Segregation of Military Generated Waste using Transfer Learning

Syed Ayaz Imam
School of Computer Science & Engineering
Vellore Institute of Technology, Vellore

M. Monica Subashini
School of Electrical Engineering
Vellore Institute of Technology, Vellore

***Abstract*— Every year, large amounts of waste are generated due to military exercises on barren lands/desserts. These wastes comprise bullet shells, bandages, clothes, broken guns, etc which are extremely toxic due to the existence of sulphuric and nitric compounds from gunpowder on them. Rainwater may wash off these toxic compounds from the waste products and they may seep into the groundwater through the soil. Hence, waste management for such waste becomes an absolute necessity. This paper introduces a classification system for military generated waste using a deep-learning technique known as transfer learning and discusses the application of VGG-16, VGG-19, InceptionV3, and Xception architectures to the problem. The proposed models based on these pre-trained weights were training for 50 epochs and we found that the architecture based on the Xception network performed the best by achieving accuracy and loss of 0. 9723(or 97.23%) and 0.0857 on the validation data respectively.**

Keywords— Waste, Deep-Learning, VGG-16, Keras, CNN, Neural Networks, Classification, VGG-19, TensorFlow, Xception, InceptionV3, Accuracy

I. INTRODUCTION

Military waste disposal is a real-world issue and poses environmental and occupational hazards. The discarded ammunition and other left-behind plastic have adsorbed nitroaromatic and nitrotoluene compounds on the surfaces. DNT and TNT are present in gun powder and its prolonged exposure may lead to adverse health complications. Several measures such as burn pits are used for disposal since they seem very convenient and easy. Nevertheless, these measures emit nitric compounds in the form of gas to the air which contributes significantly to air pollution. Therefore, it is necessary to look for solutions to avoid long term health effects and minimize environmental impact.

One such solution is the utilization of protective gear and equipment for manual pickups and waste sorting from the drill sites. These types of equipment must be examined frequently and strict quality control must be established to ensure the medical safety of on-site workers. The manpower could be deployed at drill sites for manual picking and sorting purposes. However, this approach has various shortcomings. First of all, a large pool of workers must be employed for cleaning purposes and standards for safety and health insurance must be maintained. This demands a large investment to meet the required standards and administrative needs.

Another viable solution is to use autonomous robots for cleaning and sorting of such toxic wastes. Recent works in the field of artificial intelligence, computer vision have revealed that a well-trained model has the capability of surpassing human bounds for a classification task. A rover, with a robust path planning algorithm, can very well figure out an obstacle-free path to the specified goal and with remote control, such toxic wastes can be collected into a bin. The collected waste could later be sorted with hardware add-ons aided by a well-trained model.

This would significantly reduce human involvement cutting down the health risks, investment costs, and negative environmental hazards. This research aims to set-up a model for military waste segregation system to classify and sort waste products collected from the exercise sites and uses transfer learning to classify and segregate military waste. VGG-16, VGG-19, Xception, and InceptionV3 architectures of convolutional neural network with finely tuned dense layers have been used to train the model.

II. LITERATURE SURVEY

[1]. This paper introduces a new large convolutional neural network architecture that was used in the ImageNet LSVRC-2010 contest to classify images belonging to a thousand classes with a total of over a million image samples. The paper proved to be more effective than the previously advanced architecture and was able to achieve 37.5% and 17.0% top-1 and top-5 error rates. The dropout optimization technique was used to reduce overfitting with a neural network having 60 million parameters.

[2]. This paper aims to classify the various random images of waste and categorize them into three categories namely landfill. Recycling and paper. For this purpose, the researchers have used Faster R-CNN to classify the region proposals and categories. The CNN was finely tuned and various optimization techniques that could be used for improvement were discussed in the paper.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

[3]. This paper highlights the necessity of litter quantification by briefing over the existing cleanliness issues. The researchers put forward a computer vision application to deal with the issue of litter quantification and localize the types of images. For this purpose, the images were annotated before training the model was trained, the results of the model tested on a real-world scenario showed precise detection of litters present on the ground.

[4]. In the paper, the scholars highlighted the issue of an unhygienic civic environment and created an application to maintain a clean environment. The Garbage In Images dataset was used for training purposes and the model trained by the scholars attained a mean accuracy of 87.69%. The application allowed users to tag the garbage from the camera and the image in the bounds is detected by the model. Also, due to various optimizations, the memory usage as well as the prediction time was greatly reduced thus aiding in a smooth experience.

[5]. It is evident through experiments as neural networks grows deeper, they become much more difficult to train. To ease the training, this paper puts forward a framework for residual learning. The optimized network from this framework was tested on the ImageNet dataset and an error of 3.57% was attained on the test set. The remarkable result led it to win ILSVRC 2015 for classification. The paper also used the COCO dataset as well as the CIFAR-10 dataset and to train residual networks with varying layers. On the COCO dataset, 28% relative improvement was attained.

[6]. In this paper, the scholars have experimented on an increasingly deep neural network model with 3X3 convolution filters for a large scale image recognition task. In the ImageNet challenge for the year 2014, the team was placed at first and second positions for exceptional work in localization and classification tasks. The network models were very capable in attaining high-performance metrics and gain significant improvement over other architectures.

