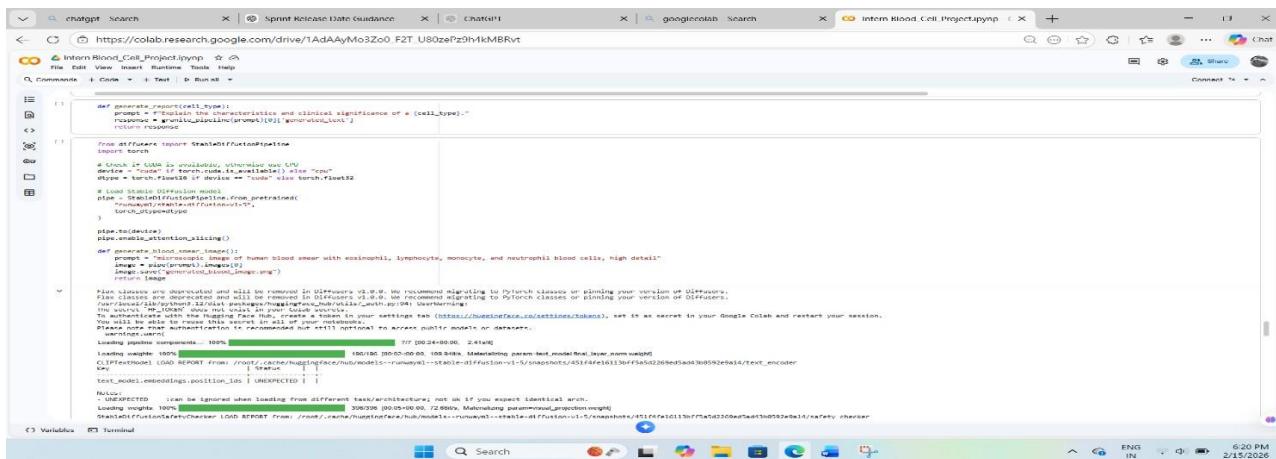


Project Demonstration Phase

Stable Diffusion Model Initialization and Device

This section of the code initializes the Stable Diffusion v1.5 model using the Hugging Face Diffusers library. It first checks whether CUDA (GPU) is available and selects either GPU or CPU for execution to optimize performance. The appropriate data type (float16 for GPU or float32 for CPU) is selected to reduce memory usage. The pipeline is then loaded and moved to the selected device, enabling efficient image generation.



