal code for distribution q

$$H(p, q) = E_{x\sim p} - \ln q(x)$$

KL Divergence. You sample from p. The KL is how much larger the average code length is when you use the optimal code for distribution q as opposed to the optimal code for distribution p.

$$\begin{aligned} KL(p, q) &= H(p, q) - H(p) \\ &= E_{x\sim p} \ln \frac{p(x)}{q(x)} \\ &\geq 0 \end{aligned}$$

Conditional Entropy. This is the entropy of r.v. X given that we have observed the state of Y.

$$\begin{aligned} H(X|Y) &= E_{y\sim p(y)} E_{x\sim p(x|y)} - \ln p(x|y) \\ &= E_{(x,y)\sim p(x,y)} - \ln p(x|y) \end{aligned}$$

Mutual Information. How much does X tell me about Y and vice versa? The channel capacity interpretation says that the mutual information tell us how much information X can carry about Y (and vice versa). This interpretation is

useful when thinking about rate distortion autoencoders.

$$\begin{aligned} I(X,Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \\ &\geq 0 \end{aligned}$$

Data processing inequality. Processing data (beyond trivial injective mappings) results in information loss. The proof is quite fun, and provides some insight into why the above (stronger) statement is true.

$$\text{For any function f:} \quad H(f(y)) \leq H(y)$$

To be continued with key inequalities and other stuff as I learn it...

CHAPTER 3

Catastrophic Forgetting

A summary of Alleviating Catastrophic Forgetting Using Context-Dependent Gating and Synaptic Stabilisation (Masse, Grant, Freedman). With my interpretations and thoughts sprinkled in. Jumping-off points are written as footnotes.

3.1 Intro

Without additional measures, neural networks are pretty bad at *continual learning* - the ability to learn new tasks without forgetting how to do previously learned tasks. This failure to continually learn is what we call *catastrophic forgetting*.

Catastrophic forgetting occurs because as a neural net learns a second task, its parameters move towards the optimum for the second task - and thus away from the optimum for the first task. This problem gets even worse as you learn more and more tasks.

One moderately successful remedy is weight stabilisation. The idea is that when learning a new task, you want to constrain parameter change in order to minimise the forgetting of previous tasks. Furthermore, you want to constrain important parameters more (important to previous tasks). The figure below shows the success of two stabilisation techniques at alleviating forgetting on the permuted MNIST task (the two techniques will be expounded on later).

pMNIST

Permuted MNIST (pMNIST) is an example of a "input reformatting" problem. Sequential learning is tested on identical input/output semantics, but changing input format. In pMNIST, the image pixel intensities (input semantics) and digit labels (output semantics) are kept the same, but the pixel locations (input format) are changed. Specifically, each new task has a fixed permutation that is applied to all input images.

The innovation presented by the paper is to pair weight stabilistion with context-dependent gating (XdG). In XdG, the neural network uses (randomly sampled, potentially overlapping) subsets of the neural network for each task. The idea is that we want to ease the burden on parameters - each parameter is now responsible for learning fewer tasks,

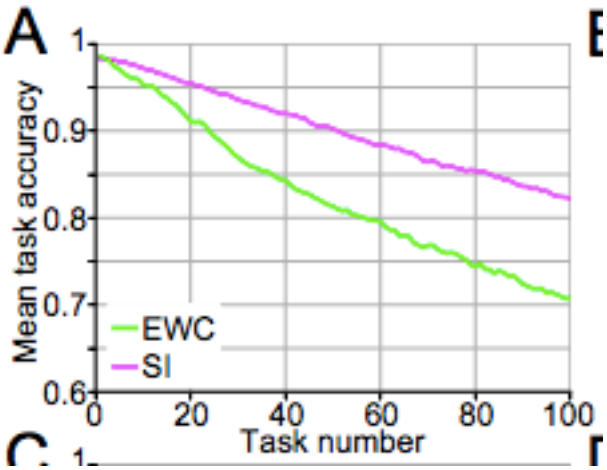


Fig. 1: Figure 2A in the paper.

and thus doesn't move as much from its last position (after learning the most recent task). The pairing of stabilisation and XdG results in a breakthrough in alleviating forgetting, as you can see in the figure below.

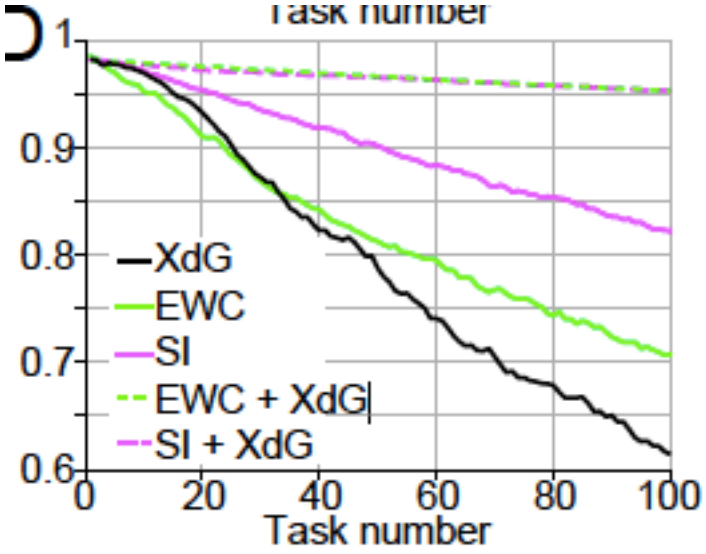


Fig. 2: Figure 2D in the paper.

With the efficacy of the method established, we turn to understanding each remedy individually and how they combine.

3.2 Weight Stabilistion

Weight stabilisation can be thought of as a form of regularisation that penalises moving away from previous parameters. Moreover, moving away from important parameters is punished more heavily than moving away from less important parameters. Mathematically, we add a penalty like so:

$$L = L_k + c \sum_i \Omega_i (\theta_i - \theta_i^{\text{prev}})^2$$

Notation

- L_k is the unstabilised loss while learning task k
- θ_i is parameter i
- c is a scaling hyper-parameter
- Ω_i is the "importance" of parameter i to previously learned tasks

How is Ω_i computed? Firstly, the importance of θ_i to each previous task is simply summed to form its total importance to all previous tasks. Assuming we're currently learning task k,

$$\Omega_i = \sum_{m<k} \Omega_i^k$$

Now, there are different ways to define Ω_i^k, the "importance" of parameter i to task k. This paper uses two different methods - elastic weight consolidation (EWC) and synaptic intelligence (SI).