[7]. Pixel normalization is often done before training because larger pixel values increase the computational needs and thus it is a preferred habit to bring down the value in a range of 0 to 1. This paper puts forward a novel approach to normalize pixel values useful for the conversion of textual data to images. GPU is used for acceleration, which decreases the computing time.

[8]. Convolutional neural networks generally rely on a large amount of data to train without overfitting. If the amount of training sample present is not enough, a high variance could be observed in the model. However, in a real-world scenario, access to large amounts of data is limited. For example, data for image analysis in the medical/healthcare domain is very limited. This paper focuses on various image augmentation techniques/algorithms such as kernel filter adjustments, image mixing, feature space augmentation, etc. The scholars use these techniques to perform experimentation and gain meaningful insights out of the results.

[9]. In this paper, the scholars have implemented transfer learning as well as a deep convolutional neural network to train their data. The training times, as well as training and test accuracies, are extensive compared in the paper, and comparisons are drawn highlighting the benefits of using transfer learning instead of using architecture from scratch.

[10]. This paper derives from the existing state of the art YOLO model for object detection. The scholars have proposed a model for joint training of the YOLO model for object detection as well as classification. The researchers jointly trained YOLO900 on the COCO dataset as well as the ImageNet dataset simultaneously. This is done to classify the unlabelled data and detect them. The results of the training were comparatively fair with 19.7mAP even though the data only 44 out of 200 classes.

[11]. Data Handling and feature learning are considered to be among the most taxing tasks in data science. This data scalability issue has been looked upon in this paper, especially for ML classifiers used for social media activity analysis. The researchers have focussed on the techniques that could be used when working with such large unstructured social media data. For visual processing, various different architectures are present in the paper for detection and recognition tasks. Several well-known datasets such as MNIST, CIFAR-10, and IMDb datasets are used in the deep learning model used for the scope of this research.

[12]. The following paper, introduces a novel convolutional neural network architecture having great performance benchmarks in ImageNet classification tasks. This architecture, namely Inception was introduced in the ImageNet Large-Scale Visual Recognition Challenge in the year 2014. The computing resources in the network were optimally utilized by expanding the depth as well as the width of the network without excessively increasing the number of parameters. This kept the computation budget constant.

[13]. It is a well-known belief that as the depth of neural networks increase, the performance also improves even though the computational needs increase too. However, if among the mid-layers, the activated function is chosen/designed

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

cautiously, we can achieve better results from the output. To mitigate the gradient washing issue, the paper introduces an architecture where the hidden networks present in the beginning can directly be connected to the last layer having SoftMax function. Also, the scholars devised a generalized ReLU function to be used as an activation function and this design is then benchmarked on MNIST, STL-10, CIFAR-10, SVHN, UCT YouTube, Fashion-MNIST data.

[14]. In the past decade, there has been a boom in AI Industry. Neural Networks have proven to be of great use for pattern recognition, natural language processing and computer vision. Designing an architecture for neural networks is still considered to be a difficult task and it needs a high-level expertise to do so. This paper mainly focusses on a reward based(reinforcement learning) approach to select the construct and select the neural blocks. The evaluation of optimal network created is done on three datasets namely, MNIST, SVHN and CIFAR-10.

[15]. This paper sheds light on the working on age classification on images consisting or real-world images for faces. Due to varying facial structures of people belonging to different ethnic groups, the work done so far in this domain has suffered a lot due to absence of wide-ranging benchmarks. For the scope of this project, the researchers have used pre-trained models (CNN) and using transfer learning, achieved remarkable improvements in the set of comprehensive benchmarks. The design architecture was also optimized by carefully reducing the dimensions of output layer and precisely tuning the hyperparameters.

Various advancements have been done in deep learning techniques for classification, however, there have been no significant developments in the application of deep learning for military waste management. This calls for a system to efficiently recycle such toxic military generated waste which can only be done by bringing optimizations to the existing systems. For this purpose, we have presented a highly trained military waste classification system to classify and segregate such toxic wastes efficiently and accurately. We have not only suggested a transfer learning aided architecture with extremely optimized hyperparameters but also used four well known pre-trained models to pick the most well-rounded model.

III. METHODOLOGY

To classify images of military waste, VGG-16, VGG-19, Xception and InceptionV3 pre-trained models are used. The weights pre-trained on the image ImageNet dataset helps to compensate for low images per class and makes up for the computational needs(Transfer Learning). A carefully constructed architecture of dense networks is also appended to the pre-trained model to make sure that the model does not suffer from high variance or high bias. Also, Deep Neural Networks involve a lot of computations and hence a GPU is often required for acceleration because they enable highly parallelized computations. Therefore, this project has been implemented on Google Colab. The flowchart depicting the methodology is shown in Fig. 1.

