ChatGPT

AI & Machine Learning Learning Plan (12–18 months)

This plan builds a deep foundation in AI/ML and specializes in large language models (LLMs). We start with math and classical ML, then progress through deep learning and NLP, culminating in LLMs and deployment. We balance theory (textbooks and papers) with practical coding (courses, videos, projects) at ~5–6 hrs/week. For example, Andrej Karpathy advises that “language models are an excellent place to learn deep learning” because skills transfer broadly [1]. Key resources include Karpathy’s *Zero to Hero* videos, fast.ai’s hands-on courses [2] [3] [4], Stanford’s CS224N (NLP with Deep Learning) [5], Hugging Face’s free LLM course [6], and landmark papers like BERT [7] and the GPT-3/4 reports [8] [9]. Each quarter ends with a capstone project to integrate learning.

Quarter 1 (Months 1–3): Foundations – Math & ML Basics

- **Month 1: Math Refresher & Intro to ML.**
- **Week 1:** *Linear algebra fundamentals.* Study vectors/matrices. Read *Mathematics for Machine Learning* (Ch.2) or similar to refresh linear algebra [10]. Watch 3Blue1Brown’s “Essence of Linear Algebra” videos (visual intuition). Practice with Python/NumPy: implement matrix operations, solve linear equations.
- **Week 2:** *More linear algebra.* Cover matrix decomposition (SVD, eigenvalues). Continue textbook or MIT OpenCourseWare linear algebra notes. Solve exercises implementing principal component analysis (PCA) from scratch.
- **Week 3:** *Probability and stats basics.* Review probability distributions, expectation, variance. Read probability sections of *Mathematics for ML* or Khan Academy stats modules. Solve probability problems (e.g. Bayes rule) and do simple simulations in Python.
- **Week 4:** *Intro to Machine Learning concepts.* Start fast.ai’s **“Intro to Machine Learning for Coders”** (Jeremy Howard) which uses Python/pandas/sklearn [2]. Lesson 0–1 covers decision trees and random forests [3]. **Assignment:** Use a simple dataset (e.g. Kaggle Titanic) to train a random forest or logistic regression and interpret results.
- **Month 2: Neural Networks Fundamentals.**
- **Week 5:** *Perceptrons and MLPs.* Learn basic neural networks. Watch **Karpathy Lecture 1 (“backprop and micrograd”)** [11] to see a step-by-step build of a small neural net. Read about simple feedforward networks (e.g. Goodfellow *Deep Learning*, Ch.6). **Exercise:** Implement a 2-layer neural net (e.g. on MNIST digits) using PyTorch or NumPy; observe learning.
- **Week 6:** *Training and optimization.* Study stochastic gradient descent (SGD) and its variants. Read fast.ai’s notes or Goodfellow (Ch.8) on optimization. Watch parts of fast.ai’s Practical DL for explanation of training loops [12]. **Exercise:** Code a training loop by hand (using PyTorch’s Tensor API) and tune learning rate, batch size.
- **Week 7:** *Regularization and tuning.* Learn about overfitting and techniques (dropout, weight decay). Continue fast.ai lessons or Karpathy Lecture 3 on *activations and batchnorm* [13]. **Exercise:** Train a deeper network (e.g. 3+ layers) on CIFAR-10; add dropout/BatchNorm and compare performance.

- **Week 8:** *Convolutional Neural Networks (CNNs).* Get a high-level intro to CNNs (e.g. Stanford CS231n lecture or Goodfellow Ch.9 [14]). Experiment with PyTorch's `Conv2d` layers: build a simple CNN for image classification. **Exercise:** Fine-tune a pretrained CNN (e.g. ResNet on CIFAR-10) and analyze feature maps.
- **Month 3: Advanced ML & Integration.**
- **Week 9:** *Recurrent nets and sequence models (intro).* Study RNN/LSTM basics from resources like Stanford CS231n or blog posts. Optionally watch Karpathy's talk "Unreasonable Effectiveness of RNNs." Do a toy task: character-level RNN for text generation (e.g. on a short book).
- **Week 10:** *Unsupervised learning overview.* Briefly learn about clustering (k-means) and dimensionality reduction (PCA continuation). See chapters from *Mathematics for ML* or fast.ai on these topics. **Exercise:** Apply k-means or PCA on a sample dataset (MNIST or images).
- **Week 11:** *Mathematical review.* Reinforce any weak areas: revisit calculus (gradients, chain rule) and probability. Use Khan Academy or the *Mathematics for ML* book sections as needed [10].
- **Week 12:** *Capstone Project Prep.* Choose a data science project that ties together Q1 topics. For example, use neural networks to classify images or tabular data end-to-end. Plan data preprocessing, model choice, and evaluation metrics. Set up a GitHub repo for code.
- **Quarter 1 Capstone:** *Mini ML Project.* Complete the chosen project (e.g. an MNIST or CIFAR-10 image classifier, or Titanic survival predictor) and write a short report (architecture, training, results). Aim to deploy it (e.g. a simple web app or notebook) so it can be showcased.

