CS 182/282A Lecture 6:

Michelle Tong, Nikhil Potu Surya Prakash

UC Berkeley - Fall 2022

Lecture Topics

1. Deep Learning Survey
2. Deep Learning Problems
3. Optimization Overview

1 Deep Learning Survey

Conceptually these 4 sections: network architecture, problem domains, problem types, and engineering concerns. These 4 topics can be thought of as the dimensions of a 4D grid describing the field of DL.

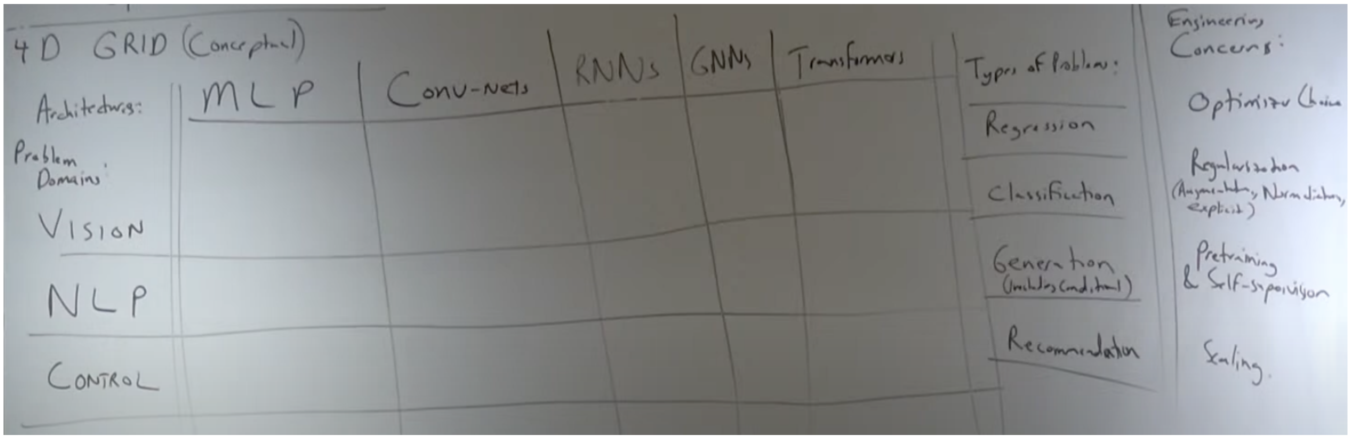


Figure 1:

1.1 Network Architectures

- Multi-Layer Perceptron (MLP)
  - network has fully connected layers
- Convolutional Neural Nets (CNNs)
  - useful for images
  - spatial regularity is embedded in the network architecture
- Recurrent Neural Nets (RNN)
  - the model architecture is different than CNNs but both architectures have a sense of internal state over time from array and weight sharing are over time
- Graph Neural Nets (GNN)

  - nearby items are more related

- Transformers
  - can access input data elsewhere and weight share

- We can tune these networks with various levels of specificity but for the scope of this class we will focus on common underlying problems that may occur.

1.2 Problem Domains

- Vision

- Natural Language Processing (NLP)

- Control
  For the scope of this class, we will explore certain domains to build intuition and experience designing networks and understand trade offs. Additionally research in this field is commonly in one of these domains so literacy is a plus.

1.3 Types of Problems

- Regression - to predict reals numbers

- Classification - to categorize

- Generation - to make/synthesize - generation as opposed to recommendation focuses on new outputs, one such example is the generation of new images of the same scene but in a different style (photo into a painting)

- Recommendation (including conditional generation) - often to commercialize and make money
  Deep learning aims to identify underlying regularities for these problems.

1.4 Engineering concerns

- Optimizer choice

- Regularization (augmentation, normalization, explicit, weight-sharing)

- Pre-training and self-supervision
  - learning models need large amounts of data to be trained well
  - Can we use external data to enhance the ML model? Yes. In theory, ML networks are able to learn regularities that are present elsewhere in large datasets. When new data is presented to the network, the model is able to focus on optimizing the nuances in the external data.

