CS 437 (Deep Learning)

Project Report

29th April, 2022

Early Warning System for Forest Fires

Group 21

Ahmad Zubair 23-10-0074
Umair Yousaf 23-10-0053

Abstract

Forest fires are a major contributor to the damage inflicted upon our climate, and global warming, as well as requiring a sizable amount of human resources and manpower to control. Early detection of forest fires is a problem most countries need an inexpensive and accurate solution to. We explore the effectiveness of using one-shot learning to solve this problem, especially due to the lack of balanced datasets available. One-shot learning uses very small datasets to train a Siamese network that can achieve accuracies around 80% with few data instances. We concluded that although it is not comparable in performance to state of the art algorithms, it is a viable option given a very carefully curated dataset.

Introduction

Wildfires can pose a significant threat to human resources and can disturb the natural ecological balance, thereby affecting plant and animal life in the regions where the fire occurs. Examples include the 2019 - 2020 bushfires in Australia, which resulted in a total of 19 million hectares of area burnt (12.6 million comprising forests and bushlands). An estimated 3,100 homes were destroyed and 33 human lives were lost. Animal lives lost were estimated at 1.25 billion, with a total of 3 billion animals impacted[1].

It is therefore imperative for a system to be developed that can aid with early detection of wildfires, so that quick action can be taken once a wildfire is identified. This system also needs to fulfill certain criterion such as:

1. Being completely autonomous
2. Working around the clock: 24/7
3. Covering a large area
4. Being able to classify fires with high levels of accuracy
5. Being able to classify fires within the shortest time possible
6. Report little to no false negatives

The fields of Computer Vision and Machine Learning have already tried to tackle this issue, with some trained networks accurately classifying wildfires with over 95% accuracy, as discussed in this document. State of the art algorithms for an early warning system for fire detection mostly employ object detection and a few bounding box algorithms as opposed to traditional deep learning for classification. Most algorithms

[1] "Australian Bushfires - WWF-Australia." *WWF*, WWF, https://www.wwf.org.au/what-we-do/bushfires

make use of PTZ cameras (and a few use satellite imagery). However, not all algorithms are image based. Many are based on sensors and IoT devices.

Related Work

Considering the problem of early detection of forest fires, significant research has been done using a variety of methods like CNNs, Cycle-GANs, ResNet, DenseNet, VGG, YOLO, SVMs, Random Forests, SSD, etc. Overall the best performing models were CNNs, achieving accuracies above 95%.

A paper published in the MDPI journal addressed the lack of datasets: the authors commented on the fact that not enough relevant datasets were available for proper training of their network hence they employed the use of CycleGANs to generate wildfire images thereby expanding the training dataset. 5 fold cross validation was used on the models, and the train sets were separated into 2 train sets, with set A containing only original images, and set B containing original and generated images. One of the limitations of the paper was the model facing difficulty in differentiating between clouds and smoke in certain parts of the image.

Another paper published in ELSEVIER was able to achieve real time results at 28 frames per second using the YOLO v3 model. The dataset was obtained from public databases (bounding box annotation by research institutes) and included videos and images. The annotations were of two object classes (“fire” and “smoke”) and two disturbance classes (“fire-like” and “smoke-like”). They used a 50-50 test train split, and there were approximately 9695 “fire” and 7442 “smoke” objects. The data was roughly balanced in all classes.

In the paper, all models (including models with manually extracted features) were run on testdata1 which contained only videos. As expected, the CNNs performed better by a significant margin. The results proved that the missed detection rate with CNNs was below 0.1% (0 in the case of YOLO v3) as opposed to models with manual feature extraction (since manual features cannot distinguish between “fire” and “fire-like”). The best accuracy achieved was 99.62% (YOLO v3). Using testdata2, which contained only images and had a larger variety of scenarios, the top 4 models were rigorously tested to discern which had the best performance. Again, the CNNs outperformed.

A research paper published in IJCESEN, made use of two datasets, both scraped from Google, Getty, and Shutterstock with help of Selenium. Dataset 1 had a size of 2,360 forest fire images, and 2,400 images of forests without fire, hence a balanced dataset.

Dataset 2 had 5,000 of each category. Both datasets were preprocessed by grayscale and resize methods using libraries. A 70-30 test train split was used. The results proved that CNN outperforms all the other 6 models by an 18% margin in accuracy in both datasets, with 98.32 % accuracy on dataset 1 and 99.32 % accuracy on dataset 2.