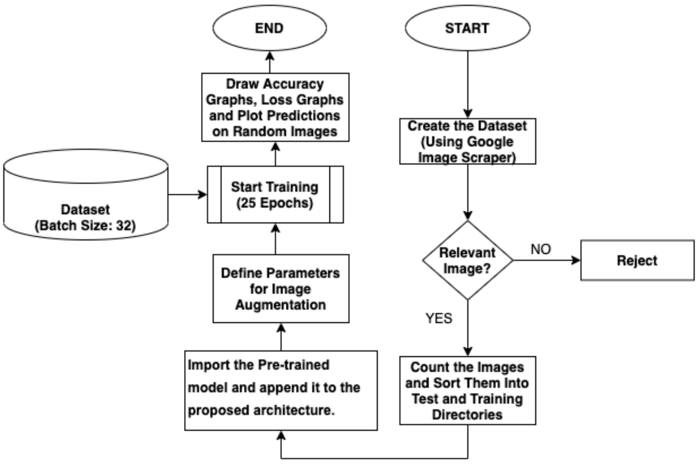

END
START
Draw Accuracy Graphs, Loss Graphs and Plot Predictions on Random Images
Create the Dataset (Using Google Image Scraper)
Dataset (Batch Size: 32)
Start Training (25 Epochs)
Relevant Image?
NO
Reject
YES
Define Parameters for Image Augmentation
Import the Pre-trained model and append it to the proposed architecture.
Count the Images and Sort Them Into Test and Training Directories

Fig. 1 Project Flowchart

3.1) Transfer Learning: Transfer Learning is an infamous deep-learning approach that allows the use of a model trained for a specific use case to be reused in an alternative task. Transfer Learning has been heavily used recently in the fields of computer vision and natural language processing since this technique significantly reduces the training time and also cuts down the resource usage. In various use cases, where the size of the data is large and challenging, engineers tend to prefer this optimization technique to build upon the weights acquired from another classification task.

For transfer learning, people generally use the models pre-trained on the ImageNet dataset. The ImageNet dataset contains millions of images from a thousand image classes. These images are labeled for supervised learning classification. Various researchers have tried to come up with different architectures to gain good training and test accuracies on this dataset by proposing new architectures. Some of the well-known architectures are AlexNet, InceptionV3, LeNet, Xception, VGG-16, VGG-19, etc.

We have used pre-trained weights from VGG-16, VGG-19, Xception, and InceptionV3 architectures for the implementation of this project. The architecture in Fig 2. depicts the pretrained layer with proposed configuration of dense layers. After experimenting with a lot of architectures and trying various combinations of hyperparameters, it was found that this architecture produced the most profound results without resulting in high variance or bias.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

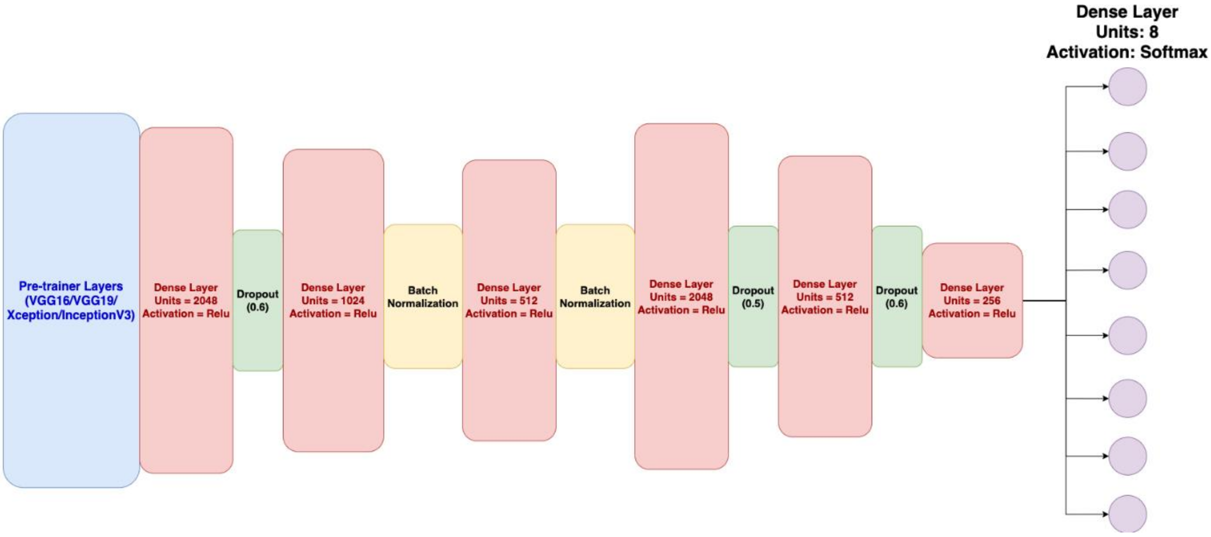

Dense Layer
Units: 8
Activation: Softmax
Pre-trainer Layers (VGG16/VGG19/ Xception/InceptionV3)
Dense Layer Units = 2048 Activation = Relu
Dropout (0.6)
Dense Layer Units = 1024 Activation = Relu
Batch Normalization
Dense Layer Units = 512 Activation = Relu
Batch Normalization
Dense Layer Units = 2048 Activation = Relu
Dropout (0.5)
Dense Layer Units = 512 Activation = Relu
Dropout (0.6)
Dense Layer Units = 256 Activation = Relu

Fig. 2 Proposed Architecture/Configuration

3.2) VGG16: VGG16 is an excellent CNN architecture that has stood the test of time very well. The architecture was used to create a model for image classification in the ImageNet Large Scale Visual Recognition Challenge and was declared the winner for the year 2014. The sheer amount of hyperparameters present in the architecture highlights the uniqueness of VGG-16. This architecture has in total of 16 layers and due to such a large neural network size, it has 138 million+ parameters. The last layers in the architecture are fully connected layers with SoftMax function as the output.