- Scaling
  - larger models (more layers, units, data) tend to work better but it needs to be trained first
  - the network is tweaked to work and also scaled to run on various components and parallel clusters
  - the network is also often scaled down to run on devices for deployment

- Experimentation
  - there are various ways to design experiments such as varying the input data or model architecture (we will cover this more in depth later in class)

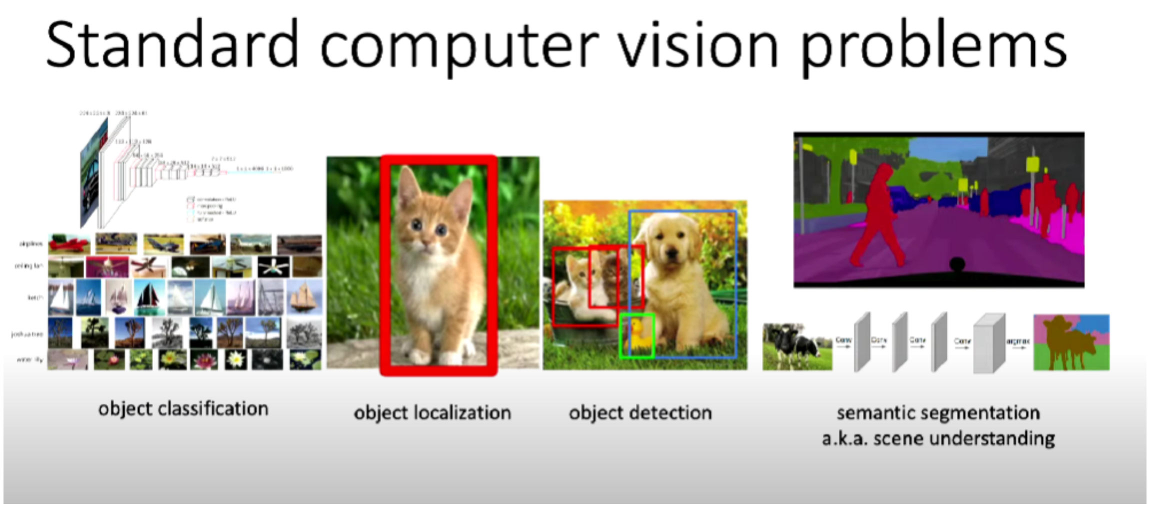
- Debugging
  - there are various ways to troubleshoot models (we will cover this more in depth later in class)

2 Deep Learning Problems

2.1 Standard Computer Vision problems

1. Object classification - What is the object? Is the image a cat or dog? This method assumes there is only one object in the image.

2. Object location - Where is the object? Where is the cat? One challenge with localization is determining how many of the object is in the image.



Standard computer vision problems

object classification

object localization

object detection

semantic segmentation
a.k.a. scene understanding

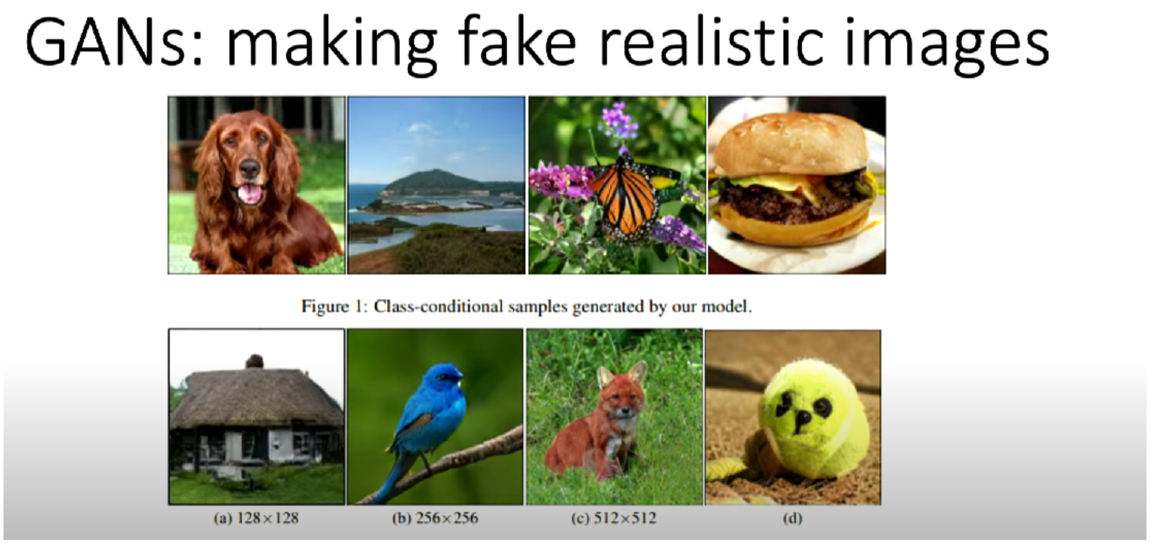
Figure 2:

3. Object detection - What and where are the objects? Where is the dog, cat, and duck in the image?

4. Semantic segmentation - scene understanding (Can we modify architecture to be better than building a classified for each pixel?)

5. Style transfer - ex. change a picture to an impressionist painting

6. GANs (Fig. 3) - making fake realistic images (Can you generate images from a class?)



GANs: making fake realistic images

Figure 1: Class-conditional samples generated by our model.

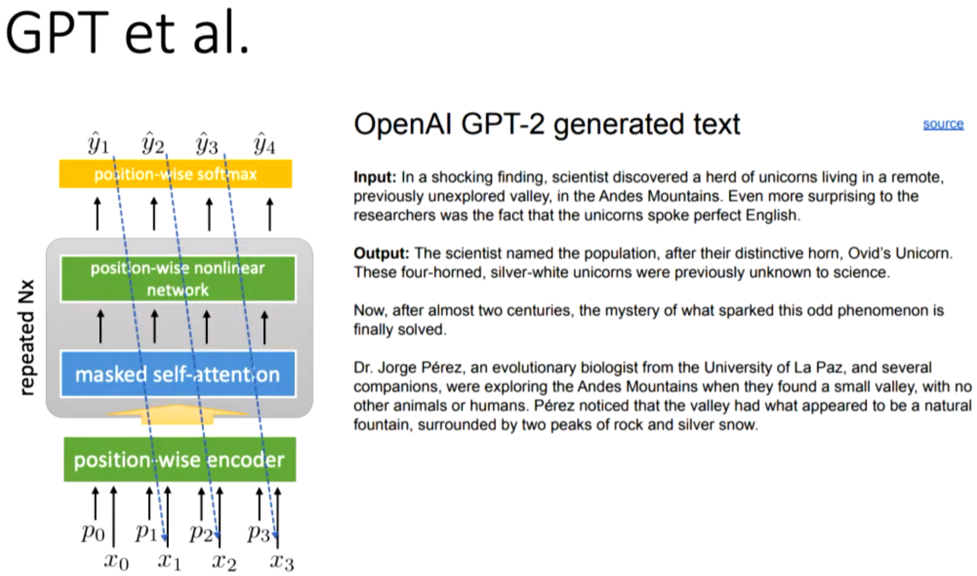
(a) 128×128 (b) 256×256 (c) 512×512 (d)

Figure 3:

7. Unpaired data testing - How does network perform with data that is not paired? Appropriately paired data generally works well.
    - Example: draw the outline of bread and tell the network to make a cat

2.2 Natural Language Processing (NLP) Problems

- OpenAI GPT-2 (Fig. 4)
    - NLP was previously rule based, but now networks can learn patterns of language
    - Example: the network learns words, grammatical types, sentence structure, flow, and pragmatics but does not necessarily learn how to reflect reality



GPT et al.

OpenAI GPT-2 generated text

source

Input: In a shocking finding, scientist discovered a herd of unicorns living in a remote, previously unexplored valley, in the Andes Mountains. Even more surprising to the researchers was the fact that the unicorns spoke perfect English.

Output: The scientist named the population, after their distinctive horn, Ovid's Unicorn. These four-horned, silver-white unicorns were previously unknown to science.

Now, after almost two centuries, the mystery of what sparked this odd phenomenon is finally solved.

Dr. Jorge Pérez, an evolutionary biologist from the University of La Paz, and several companions, were exploring the Andes Mountains when they found a small valley, with no other animals or humans. Pérez noticed that the valley had what appeared to be a natural fountain, surrounded by two peaks of rock and silver snow.