Yet another paper published in MDPI determined that training a model from scratch requires tens of thousands of images. The authors used the HPWREN camera network to collect image data. In total, 8500 unique images were collected. The location and time of historical fires from the records of the California Department of Forestry and Fire Protection was used alongside the publicly available HPWREN archive to find images that contained smoke and forest fires. Furthermore, these images were shifted and flipped using augmentation techniques to increase the dataset size. The final result was a total of roughly 85,000 smoke image segments for training. For non-smoke images, images that contained features similar to smoke such as clouds were chosen to help the model distinguish between fire smoke and clouds.

The Inception V3 model yielded 99.67% accuracy on the validation set (whose size was 10% of the overall dataset), and an accuracy and F1 score of 91% and 0.89 respectively on the test dataset.

Methodology

Given the lack of readily available class balanced datasets, the approach chosen was to use One-Shot learning, using a combination of a Siamese network with Triplet Loss.

A siamese network is a very simplistic network that can have any configuration of layers applied to the input that result in a specific vector output: often the dimensions are reduced. This in essence generates a **fingerprint** of an instance. The fingerprint will have a representation somewhere on the hyperplane.

Triplet loss is a loss function useful for training for popular problems like face recognition. The concept of triplet loss in simple words is quite simple: push similar instances of fingerprints **closer** together on the hyperplane, and pull dissimilar instances **apart**. In order to do this, the conventional method is to use triplets of images, marked as **A, P** and **N** for anchor, positive (referring to an image of the same class as the anchor) and negative (different class). During the training step, the triplet loss is calculated for triplets of images, where the loss itself is defined by the formula:

