University of science and technology of Hanoi

DEEP LEARNING PROJECT

DINOv2 for
remote sensing image classification

Group 16 - ICT B3 Class 1:

Trần Minh An - 22BI13007
Đỗ Quang Anh - 22BI13013
Hoàng Quỳnh Anh - 22BI13015
Cao Phương Đăng - 22BI13071
Nguyễn Quý Đức - 22BI13094
Phạm Quang Dương - 22BI13116

Table of contents

1. Introduction

1.1. Background

Remote sensing involves collecting data about the Earth's surface from a distance, often through satellite or aerial imagery. This technology is essential for many applications, including environmental monitoring, urban planning, agricultural assessment, and disaster response. However, using remote sensing data for image classification poses several notable challenges.

First, the high dimensionality of remote sensing data can introduce learning complexity for traditional classification algorithms. As the number of features increases, the amount of training data required to achieve reliable classifications increases exponentially. As a result, models may have difficulty generalizing to new data, which can lead to overfitting.

Additionally, variations in environmental conditions—such as changes in illumination, atmospheric effects, and seasonal changes—can adversely affect the quality of the collected imagery. These factors introduce noise and further complicate the classification task.

1.2. DINOv2: Overview

DINOv2 (unlabeled self-distillation), is an advanced self-supervised learning framework that focuses on extracting meaningful representations from images using Vision Transformers (ViTs). A key aspect of DINOv2 is the use of knowledge distillation, where a smaller learner model learns from a larger teacher model, refining its ability to learn features efficiently. This approach allows DINOv2 to capture complex patterns and relationships in image data, making it an attractive choice for many applications.

DINOv2's architecture consists of multiple layers with self-attention mechanisms and feedforward networks. This design enables efficient processing of input images, allowing the model to understand complex data structures and extract relevant features. One of the most important advantages of DINOv2 is its self-supervised nature, meaning it can learn directly from unlabeled datasets, which is especially beneficial in situations where collecting labeled data is challenging or resource-intensive.

1.3. Objective

The objective of this report is to explore how DINOv2 can be applied to remote sensing image classification. Through this analysis, we hope to illustrate the strengths of DINOv2 in autonomously extracting meaningful features from complex imagery, thereby improving the reliability of land cover classification. We aim to shed light on how this approach can support more effective monitoring and management of natural resources, ultimately contributing to better environmental stewardship and informed decision-making in various sectors, including agriculture, urban planning, and disaster response.

2. Dataset

2.1. Description

In this study, we used the EuroSAT dataset, a collection of high-resolution satellite images representing 10 distinct land cover classes, including forest, water, urban, agricultural, bare land, perennial crops, herbaceous vegetation, shrubland, snow and ice, and wetlands.

To achieve a comprehensive representation of each land cover class, we carefully selected 200 images from each class for the training phase, resulting in a total of 800 training images. For the testing phase, we collected 10 images from each class, resulting in a test set of 40 images and we also added some images from other datasets then the total of testing images is 100.

Class	***Training Images***	***Testing Images***
Annual Crops	200	30
Forest	200	40
HerbaceousVegetation	200	30
Industrial	200	40
Total	800	100

Table 1: Class Distribution in the EuroSAT Dataset

Image 1: Examples from EuroSAT Dataset (Annual Crops, Forest and Industrial)

2.2. Preprocessing

Before inputting images into the model, we executed several preprocessing steps to enhance their quality and ensure uniformity:

- **Resizing**: All images were resized to a standardized resolution of 224x224 pixels. Maintaining consistent input dimensions is crucial for the model to process images effectively.
- **Normalization**: Pixel values were normalized to a range between 0 and 1. This step is vital for improving convergence speed during training and stabilizing the learning process. We use (mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]) which is ImageNet normalization.

- **Data Augmentation**: To bolster the model's robustness and enhance its generalization capabilities, we applied data augmentation techniques, including random rotations, flips, and color adjustments. By artificially increasing the training set's diversity, we aimed to mitigate overfitting and enhance the model's performance on unseen data.
- **Dataset Splitting**: The dataset was partitioned into training, validation, and test sets to ensure the model is evaluated on data it has not encountered during training. The training set comprised 1,000 images (100 per class), the validation set included 200 images (20 per class), and the test set contained 100 images (10 per class).

