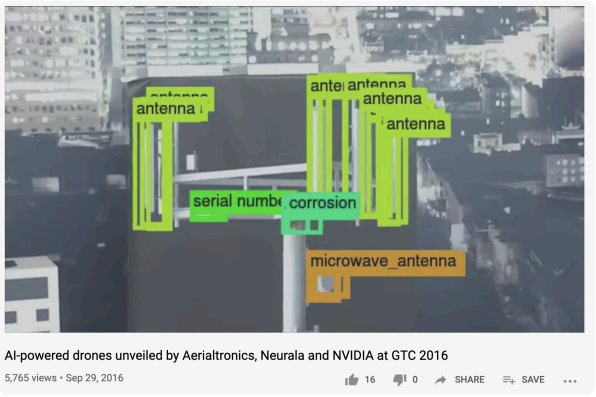
Continual Learning and Catastrophic Forgetting

by Paul Hand
Northeastern University

Outline:

- Context + initial approaches
- Evaluating algorithms
- Algorithms for CL

Example context for continval learning



AI-powered drones unveiled by Aerialtronics, Neurala and NVIDIA at GTC 2016

5,765 views • Sep 29, 2016 16 0 SHARE SAVE ...

Other example: autonomous vehicles

What are examples of situations where continual learning is desirable?

App you download that recognizes objects ... you want to teach it new objects to recognize

Toy that recognizes its owner's face

Lifelong learning setup - continual stream of experiences from which you try to learn

Correcting knowledge learned by a system

Can you simply train on new data?

Task A: $D_A = \{(x_i, y_i)\}$ Task B: $D_B = \{(x_i, y_i)\}$

First,

$$\min_\theta \sum_{x_i, y_i \in D_A} L\left(\hat{y}_\theta(x_i), y_i\right)$$ initialize randomly

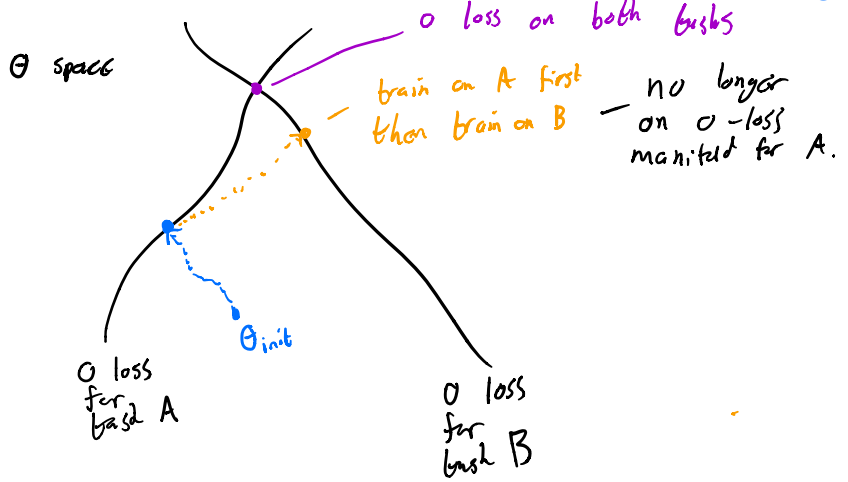
Then,

$$\min_\theta \sum_{x_i, y_i \in D_B} L\left(\hat{y}_\theta(x_i), y_i\right)$$ initialize w/ soln to above task

Failure mode: catastrophic forgetting / interference

Typically, good performance at B
worse performance at A

Visualization in parameter space of catastrophic forgetting



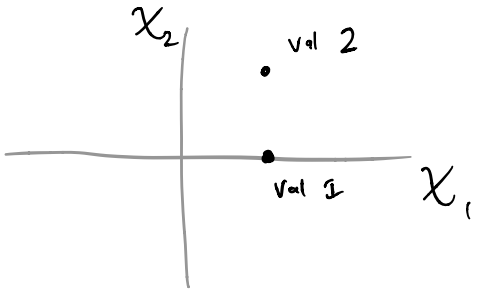
Demonstration of catastrophic forgetting using linear regression in 2-d