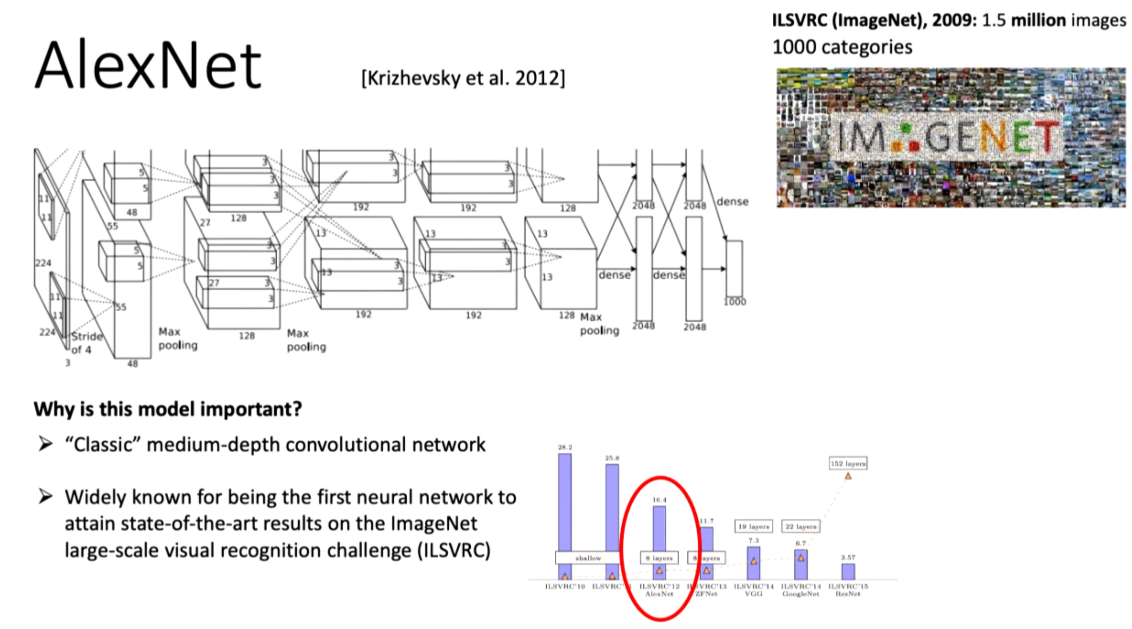
Figure 4:

2.3 Datasets

- CIFAR-10 and CIFAR-100 - dataset of images with 10 or 100 classes, 50,000 training images and 10,000 test images, labels were assigned by humans, image dimensions are 32x32x3
- imageNet - images with 1,000 classes,1.2 million training images, 50,000 evaluation images, labels were assigned by humans

2.4 Networks

- AlexNet (Fig. 5)
    - classic medium depth network
    - widely known to be the first NN to attain state of the art results on ImageNet challenge
- ResNet (Fig. 6)
    - very deep, trainable network
    - does not include a large FC layer at the end, instead just average pools over all positions and has 1 linear layer
    - network development was driven by trying to improve the optimizer



AlexNet

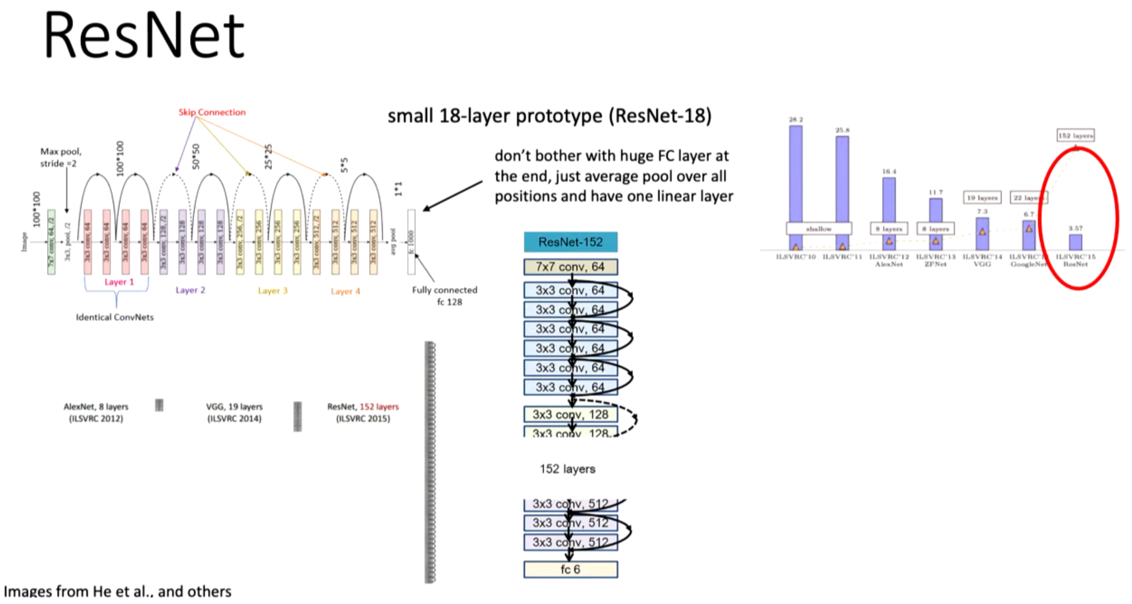
[Krizhevsky et al. 2012]

