Intellect-F4

Approaches - In the face of time constraints and the overwhelming nature of the provided data (both train and test datasets), Our approach evolved through multiple iterations as we confronted various challenges. Here's how we tackled the problem:

Step 1: Building a Custom NER Model We started by thinking of a Named Entity Recognition (NER) model on the provided train dataset. Initially, we experimented with popular models such as BERT and T5, well-known for their state-of-the-art (SOTA) performance in natural language tasks.

The data provided to us included two key files: `train.csv` and `test.csv`, which contained download links and the entities `entity_name` and `entity_value`. The goal was to extract and identify these entities.

Proposed Approach:

- Extract text from images using Optical Character Recognition (OCR).
- Use the extracted text as input to a custom NER model.
- Map the OCR-extracted text to the corresponding `entity_value` and train the NER model on it, with predictions applied to the test dataset.

Why This Approach Failed: Unfortunately, relying heavily on OCR resulted in several issues:

1. **OCR Reliability:** The extracted text often lost key contextual information, which had a significant impact on predictions.
2. **Loss of Context:** OCR outputs struggled to maintain the context of the images, leading to large discrepancies in our final predictions.

Even after fine-tuning both BERT and T5 models, the results were unsatisfactory. Despite their reputation for handling complex tasks, these models didn't produce the desired F1 scores or accurate predictions.

Step 2: Switching to LLMs for Context Preservation Realizing the limitations of NER models, we quickly shifted gears. We needed to preserve contextual information from the OCR-extracted text, so we moved on to trying large language models (LLMs) like Gemma2b and LLaMA7b.

These models demonstrated better performance than our initial NER-based approach, as they excel at maintaining context. However, they introduced a new problem:

- **Hallucination of Entities and Units:** The models occasionally generated fictitious entity names and values, leading to unpredictable results.

To mitigate this, we fine-tuned the Gemma2-2b model using the full set of train.csv images. While this improved the results slightly, giving us a 2% improvement over the base model, the gain was marginal. The overall F1 score remained low at around 0.013, a far cry from our target range of 0.4-0.5.

The Main Challenge: Our main roadblock was the OCR tool's performance. The accuracy of the extracted text was inconsistent, and the image quality significantly affected the results. This, combined with the loss of context in identifying entities, led us back to the drawing board.

Step 3: Multimodal Approach With time rapidly running out (only 1½ days left!), we needed a solution that could understand not just the text but also the context and meaning behind it, along with visual cues from the images.

Final Approach: Using Multimodal Models (Qwen2-2b-vl) The multimodal model Qwen2-2b-VL came to our rescue. It supports both image and text input as tokens, allowing us to feed it both the image and the extracted text simultaneously, preserving context.

Improvements - It can be trained on more train.csv data along with on more epochs to get it fine tune to get fit best possible to train data. We only used 10k samples to fine tune.After fine-tuning Qwen2-2b-vl, it produces the best of all predictions. We fine-tuned it on Train.csv data for two epochs, and it still yielded outstanding results.

Pros: Qwen2-2b-vl can simultaneously understand the contextual information of images and text to predict an output.

Improvements: To further optimize Qwen2-2b-vl, it could be trained on a larger portion of Train.csv data and for more epochs. This would help it fine-tune and fit the training data as well as possible. It's important to note that we only used 10,000 samples for fine-tuning, so there's potential for significant improvement with additional data and training.

Code Files :

Util folder -> contains script to fine tune the model
d_img.ipynb -> download images in batches along with progress saving
Second.ipynb -> download necessary packages and train the qwen model
run_tests.ipynb -> run the model of test to generate response and format responses and final output.

Approach	Efficiency	Pros	Cons
NER Models (BERT, T5)	Low	State-of-the-art performance in natural language tasks	Reliant on OCR, loss of context
LLMs (Gemma2b, LLaMA7b)	Medium	Better context preservation than NER models	Hallucination of entities and units
Multimodal Model (Qwen2-2b-vl)	High	Can understand both image and text input	Requires more training data and epochs for optimal performance

TEAM MEMBERS :

PRAVEEN KUMAR, MOHIT MEENA, RAJ RAUSHAN, MEHUL SIRVI

SIGNOFF :

Intellect-F4