Sensors & Transducers, Vol. 257, Issue 3, May 2022, pp. 38-44

www.sensorsportal.com

Sensors & Transducers

Published by IFSA Publishing, S. L., 2022
http://www.sensorsportal.com

A Comparative Study of Deep Learning Models for Recognition of Poisonous Foods for Dogs

Arthur Wang, Chiral Mehta, Daniel Hilal, James Kabugo, Seshasai Srinivasan and * Zhen Gao

W Booth School of Engineering Practice and Technology, McMaster University
Main St W, L8S 4L8, Hamilton, ON, Canada
E-mail: gaozhen@mcmaster.ca

Received: 17 April 2022 /Accepted: 19 May 2022 /Published: 31 May 2022

Abstract: Convolutional neural network (CNN) models are widely applied in various areas including image classification, machine translation, autonomous driving, natural language processing, face recognition, recommendation systems, among others. This work investigated and compared different deep convolutional neural network models for image classification on a custom dataset. The models were trained on a dataset composed of seven image classes. The images were collected from various sources and a dataset for training the CNN models was created. The images included fruits, vegetables, and chocolates, which are considered poisonous to dogs for which Labrador Retrievers are used as a case study. Among the trained models, the Xception model showed the best performance, with a testing accuracy of 95%. Other notable models with high performance included InceptionV3, InceptionResNetV2, MobileNetV2 and VGG-16 with testing accuracy of 93.5%, 94.4%, 92.0% and 91.5% respectively. The trained models were able to easily recognize the food classes that are considered poisons for Labrador Retrievers on independent user images, with very high accuracy.

Keywords: CNN, Image processing, Machine learning, Classification.

1. Introduction

Despite constant efforts for raising awareness about pet poisons, pet poison hotlines still show an alarming number of yearly pet poisoning cases. In 2021, the Animal Poison Control Center (APCC) reported that the poisoning calls they received from the US increased by over 22 %. Poisoning cases were attributed to variety of factors including gardening products and toxicity of essential oils [1]. One of the most common and dangerous poisoning causes are foods that are safe for humans, but toxic for pets. This is because such foods would not seem intuitively poisonous. According to the APCC, food products where in the third position, accounting for more than 12% of the cases [1].

To limit the scope of this project, only foods that are poisonous to Labrador Retrievers and safe to human beings are tackled and detected via Artificial Intelligence (AI) algorithms. Specifically, we chose to detect chocolate, fresh mushroom, grapes, leeks, unripe tomatoes, and avocados.

Neural networks in general have been used to study a variety of applications [2-7]. With the evolution of this field, deep convolutional neural networks have gained significant attention in recent years for their outstanding image recognition and classification performance. Deep learning is a form of machine learning where a deep neural network, made up of many different layers with many different nodes, is used. With automatic feature extraction, deep convolutional neural network can classify new images based on the trained network. However, such training requires large labeled datasets. In this project, we collected images from four different web search engines: Google, Yahoo, DuckDuckGo and Bing. The images were then selected, compiled and labeled. Since some food ingredients are difficult to detect visually without any chemical test, only images which

have foods that can be visually detected were selected for training our neural network models.

The current work presents the performance of an architected model trained on the dataset collected, and compared to some reputable selected convolution neural network models. The models compared include LeNet, Resnet, VGG-16 and AlexNet. The goal of these models is to identify if the food in the input image has one of the seven poisonous foods and alert the user if it does. The dataset collected was split into three parts and were used for training, validation and testing, respectively. Each food class was represented by about 500 images in the training data and 100 images in the validation and testing data.

2. Methods

The study focused on applying a custom convolutional network model alongside state of-the-art deep convolutional neural network models to classify images of selected food and fruits, which were collected from numerous online sources. The models investigated included models inspired by AlexNet, LeNet, VGGnet and Residual network models. The models were realized using the Keras API in Python and run on Google Colab. A brief description of each individual method is discussed in the following sections.

