REPORT

Cloud Detection in Remote Sensing Images Using DETR (DEtection TRansformer)

GROUP 3 ICT1
University of Science and Technology of Hanoi

Members of Group 3:

- **Đặng Minh Duy-22BI13118**
- **Nguyễn Duy Hải-22BI13147**
- **Lã Duy Anh-22BI13016**
- **Ngô Hải Anh-22bi13019**
- **Nguyễn Phúc Khang-22bi13206**

I.Introduction

1. What is Cloud Detection?

- Cloud detection refers to the process of identifying and segmenting clouds in images, particularly satellite or aerial imagery. In remote sensing, clouds often obstruct the view of Earth's surface, making it difficult to perform accurate analysis or monitoring of surface characteristics. Therefore, detecting and filtering out clouds is crucial in many fields, such as meteorology, environmental monitoring, and climate research.

2. Cloud detection can be performed using various methods, including:

- Machine Learning and Deep Learning Models: Modern techniques use advanced models like convolutional neural networks (CNNs) or DEtection TRansformers (DETR) to detect clouds more accurately. These models can identify different types of clouds and deal with complex or overlapping structures.

3. What are Remote Sensing Images?

- Remote sensing images are pictures of Earth or other objects captured from a distance, typically by satellites, drones, or aircraft. These images provide data across various spectral bands (visible light, infrared, microwave, etc.), allowing us to observe and study different characteristics of Earth's surface and atmosphere.

4. Remote sensing images are widely used for:

- Earth Observation: Monitoring land use, vegetation, water bodies, and urban development.
- Weather and Climate Monitoring: Analyzing atmospheric conditions, including cloud cover, temperature, and moisture.
- Disaster Management: Tracking natural disasters like floods, fires, and hurricanes.

5. Cloud Detection in Remote Sensing Images Using DETR:

Cloud detection in remote sensing involves identifying and categorizing cloud formations in satellite imagery. Clouds can obstruct visibility and affect analyses of Earth's surface. Therefore, accurate cloud detection is crucial for applications like land use monitoring, weather forecasting, and climate studies.

How DETR is Used for Cloud Detection:
1. Object Detection Capability: Since clouds can be considered as "objects" in satellite images, DETR's object detection functionality is adapted to identify different types of clouds. It assigns a class label to each detected cloud, such as cirrus, cumulus, or stratus clouds.

2. Complex Cloud Structures: Traditional methods often struggle to detect and classify clouds with complex or overlapping shapes. DETR, with its transformer-based architecture, excels in such cases by attending to different parts of the image and distinguishing between various cloud types.

3. Filtering Out Similar Cloud Types: One challenge in cloud detection is dealing with cloud types that have similar shapes but different physical properties. DETR's attention mechanism helps filter out these similarities and focus on the unique characteristics of each cloud type.

4. Improving Detection Accuracy: By using deep learning, especially a powerful model like DETR, the detection of clouds becomes more robust, minimizing errors from occlusion, false positives, or poor contrast in satellite images.

Applications:
- Weather Forecasting: By accurately detecting cloud types, meteorologists can better predict weather patterns.
- Earth Observation: Cloud detection helps scientists remove or correct cloud-covered areas in satellite imagery, improving the quality of surface analysis.
- Climate Studies: Understanding cloud distribution and types is crucial for climate monitoring and modeling.

II. Problem Definition

The problem is object detection for clouds by using satellite imagery. Specially, there are four different types of clouds we want to detect:
- Gravel Clouds: Clouds resembling gravel in texture - (Cirrostratus).
- Fish Clouds: Cloud structures that resemble fish scales - (Cumulus).
- Sugar Clouds: Thin clouds with granular textures, similar to sugar particles - (Altocumulus).
- Flower Clouds: Floral-like cloud structures - (Stratocumulus).

The key challenge we need to deal with is the difference between four cloud types and filter them

III. Dataset

the dataset for research by using satellite images and corresponding annotations (bounding boxes and labels for the clouds). The dataset structure is as follows:
- Images: Stored in a directory containing the satellite images in JPEG format.
- Annotations: A JSON file mapping each image to bounding box coordinates and cloud type labels.

The dataset contains 70% train set, 20% validation set, 10% test set.

