

10417/10617
Intermediate Deep Learning:
Fall 2023

Yuanzhi Li / Russ Salakhutdinov
Machine Learning Department

Pre-request

- The first half of 10-301 + 10-601 or similar.

Evaluation



3 Assignments, worth 70%.



Typically assignments are 2 weeks long, and have theory part + coding part



Projects, 30%:



- Midway report 5%, Final Project 25%.



Homework Dates – Check the website for updates!

Evaluation



5 late days for all assignments.



No more than 3 late days per assignment. After 3 late days, you will get 0.



3 late days for projects: can be split between project midway report and final project.



Project: Teams of 2-3 people per project.

Project



The idea of the final project is to give you some experience trying to do a piece of original research in machine learning and coherently writing up your result.



What is expected: A simple but original idea that you describe clearly, relate to existing methods, implement and test on some real-world problem.







To do this you will need to write some basic code, run it on some data, make some figures, read a few background papers, collect some references, and write an 8-page report describing your model, algorithm, and results.

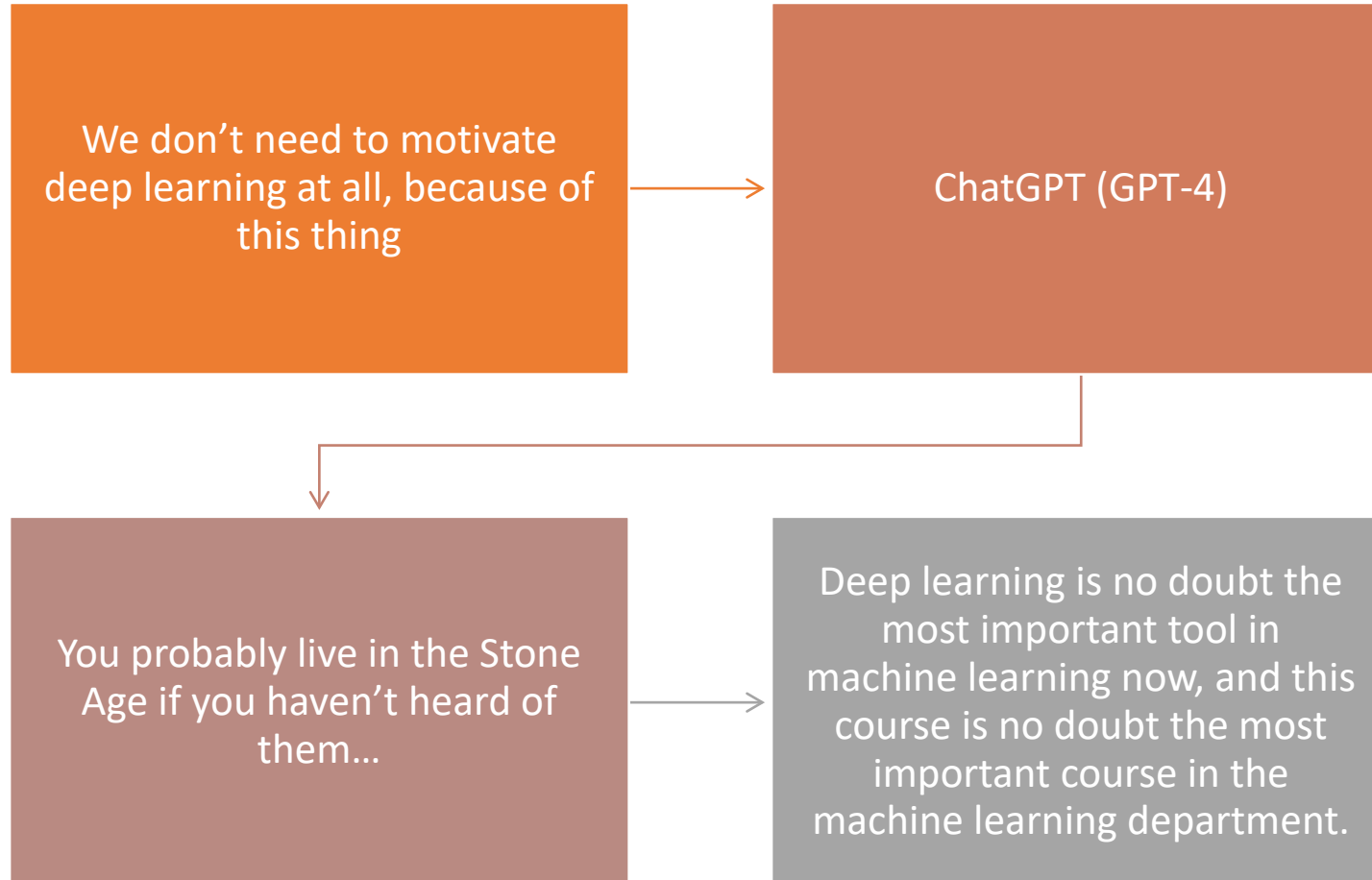
Deep Learning

- This year is special for deep learning, and learning deep learning.
- Historically, we need to spend quite a bit of time in the course motivating the importance of deep learning in practice...