$D_A = \{((1,0), 1)\}$

$D_B = \{((1,1), 2)\}$

$y = (1\ 1) \cdot x$

true response



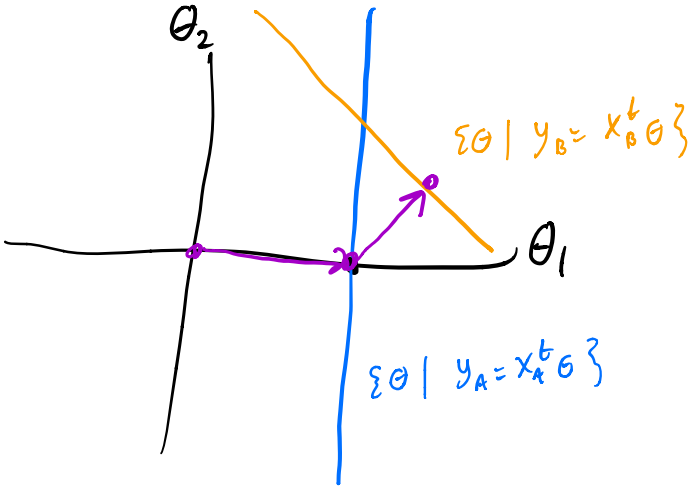
Note: This is feasible can learn A & B together

Model: $y = x^t\theta + \varepsilon$

Linear regression

$\min_\theta \| y - x^t\theta \|^2$ inialized at $\theta = \theta_0$

Soln $\theta = P_{y=x^t\theta}\,\theta_0$



How can you mitigate forgetting?

Train from scratch w/ new data and old data

Drawbacks

- Many industry nets take days – weeks to train
- Waste of power and compute
- Original training data may be unavailable

When might original training data be unavailable?

—Training data has privacy restrictions (eg through law, medical data)
— Proprietary company data
— Literally lost the data

- Need access to GPUs / cloud / steady internet

When would cloud/gpu access be an issue?

Cell phone app
Ethical /legal issues

Humans can learn incrementally, so it is possible to do

Replay old training data w/ new data

- requires storing old data
- Storage costs grow linearly w/ tasks

Dilemma: plasticity – stability

Reviews: Parisi et al. 2019
Chen and Liu 2018

Evaluating Continual Learning Algorithms

(Kemker et al. 2017)

Data permutation tasks

img, class 0-9

Task 1: Training data - MNIST (x_i, y_i)
Given x_i, predict y_i

Task 2: Fix an image permutation P_2

Training data - MNIST $(P_2 x_i, y_i)$
Given $P_2 x_i$, predict y_i

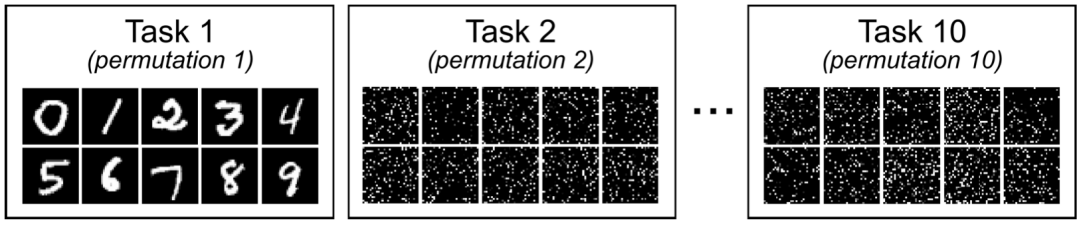


Figure 2: Schematic of permuted MNIST task protocol.

