# Wilfire detection

AOYAGI Hugo,
AYMES Malo,
FIZYCKI Tom,
THIAW Mouhamadou Lamine Bara

February 20, 2025

**Abstract**

Wildfires are frequent phenomena posing significant risks to human life and the environment. To address the challenge of early detection, we present in this project a data-efficient computer vision based approach leveraging a small dataset and large scale pretrained foundation models. Our experiments achieve F1-scores exceeding 98 % highlighting the feasibility of adapting existing models for accurate and cost-effective wildfire detection at scale.

## 1 Introduction

Wildfires represent a growing global threat causing multiple ecological damages and property losses. Early and accurate detection is a critical step in mitigating their impact, but this process poses substantial challenges. Traditional fire detection methods based on human observations and sensors are limited by infrastructure costs and coverage constraints. Then, the emergence of computer vision models with the help of satellite imagery has appeared to be a more scalable solution, enabling continuous and wide-area monitoring of potential outbreaks. However, developing such models requires large, high-quality datasets that capture varying fire conditions, landscapes, and climatic factors, which are not easy to obtain. One of the project constraints was to use only validation data for training, providing us with 5040 samples, which places our work in the frugal learning domain, discussed in [DS22] on a different task, but using similar amounts of data. This is what led us to explore different alternatives such as pre-trained foundation models.

This report outlines the motivation behind our work.

All the codes are available **here** in our GitHub repository.

## 2 Problem Statement

For this project, we are thus given a dataset with RGB satellite images (350px per 350px) of various places and environments, the images are split in two classes: wildfire and nowildfire. The objective is to provide a binary classification model that takes as input an image and assigns one of two classes. In order to rank our model amongst the many possible solutions, we use the accuracy and F1 metrics on our new validation set.

The original dataset contains around 43000 images, however, for this project, one of the constraint was to limit any supervised training to the validation set which is to be split into a new train and validation set. The entire set can however be used for unsupervised training.

To overcome this limitation on our dataset and still achieve good performance, we propose to leverage pre-trained foundation models which were trained on very large amounts of data. Using a pretrained model can help classify the "hard" cases the model might encounter in the wildfire dataset, such as images containing clouds, containing many man-made infrastructure or rare that might appear only a few times in our dataset. Many models or configurations of foundation models trained on satellite images exist, we will thus need to compare them to find the one most suitable for our use case.

## 3 Method and Experiments

The issue we faced as the small amount of data (5040 images), much less than would be required to train a Deep Neural Network capable of learning meaningful representations with a good ability to generalize. In order to deal with this issue, we choose to leverage **pre-trained foundation models**, trained on a massive amount of data and with versatile features that could be used for our task.

Given the type of task, we chose the **SatlasPretrain** [BWG+23] models, a set of foundation models trained on over 30TB of remote sensing data from multiple satellite constellations, include a lot of historical data, including a lot of RGB satellite imagery similar to our training data. With 137 label types, these foundation models (*SatlasNet*) were trained to compute image features to be used by some very different tasks: semantic segmentation (per-pixel classification), per-pixel regression, object detection, instance segmentation, polyline prediction, prediction of object (point, polyline, polygon) properties, and **image classification**. These models achieved great performance on a number of baselines, and, given the nature of our task and data, perfectly fit our needs.

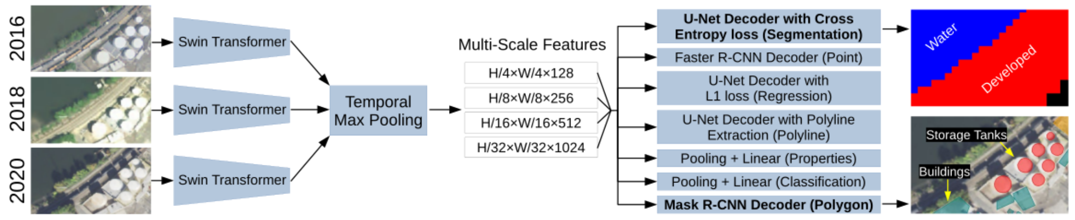
2016
2018
2020
Swin Transformer
Swin Transformer
Swin Transformer
Temporal Max Pooling
Multi-Scale Features
H/4×W/4×128
H/8×W/8×256
H/16×W/16×512
H/32×W/32×1024
U-Net Decoder with Cross Entropy loss (Segmentation)
Faster R-CNN Decoder (Point)
U-Net Decoder with L1 loss (Regression)
U-Net Decoder with Polyline Extraction (Polyline)
Pooling + Linear (Properties)
Pooling + Linear (Classification)
Mask R-CNN Decoder (Polygon)
Water
Developed
Storage Tanks
Buildings

Figure 1: Training architecture of one multi-image *SatlasNet*: each task is assigned a prediction head. *Source: github.com/allenai/satlas/*

The team behind *Satlas* created a number of different models, available on their open-source Github github.com/allenai/satlaspretrain_models. Given that the task on our data consists in classifying single RBG images, we are restricted to using the models trained on single-image RBG data (while some models use multiple images and more spectral channels). We then add a classification head, which consists in a simple MaxPooling+Linear module with Softmax activation.

With those models, we decided to study the different options for affixing and training a classification head to a *SatlasNet* and potentially finetuning the model. Those options were:

- **Pre-trained Backbone:**

  We choose between a ResNet50 [HZRS15] backbone, or a Swin Transformer [LLC+21] backbone (*Base* or *Tiny*). This backbone computes multi-scale features, at different resolutions.