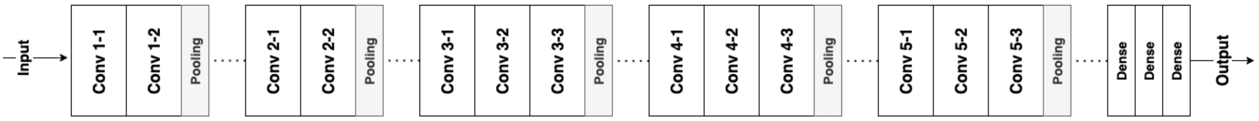


Fig. 3 VGG-16 Configuration

3.3) VGG19: The VGG19 architecture was created by the Visual Geometry Group, the same group who had devised VGG-16 architecture. The VGG-19 architecture is comprised of 19 layers in total consisting of three fully connected layers preceded by sixteen convolutional layers. Fig. 4 depicts the network configuration of VGG-19.

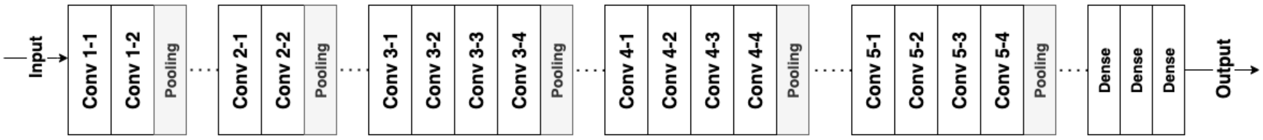


Fig. 4 VGG-19 Configuration

3.4) Xception & InceptionV3: Xception & Inception are 71 layers of deep CNN models. The models are available in Keras and they have been trained over millions of images belonging to a thousand classes. Xception, as well as InceptionV3, were introduced by Francois Chollet, a researcher at Google. Both the architectures have depth-wise separable convolutional layers however, the Xpection architecture performs marginally better compared to Inception due to more efficient use of model parameters.

3.5) CNN: Convolutional Neural Network was a remarkable discovery that made major contributions in computer vision. The CNNs are extremely precise and competent for certain usages. From building computer vision tools to applications in natural language processing, CNNs have been found to be very useful. The additional feature extraction layers present in the CNNs make them differ from traditional multi-layered perceptron networks.

Nowadays, CNNs are widely used for computer vision/image recognition tasks. The following are the essential layers present in the convolutional neural networks:

a) **Convolution layer:** All the images seen on a computer can be regarded as an array containing the colour intensity values of image pixels. A kernel matrix is used by CNNs to extract information from pixel array at the convolution layer.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

b) **Pooling:** To reduce the computational time, the pooling layer reduces the dimensions of images while retaining the necessary information.

c) **Flattening:** The matrix passed from the preceding layers is flattened or unrolled into a vector of features.

d) **Full-Connection:** Once the feature vector is received, it is passed to the dense network and the weight of each connection is successively re-adjusted in training after each epoch until the desired accuracy or performance measure is attained.

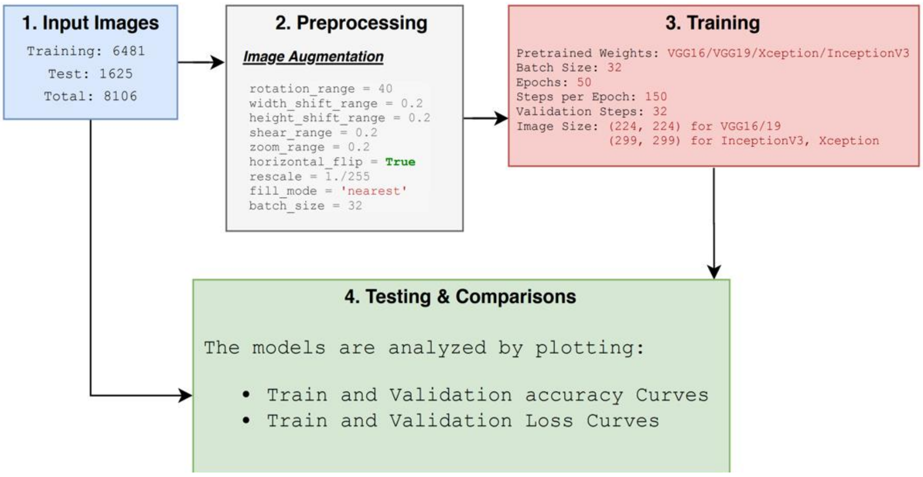
e)

IV. IMPLEMENTATION

This project is implemented using Keras library with a TensorFlow backend (version 2.1.4) on the Google Colaboratory platform and we have trained the models using transfer learning. Usually, the data is splitting into two or three sets. This is done to ensure that the model generalizes well by training on one particular set and tested on other set. For the scope of this project, the dataset is split into training and validation directories and the images are then zipped and uploaded to google drive, access it from Google Colab.

