ARTIFICIAL INTELLIGENCE AND NEURAL NETWORKS

Lecture 4b – Neural Computation, Activations, and Training

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

What we are trying to answer

How do many simple neural computations become a trainable system, and why do activation choice and training stability matter so much once networks become deep?

Why this lecture matters

If we only say that neural networks are powerful, later lectures turn into vocabulary. If we understand how they compute and learn, the later models stop feeling like magic.

What changes from NN3

NN3 introduced hidden layers and backpropagation. NN4 slows down and asks what each unit computes, how signals move, and why training can become fragile.

From NN3 to NN4

Last time

We learned why hidden layers matter and why backpropagation is the basic way to train them together.

Today

We stay with the same family of models, but now the focus is the machinery: activations, signal flow, gradient flow, and the choices that make training work or fail.

One simple sentence

NN3 said that deep models can represent more. NN4 asks what lets that power become trainable in practice.

A Network Does Two Jobs

First job: compute

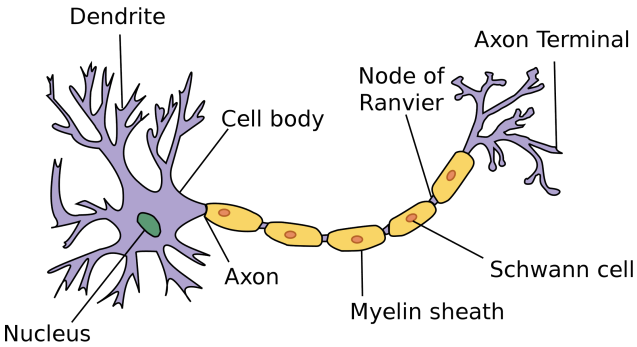
Inputs come in, pass through layers, and become an output.

Second job: learn

If the output is bad, the network has to update its parameters so the next answer gets better.

This is the whole lecture in two verbs: compute and learn.

From Biology to Engineering



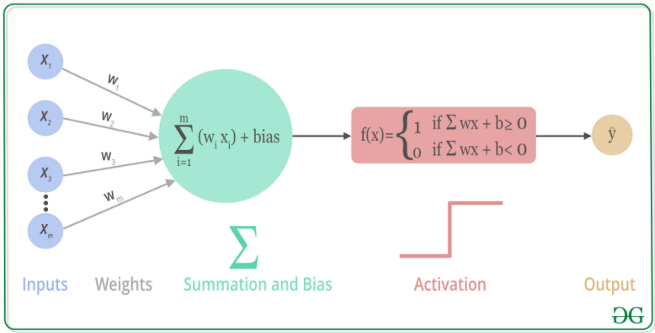
The original metaphor

Researchers borrowed the rough idea that many simple units could work together to produce rich behavior.

The classroom reality

Artificial neural networks are not copies of biological brains. They are trainable mathematical systems designed to transform signals into useful outputs.

Signal Flow in One Unit



What happens inside

Inputs arrive, weights scale them, a bias shifts the total, and an activation turns that total into the next signal.

Why this matters

Later networks may be huge, but they still repeat this same local logic again and again.

Why Stability Became Part of the Story

Representation is only half the problem

A deep network may be expressive enough in theory and still fail to learn well if gradients vanish, signals explode, or optimization stalls.

Why this lecture exists

Once a model is deep, activation choice, initialization, normalization, and learning rate stop feeling like details. They shape whether training is even feasible.

One Neuron Formula

Minimal form

$$z = w^\top x + b, \qquad a = g(z)$$

Read it plainly

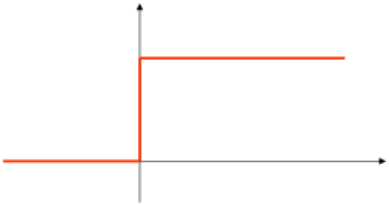
The neuron first builds a weighted summary of the evidence and then reshapes that summary through an activation.

What not to forget

Without the activation, stacking layers would mostly collapse into one larger linear map.

Why Activation Exists

$$u(t) = \begin{cases} 1 & t > 0 \\ 0 & t < 0 \end{cases}$$



Without activation

More layers mainly repeat linear mixing.

With activation

The network can bend, gate, and reshape the signal before handing it to the next layer.

The big idea

Nonlinearity is the reason depth becomes expressive instead of redundant.

Sigmoid and Tanh as Smooth Choices

Sigmoid

Outputs stay between 0 and 1, so the curve feels natural when we want an output that behaves like a probability.

What to remember

It is easy to read, but the flat ends can weaken gradients when activations drift too far.