ILSVRC (ImageNet), 2009: 1.5 **million** images
1000 categories

Why is this model important?

- "Classic" medium-depth convolutional network
- Widely known for being the first neural network to attain state-of-the-art results on the ImageNet large-scale visual recognition challenge (ILSVRC)

Figure 5:



ResNet

small 18-layer prototype (ResNet-18)

don't bother with huge FC layer at the end, just average pool over all positions and have one linear layer

152 layers

Images from He et al., and others

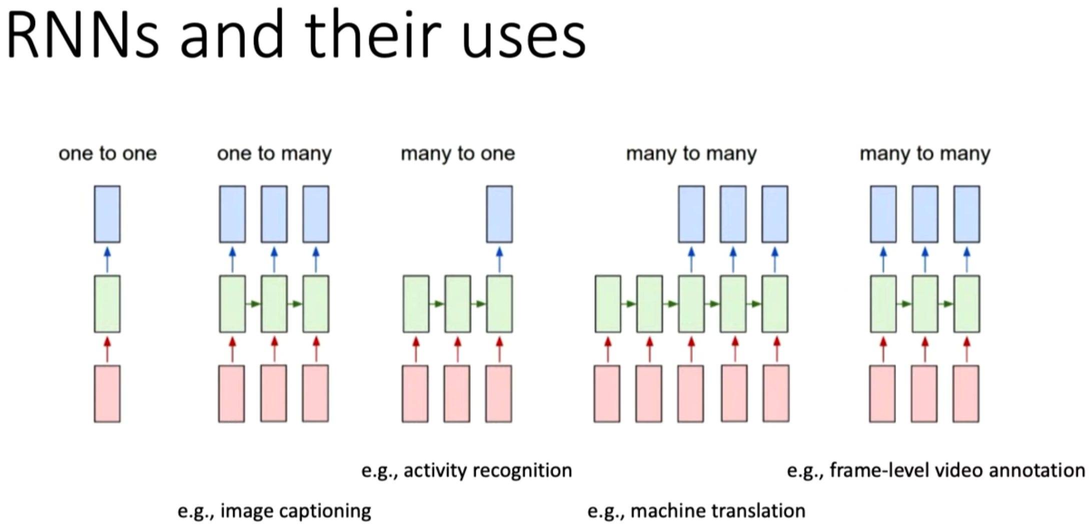
Figure 6:

  - network is leading to super-human performance which means the performance is better than that of humans doing the task, it actually achieved superhuman accuracy on some tasks

- fully convolutional networks
  - low-res (but high-depth) processing in the middle integrates context from the entire image
  - up-sampling at the end turns these low-res feature vectors into high-res per pixel predictions
- U-Net architecture
  - concatenate activations from conv layers to upsampling layers
- RNNs (Fig. 7)

- the network addresses the question, how can time oriented data, such as time series or sequential data, be digested and used for different problems

