WE DIDN'T START THE FIRE: ROBUST WILDFIRE DETECTION WITH LIMITED LABELED DATA

Ivanina Ivanova, Abhay Mathur
Institut Polytechnique de Paris
{ivanina.ivanova, abhay.mathur}@ip-paris.fr

COURSE PROJECT FOR IA323

1 Problem Statement

This project aims to develop a fire detection system using a designated dataset. The dataset consists of three subsets: a training set, a validation set, and a test set. Specific constraints and guidelines must be strictly followed to ensure compliance with the project requirements.

1.1 Dataset Access and Constraints

The dataset is available for download from Kaggle [1]. It comprises a training set, a validation set, and a test set. A critical restriction is imposed on the training set: its labels are inaccessible. Any direct utilization of annotations from the training set will lead to disqualification.

1.2 Dataset Partitioning

To facilitate model training, the original validation set must be partitioned into a newly defined validation set and a new training set. The original training set may be leveraged in a creative manner; however, its labels must not be employed under any circumstances.

1.3 Model Development

A deep neural network (DNN) will be trained utilizing the newly defined training and validation sets. Various methodologies and supplementary resources may be incorporated to enhance model performance, provided that all constraints related to annotation usage are rigorously upheld.

2 Dataset Analysis

The dataset consists of satellite images of areas that previously experienced wildfires in Canada. According to the problem statement, we discard the labels of the training set and split the validation dataset into labeled training and validation sets in a 4:1 ratio. The resulting splits are as follows:

- Train: 5040 images
- Train unlabeled: 30249 images
- Val: 1260 images
- Test: 6299 images

[1] Wildfire Dataset: https://www.kaggle.com/datasets/abdelghaniaaba/wildfire-prediction-dataset/code

Figure 1: Some sample images from the dataset.

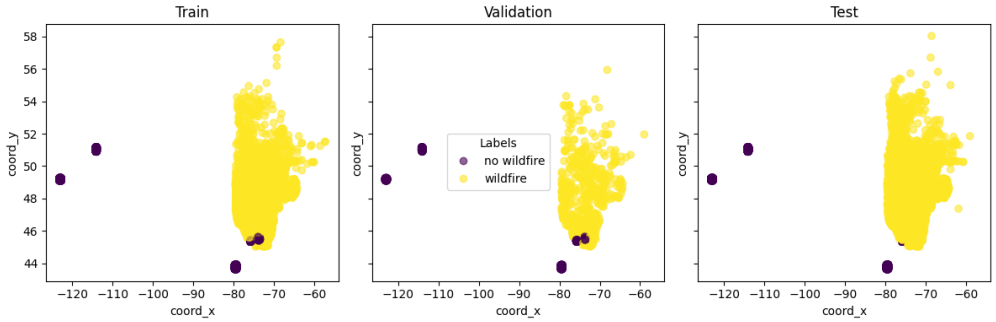


Figure 2: Spatial Distribution of the Dataset. All splits show a similar spread of zones where fires occured.

We show some representative images from the dataset in Figure 1. A visual inspection of the dataset showed that most wildfire images contain more vegetation, while non-wildfire images pertain to urban locations.

The coordinates of the location of the images can be extracted from the filename. We show the spread of images with respect to labels in Figure 2. This shows that there are some clusters consistent across splits where wildfires can be expected.

3 Baselines: Using Available Labeled Data

3.0.1 Naive Coordinates Classifier

A simple baseline that assigns labels based solely on coordinate information. This approach assumes that spatial location alone is a strong enough feature for classification.

3.0.2 Image Classifiers

We train a series of image classifiers using Convolutional Neural Networks (CNNs) to classify images based on their visual content. We experiment with different CNN architectures, including ResNet18, ResNet34, ResNet50, and ResNet101.

3.0.3 Fused Classifiers

We combine the coordinate-based classifier with the image classifiers to create a fused model that leverages both spatial and visual features for classification.

