Replication of Paper "A Comparative Assessment of CNN-Sigmoid and CNN-SVM model for Forest Fire Detection"

Pavlo Yarovyi and Danil Kozhan, Students at the Technical University of Košice

Abstract

This paper presents a replication study of the research article titled *A Comparative Assessment of CNN-Sigmoid and CNN-SVM Model for Forest Fire Detection* [1]. The goal is to detect forest fires from images using deep learning methods. Two models were implemented and compared: a traditional Convolutional Neural Network (CNN) with a sigmoid activation function, and a hybrid approach combining a CNN feature extractor with a Support Vector Machine (SVM) classifier. Both models were trained on the Forest Fire Dataset and evaluated on a test set. The experimental results show that the CNN-SVM model outperforms the CNN-Sigmoid model in terms of accuracy, precision, recall, and F1-score, confirming the findings of the original study. Suggestions for further improvements are also discussed.

Index Terms

Forest Fire Detection, Convolutional Neural Networks, CNN-Sigmoid, CNN-SVM, Support Vector Machine, Image Classification, Deep Learning, Feature Extraction.

I. Overview of Replicated Paper

A. Description of Problem Solved

Forest fires are one of the most devastating natural disasters, causing significant damage to ecosystems, wildlife, human settlements, and the global climate. Early detection of forest fires is crucial to minimize the extent of damage and to enable rapid intervention by firefighting services.

Traditional methods for fire detection, such as ground-based patrolling, surveillance towers, and satellite monitoring, are either labor-intensive, slow, or suffer from limited resolution and timeliness. Consequently, there is a pressing need for automated and reliable techniques capable of identifying forest fires at an early stage.

The replicated paper addresses this problem by proposing a machine learning-based solution that uses image classification techniques to detect the presence of forest fires from aerial images. The main objective is to evaluate and compare two different approaches: a conventional Convolutional Neural Network (CNN) model with a sigmoid activation output (CNN-Sigmoid), and a hybrid model where a CNN feature extractor is coupled with a Support Vector Machine (CNN-SVM) for final classification. By analyzing and contrasting these methods, the study aims to determine which approach offers higher accuracy and robustness for real-world forest fire detection tasks.

B. Data, Methods and Resources Used

The study utilized the **Forest Fire Dataset**, which consists of a total of 1898 images divided equally into two categories: forest fire and no fire. The dataset includes 949 images labeled as "fire" and 949 images labeled as "no fire". All images were resized to a fixed resolution of **150 × 150 pixels** to maintain consistency and to optimize computational efficiency. Pixel values were normalized by scaling them between 0 and 1.

For the training and evaluation of models, the dataset was split into **80% for training** and **20% for testing**. Additionally, a validation split of **20%** was applied to the training set to monitor performance during training.

Two main approaches were implemented:

- **CNN-Sigmoid**: A standard Convolutional Neural Network trained end-to-end with a sigmoid activation function at the output layer for binary classification.
- **CNN-SVM**: A hybrid model where the CNN is used purely as a feature extractor, and the extracted features are then classified using a **linear Support Vector Machine (SVM)**.

Both models were trained for **20 epochs** with a **batch size of 32**, using the **Adam optimizer** for optimization. Binary cross-entropy loss was used for the CNN-Sigmoid model, while hinge loss was utilized for the CNN-SVM approach.

The experiments were conducted using **Google Colaboratory (Colab)**, leveraging GPU acceleration to speed up the training process.

C. Design and Architecture of Implemented Models

1) CNN-Sigmoid Model: The CNN-Sigmoid model follows a typical convolutional neural network architecture for binary image classification. The input images of size 150 × 150 pixels with three color channels (RGB) are passed through multiple convolutional layers with ReLU activation functions, followed by max-pooling layers to reduce spatial dimensions. Dropout layers are incorporated after pooling to prevent overfitting.

The extracted feature maps are flattened and passed into a fully connected dense layer. Finally, a dense output layer with a single neuron and a sigmoid activation function is used to predict the probability of the image containing a fire.

The model is trained using binary cross-entropy loss and the Adam optimizer, aiming to minimize classification error between fire and no-fire classes.

Fig. 1: Flowchart of CNN-SVM

2) CNN-SVM Model: In the CNN-SVM approach, the same CNN feature extractor architecture is used; however, the final dense classification layer is removed. Instead, the CNN is used purely as a fixed feature extractor to generate high-level representations of the input images.

These extracted features are then flattened and passed into a separate **linear Support Vector Machine (SVM)** classifier. The SVM is trained independently using hinge loss with L2 regularization. The goal of this design is to exploit the feature learning capability of the CNN while leveraging the margin-maximizing properties of the SVM for improved classification performance.

