Vehicle Damage Classification

Mudit Bhargava
University of Massachusetts Amherst
mbhargava@cs.umass.edu

Abstract

The paper is an attempt to solve the problem of vehicle damage classification. Given a damaged car image, the task is to classify the damage as 'Minor', 'Moderate' or 'Severe'. Due to the scarcity of freely available data, transfer learning is used. Within transfer learning I propose a two-stage fine-tuning process, where the pre-trained VGG-16 is first fine-tuned on the Stanford Car dataset followed by fine-tuning on the target, damaged car dataset. The two-stage transfer learning method obtains a best accuracy of 74.22% on the test set, which is slightly higher than the 71% accuracy reported by any previous work on the same problem. The project is a part of CS682 course project and is not shared with any other class

1. Introduction

Over the last few years a lot of progress has been made in the domain of object recognition[1][10]. Yet there are many interesting sub-problems that have not been explored as much. One of them is Vehicle Damage Recognition & Classification. Vehicle damage recognition problem is the ability to identify if a car is damaged or not. Vehicle damage classification on the other hand is the ability to classify the extent of damage. The classification problem is a super-set of damage recognition, since it needs to recognize the damage before classifying it. A solution to such a problem can directly be used to speed up tedious and complicated claims processing in vehicle insurance industry. It can also form the basis for solving a more difficult problem of automatically detecting & classifying severity of on-road accidents using street cameras. Although Convolutional Neural Networks (CNN) have proven to be highly effective for the problem of object recognition & classification, training them requires a large amount of data. This introduces the biggest challenge for the current task which is availability of large, open, standardized datasets for damaged cars. This challenge has restricted the use of CNNs on this task and hence most of the previous work on this problem was focused on traditional computer vision techniques[2][8]. Often, one may attempt to increase the available data by combining multiple datasets, but this often leads to problems. I demonstrate one such problem that I encountered in the damage recognition task due to mixing of two datasets.

With the availability of state-of-art models pre-trained on millions of images and recent advancements in understanding CNNs, transfer learning has proved to be an effective[6][11] option for learning from scarce data. Yet, the methodology can be applied in many different ways and not all may prove helpful[3]. The paper mainly makes 3 contributions

- Apply different transfer learning techniques (like feature extraction and fine-tuning) on the damage classification problem
- I experiment with a novel transfer learning technique, where the fine-tuning is divided into two stages
  - fine-tune the pre-trained model on a large car dataset
  - use the above model to further fine-tune for the damage classification problem
- An analysis of how mixing two datasets resulted in suspiciously high accuracy and how previous work may have overlooked the problem

2. Related Work

Most similar work to this project comes from a project done by Ting Neo[5] where she used transfer learning on damaged cars to recognize damaged cars and predict the severity of damage. The damaged car dataset (described in section 3) used for this project is built upon the dataset prepared by her. I used 140 extra images from other dataset and data augmentation (random left-right flip) to further boost the training data. As compared to transfer learning methods adopted by Ting Neo, I implement a novel two-stage transfer learning method (described in section 4.1)

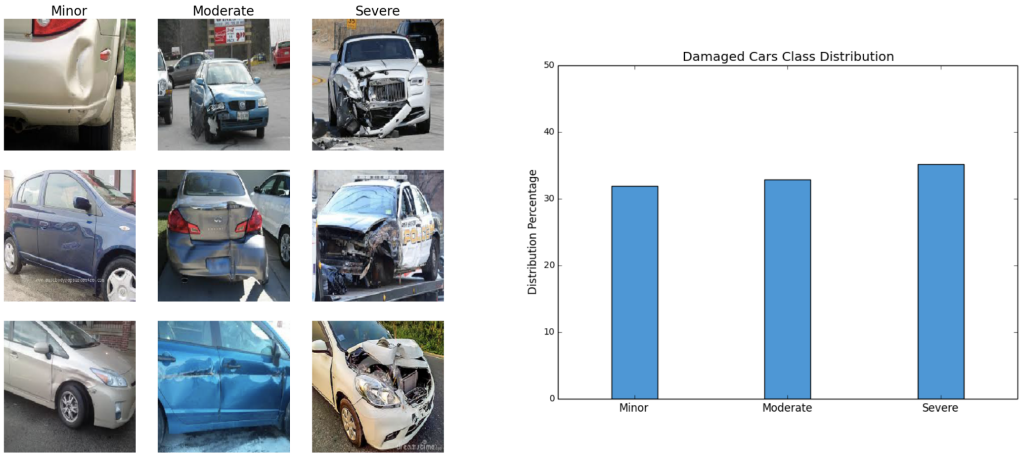

Figure 1. Sample of images of each class from damaged car data. The figure on the right shows that all classes are equally balanced in the dataset

