NULLVECTOR
.monster

PHASE 03 — DEPTH

DAYS 43–63 · DEEP LEARNING · NEURAL NETWORKS · NVIDIA NCA-AIIO

Phase 2 taught you to use ML tools. Phase 3 teaches you how they actually work — all the way down.

Neural networks, backpropagation, CNNs, RNNs. You will derive gradients by hand before using autograd. You will build a convolutional neural network that recognizes images. You will understand why GPT-4 having 1.7 trillion parameters is significant. By Day 63 you will have earned the NVIDIA NCA-AIIO certification — validating your understanding of AI hardware and infrastructure.

If Phase 2 felt hard at times, Phase 3 will feel harder. That is not a warning — it is information. The people who get through Phase 3 are not the most talented. They are the ones who treat confusion as a signal to slow down, not to stop. Slow down. Ask questions. Use Claude to explain things differently. Come back tomorrow.

// PHASE 3 AT A GLANCE

Duration	21 days · 3.5 hours per day
Milestone	Day 63 — NVIDIA Certified Associate: AI Infrastructure & Operations
Primary	fast.ai Practical Deep Learning (free) — course.fast.ai
Parallel	3Blue1Brown Neural Networks series — youtube.com/watch?v=aircAruvnKk

Framework	**PyTorch — pytorch.org/tutorials**
NVIDIA cert	**learn.nvidia.com/en-us/training/self-paced-courses**
NVIDIA exam	**nvidia.com/en-us/training/certification**
Your library	**Mastering PyTorch 2E (Packt — your Humble Bundle)**

WEEK 1 · DAYS 43–49 · NEURAL NETWORKS FROM SCRATCH

DAY 043 · HOW NEURAL NETWORKS WORK

The most important 19 minutes you will watch this phase.

■ INDUSTRY (15 MIN)

Search: 'AI winters and AI summers — history of deep learning'

Context: neural networks are not new. What changed is compute and data.

■ STUDY (90 MIN) — 3Blue1Brown Neural Networks

Watch the full 4-video neural network series (about 1 hour total).

This is the clearest visual explanation of neural networks ever made.

After each video: write 3 sentences in your own words explaining what you just watched.

Do not move on until you can explain: what a neuron is, what an activation function does,

what gradient descent is moving toward, and what backpropagation calculates.

→ youtube.com/watch?v=aircAruvnKk

■ BUILD (90 MIN) — Single neuron from scratch

Build a single neuron in Python — NumPy only, no PyTorch yet:

def sigmoid(x): return 1 / (1 + np.exp(-x))

def neuron(inputs, weights, bias): return sigmoid(np.dot(inputs, weights) + bias)

def train(X, y, lr=0.1, epochs=1000): # gradient descent loop

Train it to learn the AND logic gate (0,0)→0, (0,1)→0, (1,0)→0, (1,1)→1

Print the loss every 100 epochs. Watch it decrease. That is learning.

✓ DONE WHEN: Single neuron trained on AND gate. Loss decreases over epochs. You can explain why.

DAY 044 · BACKPROPAGATION BY HAND

Derive the gradients before you automate them.

■ INDUSTRY (15 MIN)

Watch: 3Blue1Brown 'Backpropagation calculus' (12 min)

youtube.com/watch?v=tIeHLnjs5U8 — watch this before building today.

■ STUDY (90 MIN) — Backpropagation mathematics

For a 2-layer network, derive the gradient update rules on paper:

Loss = (y_pred - y_true)²

dLoss/dw = dLoss/dy_pred × dy_pred/dz × dz/dw

This is the chain rule. It is the entire foundation of deep learning.

If the math feels hard: that is normal. Spend the study block on it.

Ask Claude: 'explain backpropagation with a very simple numerical example'

■ BUILD (90 MIN) — 2-layer network from scratch

Extend your Day 43 neuron to a 2-layer network — still NumPy only:

Layer 1: 2 inputs → 4 hidden neurons

Layer 2: 4 hidden → 1 output

Implement forward pass and backward pass manually.

Train on XOR (which a single neuron cannot solve).

When it works, print the final weights and bias values.

