COGS 185 — Advanced Machine Learning Methods

Notes taken by Nolan Chai

Spring 2023

Contents

Preface

These are a collection of notes personally taken by me, specifically for readings and allotted content for UCSD's COGS 185 Advanced Machine Learning Methods taken in Spring 2023. These notes are not endorsed by the lecturers nor staff, and I have modified them (often significantly) over random periods of time. They may become nowhere near accurate representations of what was actually lectured, or written in the books, and are simply to aid in my own understanding. In particular, all errors are almost surely mine.
Notes are taken real time, and will be reviewed, updated, and revised within 48 hours of each lecture.

My other notes are available **here**.

0 Introduction

The course site is available ***here****.*

The main thing we'll be going over in this course is **context** - the most important thing in modern machine learning. In general, when we say context - we mean **two things**: context within an instance such as yourself as a person (reasons why you decide on things, such as why you chose UCSD, or a particular subject), and context across instances (across a population - similarities with other people).

We'll proceed incrementally, beginning with a review of supervised learning into multiclass/multi-label classification, structural predictions, sequence modeling, semi-supervised and unsupervised learning, self-supervised learning, sparse coding, and reinforcement learning.

Logistics

The course is structured as a hybrid course, with lectures available both online and in-person. The grade breakdown will be as follows:

- Assignments (4 total): 50%
- Midterm: 25%
- Final Project: 25%
- Bonus Points: 3% (Piazza, Final Project)

The midterm will be **Thursday of week 4**. Attendance is also not mandatory.

1 A Review of Supervised Learning

To begin, what exactly is a *pattern*? Is it repetitive? Not really. You can refer to a plank of wood, or a wooden tabletop, that contains a 'pattern' but it may not necessarily be repetitive. However, they can contain *subjective, explicit, and implicit descriptions.* They do not necessarily have to contain common features.

Then what is not a pattern? No "pattern" is also a pattern. Unpredictability can be a pattern. In essence, everything is a pattern.

1.1 Structure

But also, what is structure, then? When we look at a small segment of images, it's very difficult to understand the underlying structure without *context.* Supervised learning is basically a massive encoder for context.

The main scheme for structures is as follows:

- Structure within a data sample *(supervised)*
- Structure between data samples *(semi-supervised)*
- Structure within a sample *(unsupervised)*

1.2 Trends in AI

We've started from logical and hard-coded artifical intelligence (traditional AI) to statistical analyses to machine learning, and now, finally artificial intelligence (modern AI/ML). There is emphasis on two things: simplicity of capability / structure, and scalability.

AutoML was a hot topic for a while until the release of ChatGPT, where the importance of searching for best parameters was trivialized. Success in artificial intelligence and machine learning revolves around the availability of large amounts of training data (e.g., ImageNet), the access of modern computing infrastructures (e.g., Nvidia GPUs), and new developments in neural networks with deep structures (e..g., AlexNet).

AlexNet was one of the most fundamental models, as it displayed the importance of neural network architectures; and now, ChatGPT is the second major development in terms of waking up the world to how important such landmarks are.

