PyTorch Image Classification Report

https://github.com/mohammad-yousuf/diffusion-classifier-pytorch

Model Overview

Two models are utilized for image classification:

1. **Stable Diffusion Model (stabilityai/stable-diffusion-2-1-base)**

2. **ResNet-18 Model (pre-trained on ImageNet)**

Model and Dataset Configuration

Model Identification

- The Stable Diffusion model (`stabilityai/stable-diffusion-2-1-base`) is a latent diffusion model used to generate high-quality images from text prompts.

Device Setup

- The models run on CUDA if available, otherwise on CPU.

Image Processing

- **Conversion to RGB**: Ensures all images are in RGB format.
- **Image Transformation**: Images are resized to 512x512 pixels, center cropped, and normalized.

Dataset Details

- **Dataset**: UC Merced Land Use dataset (remote sensing data).
- **Classes**: 21 land use classes (e.g., agricultural, airplane, beach).
- **Images**: Each class contains 100 images of size 256x256 pixels.
- **Split**: Only the test split is used for evaluation.

Data Loading and Preprocessing

- Images are loaded from the specified dataset path.
- The dataset is split into 80% training and 20% testing sets.
- Transformations are applied to ensure images are resized, center cropped, and normalized.

Model Loading

Stable Diffusion Model Components

- **Variational Autoencoder (VAE)**
- **Tokenizer**
- **Text Encoder**
- **U-Net**

- **Euler Discrete Scheduler**

Training and Evaluation

ResNet-18 Model

- **Training**:
  - The model is trained on the training dataset using a cross-entropy loss function and Adam optimizer.
  - Training involves iterating over multiple epochs and updating model weights based on the loss.
- **Evaluation**:
  - The trained model is evaluated on the test dataset.
  - Metrics such as accuracy, precision, recall, and F1 score are calculated.

Stable Diffusion Model

- **Evaluation**:
  - The model evaluates image classification adaptively using a probability-based approach.
  - The evaluation process involves adding noise to latent representations, predicting noise with U-Net, and calculating mean squared error (MSE) loss.
  - The most probable prediction is determined using the top-k approach.

Performance Metrics

Stable Diffusion Model

- **Final Accuracy**: 50.12%
- **Accuracy**: 0.5012
- **Precision**: 0.5349
- **Recall**: 0.5012
- **F1 Score**: 0.4881

ResNet-18 Model

- **Accuracy**: 88.5986
- **Precision**: 92.0790
- **Recall**: 88.5986
- **F1 Score**: 87.3919

Visualization

- Sample images from the test dataset are displayed with their predicted and true labels.
- The visual comparison helps in understanding model performance qualitatively.

Challenges Faced

Preprocessing for Stable Diffusion

- **Complexity**: Preprocessing images for the Stable Diffusion model required multiple steps, including resizing, center cropping, and normalization.
- **Custom Transformations**: Ensuring that all images were converted to RGB and appropriately transformed added complexity to the preprocessing pipeline.

Computational Intensity

- **High Resource Demand**: Both models, especially the Stable Diffusion model, required significant computational resources. Running these models on large datasets or multiple epochs was computationally intensive and time-consuming.
- **Hardware Limitations**: Limited GPU memory and availability impacted the ability to process large batches and required optimizations to fit within hardware constraints.

Deciding on Prompts Filtering and Stages

- **Adaptive Evaluation**: Determining the optimal number of prompts to filter in various stages was challenging. The process involved balancing between filtering too many prompts early or evaluating too many prompts unnecessarily.
- **Tuning Parameters**: Deciding on parameters such as the number of samples and keep counts required careful tuning to ensure effective filtering while maintaining high accuracy.

Future Enhancements

Strongly Guided Image Generation

- **Specific Data Types**: Implementing strongly guided image generation techniques tailored to specific data types can improve classification accuracy. For example, generating images guided by specific class attributes or using domain-specific priors.
- **Enhanced Prompts**: Developing more sophisticated prompt engineering techniques to provide stronger guidance to the model can result in better quality and more accurate image generation.

Computational Optimization

- **Efficient Algorithms**: Exploring more efficient algorithms and optimization techniques to reduce computational load and improve processing times.
- **Hardware Utilization**: Leveraging more powerful hardware or distributed computing environments to handle large-scale data processing and model training.

Summary

- The code implements a pipeline for image classification using both the Stable Diffusion and ResNet-18 models.
- The process includes data preprocessing, model loading, training, and evaluation.

- Performance metrics indicate that the ResNet-18 model significantly outperforms the Stable Diffusion model for image classification tasks.
- The approach highlights the importance of accurate image classification in remote sensing data, with potential applications in environmental monitoring, urban planning, and resource management.
- Addressing the challenges and exploring future enhancements can lead to more robust and efficient image classification solutions.