

LAB 1

Fine-Tuning BLIP

BLIP (Bootstrapping Language-Image Pre-training) is a multimodal deep learning model designed to bridge the gap between vision and language.

Key Characteristics of BLIP

- **Purpose:** It is used for tasks that involve both images and text, such as **image captioning**, where the model generates a descriptive sentence for a given image.
- **Core Architecture:**
 - **Vision-Language Model:** It processes visual data (images) and linguistic data (text) simultaneously to understand their relationship.
 - **Pre-trained Foundation:** The model used in the notebook is pre-trained on large-scale datasets, allowing it to have a strong baseline understanding before being "fine-tuned" on specific data.
- **Components:**
 - **BlipProcessor:** A unified tool that handles both image preprocessing (resizing and normalization) and text tokenization.
 - **BlipForConditionalGeneration:** The specific model class used to generate text (captions) based on visual input.

1. Project Overview

The objective of this notebook is to fine-tune a pre-trained BLIP model on a custom image captioning dataset. BLIP is a multimodal model capable of understanding and generating descriptions for images.

2. Environment & Prerequisites

The implementation is designed to run in a **Google Colab** environment with specific hardware acceleration:

- **Accelerator:** Standard GPU Class (required for efficient training of the ~990MB model).
- **Runtime:** Python 3.
- **Dependencies:** Hugging Face `transformers` and `datasets` libraries.

3. Core Components

The workflow relies on several critical assets retrieved from the Hugging Face Hub:

- **Model Weights:** The primary model file, approximately 990MB in size.
- **Processor (BlipProcessor):** A unified tool that handles:
 - **Vision:** Image resizing and normalization.
 - **Language:** Text tokenization and encoding for the captions.
- **Configuration:** A `config.json` file (~4.72kB) that defines the specific model architecture and hyperparameters.

4. Implementation Steps

A. Data Preparation

1. **Dataset Loading:** The notebook utilizes the `datasets` library to load image-caption pairs.
2. **Preprocessing Logic:**
 - Images are converted into pixel values using the processor.
 - Captions are tokenized into input IDs.
 - The processor ensures that both modalities are aligned for the model's expected input format.

B. Training Configuration

- Monitoring:** The notebook uses Jupyter widgets (`FloatProgress`, `HBox`) to provide real-time visual feedback on download progress and training metrics.
- Optimization:** Training is performed on a GPU, leveraging standard deep learning optimization techniques (like AdamW, though specific hyperparameters depend on the exact cell execution).

C. Fine-Tuning Process

- The model is placed in training mode.
- For each batch, the model receives an image and its corresponding caption.
- The loss is calculated based on the model's ability to predict the next token in the caption given the image context.
- Backpropagation updates the model weights to improve accuracy on the specific custom dataset.

5. Summary of Downloaded Resources

| Asset | Size | Purpose |
|---------------------------------------|----------|--------------------------------|
| <code>pytorch_model.bin</code> | ~990 MB | Pre-trained BLIP weights |
| <code>config.json</code> | ~4.72 KB | Model architecture definitions |
| <code>preprocessor_config.json</code> | ~431 B | Image transformation settings |