Journal of Ambient Intelligence and Humanized Computing (2023) 14:13971–13979
https://doi.org/10.1007/s12652-022-04105-3

ORIGINAL RESEARCH

Automated vehicle inspection model using a deep learning approach

Mohamed Mostafa Fouad[1] · Karim Malawany[1] · Ahmed Gamil Osman[1] · Hatem Mohamed Amer[1] · Ahmed Mohamed Abdulkhalek[1] · Abeer Badr Eldin[1,2]

Received: 28 August 2021 / Accepted: 6 June 2022 / Published online: 3 July 2022

Abstract

Image-based inspection is a growing area with a large scope of automation. The automatic classification of vehicle damages would make the insurance claim much faster and more efficient. This can effectively reduce the claiming cost. This paper presents, an image classification model using an adapted version of pre-trained convolutional neural networks. The pre-trained neural networks were, the VGG-19 and DenseNet-169. The proposed model is a pipeline that established with fully connected layers for additional damage classification. The final proposed model improves the feature extraction process. The dataset had a class imbalance problem, so a weighted loss function had been used to solve such problem. The model employed binary cross-entropy as a loss function, and sigmoid activation was applied to the output layers as independent layers. Finally, the model presents a multi-label classifier, where one image may be assigned to many labels. The model classifies vehicle damage through five classes: broken glass, broken headlights, broken taillights, scratches, and dents. A four-layer neural network was employed for the classification, along with several regularization approaches to handle overfitting problem. The final results showed that the DenseNet-169 had a better accuracy of 81%, whereas VGG-19 had a 78%. Another approach had been proposed where it had a mix of transfer and ensemble learning approaches. This final approach had an accuracy of 85.5% and F1-scores of 0.855.

Keywords Deep learning · Computer vision · Deep convolutional neural networks · Transfer learning

1 Introduction

In different tasks such as computer vision, robotics, and natural language processing, Artificial Intelligence (AI) technologies have proved to be very accurate for decision making. According to (Pleiss et al. 2017), the classification using the DenseNet had proven the final model accuracy. DenseNet architecture was used to reduces both the number of parameters and the computation time in terms of the number of floating-point operations. The main advantage of the architecture of the DenseNet is the feature reuse. The extracted feature of all previous layers was used to feed the new DenseNet layer. The accumulated feature attached to these types of networks nominated the DenseNet to be used in different applications. Ruiz et al. (Ruiz et al. 2020) used three DenseNet models and ensemble learning for Alzheimer's disease detection from the 3D MRI images. While each of the three networks was trained individually, the outputs of networks were then fused using probability-based fusion. The accuracy achieved by the first model reached 53.33%, and the second model accuracy reached 57.50%. Finally, the accuracy of the last proposed model reached 66.67%. The total accuracy gained from the fusing of the three models was 83.33%. The researchers in (Ahuja et al. 2021) had developed a new deep learning pipeline to classify COVID-19, and tuberculosis from the X-Ray images. The pipeline proposed a hybrid model from several

✉ Mohamed Mostafa Fouad
mohamed_mostafa@aast.edu

Karim Malawany
Karim.Malawany@yahoo.com

Ahmed Gamil Osman
Ahmed.Gamil@student.aast.edu

Hatem Mohamed Amer
Hatem.amer1@hotmail.com

Ahmed Mohamed Abdulkhalek
hoda199977@yahoo.com

Abeer Badr Eldin
abadredin@hotmail.com

[1] Arab Academy for Science, Technology and Maritime Transport, Cairo, Egypt

[2] Department of Computers and Information Systems, Sadat Academy for Management Sciences, Cairo, Egypt

pre-trained deep learning models [ResNet (He et al. 2016a, b), DenseNet, Inception-V3 (Szegedy et al. 2016). and VGG-16 (Simonyan et al. 2014)]. The new hybrid model had proposed an extra layer with convolutional neural network blocks to combine both ResNet and DenseNet models (DenResCov-19). A comparison between the models was conducted, by the researchers, using different datasets. The DenResCov-19 proved its efficiency with a higher F1-score per each tested dataset.

In most cases, insurance companies require inspectors, who can take hours or days to examine the scene of an accident. Damage assessment knowledge might be turned into an AI challenge, using images taken by the client and processed automatically by the insurance company. Automation of the vehicle damage inspection system may reduce waiting time and give the insurance claim an effective approach to react quickly. In much-related research of (Patil et al. 2017), were the researchers used a vehicles' damages dataset that was collected through the web crawler. The paper suggested eight classes for damage classifications (bumper dent, scratch, door dent, glass shatter, broken headlights, broken taillights, smashed and not damaged). The collected dataset was split into 80% of the data for training and 20% for testing. The researchers had tested damage classification using transfer learning and ensemble learning approaches where the achieved accuracy reached 89.5%. Damage type, location, and severity were all detected using multiple datasets in (HV et al. 2019). The data were collected by Google Search Scraping. The acquired data was divided into three groups, with each class having its own dataset. The first classification was based on the type of damage (dent, glass, hail, and scratches). The second category showed the damage location (front, rear, top, and sides). The last dataset represents the damage severity (small, medium, and large). The model used the pre-trained VGG-16 model. The model classification accuracies for the type of detection, location, and severity of the damage, were 75.1%, 68.7%, and 54.2% respectively. These accuracies were improved after detecting overfitting and coping with the model's poor learning rate.

