# *Classification of Animal Species Using Video Dataset with Deep Learning*

Edy Sucipto - 23225307
Program Studi Magister Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jalan Ganesha 10 Bandung
E-mail (23522307@mahasiswa.itb.ac.id):

***Abstract*— In the world of electricity, it is crucial to maintain reliability. One cause of disturbances is animals entering the environment of high-voltage substations. Therefore, repelling these animals is necessary, and for early detection of their presence, one method is using deep learning. Thus, when these animals are seen by CCTV cameras, substation guards can be alerted and early prevention can be carried out. And this animal classification research has been widely conducted, both in the form of state-of-the-art models and by other individual researchers, with satisfactory results. Therefore, these existing models are used to obtain the best model using the available dataset according to needs. The models can be used directly or through transfer learning. To meet specific needs, a custom video dataset was created with four classes: cat, ferret, snake, and monkey, which are the types of animals most often causing disturbances at substations. And the results in this research has an accuracy rate of 91.23% was achieved using VGGNET as the feature extractor. But this cannot be compared with some other studies due to the use of different datasets. Regardless, this research ranks as the 4th best. With an accuracy rate above 90% and an average recall rate above 0.9, the model is deemed suitable for testing in a real-world environment as initially intended. Additionally, several conclusions were drawn regarding the importance of dataset quality, the types of models available, the use of software and hardware, and the implementation of features in the code for smooth debugging processes.**

***Keywords*— *Animal classification, deep learning, state of the art, video dataset, transfer learning (key words)***

## I. INTRODUCTION

In the high-voltage electrical installation installed at the Substation, most of the installation equipment is not protected by a safety layer, due to inspection and maintenance efficiency and flexibility. This results in a mandatory rule that no foreign objects are allowed in the Substation, as they can directly cause a short circuit, leading to power outages in the community.

Short circuits in the Substation can be caused by many factors, such as damaged equipment, dirty equipment, heavy rain, kite threads, wild animals, lightning, and others. Among these, one of the factors considered to be within control is the presence of wild animals. If a short circuit occurs due to wild animals, the office unit is deemed negligent in supervision and maintenance.

Many efforts have been made to address this issue, however, these efforts are still not optimal. Besides the variety of animals present, such as snakes, squirrels, cats, monitor lizards, rats, dogs, birds, or even monkeys, the measures sometimes hinder the team from performing inspections and tests, as testing requires direct contact with the equipment, or may even endanger the equipment itself due to modifications outside factory specifications. Therefore, a more straightforward and effective handling method is needed that does not have negative effects on the equipment.

Based on the above background, one way to solve this problem is to create a machine learning algorithm model that can recognize various types of animals in the Substation, monitored directly through CCTV. This can be the beginning of developing equipment/system to repel these animals when they enter the Substation area.

## II. RELATED WORK

Research aimed at classifying animal species has been conducted before. Here are some studies that will be compared and used as references in this research with the available dataset.

One existing model is the state-of-the-art VGGNET [1] with weights from ImageNet, trained on 1,281,167 training data and 1000 classes [2] achieving an accuracy of 92.7% [3]. However, VGGNET is designed to detect data in the form of images, while this research prepares data in the form of videos.

Another study was conducted by Sreedevi dan Edison [4] where the data also achieved a good accuracy of 99.6%. The dataset used was the IwildCam dataset on Kaggle [1] with 7 classes: tiger, raccoon, wild dog, deer, fox, mountain lion, cheetah. However, this study used image datasets, and the class types are different from the objectives of this research.

Another study by Xiao et al., [5], where the testing location aligns with the main objective, which is in the Substation environment, and the class types are almost similar: bird, cat, dog, kite, rat. The dataset was obtained from the surrounding environment as well as from PASCAL VOC and MS COCO [6] supplemented with crawler technology. This study also developed a tool for direct analysis using the YOLO V4 model. However, the results of this study showed AP and Map values all below 70%. The main goal was detecting and tracking.