Elastic weight consolidation (EWC) defines importance as:

$$\Omega_i^k = E_{X \sim D^k, y \sim P_\theta(y|x)} \Big(\frac{\partial log p_{\theta_i}(y|x)}{\partial \theta_i} \Big)^2$$

That is, we take the diagonal elements of the Fisher Information matrix:

$$F = E_{X \sim D^k, y \sim P_\theta(y|x)} \Big(\frac{\partial log p_{\theta_i}(y|x)}{\partial \theta_i} \Big)^2$$

Notation

- D^k is the input data for task k
- $p_{\theta_i}(y|x)$ is the output distribution defined by the neural net

This definition of "importance" bears an intuitive interpretation. If slightly perturbing a parameter results in a significant change in the output distribution, that parameter is important to maintaining the output distribution (and thus not forgetting it).[1]

Synaptic intelligence (SI) defines importance as:

$$\Omega_i^k = max\Big(0, \frac{w_i^k}{(\Delta\theta_i)^2 + \xi}\Big)$$
$$\Delta\theta_i = \sum_t (\theta_i(t) - \theta_i(t-1))$$
$$w_i^k = \sum_t (\theta_i(t) - \theta_i(t-1)) \cdot -\frac{\partial L_k(t)}{\partial \theta_i(t)}$$

Notation

- $(\Delta\theta_i)^2$ is the normalising term
- ξ is the damping hyper-parameter

[1] The point of EWC seems to be to minimise the shift of the output distribution $p_\theta(y|x)$ - so can we instead use a KL-divergence penalty such as $KL(p_{\theta_t}(y|x), p_{\theta_{t+1}}(y|x))$? Can we phrase other ideas presented in this paper in an information theoretic manner?

- t here refers to batch number (within training for a single task)

This definition of importance needs a little more unpacking than EWC. Let's start by inspecting w_i^k.

$(\theta_i(t) - \theta_i(t-1))$ represents the direction (sign) and magnitude of the parameter change. But since this term will get normalised anyways, only the direction of the change really matters.

$-\frac{\partial L_k(t)}{\partial \theta_i(t)}$ is meaningful both in direction and magnitude. As long as we're not at a local extremum, its direction indicates which way to go from $\theta_i(t)$ in order to locally decrease loss. Its magnitude tells us how much a small step in that direction will decrease the loss.

It's important to realise that $(\theta_i(t) - \theta_i(t-1))$ represents the parameter change from the just-completed batch, while $-\frac{\partial L_k(t)}{\partial \theta_i(t)}$ roughly represents the desired parameter change for the upcoming batch. If their signs disagree, that means that θ_i needs to reverse course - meaning that the last parameter change was not useful!

We can thus interpret Ω_i^k here as capturing how much θ_i moved in a useful direction while learning task k. There seems to be the underlying assumption that if θ_i moved a lot in a useful direction while learning task k, that parameter must be important to task k.[2]

3.3 Gating

The paper presents a sequence of ideas naturally leading up to context-dependent gating (XdG).

Context signalling is the first idea presented. It is motivated by the idea that catastrophic forgetting might partially be the result of the neural net not knowing what context its currently being tested on. So we invent a "context signal," a one-hot vector encoding what context is currently being tested. The paper says that this one-hot vector is then "projected" onto the hidden layers, and that the weights projecting the context signal onto the hidden layers could be trained by the network. As you can see in Figure 2B below, context signalling combined with stabilisation is an improvement upon stabilisation alone.

Question

I'm actually not sure what this "projecting" means. I initially thought that projecting meant applying a differentiable context-dependent mask onto the hidden layers, but how can such a mask be applied without training?

The **split network** is motivated by a desire to decrease the number of tasks each parameter is involved in learning. The network is split into five sub-networks with 733 hidden neurons each (so that the total number of weights remains the same). One sub-network is used for each task (with the other hidden neurons zeroed). Choosing the number of sub-networks involves a trade-off between allevating forgetting (more sub-networks) and maintaing representational power per task (less sub-networks, thus more hidden neurons per task). As you can see in Figure 2C below, split networks in combination with context signalling and stabilisation is our best remedy yet.

Context-dependent gating (XdG) assigns unique (potentially overlapping) sub-networks for each task. It does this by gating (i.e. zeroing) $X\%$ of hidden neurons, randomly chosen for each task.[3] In the paper, $X = 80$ to allow fair comparison with the split network. Again, the choice of X is a trade-off between alleviating forgetting via keeping more weights fixed, and maintaing per-task representational power by keeping more hidden neurons active.

[2] Why does SI work better than EWC? To me, EWC seems better justified than SI . The underlying assumption behind SI seems iffy. Yes, if a parameter needs to be changed a lot in order to perform a task, it must be important to doing that task well. But what if we have an already-learned parameter which is also important for the current task? For example, weights in a CNN that act as edge detectors seem useful for any visual task.

[3] Is XdG related to dropout? They both seem intended at distributing representational power. They both seem to be mixtures of experts where experts share paremeters. They differ in how they combine experts. XdG uses only one expert at a time, since contexts are discrete and are observed with total certainty. On the other hand, dropout assumes total ignorance about the context (and context refers to batch-wise distribution). That is, dropout always gives equal weight to each expert. I'm not sure how insightful this observation is, but it seems interesting… Reading into the literature on mixture models seems relevant to the problem of applying XdG to transfer learning.

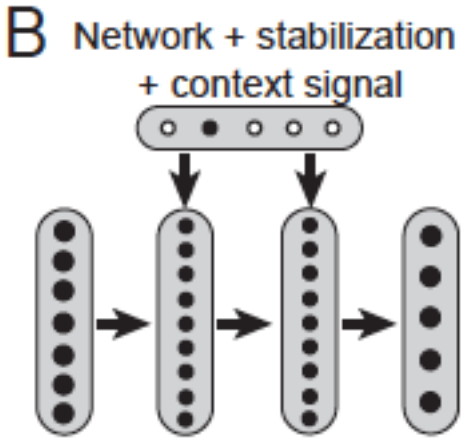


Fig. 3: Figure 1B in the paper.

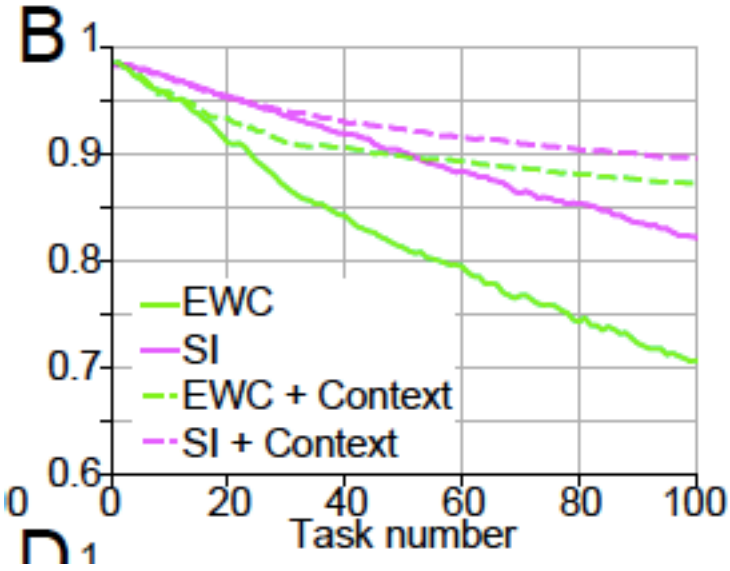


Fig. 4: Figure 2B in the paper.