4 Learning from Unlabeled Data

In this section, we discuss our methods for learning from unlabeled data. We explore various self-supervised learning techniques to extract useful representations from the dataset. These representations are then used to train a classifier on the labeled data.

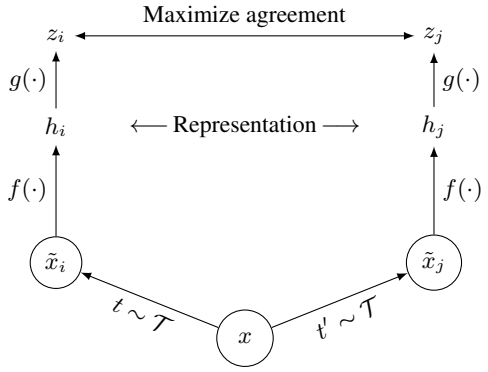


Figure 3: Framework for SimCLR [1]. Two separate data augmentation operators are sampled and applied to each data example to obtain two correlated views. A base encoder network $f(\cdot)$ and a projection head $g(\cdot)$ are trained to maximize agreement using a contrastive loss. After training is completed, we simply use the encoder $f(\cdot)$ and representation h for downstream tasks.

4.1 SimCLR

SimCLR [1] is a self-supervised learning method that learns visual representations by maximizing the agreement between differently augmented views of the same data example.

The training pipeline of SimCLR consists of the following steps:

4.1.1 Data Augmentation

Each input image is transformed into two different augmented views through a series of random augmentations such as cropping, resizing, flipping, color jittering, and Gaussian blurring. These augmentations are crucial as they create diverse views of the same image, which the model learns to recognize as similar.

4.1.2 Encoder Network and Projection Head

Both augmented views are passed through an encoder network (typically a ResNet) to obtain their respective feature representations. The encoder network is shared between the two views.

The feature representations are then passed through a small neural network called the projection head, which maps them to a latent space where the contrastive loss is applied. The projection head typically consists of a few fully connected layers.

4.1.3 Contrastive Loss

The core idea of SimCLR is to bring the representations of the two augmented views of the same image closer together in the latent space while pushing apart representations of different images. This is achieved using a contrastive loss function, specifically the normalized temperature-scaled cross-entropy loss (NT-Xent loss). The loss function is defined as:

$$\mathcal{L}_{i,j} = -\log \frac{\exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k \neq i]} \exp(\text{sim}(\mathbf{z}_i, \mathbf{z}_k)/\tau)} \tag{1}$$

Where $\mathbf{z}_i$ and $\mathbf{z}_j$ are the latent representations of the two augmented views of the same image, $\text{sim}(\cdot, \cdot)$ denotes the cosine similarity, τ is a temperature parameter, and N is the batch size.

4.2 Variational Autoencoder (VAE)

4.2.1 β-VAE

A Variational Autoencoder (VAE) was used as a self-supervised learning approach to extract latent representations from the training dataset. The VAE comprises an encoder network, which maps input images to a latent space distribution,

and a decoder network, which reconstructs the input from latent variables. The encoder network consists of a series of convolutional layers followed by fully connected layers, while the decoder network mirrors the encoder architecture. The VAE was trained using the unlabeled training dataset. The training objective that we decided to use is Beta-VAE [2]. Beta-VAE is a modification of the original VAE that introduces a hyperparameter β to the loss function. The β parameter controls the trade-off between the reconstruction loss and the Kullback-Leibler divergence. The loss function for the Beta-VAE is given by:

$$\mathcal{L}(\theta, \phi, x, z, \beta) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta D_{KL}(q_\phi(z|x)\|p(z)) \tag{2}$$

where $p(z|x)$ is the learned posterior, $p(z)$ is the prior distribution (assumed to be Gaussian), and $p(x|z)$ represents the likelihood of reconstructing the input image from the latent space. Well chosen β values can lead to disentangled representations in the latent space. The VAE was trained for 50 epochs using the Adam optimizer with a learning rate of 0.0001.