Impact of Deep Learning

- Speech Recognition 
- Computer Vision 
- Recommender Systems 
- Language Understanding
- Drug Discovery and Medical Image Analysis 

This year...



Course structure



Most of the materials in this year will be similar to previous years.

Deep learning basics (from perceptron to multi-layer perceptron)
Deep learning optimization basics
Different deep-learning networks and applications



But we are going to add 5 new lectures on transformer models and artificial general intelligence (AGI).



We will learn what is the underlying architecture of GPT-4 and how these models are trained.



You will also be able to train a storytelling language model yourself, with less than 50 lines of code, based on the dataset called TinyStory.

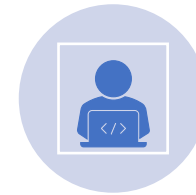
Deep Learning Essentials:



We will learn the following essential things for deep learning:



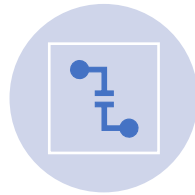
Neural network architectures.



Backpropagation algorithm.



Different training objectives: Label prediction models (supervised learning), autoregressive models (unsupervised learning), and diffusion models (distribution learning).



Attention mechanism (transformer), positional encoding etc.

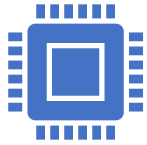


Ensemble/Distillation, Mixture of Experts, etc.



Probabilistic deep learning models.

What is special about this course



Key question: Deep learning is also taught in other courses, like 10301-10601, 10-315, 10-701, and 10-715.



So what is special about this course?



All those courses essentially skim through deep learning and treat it as a “black box”.

You will learn the mathematical definition and some empirical numbers.

You will learn how deep learning works, but not why it works.

Industries are not looking for people who can code deep learning models – This is the easy part. Industries are looking for people who actually know how to use deep learning in innovative ways.



This course: We will teach you “why” deep learning works, and using the principles to guide deep learning applications.

This course: The “Physics” of deep learning.



Intermediate deep learning = We will go beyond just memorizing the formulas for deep learning, but actually digging into the inner-workings.



The “physical principle”.

Why do I choose to use certain deep learning architecture? What is the pros and cons in practice?

How do different deep learning models react to different input data distributions/different augmentation techniques, etc.?

What features are the deep learning models actually learning? How can I debug a model?