Image 2: Examples of Test Dataset Images (Annual Crop, Forest)

Image 3: Examples of Training Dataset Images (Annual Crop, Forest)

3. Model Implementation

3.1. Model Architecture

The architecture of DINOv2 can be outlined as follows:

- **Input Layer:** This layer accepts images resized to 224x224 pixels, ensuring consistency in input sizes for efficient processing.

- **Embedding Layer:** This layer transforms the input image into a higher-dimensional feature space, allowing the model to capture more complex details and representations.

- **Transformer Blocks:** DINOv2 consists of several layers of transformer blocks, each equipped with self-attention mechanisms. These blocks allow the model to learn contextual relationships in the image data, effectively understanding how different regions of the image relate to each other.

- **Classification Head:** At the top of the architecture is a fully connected classification head, which predicts probability distributions over 10 target classes. This head integrates the features learned from previous layers to make educated predictions about the land cover types depicted in the input images.

By using the DINOv2 architecture, we want to exploit its strengths in efficiently processing and classifying remotely sensed images, and improve our analysis and understanding of land cover dynamics.

3.2. Training Process

The Pre-trained model used in this project is ‘facebook/dinov2-base-imagenet1k-1-layer’ from huggingface.
The training of the DINOv2 model was conducted on a remote sensing dataset using Google Colab with GPU support, ensuring efficient computation. For this process, we employed a learning rate of 0.0001, a batch size of 32, and carried out training over a span of 10 epochs.

The dataset was meticulously organized, with 200 images selected for each of the 4 land cover classes, resulting in a balanced training set of 800 images. This careful curation helps the model learn effectively from a diverse range of examples.

To monitor the model's performance throughout the training process, we tracked both training accuracy and loss. This monitoring enables us to evaluate the effectiveness of our training strategy and make adjustments if necessary, ensuring the model was learned as intended.

By focusing on these parameters, we want to fine-tune the model's ability to classify remote sensing images accurately, ultimately enhancing its performance in real-world applications.

3.3. Loss Function and Optimizer

We used the cross-entropy loss function, which is standard for multi-class classification tasks, and the Adam optimizer which is the most popular recently with the learning rate 0.0001 for training. Cross-entropy loss measures the difference between the predicted and true class distributions, while Adam provides efficient gradient updates to optimize the model.

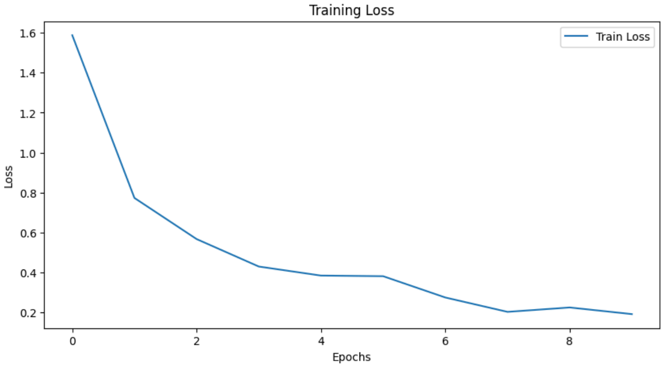


Image 4: Loss over Epochs

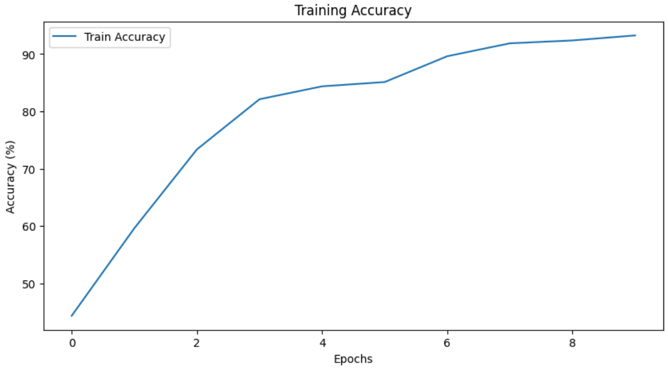


Image 5: Accuracy over Epochs

4. Results

4.1. Confusion Matrix

A confusion matrix was generated to visualize the model's performance across different classes. The confusion matrix offers insights into which classes were frequently misclassified and highlights areas where the model may require improvements.