Example JSON:


```
"images": [ { "id": 0,
              "license": 1,
              "file_name": "a3c4ec6_jpg.rf.00e4bf7a6ab087995f9040c250a6df8c.jpg",
              "height": 640,
              "width": 640,
              "date_captured": "2024-09-26T04:16:54+00:00" },
```


Each label corresponds to one of the cloud types:
- 1: Gravel Clouds
- 2: Fish Clouds
- 3: Sugar Clouds
- 4: Flower Clouds

IV. Model: DEtection TRansformer (DETR)

1. What is DETR?

DEtection TRansformer (DETR) is a deep learning model designed for object detection tasks. It was introduced by Facebook AI Research and combines transformers (originally used in natural language processing) with convolutional neural networks (CNNs) for image feature extraction. The unique feature of DETR is its ability to directly predict bounding boxes and classify objects in images without the need for traditional methods like region proposals or anchor boxes, which are common in object detection models like Faster R-CNN.

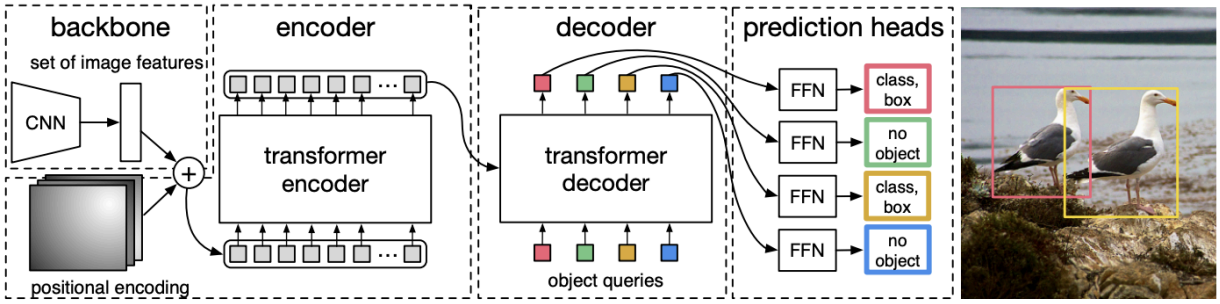


Fig. 2: DETR uses a conventional CNN backbone to learn a 2D representation of an input image. The model flattens it and supplements it with a positional encoding before passing it into a transformer encoder. A transformer decoder then takes as input a small fixed number of learned positional embeddings, which we call *object queries*, and additionally attends to the encoder output. We pass each output embedding of the decoder to a shared feed forward network (FFN) that predicts either a detection (class and bounding box) or a "`no object`" class.

2. Key Features of DETR:

1. End-to-End Object Detection: DETR directly predicts the object class and its position (bounding box) in the image.

2. Attention Mechanism: The transformer-based architecture allows DETR to focus on specific parts of an image, making it better at detecting complex or overlapping objects.

3. Simplified Architecture: Unlike traditional object detectors that rely on many handcrafted processes (like anchor generation), DETR uses a more straightforward design while maintaining high accuracy.

3. Key components of DETR:

1.Encoder: Processes the input image into a series of feature maps.

2.Decoder: Predicts object classes and corresponding bounding boxes.

3.Set-Based Loss: Assigns ground-truth objects to predicted objects with a bipartite matching algorithm, improving detection accuracy for similar objects.

V. Training process

1. Data Preparation

- A unique CloudDetectionDataset class loads the dataset and gets the photos and annotations ready for the DETR model. The DetrForObjectDetection from Roboflow's transformers library is used to process each image.
- The dataset is divided into training and validation sets.

Code:

```
image_processor = DetrImageProcessor.from_pretrained(CHECKPOINT)
```

2. Model training

- Using the cloud detection dataset, a pre-trained DETR model (facebook/detr-resnet-50) is refined.
- AdamW is used to optimize the model at a 5e-5 learning rate. To prevent overfitting, a learning rate scheduler lowers the learning rate every ten epochs.
- The training loop computes the loss for each batch and uses backpropagation to update the model weights during each ten epoch iteration.

Code:

```
for epoch in range(num_epochs): model.train() running_loss = 0.0
for batch in train_loader: pixel_values, labels = batch outputs
= model(pixel_values=pixel_values, labels=labels) loss =
outputs.loss loss.backward() optimizer.step()
optimizer.zero_grad() # Zero the gradients after updating the
model running_loss += loss.item()
```