that gives slightly better results on damage classification. Ting Neo also obtained a 92% accuracy on damage recognition by mixing good car images (from Stanford data) with damaged car images. I describe later on how this approach may not be correct and mixing images from Stanford results in the model learning things not specific to the task. Apart from this some work was also done by Kulkarni Et al[7] on damaged cars using transfer learning. The task was different where they were learning to classify the damage location rather than classifying damages. Also the dataset used by them is not publicly available. Due to the lack of large, open datasets on damaged cars most of the other approaches have focused on classical computer vision techniques. Rangaswamy Et al[8] used Hog and Sift features along with an SVM classifier for detection of damaged cars. Similarly Jaywardena et al [2] proposed a method for vehicle scratch damage detection by registering 3D CAD model of undamaged vehicle on the image of the damaged vehicle.

3. Dataset

As discussed, one of the biggest challenges in this problem is the lack of large open datasets on damaged cars. The section outlines the details of how the datasets were prepared

Damaged Cars: A small dataset, $\approx$ 1150 images of damaged cars was collected by Ting Neo[5] for a similar problem. All the images were classified as 'Minor', 'Moderate' or 'Severe'. Another small dataset, $\approx$ 600 images of damaged car images were prepared by Shanta Rangaswamy Et al[8]. These images were not classified as in the earlier dataset. To maintain the class distribution, only $\approx$ 140 images were taken from this dataset and manually labelled as 'Minor', 'Moderate' and 'Severe' and combined with the earlier dataset. The resulting dataset contains 1291 images. The class distribution and some sample images for each classes are shown in figure 1. All the 1291 images were then re-sized to 224x224 and normalized using mean and standard deviation as required by Pytorch pre-trained models

Stanford Car Dataset[4]: For one of the experiments conducted later, we fine-tune the Imagenet[9] pre-trained model on the Stanford car dataset, before fine-tuning it on the damaged car dataset. The Stanford car dataset contains around 16,185 car images annotated with type of car. The dataset has 196 unique car types. The Stanford car dataset does **not** contain any damaged car images

4. Approach

Damage Recognition:The problem of damage recognition can be formally described as a binary classification problem, where given a dataset of damaged and undamaged cars, we train a network to predict if the car in the given image is damaged or not. As I show in section 5.1, mixing damaged car dataset and undamaged car dataset resulted in unexpectedly high accuracy on the baseline. Further analysis shows that the model was learning irrelevant differences between the two datasets.

Damage Classification:The damage classification task can be described as a multi-class classification problem, where given a dataset of damaged cars, we train a network to predict if the car in the given image has 'Minor Damage', 'Moderate Damage' or 'Severe Damage'. As described in section 5.2, the damaged cars dataset alone does not suffer

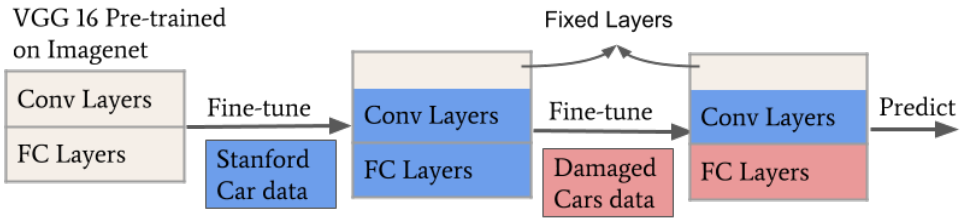

Figure 2. Transfer Learning, New Approach: The pre-trained Imagenet model is first fine-tuned on Stanford dataset and then fine-tuned on the target, damaged cars dataset

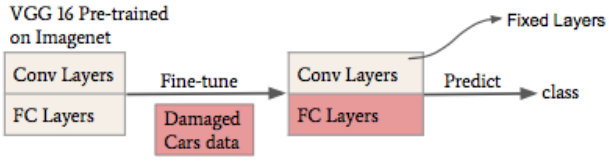

Figure 3. Transfer Learning: Traditional Approach

from the problems described above.