Fig. 3 shows the implementation of the project as a flowchart. The breakdown of project can be summarized in four steps as:

a) **Input Images:** One of the prime task to initiate the project is to gather 'trash' images usually found at military dump yards or dill sites. Due to the unavailability of a public repository for it, a dataset is produced by scraping images from google image. An online tool (imagecyborg) has been used for this purpose. Images for waste items in 8 different classes were downloaded using the web scraper which contributed to a total of 8106 images. The images belongs to eight different classes and they are pistols/revolvers, syringes, knives, bullet shells, bottles, band-aids, boots, automatic rifles. We have created training and test directories of images with a split ratio of 80:20 and uploaded the dataset google drive to access it through Google Colab.



1. Input Images

```
Training: 6481
Test: 1625
Total: 8106
```

2. Preprocessing

Image Augmentation

```
rotation_range = 40
width_shift_range = 0.2
height_shift_range = 0.2
shear_range = 0.2
zoom_range = 0.2
horizontal_flip = True
rescale = 1./255
fill_mode = 'nearest'
batch_size = 32
```

3. Training

```
Pretrained Weights: VGG16/VGG19/Xception/InceptionV3
Batch Size: 32
Epochs: 50
Steps per Epoch: 150
Validation Steps: 32
Image Size: (224, 224) for VGG16/19
            (299, 299) for InceptionV3, Xception
```

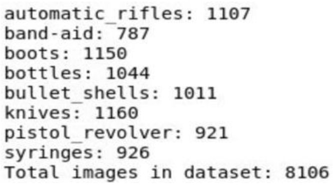
4. Testing & Comparisons

The models are analyzed by plotting:

- Train and Validation accuracy Curves
- Train and Validation Loss Curves

Fig. 5 Implementation Block Diagram

Some random images taken from the dataset are:



```
automatic_rifles: 1107
band-aid: 787
boots: 1150
bottles: 1044
bullet_shells: 1011
knives: 1160
pistol_revolver: 921
syringes: 926
Total images in dataset: 8106
```

Fig. 6 Class wise distribution of sample size

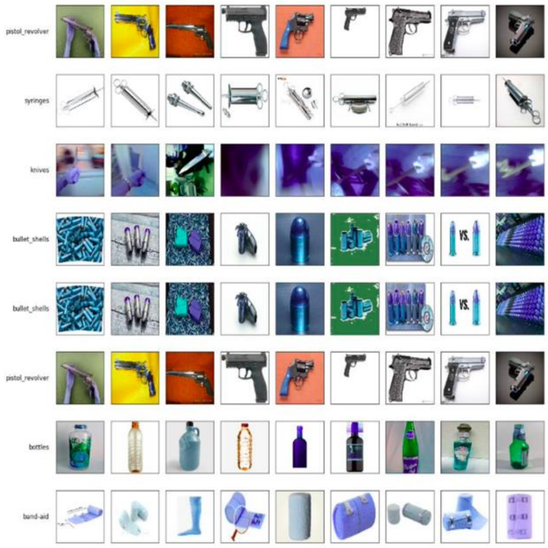


Fig. 7 Random Samples Taken from the Dataset

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

b) **Pre-Processing:** The Pre-processing of images is crucial, especially when dealing with less amount of data which may lead to high bias. We have used ImageDataGenerator class from Keras which is capable of real-time data augmentation. Various parameters such as rotation range, fill mode, batch size, rescale, width shift range, height shift range shear range, and zoom range were set to seemingly increase dataset size.

c) **Model Training:** After pre-processing, the models based on pre-trained architectures are trained for 50 epochs.

d) **Model Testing:** The last step of the project is an in-depth analysis of the trained model. The accuracy, validation accuracy, loss, validation loss is plotted against the number of epochs to looks for signs of overfitting/underfitting. The architecture that attains the highest accuracy, has the lowest loss and show no signs of underfitting or overfitting is declared the most optimal one.

V. RESULTS

After training the models, they are downloaded and saved to Google Drive(in h5 format). Also, the next step would be to test the models and evaluate their performances. Model testing is important to test out the performance on real-world data. It helps to analyse the fit and further steps could be taken if the model doesn't perform as expected.

