February, 2025

IA23 - Deep Learning Based Computer Vision

Final Project Report

Eduardo Guimarães

Filipe Lacerda

Joaquim Minarelli Gaspar

Sami Kouzeiha

Self-Supervised Learning for Fire Detection

Introduction

For this project, the group received the task of training our models in the Wildfire Prediction Dataset, as it appears in its Kaggle page [1]. In addition to all images of the dataset, the task also required not using the labels in the training partition of the original dataset, only a partition made from the smaller validation set.

To solve the problem of training different models in a few labelled data samples, the group uses two different approaches in Self-supervised Learning(SSL), focusing of contexts of *"label scarcity"*, the quantity of images during training doesn't excel a certain number of examples for each label. For comparison purposes, some Convolutional Neural Networks without any use of SSL were also training using the same data. The two approaches used can be found in [11, 8].

Dataset

The data consists of satellite images taken from Forest Fires website, each showing forest fires that occurred mainly in the territory of southern Quebec. The 350 × 350 images are separated in two classes, *wildfire* and *nowildfire*, each one amounting to 22710 and 20140 images, respectively.

The training, validation and test splits are done as follows:

- **Training.** 15750 *wildfire* and 14500 *nowildfire*;
- **Validation.** 3480 *wildfire* and 2820 *nowildfire*;
- **Test.** 3480 *wildfire* and 2820 *nowildfire.*

Approaches and Architectures

In this section, we detail the two different architectures and methods implemented to solve the classification problem without using the labels from the training set.

Convolutional neural network

This first test comes just as a comparison, the intention is to use it to assess how needed SSL is for the problem at hand. It checks the complexity needed to learn from the data and evaluate how models with different depths handle the task.

For training, this dataset was split into two subsets: 4,410 images (70%) for training and 1,890 (30%) for validation. The test dataset will remain untouched until the completion of training to ensure that its data is not inadvertently used for model optimization. The 70-30 split was chosen to maximize the training set size, enhancing the model's ability to learn relevant patterns.

The first classification architecture consists of two convolutional blocks followed by a dense linear block. The convolutional blocks extract image features, while the linear block weighs these features to determine the presence of wildfires. Each convolutional block applies a ReLU activation function and max pooling to reduce dimensionality. In addition to the training hyperparameters (number of epochs, batch size, learning rate, and weight decay), the architecture is influenced by factors such as the number of channels in each convolutional layer and the number of units in the fully connected layers. The linear block consists of three fully connected layers, with a sigmoid activation in the final layer to produce a probability distribution.

SSL

Here we detail the methods used for the Self-supervised Learning approach of the project. We also train the conventional CNN above in each one of the scarce label situations, for comparison, as well as another architecture comes from [7], the ResNet18. The network follows a residual learning, made to ease the training of networks that are substantially deeper than those used previously. Each one of its layers learn residual functions with reference to the layer inputs, instead the function of the input itself.

FixMatch Algorithm

For the first SSL approach, we implement the FixMatch Algorithm [11]. For each unlabeled sample, a *weakly-augmented* version of the image is forwarded into the model and receives a *pseudo-label.* This label is kept only if the prediction has a high score, i. e., if its score is superior to a threshold. Then, the model is trained to predict the *pseudo-label* for the *strongly-augmented* version of the same image. The full algorithm can be seen in 1, with H being the cross entropy loss for labeled data, α a weakly-augmented version of an image and $\mathcal{A}$ the strongly-augmented version of the same image.

```
Input. Labeled batch $\mathcal{X} = \{(x_b, p_b) : b \in (1, \ldots, B)\}$, and unlabeled batch
  $\mathcal{U} = \{u_b : b \in (1, \ldots, \mu B)\}$, confidence threshold $\tau$, unlabeled batch ratio $\mu$, unlabeled
  loss weight $\lambda_\mu$;
$\ell_s = \frac{1}{B} \sum_{b=1} H(p_b, \alpha(x_b))$ {Cross-entropy};
for $b = 1$; $b < \mu B$; $b + +1$ do
    $q_b = p_m(y|\alpha(\mu_b); \theta)$ {Prediction after weak augmentation};
end
$\ell_\mu = \frac{1}{\mu B} \sum_{b=1} \mathbb{1}\{\max(q_b) > \tau\} H(\arg\ max(q_b), p_m(y|\mathcal{A}(\mu_b)))$
```

Algorithm 1: FixMatch algorithm.

The main reason for using such technique is its simplicity, only requiring a different training loop in relation to a conventional CNN. Nevertheless, it's not without is downsides. The results are highly dependable in choosing good hyperparameters, like the threshold for keeping a pseudo-label and the ration between the batch size of labelled and unlabeled data. The method also relies a lot in the batch size of unlabeled data, what can limit the algorithm's performance in weaker setups. The original paper [11] for example, uses $\mu = 7$, that was reduced for $\mu = 4$ for this project.