Quarter 2 (Months 4–6): Deep Learning Mastery

- **Month 4: Deep Nets and PyTorch.**
- **Week 13:** *Deep learning from scratch.* Continue Karpathy's series. Watch **Lecture 2 ("language modeling: bigram")** [15] to learn how to implement a simple character-level model and gain intuition for data pipelines (embedding, loss, sampling). **Exercise:** Build a small word or char N-gram model in PyTorch.
- **Week 14:** *Advanced backprop & autograd.* Watch **Karpathy Lecture 4 ("Backprop Ninja")** [16] for a manual backprop walkthrough through a multi-layer MLP. Understand PyTorch's `autograd`. **Exercise:** Manually implement one training step (forward + backward) on a tiny network without high-level libraries.
- **Week 15:** *Deep architectures.* Learn about modern architectures: residual networks (ResNet), transformers preview. Read summaries of ResNet or watch fast.ai Lesson on **transfer learning and data augmentation** [4]. **Exercise:** Use PyTorch's `resnet18` pretrained model on a new dataset (e.g. cats vs dogs).
- **Week 16:** *Model evaluation & debugging.* Study techniques for diagnosing models (learning curves, confusion matrices). Use tensorboard or plotting. Read about hyperparameter tuning (grid search, learning rate schedules). **Exercise:** Take a trained network, vary hyperparameters (batch size, LR) and plot validation curves.
- **Month 5: Sequence Models & Attention.**
- **Week 17:** *RNN/LSTM deep dive.* Solidify sequence models: implement an LSTM classifier on text. Use tutorials (PyTorch official or Stanford CS224N notes). Observe vanishing gradients and why LSTM/GRU help.
- **Week 18:** *Attention and Transformers intro.* Read "Attention Is All You Need" (Vaswani et al. 2017) or a tutorial. Watch an explainer video on Transformers. **Exercise:** Use PyTorch's `nn.MultiheadAttention` or Hugging Face's `nn.Transformer` module on a toy task (e.g. copy a sequence).

- **Week 19:** *Transformers in practice.* Start Hugging Face's **Transformers** library tutorials. Load a pretrained model (e.g. a small transformer) and run inference on custom text. **Exercise:** Use a Hugging Face pipeline for text generation or translation.
- **Week 20:** *Mid-Quarter Check-In.* Review concepts so far. If needed, revisit challenging topics (e.g. re-study backprop or math gaps). Prepare for end-of-quarter project.
- **Month 6: NLP Fundamentals & LLM Tools.**
- **Week 21:** *NLP pipeline basics.* Learn tokenization, embeddings, and common tasks (text classification, NER). Use Hugging Face tokenizers and Datasets libraries. **Exercise:** Fine-tune a small transformer (e.g. DistilBERT) on a text classification dataset (e.g. IMDb reviews) using Transformers.
- **Week 22:** *Stanford CS224N (optional).* Watch parts of CS224N lectures on sentiment analysis or parsing. This course covers both classical and neural NLP [5] . **Exercise:** Implement a simple sequence classification or tagging model using PyTorch (e.g. LSTM-based).
- **Week 23:** *Large Language Models overview.* Read about BERT: see Hugging Face's "BERT 101" post [7] . Understand its encoder structure and tasks (masked LM, NSP). **Exercise:** Use a pretrained BERT to extract embeddings or do fill-mask on sentences.
- **Week 24:** *Introduction to Hugging Face LLM course.* Begin the Hugging Face **LLM Course** [6] (cover Setup and Transformer basics). By end of this week, you should know how to load models from the Hub, fine-tune on data, and share results [17] .
- **Quarter 2 Capstone:** *Deep Learning Project.* Build and train a non-trivial deep learning model. For instance, implement a CNN from scratch (following a PyTorch tutorial) on the CIFAR-10 dataset, or fine-tune BERT on a custom text classification task. Document your code, results, and any insights (share via GitHub).