The models were trained for 50 epochs and the VGG16 based model trained the quickest (in approximately 65 mins.) followed by VGG19, InceptionV3, and Xception based networks. Various other training data has been tabulated in the Table. 1 for reference:

Architecture	Number of Epochs	Time taken per Epoch	Total Training Time	TensorFlow Version
VGG-16	50	77s (on average)	65 minutes (approx.)	2.1.4
VGG-19	50	92s (on average)	75 minutes (approx.)	2.1.4
Xception	50	180s (on average)	150 minutes (approx..)	2.1.4
InceptionV3	50	120s (on average)	100 minutes (approx.)	2.1.4

Table 1. Summarised Information

For optimization, the models were finely tuned to attain the best performance. The learning rate of Adam optimizer was kept as 0.0006 in Keras without weight decay. For the transfer learning experiments, the weights of the pre-trained model were initialized to the ImageNet dataset. Simple data augmentation methodologies such as horizontal and vertical flip, zoom, shear, and random rotations were done. For this experiment, a batch size of 32 was chosen for both the architectures, and the dense layer weights were trained on Colab with a Tesla K-80 GPU which has CUDA acceleration.
They have trained the model for 50 epochs with 150 steps per epoch and 32 validation steps. The loss and accuracy of the model after the last epoch has been depicted in Table 2. After training the models, the validation accuracies of VGG-16, VGG-19, Xception, and InceptionV3 were found to be 0.9492, 0.9180, 0.9723, and 0.9717 respectively whereas the validation losses were 0.1510, 0.2794, 0.0857, and 0.1239 respectively.

Architecture	Training Accuracy	Training Loss	Validation Accuracy	Validation Loss
VGG-16	0.9275	0.2604	0.9492	0.1510
VGG-19	0.906	0.3225	0.9180	0.2794
Xception	0.9727	0.1036	0.9723	0.0857
InceptionV3	0.9632	0.1468	0.9717	0.1239

Table 2. Accuracy and Loss Values of the Architectures

After the last epoch, we observed that the model trained with Xception pre-trained layers performs the best, slightly outperforming InceptionV3. Xception trained model also outperforms VGG-16 and VGG-19 attaining much higher validation accuracy and lower loss value. Fig. 8 shows an abstract of Table 2 as bar-chart and it's clear that Xception model was far more superior that the rest. Once the weights are trained, it is recommended to analyse accuracy graphs and loss graphs to evaluate the performance. These learning curves help in analysing the performance over successive epochs or over time and give conclusive evidence to declare the model as good-fit, under-fit, or over-fit.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

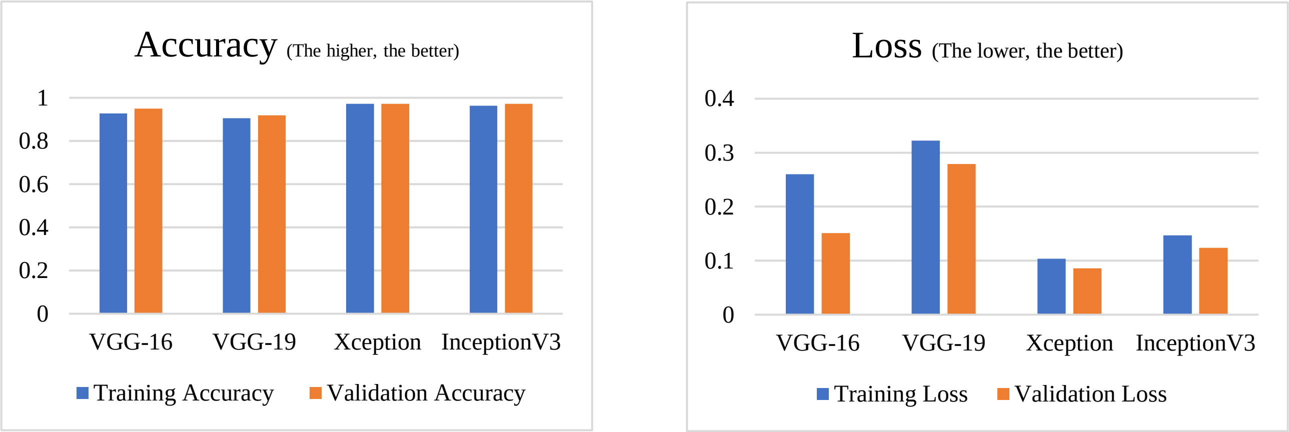


Fig 8. Accuracy Bar Plot and Loss Bar plots (Last Epoch)

In Table 3., the accuracy curves indicate that all the models have achieved high accuracies on training data and validation data. This signifies that the trained models are highly capable of determining correct classes if images of waste are fed to them. However, one must not rely on accuracy solely for evaluating the performance of the model. The validation and training losses must also be taken into consideration to look for signs of overfitting or underfitting. The formula to calculate accuracy is:

$$Accuracy = \frac{True\ Positive + True\ Negative}{True\ Positive + True\ Negative + False\ Positive + False\ Negative}$$

