CLIP ViT Zero-Shot Image Classification — Detailed Beginner Guide

1. Introduction

CLIP (Contrastive Language–Image Pretraining) is a multimodal model by OpenAI that learns to link images and text. Unlike traditional models that require training on labeled datasets, CLIP performs zero-shot classification—meaning it predicts classes it was never trained on.

2. Zero-Shot Classification Explained

Zero-shot classification allows the model to make predictions without seeing labeled examples. You give the model:

• An image

• A list of text prompts (labels)

CLIP calculates which label best describes the image using similarity scores.

3. CLIP Architecture

CLIP contains:

• A Vision Transformer encoder (ViT-B/32 or ViT-L/14)

• A text transformer encoder

Both encoders convert input into embeddings. Similarity is computed using cosine similarity.

4. Workflow Steps

1. Load CLIP model + processor

2. Load image

3. Define text labels

4. Convert both to embeddings

5. Compute similarity

6. Pick class with highest probability

5. Python Code (Hugging Face)

```
from PIL import Image
import requests
from transformers import CLIPProcessor, CLIPModel

model = CLIPModel.from_pretrained("openai/clip-vit-base-patch32")
processor = CLIPProcessor.from_pretrained("openai/clip-vit-base-patch32")

url = "https://raw.githubusercontent.com/openai/CLIP/main/CLIP.png"
image = Image.open(requests.get(url, stream=True).raw)

labels = [
"a diagram",
"a cat",
"a dog",
"a piece of engineering equipment"
]

inputs = processor(text=labels, images=image, return_tensors="pt", padding=True)
outputs = model(**inputs)

probs = outputs.logits_per_image.softmax(dim=1)
print(probs)
```

6. Best Prompting Practices

• Use descriptive labels (“a high-resolution photo of a hydraulic pump”).

• Provide multiple text variations.

• Avoid extremely short labels.

• Consider domain-specific vocabulary for industry applications.

7. Real-World Use Cases

• Manufacturing: Classify machine images, detect anomalies.

• Safety: Identify PPE like helmets or gloves.

• Retail: Auto-tag products for inventory.

• Media: Automatic image captioning & moderation.

• Industrial plants: Categorize valves, sensors, hydraulic parts.

8. Deployment Patterns

CLIP can be deployed with:

• FastAPI — API endpoint

• Gradio — interactive UI

• Hugging Face Spaces — cloud-hosted app

• Docker — production container

• ONNX — optimized inference

9. Example Gradio App

```
import gradio as gr

from transformers import CLIPProcessor, CLIPModel

from PIL import Image

model = CLIPModel.from_pretrained("openai/clip-vit-base-patch32")

processor = CLIPProcessor.from_pretrained("openai/clip-vit-base-patch32")

def classify(image, labels):

labels = [l.strip() for l in labels.split(",")]
```

```
inputs = processor(text=labels, images=image, return_tensors="pt", padding=True)

outputs = model(**inputs)

probs = outputs.logits_per_image.softmax(dim=1)[0].tolist()

return {labels[i]: float(probs[i]) for i in range(len(labels))}


gr.Interface(

fn=classify,

inputs=[gr.Image(type="pil"), gr.Textbox(label="Labels (comma-separated)")],

outputs="label"

).launch()
```

10. Summary

CLIP ViT models allow fast, powerful, zero-training classification. Perfect for AI demos, industrial solutions, and rapid prototyping.