✓ DONE WHEN: 2-layer network solves XOR. Forward and backward pass both implemented manually.

DAY 045 · PYTORCH PROPERLY

The framework that powers modern AI research.

■ INDUSTRY (15 MIN)

Search: 'why did PyTorch win over TensorFlow' — any developer discussion.

Understanding why tools won matters as much as learning to use them.

■ STUDY (90 MIN) — PyTorch fundamentals

Work through the official PyTorch Basics tutorial — all 7 sections.

Key concepts: tensors, autograd, nn.Module, DataLoader, training loop.

The autograd system does automatically what you did manually on Days 43–44.

Now that you did it manually, autograd will make sense.

→ pytorch.org/tutorials/beginner/basics/intro.html

■ BUILD (90 MIN) — Rebuild Day 44 in PyTorch

Rebuild your 2-layer XOR network using PyTorch nn.Module:

class XORNet(nn.Module):

def __init__(self): define layers with nn.Linear

def forward(self, x): define the forward pass

Use nn.BCELoss and torch.optim.Adam.

Compare: how many lines vs your manual implementation? What does PyTorch hide?

✓ DONE WHEN: XOR network rebuilt in PyTorch. Training loop runs. Loss decreases to near zero.

DAY 046 · FAST.AI — TOP DOWN LEARNING

Build a state-of-the-art model before understanding everything.

■ INDUSTRY (15 MIN)

Watch Jeremy Howard explain the fast.ai philosophy (5 min).

You are about to do something counterintuitive: build before fully understanding.

→ youtube.com — search 'Jeremy Howard fast.ai why top down learning works'

■ STUDY + BUILD (3 HRS COMBINED) — fast.ai Lesson 1

Go to course.fast.ai. Watch Lesson 1 completely.

Type every code example. Run every cell.

By end of Lesson 1 you will have trained an image classifier better than

anything that existed before 2012, in about 5 lines of code.

Deploy it to HuggingFace Spaces — Lesson 1 walks through this step by step.

You will have a live image classifier at a public URL by the end of today.

→ course.fast.ai

✓ DONE WHEN: Image classifier trained with fast.ai. Deployed to HuggingFace Spaces. Public URL live.

DAY 047 · CNNS — CONVOLUTIONAL NEURAL NETWORKS

How AI sees images.

■ INDUSTRY (15 MIN)

Search: 'how self-driving cars see the world computer vision'

CNNs are what make autonomous vision possible.

■ STUDY (90 MIN) — CNN architecture

fast.ai Lesson 2: deeper into how CNNs work.

Key concepts: convolutional filters, feature maps, pooling, receptive field.

A convolutional filter is a small matrix that slides over an image looking for patterns.

Early layers find edges. Middle layers find shapes. Deep layers find objects.

Read: 'CS231n CNN for Visual Recognition' overview — cs231n.github.io

■ BUILD (90 MIN) — CNN from scratch in PyTorch

Build a CNN using PyTorch nn.Conv2d — not fast.ai, not a library wrapper:

Conv2d(1, 32, 3) → ReLU → MaxPool2d(2)

Conv2d(32, 64, 3) → ReLU → MaxPool2d(2)

Flatten → Linear(64*5*5, 128) → ReLU → Linear(128, 10)

Train on MNIST (handwritten digits) — torchvision.datasets.MNIST

Target: above 98% accuracy. It should get there in under 10 epochs.

✓ DONE WHEN: CNN trained on MNIST. Above 98% accuracy. You can label each layer and explain what it does.

DAY 048 · TRAINING IMPROVEMENTS

The difference between a model that works and one that works well.

■ INDUSTRY (15 MIN)

Search: 'what is overfitting in machine learning simple explanation'

Overfitting is the single most common failure mode in all of ML.

■ STUDY (90 MIN) — Regularization and optimization

Key concepts: dropout, batch normalization, learning rate scheduling, data augmentation.

Dropout: randomly zero out neurons during training — prevents memorization.

Batch norm: normalize layer inputs — stabilizes and speeds up training.

Data augmentation: random flips, rotations, crops — more data without more data.

fast.ai Lesson 3 and 4 cover these techniques with practical examples.