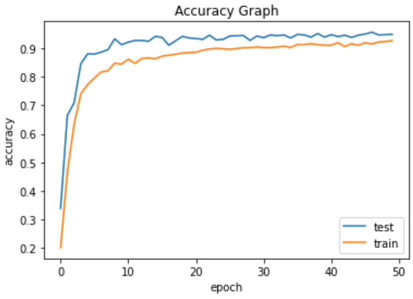
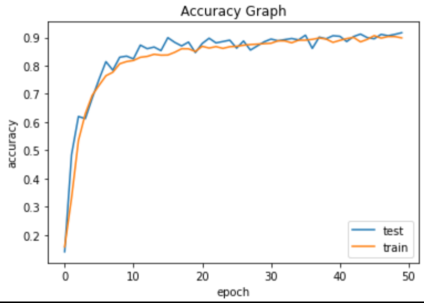
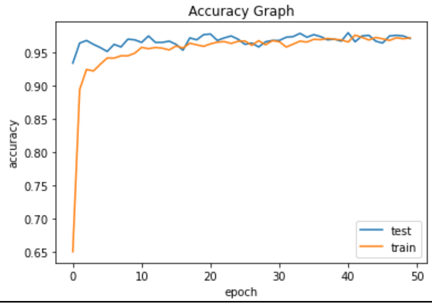
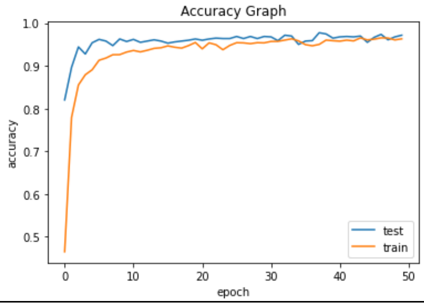
Accuracy Plots	
VGG-16	**VGG-19**
	
Xception	**Inception V3**
	

Table 3. Accuracy Graphs for VGG-16, VGG-19, Xception and InceptionV3 based Architectures

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

For multiclass classification, we have used SoftMax loss which is simply SoftMax activation function(in the last layer) with categorical cross-entropy. The categorical cross-entropy creates a one-hot encoded vector of the classes automatically and each vector can be considered as the probability of prediction. The depiction of encodings in Table 4 illustrates that target values are by default one hot encoded under the hood and therefore, the most proper loss function is categorical cross-entropy. Loss value may be defined as of measure of improvement for each successive iteration/epoch/optimization and it is calculated by the formula:

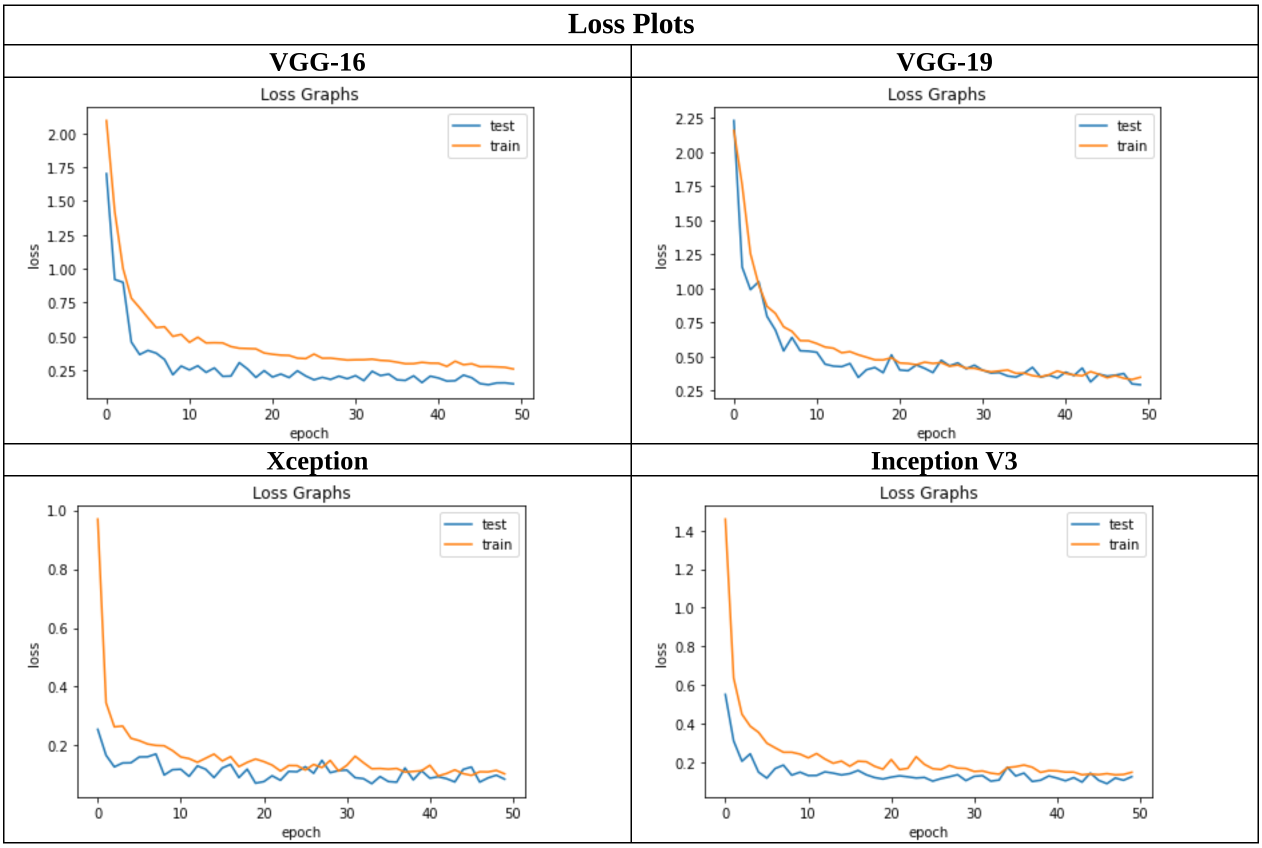
$$Loss = -\sum_{i=1}^{n} y_i \cdot \log \widehat{y_i}$$