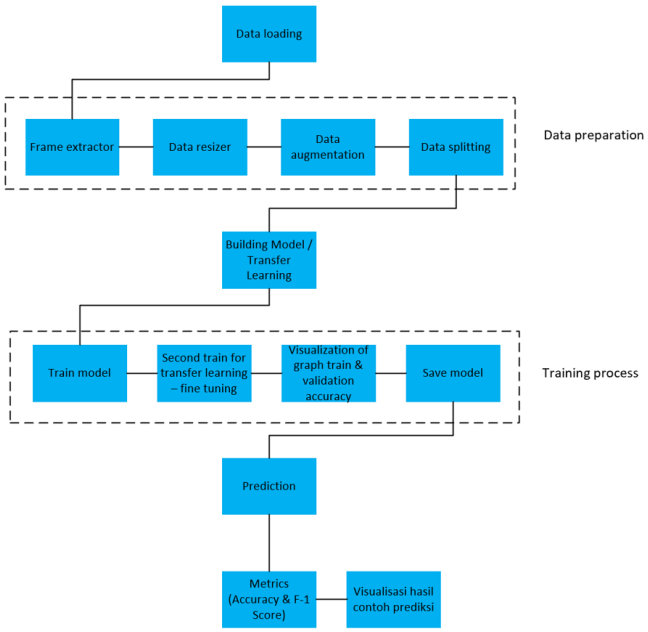
Another reference study is by Tran et al., [7] where this model is specifically designed to analyze video datasets. This model is used as a demo tutorial on the TensorFlow website and serves as a reference for writing 3DCNN code in TensorFlow [8]. The demo resulted in an accuracy of 72.17% from 50 epochs using the UFC101 dataset (UCF Center for Research in Computer Vision, 2011).

Lastly, using the state-of-the-art I3D (Inflated 3D ConvNet), which is one of the best models in 3DCNN. It is also a demo tutorial on the TensorFlow website [9]. This study was conducted by (Carreira and Zisserman, 2017) [10]. This study used the Kinetics dataset [11] which has 400 action video datasets, achieving accuracies of 98.0% and 80.9%.

## III. METHODOLOGY

In this research, several tests will be conducted based on existing models using a video set. Before testing, the dataset will undergo preprocessing to meet the model's requirements. After that, training and testing will be performed for each model with some parameter tuning. Finally, testing will be carried out to determine the accuracy of the model against the existing dataset, complemented by other metrics for evaluation. Figure 1 shows pipeline for testing process.

Fig. 1. Pipeline for testing process.




### A. *Dataset*

The video dataset used was obtained from video providers such as Itemfix [12] and YouTube [13] supplemented by videos directly sourced from CCTV storage. Only videos captured by CCTV are used in this dataset, as the primary goal of this research is to detect animal species using CCTV. Therefore, the video specifications tend to be small, blurry, but steady.

The animal types or classes in the dataset are limited to only four types: cat, ferret, monkey, and snake. This is aligned with the primary causes of disturbances caused by animals in the electrical transmission environment, and due to the limited datasets available.

Each class in the dataset consists of 73 videos, each with a duration of 5 seconds, 30 fps, 1280x720p, and in mp4 format. Data augmentation will also be performed in various ways. The data includes black padding resulting from resizing, either on the sides of the video or at the top and bottom, to ensure uniform dataset size.

Fig. 2. Dataset cat, ferret, monkey, snake

### B. *System Design*

Since the model to be used has a predetermined input size for the first layer, different resizing is still necessary after sampling frames from the dataset. Then, data augmentation is performed, including image flipping, image rotation, color jittering, and cropping, to increase the amount of data, which can improve accuracy and reduce overfitting [14].

Before splitting the data, a data augmentation process is applied for each scenario. The data augmentation performed includes:

- rescale=1./255,
- rotation_range=20,
- width_shift_range=0.2,
- height_shift_range=0.2,
- shear_range=0.2,
- zoom_range=0.2,
- horizontal_flip=True,
- fill_mode='nearest'

After data augmentation, the data is split with 80% used for training and 20% for testing. Regarding the frame extractor, the original dataset consisting of 292 videos in total (73 for each class) is transformed into 7341 images for training and 1836 images for testing. These image datasets are stored in memory except for scenarios where the full VGGNET library is used.

In this research, several structures are used to perform the classification task to find the best results, specifically for the available dataset. These include well-known (state-of-the-art) models used directly or through transfer learning from those structures, as well as models from previous research applied directly. The following is a list of deep learning model structures to be used:

a. VGG16 weighted by ImageNet
b. Inflated 3D ConvNet
c. Sreedevi K L et al. (2022)
d. D. Tran et al. (2017)

These models will be adapted and used for prediction with the existing dataset, either by using the pre-built model directly, building the model manually from scratch, or utilizing transfer learning.

The model creation and transfer learning process follow the existing model structures, with various parameter tuning to achieve the best results. The number of parameters and the execution time of the training process are also documented.