■ BUILD (90 MIN) — Improve your Day 47 CNN

Add to your MNIST CNN:

nn.Dropout(0.25) after each pooling layer

nn.BatchNorm2d after each conv layer

Random rotation and flipping via torchvision.transforms

Learning rate scheduler: torch.optim.lr_scheduler.StepLR

Measure accuracy before and after each addition. Document the improvement.

✓ DONE WHEN: Improved CNN trained. Accuracy higher than Day 47. Impact of each technique documented.

DAY 049 · WEEK 1 CAPSTONE

Transfer learning — the skill that makes everything practical.

■ INDUSTRY (15 MIN)

Search: 'what is transfer learning simple explanation'

Transfer learning is why you don't need Google's compute budget to build great AI.

→ youtube.com — search 'Stanford CS231N transfer learning'

■ STUDY (60 MIN) — Transfer learning

Pre-trained models (ResNet, EfficientNet, ViT) have already learned features

from millions of images. Transfer learning repurposes those features for your task.

fast.ai makes this trivially easy. Understanding why it works makes you better.

Read: 'Transfer learning in computer vision' — pytorch.org/tutorials/beginner/transfer_learning_tutorial.html

■ BUILD (2 HRS) — Custom image classifier with transfer learning

Using fast.ai, build an image classifier for something you care about.

Examples: classify dog breeds, food types, plant diseases, car models.

Use a real dataset from Kaggle or collect your own images with DuckDuckGo.

Fine-tune a ResNet18 or EfficientNet model. Target above 90% accuracy.

Deploy to HuggingFace Spaces. This becomes portfolio project #3.

✓ DONE WHEN: Custom classifier deployed. Above 90% accuracy. Portfolio project #3 live at public URL.

WEEK 2 · DAYS 50–56 · RNNS + SEQUENCES + NVIDIA CERT PREP

DAY 050 · RNNS — RECURRENT NEURAL NETWORKS

AI that works with sequences and time.

■ INDUSTRY (15 MIN)

Search: 'how does Siri / Alexa understand speech' — the answer is RNNs and transformers.

■ STUDY (90 MIN) — RNN fundamentals

Read: Andrej Karpathy's 'The Unreasonable Effectiveness of Recurrent Neural Networks'

karpathy.github.io/2015/05/21/rnn-effectiveness/

Key concept: RNNs process sequences by maintaining a hidden state — memory.

The hidden state at step T contains information from steps 1 through T.

This is how language models understand context.

→ youtube.com — search 'Andrej Karpathy RNN effectiveness'

■ BUILD (90 MIN) — Character-level RNN

Build an RNN that generates text character by character:

Load a text file (a book, song lyrics, anything)

Encode characters as integers

Build: nn.Embedding → nn.RNN → nn.Linear

Train to predict the next character given the previous ones

Generate 200 characters of text from a seed

The output will be garbled but structured. That is the model learning.

✓ DONE WHEN: Character RNN trained. Generates text that resembles the training data's style.

DAY 051 · LSTMS — SOLVING THE VANISHING GRADIENT

Memory that actually works.

■ INDUSTRY (15 MIN)

Search: 'vanishing gradient problem explained simply'

This problem is why RNNs struggled and why LSTMs were invented.

■ STUDY (90 MIN) — LSTM architecture

Read Chris Olah's 'Understanding LSTMs' — one of the clearest ML blog posts ever written.

Key concepts: forget gate, input gate, output gate, cell state.

The cell state is the highway — information flows through it with minimal modification.

Gates are learned functions that control what information is kept or discarded.

→ colah.github.io/posts/2015-08-Understanding-LSTMs/

■ BUILD (90 MIN) — Upgrade to LSTM

Replace the nn.RNN in your Day 50 model with nn.LSTM.

Compare: train both on the same text for the same number of epochs.

Measure: loss after training, quality of generated text (subjectively).

The LSTM should produce more coherent text.

Document what changed and why in your log.

✓ DONE WHEN: LSTM model trained. Comparison with RNN documented. Generated text is more coherent.

DAY 052 · SEQUENCE MODELING PROJECT

Build something that uses sequences.