(Van de Ven and Tolias 2019)

Comments:

Each task is equally difficult

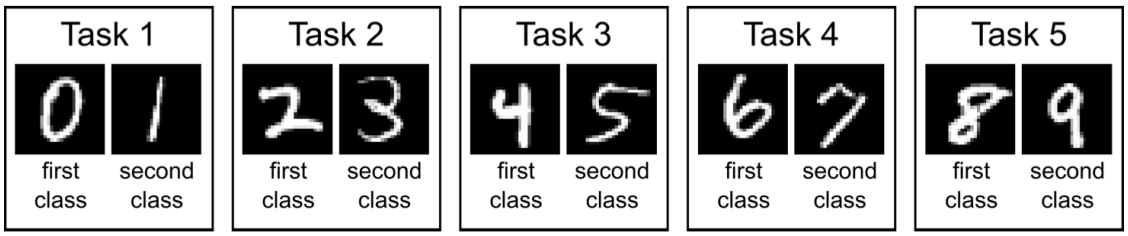
When/why are these tasks equally difficult?

If the net is a MLP, then task 1 and task 2 are equally difficulty because of the permutation invariance of an MLP

If the net is a CNN, then task 1 is easier than tasks 2,3, ...

Incremental class learning

Learn a base task set, then learn additional classes



(Van de Ven and Tolias 2019)

Shared features w/ new classes

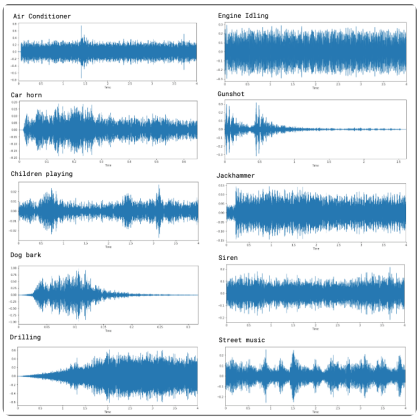
Are these tasks equally difficult?

Is task 2 harder/easier/equally difficult as task 1 for an MLP? CNN?

Don't know

Multimodal learning

Learn an image classification task
then learn audio classification



Different features must be learned

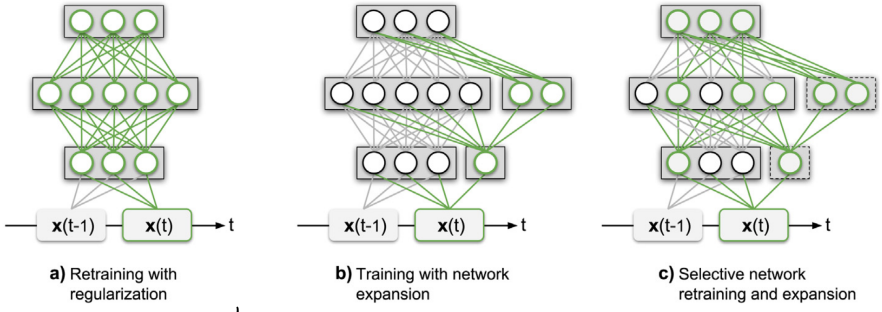
\

Approaches for continual learning

- Train whole net w/ regularization
- Dynamic architectures (add neurons)
- Complementary Learning Systems (memory + replay)

(and more)

G.I. Parisi, R. Kemker, J.L. Part et al. / Neural Networks 113 (2019) 54–71



regularization

dynamic architectures

Regularization approaches

Update network weights but penalize changes in order to minimize forgetting

(Li + Hoiem 2018)

Learning without Forgetting (LwF)

Consider predictor with shared parameters across tasks and some task specific parameters

At new task, update:

Shared params, new params, AND old params

So that

output of old task on new data doesn't change too much.