After training, the model is saved in *.h5 file format, with accuracy and F-1 score values recorded. Some models also include visualizations to facilitate evaluation, such as graphs of model training and validation accuracy, examples of augmented data, confusion matrices, examples of correctly predicted data, and examples of incorrectly predicted data.

## IV. Testing Scnearios

It is well known in the field of deep learning that no single model fits all types of problems. Therefore, in this research, several experiments will be conducted to achieve the best possible results. The aspects to be modified include the layer structure, the number of epochs, the learning rate values, and so on.

### *A. VGGNET*

Since VGGNET has pre-trained weights from ImageNet ready for class prediction, the first approach in this research is to directly predict using this model on the existing dataset. The dataset undergoes frame extraction first, with samples taken for every 5 frames, then resized to the VGGNET format of 244 x 244 pixels. The VGGNET library is downloaded and loaded using Python, and then used to predict the class of all sample frames, with results saved in a Microsoft Excel file. The file displays the filename, prediction, and accuracy value. The overall accuracy is calculated from this data.


Fig. 3. Visualisasi prediction using VGGNET library

The next scenario uses VGGNET as a feature extractor, replacing the original classifier model with a custom one, and retraining with the existing dataset. This approach aims to enhance the model's suitability for different datasets.

This scenario is further divided into several sub-scenarios, involving parameter tuning experiments such as classifier dense layers, epochs, batch size, learning rate, dropout, and frame sampling. This also serves as a learning tool for the researcher to validate the theory on parameter changes in deep learning.

TABLE I. Scenarios of VGGNET as feature extractor

| sub | frame sample | classifier dense | batch size | learning rate | epoch | dropout | GlobalAveragePooling2D / Flatten |
|---|---|---|---|---|---|---|---|
| a | 5 | 2 (4096 4096) | 32 | 0.0001 | 100 | no | Flatten |
| b | 5 | 2 (4096 4096) | 32 | 0.0001 | 10 | no | Flatten |
| c | 5 | 1 (4096) | 32 | 0.0001 | 10 | no | Flatten |
| d | 5 | 1 (512) | 32 | 0.0001 | 10 | no | Flatten |
| e | 5 | 1 (512) | 32 | 0.0001 | 10 | yes (0.5) | GlobalAveragePooling2D |
| f | 5 | 1 (512) | 64 | 0.0001 | 10 | yes (0.5) | GlobalAveragePooling2D |
| g | 5 | 1 (512) | 16 | 0.0001 | 10 | yes (0.5) | GlobalAveragePooling2D |
| h | 5 | 1 (512) | 16 | 0.00001 | 10 | yes (0.5) | GlobalAveragePooling2D |
| i | 5 | 1 (512) | 16 | 0.001 | 10 | yes (0.5) | GlobalAveragePooling2D |
| j | 10 | 1 (512) | 16 | 0.001 | 10 | yes (0.5) | GlobalAveragePooling2D |

Another VGGNET scenario involves fine-tuning in transfer learning. Some layers of the original VGGNET structure are removed and customized to meet specific needs, potentially increasing the accuracy of the built model. In this section, several experiments are conducted not only to find the best results but also to gain insights into the impact of parameter tuning changes.

### B. *INFLATED 3D CONVNET (I3DCNN)*

In this scenario, the IDE3 model structure is built from scratch, but the dataset remains the same. Retraining is performed with class types adjusted to the dataset. The model is expected to detect the classes and actions present in the dataset. Two scenarios are planned, differing only in the number of epochs.

TABLE II. SCNEARIOS I3DCNN

| sub | frame sample | classifier dense | batch size | learning rate | epoch | dropout | GlobalAveragePooling2D / Flatten |
|---|---|---|---|---|---|---|---|
| x | | | 8 | 0.0001 | 25 | | |
| y | | | 8 | 0.0001 | 50 | | |

### C. *SREEDEVI K L et al.*

As planned, this research also employs several models from previous studies that, while not state-of-the-art, have achieved good results, and are expected to perform well with the existing dataset. The structure mentioned in those studies is duplicated, and the data is adjusted to the model's requirements. Several different types of scenarios are implemented to achieve the best results.