VGG-16 and VGG-19 were tested on vehicles' damaged classification in (Simonyan et al. 2014). The damage severity of both models showed a similar performance (Sruthy et al. 2021). The model achieved detection accuracy percentage of 94%, 71%, and 61% in damage detection, damage location, and damage severity in VGG-16, respectively. The accuracy of damage localization was 74.39% in the first model and 76.48% in the second model. The damage severity recognized from both models were 54.8% and 58.48% respectively.

Through a comparison of two transfer learning models, this paper proposes automation for damage inspection process. The research recommends that the transfer categorization of learning be used to differentiate between the represented classes: broken glass, broken headlights, damaged taillights, and dents. The small size of the vehicle damage dataset was one of the challenges. As a result, web scrapers were used to create the dataset (Fig. 1), which was then manually labelled. Different image classification systems, such as DenseNet and VGG-19, were evaluated to see which obtained the highest results. Fully connected layers were also developed to speed up the training process by using batch normalization and dropouts to minimize the problem of overfitting. The model also contains a multi-label classifier that helps classify images into multiple labels, making it more commercially viable.

In addition to the introduction section, this paper is divided into six sections, which are ordered as follows; the second section introduces the key concepts and approaches. The dataset description that was used is summarized in the third section. This section outlines the data collection process. In the fourth section, the issue of class imbalance is briefly examined. In the fifth section, the experiments are outlined, and the findings are summarized and discussed in the next section. The conclusion of the paper is presented in the final section.

2 Preliminaries

2.1 Transfer learning

In many scenarios, collecting enough data for training is a difficult, costly, and time-consuming process. Therefore,

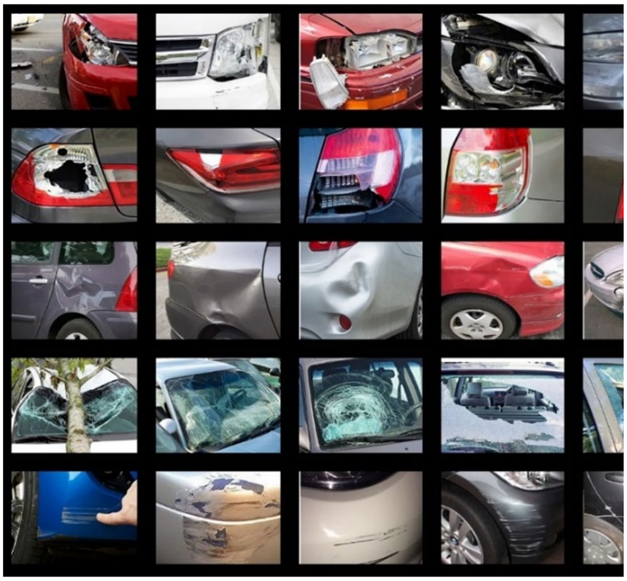

Fig. 1 Samples of collected data. Rows from top to bottom indicate damage types: broken headlights, broken taillights, dents, broken glass, Scratches

transfer learning has become the best choice for many machine learning applications. Its main function is to transfer the knowledge learned by a model using a large general dataset from another domain. Transfer learning was a suitable choice for solving the proposed problem due to the limited amount of available images. According to (Zhuang et al. 2020), using transfer learning does not guarantee a positive result, especially if the intersection between domains is limited, which creates a negative transfer phenomenon (Wang et al. 2019). Figure 2 illustrates the general model for the transfer learning approach.

2.2 Dense convolutional network (DenseNet)

While the traditional convolutional network model connects layers through connections directly, DenseNet (Huang et al. 2017) introduces the Dense Block. Dense Block connects all layers directly as a block. Figure 3 shows the DenseNet flow, where each layer receives variation in the input from all prior layers and transmits the generated feature maps to all succeeding layers. The model is based on using Deep Residual Network (ResNet) in (He et al. 2016a, b) and Inception Network introduced in (Ioffe et al. 2015). The model had improved the efficiency of the prediction through the augmentation of the feature-maps learned by all prior layers as expressed in Eq. 1, where layers $x_0,\ldots, x_{\ell-1}$, are used to feed the next layer x_ℓ.