LEARNINGWITHOUTFORGETTING:

Start with:

θ_s: shared parameters

θ_o: task specific parameters for each old task

X_n, Y_n: training data and ground truth on the new task

Initialize:

$Y_o \leftarrow \text{CNN}(X_n, \theta_s, \theta_o)$ // *compute output of old tasks for new data*

$\theta_n \leftarrow \text{RANDINIT}(|\theta_n|)$ // *randomly initialize new parameters*

Train:

Define $\hat{Y}_o \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_o)$ // *old task output*

Define $\hat{Y}_n \equiv \text{CNN}(X_n, \hat{\theta}_s, \hat{\theta}_n)$ // *new task output*

$$\theta_s^*,\ \theta_o^*,\ \theta_n^* \leftarrow \underset{\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n}{\text{argmin}} \left(\lambda_o \mathcal{L}_{old}(Y_o, \hat{Y}_o) + \mathcal{L}_{new}(Y_n, \hat{Y}_n) + \mathcal{R}(\hat{\theta}_s, \hat{\theta}_o, \hat{\theta}_n) \right)$$

parameter for plasticity-stability

modified cross-entropy loss

cross-entropy loss

weight decay

Note: does not require seeing old data!

Why optimize the task specific parameters for the previous task instead of leaving them fixed?

The shared parameters get updated in response to the new task, so updating parameters for old task may be needed to accommodate those changes in shared parameters

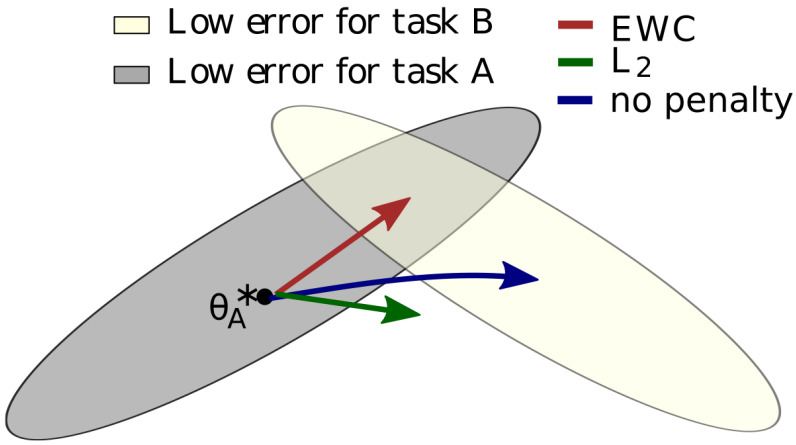
Learning without forgetting is a “regularization” based method for continual learning. Where is the regularization?

Elastic Weight Consolidation (EWC)

(Kirkpatrick et al, 2017)

When training on task B, identify weights that were important to A and penalize updates to those weights

Try to stay in low error region for A



Minimize

$$L(\theta) = L_B(\theta) + \sum_i \frac{\lambda}{2} F_i (\theta_i - \theta^*_{A,i})^2$$

parameter

Solution to task A

where

$$F_i = \mathbb{E}_x \left(\partial_{\theta_i} \log p_\theta(x) \right)^2$$

diagonal entry of Fisher information matrix of predictor at θ_A^*

How does this figure compare to the catastrophic forgetting visualization above?

Motivated by a Bayesian learning perspective

What is Bayesian learning?

$(x_i, y_i) \sim$ dist of data iid

$D = \{(x_i, y_i)\}_{i=1 \dots n}$

model: $x \mapsto$ distribution on y params θ

$\max_\theta P(D|\theta)$ vs. $\max_\theta P(\theta|D)$

MLE

frequentist

MAP

max a posteriori estimation

prior $P(\theta)$

collect D

update prior $P(\theta|D)$

Bayesian

Bayes Theorem:

$$\log p(\theta|D) = \log P(D|\theta) + \log P(\theta) - \log P(D)$$

If $P(\theta) \equiv \text{const}$ (uninformative prior) $\Rightarrow$ equivalent MLE $\leftrightarrow$ MAP

Does there exist a prior $P(\theta)$ uninformative over $\mathbb{R}^d$?