Quarter 3 (Months 7–9): Advanced NLP and LLMs

- **Month 7: Transformer Models.**
- **Week 25:** *Transformer architecture.* Deepen understanding: revisit the Transformer encoder/decoder blocks. Read transformer-friendly blog posts. **Exercise:** Build a mini-transformer in PyTorch using `nn.Transformer`; try self-attention on toy data.
- **Week 26:** *GPT and autoregressive LMs.* Study how GPT-type models (Decoder-only transformers) work. Read about GPT-2/GPT-3 design (e.g. OpenAI blog). **Exercise:** Use Hugging Face's GPT-2 model to generate text (e.g. story continuation) and experiment with temperature/length.
- **Week 27:** *Fine-tuning and prompts.* Learn prompt engineering and few-shot techniques. Fine-tune a small GPT-2 or T5 model on a niche dataset (poetry, code snippets, etc.). **Exercise:** Deploy your fine-tuned model locally to see its outputs on new prompts.
- **Week 28:** *Hugging Face LLM course (continued).* Complete advanced modules (chapters 10–12 cover fine-tuning and reasoning models) [18] . Practice sharing a model on the Hub or building a demo with Gradio.
- **Month 8: LLM Applications.**
- **Week 29:** *Question Answering and Summarization.* Study transformer applications: QA and summarization. Follow tutorials (e.g. HF pipelines). **Exercise:** Fine-tune a model on a QA dataset (SQuAD) or use a pretrained summarizer on articles.
- **Week 30:** *Dialogue and Conversational AI.* Explore chatbots using LLMs. Read about frameworks like Rasa or use GPT dialogue examples. **Exercise:** Build a simple chatbot by combining a transformer with rule-based logic, or use GPT-2 with conversational prompts.

- **Week 31:** *Limitations and Ethics.* Learn about bias and limitations of LLMs (e.g. Hugging Face course section on bias). Read case studies of LLM misuse. **Exercise:** Analyze a deployed model for common failure modes (e.g. bias, hallucination) and suggest mitigations.
- **Week 32:** *Case Study: BERT and Beyond.* Read the original BERT paper or summaries. Compare BERT vs GPT approaches. **Exercise:** Write a short report comparing encoder (BERT) vs decoder (GPT) models for a given task (e.g. NER vs text generation).
- **Month 9: Deployment and Scaling.**
- **Week 33:** *Model Deployment Basics.* Learn how to deploy models as services. Read fast.ai’s guidance on creating web apps [4] (e.g. using Gradio/Streamlit) and Hugging Face Spaces. **Exercise:** Deploy one of your models (from previous weeks) on a free platform so others can try it (e.g. a web demo).
- **Week 34:** *ML Infrastructure.* Understand containerization (Docker) and cloud services for ML (AWS SageMaker/GCP AI Platform). **Exercise:** Dockerize your app or set up a simple AWS/GCP inference endpoint for your model.
- **Week 35:** *Case Study: GPT-3 and GPT-4.* Read about GPT-3 (175B parameters) [8] and GPT-4 capabilities [9]. **Exercise:** Try using an API (if accessible) for GPT-3/GPT-4 to solve a task (e.g. translation, coding). Analyze its performance compared to smaller models.
- **Week 36:** *Final Review.* Go over any remaining weak spots. Prepare a comprehensive final project plan and collect resources.
- **Quarter 3 Capstone:** *LLM Application Project.* Create an end-to-end NLP/LLM system. Examples: a small question-answering chatbot, a custom text summarizer, or a content generator. Use transfer learning (fine-tune a transformer) and deploy it (e.g. via Hugging Face Space). Document your architecture, training process, and demo usage.