Where,
n is the total number of output classes,
y_i *is the i-th scalar value in the output,*
$\widehat{y_i}$ *is the corresponding target value.*

Target Feature	Encoding
Automatic Rifles	[1, 0, 0, 0, 0, 0, 0, 0]
Band Aid	[0, 1, 0, 0, 0, 0, 0, 0]
Boots	[0, 0, 1, 0, 0, 0, 0, 0]
Bottles	[0, 0, 0, 1, 0, 0, 0, 0]
Bullet Shells	[0, 0, 0, 0, 1, 0, 0, 0]
Knives	[0, 0, 0, 0, 0, 1, 0, 0]
Pistols/Revolvers	[0, 0, 0, 0, 0, 0, 1, 0]
Syringes	[0, 0, 0, 0, 0, 0, 0, 1]

Table 4. Target Features and their Encoding

The graph below depicts a loss value plot representing curves for training and validation losses. The graphs show a smooth reduction of loss training loss for successive epochs and converge to low loss values. It can also be observed that the curve for training loss stabilizes and the gap between the test/validation curve and the training curve is less. This indicates that the model is a good fit and is capable to classify the images well.



Loss Plots

VGG-16

VGG-19

Xception

Inception V3

Table 5. Loss Graphs for VGG-16, VGG-19, Xception and InceptionV3 based Architectures

VI. CONCLUSION

From the analysis of experiment done in the preceding section, we have observed that Xception-based architecture is the most suitable to classify military waste products. This is due to the fact that Xception model has no visible signs of high variance and bias, has high accuracies and low losses for training and validation sets. Even though the models can be highly trained, it is a known fact that generally, waste products have a tendency to bend their shape due to external factors. These external factors may include compression or simply, tearing. However, even after external shape deformity, the waste products have their materialistic property intact. This all the models delivered well on these parameters.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)

For the future scope of the project, the size of the dataset can be tweaked to look for changes in performance and further work could be done for improvement if a new state of art architecture gets introduced in the future. Also, a real-time object classification model based on YOLO, R-CNN/faster R-CNN, or SSD could be developed if a real-time classification system is to be developed for the segregation of waste at the time of pickup.

VII. REFERENCES

[1] Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2017). Imagenet classification with deep convolutional neural networks. Communications of the ACM, 60(6), 84-90.

[2] Awe, O., Mengistu, R., & Sreedhar, V. (2017). Smart trash net: Waste localization and classification. arXiv preprint.

[3] Rad, M. S., von Kaenel, A., Droux, A., Tieche, F., Ouerhani, N., Ekenel, H. K., & Thiran, J. P. (2017, July). A computer vision system to localize and classify wastes on the streets. In International Conference on computer vision systems (pp. 195-204). Springer, Cham.

[4] Mittal, G., Yagnik, K. B., Garg, M., & Krishnan, N. C. (2016, September). Spotgarbage: smartphone app to detect garbage using deep learning. In Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing (pp. 940-945).

[5] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[6] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556.

[7] Sane, P., & Agrawal, R. (2017, March). Pixel normalization from numeric data as input to neural networks: For machine learning and image processing. In 2017 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET) (pp. 2221-2225). IEEE.

[8] Shorten, C., & Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. Journal of Big Data, 6(1), 60.

[9] Alshalali, T., & Josyula, D. (2018, December). Fine-Tuning of Pre-Trained Deep Learning Models with Extreme Learning Machine. In 2018 International Conference on Computational Science and Computational Intelligence (CSCI) (pp. 469-473). IEEE.

[10] Redmon, J., & Farhadi, A. (2017). YOLO9000: better, faster, stronger. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 7263-7271).

[11] Kumar, N., & Sukavanam, N. (2017). Deep Network Architecture for Large Scale Visua Detection and Recognition Issues. Journal of Information Assurance & Security, 12(6).

[12] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).

[13] Chen, Z., & Ho, P. H. (2019). Global-connected network with generalized ReLU activation. Pattern Recognition, 96, 106961.

[14] Zhong, G., Jiao, W., Gao, W., & Huang, K. (2020). Automatic design of deep networks with neural blocks. Cognitive Computation, 12(1), 1-12.

[15] Mallouh, A. A., Qawaqneh, Z., & Barkana, B. D. (2019). Utilizing CNNs and transfer learning of pre-trained models for age range classification from unconstrained face images. Image and Vision Computing, 88, 41-51.

(This work is licensed under a Creative Commons Attribution 4.0 International License.)