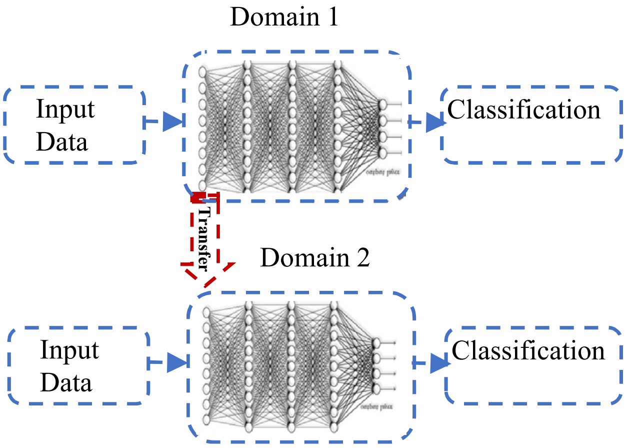

Fig. 2 The general model for transfer learning

$$x_\ell = H_\ell\left(\left[x_0, x_1, \ldots, x_{\ell-1}\right]\right) \tag{1}$$

where H_ℓ is a pipeline of non-linear transformation functions (Huang et al. 2017). The main issue with the DenseNet is the extensive large memory utilization according to (Lu et al. 2021).

2.3 Overfitting and long training time problems

The DenseNet as a multilayer neural network may suffer from one or both well-known problems entitled: overfitting and long training time. Both problems are caused by the designing principle of the neural network. The researchers (Garbin et al. 2020) have discussed these issues and have provided answers to issues such as the following:

2.3.1 Dropout approach

Overfitting is a concern since it leads to an increase in testing errors when training errors are low. To mitigate the effects of such a problem, a number of approaches have been developed. When combining many models into a single ensemble model, these methods are very effective (Goodfellow et al.

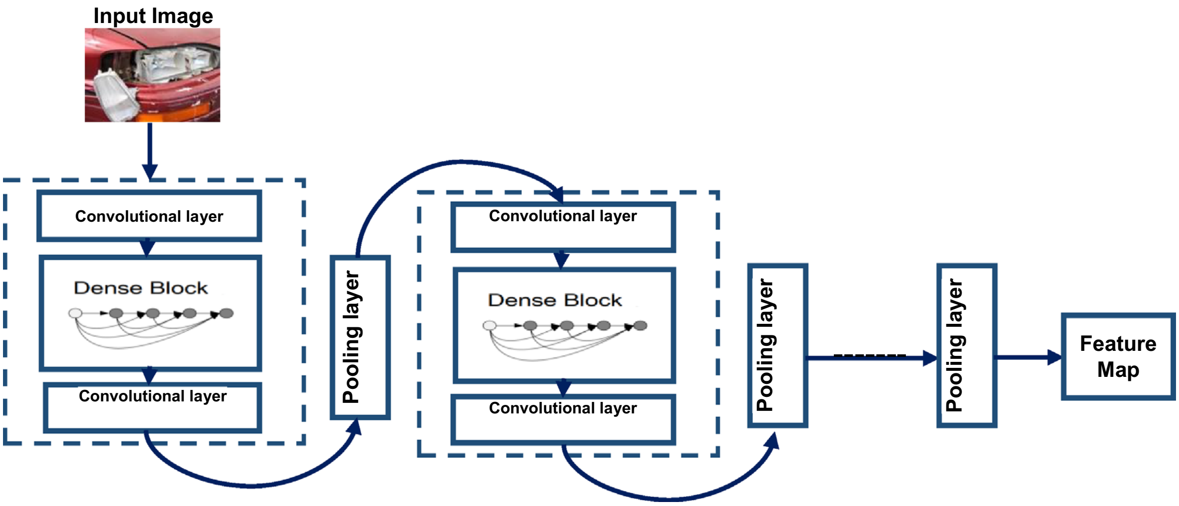

Fig. 3 The general architecture for the DenseNet network

2017). Because each model requires a significant amount of training time, the key problem with the ensemble model is the accumulative time. The addition of the dropout layer to the network architecture is another approach to the overfitting problem (Srivastava et al. 2014). This layer drops out a unit in a neural network, which means temporarily removing units together with its outgoing and ingoing connections. Dropout is randomly performed based on probabilistic equation per each iteration. The random removal of training also means that, at any time, only part of the original network is formed and the original neural network is divided into multiple sub-networks. The previous statement states that the neural network cannot rely on a certain feature and must therefore spread it's out weights.

2.3.2 Batch normalization approach

The reduction of time in the training phase can be done through adapting the initial weights or reducing the learning rate. The dependence of a layer on its preceding layers complicates the learning process even further. A small change in one layer can be amplified in the succeeding layers. The batch normalization (Ioffe et al. 2015) provided a good solution to such training-based delay problem. The approach normalizes the inputs per layer which makes a deep neural network faster and more stable. The standardizing and normalizing operations are performed by adding a new layer to the output of a previous layer. The process includes the calculation of the mean and the standard division of the batch. Then subtracting the mean and dividing the whole input by the standard division and the smoothing term (ε) (Ioffe et al. 2015). The smoothing term (ε) assures numerical stability within the operation by stopping a division by a zero value. The standard division and the mean are entitled gamma and beta, are trainable parameters that get updated during backward propagation.