2.1. Customized CNN

The customized convolutional neural network applied in this work, consisted of two convolutional layers, two maxpooling layers, a flattened layer, a dense layer and a softmax output layer. The first convolutional layer had an input size of 224 × 224 × 3, 32 filters, kernel size of 6 × 6 and a ReLU activation unit and this was followed by a maxpooling layer of size 2×2. The second convolutional layer consisted of 64 filters, a kernel size of 3 × 3 and a ReLU activation unit followed by a 2 maxpooling layer. Then a flattened layer was added, which was followed by a dense layer of 128 units and a softmax output layer.

2.2. LeNet

Originally presented in Ref. [8] for the purposes of handwritten digit recognition, LeNet-5 also serves as a valid but relatively poor image recognition neural network architecture. The architecture is structured as follows. First, the images are resized to 32 × 32 and re-scaled. Traditionally, the images are to be normalized such that a white pixel has a value of -0.1 and a black pixel that of 1.175. These images are convolved with six 5 × 5 filters with a stride of 1 to produce six 28 × 28 feature maps. These feature maps are sub-sampled with a 2 × 2 window and a stride of 2 that takes the average of the pixel values in a window. The resulting 14 feature maps are convolved with 16 5 × 5 filters with a stride of 1. The feature maps are again sub-sampled using average pooling filters of the same size, as previously done. The 5 × 5 feature maps of the previous layer are flattened and individually connected to a dense layer of 120 neurons and passed on an 84-neuron layer before being sent to the classifying output layer, which in this case has seven neurons. The tanh function served as the activation function for all layers except the last which uses the sigmoid function. An implementation of LeNet-5 was developed in Google Colab and trained to identify seven classes of food that are poisonous to Labradors.

2.3. ResNet

Residual neural network (ResNet) models were developed for realizing neural network models with considerable depth, that provide a reasonable model complexity, and that are easy to train. Moreover, deep and complex neural network architectures are notably difficult to train [9]. Deeper networks have been reported to yield better performance than their shallow counterparts due to their ability to extract high level features from the input data, enabling them to significant improvement in the model accuracy. However, it must be noted that far too many additional layers have also been noted to degrade model performance [9]. Nevertheless, the residual networks have been developed and proposed as powerful models for image classification [10].

There are various ResNet model formulations, and their general architecture is characterized by internal residual blocks, that are key to creating deeper neural network architectures with lower complexity. In this study, two state-of-the-art residual networks, ResNet50 and ResNet50V from Keras API were applied. The input image data was pre-processed as required in each case. Pre-trained models were employed, and the architecture of each model was maintained. For instance, the ReLU activation function originally applied in the model was retained. The only modifications were in the input layer, which was set to a target image size of 224 × 224 × 3 and the output layer modified to generate seven outputs and using a softmax activation function. All trainable parameters were re-trained. The Adam optimizer, with a learning rate of 0.0001 was applied. The categorical cross entropy loss was selected as the loss function during model training.

2.4. VGGNet

VGGNet is another convolutional neural network architecture which was proposed in 2014 [9]. The authors of VGGNet demonstrated the improvement of model accuracy with the use of deeper convolutional neural networks for image recognition [9]. To achieve this, a 3×3 convolution filter was utilized in the

model's architecture. With this technique, convolutional neural networks with a depth of 16 and 19 layers, were developed. These models are commonly referred to as VGG-16 and VGG-19. Like many other models, VGG-16 and VGG-19 are reported to have excelled on ImageNet data back in 2014 competitions. In the present work, pre-trained models from the Keras API were utilized. Again, all trainable parameters were re-trained. Moreover, the models were modified to accommodate input image size of 224 × 224 × 3 and seven outputs. Furthermore, the input images were pre-processed to match the pre-trained model image input requirement. During training, the Adam optimizer and categorical cross entropy loss function were applied.

2.5. AlexNet

AlexNet is a convolutional neural network that was first presented by Krizhevsky, et al. [11]. It was first utilized in the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) 2012, where it successfully proved that deep CNN can be used for image classification problems.

