Data Mining & Machine Learning: Visual Question Answering with Florence-2

Authors:

Zahra Jafarinejad (D03000083)

Farshad Farahtaj (D03000028)

Seyyed Alireza Khoshsolat (D03000041)

A Comprehensive Study on Fine-Tuning, Resource Management, and Web Deployment

Institution: Università degli Studi di Napoli Federico II
Date: March 11, 2025

Abstract

This report presents an in-depth exploration of adapting a pre-trained Visual Question Answering (VQA) model to a custom dataset comprising 50,000 cartoonish images and deploying it through a user-friendly web interface. Initial experiments with PaliGemma and BLIP encountered significant obstacles, such as high resource demands and preprocessing incompatibilities. Florence-2 emerged as the optimal choice due to its compact size, efficiency, and adaptability. The model was fine-tuned on the custom dataset and deployed on Hugging Face Spaces using a Streamlit application, achieving practical utility with over 1,000 monthly users. This report details the dataset analysis, model selection process, fine-tuning methodology, deployment strategy, and future directions, offering a thorough account of the project's development and outcomes.

---

1. Introduction

Visual Question Answering (VQA) is an interdisciplinary task at the intersection of computer vision (CV) and natural language processing (NLP), requiring a system to interpret an image and a natural language question to provide an accurate textual response. Conducted at Università degli Studi di Napoli Federico II, this project aimed to adapt a pre-trained VQA model to a custom dataset of abstract cartoonish scenes and deploy it as an accessible web application, emphasizing efficiency and usability.

The process involved evaluating multiple models, refining preprocessing pipelines, overcoming resource constraints, and deploying the final solution on a cloud platform. The report follows this structure: exploratory data analysis, dataset and preprocessing details, model selection and comparison, fine-tuning of Florence-2, deployment on

Hugging Face Spaces, a live demo summary, and a conclusion with future work recommendations.

---

2. Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) was conducted to understand the dataset's composition and inform subsequent preprocessing and modeling decisions.

- Dataset Composition:
    - Images: 50,000 fully colored PNG images depicting abstract cartoonish scenes, with a native resolution of 700×400 pixels. The dataset is split into 20,000 training images, 10,000 validation images, and 20,000 testing images.
    - Questions and Answers: Stored in JSON files:
        - Training Set: Two JSON files for questions (open-ended and multiple-choice) and one annotation file with answers.
        - Validation Set: Identical structure to the training set.
        - Testing Set: Two JSON files for questions (open-ended and multiple-choice), with no annotations provided, as these are reserved for inference.
- Insights from Exploration:
    - Word clouds for training and validation questions revealed frequent terms (e.g., "what," "where," "how many"), indicating common question types.
    - Answer distributions showed recurring responses (e.g., colors, numbers), visualized via word clouds and top 30 frequency lists.
    - Question length distributions (word counts) highlighted typical query complexity, aiding in text preprocessing decisions.

---

3. Dataset & Preprocessing

3.1 Dataset Overview

- Total Images: 50,000, divided as follows:
    - Training: 20,000 images.
    - Validation: 10,000 images.

    - Testing: 20,000 images.
- JSON File Structure:
    - Training/Validation Sets: Three files per split: one for open-ended questions, one for multiple-choice questions, and one for annotations (answers to both question types).
    - Testing Set: Two files: one for open-ended questions and one for multiple-choice questions, without annotations.
- Image Characteristics:
    - Format: PNG, fully colored.
    - Resolution: 700×400 pixels.

3.2 Preprocessing

Preprocessing was critical to align the dataset with model input requirements, addressing both text and image data.

- Text Preprocessing:
    - Issue: The raw JSON files included multiple-choice questions, which were incompatible with the input structures of PaliGemma, BLIP, and Florence-2 due to their complexity (e.g., requiring option parsing).
    - Solution:
        - Excluded multiple-choice questions, focusing solely on open-ended QA pairs for simplicity and model compatibility.
        - Created new JSON files with a standardized structure:
            - image: Path to the corresponding image file.
            - question: Open-ended query (e.g., “What is in the image?”).
            - answer: Ground-truth response (e.g., “A red car”).
        - Normalized text by converting to lowercase and trimming whitespace for consistency.
- Image Preprocessing:
    - Issue: Each model required a specific input resolution, typically 224×224 pixels, while the original images were 700×400 pixels.
    - Solution:

        - Resized all images to 224×224 pixels using an image processing library (e.g., PIL or OpenCV), ensuring compatibility across all evaluated models.
        - Converted images to RGB format to maintain consistency, as some models expect three-channel input

---

4. Model Selection

Three vision-language models—PaliGemma, BLIP, and Florence-2—were evaluated based on architecture, resource requirements, preprocessing needs, and fine-tuning feasibility. Florence-2 was selected as the final model.