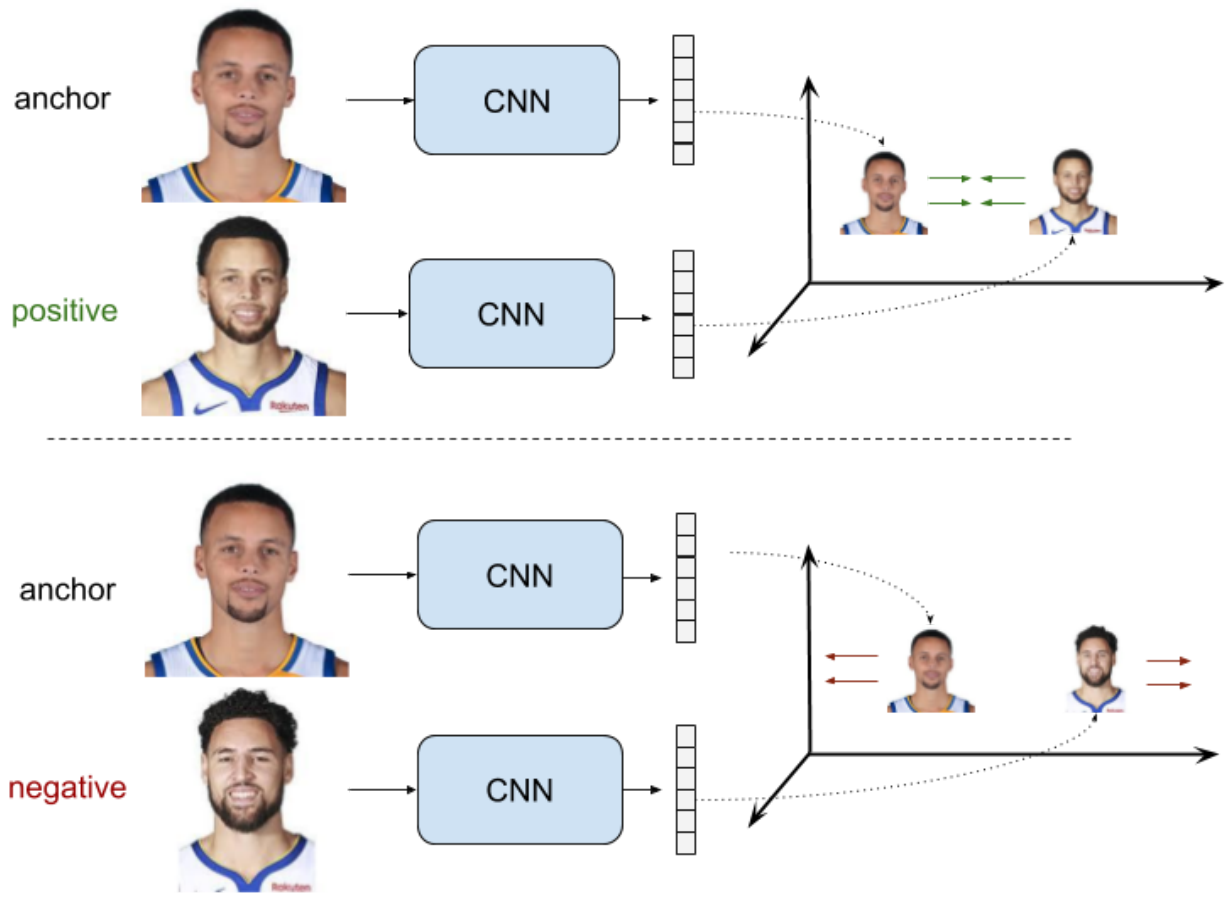

Figure 1: Siamese Networks (Taken from: here)

$$Loss = \sum_{i=1}^{N} \left[\| f_i^a - f_i^p \|_2^2 - \| f_i^a - f_i^n \|_2^2 + \alpha \right]_+$$

Figure 2: Triplet Loss

where a (**alpha**) is the margin. The loss calculates the Manhattan Distance between the anchor and positive instance, and the distance between the anchor and negative instance, then compares them to a margin that may be arbitrary (a hyperparameter). The loss is then broken down through its derivatives with respect to each weight, and then the weight adjustments are back propagated.

In order to make the model more versatile and robust, so that it does not mistake an evening sun for a forest fire, it was crucial to add two **additional** classes as opposed to the regular “fire” and “no fire” classes: “fire-like” and “smoke-like”.

For the testing step, only one of the 3 Siamese networks is required (see figure), to generate a fingerprint of a test instance. Once a fingerprint is generated, a set called the support set is also generated. The **support set** contains a median/average fingerprint of each class, and is representative of it. After the set is generated, the test instance is compared with each element in the set, and the euclidean norm is calculated. The prediction of the model is the class to which the test instance's fingerprint is closest.

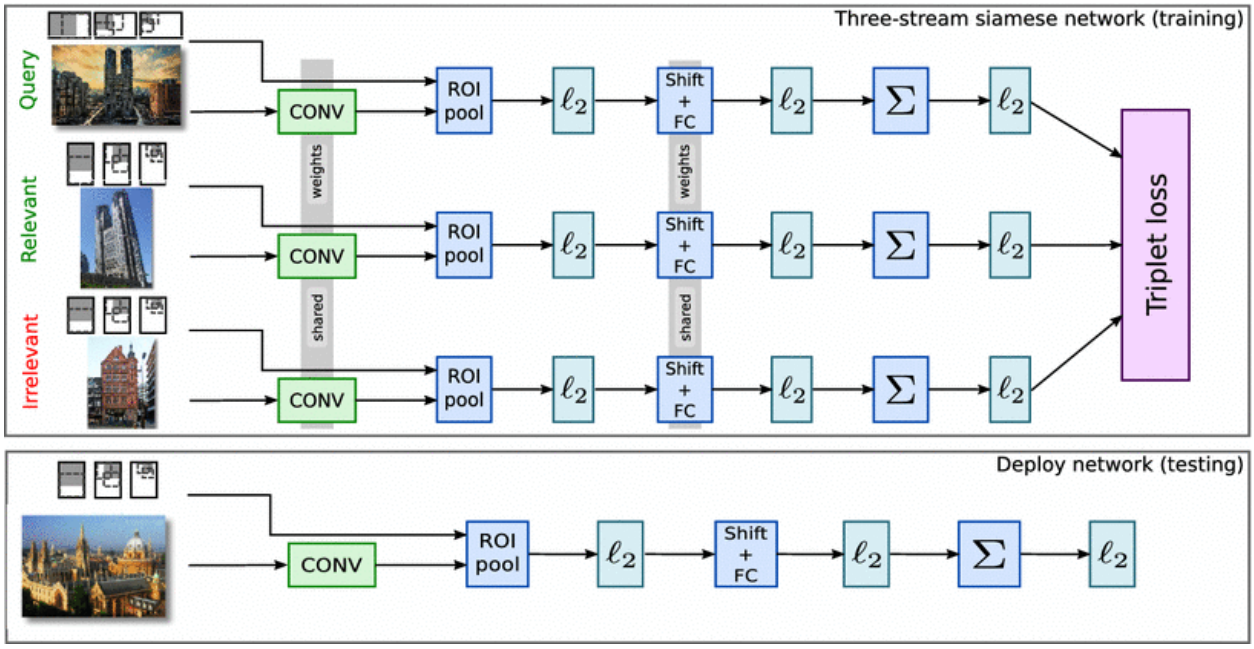

Figure 3: Testing Network (taken from: here)

Dataset

Although we identified a few datasets, the issue of an unbalanced dataset remained throughout. Hence we used the Mendeley dataset. This was split into training and testing data, with each consisting of 2 classes: "fire & nofire". There were 760 instances for each class in the training data, and 190 instances of each class in testing data. We manipulated this dataset to add 2 more classes to both testing and training data, namely "like-fire & like-smoke".