Tanh

Outputs stay between -1 and 1, so hidden activations are more centered around zero.

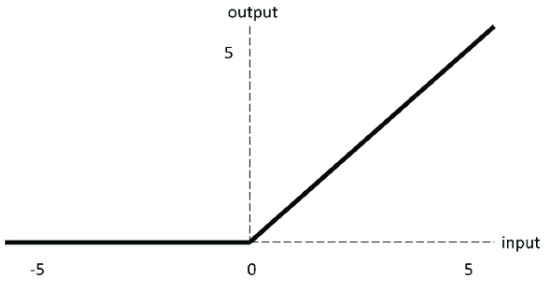
What to remember

That zero-centered behavior can make hidden signals feel less one-sided, even though saturation can still appear.

How to read these two curves

Both curves are smooth and bounded. The tradeoff is interpretability versus saturation behavior, not “old is useless” versus “new is correct”.

Why ReLU Changed Practice



What ReLU does

It keeps positive signals and cuts off negative ones.

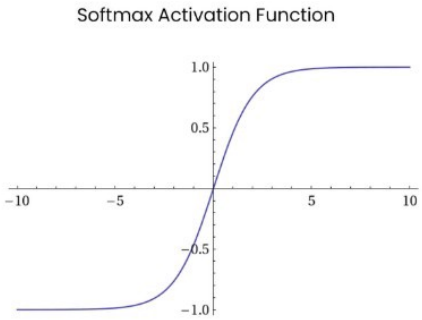
Why it became influential

Its simple shape often makes deeper optimization easier than fully saturating activations.

Important nuance

ReLU solves some problems and creates others, such as dead units. No activation is universally best.

Softmax Gives the Output Meaning



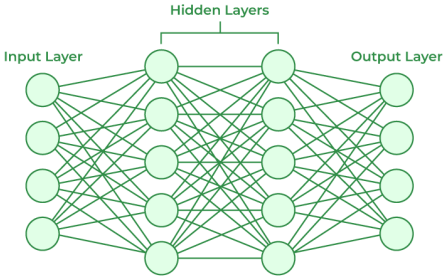
What softmax changes

Several raw scores become a competition for probability mass, so the outputs can be read as a distribution across classes.

Why this is practical

The last layer is not decoration. It has to match the kind of decision the task is asking for.

From One Unit to One Layer



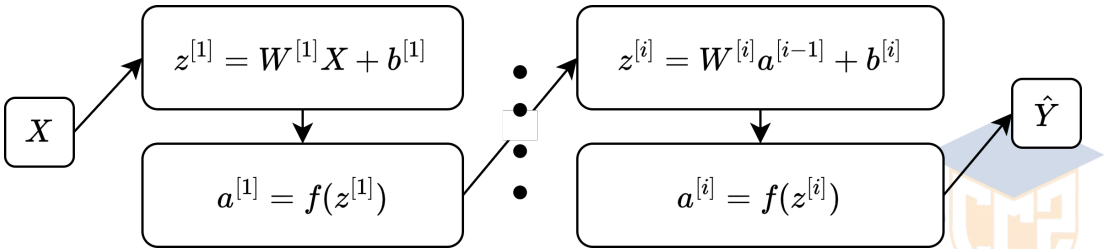
A layer view

One neuron makes one learned signal. A layer creates many signals in parallel from the same input.

Why depth matters

Later layers do not see raw input anymore. They see the representation produced by earlier layers.

Hidden Layers Build Representations



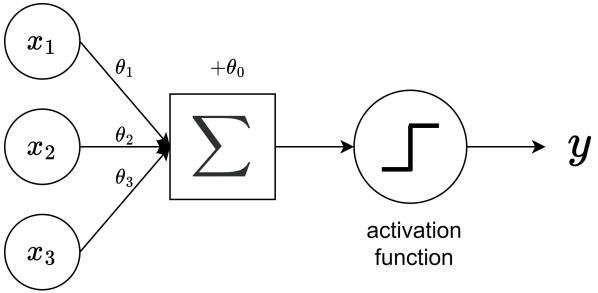
What hidden means

These units are not directly observed in the world. They are internal features the network invents because they help later prediction.

Why students should care

This is the core reason neural models can handle raw images, sound, and text more naturally than many shallow tabular models.

Forward Pass as Transformation



Forward pass

Signals travel from input to output, being linearly mixed and nonlinearly transformed at each layer.

A teacher's sentence

Each layer asks, "What should this information look like before I hand it onward?"