As mentioned in the earlier sections lack of large, open standardized car damage datasets is one of the main challenges for the damage classification task. With the availability of state-of-the art pre-trained models, we resort to Transfer Learning to overcome this problem. In the below sections, I give a brief introduction to transfer learning followed by the different ways in which it was used for damage classification task

4.1. Transfer Learning

With the recent advancement in interpretability of Convolutional Neural Networks, it was discovered that almost all the trained CNNs, irrespective of the task, learned stripes and color blobs[11]. As the layers get deeper, the features learned become more specific to the target dataset. Hence, if the source task (on which the model was pre-trained) is similar to the target task, features learned by many layers can be re-used or can even be great starting points for further training. Indeed this hypothesis was proved correct[6]. Due to the efficient re-use of pre-trained weights, training on target task can be done much faster, with lesser data as compared to starting from randomly initialized weights. Apart from that, fine-tuning on a pre-trained network also improves the generalization performance[11], an important aspect when the target training dataset is small. For all the experiments, a VGG-16 (configuration D)[10] pre-trained on Imagenet[9] was used from Pytorch pre-trained models. Since a few of the categories in Imagenet are dedicated to cars and vehicles, I expect transfer learning to be useful for car damage classification task.

Baseline Approach: The simplest way to apply transfer learning is to use pre-trained models as high-dimensional feature extractors by removing the output layer and extracting the outputs of last fully connected layer as image features. Off the shelf, classifiers can then be trained on these image features.

Traditional single-stage Fine-tuning: Feature extraction may not perform well, since models were trained on 1000 other categories apart from cars and the deeper layers in the network become very specific to the target task. One way to overcome this is to fine-tune (train with small learning rate) some or all of the layers of the pre-trained network. For the damage classification task, we fine-tune the fully connected layers of VGG-16 on the damage cars dataset. Figure 3 shows the experimental setup.

A novel two-stage fine-tuning: Unlike the traditional single-stage fine-tuning process described above, I propose a two-stage process as follows

- First, the pre-trained network will be fine-tuned on a larger Stanford Car Dataset[4] (≈16,000 images). This will help the pre-trained model to learn features more specific to cars. e.g wheels, car viewing angles, windshield etc
- After the fine-tuning on cars dataset, the model will be further fine-tuned for the actual task of classifying damaged cars. I expect this model to perform and generalize better than the traditional fine-tuning, since after fine-tuning on Stanford (a much larger dataset), the pre-trained model is closer to the final target task of car damage classification.

Figure 2 shows the basic experimental setup

4.2. Differential Learning Rates

Differential learning rate can be defined as the process of assigning different learning rates to different layers in the model as compared to training all the layers with the same

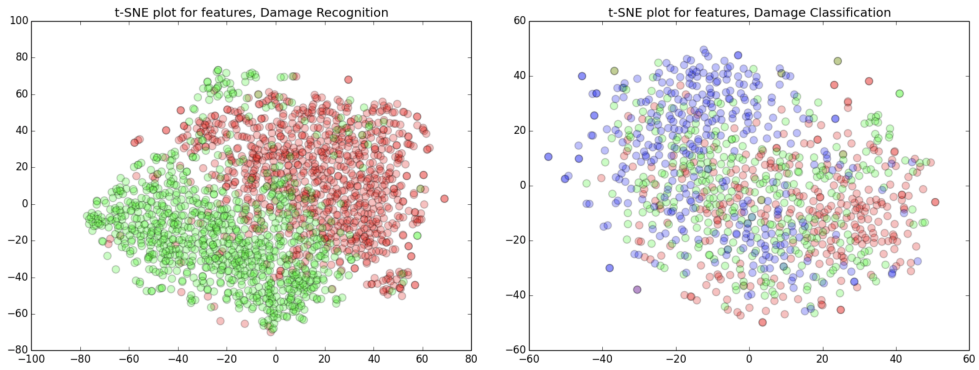

Figure 4. t-SNE plots for features extracted for damage recognition (left). The green dots represent the undamaged car image features, while the red dots represent the damaged car image features. On right, the t-SNE plot for damage classification images. The blue represent the minor class, green the moderate and red, the severe class