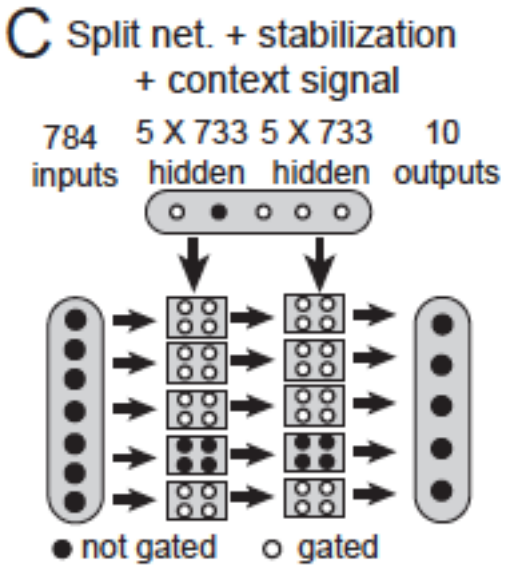


Fig. 5: Figure 1C in the paper.

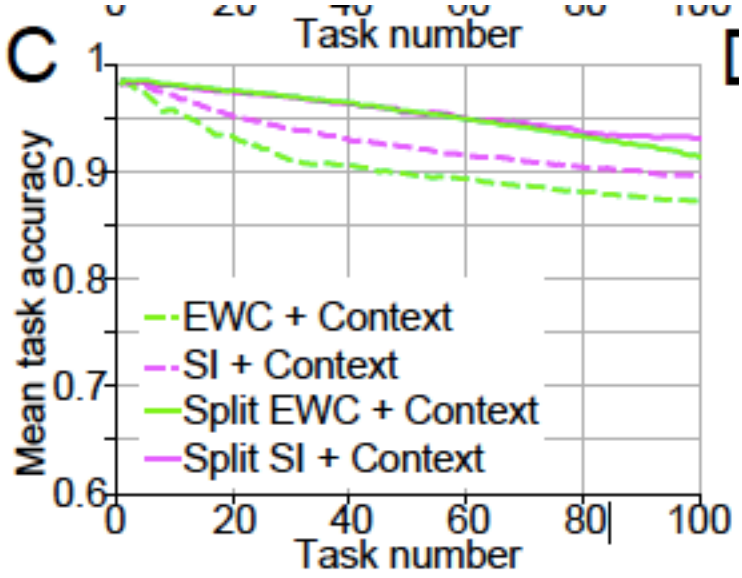


Fig. 6: Figure 2C in the paper.

XdG combined with stabilisation gives the best results yet. Interestingly, XdG only works when combined with stabilisation...

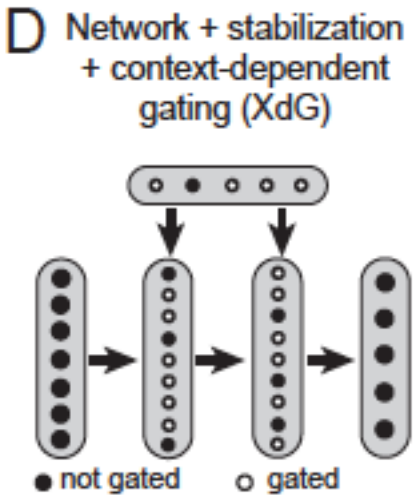


Fig. 7: Figure 1D in the paper.

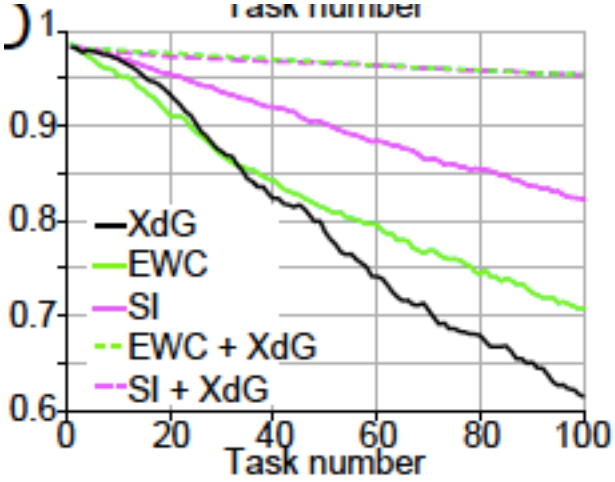


Fig. 8: Figure 2D in the paper.

The paper shows that the XdG and stabilisation method works on two other tasks. The first task was sequential learning of ImageNet (the network learned 10-class subsets of 1000 total classes). The second task was a series of eye saccade tasks. Notably, XdG and stabilisation successfully alleviated forgetting on a network trained using RL (to perform the second task).

3.4 The interaction between XdG and stablistion

This is my favorite part of the paper, where the authors try to understand why XdG combined with stabilisation works better than stabilisation alone. They do this by making a series of hypotheses, and testing them empircally - science!

The core argument is "that to accurately learn many sequential tasks with little forgetting, the network must balance two competing demands:"

1. "it must stabilise synpases that are deemed important for previous tasks"
2. "yet remain flexible so that it can adjust synaptic values by a sufficient amount to accurately learn new tasks."

The importance of the first demand, stabilising important weights, was empirically tested via an experiment in which individual weights in a trained network are perturbed by a fixed amount and the corresponding change in accuracy is measured. As expected, the authors found a strong negative correlation between the synaptic importance of the perturbed weight and the corresponding change in accuracy (R = -0.904).

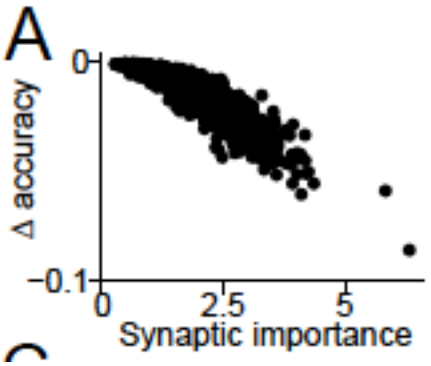


Fig. 9: Figure 3A in the paper.

The importance of the second demand, allowing weights to change sufficiently, was shown by plotting distance moved in parameter space while learning a new task, and the accuracy achieved on that task. Recall that the importance of a parameter is summed over all previous tasks, so the importance of a parameter accumulates as it is involved in more tasks.

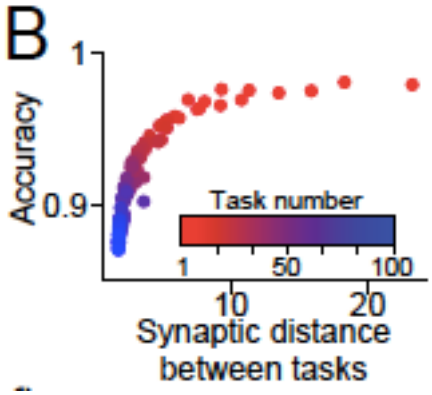


Fig. 10: Figure 3B in the paper.