Deep learning



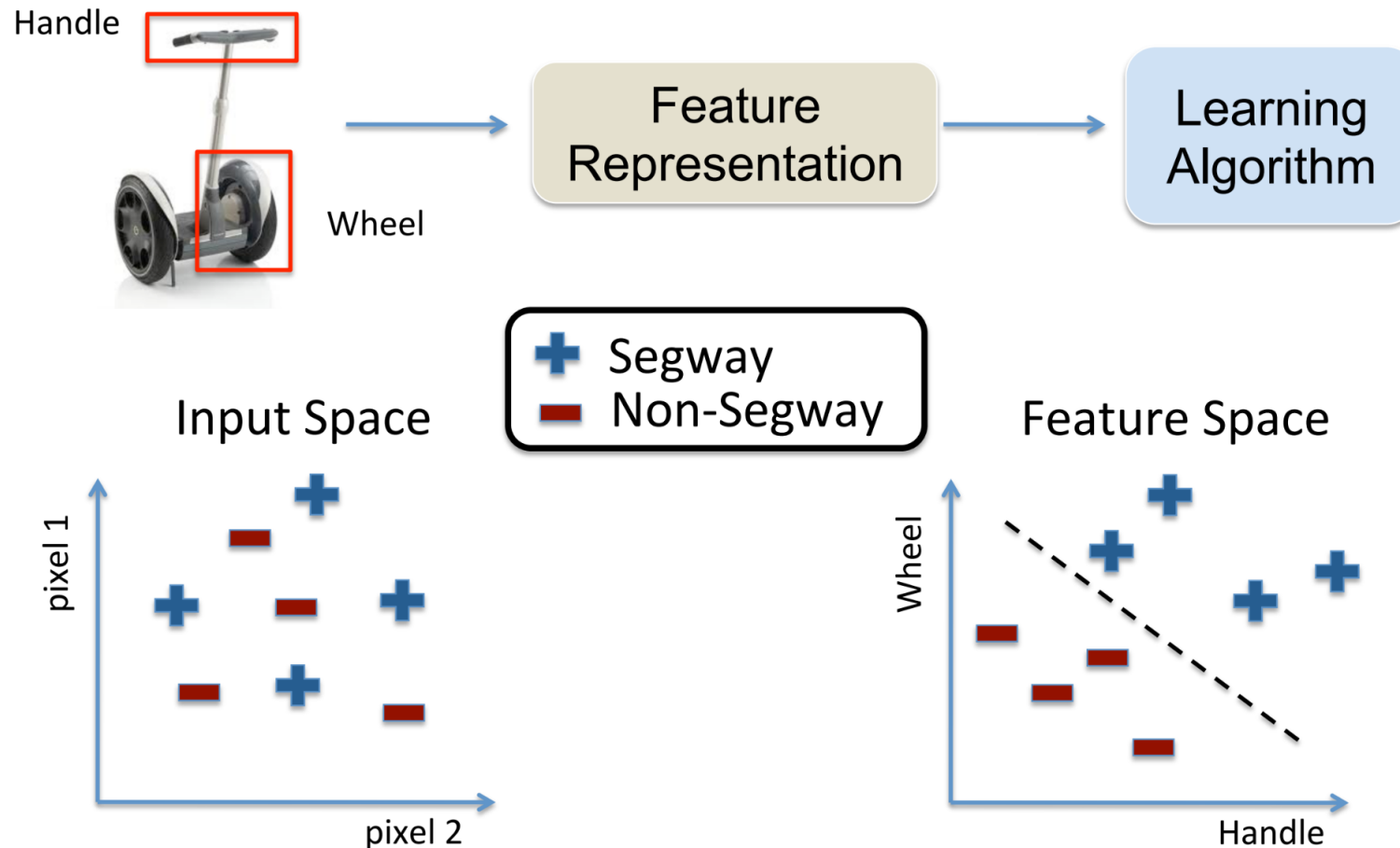
Deep learning is defined as “hierarchical feature learning” by Samy Bengio (Turing Award Winner)

Ok, so there are two terms here, hierarchical and feature learning.

What is feature learning?