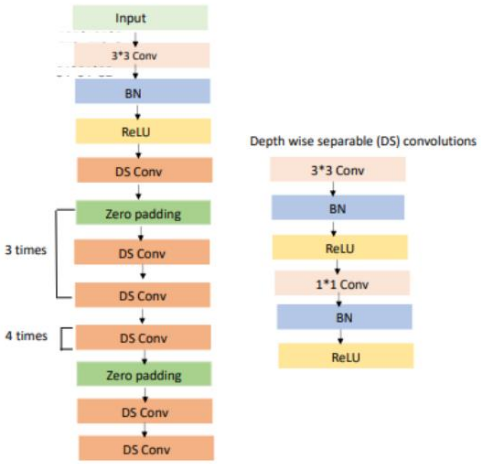



Fig. 4. Structur model Sreedevi K L et all

### D. *D. TRAN et al.*

Finally, the structure from Tran et al., intended for video classification is used. The original structure is rebuilt, followed by training with the existing dataset and class types. Various scenarios are also implemented to obtain the best results.

## V. RESULT AND DISCUSSION

### A. *VGGNET*

The first scenario using VGGNET involves direct classification with VGGNET that has been pre-trained on the ImageNet dataset. The existing video dataset is processed by extracting frames every 5 seconds and storing them on a hard disk in separate folders according to their class names. The library is downloaded, and then classification prediction is performed on all the data. The results are very unsatisfactory, with an accuracy of 0.1% and an F-1 score of 0.01. Upon further examination, additional classes similar to the existing classes were included, as there was suspicion that the class names in ImageNet differed from those in the current dataset.

TABLE III. MATCHING CLASSES BETWEEN IMAGENET AND THE EXISTING DATASET

| No. | Cat | Snake |
|---|---|---|
| 1 | Tabby cat (n02123045) | Garter snake, grass snake (n01735189) |
| 2 | Siamese cat (n02123597) | Green snake, grass snake (n01728920) |
| 3 | Persian cat (n02123394) | King snake, kingsnake (n01734418) |
| 4 | Tiger cat (n02123159) | Water snake (n01739381) |
| 5 | Egyptian cat (n02124075) | Vine snake (n01729322) |
| 6 | Cougar (n02125311) | Night snake, Hypsiglena torquata (n01729977) |
| 7 | Cheetah (n02128385) | Boa constrictor, Constrictor constrictor (n01751748) |
| 8 | Lion (n02129165) | Python, Python reticulatus (n01756291) |
| 9 | Leopard (n02129604) | Rattlesnake, rattler (n01770393) |
| 10 | | Coral snake (n01748264) |
| **No.** | **Monkey** | **Ferret** |
| 1 | Capuchin, ringtail, Cebus capucinus (n02481823) | Weasel (n02443484) |
| 2 | Howler monkey, howler (n02483708) | Mink (n02443114) |
| 3 | Titi, titi monkey (n02484975) | Polecat, Foulmart, Foulmart, Mustela putorius (n02441942) |
| 4 | Spider monkey, Ateles geoffroyi (n02487347) | Otter (n02444819) |
| 5 | Marmoset (n02490219) | Skunk (n02134656) |
| 6 | Baboon (n02486410) | Badger (n02138441) |
| 7 | Macaque (n02486261) | |
| 8 | Gibbon (n02488291) | |
| 9 | Colobus, colobus monkey (n02488702) | |
| 10 | Proboscis monkey, Nasalis larvatus (n02489166) | |

After examining all the classes, some correct class predictions were found, but the accuracy was still insufficient, only increasing to 0.2%. Upon further investigation, it was discovered that the difference lies between the dataset used by ImageNet and the dataset obtained from the frame extractor. While both have the same size of 224x224 pixels, the ImageNet dataset focuses on the classes, whereas the existing dataset mainly captures the overall environment. This is believed to be the reason why the class predictions from the existing dataset differ significantly.

Fig. 5. Example image from ImageNet

Fig. 6. Example image from video dataset existing

The next scenario involves using VGGNET as a feature extractor. In this case, the researcher conducted experiments for validation purposes. After testing all scenarios in this section, VGGNET with its original design structure, trained for 100 epochs, produced the best results, with an accuracy of 91.23% and an F-1 score of 0.91.