3 Dataset description

The data was scraped from the Google, Bing, and DuckDuckGo search engines using a web scraper. Broken glass, broken headlights, broken taillights, dents, and scratches are the five categories. The dataset includes 5853 photos, with 70% of them being utilized for training and 30% for validation and testing.

Because the test data was so little, a 653-picture dataset from (The Peltarion cloud platform, 2020) as used as augmentation to the collected data. Both datasets were manually labelled. Table 1 gives a summary of the number of images available for each class.

Table 1 The number of images in each class in different datasets

Classes	Training set	Validation set	Testing set
Broken headlights	484	207	71
Broken taillight	434	186	74
Dents	984	421	259
Scratches	1070	458	174
Broken glass	1122	481	75

4 The proposed model

4.1 Data preprocessing and augmentation

Data preprocessing is an essential step in building a model for machine learning. This process is responsible of preparing the input data to be understood by any proposed machine learning model. The proposed preprocessing for the current paper was the images resizing to 150 × 150 pixels RGB. Pixels' values are rescaled to a value between 0 and 1.

Since few samples of data are provided in such areas of interest, image augmentation was introduced in the model. It is a tactic used to expand the data set artificially without collecting new data. Usually, methods of data augmentation modify the shapes through shifting, flipping, brightness. Another method used is the shape's geometry such as rotation, zoom, and scaling. Augmenting image data is used for expanding the training data set to improve model performance and reduce the overfitting phenomena of the training set. The technique was applied randomly to the training set. The model applied a pipeline of different transformations such as rotation by 40 degrees, 0.2 horizontal shift, 0.2 vertical shift, 0.4 zooming, and by flipping the images horizontally. Figure 4 show illustration of the image augmentation processes.

4.2 The problem of the class imbalance

Figure 5 shows that there is a noticeable class imbalance in the newly acquired dataset. A situation in which the number of samples from one class is much lower than the number of samples from the other classes. When building a classifier, it can be quite an issue because it forces the model to become biased toward specific classifications. Several strategies have been developed to improve the accuracy of the machine learning model and minimize the biased effect. To balance all of the classes, one of these techniques is to collect more data. However, in the current automobile accident use case, obtaining this solution is challenging. As a result, a weighted loss function is applied to balance the learning process, this approach was first introduced in (Xie et al. 2015) as a simple method for automatically balancing