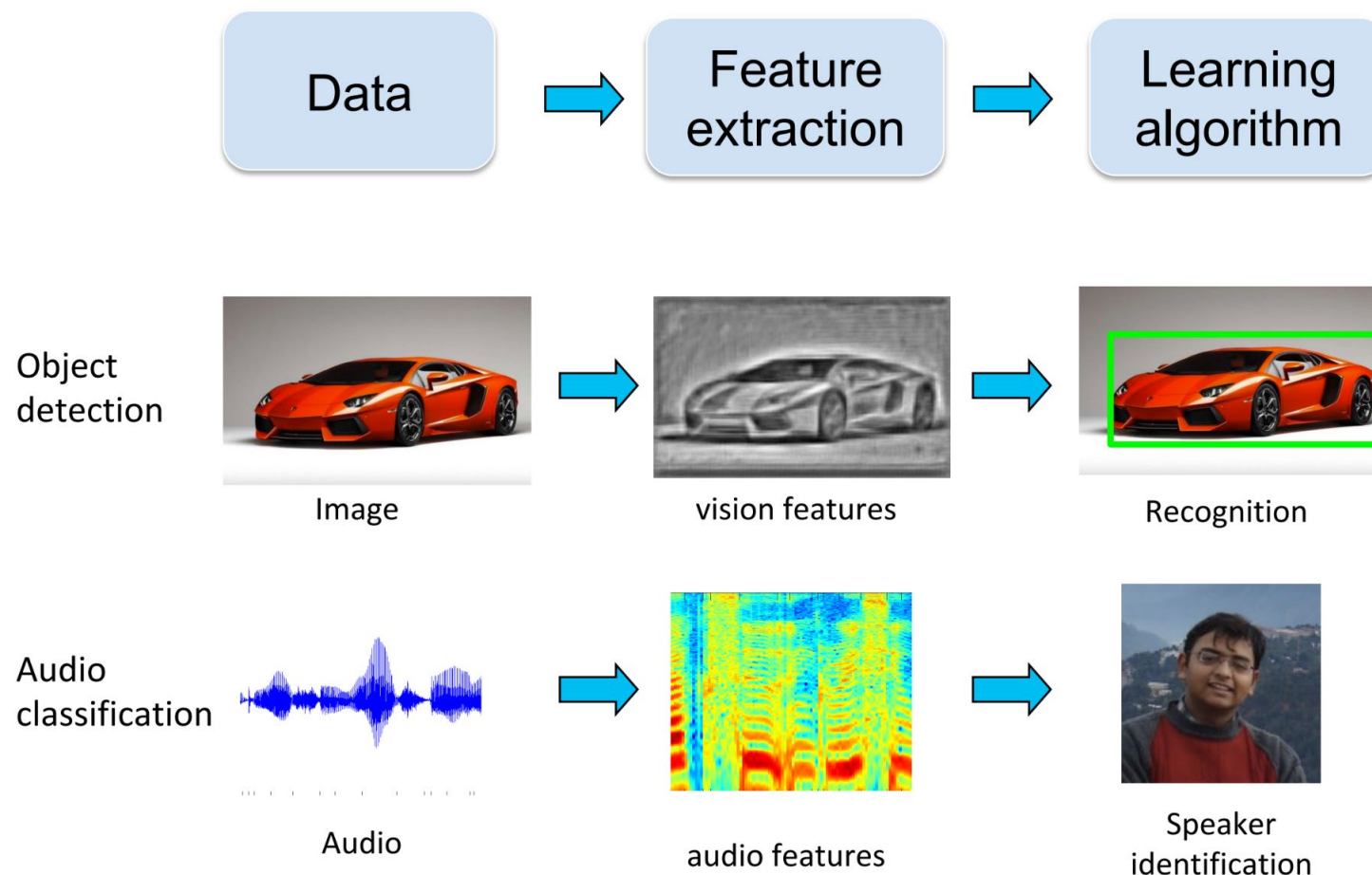
Feature learning

Learning Feature Representations

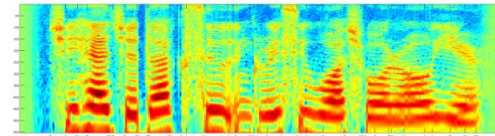
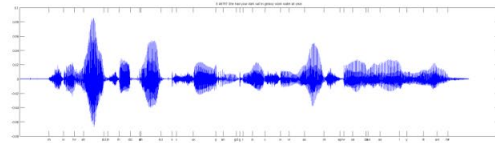


Hand-crafted features (traditionally)

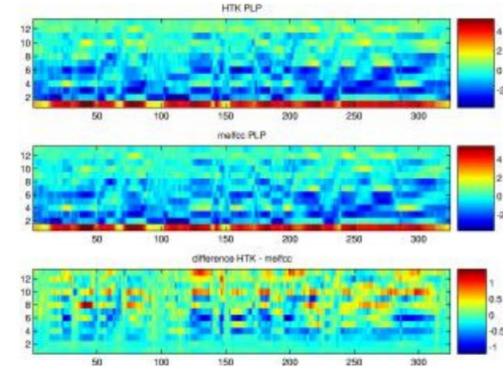
Traditional Approaches



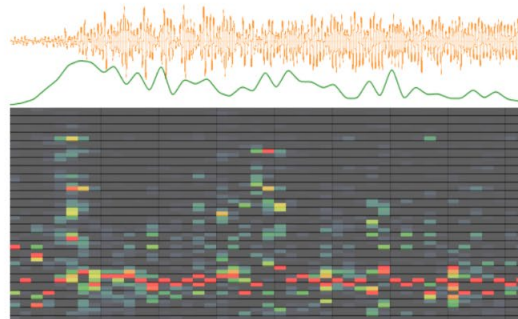
Audio Features



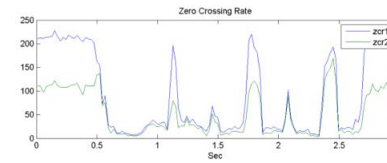
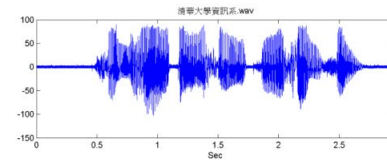
Spectrogram



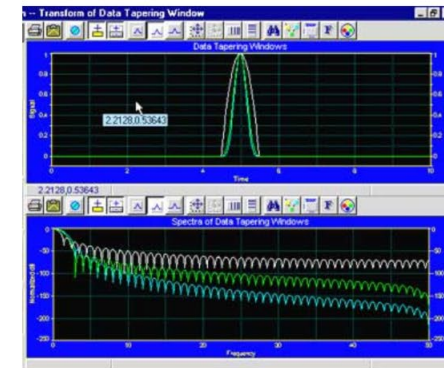
MFCC



Flux



ZCR



Rolloff

Traditional approach



Using hand-crafted features to map the input data to the feature space.