Implementing the AlexNet is relatively simple compared to other modern CNN architectures. It consists of eight layers; it has five convolutional layers, of which the first, second and the fifth are followed by max-pooling layers. The convolutional window shape starts with a size of 11 × 11, and reduces gradually to 5 × 5 and 3 × 3. As for the maxpooling layers, they have a pooling window of size 3 × 3 and a stride of 2 steps. AlexNet also has two fully connected hidden layers, and one fully connected output layer.

Additionally, the model applies the ReLU activation function. It also applies data augmentation and dropout to reduce overfitting. The dropout technique is applied in the two first fully connected layers with a dropping ratio of 50%. In summary, the model has a total of about 6.84 million trainable parameters.

For this work, AlexNet was implemented on the Keras platform using the Google Colab environment. The image target size of 224×224 was also used. Finally, like in the other models, to facilitate the comparison of the models, the same training, validation, and testing datasets were used with this model.

2.6. Inception

The Inception network architecture was developed with a purpose of increasing the computation efficiency within the network during model training and to achieve better accuracy [12, 13]. The Inception neural network architecture utilizes Inception blocks or factorized convolutions, and it is characterized by aggressive regularization. The model architecture allows for the creation of highly deep and wide convolutional neural networks while maintaining relatively less computational requirements than in many similar networks. The combination of Inception and residual networks was also noted to improve the overall performance and the accuracy of the deep convolutional neural network model by enabling faster training times than with Inception only models [14]. Therefore, in this study, pre-trained InceptionV3 and InceptionResNetV2 models from the Keras API were employed. The models were modified to allow a custom image input size of 224 × 224 × 3, adding a flattened layer and an output layer of seven outputs. All trainable parameters were retrained. The Adam optimizer, a learning rate of 0.0001, and a categorical cross entropy loss function were employed.

2.7. Xception

The Xception architecture was inspired by the Inception architecture. In the Xception model, the Inception blocks are replaced with depth-wise separable convolutions [15]. This deep neural network architecture model was found to perform better than Inception V3 on a large image classification dataset. The pre-trained Xception model in Keras API was modified as in the case of the pre-trained Inception models in 2.6 by adding flattened layer and output layer of seven output classes. Again, all trainable parameters were re-trained.

2.8. MobileNet

MobileNets architecture is a streamlined version of Xception architecture. This type of deep convolutional network architecture can be used to create a lightweight model, which can be utilized in mobile applications and in embedded systems. The model also aims to achieve a trade-off between latency and accuracy [16]. In this work, the performance of pre-trained MobileNetV2 model from the Keras API was compared with the other convolutional neural networks. The model was also modified as described in 2.7 by adding a flattened layer and an output layer. The image input size was the same as in previous models and the optimizer, learning rate, and loss function are the same as in 2.6.

3. Results and Discussions

3.1. Model Evaluation

The performance of each model was determined using the accuracy metric. Since, the classes were balanced in the dataset, accuracy was considered a good measure of model performance on this data. This was monitored during model training and validation. Fig. 1 shows the behaviour of the loss and accuracy for each epoch. On the other hand, Table 1 presents the

overall results obtained after fine tuning the models' hyperparameters such as the learning rates, number of epochs and batch sizes. The trained models were evaluated on a test dataset, and the outcome is summarized in Table 1. As shown in Fig. 1, in all cases, the training and validation losses were quite far apart, which is indicative of an over-fitting problem. This is evident form the fact that in most cases, the validation error continued to increase or significantly fluctuate as the number of epochs increased, whereas the error in the corresponding training data was continuously decreasing. Further, this behaviour persisted even after tuning the hyperparameters of several models. The overfitting problem observed in this present work can be attributed to the limited training examples for different food classes. Therefore, in future, it would be important to consider collecting more images or apply other available techniques such as data augmentation in order to improve the generalization of the model.