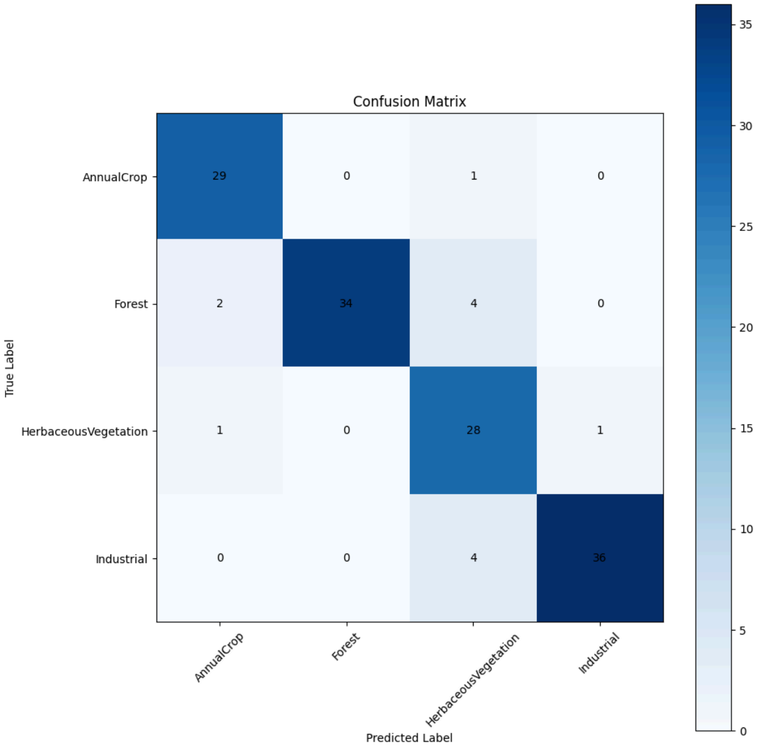
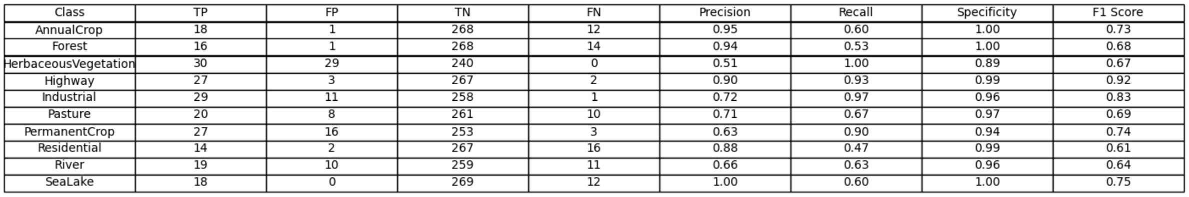


Image 6: Confusion Matrix

4.2. Performance Metrics

To evaluate the model's performance, we calculated various metrics, including accuracy, precision, recall, and F1-score. These metrics offer a comprehensive overview of the model's performance across different classes.

- **Accuracy:** The overall accuracy of the model across all test images.
- **Precision:** The ratio of true positive predictions to the total positive predictions, reflecting the model's ability to accurately identify a specific class.
- **Recall:** The ratio of true positive predictions to the total actual positives, measuring the model's ability to capture all instances of a class.
- **F1-score:** The harmonic mean of precision and recall, providing a balanced view of the two metrics.



Class	TP	FP	TN	FN	Precision	Recall	Specificity	F1 Score
AnnualCrop	18	1	268	12	0.95	0.60	1.00	0.73
Forest	16	1	268	14	0.94	0.53	1.00	0.68
HerbaceousVegetation	30	29	240	0	0.51	1.00	0.89	0.67
Highway	27	3	267	2	0.90	0.93	0.99	0.92
Industrial	29	11	258	1	0.72	0.97	0.96	0.83
Pasture	20	8	261	10	0.71	0.67	0.97	0.69
PermanentCrop	27	16	253	3	0.63	0.90	0.94	0.74
Residential	14	2	267	16	0.88	0.47	0.99	0.61
River	19	10	259	11	0.66	0.63	0.96	0.64
SeaLake	18	0	269	12	1.00	0.60	1.00	0.75

Table 2: Performance Metrics Table

4.3. Case Study

Here is some of our test case:

Image	***Class***	***Test Result***
	Sea Lake	Sea Lake
	River	Pasture
	Residential	Residential
	Permanent Crop	Permanent Crop

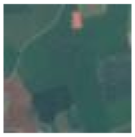
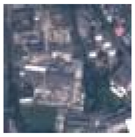
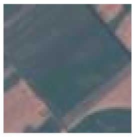
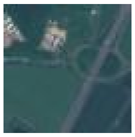
	Pasture	Pasture
	Industrial	Residential
	Annual crop	Annual Crop
	Highway	Highway
	Herbaceous Vegetation	Herbaceous Vegetation
	Forest	Sea Lake

Table 3: Some case study

When we tried to classify some images using this method, we found that there were some errors that could occur. There are 2 or 3 of 10 images tested that returned incorrect results.

There are a few reasons why these errors can occur in the classification of remote sensing images. The first reason is image quality. Remote sensing images often contain noise, artifacts, or varying lighting conditions, which can complicate feature extraction and lead to misclassification. Additionally, class overlap can present a challenge; similar land cover types, such as different vegetation types, can look similar, making it difficult for the model to distinguish between them.

The model's ability to generalize can also be limited if the training data is not diverse enough. Images in the test set that differ significantly from the training examples - perhaps due to seasonal changes or new environmental conditions—can make it difficult for the model to classify accurately.

5. Analysis and Discussion

5.1. Model Strength

The DINOv2 model has several strengths that improve its performance in remote sensing image classification. One of its most notable advantages is its ability to leverage supervised self-learning. This feature allows the model to learn from large amounts of unlabeled data, making it particularly valuable in remote sensing applications where collecting labeled datasets can be difficult and resource-intensive.

In addition, the DINOv2 architecture, with its Vision Transformer (ViT) backbone, excels at capturing complex spatial relationships and contextual information in images. This ability is critical in remote sensing, where different land cover classes can exhibit complex relationships that are critical for accurate classification.

Additionally, DINOv2's scalability allows it to efficiently handle large datasets, which is essential when processing high-resolution imagery collected over large geographic areas. Models can be trained on multiple GPUs, facilitating faster convergence and allowing experimentation with larger datasets.

5.2. Challenges and Limitations

Despite its strengths, the DINOv2 model faces several challenges in remote sensing imagery classification tasks.

- **Data quality and variability**: The performance of DINOv2 can be significantly affected by the quality and variability of input data. High-resolution satellite imagery often contains noise, artifacts, and varying lighting conditions, which can complicate feature extraction and reduce classification accuracy. If the training dataset doesn't accurately reflect real-world conditions, the model may struggle to generalize effectively.

- **Interpretability**: Like many deep learning models, DINOv2 functions as a "black box," making it difficult to interpret its decision-making process. This lack of transparency is a concern in critical applications such as environmental monitoring and urban planning, where stakeholders need to trust the model's predictions.

5.3. Comparison

5.3.1. CPU vs. GPU

In remote sensing image classification with the DINOv2 model, the choice between CPU and GPU significantly impacts efficiency.

- **Architecture and Design**: CPUs excel at sequential tasks, while GPUs, with their parallel processing capabilities, are better suited for large-scale data like image classification.

- **Performance**: GPUs significantly reduce training time for deep learning tasks due to their ability to handle millions of operations simultaneously, while CPUs are slower because of their sequential processing.
- **Memory Bandwidth**: GPUs offer higher memory bandwidth, enabling faster data processing for large image datasets, while CPUs may experience bottlenecks.
- **Energy Efficiency**: GPUs are more energy-efficient for deep learning tasks, consuming less power per calculation than CPUs.
- **Ease of Use**: CPUs are easier to integrate with existing software, while GPUs may require specialized libraries like PyTorch or TensorFlow, though these have become more accessible.
- **Application in Remote Sensing**: GPUs are ideal for real-time processing and managing large datasets, while CPUs work better for smaller datasets or simpler models.

In summary, GPUs are preferred for DINOv2 remote sensing image classification due to their superior speed and efficiency, while CPUs are suitable for simpler tasks.