The screenshot shows a Google Colab interface with several tabs open. The main code cell contains Python code for initializing a Stable Diffusion pipeline. It imports necessary libraries, checks for CUDA availability, and loads the pre-trained model. The pipeline is then moved to the selected device (CPU or GPU). The output shows the successful loading of components and the creation of a synthetic blood smear image named 'generated_blood_image.png'.

```
def generate_responsible_call():
    """Generate a responsible call describing the characteristics and clinical significance of a (call_type)."""
    response = generate_responsible_prompt([{"call_type": "responsible"}])
    return response

from diffusers import StableDiffusionPipeline
import torch

# Check if CUDA is available, otherwise use CPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
dtype = torch.float16 if device == "cuda" else torch.float32

# Load Stable Diffusion model
pipe = StableDiffusionPipeline.from_pretrained(
    "runwayml/stable-diffusion-v1-5",
    torch_dtype=dtype
)
pipe.to(device)
print(pipe)

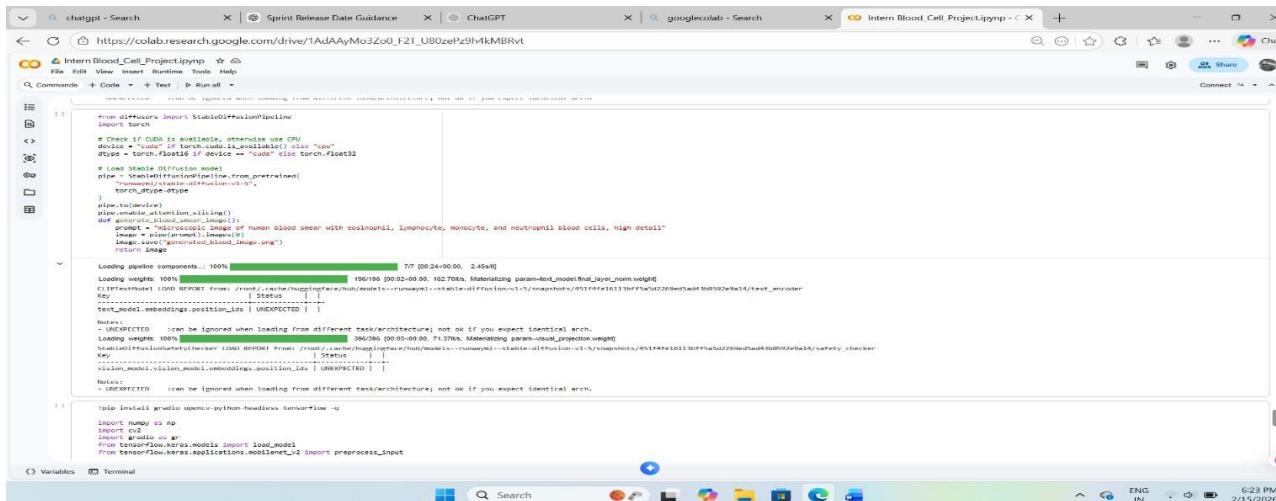
def generate_blood_smear_image():
    """Prompt: "Microscopic image of human blood smear with eosinophil, lymphocyte, monocyte, and neutrophil blood cells, high detail"
    image = pipe(prompt=prompt).images[0]
    image.save("generated_blood_image.png")"""

# Note: All classes are deprecated and will be removed in Difffusers v2.0.0. We recommend migrating to PyTorch classes or pluming your version of Difffusers.
# https://github.com/huggingface/difffusers/pull/1044#issuecomment-1139754266
# To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
# Note: This token authentication is recommended but still optional to access public models or datasets.
# Please note that authentication is recommended but still optional to access private models or datasets.

Loading pipeline components... 100% [██████████] 77/00:24<0:00. 2.4sHQ
Loading weights... 100% [██████████] 366/00:00. 162.70%k. Mathttoring param-test_model/bnd_layer_norm.weight
CLIPTextModel LOAD REPORT From: /root/.cache/huggingface/hub/modules--runwayml--stable-diffusion-v1-5/snapshots/45194fe101130f5a5d2269ed9a3d3bd592e9a3d/text_encoder
Key: *****
test_model.embeddings.position_ids | UNEXPECTED [ ]
Notes:
+ UNEXPECTED: can be ignored when loading from different task/architecture, not or if you expect identical arch.
Loading weights... 100% [██████████] 366/00:00. 162.70%k. Mathttoring param-test_model/projection.weight
StableDiffusionSafetyChecker LOAD REPORT From: /root/.cache/huggingface/hub/modules--runwayml--stable-diffusion-v1-5/snapshots/45194fe101130f5a5d2269ed9a3d3bd592e9a3d/safety_checker
Key: *****
test_model.safety_checker.model.unet.downblocks[0].attn1[0].q.weight | UNEXPECTED [ ]
Notes:
+ UNEXPECTED: can be ignored when loading from different task/architecture, not or if you expect identical arch.
```

Blood Smear Image Generation Using Text-to-Image Prompt

This part of the code defines the function `generate_blood_smear_image()` to create synthetic microscopic blood smear images. A detailed medical prompt describing eosinophils, lymphocytes, monocytes, and neutrophils is provided to the model. The Stable Diffusion pipeline generates a high-resolution image based on this prompt and saves it as `generated_blood_image.png`. The console output below confirms successful loading of model weights and components.