Original data: Alice is very happy today, so she jumps into the sky and swims in the water. Eventually, she dances in an anti-clockwise fashion and meets a gentleman called Bob who is rotating clockwise.



Some possible features: (Names: Alice, Bob), (Actions: Jump, Swim, Dance, Meet, Rotate), (Emotions: happy)...



Then we make some predictions based on the features.

Deep learning

- Instead of using prescribed features, we train the models to learn the features automatically.
- Mathematically, instead of predicting the label using (a linear) combination of prescribed features: $y = WP(x)$, where P is a prescribed feature mapping and W is a trainable parameter, we also train P (by parameterizing it in some way).
- How do we parameterize P ?
 - This is where the word “hierarchical” comes from.



Hierarchical

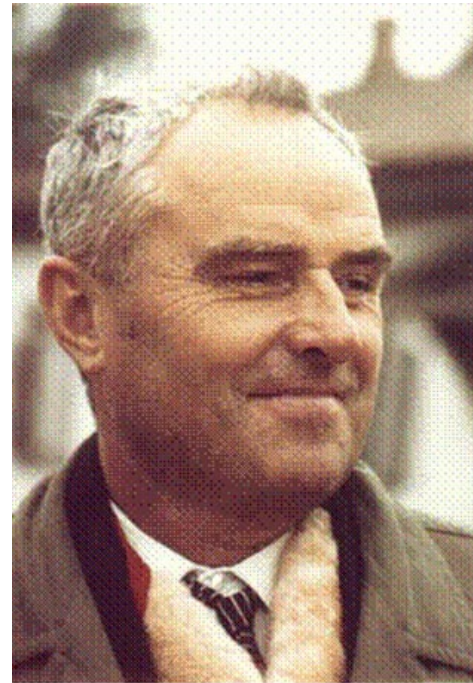
Deep learning is not the only feature learning method out there, but it is the “only” hierarchical feature learning method.

We will discuss the term hierarchical more thoroughly later

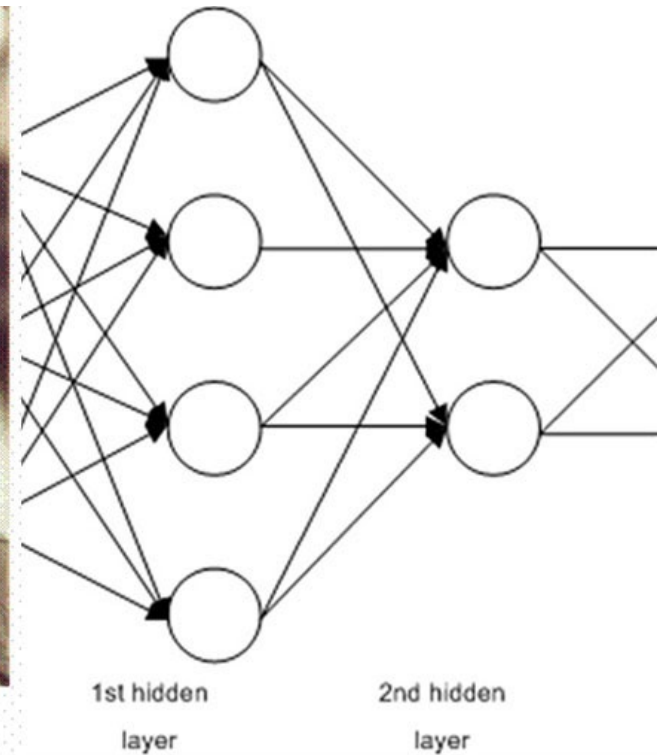
Before that, let us first talk about the history of deep learning, and what we can learn from the history.

Deep learning

- The idea of Deep learning is not new at all, it is actually more than 50 years old.
- People are already testing the idea of deep learning in 1967, using (almost the exact same) architecture and training the model using (almost the exact same) algorithm as we use today.
- But deep learning was never a real thing before the year 2010...
- So why is that?

