ARTIFICIAL INTELLIGENCE AND NEURAL NETWORKS

Lecture 1b – Neural Networks and Perceptrons

Chizhi Chris ZHANG

zhangchizhi@ciomp.ac.cn
Advanced Computing and Digital Technology Research Center
University of Chinese Academy of Sciences

Spring 2026

Today's Question

Main question

Why did modern AI need neural networks, and what do neural networks add when one simple rule or one simple boundary is not enough?

A familiar difficulty

Recognizing handwriting, hearing speech in noise, understanding a face at a strange angle, or reading the tone of a sentence all feel natural to humans and awkward to describe as a long rulebook.

What this lecture will not do. It will not ask you to memorize a museum of model names. It will build one clear line: simple neuron, limit, hidden layer, training, model families, and limits.

Some Tasks Resist Handwritten Rules

Why start here

Many tasks are easy for people because we solve them through pattern recognition, context, and experience rather than through explicit lists of instructions.

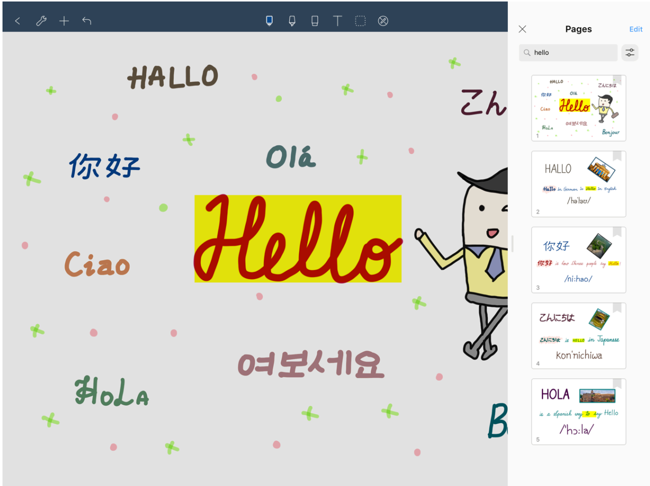
Examples students immediately understand

Recognizing a friend's face, reading a messy note, hearing a word through background noise, or deciding whether a sentence sounds sarcastic.

Why rules struggle

The number of cases is too large, and the clues matter together rather than one by one.

Handwriting Is Harder Than It Looks

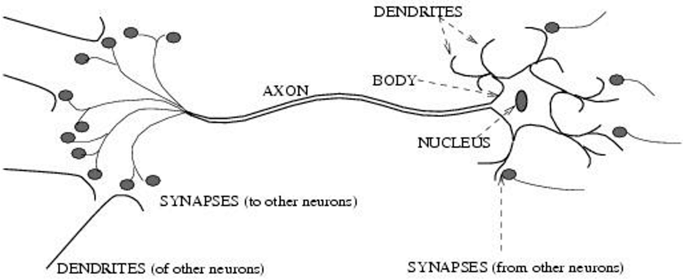


The same word can be slanted, rushed, broken, or mixed with other marks. Humans still read it surprisingly well.

Why this matters. A useful model usually has to notice strokes, corners, letter parts, and word shapes before it can make the final call.

Why this points toward neural networks. The model often needs intermediate clues such as strokes, corners, letter parts, and word shapes before it can make a final judgment.

The Brain Analogy Helps and Misleads



The idea of many simple units working together gave researchers a concrete way to imagine distributed computation.

Why the analogy misleads

Artificial neural networks are inspired by biology, but they are simplified mathematical systems. They do not copy a real brain cell by cell.

Why Layered Patterns Matter

In images. Pixels are not born with labels such as edge, corner, eye, or cat. The model must build higher-level features from lower-level clues.

In language

The meaning of a sentence depends on word order, context, and composition. A single local rule misses too much.

The broad lesson. Neural networks become useful when the model must learn a better internal representation before it can produce a good answer.

Keep One Judgment Question

Keep this question in mind

When is a simpler model already enough, and when does the task truly need representation learning from a deeper network?

Why this question matters

It stops neural networks from becoming automatic upgrades. Bigger is not always better, and deeper is not always necessary.

This question returns later. It will come back when we discuss multilayer perceptrons, image models, sequence models, generative models, and agents.

One Neuron as a Weighted Vote

The weighted-vote picture

Each input contributes some evidence. Positive weights push the score upward, negative weights pull it downward, and the bias changes how much evidence is needed before the neuron says yes.

A small spam example

"Unknown sender", "many links", and "urgent wording" may each add a bit of evidence. "Known contact" may subtract some of it.

How to imagine it

Several weak hints are combined into one score, and that score becomes one judgment.

This is why the neuron idea lasts. It is simple, but it already captures how many practical decisions are made.

The Smallest Useful Formula

A neuron in symbols

$$z = \sum_i w_i x_i + b$$

$$a = \phi(z)$$

How to read it. Inputs x_i are weighted by w_i, combined into a score z, shifted by a bias b, and then transformed by an activation function ϕ.