The screenshot shows a Google Colab interface with several tabs open. The main code cell contains Python code for generating a blood smear image. It imports necessary libraries, checks for CUDA availability, and loads the pre-trained model. The pipeline is then moved to the selected device (CPU or GPU). The output shows the successful loading of components and the creation of a synthetic blood smear image named 'generated_blood_image.png'.

```
def generate_responsible_call():
    """Generate a responsible call describing the characteristics and clinical significance of a (call_type)."""
    response = generate_responsible_prompt([{"call_type": "responsible"}])
    return response

from diffusers import StableDiffusionPipeline
import torch

# Check if CUDA is available, otherwise use CPU
device = torch.device("cuda" if torch.cuda.is_available() else "cpu")
dtype = torch.float16 if device == "cuda" else torch.float32

# Load Stable Diffusion model
pipe = StableDiffusionPipeline.from_pretrained(
    "runwayml/stable-diffusion-v1-5",
    torch_dtype=dtype
)
pipe.to(device)
print(pipe)

def generate_blood_smear_image():
    """Prompt: "Microscopic image of human blood smear with eosinophil, lymphocyte, monocyte, and neutrophil blood cells, high detail"
    image = pipe(prompt=prompt).images[0]
    image.save("generated_blood_image.png")"""

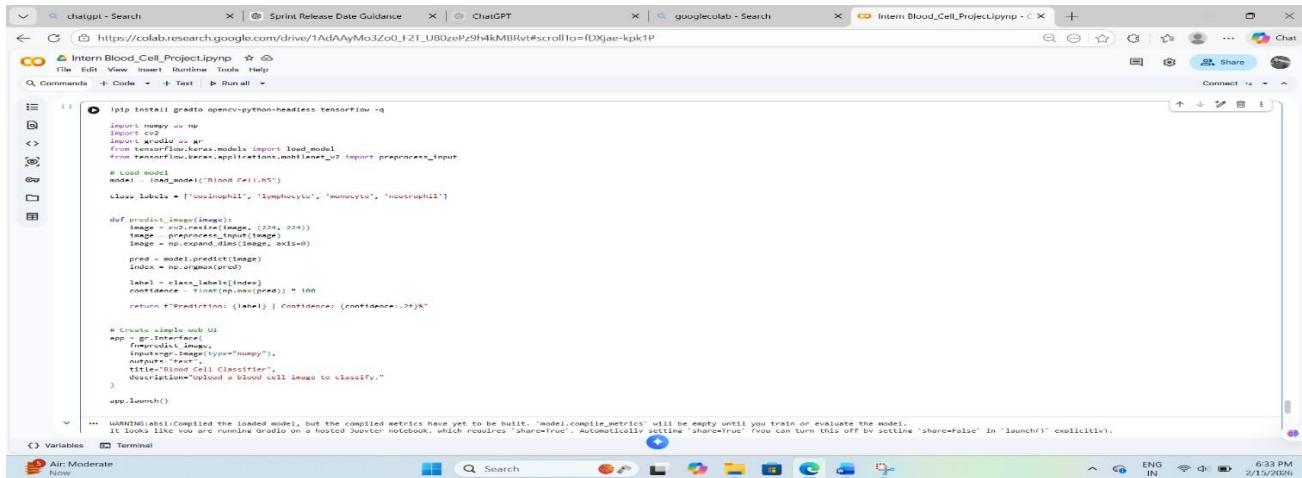
# Note: All classes are deprecated and will be removed in Difffusers v2.0.0. We recommend migrating to PyTorch classes or pluming your version of Difffusers.
# https://github.com/huggingface/difffusers/pull/1044#issuecomment-1139754266
# To authenticate with the Hugging Face Hub, create a token in your settings tab (https://huggingface.co/settings/tokens), set it as secret in your Google Colab and restart your session.
# Note: This token authentication is recommended but still optional to access public models or datasets.
# Please note that authentication is recommended but still optional to access private models or datasets.

Loading pipeline components... 100% [██████████] 77/00:24<0:00. 2.4sHQ
Loading weights... 100% [██████████] 366/00:00. 162.70%k. Mathttoring param-test_model/bnd_layer_norm.weight
CLIPTextModel LOAD REPORT From: /root/.cache/huggingface/hub/modules--runwayml--stable-diffusion-v1-5/snapshots/45194fe101130f5a5d2269ed9a3d3bd592e9a3d/text_encoder
Key: *****
test_model.embeddings.position_ids | UNEXPECTED [ ]
Notes:
+ UNEXPECTED: can be ignored when loading from different task/architecture, not or if you expect identical arch.
Loading weights... 100% [██████████] 366/00:00. 162.70%k. Mathttoring param-test_model/projection.weight
StableDiffusionSafetyChecker LOAD REPORT From: /root/.cache/huggingface/hub/modules--runwayml--stable-diffusion-v1-5/snapshots/45194fe101130f5a5d2269ed9a3d3bd592e9a3d/safety_checker
Key: *****
test_model.safety_checker.model.unet.downblocks[0].attn1[0].q.weight | UNEXPECTED [ ]
Notes:
+ UNEXPECTED: can be ignored when loading from different task/architecture, not or if you expect identical arch.

! pip install gradio opencv-python-headless tensor-time -q
import numpy as np
import cv2
import imageio as im
from tensorflow.keras.models import load_model
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
```