So why does XdG help balance these two demands? The authors claim that XdG results in lower mean importance and "having larger number of synapses with low importance." In turn, this means that low-importance parameters can be adjusted without hurting performance on previous tasks. That is, XdG means that we can fulfill demand two at a lower cost to demand one. This claim is validated by the figure below, from data collected while learning the 100th MNIST permutation.[4]

Observations & Questions about Figures 3C-E

[4] But why are there more low-importance weights? The obvious reason is that each parameter is now involved in learning fewer tasks - so each parameter accumulates importance at a slower rate. But this is also true for split networks, and XdG is better than split networks, so there there's more work to be done in explaining why XdG is so successful...

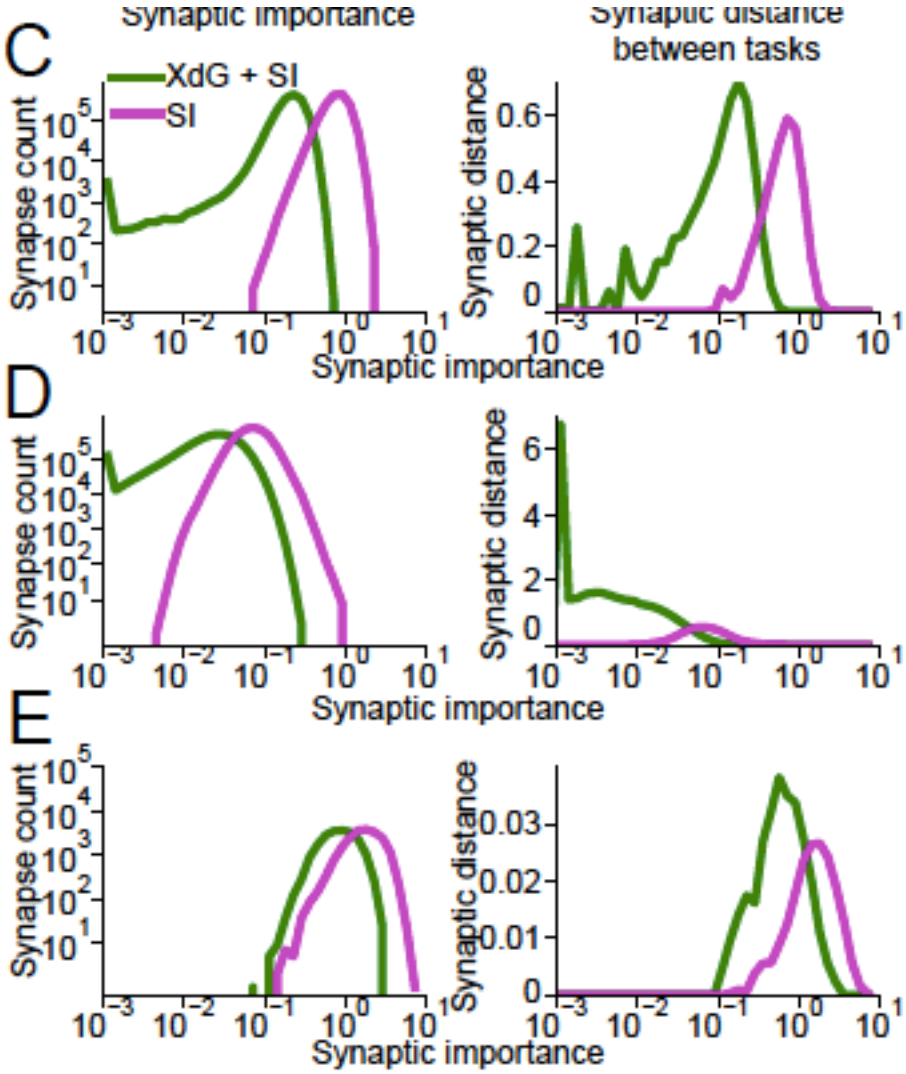


Fig. 11: Figures 3C-E in the paper. C shows the weights between the input layer and the first hidden layer, D shows the weights between the first and second hidden layers, and E shows the weights between the second hidden layer and the output layer.

- It's interesting to note the range of the synaptic distance moved plots (right panels). The weights in D move an order of magnitude more than the weights in C, which in turn moves an order of magnitude more than the weights in E. Here's a guess as to why. The weights in C are less task-specific than the weights in D because the first hidden layer should encode basic features such as edges. The weights in D are the most task-specific because they encode geometric shapes that are combinations of building blocks such as edges. The weights in E are the least task-specific because presumably its easier to change the hidden layer features than to change the combnination of features since the features are so task-specific? I'm really not sure though, this is a really kooky guess...
- The peaks are interesting to note too. In Figure C, the peaks on the left and right panels are almost identically located. I guess this is because high-intensity pixels tend to cluster near the center of images of digits. In Figure D, we see that the green peak on the right panel is very far away from the green peak. I guess this is because the network has some liberty as to which neuron encodes what features and there's redundancy amongst the features learned in the first hidden layer.
- Why does there seem to be more weights with non-zero importance when XdG is used? One guess is that the reduced number of parameters per-task means each parameter is more "needed."

3.5 Transfer Learning

The paper suggests that gating can be used to facilitate transfer learning. Instead of re-training a neural net to perform a new task, we can train a gating function to combine sub-networks in order to perform the new task.

The underlying assumption here is that there exist sub-networks which learn composable concepts. It's certainly reasonable to think that we can design networks to learn composable concepts. For instance, it's been shown that auto-encoders have structured latent-spaces. Convolutional nets seem to build concepts up compositionally. And of course, word embeddings have 'king' - 'man' + 'woman' = 'queen'. One can imagine teaching a CNN to classify zebras if it's learned to classify horses ('zebra' = 'horse' + 'stripes').

The authors identify three opportunities to make progress on transfer learning:

1. Algorithms for identifying what "network modules," or sub-networks, are useful for a new tasks (and how to combine them).
2. Algorithms for identifying the current context, and performing gating accordingly.
3. Networks that represent learned information in a modular manner.

3.6 Jumping-off Points

Some more jumping-off points:

- The measures of importance used in stabilisation seem applicable to model compression. Also, model compression between tasks seems like it might help alleviate catastrophic forgetting...
- Can we explain these *observations* about Figures 3C-E?
- The methods of parameter importance used in EWC and SI seem applicable to feature attribution. It'd be interesting to try these methods of feature attribution in combination with feature visualisation, as presented in Building Blocks Moreover, feature visualisation can help us better understand why XdG and stabilisation works. For instance, we can visually compare neurons with lower mean importance to neurons with higher mean importance.
- Can gating according to importance do better?
- Do stabilisation techniques have a relation to the replay buffer?