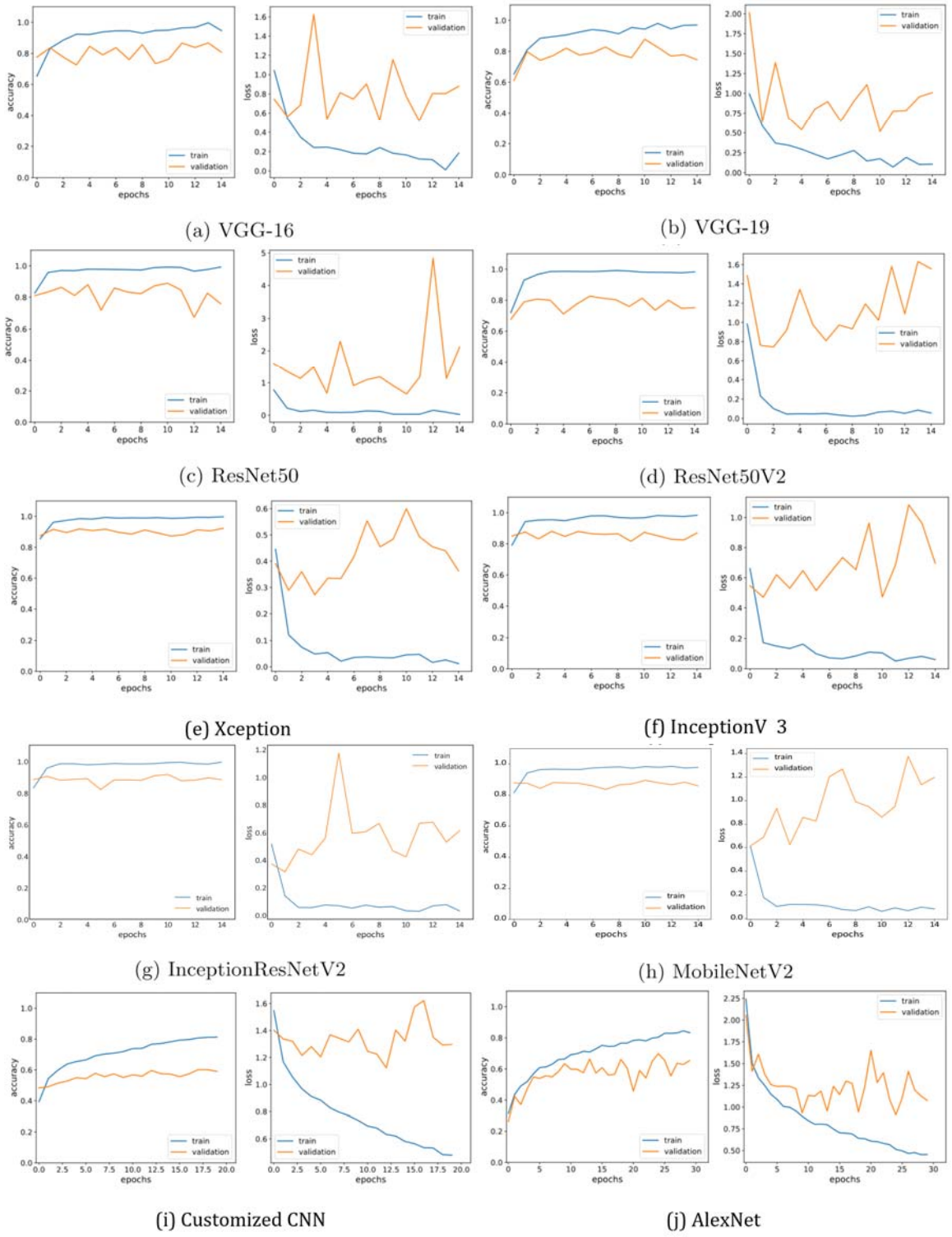


Fig. 1. Training and validation loss and accuracy results.

Table 1. Model accuracy.