Latent Representation Classifier The latent representations obtained from the VAE were then used to train a classifier. The classifier consists of a two fully connected layer with a ReLU activation fucntion and a Dropout of 0.2 layer in between. The classifier was trained using the labeled train dataset. The training objective was to minimize the binary cross-entropy loss between the predicted and true labels. The linear classifier was trained for 20 epochs using the Adam optimizer with a learning rate of 0.0001.

4.2.2 ResNet-based Encoder

To further improve the quality of the learned latent representations, we experimented with a ResNet-based encoder. The idea is to replace the standard convolutional layers with a ResNet architecture, this modification will help in capturing richer hierarchical features and improved the overall reconstruction quality, leading to better downstream classification performance.

4.2.3 Vector Quantized Variational Autoencoder (VQ-VAE)

Additionally, we experimented with a Vector Quantized Variational Autoencoder (VQ-VAE) [3]. The difference between VQ-VAE and the standard VAE is that VQ-VAE is using a discrete latent space instead of a continuous one. Instead of learning a probabilistic distribution in the latent space, VQ-VAE learns a set of discrete latent embeddings. The encoding process involves mapping an input image to the closest embedding vector from a learned codebook, effectively quantizing the latent representations. The loss function for VQ-VAE consists of three terms:

$$\mathcal{L}(x, z) = \log p(x|z_q(x)) + |sg[z_e(x)] - e|_2^2 + \beta|z_e(x) - sg[e]|_2^2 \tag{3}$$

where z_e is the encoder output, z_q is the nearest codebook entry, and $sg[\cdot]$ is the stop-gradient operator. The first term represents the reconstruction loss, the second term ensures the encoder output stays close to the codebook entries, and the third term encourages the codebook entries to move toward the encoder outputs. The advantage of VQ-VAE is that it prevents posterior collapse, a common issue in VAEs where the latent representations become overly dependent on the prior, leading to poor representation learning. By enforcing a discrete latent space, VQ-VAE learns more interpretable and structured representations, which significantly improved the downstream classification task.

ResNet VQ-VAE Same as the VAE we implemented a ResNet-based encoder for the VQ-VAE (ResNet VQ-VAE). The combination of ResNet and VQ-VAE provided both robust feature extraction and structured latent representations, leading to improved performance in our classification task. We used the same classifier architecture as in the VAE experiments.

Our results demonstrate that moving from a standard VAE to a ResNet-based encoder and finally to VQ-VAE with ResNet led to progressively better latent representations and improved classification performance.

4.3 Pseudo-Labeling

Pseudo-labeling is a semi-supervised learning technique that leverages unlabeled data to improve model performance. The approach involves training an initial model on labeled data and then using its predictions on unlabeled samples as pseudo-labels. These pseudo-labeled samples are subsequently incorporated into the training process alongside the original labeled data, effectively increasing the dataset size and improving generalization.

In our experiments, we applied pseudo-labeling from larger models to simpler CNN-based architectures to assess its impact on classification performance.

5 Experiments and Results

5.1 Experimental Setup

To evaluate the effectiveness of different classification approaches, we conducted a series of experiments comparing models with and without pre-training. Our experiments were designed to assess the impact of feature extraction, coordinate-based classification, and various pre-training strategies on classification performance.

We first established baseline models using a simple Coordinate Classifier, a standard CNN, and a combined CNN + Coordinate Classifier. These models allowed us to determine the effectiveness of coordinate information in enhancing classification accuracy.

The Coordinate Classifier is an MLP with one hidden layer with 64 units and a ReLU activation function. The CNN model consists of four convolutional layers followed by two fully connected layers with ReLU activation functions. The CNN + Coordinate Classifier combines the CNN model with the Coordinate Classifier to leverage both feature-based and coordinate-based classification.

Next, we evaluated feature-based classifiers using different ResNet architectures (ResNet18, ResNet34, ResNet50, and ResNet101). We further extended this analysis by incorporating coordinate information alongside feature-based classification, measuring the improvements in accuracy and F1-score across varying model complexities.