Then, the instances of each class were manipulated such that we had an even split between all 4 classes in training and testing. The final dataset contained 78 testing instances and 212 training instances, split into 4 total classes: "fire, nofire, like-fire & like-smoke". All images were of 250x250 size. The final dataset can be found here.

For a conventional NN, this may seem like a very small amount of training images. However, the whole purpose of OneShot/FewShot Learning is to use a small amount of images to achieve near similar accuracy to the conventional NN.

Some images from the training set are provided below:

Fig 1: Fire

Fig 2: Like-fire

Fig 3: Like-smoke

Fig 4: No Fire

Figure 4: Dataset

Results & Evaluation

At first, we trained the simple neural network on the 4 classes mentioned previously, but when testing, we opted to include only 2 classes “fire” & “nofire”, as in real world use, only these 2 classes will be required. This resulted in an accuracy and F1 score of 80% and 0.81. However, to rigorously test the abilities of one-shot and in order to push it to its limits, we decided to employ the 4 class model for further training and testing. So, during testing, the model must distinguish now between all 4 classes and not just “fire & nofire”. In practice, 2 classes should suffice for the early detection use case.

Perhaps the most important finding of this paper is that although one-shot learning is not comparable in performance to state of the art models, given a carefully curated dataset as per its deployment and use case, it can be a viable option.

In order to increase the accuracy and performance of the model would be to use extreme triplets, i.e. the representation of A and P in the hyperplane are farther apart as compared to the distance between A and N. In addition to this, if the dataset is collected with this principle in mind, the model would perform significantly better.

1. Comparison: Configurations of Siamese Networks

Description of Model	**Test Accuracy**	**Test F1 Score**
Simple Neural Network	59%	0.589
Basic convolution	60%	0.594
Convolution with Pooling & final Dense Layer	73%	0.722
Convolution with modified pooling & final Dense Layer & Upsampling	72%	0.687
Convolution with modified pooling & final Dense Layer & Upsampling **(With 2 classes used for testing)**	83%	0.827

Some hyperparameter data for all these runs:

- Epochs = 50
- Margin (for triplet loss) = 25,000
- Learning rate = 0.0001
- Optimizer = Adam

Variations of hyperparameters were tested, and the aforementioned values were optimal in terms of accuracy and used for all subsequent runs. The data provided shows results when significant changes in accuracy/model occurred, and omits updates that resulted in minimal change to accuracy & F1 score. Classes for test data were 4 in all runs here, except the ones labeled explicitly.

It can be seen that the addition of pooling and dense layers resulted in the most significant jump in accuracy and F1 scores, and using 2 classes (each with 190 images) for testing results in a further big jump in scores.

2. Comparison: One-shot & State of the Art Algorithms

Standard CNNs, DenseNets, or YOLOs perform significantly better than One-shot on any dataset provided, giving an output of nearly 95% accuracy and above. This can be seen in the papers mentioned in the Related Work section.

As compared to most algorithms, one-shot trains quite rapidly, given the small number of data points.

Conclusion & Improvements

Although there are many variations of models available for early detection, datasets are mostly limited. Despite this, many state of the art algorithms have achieved above 95% accuracy. One-shot learning is a promising technique to be used where datasets are lacking. Siamese networks and triplet loss in conjunction form the one-shot technique.

Perhaps apart from the focus upon the large variations of the Siamese network, the biggest improvement that can be done would be making variations of datasets with extreme triplets, and a comparison drawn between the maximum performance achievable. As well as this, more variations of networks can be done. Due to the memory limits of the hardware available, we were unable to test the possibility of an autoencoder style upsampling, and its performance, which appeared to be the most promising strategy.

Overall, one-shot encoding is a quick solution to problems with dataset constraints. Undoubtedly it may not perform as well, but can guarantee strong results when the weights and hyperparameters are tuned for the purpose at hand.