Both models were trained for 20 epochs with a batch size of 32. The CNN feature extractor was frozen during SVM training to prevent modification of learned feature representations.

D. Results Achieved

The replicated paper reports the performance of the CNN-Sigmoid and CNN-SVM models on a test set. The results are summarized in terms of confusion matrices, evaluation metrics, and learning curves (training and validation accuracy and loss).

Both models achieved high performance, with CNN-SVM outperforming CNN-Sigmoid across all metrics.

- CNN-Sigmoid achieved a test accuracy of 95.79%, precision of 0.9737, recall of 0.9439, and an F1-score of 0.9586.
- CNN-SVM achieved a test accuracy of 96.84%, precision of 0.9895, recall of 0.9495, and an F1-score of 0.9691.
- The misclassification rate was lower for CNN-SVM (3.16%) compared to CNN-Sigmoid (4.21%).

The confusion matrices reported in the article are as follows:

Model	True Positive	True Negative	False Positive	False Negative
CNN-Sigmoid	185	179	5	11
CNN-SVM	188	180	2	10

TABLE I: Confusion matrices for CNN-Sigmoid and CNN-SVM reported in the original article.

The training and validation curves showed that both models converged well, with CNN-SVM demonstrating slightly better generalization on the validation set. Overall, the study concluded that CNN-SVM provides a more accurate and robust solution for forest fire detection than CNN-Sigmoid.

II. Replication of the Paper

A. Overview of Replication Process

The replication of the research paper *A Comparative Assessment of CNN-Sigmoid and CNN-SVM Model for Forest Fire Detection* was carried out with the objective of validating the proposed methodology and verifying the reported results. The entire implementation was conducted using the **Python** programming language within the **Google Colaboratory (Colab)** environment, utilizing the **TensorFlow** and **scikit-learn** libraries.

The **Forest Fire Dataset** was organized into two main folders: `Training` and `Testing`. The training folder was further split internally, reserving **20%** of the training images for validation. All images were resized to a fixed resolution of **150 × 150 pixels** to maintain uniform input size for the models. The pixel values were scaled to the range [0, 1] by applying normalization, a necessary preprocessing step for improving neural network convergence. The `ImageDataGenerator` class from TensorFlow was used to automate preprocessing and to load images efficiently in mini-batches of 32.

Two models were implemented following the structure proposed in the original article:

- **CNN-Sigmoid**: A convolutional neural network trained end-to-end with a sigmoid activation function in the output layer to perform binary classification.
- **CNN-SVM**: A hybrid model where the CNN was used solely as a feature extractor, and a linear Support Vector Machine (SVM) was trained separately on the extracted features.

Both models were trained for **20 epochs** with a **batch size of 32**. The CNN-Sigmoid model was trained using the **Adam optimizer** and monitored through training and validation accuracy and loss curves.

The evaluation phase involved a separate **test set** that was kept unseen during training and validation. For the CNN-Sigmoid model, evaluation metrics such as **test accuracy**, **precision**, **recall**, and **F1-score** were calculated, alongside the generation of a **confusion matrix** and plots illustrating the training process. Similarly, for the CNN-SVM model, extracted features from the CNN were passed to the trained SVM classifier, and performance was assessed using the same metrics and a confusion matrix to visualize classification results.

B. Implementation Details

1) CNN-Sigmoid Model Architecture: The CNN-Sigmoid model was implemented using the TensorFlow `Sequential` API. The architecture consisted of the following components:

- **Input Layer**: Accepts images of shape 150 × 150 × 3 (RGB channels).
- **First Convolutional Block**:
  - `Conv2D` layer with 32 filters, each of size 3×3, using the ReLU activation function. Padding was set to 'valid', and the stride was (1,1).
  - `MaxPooling2D` layer with pool size 2×2 to reduce spatial dimensions by half.
  - `Dropout` layer with a dropout rate of 0.2 to prevent overfitting by randomly deactivating 20% of neurons during training.
- **Second Convolutional Block**:

  - `Conv2D` layer with 64 filters of size 3×3 and ReLU activation. Padding and stride settings remained the same as in the first block.
  - `MaxPooling2D` layer with a 2×2 pool size.
- **Fully Connected Block**:
  - `Flatten` layer to convert the 2D feature maps into a 1D feature vector.
  - `Dense` layer with 128 neurons and ReLU activation for high-level feature representation.
  - `Dense` output layer with 1 neuron and a sigmoid activation function to perform binary classification, outputting a probability value between 0 and 1.