One Layer at a Time

Compact form

$$Z^{[l]} = W^{[l]}A^{[l-1]} + b^{[l]}, \qquad A^{[l]} = f\left(Z^{[l]}\right)$$

How to read it

Take the current representation, mix it with weights, shift it with a bias, apply an activation, and pass the new representation forward.

Why this formula matters

Almost every modern neural architecture is still a variation on this story.

Backpropagation Is Credit Assignment

The basic question

If the final prediction is wrong, which parameters inside many layers deserve how much blame?

What backpropagation does

It uses the chain rule to send error information backward through the computation graph so every weight can be updated coherently.

What it is not

It is not human-style reflection. It is a systematic way to distribute error through connected computations.

Local Changes Travel Through the Network

The bookkeeping idea

Each layer only needs to answer one local question: if my input changed a little, how much would my output change?

A group-project analogy

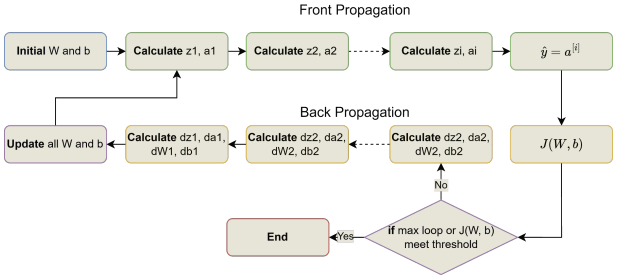
When the final result is disappointing, we do not blame every earlier step equally. We ask which part affected which later part. Backpropagation does the same with numbers.

One compact formula

$$\frac{\partial L}{\partial w} = \frac{\partial L}{\partial a}\frac{\partial a}{\partial z}\frac{\partial z}{\partial w}$$

The chain rule links these local sensitivities so an early weight can still receive useful feedback from a far-away loss.

Training Is Forward Then Backward



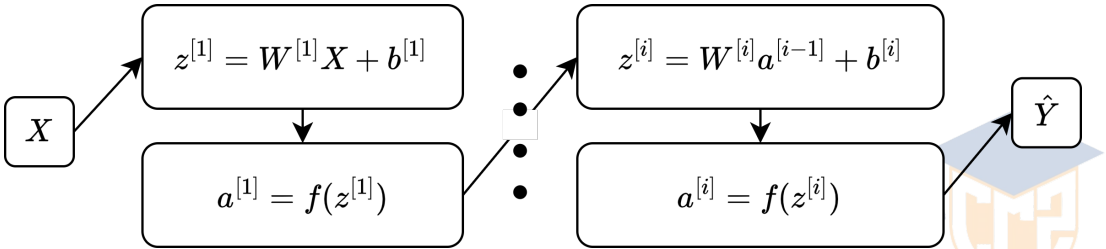
What repeats every step

Make a prediction, compare it with the target, compute the loss, send gradients backward, and update the parameters.

Why this cycle matters

Deep learning feels complicated from the outside, but the core training loop is still repeated error correction.

Inference Is Simpler Than Training



Inference phase

Once the parameters are learned, we usually only need the forward pass to produce an answer for a new case.

The clean contrast

Training teaches the model. Inference uses the model.

Batches, Epochs, and Learning Rate

Batch

A small group of examples used for one update step.

Epoch

One pass through the full training set.

Learning rate

The step size of the update. Too small can be painfully slow; too large can make training unstable.

Why this belongs in memory

Many “mysterious” training results are really about the rhythm of updates, not only about the architecture.

Loss Is the Training Compass

Why loss matters

The network does not directly optimize human satisfaction or social value. It optimizes the loss function we define.

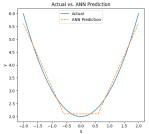
What this shows

If the loss captures the wrong goal, the network can train perfectly toward the wrong destination.

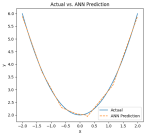
What students should notice

Architectures get attention, but objectives quietly decide what the model is actually rewarded for learning.

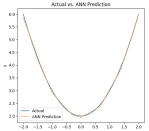
Model Capacity Changes the Fit



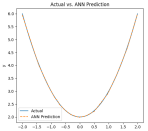
Too simple



Closer to the pattern



Starting to chase local bumps



Chasing noise

How to read the sequence

From the first panel to the fourth, the model goes from too stiff, to more suitable, to overly eager to follow every wiggle in the sample. That is the basic tradeoff behind model capacity: more flexibility can help, but past a point it starts fitting noise instead of structure.