- And of course, applying gating to transfer learning...

CHAPTER 4

Computational Psychiatry

This page is a summary of what I've learned about computational psychiatry by reading a few papers by Quentin Huys and his collaborators. The goal is not to cover the details of the methodology, but rather to get a big picture of what computational psychiatry investigates and what the investigation process looks like.

Computational neuroscience (or sometimes theoretical neuroscience), as I understand it, is concerned with conjecturing and testing mathematical/algorithmic models of natural intelligence. Computational psychiatry uses the methodology of computational neuroscience to study how intelligent agents can go "wrong."

The methodologies employed by computational psychiatry (CP) are diagrammed below.[1]

We'll be talking about RL models, which are algorithmic models. Within computational psychiatry, RL models typically consist of two parts: **(1)** an RL algorithm capturing internal learning and evaluation and **(2)** a link function which links internal evaluation to actions. RL models in computational psychiatry also typically have a small number of parameters that are neuroscientifically meaningful. These parameters are typically fit to experimental data. Hypotheses regarding these parameters can then be tested - for example, one can test whether dopamine antagonists reduce learning rate.

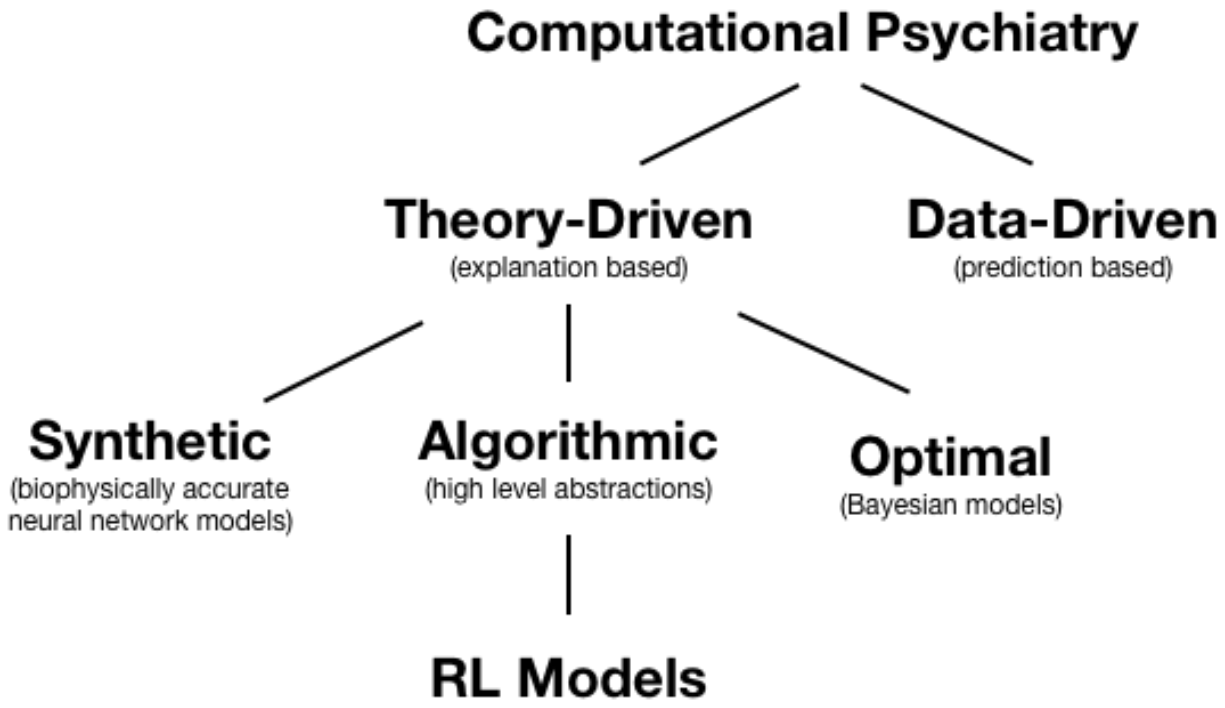
In particular, we'll be talking about the paper *Mapping anhedonia onto reinforcement learning: a behavioral meta-analysis*. [2] I chose this paper because it seems to encapsulate the entire process: conjecturing models, fitting models, model selection, investigating the parameters of the model, and then relating it all back to neuroscience.

It's been experimentally observed that for people subjectively reporting anhedonia, their behaviour is less effected by reward feedback. But we're not sure why this is - is it because of **(1)** anhedonia, i.e. a reduced internal enjoyment of rewards, or **(2)** reduced dopamine levels, i.e. a reduced ability to learn from reward feedback. Note that in the paper, reason **(1)** is called a reduction in "primary sensitivity" to rewards.

The difference between primary sensitivity to rewards and ability to learn from rewards is made clear when we consider what they mean in a RL problem. Assume that we want to learn a Q-function $Q(s, a)$ (which equals expected future reward given state s and action a). One way to learn such a Q-function is as follows:

$$\delta_t(s, a) = \rho r_t(s, a) - Q_t(s, a)$$
$$Q_{t+1}(s, a) = Q_t(s, a) + \epsilon \delta_t(s, a)$$

[1] Computational psychiatry as a bridge from neuroscience to clinical applications



In the above equations:

- ρ corresponds to primary sensitivity to reward
- $\epsilon \in [0, 1]$ corresponds to learning rate, i.e. ability to learn from reward. This is physiologically realised as dopamine (DA) signalling; it's been found that $DA \propto \delta_t$. I.e. the effect of dopamine is the control learning rate ϵ.

The question asked by the paper is if we can use the above Q-learning algorithm to figure out if the reduced behavioural effect of reward feedback in people reporting anhedonia is due to **(1)** reduced primary sensitivity or **(2)** a reduced ability to learn.

The experimental data was collected from a binary classification task given to humans from several groups (e.g. suffers from major depressive disorder, administered dopamine antagonist, etc.) The task was to classify a cartoon face as having a "short" or "long" mouth (the face is only shown for 600 ms total, and the mouth is shown for 100 ms). The twist is that the reward schedule is lopsided. Half of the participants are rewarded 75% of the time for correctly classifying long mouths and only 30% of the time for correctly classifying short mouths (vice versa for the other half of participants). The idea is that, in the face of uncertainty, participants should learn a bias for one of the actions.

The model used consists of the Q-learning model described above (the RL algo part) and several candidate link functions. All the candidate link functions take the following form:

$$p(a_t|s_t) = \frac{1}{1 + exp[-(W_t(a_t, s_t) - W_t(a'_t, s_t))]}$$
$$W_t(a_t, s_t) = \gamma I(a_t, s_t) + \xi Q_t(a_t, s_t) + (1 - \xi) Q_t(a_t, s'_t)$$

In the above equations:

- $p(a_t|s_t)$ is modelled as a logistic function where the score is based on the relative scores of action a_t and binary alternative a'_t.
- $W_t(a_t, s_t)$ is called "weights" in the paper, but I think a better label is simply score function. The higher the value of the function, the better the action being evaluated given current state s_t.