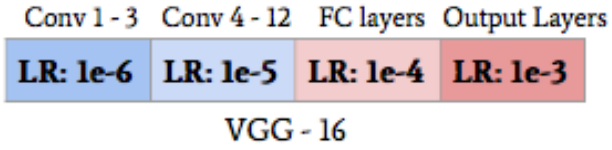

Figure 5. An example of differential learning rate

learning rate. The intuition comes from the basic concepts of transfer learning which state that earlier layers learn more generic features as compared to the latter layers. Hence we would want to change the earlier layers much lesser (lower learning rate) as compared to the latter ones (higher learning rate). Differential learning rates when applied on final fine-tuning (on damaged car dataset) task shows a slight improvement in validation accuracy. An example of differential learning rate is show in Figure 5

5. Experiments and Results

The 1291 damaged car images were randomly split into train(76.5%), validation(8.5%) and test(15%) sets. The same validation and test sets were used in all the experiments so that results could be compared. The training dataset was also boosted by random horizontal flips with a probability of 0.5. In all the training processes, the model was saved if it achieved a better validation accuracy than before. Model Ensembles were used to report the final metrics. Since its a multi-class classification task Accuracy and F1 scores were used to measure the performance of trained models. For experiments involving fine-tuning on Stanford Car dataset, all the 16,185 images were used and randomly split into train(72%), validation(8%) and test(20%)

5.1. Damage Recognition

Before I dive into experiments on the damage classification problem, I describe some experiments that were performed on damage recognition, where the task was simply to determine if a car in an image is damaged or not. As done in some previous works[5][8], the dataset for the task was prepared by taking damaged car images and mixing them with equal number of non-damaged cars from Stanford car dataset. Then a pre-trained VGG-16 was used to extract features for every image. An SVC classifier was used predict damage/no-damage on top of these high dimensional features. This simple method obtained a validation accuracy of **94.7%** and a test accuracy of **95.7%**. On test set, F1 scores for both damaged and undamaged classes was 0.96. To further debug the issue, I used the feature vectors (from pre-trained VGG model) and plotted them using t-SNE (figure 4, left plot). As it can be seen the red (damaged) and green (undamaged) dots are almost linearly separable even when reduced to 2 dimensions. This suggests a possibility that the damaged cars dataset and Stanford Car Dataset are inherently different and the model may be learning to differentiate on irrelevant differences in the datasets rather than damage on cars. To further analyze this point another experiment was conducted in which the damaged part of the 150 images was grayed out (occluded). Even after this occlusion the, the accuracy on test set slightly decreased to 89.3% and F1 score on test set for damaged class decreased a little to 0.87. Hence it can be concluded that mixing images from Stanford car dataset and damaged cars dataset, biases the model into learning irrelevant details that are able to differentiate between the two datasets with a high accuracy. These details may have been missed in the previous work by Ting Neo who reported a 92% accuracy using the same approach.

5.2. Damage Classification

In this section we discuss about experiments and results from damage classification task. The task only uses damaged car images and does not suffer from the problems described above. This can be seen from the t-SNE plot in figure 4 (right plot), where the classes don't seem to be linearly separable as in the above case.