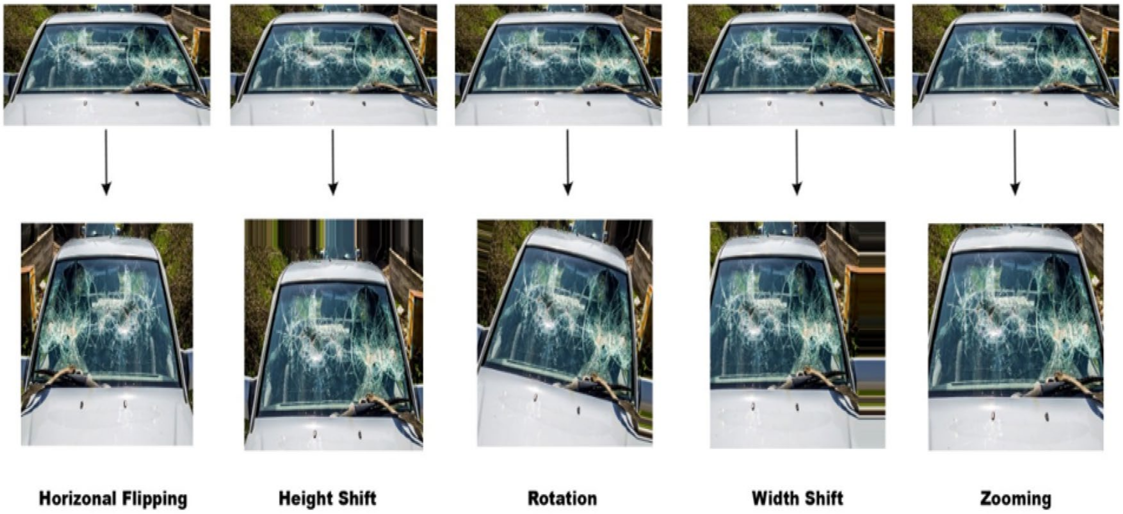

Fig. 4 Image augmentation processes

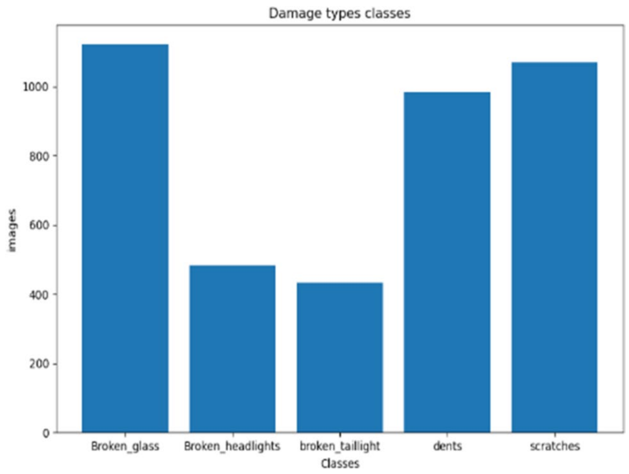

Fig. 5 Damage type classes per training set

negative and positive classes, and it shown considerable benefits (Cui et al. 2019). The function adds weight to the loss function, which is responsible for balancing the positive and negative labels of each data set.

The model used binary cross-entropy (BCE) to calculate the loss function. The network's output layer has a sigmoid nonlinearity function, to optimize the probability of correctly classifying input data. Cross-entropy is suggested here since it is the appropriate choice for classification, while mean squared error (MSE) is one of the best choices for regression error (Song Yang. 2021). The mean square error is obtained by assuming the objective is continuous and normally distributed and maximizing the probability of DenseNet's performance under these assumptions.

The proposed model calculates the positive and negative frequencies per each class (Eq. 2), as the first step to weigh the function. It adjusts the weight and multiplies them to the loss function. The weighted BCE loss can be explained mathematically through Eq. 3.

$$w_{\mathrm{pos}} = \mathrm{freq}_{\mathrm{neg}},\ w_{\mathrm{neg}} = \mathrm{freq}_{\mathrm{pos}} \tag{2}$$

$$\frac{1}{N}\sum_{i=1}^{N} w_{\mathrm{pos}} y_i \log\left(p\left(y_i\right)\right) + w_{\mathrm{neg}}\left(1 - y_i\right)\log\left(1 - p\left(y_i\right)\right) \tag{3}$$

where N represents the number of training examples, y is the true label for training examples N, $p(y)$ is the predicted label for training examples N, W_{pos} represents the positive calculated weights, and W_{neg} represents the negative calculated weights.

5 Experiments and training

DenseNet-169 (Huang et al. 2017) was used as a pre-trained model. All layers were frozen and the top fully connected layers were removed. The pre-trained model was used to extract features from data then fully connected layers were implemented to act as a classifier. The classifier consists of 4 blocks, each block contains a dense layer followed by Batch normalization (Ioffe et al. 2015), ReLU activation, and a dropout layer. Dense layers have 2048, 1024, 512, and 256 units, respectively. The output layer contains 5 units with sigmoid activation $\frac{1}{1+e^{-x}}$ since a multi-label classifier is being developed.

During training Adam optimizer (Huang et al. 2017) was used since it had improvements over the Stochastic Gradient Descent and it was less prone to overfitting. Also, Adam required less memory. It combines the advantages of extensions of Stochastic Gradient Descent, Adaptive Gradient Algorithm (AdaGrad), and Root Mean Square Propagation (RMSProp) in (Kingma et al. 2015).

Instead of using the average first moment (the mean) like in RMSProp, Adam uses the average of the second moments of the gradients to adjust the parameter learning rates. The algorithm creates an exponential moving average of the gradient and the squared gradient, with the β_1 and β_2 parameters controlling the decay rates of these moving averages. The bias of moment estimations towards zero is caused by the initial value of the moving averages of β_1 and β_2 values near 1.0 (recommended). To overcome this bias, first, calculate the flawed estimates, then calculate the bias-corrected estimates. *Algorithm1* shows in more depth the Adam optimizer.

Algorithm 1: Adam optimizer (Patterson et al. 2017.)

```
$m_0 \leftarrow 0$ $(Intialize\ 1^{st}\ moment\ vector)$
$v_0 \leftarrow 0$ $(Intialize\ 2^{nd}\ moment\ vector)$
$t \leftarrow 0$ $(Intialize\ timestep)$
$While\ \theta_t\ not\ converged\ do$
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1}) \left(\begin{matrix} Get\ gradients\ w.r.t\ stochastic \\ objective\ at\ timestep\ t \end{matrix}\right)$
$m_t \leftarrow \beta_1 . m_{t-1} + (1-\beta_1) . g_t \left(\begin{matrix} Update\ biased\ first \\ moment\ estimate \end{matrix}\right)$
$v_t \leftarrow \beta_2 . v_{t-1} + (1-\beta_2) . g_t^2 \left(\begin{matrix} Update\ biased\ second\ raw \\ moment\ estimate \end{matrix}\right)$
$\widehat{m}_t \leftarrow \frac{m_t}{(1-\beta_1^t)} \left(\begin{matrix} Compute\ bias - corrected\ first \\ moment\ estimate \end{matrix}\right)$
$\hat{v}_t \leftarrow \frac{v_t}{(1-\beta_2^t)} \left(\begin{matrix} Compute\ bias - corrected\ second\ raw \\ moment\ estimate \end{matrix}\right)$
$\theta_t \leftarrow \theta_{t-1} - \alpha . \frac{\widehat{m}_t}{\sqrt{\hat{v}_t + \epsilon}} (Update\ parameters)$
```