Model	Accuracy		
	Training	Validation	Testing
Customized-CNN	0.813	0.590	0.716
LeNet	0.672	0.496	–
ResNet50	0.989	0.795	0.885
ResNet50V	0.979	0.777	0.874
VGG-16	0.960	0.849	0.915
VGG-19	0.959	0.813	0.864
AlexNet	0.813	0.637	0.757
InceptionV3	0.985	0.895	0.935
InceptionResNetV2	0.993	0.883	0.944
Xception	0.993	0.923	0.954
MobileNetV2	0.985	0.884	0.920

The custom CCN model showed relatively low classification accuracy. In this model, the observed accuracy after several model tuning was 0.716 on the test data (c.f. Table 1). Moreover, overfitting in this case was highly pronounced as shown in Fig. (1i). The performance of this model could be improved by further tuning the hyperparameters of the model, as well as training it on a much larger dataset.

An informal hyperparameter exercise was conducted to improve the performance of the LeNet model. For instance, changes were made to the optimizer, input color scale, activation function, and sub-sampling style, to improve the model's performance. A validation accuracy of 49.6% was achieved when color images were considered, ReLU replaced tanh as the main activation function, softmax was used instead of the sigmoid function, and the input images were free of imposed distortion such as shear or flip. The top three results from this hyperparameter tuning exercise are shown in Fig. 2, with plots displaying the change in loss function value and accuracy over time in epochs. Specifically, the training and validation loss and accuracy for the top three modified networks are shown in this figure.

The performance of the pre-trained models, from the Keras API, showed better performance than the other models like customized CNN, LeNet and AlexNet. Apart from their utilization of pre-trained parameters, these models presented a much deeper convolutional neural network architecture, which, for example, the customized CNN lacked. This means that the models with deeper networks were able to extract more relevant features during model training than those with relatively shallow architectures such as the customized CNN model. Besides the noticeable overfitting problem, the ImageNet pre-trained models, especially the Xception model, showed good results. The Xception showed a high accuracy on the validation and testing image data, with an accuracy of 0.923 and 0.954, respectively (c.f. Table 1).

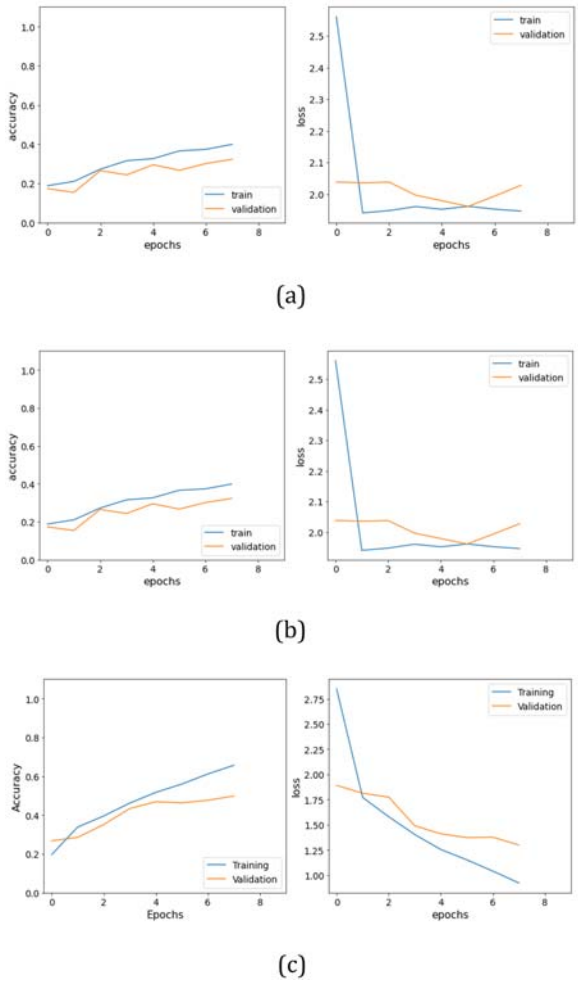
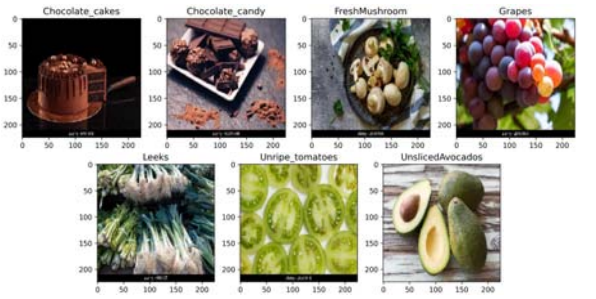