5.2.1 Baseline Implementations & Traditional One-stage Fine-tuning

Baseline Implementations: Two baseline implementations were done

- An SVM classifier was trained on raw image pixels to predict 'minor', 'moderate' or 'major' damage. It achieved an accuracy of 46.4% on validation set and 40% on the test set [see figure 8]
- For the second baseline, high dimensional features were extracted from a pre-trained VGG-16. An SVM classifier was trained on these high dimensional features. This method gave an accuracy of 66.96% on the validation set and 62.37% on the test set [see figure 8]

Traditional One-stage Fine-tuning: As described in section 4.1, traditional one-stage fine-tuning was implemented by taking a VGG-16 pre-trained on Imagenet and fine-tuning it directly on the damaged cars dataset. Since the dataset size is small it was decided to fine-tune only the fully connected layers of VGG net to avoid any over-fitting concerns. As described in section 4.2 differential learning rates were also tried and it gave a slight improvement in the validation accuracy. A learning rate of 5e-4 was chosen for the output layer and 5e-5 was chosen for the other two FC layers. The results are shown in figure 6. The best model along with differential learning rates obtained an accuracy of 78.57% on the validation set and 70.1% on the test set.

5.2.2 New Two-stage Fine-tuning

As described in the section 4.1, we try a 2 stage fine-tuning process. In the first stage, the pre-trained VGG-16 is fine-tuned on Stanford dataset.

Stage 1: Fine-tuning on Stanford Car Dataset: The task performed is classification of car images based on their car types (196 classes). Since this is a much larger dataset, we can afford to fine-tune many layers of VGG-16. Experimental analysis found that fine-tuning from layer 4 on-wards learned better and gave a better validation accuracy. The experiment results are shown in figure 7

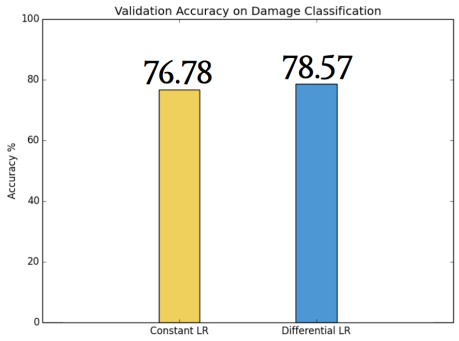

Figure 6. Effect of Differential Learning on Validation accuracy of damage classification

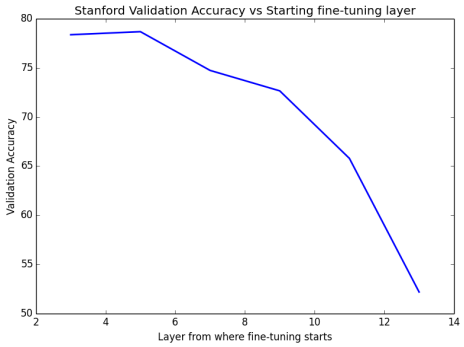

Figure 7. Effect of starting position of fine-tuning layer on Stanford validation accuracy. As the starting point for fine-tuning is pushed deeper, the validation accuracy decreases

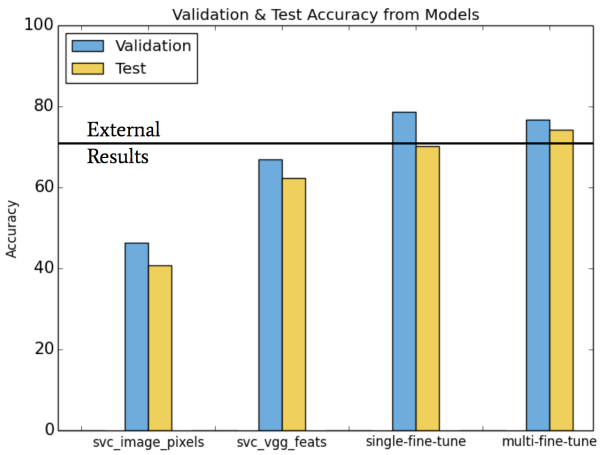

Figure 8. Test & Validation Accuracy comparisons for all the models. The black horizontal line denotes the best accuracy obtained by previous work on the same task. svc_image_pixels denotes svc classification on raw images. svc_vgg_feats denotes svc classifier on high dimensional features obtained from VGG16