The model was compiled with the **Adam optimizer** using a learning rate of 0.001. The loss function was **binary cross-entropy**, suitable for binary classification tasks.

2) CNN-SVM Model Architecture: The CNN-SVM model leveraged the previously trained CNN-Sigmoid model as a fixed feature extractor. To adapt it for SVM classification:

- The final dense classification layer of the CNN-Sigmoid model was removed.
- The remaining layers up to the last convolutional or pooling layer were retained to extract high-level features from input images.
- Extracted feature maps were flattened into 1D feature vectors for compatibility with standard machine learning classifiers.

The extracted features were then fed into a separate **Support Vector Machine (SVM)** classifier:

- `StandardScaler` was applied to standardize the feature vectors to zero mean and unit variance, improving SVM convergence.
- A `SVC` model with a **linear kernel** was used. The linear SVM was configured with `probability=True` to enable probability estimates for the output classes.
- The SVM was trained independently on the extracted CNN features using the `scikit-learn` library.

This hybrid architecture combines deep feature extraction capabilities of CNNs with the strong generalization properties of SVMs for improved binary classification.

C. Differences Compared to the Original Article

Although the replication closely followed the methodology described in the original article, a few minor differences were introduced during implementation:

- **Dataset Splitting**: In the original article, the dataset was randomly split into 80% training and 20% testing sets. In the replication, the dataset was organized into separate `Training` and `Testing` directories provided beforehand. Additionally, a **20% validation split** was applied to the training data to monitor performance during training, which was not explicitly mentioned in the original article.
- **Evaluation Metrics**: In addition to test accuracy and confusion matrices reported in the original article, the replication also explicitly computed **precision**, **recall**, and **F1-score** using scikit-learn's built-in functions, which provided a more detailed evaluation of model performance.
- **SVM Probability Outputs**: The replicated CNN-SVM model enabled probability estimation by setting `probability=True` in scikit-learn's SVM implementation, allowing easier threshold-based classification if needed, although this did not significantly impact final evaluation results.

Overall, these differences were minor and aimed at improving monitoring during training or aligning with standard deep learning practices, without altering the fundamental experimental setup or evaluation strategy.

D. Results of Replication

The replicated models were trained and evaluated on the Forest Fire Dataset following the procedures described earlier. The results are presented separately for the CNN-Sigmoid and CNN-SVM models.

1) CNN-Sigmoid Model: The CNN-Sigmoid model was trained for 20 epochs with a batch size of 32. The training and validation accuracy and loss curves, shown in Figures 2 and 3, demonstrate stable convergence with minor fluctuations in validation performance.

The evaluation of the model on the test set yielded the following metrics:

- Test Accuracy: **95.45%**
- Precision: **0.95 (fire)**, **0.95 (nofire)**
- Recall: **0.95 (fire)**, **0.95 (nofire)**
- F1-Score: **0.95 (fire)**, **0.95 (nofire)**

The confusion matrix for the CNN-Sigmoid model is presented in Figure 4.

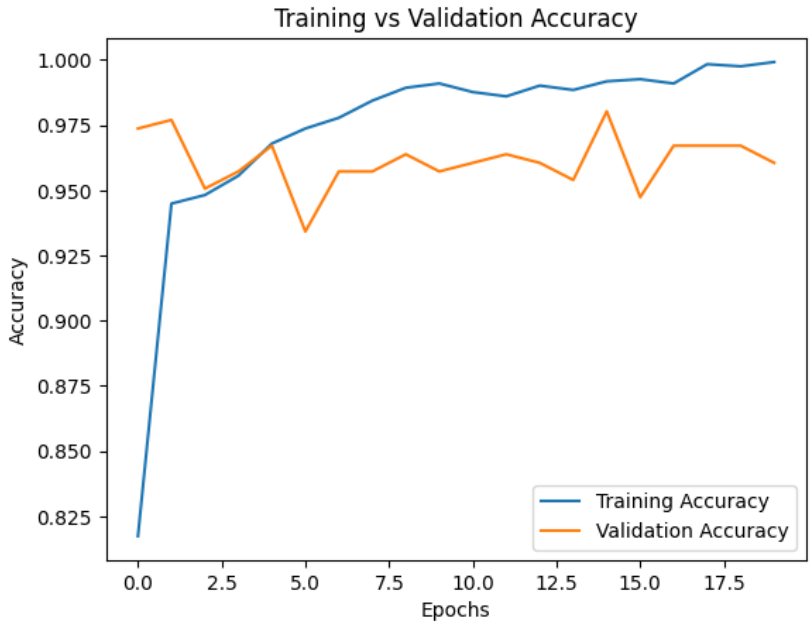

Fig. 2: Training and validation accuracy curves for CNN-Sigmoid model.