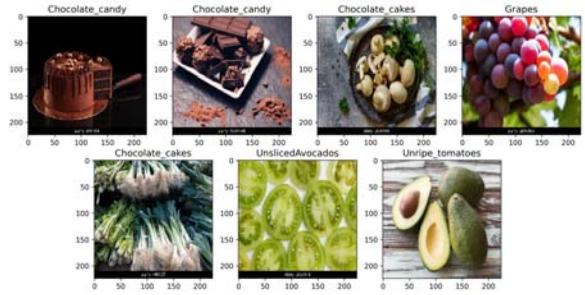
Fig. 2. LeNet hyperparameter tuning: (a) Adam optimizer, softmax output activation, relu over tanh activation, distortion allowed in training dataset; (b) RMSProp optimizer, softmax output activation, relu over tanh activation, no distortion allowed in training dataset of color images, and (c) Adam optimizer, softmax output activation, relu over tanh activation, no distortion allowed in training dataset of color images.

3.2. Performance of Trained Models

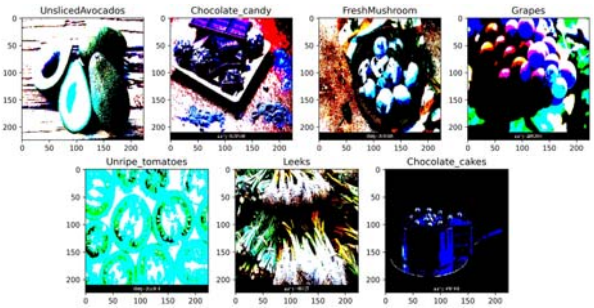
In addition to the validation and testing datasets, the performance of models on an independent set of images was assessed. This was done to further evaluate the practical viability of the trained models in predicting whether the food represented in a particular image was a potential poison for the Labrador Retriever. Moreover, this was done to further compare the robustness of the trained deep convolutional neural network models. Fig. 3 demonstrates the observed prediction performance of select trained models on a smaller independent dataset. The results showed that among the 7 images, the customized CNN could correctly classify only the image of grapes. On the other hand, VGG-16, InceptionV3 as well as Xception were able to classify the images as shown in Fig. 3.



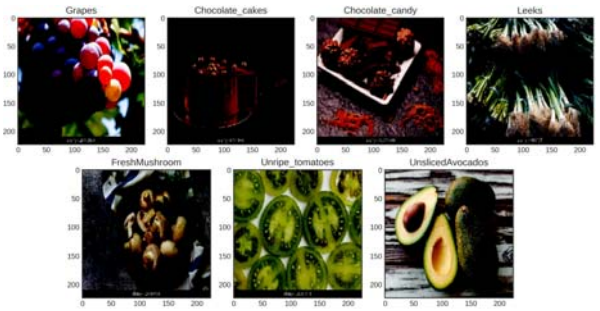
(a) True labels



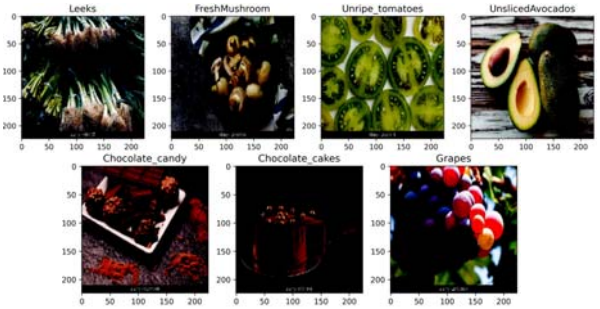
(b) Customized CNN



(c) VGG-16



(d) InceptionV3



(e) Xception

Fig. 3. Performance of select trained models on custom images.