A mini-batch size of 64, weight decay of 0.0001 (l2 regularization) were used. The pre-trained model is followed by batch normalization before feeding its output to the fully connected layers. Each dense layer is followed by batch normalization before applying ReLU activation and a dropout after, with the value of 0.7, 0.6, 0.5, and 0.4 respectively for each layer. Figure 6 shows the main building blocks of the proposed vehicle's damage inspection model.

The initial learning rate value was set to 0.001. A learning rate decay technique was used; it works as follows:

1. Monitor validation accuracy metric per epoch.
2. If it does not improve for 5 epochs reduce the learning rate by a factor of 0.3 (different factors were experimented with but 0.3 was the best fit for the model)
3. Go to step 1.

This technique was used to improve the model accuracy and decrease loss while speeding up the training process. The learning rate was decreased three times while training for 50 epochs. Experiments were also done without applying learning rate decay to observe the difference.

As shown in Fig. 7 the loss is very unstable during training with a fixed learning rate. This is probably caused because the learning rate is too large. Decreasing the learning rate manually is very time-consuming, therefore using a learning rate decay approach, as shown in Fig. 8 was the appropriate solution (Ioffe et al. 2015).

6 Results and discussion

6.1 Comparison between VGG-19 and DenseNet

The VGG-19 is a convolutional neural network developed by the Visual Geometry Group at the University of Oxford's Department of Engineering Science. That's a total of 19 layers. It was trained on the 1000-class classification task from the ImageNet challenge (ILSVRC). Various pre-trained models were employed in the construction of the model during the experiment to compare the accuracy of each model. VGG-19 (Simonyan et al. 2015) was chosen to be compared with DenseNet-169. Both models were used with the same

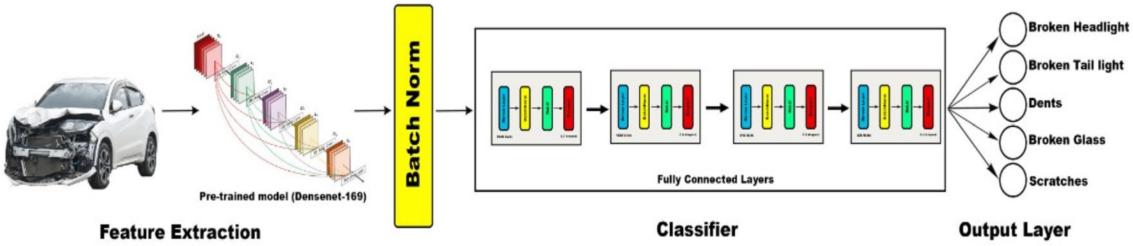

Fig. 6 Model architecture

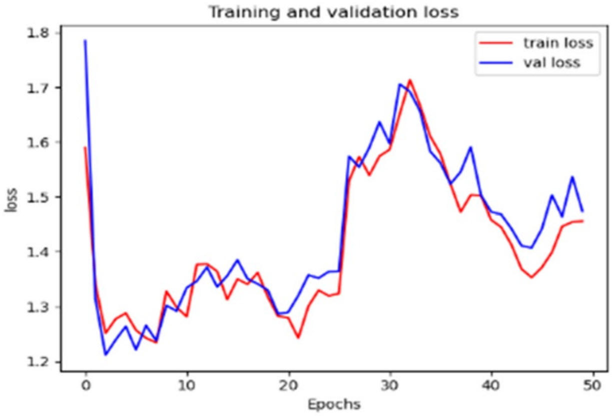

Fig. 7 Training while applying learning rate decay

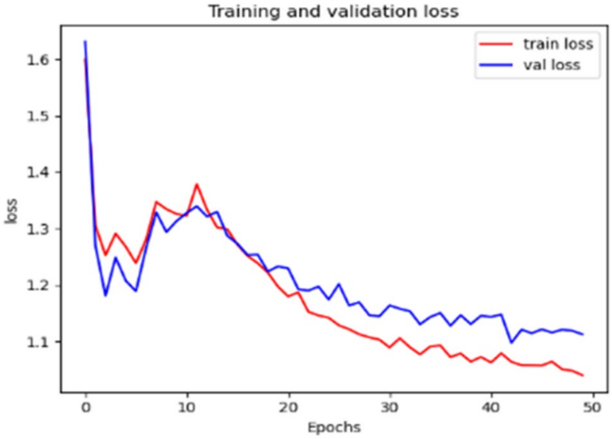

Fig. 8 Training with fixed learning rate

Table 2 DenseNet-169 and VGG-19 evaluation criteria

Reference	Methodology	Accuracy
(Patil et al. 2017)	AlexNet	89.89%
(HV et al. 2019)	VGG-16	75.1%
(Sruthy et al. 2021)	VGG-16, VGG-19	54.8%, 58.48%

fully connected layers mentioned and all hyperparameters were similar. Preprocessing of the data in the VGG-19 model is not trained on normalized data. When utilizing VGG-19, just the mean was eliminated from the dataset (Kingma et al. 2015). The model accuracy (Eq. 4) and the F1-score (Eq. 5) were used to compare the two models for evaluation