5.3.2. DINOv2 vs. CNNs

When classifying remote sensing images, DINOv2 and CNNs have some strengths and limitations:

- **Feature Learning**: DINOv2 uses self-supervised learning to extract features from unlabeled data, capturing complex patterns. CNNs rely on labeled data and focus on local features, limiting their broader context understanding.
- **Architecture**: DINOv2, based on Vision Transformers, models long-range dependencies for comprehensive image understanding. CNNs use hierarchical convolutional layers but struggle with global context.
- **Data Requirements**: DINOv2 excels with limited labeled data, using large amounts of unlabeled data, while CNNs need substantial labeled datasets.
- **Generalization**: DINOv2 generalizes well to unknown data, whereas CNNs risk overfitting with limited data.
- **Computational Efficiency**: DINOv2 is more computationally demanding, while CNNs are generally more efficient and suitable for resource-limited environments.

6. Conclusion

6.1. Summary

We explore the application of DINOv2 to classify remotely sensed images, finding its advantages over traditional methods. Remote sensing is important for fields such as environmental monitoring, urban planning, and agriculture.

DINOv2, using self-supervised learning and Vision Transformers, efficiently processes high-dimensional, variable, and unlabeled datasets. Compared with CNN, DINOv2 excels in capturing global context and generalizing with limited labeled data. In addition, GPUs outperform CPUs in processing large-scale datasets, resulting in better speed and efficiency.

Overall, DINOv2 improves the accuracy and efficiency of remotely sensed image classification, improving environmental monitoring and analysis.

6.2. Future Work

Although DINOv2 has shown promising results in remote sensing image classification, future research can enhance its performance.

- **Improving preprocessing and data augmentation**: Advanced techniques like synthetic oversampling, normalization, and image augmentation can improve robustness and reduce noise.
- **Diversifying training samples**: Transfer learning and fine-tuning DINOv2 on datasets from different regions can improve its adaptability to varied environments.
- **Integrating with other models**: Combining DINOv2 with CNNs can leverage both local and global feature extraction, improving classification accuracy.
- **Enhancing training with limited data**: Exploring self-supervised and semi-supervised learning or using synthetic data can improve performance with smaller labeled datasets.

7. References

1. Facebook Research. (2023). DINOv2. GitHub Repository. Retrieved from https://github.com/facebookresearch/dinov2
2. Hugging Face. Pre-trained DINOv2 Model. Retrieved from https://huggingface.co/models?search=facebook/dinov2
3. Oquab, M., Darcet, T., Moutakanni, T., Vo, H., Szafraniec, M., Khalidov, V., Haziza, D., Massa, F., El-Nouby, A., Caron, M., Schmid, C., & Jegou, H. (2023). DINOv2: Learning Robust Visual Features without Supervision. arXiv. Retrieved from https://arxiv.org/html/2304.07193v2
4. Picsellia. (2023). DINOv2 Steps-by-Steps Explanations. Picsellia Blog. Retrieved from https://www.picsellia.com/post/dinov2-steps-by-steps-explanations-picsellia
5. Jalammar, M. (2018). The Illustrated Transformer. Retrieved from https://jalammar.github.io/illustrated-transformer/
6. Facebook. (2023). The Curious Machine - Visual Explanation of DINOv2 [Video]. Facebook. Retrieved from https://www.facebook.com/thecuriousmachine/videos/531241692641792/?vh=e&mibextid=WC7FNe&rdid=KyHm8nDN1opEiqK6
7. YouTube. (2023). DINOv2 Explained [Video]. Retrieved from https://www.youtube.com/watch?v=j3VNqtJUoz0
8. YouTube. (2023). Deep Learning with Vision Transformers [Video]. Retrieved from https://www.youtube.com/watch?v=Pa2yTy48JA8&t=284s
9. Neptune. (n.d.). *How to Use Google Colab for Deep Learning: Complete Tutorial*. Retrieved from https://neptune.ai/blog/how-to-use-google-colab-for-deep-learning-complete-tutorial.

   This comprehensive tutorial provides a step-by-step guide on using Google Colab for deep learning projects. It covers setting up environments, utilizing GPUs, and implementing various deep learning frameworks, making it an essential resource for practitioners.
10. Zenodo. (2022). *Remote Sensing Dataset for Deep Learning*. Retrieved from https://zenodo.org/records/7711810#.ZAm3k-zMKEA.

    This record on Zenodo presents a deep learning approach utilizing the DINOv2 model specifically for remote sensing applications. It includes datasets, methodologies, and results, contributing valuable insights to the field.