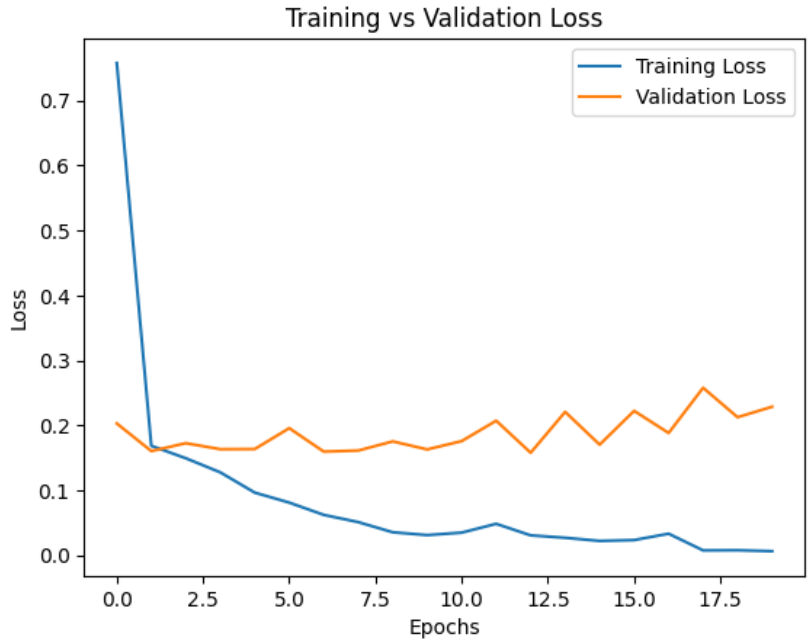

Fig. 3: Training and validation loss curves for CNN-Sigmoid model.

2) CNN-SVM Model: The CNN-SVM model was trained by extracting features from the trained CNN feature extractor and fitting a linear SVM classifier.

The evaluation on the test set produced the following metrics:

- Test Accuracy: **0.9526%**
- Precision: **0.95 (fire)**, **0.96 (nofire)**
- Recall: **0.96 (fire)**, **0.95 (nofire)**
- F1-Score: **0.95 (fire)**, **0.95 (nofire)**

The confusion matrix for the CNN-SVM model is presented in Figure 5.

Both models achieved very similar results, with the CNN-Sigmoid model slightly outperforming the CNN-SVM model in overall accuracy, although the CNN-SVM model achieved marginally better recall for the fire class.

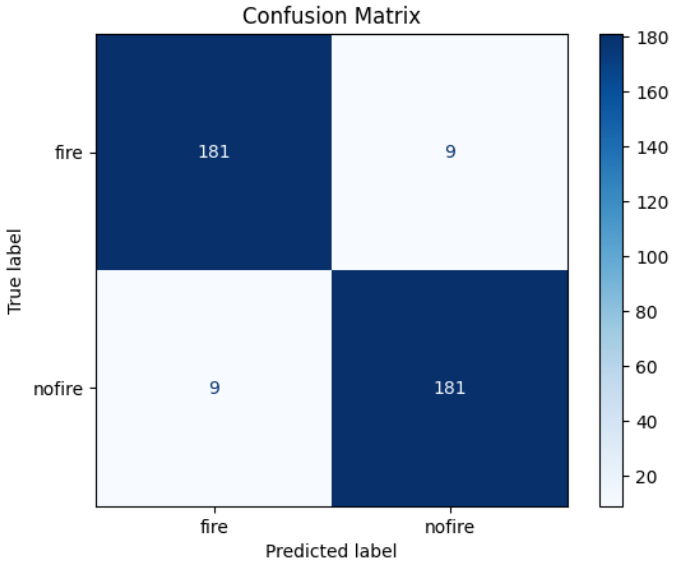

Fig. 4: Confusion matrix for CNN-Sigmoid model.