Why this is enough for today. Students do not need every derivation yet. They need the story: combine evidence, shift the boundary, transform the result, pass it on.

Threshold Changes the Decision

What threshold really changes

The score can stay the same while the final decision changes, because the cutoff moved.

A familiar example

A spam filter may score two emails similarly, but the inbox decision changes when the threshold for caution changes.

Why this matters

Model output is not policy. The threshold already reflects the cost of mistakes, the level of caution, and the context of use.

That idea comes back later when we talk about calibration, false alarms, and deployment choices.

Why this matters. The model output and the final policy decision are related, but they are not the same thing.

One Neuron Can Already Do Something Useful

One score can support several tasks

The weighted sum $w^\top x + b$ is already useful. What changes is how we decide to read it.

Classification version

A bank may turn the score into a yes-or-no decision: approve or reject, suspicious or ordinary, pass or fail.

Scoring version

The same score can also stay continuous and help rank items, compare risk, or express how strongly the evidence points in one direction.

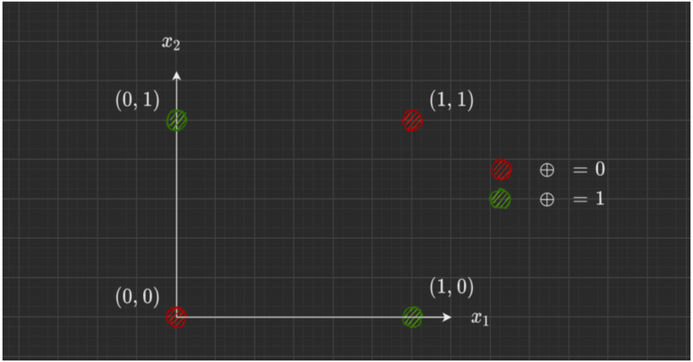
Two simple readings

$$\hat{y} = \mathbb{1}(w^\top x + b \geq 0)$$

$$s = w^\top x + b$$

The first makes a hard choice. The second keeps the amount. That is why even one neuron is already more than a toy.

XOR Shows the Limit



It is a tiny example with a big lesson. One straight boundary cannot separate the two classes correctly.

That is why XOR stuck in memory. The counterexample is so small that nobody can pretend the limitation is vague.

Why researchers cared. If a simple model fails on such a clean example, then the field has found a real boundary in model capacity.

Activation Changes the Shape

Why activation matters

If every layer only performed a linear transformation, stacking many layers would still collapse into one big linear transformation. **The essential point.** Activation

functions introduce nonlinearity. That is what lets deeper networks bend the decision surface and represent richer patterns.

One sentence to remember. Depth without nonlinearity is mostly repetition, not extra expressive power.

Hidden Layers Learn Middle Steps

What hidden layers really do. They allow the network to build intermediate features instead of jumping straight from raw input to final output.

A handwriting intuition

Early layers may react to short strokes, later layers to letter pieces, and later still to whole characters or words.

Why this matters. The network is not only learning answers. It is learning how to describe the problem in a more useful internal language.

A Network Is Repeated Composition

A compact expression.

$$h_1 = \phi(W_1 x + b_1)$$

$$h_2 = \phi(W_2 h_1 + b_2)$$

$$\hat{y} = W_3 h_2 + b_3$$

How to read it. Each layer takes the previous layer's output, transforms it, and passes a new representation forward.

Why this helps. Complex reasoning in a network is often built as a sequence of simpler transformations rather than one giant jump.

From Edges to Meaning

Why depth feels natural

Early layers can react to small clues. Later layers combine those clues into parts, patterns, and finally a judgment.

A handwriting story

One unit may respond to a short stroke. A later combination notices a loop or a corner. A still later layer has enough evidence to say, “this probably looks like a 9”.

The same idea appears elsewhere

In language, tiny patterns become words, phrases, and meaning. In sound, local frequencies become syllables and then recognized speech.

Depth matters because many tasks are easier as a chain of small judgments than as one giant leap.

The Training Loop in Plain Language

Training cycle. Predict, compare with the target, measure the error, adjust the parameters, and repeat many times.

A classroom analogy

It is like practicing with feedback. One attempt is not the skill. Improvement appears across many corrections.

Why this matters. Neural networks are not powerful because they were handed perfect knowledge. They are powerful because they can be trained from many examples.

Why the History Matters

This is not a list of dates

The history of neural networks matters because it shows a repeated pattern in science: a strong idea may arrive before data, computing, or training methods are ready for it.

What students should notice

The field was not a straight line of success. It had optimism, disappointment, return, and reinvention.

The lesson from the history. An idea can be early without being wrong.

The First Dream

The early intuition. Researchers such as McCulloch and Pitts asked whether intelligence could emerge from many simple units connected together.

Hebb's intuition. "Neurons that fire together wire together" gave a memorable way to connect learning with changing connections.

Why this still matters. Even now, the central idea has not disappeared: learning changes how signals are weighted and combined.

Perceptron Brings Hope

Why perceptron was exciting

It turned the artificial neuron from a thought experiment into a trainable model. **Why the excitement was reasonable.** If one artificial neuron could learn a decision boundary, it was natural to imagine that richer networks might learn far more.