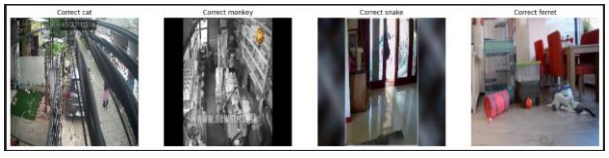



Fig. 7. Example correct prediction

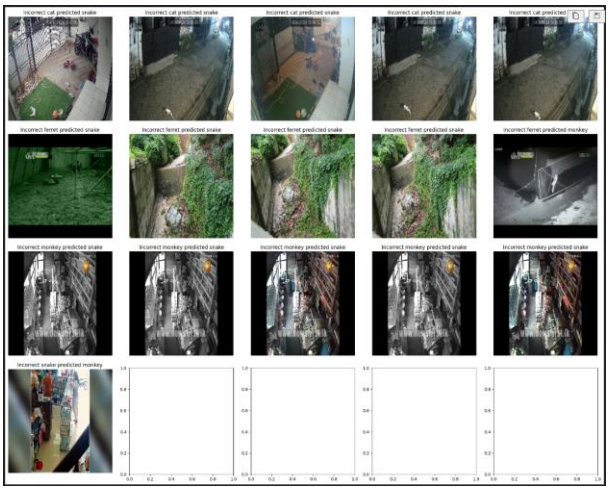


Fig. 8. Example wrong prediction

The final scenario involving VGGNET is fine-tuning the model. In this scenario, it was hoped that the results would be more satisfactory than the previous scenario. However, after training and retraining for 100 epochs and then testing, the accuracy obtained was 91.07%, slightly lower than the feature extractor scenario. This is believed to be due to the limited size of the existing dataset, particularly regarding its variety. As is known, the dataset consists of frame extracts, so most of the data are almost identical, especially the environment of the object, which tends to be static.

### B. *Sreedevi K L et all*

In this scenario, the author attempted to understand the structure described in the paper, as no code was provided. After converting the structure's flowchart into code, it became evident that the required image size for this structure is not the usual 224x224 but rather 125x125.

After training and testing for 10 and 100 epochs, the results were still unsatisfactory, with accuracy values of 26.96% and 22.87%. This is in stark contrast to the results mentioned in the study, where the average values were nearly perfect.

Upon further examination, it is believed that the causes of these results are:

- Different dataset types: Similar to the VGGNET scenario, the IwildCam dataset is highly focused on the object, whereas the available dataset is not.
- The author's understanding of the described structure: After reviewing the number of parameters, only 246,020 parameters were found, far fewer than others, which can reach hundreds of millions of parameters.
- Different parameter tuning settings: As the paper did not specify the settings used, the author only applied commonly used settings.
- Different function: In the study, the structure was used for object detection, not class prediction.
- Indications of overfitting.

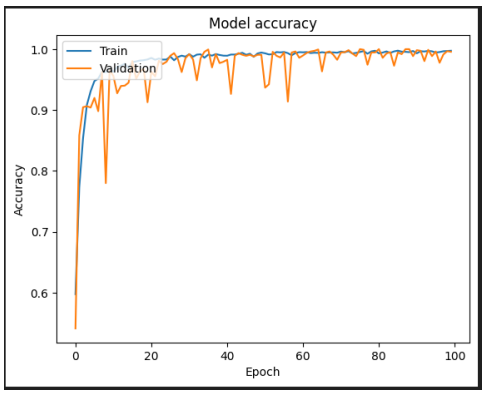



Fig. 9. Graph Training and Validation accuracy Sreedevi K L et all

Fig. 10. Example dataset of IWildCam

### C. *INFLATED 3D CONVNET*

The next scenario involves using one of the state-of-the-art methods in video action classification processing, namely I3DCNN. After conducting tests according to the planned scenarios, the results were very good, with an accuracy of 96%.

However, after examining the training and validation accuracy graphs, there is a suspicion of overfitting because the validation data results are very poor. It is believed that this occurs due to the insufficient dataset, which consists of only 73 videos for the entire training, validation, and testing data. In this scenario, data augmentation and a dropout rate of 0.5 have already been applied.

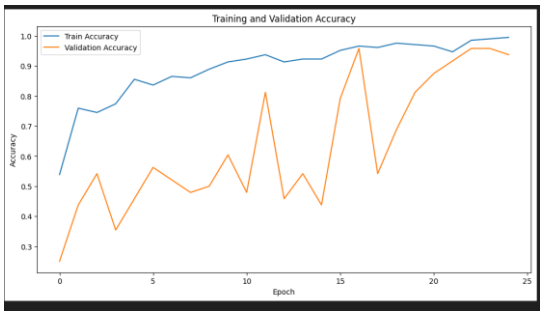



Fig. 11. Graph Training and Validation accuracy I3DCNN

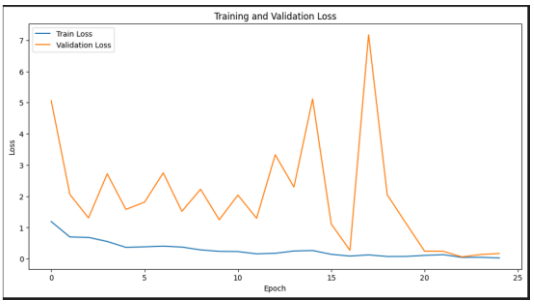



Fig. 12. Graph Training and Validation lost I3DCNN

### *D. D. Tran et all*

In the final scenario, which involves implementing research by Tran et al., the model specifically designed for video classification was used. The results were astonishing, with an accuracy of 100% after training for 100 epochs and conducting testing.