Gradio-Based Deployment of Blood Cell Classification Model

This section of the code deploys the trained Blood Cell classification model using the Gradio library. The saved model `Blood_Cell.h5` is loaded, and class labels such as eosinophil, lymphocyte, monocyte, and neutrophil are defined. The `predict_image()` function preprocesses the uploaded image by resizing it to 224x224 pixels, applying MobileNetV2 preprocessing, and expanding dimensions before making predictions. The predicted class is determined using `argmax`, and the confidence score is calculated and displayed as output.



```
! pip install gradio opencv-python-headless tensorflow -q
import numpy as np
import gradio as gr
from tensorflow.keras.models import load_model
from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
# Load model
model = load_model('Blood_Cell.h5')
class_labels = ['Eosinophil', 'Lymphocyte', 'Monocyte', 'Neutrophil']

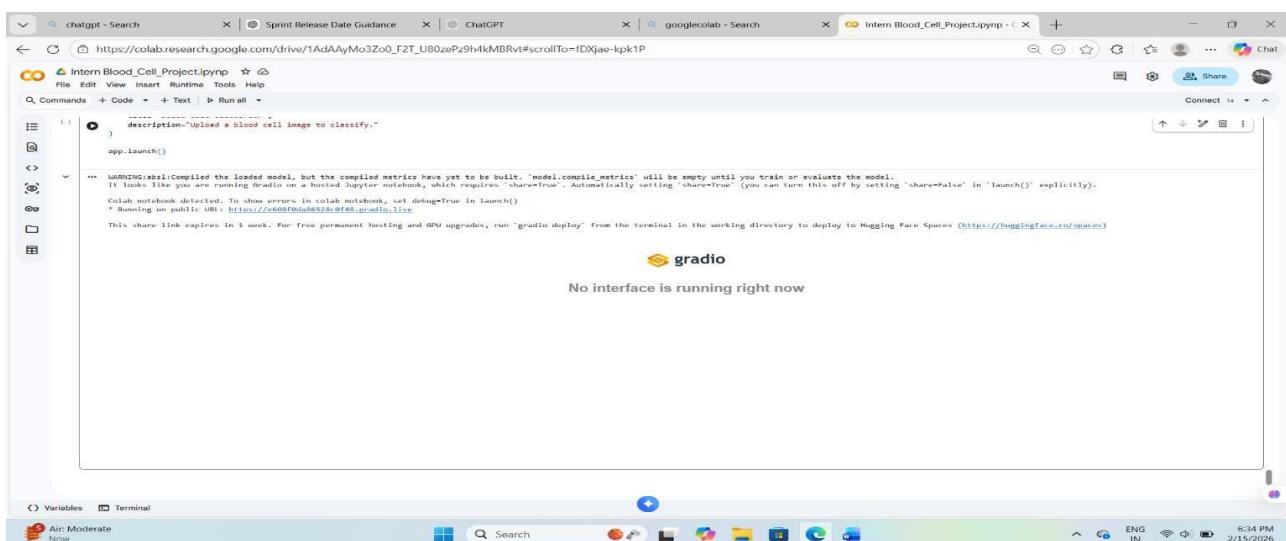
def predict(image):
    image = cv2.resize(image, (224, 224))
    image = preprocess_input(image)
    image = np.expand_dims(image, axis=0)
    pred = model.predict(image)
    index = np.argmax(pred)
    label = class_labels[index]
    confidence = float(np.max(pred)) * 100
    return f'Prediction: {label} | Confidence: {confidence:.2f}%'

# Create simple web UI
app = gr.Interface(
    predict,
    inputs='image',
    outputs='text',
    title='Blood Cell Classifier',
    description='Upload a blood cell image to classify.'
)
app.launch()

WARNING:Absl:Compiled the loaded model, but the compiled metrics have yet to be built. 'model.compile_metrics' will be empty until you train or evaluate the model.
It looks like you are running Gradio on a hosted Jupyter notebook, which requires 'share=True'. Automatically setting 'share=True' (you can turn this off by setting 'share=False' in 'launch()' explicitly).
```