1.3 The three components of learning algorithms

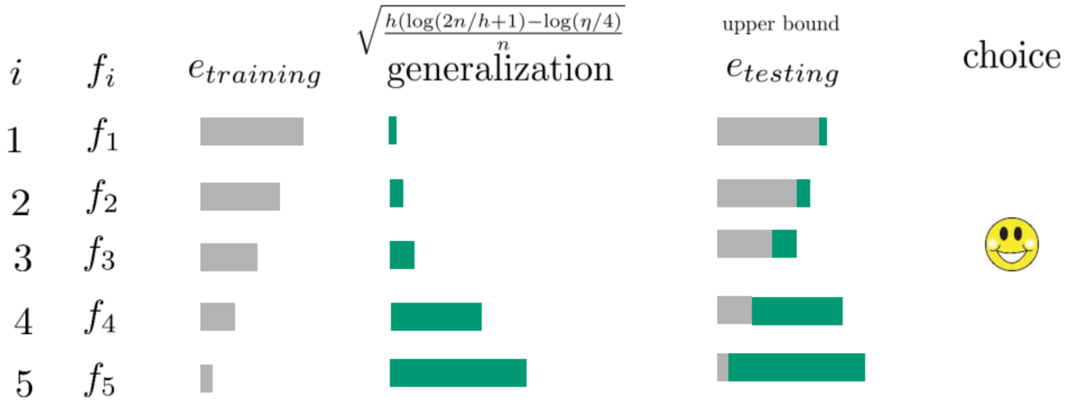
We classify the main three components of learning algorithms under representation, evaluation, and optimization. These are all typically very important, but in today's world, having good data is **far** more important than optimizing well.

Overall, Learning = Representation + Evaluation + Optimization.

1.4 Structural Risk Minimization

Let $\phi(f)$ = the set of functions representable by f.
Suppose: $\phi(f_1) \subset \phi(f_2) \subset ...\phi(f_n)$
Then: $h(f_1) \leq h(f_2) \leq ...h(f_n)$
We are trying to decide which machine to use. We train each machine and make a table based on our standard optimization for generalization:

$$e_testing \leq e_{training} + \sqrt{\frac{h(\log(2n/h + 1) - log(n/4)}{n}}$$



The more complex the model is, the smaller the training error. However, the less generalization capability it will subsequently have. Or...at least this is the case with older models.

Suppose we increase the size and model complexity of ChatGPT's parameters even further than it is currently; let's say 100x. Although this would typically result in greater training error, it somehow doesn't decrease in generalization - rather, it gets better. This is the current state of LLMs, and why there is so much hype around it - we want to push the boundaries to see how far this will go.

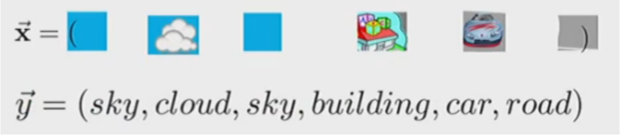
When you begin to solve a machine learning problem, you want to focus on first understanding it by formulating it between an input and output state.

1.5 Context

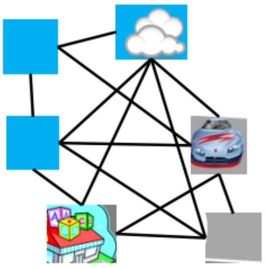
Context comes from both *within-data* (parts/components) and *between-data* (configurations). For instance, some problems may exist where, from a small section of an image, you need context to understand the full image.

To formalize the problem, we can look at the image as a whole, then divide it into patches like so:

While subsequently labeling the images, which is a problem that we deal with - structural information:


$\vec{\mathbf{x}} = ($ $)$
$\vec{y} = (sky, cloud, sky, building, car, road)$

Which we can then use to vectorize into graphs:



In today's world, we use dense graphs utilized by transformers. Before, we had primarily focused on sparse graphs due to the fear that we could not deal with dense graphs. Transformers are able to fully utilize dense graphs due to its adaptive attention mechanism and connect everything - this is also related to why transformers have such an incredible number of parameters.

This problem formulation can be represented by the following random fields:

- Markov Random Fields:

$$p(\overrightarrow{y} \mid \overrightarrow{x}) \propto p(\overrightarrow{y})p(\overrightarrow{x} \mid \overrightarrow{y})$$

$$\rightarrow \Pi_{(i,j)\in\mathcal{N}} p(y_i, y_j)\ \Pi p(x_i \mid y_i)$$

- Conditional Random Fields:

$$p(\overrightarrow{y} \mid \overrightarrow{x})$$

$$\rightarrow \Pi_{(i,j)\in\mathcal{N}} p(y_i, y_j \mid x_i, x_j)\ \Pi p(y_i, x_i)$$

2 Basics

We begin with another emphasis on **context.** In large language models and natural language processing, words are embedded via vectors in a latent space. The professor displays a video which models this in relation to the human brain, where words are mapped and grouped by *semantics* (closeness depending on words) - a *brain map*.