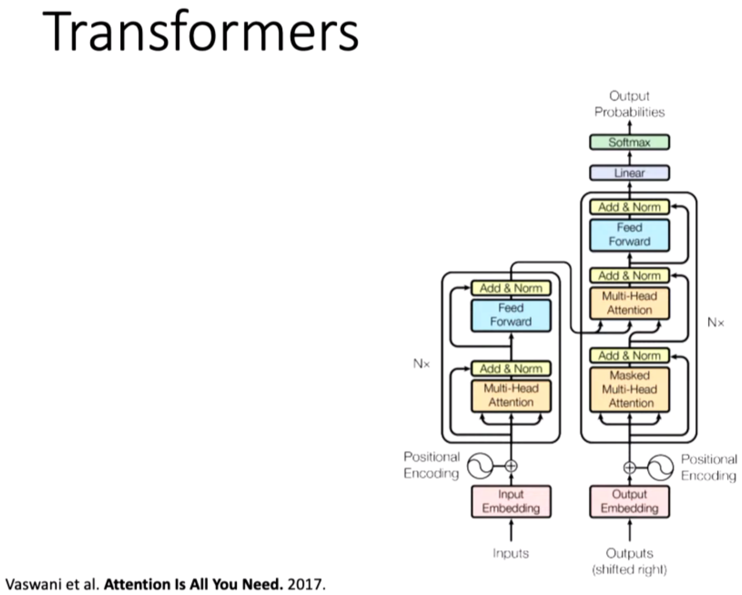
- Transformers (Fig. 8)
  - aims to solve sequence-to-sequence tasks while handling long-range dependencies with ease



RNNs and their uses

one to one
one to many
many to one
many to many
many to many
e.g., activity recognition
e.g., frame-level video annotation
e.g., image captioning
e.g., machine translation

Figure 7:



Transformers

Vaswani et al. **Attention Is All You Need.** 2017.

Figure 8:

3 Optimization Overview

In this section, a brief overview of important ideas in numerical optimization algorithms are presented. For a detailed understanding of what each of the methods does, refer to the course material of EE 227C. For ML networks, we need to solve the algorithm using an optimizer. In practice, people use optimizers that were used for similar problems before as a starting point and hyper-parameter tuning.

- Important optimization considerations
  - Learning rate - What rate are we moving down the gradient? How large is our step size?
  - Momentum based methods
  - Adaptive approaches
  - two common optimizer choices - Stochastic Gradient Descent which is mostly influenced by the learning rate hyper-parameter or ADAM optimizer which is an adaptive approach

First we want to understand optimization in terms of GD then we will understand optimization in terms of SGD. Let's begin by breaking down the learning rate in terms of a least-squares perspective.

3.1 Singular value Decomposition (SVD)

Let us recall the singular value decomposition of a matrix X. For real matrices X, its singular value decomposition can be written as

$$X = U\Sigma V^T \text{with } X \in \mathscr{R}^{m\times n}, U \in \mathscr{R}^{m\times m}, V \in \mathscr{R}^{n\times n} \text{ and } \Sigma \in \mathscr{R}^{m\times n}. \tag{1}$$

The matrices U and V are orthonormal matrices and satisfy the properties $U^TU = UU^T = I_{m\times m}$, $V^TV = VV^T = I_{n\times n}$. The matrix Σ is a collection of singular values of X along its diagonal. The singular values of X are the positive square roots of non-zero eigenvalues of XX^T or X^TX. If X has rank 'r' then there would be r singular values of X. Let the singular values of X be denoted by σ_i for $i \in \{1, \dots, r\}$. The matrix Σ can be written as

$$\Sigma = \left[\begin{array}{c|c} diag(\sigma_1 \dots \sigma_r) & 0_{r\times(n-r)} \\ \hline 0_{(m-r)\times r} & 0_{(m-r)\times(n-r)} \end{array}\right]$$

We can also see that columns of U are the extended eigenvectors of XX^T and similarly the columns of V are the extended eigenvectors of X^TX.

3.2 Least Squares

Optimization algorithms can be understood and analysed easily using simple optimization objective to which closed form minimizers exist. Least Squares problem is one such elegant optimization problem.