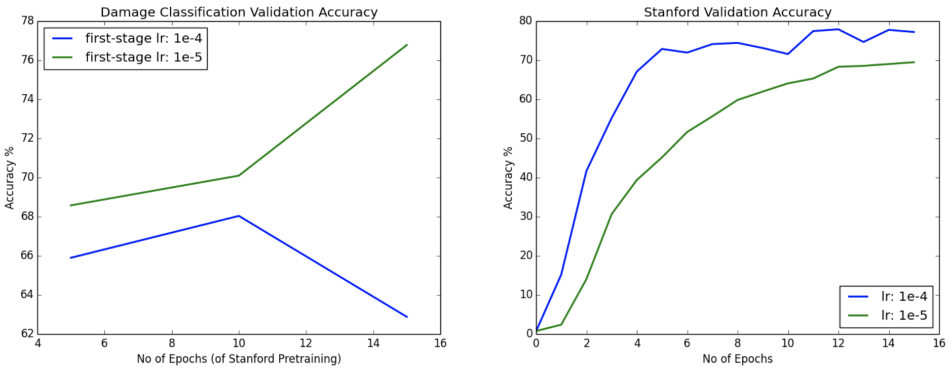

Figure 9. Effect of Stanford fine-tuning on the subsequent damage classification task

Accuracy(%)	svc_image_pixels	svc_vgg16_feats	single_fine_tune	multi_fine_tune
validation	46.42	66.96	78.57	76.78
test	40.72	62.37	70.10	74.22

Table 1. Test & Validation accuracy for all the models. svc_image_pixels denotes svc classification on raw images. svc_vgg16_feats denotes svc classifier on high dimensional features obtained from VGG16

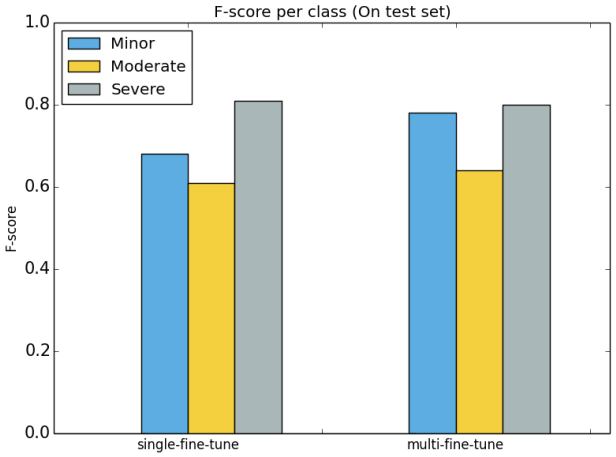

Figure 10. per class F1 scores on the test set for best performing two stage fine-tuning model

Stage 2: Fine-tuning on damaged car dataset: Models with highest validation accuracy from stage 1 are saved and used as pre-trained models for fine-tuning on damaged car dataset. With this two-stage fine-tuning process along with differential learning rates, I obtained an accuracy of 76.78% on validation set and 74.22% on the test set[see figure 8]. An interesting observation that I made during the experiments is that highest validation accuracy in stage 1 (Stanford fine-tuning) does not always correspond to highest validation accuracy in stage 2. The results are shown in figure 9. From the figure 9 (right graph) we can see that even though a learning rate of 1e-4 (blue-line) does better than learning rate of 1e-5 (green line) in terms of validation accuracy on the Stanford fine-tuning task, the trend is exactly reversed when the models are used in stage 2 (left graph). Here we see that model trained in stage 1 with learning rate of 1e-5 (green) does better in stage 2 as compared to the other (blue) model

6. Results & Analysis

The validation & test accuracy for all the models is provided in figure 8 and Table 1. The black horizontal line is the accuracy reported by Ting Neo(71%) on the damage classification task. We can see that model trained with multi-stage fine-tuning performs slightly better than Ting Neo's model and all the other models. Another thing to notice is that the difference between validation and test set accuracy is lesser for multi-stage fine-tuning method as compared to the single stage fine-tuning method which shows that multi-stage fine-tuning might be generalizing better. We also see that any form of fine-tuning, even feature extraction, pushes up the validation & test accuracy by 20% as compared to using SVM classifiers on raw pixels. Figure 10 shows the per class F1 scores on the test set for single and two-stage fine-tuning models. Overall we can see that Minor and Severe class have a good F1 score of 0.78 and 0.8, while the moderate class has a lower F1 score of 0.64. In figure 11 we analyze some of the results predicted by the best model. The first row shows some examples from Minor, Moderate and Severe that the model predicted correctly. The second row shows examples the model did not do well on and mentions what it predicted it as. The incorrect results show the subjective nature of the problem and