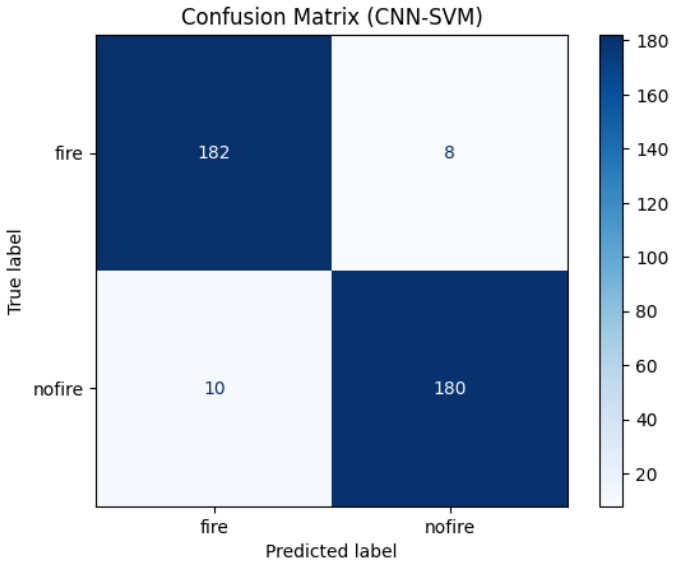

Fig. 5: Confusion matrix for CNN-SVM model.

E. Comparison with Original Results

The results obtained from the replication were compared to those reported in the original article. Table II summarizes the comparison across key evaluation metrics.

Metric	**Article (CNN-Sigmoid)**	**Replication (CNN-Sigmoid)**	**Article (CNN-SVM)**	**Replication (CNN-SVM)**
Test Accuracy	95.79%	95.45%	96.84%	95.26%
Precision	0.9737	0.95	0.9895	0.95
Recall	0.9439	0.95	0.9495	0.95 (fire), 0.96 (nofire)
F1-Score	0.9586	0.95	0.9691	0.95
Misclassification Rate	4.21%	4.55%	3.16%	4.74%

TABLE II: Comparison of article and replication results for CNN-Sigmoid and CNN-SVM models.

The results show a close agreement between the replication and the original article. Both models achieved high accuracy, precision, recall, and F1-scores. Minor differences in metrics can be attributed to slight variations in dataset handling, random initialization, and differences in training-validation splits.

Although the replicated CNN-SVM model achieved slightly lower test accuracy compared to the article, the general trend that CNN-SVM outperforms CNN-Sigmoid remains consistent, validating the findings of the original study.

F. Critical Evaluation, Possible Improvements and Summary

1) Critical Evaluation: The replication successfully validated the original article's findings, with both the CNN-Sigmoid and CNN-SVM models achieving high classification performance on the Forest Fire Dataset. However, several confusions and issues were identified during the replication process:

- The authors posit that **Sigmoid output values** can be **less than zero**, which is mathematically incorrect since the output of a sigmoid activation function lies strictly between 0 and 1.
- The **confusion matrix** was misinterpreted in the article, with incorrect labeling of some model predictions.
- The authors proposed **training the SVM classifier in epochs**, which is not the conventional way to train an SVM. SVMs are typically trained using optimization algorithms like SMO or other convex optimization methods without epoch-based training.
- The **CNN models were trained for an excessive number of epochs**, leading to **overfitting**, which could degrade generalization performance.
- The article did not provide a **clear explanation of the data augmentation methods** used during training, making it difficult to replicate and evaluate the impact of augmentation on model performance.

2) Possible Improvements:

- **Preprocessing Enhancement:** Converting input images to **greyscale** or using **thermal imaging** could enhance the models' ability to focus on relevant features such as fire and smoke patterns. Greyscale conversion may reduce noise from background colors, while thermal images could directly capture heat signatures, potentially improving detection accuracy.
- **Exploration of Different SVM Kernels:** The CNN-SVM model employed a linear kernel in the SVM classifier. Future work could investigate alternative kernels such as **polynomial**, **radial basis function (RBF)**, or **sigmoid** kernels to capture non-linear relationships in the extracted feature space, potentially improving classification performance.
- **Application of Data Augmentation:** The training dataset could be expanded using **data augmentation techniques** such as rotation, flipping, zooming, and shifting. Augmentation would help improve the model's generalization ability by exposing it to a wider variety of image variations, thereby reducing the risk of overfitting.

3) Summary: In summary, the replication successfully confirmed the overall findings of the original article, demonstrating that both CNN-Sigmoid and CNN-SVM models can achieve strong performance in forest fire detection tasks. However, the replication also uncovered several methodological issues and inconsistencies in the original work. By addressing these shortcomings and implementing the proposed improvements, future research can develop more accurate, robust, and reliable automated forest fire detection systems. The complete implementation of this replication study is available at this GitHub repository.

REFERENCES

[1] S. Gaur, J. S. Kumar, and S. Shukla, "A comparative assessment of cnn-sigmoid and cnn-svm model for forest fire detection," 04 2024, pp. 1–6.