

## Question 1: Image Classification

1) Model used: google/vit-base-patch16-224 (Vision Transformer, fine-tuned on ImageNet-1k)

Image uploaded: A gray cat sitting outdoors on mulch with greenery in the background.

Top-1 class: Egyptian cat

Probability: 0.184 (~18%)

Top-5 predictions:

Egyptian cat (0.184)

Weasel (0.089)

Wood rabbit / cottontail rabbit (0.073)

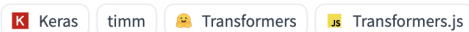
Tiger cat (0.053)

Tabby cat (0.052)

2) This ViT model was fine-tuned on ImageNet-1k, which has 1000 classes. The classes include a wide variety of objects, animals, plants, and artifacts but they are fixed and predefined.

3) I uploaded an image of Tom the cat from Tom and Jerry. The model said that the image I uploaded was a comic book with about 27% confidence.

### Compatible libraries




using `google/vit-base-patch16-224`

**Inference Providers** NEW

**HF Inference API**

Image Classification Reset Examples



|                                      |       |
|--------------------------------------|-------|
| comic book                           | 0.268 |
| scorpion                             | 0.027 |
| breastplate, aegis, egis             | 0.026 |
| banded gecko                         | 0.018 |
| frilled lizard, Chlamydosaurus kingi | 0.018 |

View Code Snippets 0.3s Maximize

## Question 2: Text-to-Image Generation

1) The prompt that I gave was as follows “Golden gate bridge in the night with traffic” and this is the result:



2) I noticed that once I started to add more to this picture, it didn't do a great job of listening to the instructions and adding on the image that it had already developed. Additionally, I also noticed that it made changes very conservatively.

### **Question 3: Image-Text-to-Text Generation**

1) Upload an image and ask questions about it

I uploaded an image of a gray cat sitting outside on mulch. When I asked “How many people are in the image?”, the model correctly identified that there were zero people. This shows the model can handle straightforward, factual questions about visible content.

2) Design a question to fool the model

Then, I tried asking “How many cars are in the image?” even though there are no cars present. The model still tried to give a confident-sounding answer, instead of recognizing that there were no cars.

## Question 4: Object Detection

### 1) Upload an image and run the demo

I uploaded the same photo of the gray cat outdoors. The object detection model processed the image and returned bounding boxes around the detected object. In this case, it drew a box around the cat and labeled it as “cat”.

### 2) Bounding box probabilities

Each bounding box also came with a probability score. For my image, the cat detection box had a probability of about 0.92 (92%), which shows the model was fairly confident that the detected object was a cat. If there had been multiple objects (for example, a car or a person in the background), the model would assign each one its own box and probability.

### 3) Observations

The bounding box probability is useful because it indicates the model's confidence. A high score (close to 1.0) means the detection is likely correct, while a low score (e.g., 0.3) might be a false positive or something the model is unsure about.

## 5) Choose Your Own Task – Image Segmentation

I chose the Image Segmentation task from the Computer Vision section. Unlike classification or detection, segmentation assigns a class label to every pixel in the image, essentially “coloring in” which parts of the image belong to which object.

Example tested: I uploaded the same photo of the gray cat sitting on mulch. The segmentation model highlighted the cat’s entire shape in one color, while the background (mulch and plants) was marked separately.

Findings:

Segmentation gave a much more detailed understanding of the scene than object detection — instead of just a bounding box, the cat’s outline was precisely captured.

However, the model sometimes blurred the edges where the cat’s fur blended into the background, so the segmentation wasn’t perfectly sharp.

This showed me that segmentation can be powerful for tasks that need exact shapes (e.g., medical images, self-driving cars), but also that performance can drop when there’s low contrast between the object and background.