4.1 Comparison of PaliGemma, BLIP, and Florence-2

- PaliGemma: A Versatile Vision-Language Model (VLM)
    - Architecture:
        - Vision Encoder: SigLIP (So400m), a Vision Transformer with 400M parameters, pre-trained via contrastive learning on 40B image-text pairs.
        - Language Decoder: Gemma-2B, a 2.6B-parameter decoder-only transformer from Google's open-source LLM family.
        - Fusion: Cross-attention layers integrate visual and textual inputs.
    - Strengths: Handles over 40 tasks (e.g., VQA, captioning, OCR) and runs on mid-range GPUs compared to larger models.
    - Limitations: Requires 40GB+ GPU memory for fine-tuning, sensitive to prompt formatting (e.g., "answer en"), and demands complex preprocessing.
- BLIP: Bootstrapping Language-Image Pre-training
    - Architecture:
        - Image Encoder: Vision Transformer (ViT-B/16 or ViT-L/16) for patch embeddings and global features.
        - Text Encoder-Decoder: Multimodal Mixture of Encoder-Decoder (MED) with shared weights, supporting both understanding and generation tasks.

        - Fusion: Cross-attention layers merge image and text embeddings.
    - Strengths: Robust to noisy data via bootstrapping (synthetic captions and filtering).
    - Limitations: Resource-heavy (requires large-scale pre-training on 14M+ images), complex preprocessing (caption filtering removes ~15% of data), and prone to training instability.
- Florence-2: A Unified Vision-Language Foundation Model
    - Architecture:
        - Image Encoder: Vision Transformer pre-trained on the FLD-5B dataset (5.4B annotations).
        - Text Encoder-Decoder: Sequence-to-sequence transformer (0.2B–0.7B parameters) with prompt-based multitask learning.
        - Fusion: Cross-attention integrates visual tokens and text prompts.
    - Strengths: Compact size enables deployment on mid-range GPUs, unified architecture reduces task-specific engineering, and minimal preprocessing is required.
    - Limitations: Requires precise prompt formatting (e.g., “answer en”).

4.2 Challenges with PaliGemma & BLIP

- PaliGemma:
    - Accessibility: Limited pre-trained checkpoints on platforms like Hugging Face were incompatible with single-task VQA fine-tuning.
    - Formatting: Required complex prompt engineering (e.g., “answer en”) and resolution adjustments.
    - Resources: Fine-tuning demanded 40GB+ GPU memory, exceeding available resources and causing crashes.
- BLIP:
    - Memory: Even the smallest variant (ViT-B/16) crashed GPUs due to high memory usage during contrastive learning.
    - Preprocessing: Synthetic caption generation and filtering removed ~15% of valid QA pairs, complicating the pipeline.
    - Stability: Gradient accumulation and sharded training failed to prevent kernel crashes.

4.3 Why Florence-2 Succeeded

- Efficiency: With 771M parameters (versus PaliGemma's 3B and BLIP's 1.3B), Florence-2 enabled stable training on limited GPUs.
- Simplicity: Prompt-based design minimized dataset reformatting, requiring only image resizing and basic JSON adjustments.
- Stability: Freezing image encoder layers reduced memory usage, though full training was feasible without freezing in this case.

Model	GPU Memory	Dataset Complexity	Training Stability
PaliGemma	High Demand ❌	High ❌	Low ❌
BLIP	High Demand ❌	Moderate ❌	Unstable ❌
Florence-2	Medium Demand ✅	Low ✅	Stable ✅

---

5. Fine-Tuning Florence-2

5.1 Methodology

- Dataset Preparation:
  - Image Resizing: Converted 700×400 images to 224×224 pixels to match Florence-2's input requirements.
  - Question Filtering: Excluded multiple-choice questions, retaining open-ended QA pairs. New JSON files were structured with image, question, and answer fields.
  - Example Entry: { "image": "path/to/image.png", "question": "What is the color of the car?", "answer": "red" }.
- Model Setup:
  - Loaded pre-trained Florence-2 weights from Hugging Face, originally trained on medical prescription data but adaptable to cartoonish scenes.
  - No layers were frozen, allowing all 771M parameters to update during training for maximum adaptability.
- Training Workflow:
  - Pipeline: Loaded JSON files, processed images and questions, and fine-tuned the model.
  - Hyperparameters:

        - Batch Size: 8 (optimized for GPU memory).
        - Epochs: 3.
        - Optimizer: AdamW with linear learning rate scheduling.
    - Training was conducted on a mid-range GPU, leveraging Florence-2's efficiency.