■ INDUSTRY (15 MIN)

Search: 'how does Google Translate work neural machine translation'

Translation is sequence-to-sequence — the architecture that led to transformers.

■ STUDY (60 MIN) — Sequence modeling applications

Sentiment analysis: classify text as positive, negative, or neutral.

Time series forecasting: predict future values from past values.

Named entity recognition: identify names, places, organizations in text.

All three use the same underlying sequence architecture.

Read: pytorch.org/tutorials/intermediate/seq2seq_translation_tutorial.html

■ BUILD (2 HRS) — Sentiment analyzer

Build a sentiment analysis model using an LSTM:

Dataset: IMDB movie reviews — torchtext.datasets.IMDB or Kaggle

Architecture: Embedding → LSTM → Linear → Sigmoid

Target: above 85% accuracy on test set

Deploy to HuggingFace — user types a review, gets positive/negative prediction.

Portfolio project #4.

✓ DONE WHEN: Sentiment analyzer deployed. Above 85% accuracy. Portfolio project #4 live.

DAY 053 · GPU TRAINING + COMPUTE

Understanding the hardware that powers AI.

■ INDUSTRY (15 MIN)

Search: 'why does AI need GPUs explained simply'

This directly connects to the NVIDIA cert you're preparing for.

■ STUDY (90 MIN) — GPU fundamentals + NVIDIA cert prep start

Go to NVIDIA's Deep Learning Institute. Create a free account.

Find and start: 'Fundamentals of Deep Learning' course.

Key concepts: CUDA, tensor cores, GPU memory, batch size vs learning rate.

Understand: why parallelism makes GPUs fast for matrix math.

If you have no local GPU: use Google Colab (colab.google.com) — free GPU access.

→ learn.nvidia.com/en-us/training/self-paced-courses

■ BUILD (90 MIN) — GPU vs CPU training comparison

Take your Day 47 MNIST CNN. Train it twice:

Once on CPU: device = torch.device('cpu')

Once on GPU: device = torch.device('cuda') or 'mps' for Mac

Measure exact training time for both. Print the speedup ratio.

If no local GPU: run this on Google Colab — free T4 GPU.

✓ DONE WHEN: GPU training comparison complete. Speedup measured and documented. NVIDIA DLI started.

DAY 054 · NVIDIA CERT PREP

Deep dive on AI infrastructure and operations.

■ INDUSTRY (15 MIN)

Search: 'NVIDIA H100 GPU explained simply — why it matters for AI'

The hardware landscape is part of the NVIDIA cert. Understand it conceptually.

■ STUDY (2 HRS) — NVIDIA DLI + cert prep

Continue NVIDIA DLI 'Fundamentals of Deep Learning' course.

Complete as much as possible today — this is the core exam material.

Key topics for the NCA-AIIO exam:

AI hardware: GPU architecture, tensor cores, NVLink, NVMe

AI infrastructure: data centers, cooling, networking, storage

AI operations: monitoring, deployment, model serving, scaling

MLOps fundamentals: CI/CD for ML, experiment tracking, model versioning

→ learn.nvidia.com/en-us/training/self-paced-courses

→ nvidia.com/en-us/training/certification

■ BUILD (60 MIN) — Training pipeline with best practices

Add professional practices to your MNIST CNN:

Checkpoint saving: save model every 5 epochs

Early stopping: stop if validation loss doesn't improve for 3 epochs

TensorBoard logging: pip install tensorboard, log loss and accuracy

Reproducibility: set random seeds at the start

These are what MLOps engineers care about.

✓ DONE WHEN: NVIDIA DLI course mostly complete. Professional training pipeline with checkpointing and logging.

DAY 055 · FAST.AI DEEP DIVE

Modern deep learning techniques.

■ INDUSTRY (15 MIN)

Search: 'Jeremy Howard fast.ai impact on AI democratization'

fast.ai has trained tens of thousands of practitioners. You are one of them now.

■ STUDY + BUILD (3 HRS) — fast.ai Lessons 5–7

Lesson 5: NLP with fast.ai — text classification and language models.