- $I(a_t, s_t)$ is called the "instruction" in the paper, it is simply an identity function on if a_t is the correct action given state s_t (evaluates to 1 if correct, 0 if wrong).
- γ corresponds to how much a subject cares about being right
- $\xi Q_t(a_t, s_t) + (1-\xi) Q_t(a_t, s'_t)$ corresponds to the expected value of the expected reward - we take an expectation since we're uncertain if the observed current state s_t is truly the current state. Hence ξ corresponds to the probability confidence we assign to our observation of being in state s_t.

Using the above framework, we can specify three candidate link functions:

- In the "Belief" model, ξ is fitted from the data.
- In the "Stimulus-Action" model, $\xi = 1$, i.e. participants are assumed to be certain in their state observations.
- In the "Action" model, $\xi = 0.5$, i.e. participants are totally uncertain about state observations.

In all the above candidate link functions, γ is fit from data. The full set of fitted parameters are $\{\gamma, \rho, \epsilon, Q_0\}$ and additionally ξ for the "Belief" model.

The fitting methodology used is expectation maximisation (EM) I remember sitting in my ML class being very surprised when I learned that EM works - a note on EM is (hopefully) coming.

Model selection was done on the group level. We want to know which model best describes a group of participants (e.g. control group, anhedonia group, etc.) It was found that the "Belief" model best describes all groups studied. The model comparison technique used was Bayesian Model Comparison - a technique that balances fit and complexity by computing posterior proabilities for models. Unfortunately, I don't know much more. The paper contains more details. This may be a topic for a future note.

The findings are that there is significant negative correlation between primary sensitivity ρ and measures of anhedonic depression (AD), and no significant correlation between learning rate ϵ and AD. Additionally, it was found that administering a dopamine antagonist had the opposite effect - it reduced ϵ but did nothing much to ρ.

This means that anhedona affects decisionmaking primarily via reducing primary sensitivity ρ and not via learning rate ϵ (related to dopamine). An obvious clinical application seems to be to decide against administering dopaminegernic treatments for depression. I'll need to think about slash read into other clinical applications of this conclusion...

Some lingering questions, (which I suspect I can answer by delving into the methodology and experimental results):

- It seems that we can fit ρ via behavior due to the presence of I in the score function W. But what justifies the presence of I? Why wasn't a model considered with $\gamma = 0$ - i.e. ignoring the I term. Also, can't rho be more easily fitted if there was a no-action option with a fixed cost to taking action? Maybe the fixed cost would also be dulled...

If you found that interesting, other cool papers are A Formal Valuation Framework for Emotions and Their Control and The role of learning-related dopamine signals in addiction vulnerability. Also read the summarised papers for more detail.

CHAPTER 5

Deep Deterministic Policy Gradient

Pseudo-code from Spinning Up, and accompanying comments for future review. DDPG is an algorithm, specifically adapted for continous action spaces, for learning a Q-function and a corresponding deterministic policy.

Fig. 1: (Deep Deterministic Policy Gradient, Spinning Up, Open AI)

- **Idea:** We interleave learning a Q-function with learning a policy to "boostrap" us off the ground.
- **Line 2:** Target parameters are essentially copies of previous parameters. They're kept to prevent potential instability in the gradient update on line 13. If the "target" had the same parameters, a possible failure mode might be that we fit the "target" to our Q-function rather than the other way around.
- **Line 4:** Noise is added the policy at train-time to maintain exploration
- **Line 7:** We maintain a replay buffer in order to stop the Q-function from forgetting how to judge "bad" off-policy states and actions - forgetting such "bad" states can cause the policy to repeat "bad" actions.
- **Line 14:** This is the line that makes DDPG specifically adapted for continous action spaces. The idea is that we want to solve $a^*(s) = argmax_a Q(s,a)$ via gradient ascent since brute force search is clearly intractable for continous action spaces. But instead of doing gradient ascent everytime we want to perform an action, we train a model according to the objective $\theta^* = max_\theta Q(s,a)$.
- **Line 15:** In practice, ρ is often very close to 0

CHAPTER 6

Spinning Up Exercises

Documenting solutions as I work through Open AI's Spinning Up exercises. I won't be writing down every detail of every solution, but rather what I see as the key insights to each exercise. I couldn't do some of the exercises without checking the solutions.

6.1 P-Set 1: Basics of Implementation

Exercise 1.1: Gaussian Log-Likelihood

- Key is to realise that a diagonal multivariate Gaussian is just a univariate Gaussian along each component
- Since each component is independent, we can treat the probability of an n-dim obervation as the product of the probability of each component
- Calculate component-wise before summing to potentially make use of parallelism

Exercise 1.2: Policy for PPO

- Just need to get familiar with TF API
- MLP solves regression problem of input state to mean action of Gaussian

6.2 P-Set 2: Algorithm Failure Modes

Exercise 2.1: Value Function Fitting in TRPO

- TRPO is an advantage-based function (advantage is how much better off you are doing action a given state s compared to the average over all possible actions a')

- Therefore it makes sense that training your value-function would help TRPO do well

---

Exercise 2.2: Silent Bug in DDPG

- It is a tensor shape error
- Should squeeze scalar-output MLPs so that they have shape [batch_size] instead of [batch_size, 1] since we'll later be performing ops with shape [batch_size]
- Weird things happen when you trying adding/multiplying tensors of shape [a] with shape [a,1] (not what we want here)

CHAPTER 7

On Algorithmic Design

Here are my thoughts on designing things via designing algorithms, rather than directly designing the thing itself. I'm especially excited about designing optimisation processes. This is also an experiment in writing publicly. Rather than sharing a "complete" series of thoughts, I'm sharing an ongoing process of thinking about algorithmic design.

7.1 Algorithms, not Pencils

Art and design projects involving GANs have been pretty hot recently. And while I find them cool, I tend to find them less interesting than other computational projects such as Evolving Floorplans and Hyphae Lamps.

I think I finally understand why. GANs, and other generative models from machine learning, model distributions. In the case of design projects involving them, they model the distribution of existing human designs. We can then reach novel designs by interpolating on the learned manifold of existing designs.

The projects I find more interesting say screw it, we don't care about existing human designs, which is really exciting for two reasons.

It's exciting because you're not constraining yourself to the manifold of existing human thought.

It's exciting because you're not constraining yourself to old design mediums. More specfically, the projects I find cool design via algorithms rather than pencils – and this step up the ladder of abstraction means you can achieve previously unmanagable levels of complexity.

Designing with algorithms give rise to many opportunities for creativity. How do you parameterise your design? How do you translate parameters (genotype) to an actual object (phenotype)? If you're optimising your design for some definition of good, how do you measure your definition of good? How do you define good?