However, this raised suspicion, as a similar situation had occurred with previous models. After examining the training and validation accuracy graphs, overfitting was found to have occurred in this scenario as well.

The graphs showed that the validation data, both error loss and accuracy, did not align with the training data. In fact, the error loss tended to increase towards the end. This is a sign that overfitting occurred during the training process.

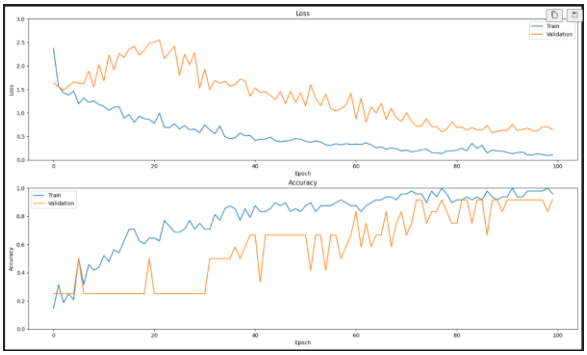



Fig. 13. Graph Training and Validation accuracy D. Tran et all

Upon further review prior to the training process, it was believed that the issue stemmed from the insufficient dataset to accurately detect the core of each class. The dataset comprised only 12 videos for training, 3 for validation, and 3 for testing. To address this, data augmentation was performed, including:

- Flip left-right
- Flip up-down
- Brightness adjustment

The accuracy then decreased to 66.67%. However, after re-examining the training and validation accuracy graphs, there was no significant change.

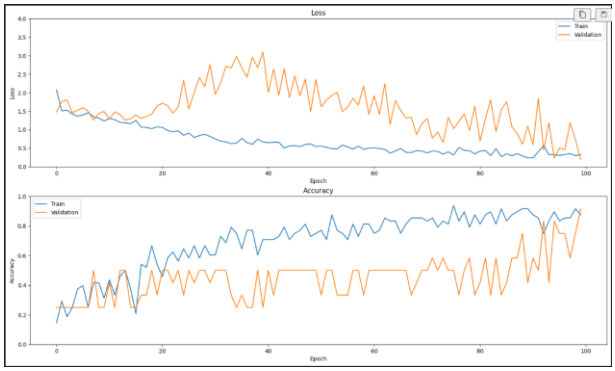



Fig. 14. Graph Training and Validation accuracy D. Tran et all after augmentation

### *E. Analysis and Evaluation of Testing Results*

From all the tests conducted, here is a summary of the results along with some additional notes:

TABLE IV. RESUME TESTING RESULT (ACCURACY & F-1 SCORE)

| Model | Name | sub | Acc (%) | F-1 Score | frame sample | classifier dense | batch size | learning rate | epoch |
|---|---|---|---|---|---|---|---|---|---|
| 1 | Pretrained VGG Full | | 0.2 | 0.02 | | | | | |
| 2 | Pretrained VGG for feature extractor | a | 91.23 | 0.91 | 5 | 2 (4096 4096) | 32 | 0.0001 | 100 |
| | | b | 87.69 | 0.88 | 5 | 2 (4096 4096) | 32 | 0.0001 | 10 |
| | | c | 86.82 | 0.87 | 5 | 1 (4096) | 32 | 0.0001 | 10 |
| | | d | 89.65 | 0.9 | 5 | 1 (512) | 32 | 0.0001 | 10 |
| | | e | 59.97 | 0.53 | 5 | 1 (512) | 32 | 0.0001 | 10 |
| | | f | 60.57 | 0.53 | 5 | 1 (512) | 64 | 0.0001 | 10 |
| | | g | 69.44 | 0.6 | 5 | 1 (512) | 16 | 0.0001 | 10 |
| | | h | 67.86 | 0.67 | 5 | 1 (512) | 16 | 0.00001 | 10 |
| | | i | 74.95 | 0.76 | 5 | 1 (512) | 16 | 0.001 | 10 |
| | | j | 61.87 | 0.61 | 10 | 1 (512) | 16 | 0.001 | 10 |
| 3 | Pretrained VGG fine tuning | r | 85.08 | 0.86 | 5 | 2 (4096 4096) | 32 | 0.0001 | 10 |
| | | | | | | -4 | | 0.00001 | 10 |
| | | s | 91.07 | 0.91 | 5 | 2 (4096 4096) | 32 | 0.0001 | 100 |
| | | | | | | -4 | | 0.00001 | 100 |
| 4 | Sreedevi K L et al. (2022) | x1 | 26.96 | 0.14 | | | 16 | 0.001 | 10 |
| | | x2 | 22.87 | 0.1 | | | 16 | 0.0001 | 100 |
| 5 | SotA I3D (Inflated 3D ConvNet) | x | 96 | 0.96 | | | 8 | 0.0001 | 25 |
| | | y | 90 | 0.89 | | | 8 | 0.0001 | 50 |
| 6 | D. Tran et al. (2017) | v1 | 50 | 0.38 | | | | 0.0001 | 10 |
| | | v2 | 100 | 1 | | | | 0.0001 | 100 |
| | | v3 | 66.67 | 0.6 | | | | 0.0001 | 100 |