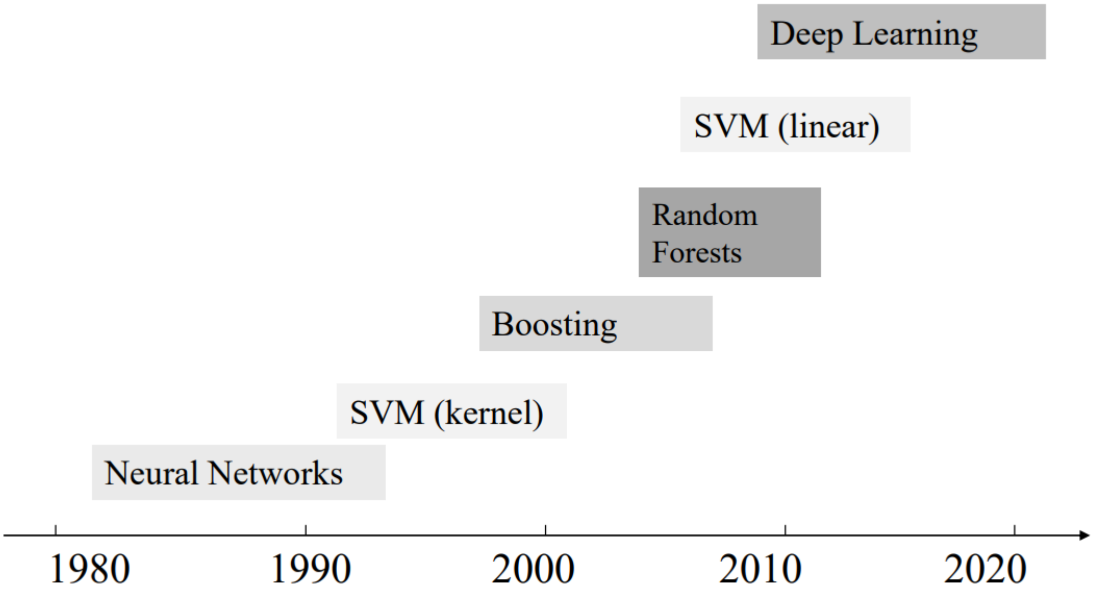
There are three **key** variables that we deal with for notation:

- Input: $x = (x_1, x_2, ...)$
- Label: $y \in -1, +1$
- Model parameter: W

2.1 A Brief History of Machine Learning

During the 1980s-1990s, there was much difficulty in categorizing neural networks statistically, which pushed SVMs and kernels by the mid 1990s - which project lower dimensional to higher dimensional spaces.

During the early 2000s, boosting - which combines multiple weak classifiers - emerged, and was very popular. This led into the popularity of random forest models by the 2010s (and is still popular today - I use this in a lot of modern research still). Ensemble learning is industrially known to always raise model performance by a few percent - you can ensemble multiple models together and subsequently reduce variance, but keep in mind that stacking ensemble models will give you diminishing returns at a point.