5.2 Challenges & Solutions

- Challenge: Dataset alignment with Florence-2's input format.
    - Solution: Iterative reformatting of JSON files and image resizing to 224×224 pixels ensured compatibility.
- Challenge: Potential resource constraints on mid-range GPUs.
    - Solution: Florence-2's compact size and optimized batch size prevented memory issues, avoiding the need for gradient accumulation or mixed precision training (though these were considered as fallback options).

5.3 Results

Florence-2 was successfully fine-tuned, achieving reliable performance on open-ended VQA tasks with the cartoonish dataset. The model consistently generated accurate answers, validated through manual inspection of validation set outputs.

---

6. Deployment

6.1 Web App Setup on Hugging Face Spaces

- Freezing the Model: Post-fine-tuning, the model weights were frozen to ensure consistent inference behavior in production.
- Model Compression:
    - Used Python's shutil.make_archive to compress the fine-tuned model into a ZIP file, reducing upload size for cloud deployment.
- Saving Components:
    - Saved model weights with model.save_pretrained("florence2_finetuned_model").
    - Saved processor configuration with processor.save_pretrained("florence2_finetuned_model").
- Deployment Steps:

    - Created a dedicated space on Hugging Face.
    - Uploaded the compressed ZIP file, extracted it as florence2_finetuned_model.
    - Developed a Streamlit web app (app.py) for user interaction.

6.2 Web App Features

- User Interface:
    - Users upload images and input questions via a text field.
    - Answers are displayed with confidence percentages derived from model output scores.
- Advanced Settings:
    - Sliders adjust inference parameters: number of beams (for beam search) and max new tokens (output length).
- History Tracking:
    - Maintains a session log of questions, answers, and confidence scores, displayed on the interface.
    - Offers a downloadable ZIP file containing the session history.
- Benefits:
    - Cost-Effectiveness: Leverages Hugging Face's cheap GPU-tier hosting.
    - Ease of Use: Streamlit simplifies UI development for non-technical users.
    - Reach: Over 1,000 monthly users interacted with the app, demonstrating its accessibility.

---

7. Live Demo

The deployed Streamlit app on Hugging Face Spaces provides a practical demonstration:

- Input: Users upload an image (e.g., a scene with a red car) and ask question (e.g., "What is the color of the car?").
- Output: The model returns an answer (e.g., "red") with a confidence score (e.g., 95%).
- Interaction: Users can also tweak settings (e.g., beams) and download their session history.

- Link of the web app:
    - http://bit.ly/3Ff3nQT

This demo highlights Florence-2's ability to interpret visual and textual inputs effectively in a real-world setting.

---

8. Conclusion & Future Work

8.1 Summary of Findings

- Fine-Tuning Success: Florence-2 was adapted to a 50,000-image cartoonish dataset, overcoming preprocessing and resource challenges with its efficient 771M-parameter design.
- Deployment Success: The Streamlit app on Hugging Face Spaces proved scalable and user-friendly, engaging over 1,000 monthly users since deployment.

8.2 Lessons Learned

- Resource Management: Optimizing batch size and leveraging Florence-2's compact size were key to training on limited GPUs.
- Model Selection: Florence-2's unified architecture and prompt-based simplicity outperformed the resource-heavy and complex PaliGemma and BLIP.
- Preprocessing: Iterative refinement of text and image data was essential for model compatibility.

8.3 Future Work

- Hyperparameter Optimization: Conduct a systematic grid search for learning rates, batch sizes, and epochs to enhance performance.
- Multimodal Enhancements: Extend capabilities to multi-image reasoning (e.g., "What changed between these images?") and complex question answering.
- Deployment Features: Add real-time feedback, voice recognition for question input, and mobile app compatibility to broaden accessibility.