The least squares approximate solution of the equation $Xw = y$ can be found using the following optimization problem.

$$w^* = \underset{w}{\text{argmin}} \quad ||Xw - y||_2^2 \tag{2}$$

Utilizing the SVD of X The same optimization problem can be formulated using change of coordinates as follows

$$\begin{aligned} &\min_w \quad ||Xw - y||_2^2 \\ =&\min_w \quad ||U\Sigma V^T w - y||_2^2 \\ =&\min_w \quad ||U(\Sigma V^T w - U^T y)||_2^2 \end{aligned} \tag{3}$$

Notice that the norm of a vector doesn't change when it is just rotated without stretching. The orthonormal matrices U and V have orthonormal columns and hence just rotate the vectors without stretching them. Therefore,

$$\begin{aligned} &\min_w \quad ||U(\Sigma V^T w - U^T y)||_2^2 \\ =&\min_w \quad ||\Sigma V^T w - U^T y||_2^2 \\ =&\min_{\tilde{w}} \quad ||\Sigma \tilde{w} - \tilde{y}||_2^2 \end{aligned} \tag{4}$$

where $V^T w = \tilde{w}$ and $U^T y = \tilde{y}$ In eq.(4), since Σ has singular values only along its diagonal, the objective can be decoupled into sums of squares of multiple scalar differences as follows

$$\min_{\tilde{w}[1],\tilde{w}[2],\ldots,\tilde{w}[r]} \quad \sum_{k=1}^{r} (\sigma_k \tilde{w}[k] - \tilde{y}[k])^2 \tag{5}$$

3.3 Gradient Descent

The gradient descent update equation for the optimization problem in (2) with a learning rate η can be written as

$$w_{t+1} = w_t - 2\eta X^T(y - Xw_t) \tag{6}$$

Similarly, the gradient descent update equation for the equivalent optimization problem in (4) can be written as

$$\begin{aligned} \tilde{w}_{t+1}[i] &= \tilde{w}_t[i] - 2\eta\Sigma^T(\Sigma\tilde{w}_t[i] - \tilde{y}) \\ &= (1 - 2\eta\sigma_i^2)\tilde{w}_t[i] + 2\eta\Sigma^T\tilde{y} \end{aligned} \tag{7}$$

Notice that the update rule is just written for the i^{th} element of $\tilde{w}$. For the stability of the difference equation in (7), we need

$$\begin{aligned} 1 - 2\eta\sigma_i^2 &> -1 \; \forall \; i \\ \implies \eta &< \frac{1}{\sigma_i^2} \; \forall \; i \\ \implies \eta &< \frac{1}{\sigma_{max}^2} \end{aligned} \tag{8}$$

Here σ_{max} and σ_{min} are the largest and smallest singular values of X respectively.

It can be seen from the above choice of η, $1 - 2\eta\sigma_{min}^2$ can be close to 1 and the convergence might take an extremely long time to converge along certain directions with small corresponding singular values. One of the ideas to improve the speed of convergence is to use the concept of 'momentum' inspired from the 'Proportional + Integral (PI) action controller' which is described in the next section.

3.4 Momentum based methods

Idea: Find a way to make the learning rate bigger without causing trouble for the large singular values.
Observation: The weights associated with the large singular values oscillate at high frequency as the learning rate is increased. So, to dampen the oscillations out, a low pass filter can be added. It is known from circuit analysis that a low pass filter outputs an exponential average of the input. This averaging can help us use larger learning rates compared to gradient descent as averaging dampens out the oscillations due to large singular values.
Implementation:

$$\begin{aligned} \tilde{w}_{t+1}[i] &= \tilde{w}_t[i] - \eta a_{t+1}[i] \\ a_{t+1}[i] &= (1-\beta)a_t[i] + \beta(\text{ “current gradient”}) \end{aligned} \tag{9}$$

For more intuition about The term “current gradient”, think about how the gradient was obtained in the previous section on least squares.

Here a_t is the internal state which dictates the averaging behavior (exponential average) and β controls how fast we average i.e., controls the weight given to the past events in the exponential average. This is very much similar to the behavior of an RLC circuit. For momentum based methods, both the weight and internal state average gradient are evolving. Note, there are several ways to mathematically implement this circuit.

3.5 Adaptive approaches:

There's a limit to how much the learning rate can be increased even by the momentum based methods. Momentum still respects SVD and the movement along the directions that are small is still small. The idea in adaptive approaches is to change the learning rates for different singular values (different directions) - More about this in the next lecture.

3.6 Citation

- Sergey Levine's slides (Fig 2-8), https://static.us.edusercontent.com/files/azVHcuABtM3V2ja2DrXVVo91