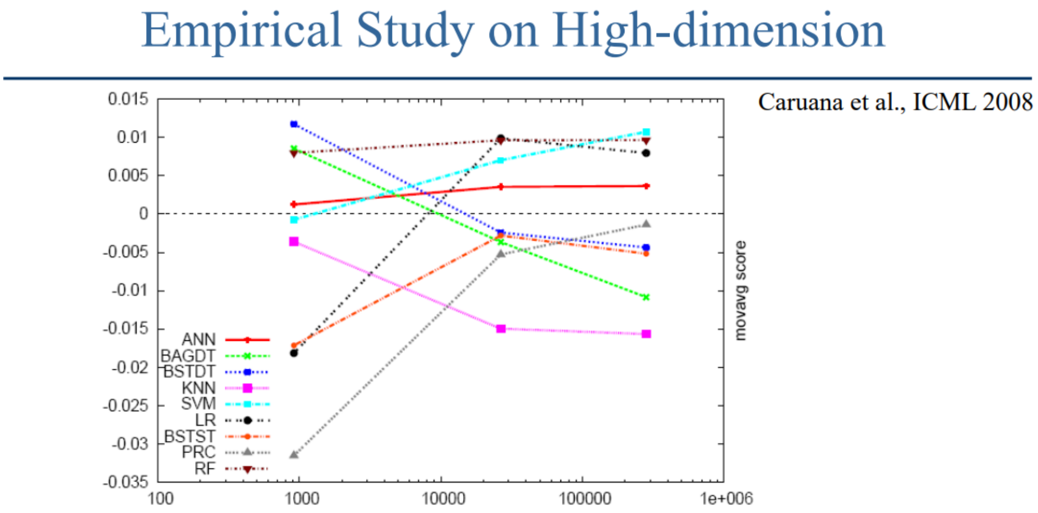
Furthermore, notice that the development of models has been historically dependent on the increase in data availability over time. Random forests' performance superseded boosting due to the fact that the randomness of training independent weak classifiers was better than simply boosting.

Once we had larger datasets, with $n > 1000$ features, the increase in dimensionality allowed linear SVMs to supersede kernel SVMs. In linear SVMs, we have a direct, explicit loss whereas kernels are implicit. Tldr; linear for higher dimensions and kernel for lower dimensions.

Interestingly enough, neural networks have become popular once again due to the abundance of data and increase in dimensionality again with convolutional neural networks and transformers.

Are deep learning models parametric or non-parametric?
Strictly following the definition of a neural network, deep learning models are parametric since parameter size is fixed and initial parameters are predefined; despite the nonparametric "flavor" of transformations done on hidden layers.

The parallelism between convolutional neural network trends to visual transformers is not the same with SVM kernels to SVM linear models. This is because the introduction of attention is much more intuitive and adaptive - it introduces a very *real* aspect of intelligence (cognitive science) - into transformers; previously, all other classifiers were similar to one another and had the same base mechanisms.



Empirical Study on High-dimension

Caruana et al., ICML 2008

As seen above, the rank of algorithm performance on high dimensional data is

1. Random Forests
2. Neural Networks
3. Boosted Trees
4. SVMs

2.2 Supervised & Unsupervised Learning

Almost everything today is linear in higher dimensions. Here, the professor goes through both supervised and unsupervised learning very quickly; I will expand this section if I ever get the chance / find time to, but this section was basically skipped as it's all basic knowledge.

2.3 Driving Factors of Machine Learning

To do well in machine learning, we need Intuition + Math/Statistics + Implementation/Coding. However, with the recent release of transformers, the importance of data quality and quantity has only been growing.

- Representation: With better and better understanding of theunderlining statistics about the data and methods.
- Evaluation: The ideal strategy is always to aim at your targetdirectly (take non-stop flight as opposed to having multiple stops).
- Optimization: Based on the chosen representation and evaluation,you pick a strategy (mathematical/statistical) to achieve your goal.
- Data: Having sufficient amount of data for learning andjustification is increasingly important.
- Computing power: In terms of both capacity and computation.

2.4 Mathematical representation for features

Given

$$S = \{(x_i), i = 1...n\}, x_i = (x_{i1}, ..., x_{im})$$

What if it was a city: $x_{i2} \in \{Los\ Angeles, San\ Diego, Irvine\}$ We cannot encode L.A. to 1, SD to 2, and Irvine to 3, as it would imply LA + SD = Irvine.

Rather, we use ***One-Hot Encoding*** in which we expand the features to N-dimensions for N number of possible states.