Improper priors

Example of Bayesian perspective on learning:

MLE training of a NN — MAP Estimation w/ noninformative improper prior

Weight Decay

$$\min_{\theta} \; -\log P(D|\theta) + \|\theta\|_2^2$$

Bayesian perspective: prior $\log P(\theta) = -\|\theta\|_2^2$

$$P(\theta) \sim N(0, I)$$

Derivation of Fisher Information

Dist $p(x|\theta)$ $p_\theta(x)$

How sensitive is it to changes in θ?

$\nabla_\theta \log p(x|\theta) = 0$ at a soln to training

Instead, look at $D_\theta^2 \log p(x|\theta)$

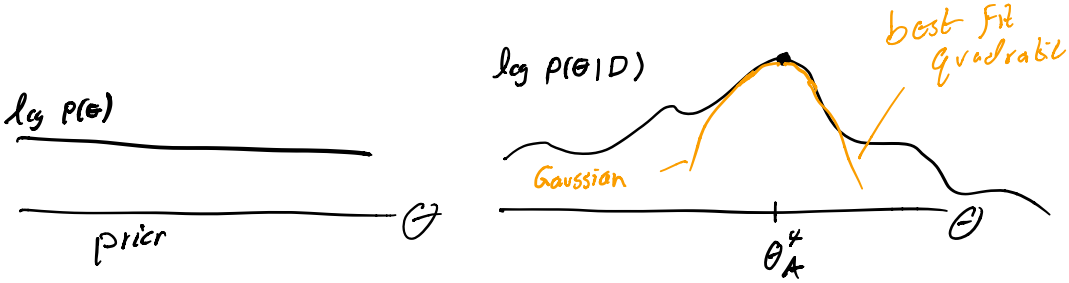
$$\mathop{\mathbb{E}}_{x\sim p_\theta} D_\theta^2 \log p(x|\theta) = \underbrace{\mathop{\mathbb{E}}_{x} \nabla \log p(x|\theta)\, \nabla \log p(x|\theta)^t}$$

Fisher Information of p_θ

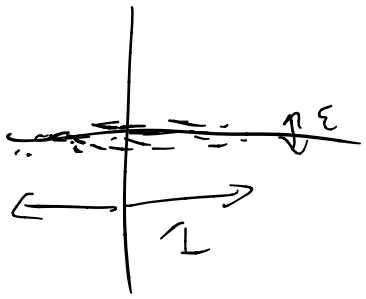
large diagonal entries $\Rightarrow$ large information content

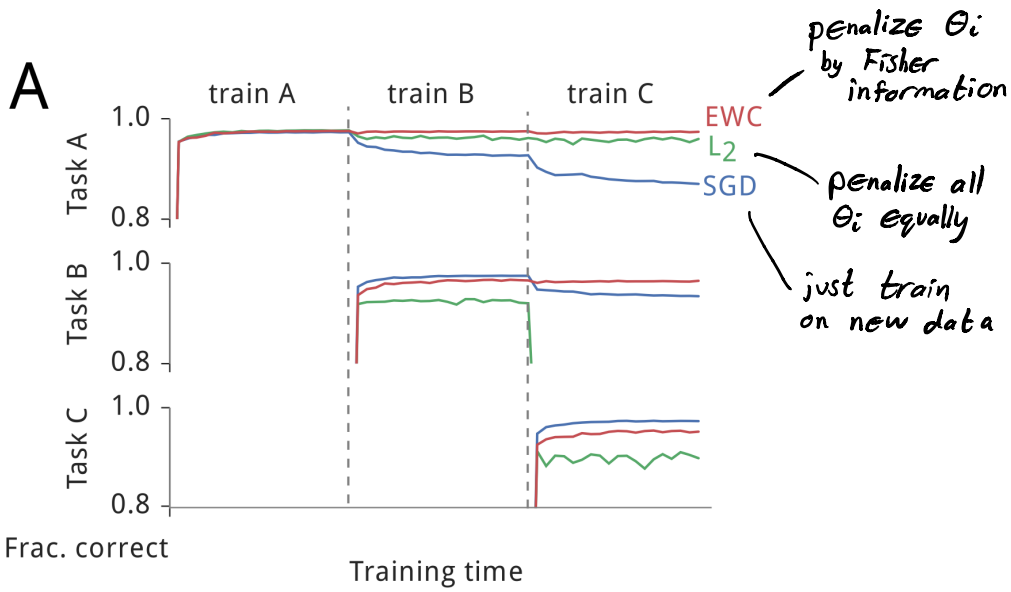
Fisher info is covariance^{-1} of $\nabla \log p$

