



Huruf Vision

Arabic Alphabet through Computer Vision

Difficulty: Beginner / Intermediate

Language: Python

Framework: PyTorch

Type: Computer Vision

Summary: Build a computer vision model that recognizes handwritten Arabic letters. Learn to load image datasets, design convolutional neural networks, train models with PyTorch, and evaluate their performance on unseen data.

Version: 1.0

Introduction

Welcome to the beginning.

This is the first project of the LEAD-AI Club journey. We aren't just here to watch tutorials, we are here to build. We chose Handwritten Arabic Character Recognition because it is the perfect playground to learn the fundamentals of Deep Learning without getting drowned in complexity.

This project is your "Hello World" to Artificial Intelligence. It is designed to take you from knowing zero Python to building a Neural Network that can "see." There are no hand holding steps here just a goal, the tools, and the resources. How you solve it is up to your team.

Objectives

By the end of this deep dive, you will have demystified the "Black Box" of AI. You will understand:

- **The Workflow:** How to take raw data and turn it into intelligence
- **Tensors:** The mathematical language of AI.
- **CNNs:** The architecture used in Self-Driving Cars and FaceID.
- **PyTorch:** The framework used by researchers at Meta, OpenAI, and Tesla.

General Instructions

- **Environment:** We recommend Google Colab (Free GPUs) or a local Anaconda environment.
- **Language:** Python 3.10+
- **Library:** PyTorch.
- **Allowed Helpers:** pandas, numpy, matplotlib, torchvision, PIL.
- **The Data:** The dataset (Images and CSV labels) will be provided in the shared repository.

The Project

Your goal is simple but powerful: Build an AI model that accepts an image of a handwritten Arabic letter and correctly identifies which letter it is.

You are free to structure your code however you like, but a successful project usually follows these four pillars. You will need to study and implement each one:

1. Data Ingestion

An AI model cannot learn if it cannot eat. You need to figure out how to load the images, convert them into numbers (Tensors), and feed them to the model in batches.

**Keywords to research:**

`torch.utils.data.Dataset` | `DataLoader` | Normalization.

2. The Architecture

You need to design the "brain" structure. For images, we use Convolutional Neural Networks (CNNs). You will need to stack layers together to extract features from the pixels.

**Keywords to research:**

`nn.Module` | `Conv2d` | `MaxPool2d` | `ReLU` | `Linear`

3. The Training Loop

This is where the learning happens. You must write a loop that shows the model images, checks its guess, calculates the error, and updates the brain to reduce that error.

**Keywords to research:**

Forward Pass | Loss Function (CrossEntropy) | Backpropagation |
Optimizer (Adam)

4. Validation

A model that memorizes the answers is useless. You must prove your model works on data it has never seen before.

**Keywords to research:**

Accuracy | Confusion Matrix | Train/Test Split

Guide And Resources

Phase A: Python and Tensors

If you can't manipulate matrices, you can't do AI.

Watch: [Python for Data Science \(First 2 hours\)](#)

Read: [PyTorch Basics: Tensors](#)

Read: [Deep Learning with PyTorch: A 60 Minute Blitz](#)

Phase B: Understanding the "Brain"

Don't just copy code. Understand how a computer "sees".

Watch: [But what is a Neural Network? \(3Blue1Brown\)](#)

Watch: [But what is a Convolution? \(3Blue1Brown\)](#)

Phase C: Implementation (PyTorch)

Read the F***** Manuel

- **Guide:** [Writing Custom Datasets in PyTorch](#)
- **Guide:** [Building the Neural Network Class](#)
- **Guide:** [The Training Loop Explained](#)



these resources will be provided on the resources channel on the LEAD-Ai discord server, don't hesitate to share your own too with the community