AI Deep Dive Pathway for Vihaan

1 Personalized Learning Approach

This course is designed to teach Vihaan AI skills he will be able to use in whatever domain interests him. Since he has already completed the demo session, he will not require another initial skills assessment.

- Build on existing game development knowledge.
- Use project-based learning with immediate visual feedback.
- Have no theoretical lectures. Everything will be hands on and he will be able to immediately see what he's building so that he remains interested.
- Develop troubleshooting skills through guided problem-solving.

2 Course Structure & Methodology

Learning Pathway

Weekly Sessions:

- 60 minute hands-on coding sessions
- Flexible scheduling, at least once a week **recommended**
- Focused on an optimal balance between the student's questions and the direction of this syllabus

Homework Philosophy:

- Purpose-driven mini-projects (not exercises)
- Designed to reinforce concepts over short challenges on throughout the week
- Optional extension challenges for deeper exploration
- **15-minute complementary** midweek homework review session

Mathematical/Theoretical Foundation:

- Contextual introduction to vectors/matrices through game physics
- Exponents/logarithms in probability systems
- Just-in-time learning of required concepts. No boring lectures!

3 Core AI Concepts

3.1 Neural Network Fundamentals

- **Convolutional Neural Networks (CNNs):**
  - Filters and feature extraction
  - Pooling operations
  - Architecture patterns (LeNet, ResNet)
  - Image recognition applications
- **Project:** Image classifier for game assets

3.2 Mathematical Foundation

- **Vectors & Matrices:**
  - Operations: dot product, cross product
  - Geometric interpretations
  - Matrix transformations
- **Tensors:** Higher-dimensional data representation
- **Project:** Implement vector operations from scratch

3.3 Transformer Architectures

- **Attention Mechanism:**
  - Scaled dot-product attention
  - Multi-head attention
  - Positional encoding
- **GPT Architecture:**
  - Decoder-only structure
  - Layer normalization
  - Feed-forward networks
- **Project:** Visualize attention patterns

3.4 Build Your Own Model

- **From Scratch:**
  - Embedding layers
  - Self-attention implementation
  - Position-wise FFNs
- **Using PyTorch:**

```
class MiniGPT(nn.Module):
    def __init__(self, vocab_size, d_model):
        super().__init__()
        self.embed = nn.Embedding(vocab_size, d_model)
        self.blocks = nn.ModuleList([
            TransformerBlock(d_model) for _ in range(4)
        ])
        self.ln_f = nn.LayerNorm(d_model)
        self.head = nn.Linear(d_model, vocab_size)
```

- **Project:** Character-level language model

3.5 Prompt Engineering

- **Key Parameters:**
  - Temperature: Controlling randomness
  - Top-p/top-k sampling
  - Frequency/presence penalties
- **Advanced Techniques:**

  - Chain-of-thought prompting
  - Few-shot learning
  - Template engineering
- **Project:** Build a prompt optimizer

3.6 Vector Databases

- **Core Concepts:**
  - Embedding vectors
  - Similarity metrics (cosine, Euclidean)
  - Approximate nearest neighbors (ANN)
- **Applications:**
  - Long-term memory for AI agents
  - Semantic search
  - Contextual retrieval
- **Project:** Knowledge retrieval system

3.7 Fine-Tuning & Baseline Models

- **Why Baseline Models:**
  - Computational efficiency
  - Domain specialization
  - Privacy/security control
- **Methods:**
  - Full fine-tuning
  - Parameter-efficient tuning (LoRA, prefix-tuning)
  - RLHF (Reinforcement Learning from Human Feedback)
- **Project:** Fine-tune model on custom dataset

3.8 AI Agents

- **Architecture:**
  - Perception modules
  - Memory systems
  - Action planning
  - Self-reflection
- **Implementation:**
  - ReAct framework
  - Tool usage
  - Multi-agent systems
- **Final Project:** Game-playing AI agent

4 Learning Resources

- **Math:** 3Blue1Brown Linear Algebra series
- **Neural Networks:** Karpathy's NN Zero to Hero
- **Transformers:** The Annotated Transformer
- **Vector DBs:** Pinecone documentation
- **Fine-Tuning:** Hugging Face PEFT examples

5 Progression Logic

Concept	Why Next?
CNNs	Visual pattern recognition foundation
Vector Math	Language is geometric space
Transformers	Attention as information routing
Build Your Own	Understand model internals
Prompt Engineering	Practical interface to models
Vector DBs	External memory for models
Fine-Tuning	Customize model behavior
Agents	Autonomous AI systems