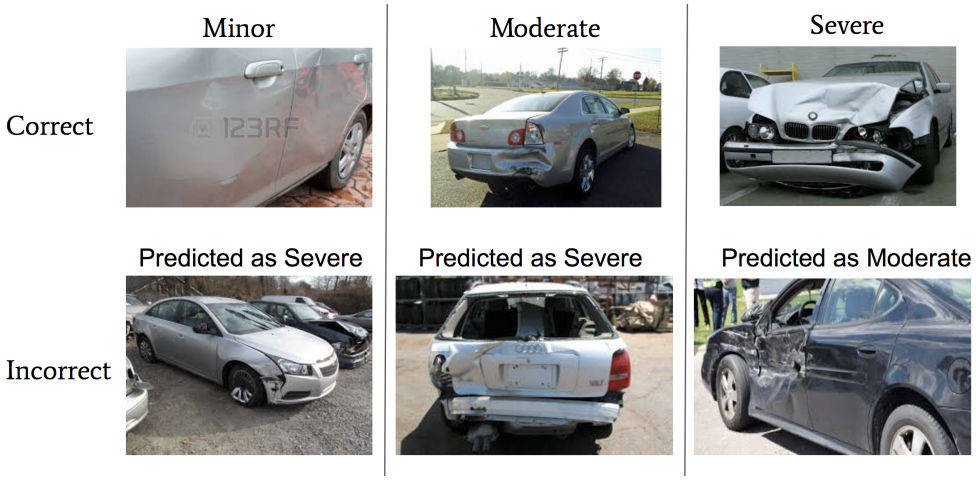

Figure 11. An analysis of the model predictions

how the data is labelled. e.g The image on second row second column that was labelled as Moderate, but predicted as Severe is debatable.

7. Conclusion and Future work

From the results and experiments in the paper we see that a two stage fine-tuning i.e fine-tuning a pre-trained model on Stanford Car Dataset and then fine-tuning on the Car damage dataset gives us slightly better accuracy & generalization for damage classification as compared to the traditional single stage fine-tuning approach. We also saw that one should be careful while mixing two independently prepared datasets and check if the model is learning as we expected. Another important aspect that came out from the experiments is that obtaining a high validation accuracy model on the first stage of fine-tuning does not correspond to high accuracy in the second stage of fine-tuning. This brings up the important question of when to stop fine-tuning in the first stage? As a future work I plan to address this question. One way to address this issue is to perform automated second stage fine-tuning after every few epochs during the first stage. But this will considerably slow down the first stage fine-tuning. Another aspect that can be addressed in the near future is to increase the data and diversity of damaged cars and standardization of the labelling process.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[2] S. Jayawardena et al. Image based automatic vehicle damage detection. 2013.

[3] A. Karpathy. Transfer Learning Methods. `http://cs231n.github.io/transfer-learning/`, 2008. [Online; accessed 19-July-2008].

[4] J. Krause, J. Deng, M. Stark, and L. Fei-Fei. Collecting a large-scale dataset of fine-grained cars. 2013.

[5] T. Neo. Assessing car damage with convolution neural networks for a personal auto claims expedition use case. `https://github.com/neokt/car-damage-detective`, 2008. [Online; accessed 19-July-2008].

[6] M. Oquab, L. Bottou, I. Laptev, and J. Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on*, pages 1717–1724. IEEE, 2014.

[7] K. Patil, M. Kulkarni, A. Sriraman, and S. Karande. Deep learning based car damage classification. In *Machine Learning and Applications (ICMLA), 2017 16th IEEE International Conference on*, pages 50–54. IEEE, 2017.

[8] V. Ravindran, L. Viswanathan, and S. Rangaswamy. A novel approach to automatic road-accident detection using machine vision techniques. *INTERNATIONAL JOURNAL OF ADVANCED COMPUTER SCIENCE AND APPLICATIONS*, 7(11):235–242, 2016.

[9] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015.

[10] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[11] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.