TABLE V. RESUME TESTING RESULT (ADDITIONAL NOTES)

| Model | Name | sub | dropout | GlobalAveragePooling2D / Flatten | Params | | | train time (minutes) | Keterangan |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | Trainable | Non-Trainable | Total | | |
| 1 | Pretrained VGG Full | | | | | | | | |
| 2 | Pretrained VGG for feature extractor | a | no | Flatten | 119,562,244 | 14,714,688 | 134,276,932 | 1,044 | like VGG16 default model |
| | | b | no | Flatten | 119,562,244 | 14,714,688 | 134,276,932 | 105 | |
| | | c | no | Flatten | 102,780,932 | 14,714,688 | 117,495,620 | 104 | |
| | | d | no | Flatten | 12,847,620 | 14,714,688 | 27,562,308 | 96 | |
| | | e | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 95 | |
| | | f | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 94 | |
| | | g | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 93 | |
| | | h | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 94 | |
| | | i | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 99 | |
| | | j | yes (0.5) | GlobalAveragePooling2D | 264,708 | 14,714,688 | 14,979,396 | 93 | |
| 3 | Pretrained VGG fine tuning | r | yes (0.5) | Flatten | 119,562,244 | 14,714,688 | 134,276,932 | 214 | |
| | | | as design | as design | | | | | |
| | | s | yes (0.5) | Flatten | 119,562,245 | 14,714,689 | 134,276,934 | 2142 | |
| | | | as design | as design | | | | | |
| 4 | Sreedevi K L et al. (2022) | x1 | | | 244,676 | 1,344 | 246,020 | 65 | |
| | | x2 | | | 244,676 | 1,344 | 246,020 | 655 | |
| 5 | SotA I3D (Inflated 3D ConvNet) | x | | | 56,080,404 | 112,188,106 | 168,268,510 | 870 | overfitting |
| | | y | | | 56,080,404 | 112,188,106 | 168,268,510 | 425 | overfitting |
| 6 | D. Tran et al. (2017) | v1 | | | 443,290 | 886,614 | 1,329,904 | 23 | |
| | | v2 | | | 443,290 | 886,614 | 1,329,904 | 28 | overfitting |
| | | v3 | | | 443,290 | 886,614 | 1,329,904 | 28 | add augmentation, still overfitting |

From the above data, it is observed that VGGNET as a feature extractor achieved the highest accuracy, which is

91.23%. Only the scenarios related to VGGNET yielded promising results, and no signs of overfitting were detected.

The pattern suggests that a higher number of epochs generally results in better outcomes, but further testing with even more epochs has not been conducted due to resource limitations. Additionally, the learning rate did not have a significant impact, with the most optimal rate being 0.0001. The most influential factor is the quality of the existing dataset. The quantity, variety, and relevance to the topic are the main concerns in developing a good model. Overfitting and the inability of the model to detect accurately are direct consequences of not using a high-quality dataset.

### *F. Compare to Related Work*

Of course, in this research, the results cannot be directly compared with previous studies due to the differences in dataset sources used. However, several conclusions can be drawn from this research.