Lesson 6: tabular data and collaborative filtering — recommender systems.

Lesson 7: diffusion models (fascinating preview of generative AI).

Pick the lesson most relevant to what you want to build. Do it completely.

Apply the technique to a dataset you find personally interesting.

→ course.fast.ai

✓ DONE WHEN: One fast.ai advanced lesson complete. Technique applied to personal dataset.

DAY 056 · WEEK 2 CAPSTONE

A complete deep learning project.

■ INDUSTRY (15 MIN)

Search: 'state of computer vision 2025 what's possible'

After two weeks of deep learning, you'll understand most of what you read.

■ STUDY (30 MIN) — Week 2 self test

Without looking at code: sketch the architecture of a CNN on paper.

Label each layer. Describe what it does. Describe what goes in and what comes out.

If you can do it: you understand CNNs. If not: spend the 30 min filling that gap.

■ BUILD (2.5 HRS) — Week 2 Project

Build one of the following (your choice):

A. Real-time object detection app using a pre-trained YOLO model

B. A style transfer app — apply the style of one image to another

C. A medical image classifier (chest X-ray disease detection — Kaggle dataset)

D. Something entirely of your choosing that uses deep learning

Deploy it. Write a 200-word technical description. Portfolio project #5.

✓ DONE WHEN: Week 2 deep learning project deployed. 200-word writeup committed to GitHub.

WEEK 3 · DAYS 57–63 · NVIDIA CERT + PHASE COMPLETION

DAY 057 · CERT SPRINT DAY 1

Focused exam preparation.

■ INDUSTRY (15 MIN)

Search: 'NVIDIA AI infrastructure data center explained'

Understanding the infrastructure context will help the cert material click.

■ STUDY (2 HRS) — NCA-AIIO exam prep

Review the official NCA-AIIO exam objectives at the link above.

Key domains: AI computing fundamentals, deep learning frameworks,

AI hardware and software stack, deploying AI solutions, AI infrastructure.

If you haven't finished NVIDIA DLI Fundamentals of Deep Learning: finish it today.

Take any available practice questions.

→ nvidia.com/en-us/training/certification

■ BUILD (60 MIN) — Model serving basics

Learn to serve a PyTorch model as an API:

pip install fastapi uvicorn

Load your trained model

Create a POST endpoint that accepts an image and returns a prediction

Run it locally: uvicorn app:app --reload

This is how production AI systems work.

✓ DONE WHEN: NCA-AIIO exam objectives reviewed. Model serving API running locally.

DAY 058 · CERT SPRINT DAY 2

Fill the gaps.

■ INDUSTRY (15 MIN)

Search: 'what is MLOps and why does it matter' — any clear explainer.

■ STUDY (2 HRS) — Weak areas

Identify your weakest NCA-AIIO domain from yesterday's review.

Spend the full 2 hours on that domain specifically.

Use NVIDIA's own documentation and the DLI course material.

Ask Claude to quiz you on specific concepts:

'Quiz me on GPU architecture for the NVIDIA NCA-AIIO certification'

■ BUILD (60 MIN) — Experiment tracking

Add MLflow or Weights & Biases to your training pipeline:

pip install mlflow

Log: hyperparameters, metrics per epoch, final model artifact

View the dashboard in your browser

Experiment tracking is what professional ML teams use to manage training runs.

✓ DONE WHEN: Weakest cert domain studied. Experiment tracking running and logging your models.

DAY 059 · CERT SPRINT DAY 3

Practice exam and final review.

■ INDUSTRY (15 MIN)

Search: 'NVIDIA DGX systems explained simply'

Understanding what NVIDIA builds helps the cert material land.

■ STUDY (2 HRS) — Final cert review

Take the full practice assessment if available.

Review every topic you got wrong. Read the explanation for each.

Key facts to have locked: GPU vs CPU differences, tensor core function,

CUDA programming model, model parallelism vs data parallelism,

inference optimization techniques (quantization, pruning, distillation)

Book your exam for Day 62 or 63.

→ nvidia.com/en-us/training/certification

■ BUILD (60 MIN) — Quantization demo

Quantize your trained MNIST CNN to INT8:

import torch.quantization

quantized_model = torch.quantization.quantize_dynamic(model, ...)