Contrastive Learning

For our second approach to SSL, we use an End-to-End Learning scheme, as presented in [8]. The main idea behind Contrastive Learning is grouping similar samples, while keeping diverse samples far from each other, using a metric to measure how close two different embeddings are, usually *cosine similarity.*

During the training loop of the model, each sample $x_b, b \in (1, \ldots, B)$ is transformed into $\hat{x}_b$ through an augmentation. For each $x_b, b = k$, the augmented $\hat{x}_b$ is considered as the *positive* sample, and the augmented versions of the other samples , $\hat{x}_b (b \neq k)$, are considered as *negative* samples. The training consists of training the network to group $(x_k, \hat{x}_k)$ closer, and $(x_k, \hat{x}_b), b \neq k$ farther from each other. The whole process makes the model learn quality representations of the samples, that will be later used for transferring knowledge to other tasks, like classification of the reduced labeled partition of the images.

A brief schema of the process can be found in Figure 1.

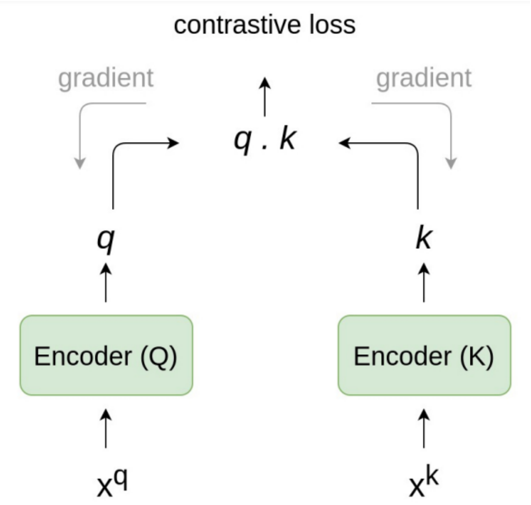

Figure 1: End-to-End learning with encoders, taken from [8].

The loss used in this project is the *infoNCE* loss, defined as below in Equation 2, where k_+ is a positive sample and τ the temperature parameter. The similarity metric, as mentioned above, is the Cosine similarity, as defined in 1.

$$sim(A, B) = \frac{A.B}{||A||||B||} \tag{1}$$

$$L_{infoNCE} = -\log \frac{\exp(sim(q, k_+)/\tau)}{\exp(sim(q, k_+)/\tau) + \sum_{i=0}^{K} \exp(sim(q, k_i)/\tau)} \tag{2}$$

The *backbone* networks employed for mapping the input samples to a latent space are also known as *encoders*, and multiple architectures could be used, like Transformers [5] and other CNNs.

For this project, different versions of ResNet [7] will be used a later compared as encoders: ResNet-18, ResNet-34, and ResNet-50. The augmentations used will be the same ones defined as *weak* in [11], comprising of horizontal flips and translations. The trained encoders are then paired with additional linear layers for classification and then fine-tuned in the smaller labeled dataset for inference.

Not lagging behind in simplicity, the main attraction to this approach is its versatility. FixMatch, if we are going to follow the original paper's footsteps, applies only (Wide)ResNets, but about any network could be used as an encoder for Contrastive Learning. This model also scales easier with the batch size used during training, since we only need to account for double the size chosen for the batch, and not $\mu \times batch\_size$, as n FixMatch.

Experimentation

For the setup of the experiments, all images were first formatted into 224 × 224, and then scaled and normalized, using ImageNet's [4] average and standard deviation. The group also created two types of augmentations:

- **Weak Augmentation.** The image is randomly flipped in horizontal direction and translated up to 12.5%;
- **Strong Augmentation.** The image goes through a random amount of transformations, as in [3]. The intensity of these augmentations, like in the paper, is also chosen uniformly between two extremes.

Specific parameters for the optimizer and other hyperparameters such as the temperature will be given in the specific section, since the architectures use vary in depth and complexity.

Bigger partitions of the validation set

Since the dataset is balanced between classes 0 and 1, accuracy was selected as the performance metric. Training was conducted with a batch size of 32 over 10 epochs, using a learning rate of 0.001 and a weight decay of 0.0002.