8/16/19

7.2 Breeding

Say you want to design via optimisation. You define a notion of good, and (let's assume) it's magically achieved. How do you define good?

Defining good is hard. Try it. What, for instance, is a good house? How do you define beauty?

We may not be able to define beauty, but we can feel it when we see something beautiful. This is the rationale behind breeding. We may not be able to define good, but given several things, we can choose which things are better than others.

The simplified breeding algorithm is simple. At each point in design space, you sample several directions, and choose the most promising ones. You move in a promising direction, and sample again. If you do this over and over again, you should end up with a point in design space that is better according to your implicit notion of good. If at every point on a hill, you walk up, you're going to end up higher than where you started (a physical analogy stolen from gradient ascent).

Breeding is an optimisation algorithm that doesn't require an explicit objective. It only requires that you know good when you see it. It works. Dogs aren't super cute by accident.

I wrote this in reflection of Ganbreeder and Joel Simon's talk on Ganbreeder and other things. By the way, I still think it's useful to try to explicitly define good, but that's a topic for another time.

8/16/19

CHAPTER 8

About

Hi :) I like having my notes and thoughts in one online place. Hopefully you also find them useful and/or interesting!

You can email me at rheza@uchicago.edu

CHAPTER 9

Projects

9.1 Do grammar vectors exist in word embedding spcae? (A Visual Exploration)

[Interactive Vis] [Writeup] [Code]

9.2 Visual Abstractions of Art

[Results] [Colab Notebook]

Basically neural style transfer without the content loss.

9.3 Reaction-Diffusion Animations

[Karl Sims' tutorial] [My Colab Notebook]

CHAPTER 10

Quotes

Here live some quotes that I want to periodically remind myself of. I don't "agree" with each quote per se, but each quote does prompt something to think about. I try to distill each quote down to their essence, but punch lines often ring empty without context. If a punch line tickles you, it's probably worth looking up the context.

A Zen koan

> Before enlightenment, chop wood, carry water. After enlightenment, chop wood, carry water.

Frank O'Hara, 'Having a Coke With You'

> . . . and what good does all the research of the Impressionists do them when they never got the right person to stand near the tree when the sun sank or for that matter Marino Marini when he didn't pick the rider as carefully as the horse it seems they were all cheated of some marvelous experience which is not going to go wasted on me which is why I am telling you about it

Richard Feynman, 'Feynman's Rainbow'

> I'm not going to psychoanalyse myself. Sometimes it is good to know yourself, but sometimes it isn't. When you laugh at a joke, if you think about why laughed, you might realise that, after all, it wasn't funny, it was silly, so you stop laughing. You shouldn't think about it. My rule is, when you are unhappy, think about it. But when you're happy, don't. Why spoil it? You're probably happy for some ridiculous reason, and you'd just spoil to know it.

'Lady Bird'

> SISTER SARAH JOAN: You clearly love Sacramento.
>
> CHRISTINE 'LADY BIRD' MCPHERSON: I do?
>
> SSJ: You write about Sacramento so affectionately and with such care.
>
> LB: I was just describing it.
>
> SSJ: Well it comes across as love.
>
> LB: Sure, I guess I pay attention.
>
> SSJ: Don't you think maybe they are the same thing? Love and attention?

Kishimi and Koga, 'The Courage to be Disliked' (p. 39, 252, 254, 262)

PHILOSOPHER: That you, living in the here and now, are the one who determines your own life.

. . .

PHILOSOPHER: Imagine that you are standing on a theatre stage. If the house lights are on, you'll probably be able to see all the way to the back of the hall. But if you're under a bright spotlight, you won't be able to make out even the front row. That's exactly how it is without lives. It's because we cast a dim light on our entire lives that we are able to see the past and the future. Or at least we imagine we can. But if one is shining a bright light on here and now, one cannot see the past or the future anymore.

. . .

YOUTH: But that's just living for the moment. Or worse, a vicious hedonism!

PHILOSOPHER: No. To shine a spotlight on here and now is to go about doing what one can do now, earnestly and conscientiously.

. . .

YOUTH: If I change, the world will change. No one else will change the world for me. . .

Simone De Beauvoir, 'The Second Sex' (p. 17)

Woman's drama lies in this conflict between the fundamental claim of every subject, which always posits itself as essential, and the demands of a situation that constitutes her as inessential.

Mike LeFevre, interviewed in 'Working' by Studs Terkel (Preface 1)

It's hard to take pride in a bridge you're never gonna cross, in a door you're never gonna open. You're mass-producing things and you never see the end result of it.

Immanuel Kant, 'Grounding for the Metaphysics of Morals' (p. 36)

Act in such a way that you treat humanity, whether in your own person or in the person of another, always at the same time as an end and never simply as a means.

Kishimi and Koga, 'The Courage To Be Disliked' (p. 146, 150)

PHILOSOPHER: It's about community feeling, after all. Concretely speaking, it's making the switch from attachment to self (self-interest) to concern for others (social-interest) and gaining a sense of community feeling. Three things are needed at this point: "self-acceptance", "confidence in others", and "contribution to others."

David Foster Wallace, 'This is Water'

The freedom to be lords of our own tiny skull-sized kingdoms, alone at the center of all creation. This kind of freedom has much to recommend it. But of course there are all different kinds of freedom, and the kind that is most precious you will not hear much talked about in the great outside world of winning and achieving and displaying. The really important kind of freedom involves attention, and awareness, and discipline, and effort, and being able truly to care about other people and to sacrifice for them, over and over, in myriad petty little unsexy ways, every day. That is real freedom. The alternative is unconsciousness, the default-setting, the 'rat race'—the constant gnawing sense of having had and lost some infinite thing.

David Foster Wallace, 'Infinite Jest' (p. 860), via 'The Gosepel According to David Foster Wallace'

No one single instant of it was unendurable. Here was a second right here: he endured it. What was undealable-with was the thought of all the instants all lined up and stretching ahead, glittering. . . He could just hunker down in the space between each heartbeat and make each heartbeat a wall and live in there. Not let his head look over. What's unendurable is what his own head could make of it all. What his head could report to him, looking over and ahead and reporting. But he could choose not to listen; he could treat his head like G. Day or R. Lenz: clueless noise. He hadn't quite gotten this before now, how it wasn't just the matter of riding out the cravings for a Substance: everything unendurable was in the head,

was the head not Abiding in the Present but hopping the wall and doing a recon and then returning with unendurable news you then somehow believed.

Adam Miller, 'The Gospel According to David Foster Wallace' (Ch. 7)

Achievement can be more dangerous than failure. At least failure leaves you with the fantasy of some uppercase Substance. But imagine what happens when "you attain the goal and realize the shocking realization that attaining the goal does not complete or redeem you, does not make everything for your life OK as you are, in the culture, educated to assume" (IJ 680).