- **FPN:**

  Every model can include a Feature Pyramid Network [LDG+17] to combine the different representations (coarse to fine-grained) to a set of features with the same number of channels.

- **Finetuning:**

  In addition to training the classification head, we can choose to finetune the backbone and/or the FPN (if present). This comes with additional computational costs.

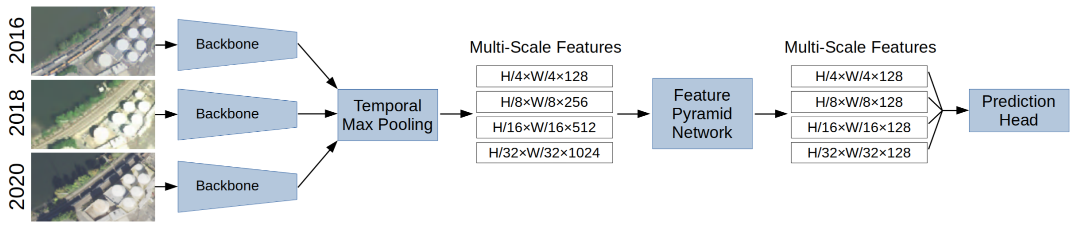

Figure 2: Architecture of a multi-image *SatlasNet* with FPN. Single-image simply discard temporal max pooling. *Source: github.com/allenai/satlaspretrain_models/*

To ensure the validity of our experiments, we loaded the data from the original validation set, and computed a single train-val split. All models were trained and validated on the exact same data, and with the same hyperparameters (learning rate, batch size, number of epochs...). We use the accuracy and F1-score metrics for evaluation.

## 4 Results and Discussion

| **Model ID** | **FPN** | **Train Backbone** | **Train FPN** | **Accuracy** | $F_1$ | $F_1$ **Gain (%)** |
|---|---|---|---|---|---|---|
| | | | **Sentinel2_SwinB_SI_RGB** | | | |
| Full Training | Yes | Yes | Yes | 0.984 | 0.981 | +4.15 |
| Frozen + FPN | Yes | No | Yes | 0.969 | 0.957 | +1.64 |
| Finetune No FPN | No | Yes | No | 0.960 | 0.952 | +1.12 |
| Frozen No FPN | No | No | No | 0.949 | 0.942 | - |
| | | | **Sentinel2_SwinT_SI_RGB** | | | |
| Full Training | Yes | Yes | Yes | 0.979 | 0.978 | +3.56 |
| Frozen + FPN | Yes | No | Yes | 0.968 | 0.964 | +2.10 |
| Finetune No FPN | No | Yes | No | 0.963 | 0.958 | +1.52 |
| Frozen No FPN | No | No | No | 0.949 | 0.944 | - |
| | | | **Sentinel2_ResNet50_SI_RGB** | | | |
| Full Training | Yes | Yes | Yes | 0.947 | 0.926 | -2.89 |
| Frozen + FPN | Yes | No | Yes | 0.964 | 0.960 | +0.70 |
| Finetune No FPN | No | Yes | No | 0.960 | 0.946 | -0.78 |
| Frozen No FPN | No | No | No | 0.958 | 0.953 | - |

Table 1: Comparison of different training strategies across models. Are presented accuracy, $F_1$ scores, as well as the relative $F_1$ improvement compared to a frozen backbone with no FPN.

The results demonstrate that full training of the models, where both the backbone and the Feature Pyramid Network (FPN) are trained, consistently achieves the highest accuracy and $F_1$-scores across all architectures. For instance, **SwinBase - Full Training** achieves an **accuracy of 98.41%** and an **$F_1$-score of 98.06%**, representing a **4.15% improvement in $F_1$-score** compared to its frozen, non-FPN, non-finetuned baseline. Similarly, SwinTiny achieves an **97.73%** $F_1$ when fully-finetuned, while the ResNet50 surprisingly underperforms compared to its frozen variant, showing an $F_1$-score decrease of 2.89%. Among the fine-tuning strategies, models that finetune the FPN while keeping the backbone frozen exhibit moderate performance gains, with SwinBase and SwinTiny improving by 1.64% and 2.10% in $F_1$-score, respectively. They here always outperform models that finetune the backbone during training, which, without FPN adjustments leads to slight improvements in Swin models but results in a minor $F_1$-score drop for ResNet50. These results highlight that while full training remains the best strategy for Swin models, fine-tuning the FPN alone can also provide competitive results with reduced computational cost.

Furthermore, we implemented various data-augmentation models to our dataset. However, none of those brought better performances on the full training approach. This can be explained by the already extremely varied pretrain dataset and the already high accuracy that is difficult to fine tune further.

## Conclusion

Finally, we find that using a pretrained model brings very satisfying performances that even exceed the performances of CNNs trained on the whole dataset of 43000 images. The Satlas Dataset is indeed extremely varied with more than 300 million labels on satellite images.

While those performances are very difficult to beat, the pretrained models available are very large and expensive to run. Other much smaller models such as simple CNNs can easily bring accuracies of around 96%. For most applications, the performance difference is not worth the environmental and financial impact of running foundation models.

## References

[BWG+23] Favyen Bastani, Piper Wolters, Ritwik Gupta, Joe Ferdinando, and Aniruddha Kembhavi. Satlaspretrain: A large-scale dataset for remote sensing image understanding, 2023.

[DS22] Sebastien Deschamps and Hichem Sahbi. Reinforcement-based frugal learning for satellite image change detection, 2022.

[HZRS15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[LDG+17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection, 2017.

[LLC+21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.