To examine the impact of pre-training, we tested models trained with self-supervised and semi-supervised learning techniques, including SimCLR and pseudo-labeling. The SimCLR pre-training was applied to ResNet architectures to assess the benefits of contrastive learning. Results from the pre-training can be found in Table 1.

Table 1: Results of Pre-Training with SimCLR. The reconstructin loss is the one defined in Eq 1

Backbone	#params	Reconstruction Loss
ResNet18	11.50M	0.164
ResNet34	21.61M	0.158
ResNet50	27.97M	0.162
ResNet101	46.96M	0.182

We use the best model from the previous experiments (Feature + Coordinate classifier with ResNet34 backbone without pre-training) to generate pseudo-labels on the unlabeled training set. We then train a CNN model and a Coordinates+CNN model with an aggregation of the pseudo-labeled and the labelled set and evaluate their performance on the test set.

Additionally, we explored latent representation learning methods by training classifiers using β-VAE and VQ-VAE. These experiments aimed to determine whether unsupervised representation learning could provide a competitive alternative to direct feature extraction from supervised learning.

For all experiments, we record classification accuracy and F1-score as primary evaluation metrics. All experiments were conducted on a single NVIDIA A100 GPU with 40GB vRAM.

5.2 Results

We present our final results in Table 2. These results demonstrate the impact of model architecture, coordinate information, and pre-training strategies on classification performance, as measured by accuracy and F1-score.

The baseline models without pre-training reveal significant differences in performance. The Coordinate Classifier alone achieves an accuracy of 85.5%, which is considerably lower than other methods, highlighting its limited capacity to classify effectively based solely on coordinate information. In contrast, the standard CNN model without coordinate features attains a much higher accuracy of 94.46%. Notably, the combination of CNN and Coordinate Classifier further improves accuracy to 98.32%, demonstrating that incorporating coordinate features enhances classification performance.

Feature-based models utilizing ResNet backbones show consistently strong performance. Among them, ResNet50 achieves an accuracy of 98.5%, outperforming both ResNet18 and ResNet34. However, ResNet101 exhibits a slight

Table 2: Performance comparison of various models with and without pre-training.

Pre-Training	Model	Backbone	#params	Accuracy	F1 Score
—	Coordinate Classifier	—	**4.417K**	0.855	0.873
—	CNN	—	14.912M	0.9446	0.9491
—	CNN + Coordinate Classifier	—	14.913M	0.9832	0.9849
—	Feature Classifier	ResNet18	11.441M	0.983	0.984
		ResNet34	21.549M	0.98	0.982
		ResNet50	27.711M	0.985	0.986
		ResNet101	46.703M	0.978	0.98
—	Feature + Coordinate Classifier	ResNet18	11.442M	0.9916	0.9924
		ResNet34	21.550M	**0.9921**	**0.9928**
		ResNet50	24.560M	0.9906	0.9915
		ResNet101	43.552M	0.99	0.991
SimCLR	Feature Classifier	ResNet18	11.523M	0.98	0.982
		ResNet34	21.631M	0.976	0.978
		ResNet50	27.988M	0.984	0.985
		ResNet101	46.980M	0.984	0.985
Pseudo Labelling	CNN	—	14.912M	0.971	0.9737
	CNN+Coordinate Classifier	—	14.913M	0.9867	0.9879
β-VAE	Feature Classifier	-	72.21K	0.887	0.9
	Feature + Coordinate Classifier	-	73.43K	0.936	0.943
		ResNet18	27.83K	0.934	0.942
		ResNet34	48.04K	0.926	0.935
		ResNet50	55.64K	0.915	0.926
		ResNet101	93.62K	0.944	0.950
VQ-VAE	Feature Classifier	-	96.64K	0.931	0.936
	Feature + Coordinate Classifier	ResNet18	46.29K	0.906	0.920
		ResNet34	66.51K	0.915	0.924
		ResNet50	70.96K	0.959	0.968
		ResNet101	108.9K	0.956	0.955

drop in accuracy to 97.8%, suggesting potential overfitting or optimization challenges with deeper networks. When coordinate information is incorporated alongside feature-based classification, performance improves across all ResNet architectures. The best-performing model in this category is the Feature + Coordinate Classifier with ResNet34, which attains the highest accuracy of 99.21% and an F1-score of 99.28%, outperforming deeper ResNet variants. This suggests that ResNet34 provides an optimal balance of depth and generalization in this setting.