G.K. Chesterton, 'The Common Man'

The phrase would probably be misunderstood; but I should begin my sermon by telling people not to enjoy themselves. I should tell them to enjoy dances and theatres and joy-rides and champagne and oysters; to enjoy jazz and cocktails and night-clubs if they can enjoy nothing better; to enjoy bigamy and burglary and any crime in the calendar, in preference to this other alternative; but never to learn to enjoy themselves. Human beings are happy so long as they retain the receptive power and the power of reaction in surprise and gratitude to something outside…

The moment the self within is consciously felt as something superior to any of the gifts that can be brought to it, or any of the adventures that it may enjoy, there has appeared a sort of self-devouring fastidiousness and a disenchantment in advance, which fulfils all the Tartarean emblems of thirst and of despair…

Viktor Frankl, 'Man's Search For Meaning'

By declaring that man is responsible and must actualise the potential meaning of his life, I wish to stress that the true meaning of life is to be discovered in the world rather within man or his own psyche, as though it were a closed system… The more one forgets himself — by giving himself a cause to serve or another person to love — the more human he is and the more he actualises himself.

C.S. Lewis, 'Mere Christianity'

I wish I had got a bit further with humility myself: if I had, I could probably tell you more about the relief, the comfort, of taking the fancy-dress off—getting rid of the false self, with all its 'Look at me' and 'Aren't I a good boy?' and all its posing and posturing. To get even near it, even for a moment, is like a drink of cold water to a man in a desert.

Paul Graham, 'Life is Short'

Relentlessly prune bullshit, don't wait to do things that matter, and savor the time you have. That's what you do when life is short.

'The Serenity Prayer'

God grant me the serenity to accept the things I cannot change,

courage to change the things I can;

and wisdom to know the difference.

Michael Lewis, 'The Undoing Project'

"Amos [Tversky] thought people paid an enormous price to avoid mild embarrassment," said his friend Avishai Margalit, "and he himself decided very early on it was not worth it." What all those who came to know Amos eventually realised was that the man had a preternatural gift for doing only precisely what he wanted to do.

Kishimi and Koga, 'The Courage to be Disliked' (p. 146, 150)

YOUTH: Are you free, now?

PHILOSOPHER: Yes. I am free.

YOUTH: You do not want to be disliked, but you don't mind if you are?

> PHILOSOPHER: Yes, that's right. Not wanting to be disliked is probably my task, but whether or not so-and-so dislikes [or likes] me is the other person's task. Even if there is a person who doesn't think well of me, I cannot intervene in that. To borrow from the proverb I mentioned earlier, naturally one would make the effort to lead someone to water, but whether he drinks or not is that person's task.

Eliud Kipchoge, 'Breaking 2'

> In life, the idea is to be happy. So, I believe in calm, simple, low-profile life. You live simple, you train hard, live a honest life. Then you are free.
>
> Only the disciplined ones are free in life. If you are undisciplined, you are a slave to your emotions and your passions.

Abraham Heschel

> Our goal should be to live life in radical amazement ... Get up in the morning and look at the world in a way that takes nothing for granted. Everything is phenomenal; everything is incredible; never treat life casually. To be spiritual is to be amazed.

Tadashi Tokieda, 'Collects Math and Physics Surprises'

> I don't think I've had an unusual life, but it would be regarded as unusual if you take the standard sort of life people are supposed to have in a certain type of society and try to fit me in it. It's just a matter of projection, if you see what I mean. If you project on the wrong axis, something looks very complicated. Maybe according to one projection, I have an unusual past. But I don't think so, because I was living my life day by day in my own way. I never tried to do anything weird — it just happened this way.

Tadashi Tokieda, 'Collects Math and Physics Surprises'

> Sometimes adults have a regrettable tendency to be interested only in things that are already labeled by other adults as interesting. Whereas if you come a little fresher, and a little more naive, you can look all over the place, whether it's labeled or not, and find your own surprises.
>
> And so that's what you do. You just look around. And sometimes you feel tired, or you feel dizzy, or you feel preoccupied by other things, and you cannot do this. But you're not always tired and you're not always preoccupied. In those moments, you can find lots of wonderful things.

Paul Graham, 'The Age of the Essay'

> The river's algorithm is simple. At each step, flow down. For the essayist this translates to: flow interesting. Of all the places to go next, choose the most interesting. One can't have quite as little foresight as a river. I always know generally what I want to write about. But not the specific conclusions I want to reach; from paragraph to paragraph I let the ideas take their course.

Steve Weinberg, 'Four Golden Lessons'

> When I received my undergraduate degree — about a hundred years ago — the physics literature seemed to me a vast, unexplored ocean, every part of which I had to chart before beginning any research of my own. How could I do anything without knowing everything that had already been done? Fortunately, in my first year of graduate school, I had the good luck to fall into the hands of senior physicists who insisted, over my anxious objections, that I must start doing research, and pick up what I needed to know as I went along. It was sink or swim. To my surprise, I found that this works. I managed to get a quick PhD — though when I got it I knew almost nothing about physics. But I did learn one big thing: that no one knows everything, and you don't have to.

David Deutsch, 'The Beginning of Infinity' (p. 24)

> That is a good explanation - hard to vary, because all its details play a functional role. For instance, we know — and can test independently of our experience of seasons — that surfaces tilted away from radiant heat are heated less than when they are facing it.

Neal King on Richard Feynman, 'How hard a worker was Richard Feynman?' (Quora)

> A colleague of Feynman's from Los Alamos told me that Feynman used to go through the Physical Review every month. For each article, he would first read the abstract, and then think about how the article should end. Then he would check the end of the paper to see if there were any surprises. If there were no surprises, he figured that he had nothing new to learn from the paper, and he'd go on to the next. But if the conclusions of the paper were different than he had guessed from his reading of the abstract, he would take the time to read and study the whole paper.

Freeman Dyson, 'No Ordinary Genius'

> The Feynman diagram approach to quantum electrodynamics was combining this very pictorial approach with strict adherance to quantum mechanics. And that's what made it so original. Quantum mechanics is generally regarded as a theory of waves. Feynman was able to do it by ignoring the wave aspect completely. The pictures show you just particles traveling along in straight lines. These then were translated into mathematics, but in a very simple fashion, so that once you had the geometrical picture, it was simple to go straight to the answer. And that made his methods very powerful, as compared to the conventional way of doing things, which is much more analytical.

David McAllester, 'Fundamentals of Deep Learning'

> Examples confuse me. Abstraction makes things clear. (Paraphrased)

Charles Townes, 'How the Laser Happenned'

> The late Richard Feynman, a superb physicist, said once as we talked about the laser that the way to tell a great idea is that, when people hear it, they say, 'Gee, I could have thought of that.'

Charles Townes

> It's like the beaver told the rabbit as they stared at the Hoover Dam. 'No, I didn't build it myself. But it's based on an idea of mine!'

Quote banks I've taken from:

- PhD Advice by Kevin Gimpel
- Paul Graham
- Michael Nielsen
- Bret Victor

CHAPTER 11

Indices and tables

- genindex
- modindex
- search