Fisher Information Matrix provides locally Gaussian approximation to a posterior distribution



$$X \sim N\left(0, \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \end{pmatrix}^{-1}\right)$$



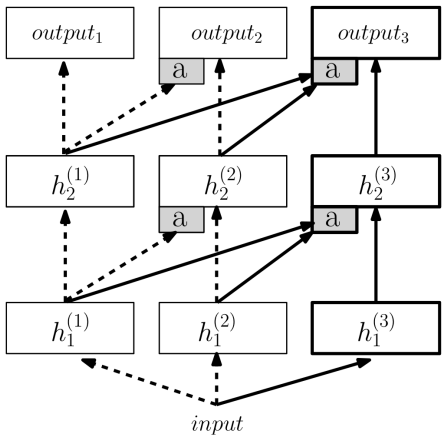


Progressive Neural Networks

(Rusu et al. 2016)

For each new task,

- add neurons
- add output layer
- add lateral connections
- dont modify old weights

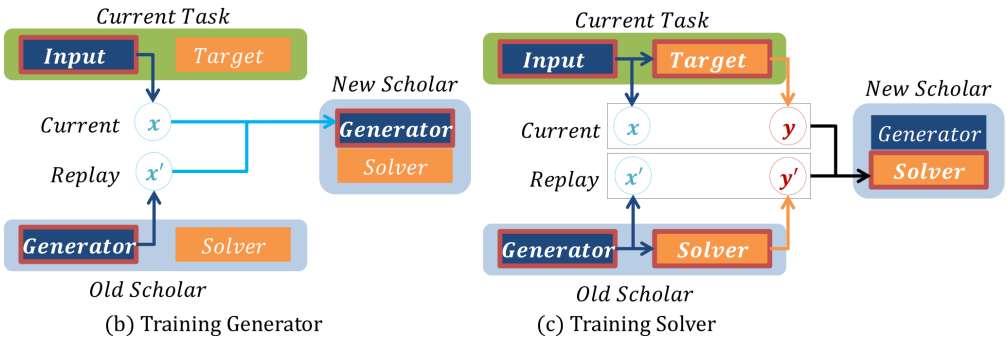


Generative Replay

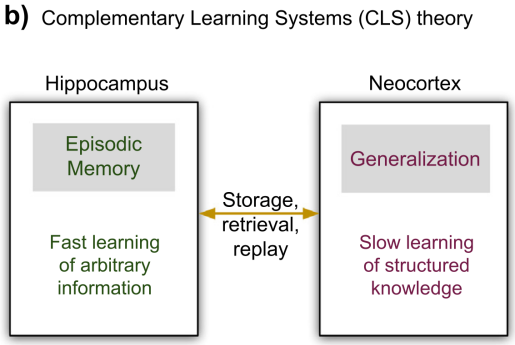
(Shin et al. 2017)

Train a generative model to output synthetic data that follows same distribution as training data.

Replay synthetic data along w/ new data



Takes inspiration from human learning



(Parisi et al. 2019)

Does generative replay avoid the data storage concerns that motivated continual learning methods?

Does generative replay avoid the data privacy concerns that motivated continual learning methods?