3.Validation

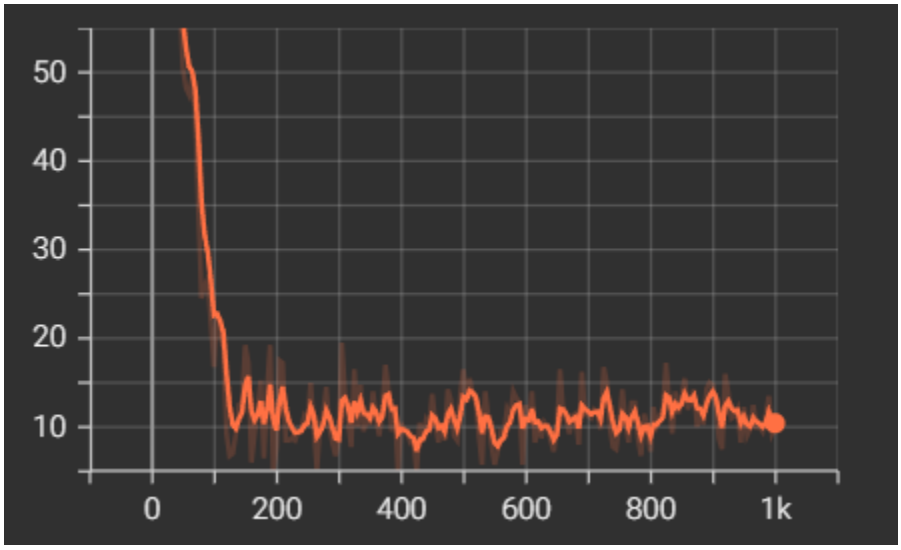
- The validation dataset is used to evaluate the model once it has been trained. The trained model is used to make predictions.

Code:

```
with torch.no_grad():

      #load image and predict

      inputs = image_processor(images=image,
return_tensors='pt').to(DEVICE)

      outputs = model(**inputs)

      # post-process

      target_sizes = torch.tensor([image.shape[:2]]).to(DEVICE)

      results = image_processor.post_process_object_detection(

            outputs=outputs,

            threshold=CONFIDENCE_TRESHOLD,  # Make sure
CONFIDENCE_TRESHOLD is defined

            target_sizes=target_sizes

      )[0]
```

VI.Experiment results

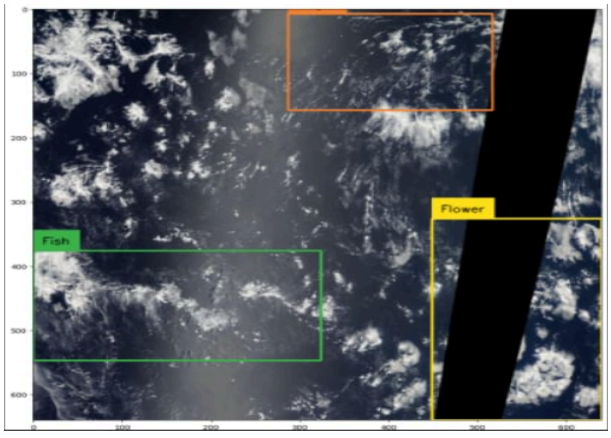
1. Train cardinality error



- Initial High Error: The cardinality error started high (~50) in the early iterations, showing that initial difficulty in matching predicted bounding boxes with ground truth.
- Rapid Decrease: Error drops significantly within the first 200 iterations, suggesting the model quickly learned object localization.
- Stabilization: After 200 iterations, the error stabilizes around 10, showing consistent performance but potential room for further improvement.

2. Visual Results

Below are the visual results for an example image processed by the model:



The bounding boxes in different colors (blue for Gravel, green for Fish, yellow for Sugar, and orange for Flower clouds) demonstrate how the model recognizes and classifies cloud types in the image.

VII. Discussion

The findings show that DETR has the capacity to recognize objects in remote sensing images. The main observations are:

- **Simplified process**: DETR's transformer-based method eliminates the need for NMS and region suggestions, simplifying and optimizing the object detection process.
- **Scalability**: The model performs well across a range of item scales, however more enhancements are required for detecting very minute things
- **Global Context Understanding**: The transformer architecture enables the model to capture long-range dependencies, which is critical in complicated scenes containing overlapping items.

Limitations:

- **Computation**: Unsure of validation accuracy
- **Confidence threshold tuning**: Setting the detection confidence level to 0.5 may have excluded some correct predictions with slightly lower confidence. Tuning this threshold for various cloud types may produce better results.

VIII. Conclusion

In this project, we effectively applied the DEtection TRansformer (DETR) model for identifying clouds in satellite imagery. The model performed well, accurately detecting and classifying various cloud types, even when their shapes were very similar.Unfortunately we are not yet able to calculate validation accuracy. With further optimization and data augmentation, this DETR-based method holds promise for remote sensing applications, offering a reliable and automated solution for cloud detection.

IX. References

Simplon “Cloud_classification Computer Vision Project” :
https://universe.roboflow.com/simplon-xajv7/cloud_classification-cjuda

https://paperswithcode.com/method/detr