Learning Rate Changes the Journey

If the step is too large

Updates can overshoot useful directions and make the loss bounce or diverge.

If the step is too small

Training may become so slow that useful structure is never reached in a practical amount of time.

Why this matters for modern AI

Large models are expensive to train, so optimization settings shape cost, time, and final quality.

Initialization Shapes the First Steps

Bad initialization

If units begin too similar or badly scaled, the network starts from a poor geometric position and gradients become less useful.

Better initialization

Reasonable starting scales help signals and gradients stay in healthier ranges as learning begins.

Training quality depends not only on data and architecture, but also on how the parameters are allowed to begin.

Training Diagnostics Tell a Story

If train and validation both struggle

The model may be underpowered, the features may be weak, or the task framing may be poor.

If training improves but validation stalls

The model may be memorizing, the split may be dishonest, or the data pipeline may be mismatched.

Why this matters

Loss curves are not only technical diagnostics. They are clues about whether the model is learning structure or just chasing the past.

Regularization Is Controlled Restraint

Why regularization appears

If the model is too free, it can fit noise, memorize quirks, and become fragile outside the training set.

How to think about it

Regularization is a way of telling the network, “learn a useful pattern, but do not become too eager to explain every tiny fluctuation.”

Regularization is not punishment for complexity. It is guidance toward patterns that are more likely to survive new data.

Normalization Helps but Not Magic

Why normalization helps

If signal scales drift too much across layers, optimization becomes harder and gradients can behave erratically.

What it buys

More predictable intermediate values, often healthier gradients, and usually easier training.

Why we should stay sober

Normalization can stabilize training, but it does not repair weak labels, a confused objective, or a misleading dataset.

Depth Does Not Repair a Broken Task

What extra depth still cannot do

It cannot rescue dishonest splits, confused labels, missing context, or a task definition that never made sense in the first place.

A realistic example

If past hiring data rewards a narrow and biased picture of “success”, a deeper network can fit that pattern more efficiently. It does not suddenly become fair just because it is expressive.

The practical rule

Before adding layers, ask whether the labels, the split, and the surrounding context deserve the extra capacity.

Powerful models amplify the structure in data. They do not invent trustworthy structure when the data lacks it.

Activation Choice Is a Tradeoff

Smooth activations

They can be intuitive and useful, but may saturate and weaken gradients.

Piecewise-linear activations

They often train well in deep nets, but can create inactive units or other failure patterns when handled poorly.

The practical lesson

Activation is not only a mathematical curve. It is also a training-behavior choice.

What a Falling Loss Does Not Prove

Misreading 1

A lower training loss does not mean the model understands the task. It may only mean the model found a more efficient way to mimic the training set.

Misreading 2

A larger model is not automatically easy to optimize. It can become more sensitive to initialization, learning rate, batch design, and noise in the labels.

A useful classroom rule

Read training curves together with validation behavior, failure cases, and data quality. Trust grows from agreement between those signals, not from one falling number.

Teams often celebrate the first beautiful loss curve and only later discover drift, subgroup error, or bad calibration.

Compute, Time, and Energy Matter Too

Why this belongs in a general course

Neural networks are not only mathematical objects. They also depend on hardware, energy, training time, and financial cost.

Why this changes judgment

Sometimes a slightly better model is not worth the extra energy, latency, or deployment complexity.

As models scale up, responsible judgment includes not just accuracy and elegance, but also efficiency and real-world resource use.

Why Vision Becomes the Next Step

Why images are the natural next topic

Pixels are high-dimensional, local patterns matter, and nearby features interact strongly. That makes vision a strong test of representation learning.

Why NN4 leads there

Once activations, layers, forward pass, backward pass, and stability are clear, convolution stops looking like magic and starts looking like specialized neural design.

Why AI4 and NN4 Belong Together

What AI4 showed

Some models organize judgment by asking a sequence of questions over rows and columns.

What NN4 showed

Neural models organize judgment by transforming signals through layers and learning those transformations from error.

Summary

- Deep networks still rely on simple local computations repeated across many layers.
- Activations are what make depth expressive instead of merely larger.
- Training is a cycle of forward computation, loss evaluation, backward credit assignment, and parameter updates.
- Stable learning depends on activation, initialization, normalization, regularization, and data quality together.
- A strong neural model still needs judgment about cost, efficiency, and real-world use.

Next Time

Where this story goes next

The later neural lectures apply the same machinery to richer data types and more specialized architectures, especially images.

What to keep in mind

The big models are built from the same ingredients we studied here: weighted sums, activations, losses, gradients, and repeated correction.

Thank You