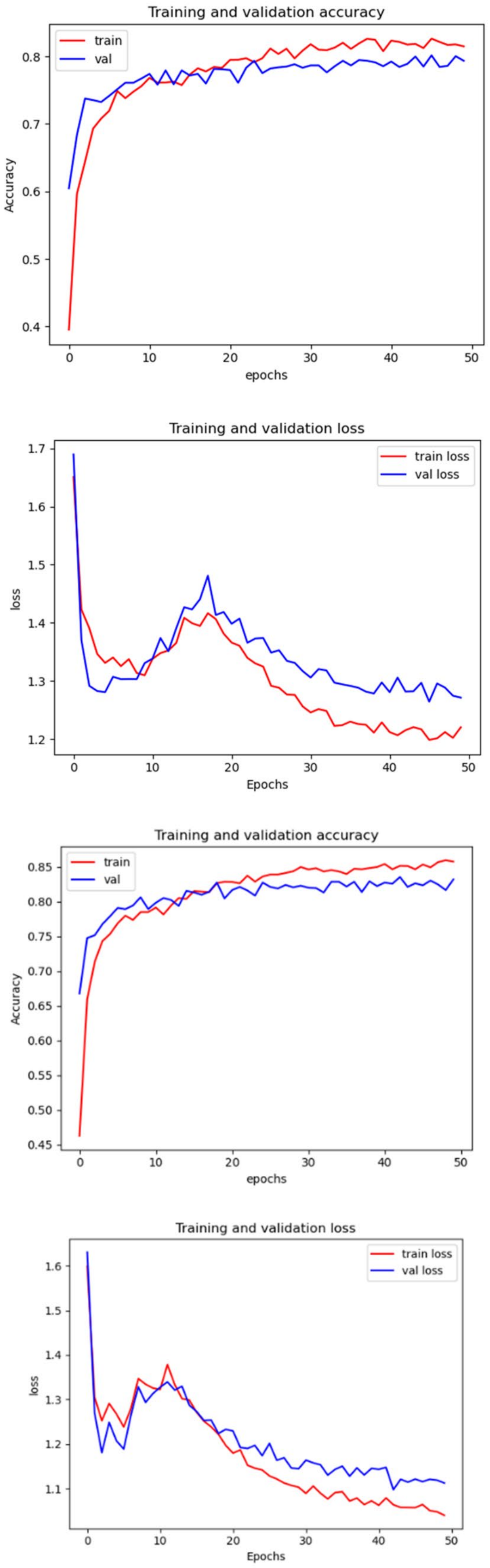

Fig. 9 Validation and training accuracy, loss (left, right) in VGG-19 (top) and DenseNet-169(Bottom) ▶

purposes. Table 2 displays the obtained results as well as the DenseNet's advantage over the VGG-19.

$$\text{Accuracy} = \frac{\text{TrueNegative} + \text{TruePositive}}{\text{TruePositive} + \text{FalsePositive} + \text{TrueNegative} + \text{FalseNegtive}} \tag{4}$$

$$F_1 = 2 \times \frac{\text{Percision} \times \text{recall}}{\text{Precision} + \text{recall}} \tag{5}$$

DenseNet has a higher accuracy than VGG-19, as seen in Table 2 and Fig. 9. This could be due to various of reasons, one of which being the fact that DenseNet is considerably more comprehensive than VGG-19. DenseNet also aids in the reduction of vanishing gradients and is significantly faster in terms of training. The scores on the testing datasets are provided in Table 2 as an outcome. This also demonstrates how transfer learning can be effective when only a little amount of data is supplied. These models can efficiently extract new features from a new dataset using features learned from prior datasets. Because two models were built, stacking both models are an effective approach to increase accuracy. Either by combining the findings of a number of different base models. Either by integrating the results of many base models or selecting the "best" base model (Ganaie el at. 2021).

Another experiment was carried out by constructing an ensemble learning of both of the previously described models, each of which was trained separately. The two models were merged together. Merging the results of the two models yields class probability forecasts for each testing image. The collected findings revealed some improvements with an accuracy of 85.5 percent and an F1-score of 0.855. A quick summary of the differences between the models' accuracies is presented in Table 3.