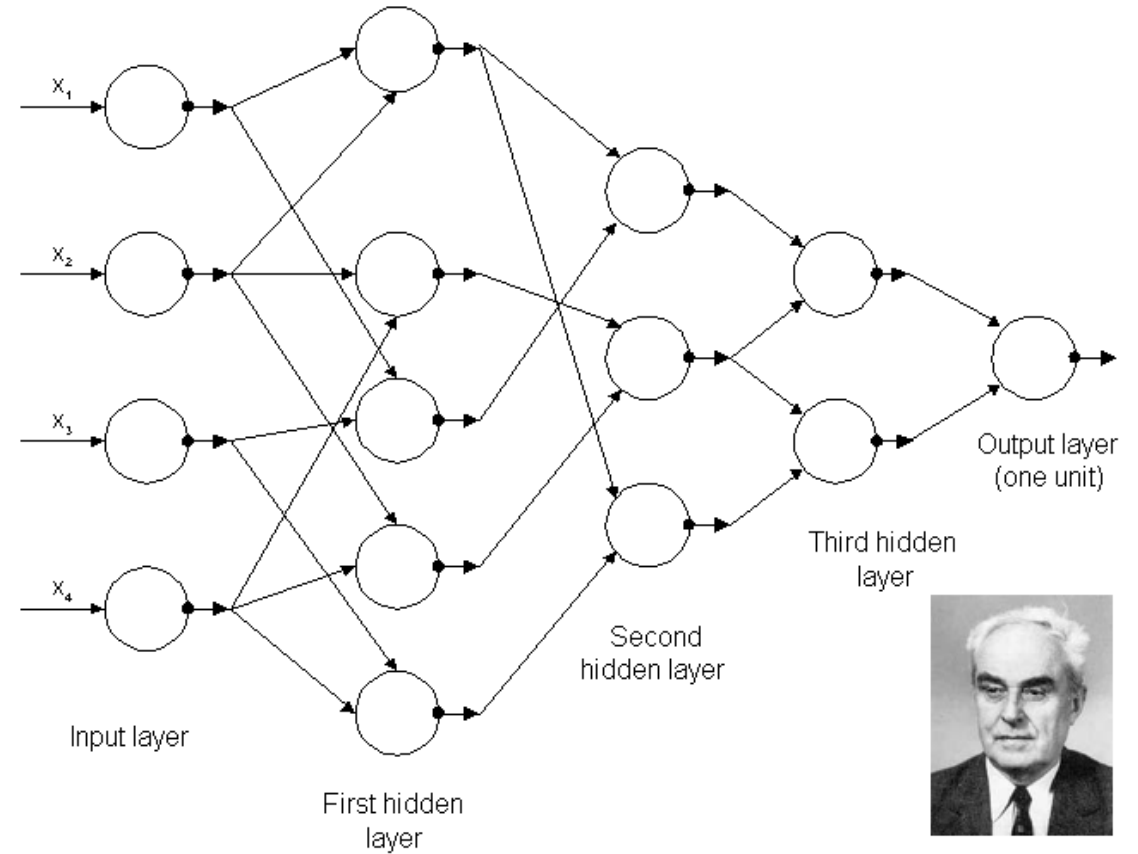


О.Г. Ивахненко (1967 г.)



The most basic deep learning architecture

- What is the architecture used in 1967?
- The architecture looks like this:



This is (almost) the same architecture used in (part of) the transformer (GPT-4) today

- This is the so-called “MLP layer” in the transformer.
- MLP = Multi-layer perceptron.
- So why there is not much success in deep learning at that time?



Deep learning: The struggle in early days.



Deep learning faces two main issues back then.



(1). Deep learning is a “feature learning model”, which means we need to parameterize the features as well. These models typically have much more trainable parameters than a non-feature learning model using prescribed features. So, deep learning models were slower to train.



(2). More importantly, the additional training parameters in deep learning make them easier to overfit.

For prescribed features, we use our own prior knowledge (which we accumulated through our lifetime, after seeing billions of inputs) to craft the features.

For deep learning models trained from scratch, it may take billions of training examples as well to discover those features.

Deep learning: The struggle in early days.

For deep learning models trained from scratch, it may take billions of training examples as well to discover those features.

So, it takes a lot of training examples.

It is still very hard (even today) to incorporate prior knowledge about the features of a human into a deep-learning model (There is one way to do that, which is by baking the knowledge into the structure of the deep-learning model, we will talk about that later).

Deep learning: The struggle in early days.



So, deep learning in the early days compared to those models based on prescribed features:

- Requires much more time to train (due to parameterizing the features).
- Requires much more samples (due to learning those features).

Clearly, it wouldn't be successful, until

- In 2009, Andrew Ng discovered that deep learning models are 100x faster to train on GPUs.
- In 2009, the ImageNet dataset was created with more than 1M training examples (now it's over 14M).
- Nowadays, we have (a single) GPU that is 2334 times faster than the ones in 2009.
- Nowadays, we have datasets that is 2,000,000 times larger than ImageNet...

Deep learning: The Bloom



Deep learning became increasingly more popular in the 2010s.



In 2012, AlexNet (a convolution neural network), trained on ImageNet, outperformed all prescribed feature models by a large margin on image recognition.