What changed

The field now had a concrete object to test, criticize, and improve.

XOR Brings a Hard Limit

Why XOR mattered historically. It gave a small and undeniable counterexample. The problem was not vague. The capacity limit was clear.

Why a small counterexample hurts. A simple failure can be more damaging than a messy large failure because nobody can hide behind complexity.

The lesson

Single-layer perceptrons are useful, but they are not enough for richer nonlinear structure.

Backprop Brings the Return

What changed in the 1980s

The field realized the early dream was limited more by training and architecture than by the basic direction itself.

Why backpropagation mattered. It made it practical to train multi-layer networks by sending error information backward through the layers.

One line worth remembering

The idea was not wrong. It was early.

Why 2012 Changed the Plot

Why this time was different. Large datasets, stronger GPUs, and better architectures arrived together. The conditions finally matched the ambition.

Why people noticed. Image recognition improved enough to make the wider AI community accept deep learning as a central route rather than a side topic.

Historical lesson. Technical success is rarely caused by algorithm alone. Data, computing, and engineering matter too.

From Recognition to Agents

What happened next. Neural networks moved from recognition tasks into language, generation, multimodal systems, and agent-style workflows.

Why people care now. People now ask AI systems not only to classify, but also to explain, generate, plan, use tools, and complete multi-step tasks.

The broad shift. AI has been moving from a pure prediction problem toward a system problem.

One Map, Several Families

How to read this map. Do not memorize model names as a list. Read them as answers to different kinds of data.

MLP	CNN	RNN family	RL	GAN / VAE
Tabular or static patterns	Images and spatial structure	Sequences and time	Action over time	Generation

Start from the task, not the model. First ask what kind of data you have and what kind of answer you need.

MLP Fits Static Patterns

What MLP is good for. Tabular data, mixed features, and static prediction problems where the input is one feature vector rather than an image or a long sequence.

Examples

Risk scoring, customer behavior prediction, pricing, and many ordinary business or scientific prediction tasks.

Why it matters. Not every neural-network problem begins with images. Many begin with rows and columns.

CNN Fits Images

Why images need a special structure. Nearby pixels matter together, and the same useful local pattern can appear in many different positions.

What CNN adds. Local receptive fields and shared parameters make the architecture better suited to spatial structure.

Typical cases. Medical images, microscopy, industrial inspection, and everyday computer vision tasks.

RNN Family Fits Sequences

Why sequence is special. Order matters. The current element often depends on what came before.

Why LSTM and GRU appeared. Standard recurrent models often struggled to remember long-range dependencies, so gated variants were introduced to retain or forget information more effectively.

Typical cases. Language, speech, time series, sensor streams, and other data where sequence is part of the meaning.

Reinforcement Learning Fits Action

Why this family is different. The goal is not only to predict. The goal is to act, receive feedback, and improve a policy over time.

Examples

Games, navigation, robotics, and any task where one action changes the next state of the world.

Why deep RL matters. Neural networks help the agent cope with complex inputs and richer decision policies.

Generative Models Change the Goal

What changes here. The model is no longer only deciding between existing categories. It is trying to generate new samples that look like they came from the same world as the training data.

Two famous routes. GANs learn through a generator and a discriminator. VAEs learn a latent space that supports reconstruction and sampling.

Why this matters. Generation is one reason neural networks moved from recognition systems toward creative and assistant-style systems.

How to Choose a Network

Two questions first

The first question. What is the structure of the data: static features, images, sequences, or a stream of actions?

The second question. What is the real goal: classify, estimate, generate, or decide over time?

Do not choose the model first

Choose the family that matches the problem. Do not start from the most fashionable model and search for a place to use it.

Bigger Is Not Always Better

Why bigger looks tempting

Larger models can represent more complicated patterns.

Why bigger disappoints. Without enough data, compute, or careful evaluation, bigger models can overfit, become unstable, or waste effort on a task a smaller model could already solve.

The practical rule

Extra capacity only helps when the problem actually needs it and the training setup can support it.

Fluency Is Not Understanding

A modern warning. Neural systems can produce fluent, impressive outputs while still misunderstanding the task, missing context, or failing outside familiar data.

Why this matters beyond specialists. Users often trust systems that sound confident. The safer habit is to ask what evidence the system saw and what kind of failure it is known to make.

Do not confuse fluency with understanding

Impressive output quality does not guarantee deep understanding or reliable reasoning.

Summary

- Neural networks matter when the task has layered, nonlinear structure that one simple rule cannot capture.
- The basic unit is actually simple: combine evidence, shift the boundary, apply an activation, and update from feedback.
- There is no single best network for everything. Different data shapes and different goals call for different model families.

Next Time

Where we go next. The next lecture turns this picture into one concrete case: the perceptron and data separability.

Keep this question in mind. If one neuron has a clear limit, what exactly changes once we add layers, better training, and the right structure for the data?
Why that matters. That question links directly to decision boundaries, learning rules, and the rest of the neural-network story.

Thank You