7 Conclusion

This paper proposed a model for vehicle inspection using deep learning. Since there was no access to any vehicle damage dataset, the data were collected using a web scrapper then manually labeled. The class imbalance was noticed and handled by implementation a customized loss function. Experiments were done using multiple pre-trained models while using a fixed classifier to test their performance and accuracy. DenseNet-169 achieved the highest scores with 81% accuracy and 0.83 F1-scores, while VGG-19 achieved 78% accuracy and 0.80 F1-scores. Finally, an ensemble learning approach was proposed to further improve the model, the final accuracy and F1-score were 85% and 0.85, respectively.

Table 3 Current approaches and results in damage type classification

	VGG-19	DenseNet
Accuracy	78%	81%
F1-score	0.80	0.83

Funding Open access funding provided by The Science, Technology & Innovation Funding Authority (STDF) in cooperation with The Egyptian Knowledge Bank (EKB).

References

Ahuja S, Panigrahi BK, Dey N, Rajinikanth V, Gandhi TK (2021) Deep transfer learning-based automated detection of COVID-19 from lung CT scan slices. Appl Intell 51(1):571–585

Cui Y, Jia M, Lin TY, Song Y, Belongie S (2019) Class-balanced loss based on effective number of samples. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 9260–9269.

Ganaie MA, Hu M (2021) Ensemble deep learning: A review. arXiv preprint arXiv:2104.02395. Accessed 14 Apr 2022

Garbin C, Zhu X, Marques O (2020) Dropout vs batch normalization: an empirical study of their impact to deep learning. Multimed Tool Appl 79(19):12777–12815

Goodfellow I, Bengio Y, Courville A (2017) Deep learning (adaptive computation and machine learning series). Cambridge Massachusetts. MIT Press, Cambridge, pp 321–359

He K, Zhang X, Ren S, Sun J (2016a) Identity mappings in deep residual networks. European conference on computer vision. Springer, Cham, pp 630–645

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

Huang G, Liu Z, Van Der Maaten L, Weinberger KQ (2017) Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 4700–4708.

Ioffe S, Szegedy C (2015) Batch normalization: accelerating deep network training by reducing internal covariate shift. In International conference on machine learning. pp. 448–456. PMLR.

Kingma DP, Ba J (2015) Adam: a method for stochastic optimization. In Proceedings International Conference Learning Representations. pp. 1–41.

Lu H, Yang R, Deng Z, Zhang Y, Gao G, Lan R (2021) Chinese image captioning via fuzzy attention-based DenseNet-BiLSTM. ACM Trans Multimed Comput, Commun, Appl 17(1s):1–18

Patil K, Kulkarni M, Sriraman A, Karande S (2017) Deep learning based car damage classification. In 2017 16th IEEE international conference on machine learning and applications (ICMLA). pp. 50–54. IEEE.

Patterson J, Gibson A (2017) Deep learning: a practitioner's approach. "O'Reilly Media, Inc".

Pleiss G, Chen D, Huang G, Li T, Van Der Maaten L, Weinberger KQ (2017) Memory-efficient implementation of dense nets. arXiv preprint arXiv:1707.06990.

Ruiz J, Mahmud M, Modasshir M, Shamim Kaiser M, Alzheimer's Disease Neuroimaging Initiative, F. T (2020) 3D Dense Net ensemble in 4-way classification of Alzheimer's disease. International Conference on Brain Informatics. Springer, Cham, pp 85–96

Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556. Accessed 20 Apr 2022

Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. In 3rd International Conference on Learning Representations (ICLR). pp.1–5.

Song Y (2021) What is the different between MSE error and cross-entropy error in NN. Cooking&amp;Coding Girl. http://neuralnetworksanddeeplearning.com/chap3.html. Accessed 22 May 2022

Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R (2014) Dropout: a simple way to prevent neural networks from overfitting. J Mach Learn Res 15(1):1929–1958

Sruthy CM, Kunjumon S, Nandakumar R (2021) Car damage identification and categorization using various transfer learning models. In 2021 5th International Conference on Trends in Electronics and Informatics (ICOEI). pp. 1097–1101. IEEE.

Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition. pp. 2818–2826.

The Peltarion cloud platform, the "Car Vehicle Damage Assessment.", 2020. https://peltarion.com/knowledge-center/documentation/tutorials/car-damage-assessment. Accessed 20 Apr 2022

Wang Z, Dai Z, Póczos B, Carbonell J (2019) Characterizing and avoiding negative transfer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 11293–11302.

Xie S, Tu Z (2015) Holistically-nested edge detection. In Proceedings of the IEEE international conference on computer vision. pp. 1395–1403.

Yashaswini HV, Karthik V (2019) Car damage detection and analysis using deep learning algorithm for automotive. Int J Sci Res Eng Trends 5(6):1896–1898

Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, He Q (2020) A comprehensive survey on transfer learning. Proc IEEE 109(1):43–76