In 2015, the residual link was discovered.



In 2017, Transformer architecture was discovered, and the BERT model tops the benchmarks by a large margin compared to previous models. (Almost) The same architecture is still used today in GPT-4. The model has 700M parameters and was trained on 3.3B tokens.



In 2019, GPT-3 was released with 175B parameters, trained on 200B tokens. The model shows early signs of a “general intelligence”, instead of a task-specific model.

So in short, deep learning typically



Requires a lot of computing.

Requires a lot of training data.

But it really works if you have them.

Why? This is the benefit of automatic “feature learning”.

- If I ask you to describe what features you use to understand a sentence or recognize an image, can you describe it?
- It is very hard for humans to tell how exactly they do those things – So it’s hard to find good handcrafted features.
- Neural networks might discover those features that “can not be spoken” automatically.

So what do
we want to
learn next?



As we saw, the advantage of deep learning is “feature learning”.



But we also talked about there are many other “feature learning” methods.



What is special about deep learning as a feature learning method?

Hierarchical Feature Learning

As we saw, deep learning is known as “hierarchical” feature learning.

Hierarchical is what makes deep learning special.

What is hierarchical? Let’s begin by thinking about how humans learn stuff.

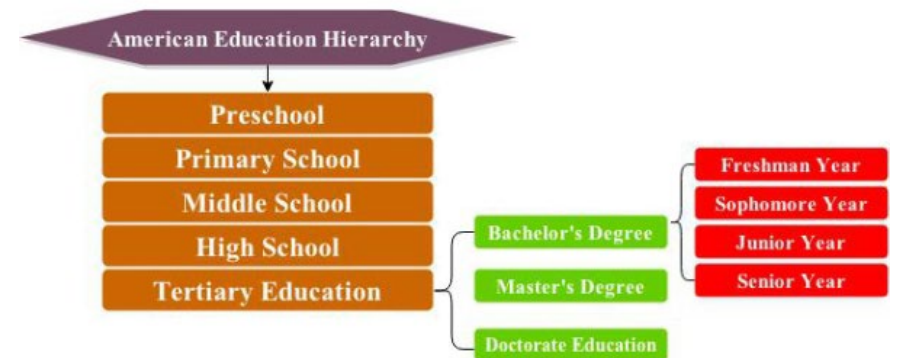
As a human, we have multiple stages of learning, from simplest to hardest.

Think about learning Math (solving linear systems).

- We first learn how to do basic arithmetic of numbers in elementary school.
- We then learn simple variable arithmetic in middle school.
- We then learn matrix operation in high school
- Finally we learn how to solve linear systems using all the above knowledge.

Hierarchical Feature Learning

- Think about learning Math (solving linear systems).
 - We first learn how to do basic arithmetic of numbers in elementary school.
 - We then learn simple variable arithmetic in middle school.
 - We then learn matrix operation in high school
 - Finally we learn how to solve linear systems using all the above knowledge.
- This is called a hierarchy.
- We first learn some basic concepts, then we learn to combine the basic concepts to form more and more complicated ones.
- In this way, we reduce the complexity gap of learning.