Pre-training strategies yield mixed results. SimCLR, a self-supervised contrastive learning approach, demonstrates potential benefits, but results for ResNet50, achieves an accuracy of 98.2%. While competitive, this performance does not surpass the best fully supervised models. Pseudo-labeling improves performance for CNN-based models, raising the accuracy of a standard CNN to 97.1% and CNN + Coordinate Classifier to 98.67%, indicating that leveraging unlabeled data can enhance classification but does not surpass the best fully supervised methods.

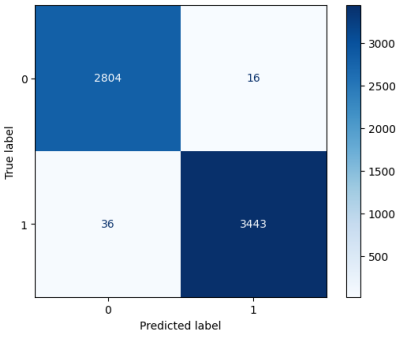
Latent representation learning methods such as β-VAE and VQ-VAE show comparatively lower performance. The best-performing β-VAE model with ResNet101 and coordinate information achieves 94.4% accuracy, while VQ-VAE with ResNet50 and coordinate information attains 95.9%. These results suggest that while variational autoencoders can learn meaningful representations, they do not outperform direct supervised learning approaches.

Overall, the highest accuracy is achieved by the Feature + Coordinate Classifier using a ResNet34 backbone, indicating that incorporating coordinate information alongside feature extraction is crucial for maximizing classification performance. Furthermore, while pre-training methods contribute to performance improvements, they do not surpass the best purely supervised models, suggesting that direct supervised learning remains the most effective strategy for this classification task.

5.3 Error Analysis

While our best model has an accuracy of more than 99%, it is essential to understand the errors made by the model. We analyze the confusion matrix of the best model and visualise the samples that were misclassified in Figure 4.

We observe that most false negatives occur when the image has occlusions, such as by clouds, or large patches of water or monochromatic fields. On the other hand, it is not entirely clear why the model misclassifies some of the false



(a) Confusion Matrix

(b) Misclassified Samples. Top and bottom rows show false positives and false negatives respectively.

Figure 4: Error Analysis: (a) Confusion Matrix of the best model, (b) Misclassified samples.

positives, but a visual inspection showed that these images contain a lot of vegetation and are visually similar to wildfire images.

6 Conclusion

In this study, we evaluated various classification models with and without pre-training to assess their performance in leveraging coordinate-based and feature-based learning. Our findings demonstrate that the incorporation of coordinate information significantly enhances classification performance across all architectures.

Pre-training methods such as SimCLR, pseudo-labeling, and variational autoencoders exhibit potential but do not consistently surpass the best fully supervised models. Among them, pseudo-labeling provides notable improvements for CNN-based models, while SimCLR achieves competitive performance with ResNet50. However, latent representation learning approaches, including β-VAE and VQ-VAE, yield lower accuracy, suggesting that their learned representations may not be as effective as direct feature extraction from supervised learning.

These results indicate that direct supervised learning with a well-structured feature extractor, complemented by coordinate information, remains the most effective approach for this classification task.

Future work could explore de-biasing the model to reduce errors on occluded images and to decorrelate wildfires with vegetation and investigate additional pre-training strategies to further enhance classification performance.

References

[1] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020.

[2] Christopher P. Burgess, Irina Higgins, Arka Pal, Loic Matthey, Nick Watters, Guillaume Desjardins, and Alexander Lerchner. Understanding disentangling in β-vae, 2018.

[3] Aäron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *CoRR*, abs/1711.00937, 2017.