One-hot encoding:

	coded values
Los Angeles	1, 0, 0
San Diego	0, 1, 0
Irvine	0, 0, 1

The **pro** is that we can naturally deal with any type of input (can associate confidence directly), but the feature dimension has become much larger. Although this may seem like we increase the computational requirements, it's not much for modern architectures.

In this way, we now have access to utilize categorical values in the form of *soft values*, with a probabilistic interpretation, that is measurable and comparable. One-hot encoding gains in its convenience in a canonical mathematical representation by sacrificing in the space complexity: one category of k-classes is turned into k real numbers in [0, 1].

2.5 Error Metrics and Object Functions

Modern developments of AI/ML has allowed for the establishment of benchmarks under a widely accepted common evaluation metrics. Being able to faithfully compare the performances of different machine learning algorithms/systems significantly propel the advancement of the machine learning field as a whole.

Furthermore, establishing a clear objective function (errors + regularization) to optimize when training machine learning algorithms is a key reason for the success of modern machine/deep learning.

2.5.1 Summary of the problem

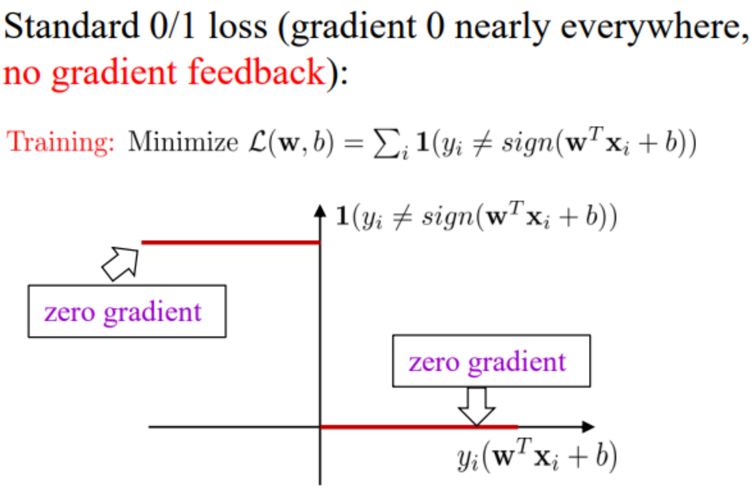
$$S_{training} = \{(x_i, y_i), i = 1, ..., n\}$$

$$x = (x_1, ..., x_m), x_i \in \mathcal{R}, x \in \mathcal{R}^m$$

(will finish editing this section later when I have time, but is essentially review of 118A/150 concepts)

2.5.2 Standard Loss (error) function

The standard error which normalizes for 1 and 0, has the most direct loss but is very difficult to decrease due to its zero gradient.



Standard 0/1 loss (gradient 0 nearly everywhere, no gradient feedback):

Training: Minimize $\mathcal{L}(\mathbf{w}, b) = \sum_i \mathbf{1}(y_i \neq sign(\mathbf{w}^T\mathbf{x}_i + b))$

$\mathbf{1}(y_i \neq sign(\mathbf{w}^T\mathbf{x}_i + b))$
zero gradient
zero gradient
$y_i(\mathbf{w}^T\mathbf{x}_i + b)$

2.5.3 Decision Boundary

Will also review this later – First example, no because it says ge/le 0, but never crosses 0

The take-home message for decision boundaries is: – – I didn't sleep last night / have a few meetings later today lmao so I will revise a lot of this later tonight

3 Support Vector Machines

4 Softmax function

5 Structured Prediction

6 Random Fields

7 Auto-Context

8 Auto-Context (Cont.)

9 Recurrent Neural Networks

10 Recurrent Neural Networks (Cont.)

11 Attention based models

12 Transformers

13 Large Language Models

14 Compressive Sensing

15 Weakly-Supervised Learning

16 Self-Supervised Learning

17 Vision Transformers

18 Generative Adversarial Networks