TABLE VI. COMPARISON OF RESULTS WITH PREVIOUS RESEARCH

| No. | Model | Accuracy (%) |
|---|---|---|
| 1 | VGGNET | 92.70 |
| 2 | Sreedevi K L et all | 99.60 |
| 3 | Chi Xiao et all | 74.43 |
| 4 | D Tran et all | 75.70 |
| 5 | I3DCNN | 98.00 |
| 6 | Our dataset using VGG as feature extractor | 91.23 |

This study achieved a mid-range accuracy, neither too high nor too low. However, for the initial purpose of creating a preliminary model for animal deterrence, this model is sufficient. In addition to having reasonably good accuracy, the recall values are also above 0.9 for all classes except for ferret (musang), which has a recall value of 0.8.

## VI. CONCLUSION

The conclusions drawn from this research highlight the importance of the dataset used and the suitability of the model to meet the determined objectives. The quantity, variety, and quality of the dataset are crucial for training an effective model. Parameter tuning is also important, but commonly used settings can serve as a good starting point because they are proven and can save time and resources.

No single model is perfect for every problem, as demonstrated by this research. Even models specifically designed for video dataset classification still cannot provide fully satisfactory results.

The implementation of comprehensive annotation features in the coding process is essential, especially for debugging. These include graphic data test and validation accuracy, information on the number of datasets before and after data augmentation, the form of data after resizing, success notes during data preparation, accuracy and F-1 score values, confusion matrices, examples of predicted datasets, and more.

Future research should consider adding more video datasets, as this can improve accuracy and reduce the risk of overfitting. Additionally, using a GPU for training is highly recommended to save time. Utilizing a GPU can speed up the process by 4-5 times, and even more with a GPU that has enhanced features (Buber and Diri, 2018).

Future research could also include additional classes, as the primary goal of this research can be implemented in various industries such as aviation, food, agriculture, and even security. Moreover, incorporating other state-of-the-art models like YOLO, ConvNeXt, etc., either through direct testing or transfer learning, can enhance the completeness of the research.

## ACKNOWLEDGMENT

The author is an employee at PT PLN (Persero) in West Jakarta. Currently, he is in the process of completing a postgraduate program at School of Electrical Engineering and Informatics, Bandung Institute of Technology.

## REFERRENCE

[1] iWildcam 2021- FGVC8, “No Title.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918

[2] Imagenet, “Imagenet.” [Daring]. Tersedia pada: https://www.image-net.org/download.php

[3] Medium, “Everything you need to know about VGG16.” [Daring]. Tersedia pada: https://medium.com/@mygreatlearning/everything-you-need-to-know-about-vgg16-7315defb5918

[4] K. L. Sreedevi dan A. Edison, “Wild Animal Detection using Deep learning,” *INDICON 2022 - 2022 IEEE 19th India Counc. Int. Conf.*, hal. 1–5, 2022, doi: 10.1109/INDICON56171.2022.10039799.

[5] C. Xiao *et al.*, “Method for Detecting and Tracking Foreign Objects in Substation Videos Based on Embedded AI,” *Proc. - 2021 Power Syst. Green Energy Conf. PSGEC 2021*, hal. 583–587, 2021, doi: 10.1109/PSGEC51302.2021.9542728.

[6] Dataset Ninja, “PASCAL VOC 2012 Dataset.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://datasetninja.com/pascal-voc-2012#download

[7] D. Tran, H. Wang, L. Torresani, J. Ray, Y. Lecun, dan M. Paluri, “A Closer Look at Spatiotemporal Convolutions for Action Recognition,” *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*,

hal. 6450–6459, 2018, doi: 10.1109/CVPR.2018.00675.

[8] Tensorflow, “Video classification with a 3D convolutional neural network.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://www.tensorflow.org/tutorials/video/video_classification

[9] Tensorflow, “Action Recognition with an Inflated 3D CNN.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://www.tensorflow.org/hub/tutorials/action_recognition_with_tf_hub

[10] J. Carreira dan A. Zisserman, “Quo Vadis, action recognition? A new model and the kinetics dataset,” *Proc. - 30th IEEE Conf. Comput. Vis. Pattern Recognition, CVPR 2017*, vol. 2017-Janua, hal. 4724–4733, 2017, doi: 10.1109/CVPR.2017.502.

[11] Google, “Kinetik Dataset.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://deepmind.google/

[12] Itemfix, “Itemfix.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://itemfix.com/

[13] Google, “Youtube.” Diakses: 29 Mei 2024. [Daring]. Tersedia pada: https://www.youtube.com/

[14] M. Elgendy, *Deep Learning for Vision Systems*. 2020.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 12 Juni 2024

Edy Sucipto 23225307