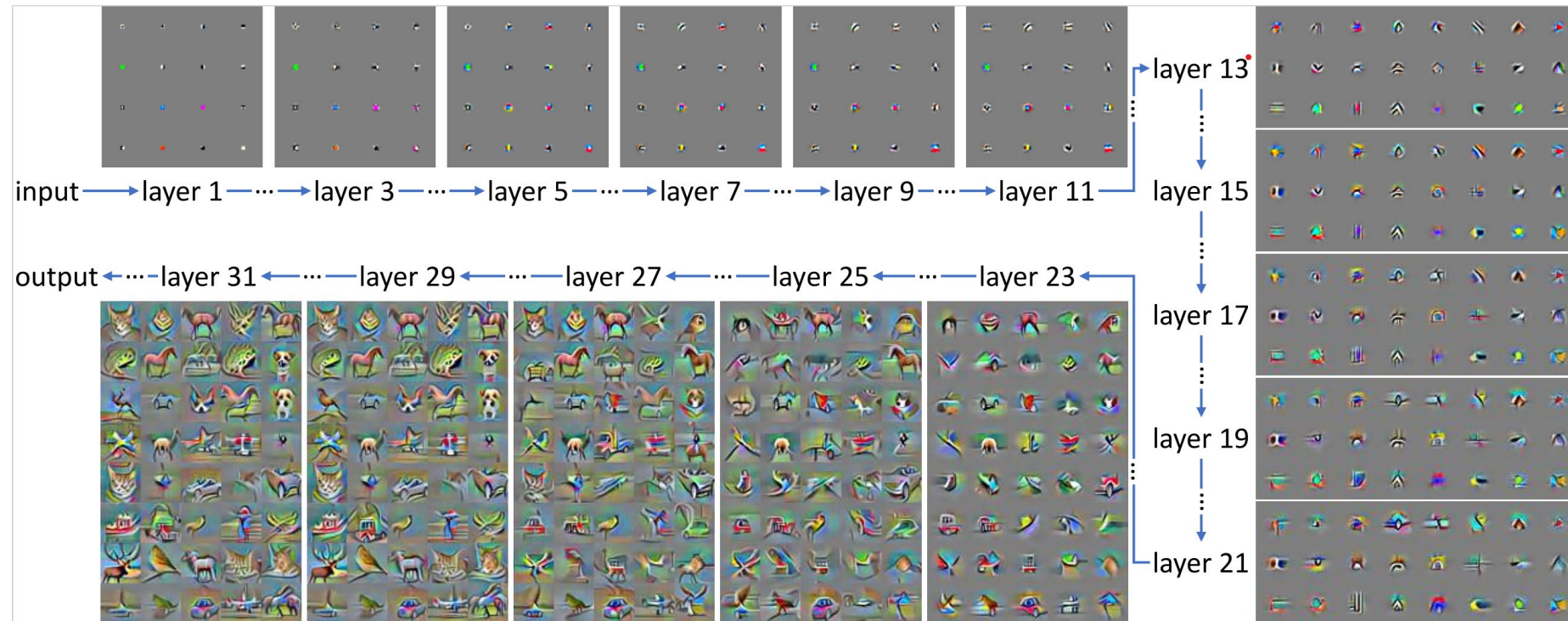


Hierarchical Feature Learning

- In this way, we reduce the complexity gap of learning.
- After mastering basic arithmetic, learning variable arithmetic is not that hard.
- After mastering the variable arithmetic of one variable, learning multi-variant arithmetic is not that hard.
- After learning multi-variant arithmetic, learning matrix operation is not that hard.
- But directly learning matrix operation is much harder (thinking about teaching an elementary school student this thing).

Hierarchical Feature Learning

- The spirit of hierarchical learning:
- We first learn the basic concepts, and then we compose those concepts to form concepts in **increasing difficulty/complexity**.



Hierarchical Feature Learning

- Deep learning tries to realize the hierarchical feature learning, by parameterizing the features using multi-layer neural networks.
- Mathematically, given input x , a L -hidden layer feed-forward neural network is defined as:
- $$F(x) = W_{L+1} \sigma(W_L \sigma(W_{L-1} \sigma \circ \dots \circ \sigma(W_1 x + b_1) \dots + b_{L-1}) + b_L) + b_{L+1}$$
- Where $W_1, W_2, W_3, \dots, W_{L+1}, b_1, b_2, \dots, b_{L+1}$, are all trainable parameters (each of them (W_l) is a matrix in dimension $d_{l+1} \times d_l$, where $d_1 = d$ is the input dimension, d_{L+2} is the output dimension).
- σ is called the activation function, it is applied element-wise, such as sigmoid function, ReLU function, GeLU function.
- $h_l(x) = \sigma(W_l \sigma \circ \dots \circ \sigma(W_1 x + b_1) \dots + b_l)$ is called the hidden embedding (of x) at layer l .

Hierarchical Feature Learning

- $h_l(x) = \sigma(W_l \sigma \circ \dots \circ \sigma(W_1 x + b_1) \dots + b_l)$ is called the hidden embedding (of x) at layer l .
- Key observation: The complexity of h as a function of x is increasing with respect to l .
- However, each hidden embedding is always a linear combination of the hidden embeddings of the previous layer, plus a (element-wise) non linear activation.
- This means that each hidden embedding is a relatively simple function of the previous layer, but $F(x)$ can be a super complicated function of x as L becomes large.
- Recall, this is exactly like how humans learn stuff.

Hierarchical Feature Learning

- Key theoretical support:
- Theorem: As long as $d_{l+1} \geq 2d$ for every $l \in [L]$, and as long as σ is non-linear, then for any (Lipschitz) function G of x , as long as $L \rightarrow \infty$, there exists a sequence of $W_1, W_2, W_3, \dots, W_{L+1}, b_1, b_2, \dots, b_L, b_{L+1}$ such that $F(x) \rightarrow G(x)$.
- This means that as long as we go deep, we can approximate any function (features) of x using deep neural networks.
- Neural network, theoretically, can learn any features of x ! But still, we need to train the network effectively in practice to reduce computation time/sample complexity ...