Quarter 4 (Months 10–12): Mastery and Deployment

- **Month 10: Cutting-Edge LLMs.**
- **Week 37:** *Large-Scale Models.* Study the evolution of LLMs. Read the GPT-3 technical paper summary [8] and the GPT-4 report [9]. **Exercise:** Use a modern open-source LLM (e.g. Llama or GPT-J) on a GPU; compare speed/memory trade-offs.
- **Week 38:** *Alignment and RLHF.* Learn the basics of Reinforcement Learning from Human Feedback (used in ChatGPT). Read blog posts from OpenAI about InstructGPT/GPT-4 training. **Exercise:** Experiment with human-in-the-loop training (e.g. adjust prompts or ratings) to refine a model’s outputs.
- **Week 39:** *Multimodality (optional).* If interested and time allows, explore vision-language models (e.g. CLIP, DALL·E). **Exercise:** Use Hugging Face's CLIP or a vision transformer on an image-text task.
- **Week 40:** *Cutting-edge Research.* Read recent LLM research (e.g. chain-of-thought prompting, retrieval-augmented generation) to stay current. **Exercise:** Implement a chain-of-thought prompt pattern for a reasoning task and compare results.
- **Month 11: MLOps and Production.**
- **Week 41:** *MLOps Overview.* Study best practices for ML in production (monitoring, versioning data/models). Learn about tools like MLflow or Weights & Biases. **Exercise:** Set up basic experiment tracking (e.g. log metrics/hyperparams) for your ongoing projects.
- **Week 42:** *Scalable Serving.* Learn about batching and optimizing inference (e.g. ONNX, TensorRT). **Exercise:** Convert one of your PyTorch models to ONNX and measure inference speed.
- **Week 43:** *Security and Privacy.* Understand issues like data privacy, model robustness to adversarial attacks. Read summaries of adversarial examples. **Exercise:** Try a simple adversarial example attack on an image classifier or text model to see how it fails.

- **Week 44:** *Tutorials and Documentation.* Practice writing clear docs. Create a README or blog post for your project. **Exercise:** Draft a walkthrough (e.g. on GitHub Pages) explaining your final project's setup and usage.
- **Month 12: Final Synthesis.**
- **Week 45:** *Review & Fill Gaps.* Review all key topics; revisit any forgotten areas (e.g. specific math or algorithms).
- **Week 46:** *Community Engagement.* Join ML forums or Kaggle competitions to test your skills. **Exercise:** Participate in a Kaggle challenge or answer questions on a community forum.
- **Week 47:** *Future Learning Plan.* Research advanced courses or areas (e.g. reinforcement learning, computer vision, theory). Update your learning roadmap based on interests.
- **Week 48:** *Capstone Preparation.* Finalize the project scope and ensure all prerequisites (data, codebase) are ready.
- **Quarter 4 Capstone:** *Full Deployment Project.* Undertake a comprehensive project. For example, build a domain-specific LLM-powered service (e.g. a legal document summarizer or a code assistant). Fine-tune or use an API, integrate any needed ML components, and deploy as a robust application (with a web interface or API). Prepare a thorough report and demonstration video of your system.

Beyond Year 1 (Optional 13–18 months)

If extending to 15–18 months, consider delving into specialized or emerging topics: reinforcement learning (e.g. David Silver's RL course), graph neural networks, advanced optimization/geometry, or contributing to open-source ML projects. You can also undertake larger research-scale projects (e.g. join an ML research lab or contribute to papers on new LLM techniques). Continue following AI conference proceedings (NeurIPS, ICML, ACL) to stay at the cutting edge.

Sources: Recommendations are based on authoritative course descriptions and expert advice [1] [2] [4] [6] [7] [8] [9] [5], among others. Each resource cited provides detailed coverage of the topic indicated.

[1] [11] [13] [15] [16] Neural Networks: Zero To Hero
https://karpathy.ai/zero-to-hero.html

[2] [3] Introduction to Machine Learning for Coders: Launch – fast.ai
https://www.fast.ai/posts/2018-09-26-ml-launch.html

[4] [12] Practical Deep Learning for Coders 2022 – fast.ai
https://www.fast.ai/posts/2022-07-21-dl-coders-22.html

[5] Stanford CS 224N | Natural Language Processing with Deep Learning
https://web.stanford.edu/class/cs224n/

[6] [17] [18] Introduction - Hugging Face LLM Course
https://huggingface.co/learn/llm-course/chapter1/1

[7] BERT 101 - State Of The Art NLP Model Explained
https://huggingface.co/blog/bert-101

[8] OpenAI Presents GPT-3, a 175 Billion Parameters Language Model | NVIDIA Technical Blog
https://developer.nvidia.com/blog/openai-presents-gpt-3-a-175-billion-parameters-language-model/

[9] [2303.08774] GPT-4 Technical Report
https://arxiv.org/abs/2303.08774

[10] Mathematics for Machine Learning | Companion webpage to the book “Mathematics for Machine Learning”. Copyright 2020 by Marc Peter Deisenroth, A. Aldo Faisal, and Cheng Soon Ong. Published by Cambridge University Press.
https://mml-book.github.io/

[14] Deep Learning
https://www.deeplearningbook.org/