Compare: model file size before and after. Inference speed before and after.

Quantization is one of the key inference optimization techniques on the exam.

✓ DONE WHEN: Practice exam taken. Exam booked for Day 62–63. Model quantization demonstrated.

DAY 060 · PORTFOLIO POLISH

Your deep learning work presented properly.

■ INDUSTRY (15 MIN)

Search: 'how to present machine learning projects in interviews'

■ STUDY (30 MIN)

Read 3 HuggingFace model cards from popular models.

A model card is a document that explains what a model does, how it was trained,

its limitations, and its intended use. This is what you write for your models.

■ BUILD (2.5 HRS) — Document Phase 3 work

For each deployed model from Phase 3:

Write a model card (see HuggingFace model card template)

Add it to your HuggingFace Space README

Write a 3-sentence description for your GitHub portfolio README

Update your nullvector-journey README with Phase 3 section:

Skills learned, models deployed, links to all 3 live demos.

✓ DONE WHEN: All Phase 3 models have model cards. GitHub README updated with Phase 3 section.

DAY 061 · FINAL BUILD DAY

One more project before the exam.

■ INDUSTRY (15 MIN)

Search: 'what can neural networks do in 2025 that surprised researchers'

■ STUDY (45 MIN) — Final exam review

One final pass through your weakest NCA-AIIO domain.

Write out key facts from memory. Check yourself against the exam objectives.

■ BUILD (2.25 HRS) — Phase 3 Capstone

Build the most impressive deep learning project you can in one session.

Suggestions:

A CNN that classifies your own photos into custom categories

A text generator trained on a book or dataset you love

A model that detects objects in a live webcam feed

Deploy it. Write a technical summary. This is the showpiece of Phase 3.

✓ DONE WHEN: Phase 3 capstone deployed. Technical summary written. Exam prep complete.

DAY 062 · NVIDIA NCA-AIIO EXAM

The certification milestone.

■ EXAM DAY

Sit the NVIDIA Certified Associate: AI Infrastructure & Operations exam.

If you pass: screenshot the result. Add to GitHub README and LinkedIn.

If you need to retake: that is fine. Book again within 48 hours.

Study your weakest domain specifically. You are close.

→ nvidia.com/en-us/training/certification

■ BUILD (90 MIN) — After the exam

Write your Phase 3 complete log entry — your longest one yet.

Regardless of exam result: document what you built this phase.

Count: models trained, apps deployed, lines of code written.

Update your GitHub README. Phase 4 starts Day 64.

✓ DONE WHEN: NCA-AIIO exam attempted. Result logged. Phase 3 complete log written.

DAY 063 · PHASE 3 COMPLETE

Deep learning is no longer a black box.

■ REVIEW (60 MIN)

Look back at Day 43. You watched a 19-minute video and built a neuron.

Look at Day 61. You built a production-quality deep learning system.

Write down 10 specific things you understand now that you didn't 21 days ago.

Keep that list. You will need it when Phase 4 gets hard.

■ PHASE 4 PREP (90 MIN) — Set yourself up for Transformers

Download and start reading 'Attention Is All You Need' — the transformer paper.

You don't need to understand it yet. Read it once. Note what confuses you.

Install: pip install transformers datasets tokenizers

Verify: import torch; import transformers — both should work without errors.

Phase 4 starts tomorrow. It is the most important phase in the entire challenge.

→ arxiv.org/abs/1706.03762

✓ DONE WHEN: Phase 3 complete. Transformer paper downloaded. Phase 4 libraries installed and verified.

DAY 63 MILESTONE · NVIDIA CERTIFIED ASSOCIATE: AI INFRASTRUCTURE & OPERATIONS

NVIDIA NCA-AIIO — Your Third Credential

You have now built neural networks from scratch, trained CNNs on image data, built RNNs and LSTMs for sequence modeling, used transfer learning to build production-quality classifiers, and earned a globally recognized AI hardware credential. Phase 4 — TRANSFORMER — begins Day 64. You will read the paper that changed the entire field. Then you will implement what it describes.