Launching the Web Interface for Model Deployment

This part shows the execution of `app.launch()`, which starts the Gradio web interface for user interaction. Once launched, the application generates a public shareable link, allowing users to upload blood cell images and receive classification results in real time. The console output confirms successful deployment and provides the temporary URL for accessing the application online.



```
app.launch()

WARNING:Absl:Compiled the loaded model, but the compiled metrics have yet to be built. 'model.compile_metrics' will be empty until you train or evaluate the model.
It looks like you are running Gradio on a hosted Jupyter notebook, which requires 'share=True'. Automatically setting 'share=True' (you can turn this off by setting 'share=False' in 'launch()' explicitly).
Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://e688hd08832c0f48.gradio.live
This share link expires in 1 week. For free permanent hosting and GPU upgrades, run 'gradio deploy' from the terminal in the working directory to deploy to Hugging Face Spaces (https://huggingface.co/spaces)

No interface is running right now
```

127.0.0.1:5000

Welcome to the HematoVision

About Blood Cells

Blood cells are vital components of our body, playing essential roles in immunity, oxygen transport, and clotting. Understanding different types of blood cells is crucial for diagnosing various medical conditions.

Predict Blood Cell Type

Upload an image of a blood cell to determine its type using our state-of-the-art classification model.

No file chosen

By clicking on choose file it will ask us to upload the image , then by clicking on the predict button it will take us to the result.html.

Activity 1: Test For Class-1 – Neutrophil

Predict Blood Cell Type

Upload an image of a blood cell to determine its type using our state-of-the-art classification model.

_8_9488.jpeg

Prediction Result

Predicted Class: neutrophil



Activity 2: Test For Class-2 – Monocyte

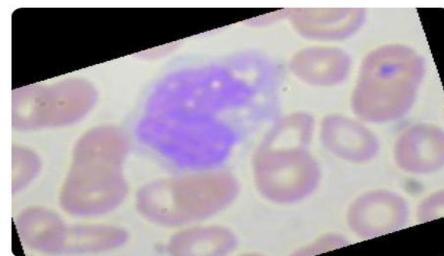
Predict Blood Cell Type

Upload an image of a blood cell to determine its type using our state-of-the-art classification model.

_3_9423.jpeg

Prediction Result

Predicted Class: monocyte



Activity 3: Test For Class-3 – Lymphocyte

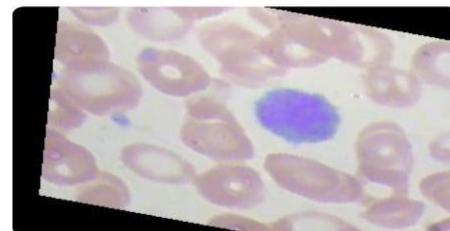
Predict Blood Cell Type

Upload an image of a blood cell to determine its type using our state-of-the-art classification model.

_5_9201.jpeg

Prediction Result

Predicted Class: lymphocyte



Activity 4: Test For Class-4 – Eosinophil

Predict Blood Cell Type

Upload an image of a blood cell to determine its type using our state-of-the-art classification model.

_3_9885.jpeg

Prediction Result

Predicted Class: eosinophil



Demo Link:

https://drive.google.com/file/d/1Fn6HTHfAA1PcC5Z_RBEgUgTjzyjEOQwQ/view?usp=drivesdkGoogle

Colab Link :

<https://colab.research.google.com/drive/1nnHWn4UKBKx6oxfna4zrmP5ovKtj4S5b>