4. Conclusions

This study examined the performance of different CNN models to classify images of poisonous pet food. Overall, the Xception model consistently produced the best results. With this model, a user can apply the trained model to check if food is edible for their Labradors with an estimated accuracy of nearly 95 percent, established by the test results. Similarly, the lightweight model, MobileNetV2 also showed quite good performance. Thus, implementing this trained model in mobile applications would allow flexibility for users to perform basic image recognition tasks to check if a particular food is edible for their Labradors.

Additionally, the benefit of transfer learning was observed in this work. ImageNet pretrained models, such as VGG-16, InceptionV3 and Xception, outperformed all other models (AlexNet, Customized CNN and LeNet) that were trained from scratch using the collected images.

Furthermore, it is also important to note that the dataset used in this study was relatively small and did not cover all the possible poisonous foods for the Labradors. Therefore, it can be said that for the identified models, even though some performed notably well on the current dataset, more data is needed to further train and develop the models before they can be considered practically useful. The need for more training data is also supported by the fact that all the models showed an overfitting problem, evident from the significant discrepancy between the training error and the validation error. Future work, would focus on collecting more image data, further optimizing the best models, and exploring the practical implementation of the findings of this work.

References

[1]. American Society for the Prevention of Cruelty to Animals. The ASPCA animal poison control center celebrates 4 millionth case!, March 2022. https://www.aspca.org/news/aspca-animal-poison-control-center-celebrates-4-millionth-case.

[2]. Simran Sandhu, Ramavtar Tyagi, Elahe Talaei, and Seshasai Srinivasan, Using neurocomputing techniques to determine microstructural properties in a Li-ion battery, *Neural Computing and Applications*, 34, 2022, pp. 9983–9999.

[3]. Seshasai Srinivasan and M. Ziad Saghir, Thermodiffusion in Multicomponent Mixtures: Thermodynamic, Algebraic and Neuro-Computing, *Springer*, 2013.

[4]. Seshasai Srinivasan and M. Ziad Saghir, A Neurocomputing Model to Calculate the Thermo-Solutal Diffusion in Liquid Hydrocarbon mixtures, *Neural Computing and Applications*, Vol. 24, 2012, pp. 287−299.

[5]. Seshasai Srinivasan and M. Ziad Saghir, Predicting Thermodiffusion in an Arbitrary Binary Liquid Hydrocarbon mixtures using Artificial Neural Networks, *Neural Computing and Applications*, Vol. 25, 2014, pp. 1193-1203.

[6]. Seshasai Srinivasan and M. Ziad Saghir, Modeling of Thermotransport Phenomenon in Metal Alloys Using

Artificial Neural Networks, *Applied Mathematical Modeling*, Vol. 37, 2013, pp. 2850−2869.

[7]. G. Side, S. D. Bhole, D. L. Chen, and E. Essadiqi. Determination of volume fraction of bainite in low carbon steels using artificial neural networks, *Computational Materials Science*, Vol. 50, 2011, pp. 3377-3384.

[8]. Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, 86, 11, 1998, pp. 2278–2324.

[9]. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556, 2014.

[10]. Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2016)* , 2016, pp. 770–778.

[11]. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton, Imagenet classification with deep convolutional neural networks, in F. Pereira, C. J. Burges, L. Bottou, and K. Q. Weinberger (Eds.), Advances in Neural Information Processing Systems, Vol. 25, *Curran Associates, Inc*., 2012.

[12]. Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2015),* 2015, pp. 1-9.

[13]. Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna, Rethinking the inception architecture for computer vision, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 2016),* 2016, pp. 2818 - 2826.

[14]. Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-ResNet and the impact of residual connections on learning, in *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI' 17),* February 2017, pp. 4278–4284.

[15]. François Chollet, Xception: Deep learning with depthwise separable convolutionsin *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR' 17)*, 2017, pp. 1800 – 1807.

[16]. Andrew G. Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam, Mobilenets: Efficient convolutional neural networks for mobile vision applications, *CoRR*, abs/1704.04861, 2017.