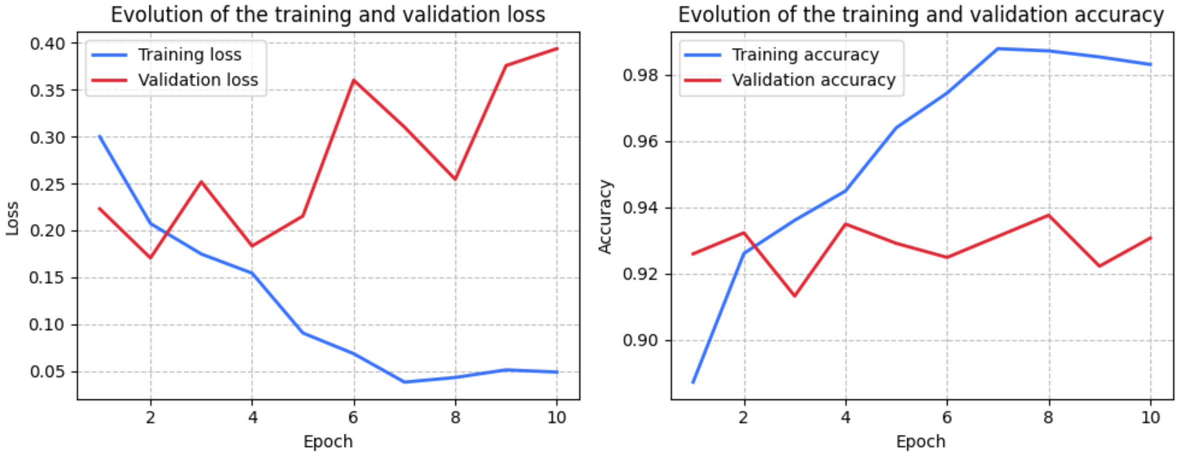

Figure 2: Loss and accuracy curves for training and validation data.

The training loss decreases as expected, while accuracy improves. In the validation set, the loss shows a slight increase, whereas accuracy remains stable between 91% and 94%. As we can see, the dataset used doesn't require network with a large depth to be solved, even when trained on small partitions of the labeled data the model learns the representations needed for a big accuracy.

As such, we move forward for the situation where the amount of images for each label is really low, and verify if the SSL methods proposed can achieve some superiority to standard supervised training.

SSL runs

For both the FixMatch and End-to-End encoders, as in [11], we use a learning rate of $1e^{-3}$, a momentum $\beta = 0.5$ and a weight decay of 0.03. The optimizer used is a Nesterov SGD [10]. The whole training is done for 100 epochs, for each different quantity of images per label $n\_labels \in 100, 250, 500$.

Method	**Accuracy**		
Images p/ label	100	250	500
Supervised			
Conventional CNN	17.58%	14.24%	14.24%
ResNet18	**26.14%**	14.24%	14.24%
SSL			
FixMatch (Wide ResNet50)	14.24%	14.24%	14.24%
Encoder ResNet34	14.25%	14.24%	**17.57%**
Encoder ResNet50	17.57%	**17.58%**	14.24%

Table 1: Results for the SSL approach of the project.

FixMatch

For the specific parameters of the FixMatch's learning loop, the ratio between unlabeled and labeled batches is $\mu = 4$, and the total batch size used is 32. The weight λ_μ for the unlabeled loss is 1. The results are grouped in .

As show in , FixMatch was not able to beat the End-to-End Encoder, or the standard supervised approach, in any of the configurations studied.

End-to-End Encoder

Three different architectures are used as the backbone, all being different versions of the ResNet [7]: ResNet34 and ResNet50. The training loop ran for 100 epochs, using all the parameters cited above.

For the fine-tuning part, different tests were done to decide the layers going to be kept frozen. The best results were found for a general fine-tune, where all layers' parameters could change, and a quick training was made with an additional classification linear layer at the end. As shown in , the encoder gets a slightly better result than the standard supervised approach.

Conclusion

The first model is relatively simple and efficient, requiring minimal computational resources while achieving strong results even with short training times. After only 10 epochs and less than 3 minutes of training, it reached an accuracy of 93%, representing a good trade-off between performance and computational cost. It also shows the simplicity of the database used, what indicates that the use of the unlabeled data might be unnecessary.

The table in also highlights this fact. Even when using few labels, standard supervised approaches with the ResNet18 and the other CNN get pretty close to the SSL results. ResNet18 even gets a large leap in accuracy for the low label regime, showing that even with a scarce labeled training set, the patterns from the images are easily learned by the networks.

Nevertheless, it must aso be said that both [11] and [8] use hundreds of epochs, and batch sizes way larger than the ones used in this report. Increasing both of these hyperparameters could, in theory, help leverage the unlabeled data, but such thing would require a longer period of time and computational power, while the standard approach with a simple CNN took no less than 3 minutes to achieve an accuracy higher than 90%.

References

[1] Ghaniaaba. A. Wildfire prediction dataset (satellite images). Kaggle, 2024. Available at: `https://www.kaggle.com/datasets/abdelghaniaaba/wildfire-prediction-dataset/data`.

[2] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*, pages 1597–1607. PmLR, 2020.

[3] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 702–703, 2020.

[4] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.

[5] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[6] George D. Greenwade. The Comprehensive Tex Archive Network (CTAN). *TUGBoat*, 14(3):342–351, 1993.

[7] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[8] Ashish Jaiswal, Ashwin Ramesh Babu, Mohammad Zaki Zadeh, Debapriya Banerjee, and Fillia Makedon. A survey on contrastive self-supervised learning. *Technologies*, 9(1):2, 2020.

[9] Patrick Knab, Sascha Marton, and Christian Bartelt. Dseg-lime–improving image explanation by hierarchical data-driven segmentation. *arXiv preprint arXiv:2403.07733*, 2024.

[10] C Liu and M Belkin. Accelerating sgd with momentum for over-parameterized learning. arxiv 2018. *arXiv preprint arXiv:1810.13395.*

[11] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin, and Chun-Liang Li. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. *Advances in neural information processing systems*, 33:596–608, 2020.