IMPACT: International Journal of Research in Engineering & Technology
ISSN (P): 2347–4599; ISSN (E): 2321–8843
Vol. 9, Issue 2, Feb 2021, 31–36

A STUDY ON INSURANCE AMOUNT PREDICTION FROM CLASSIFYING THE CARS

I. Muni Sai Haneesh & K.Naga Sumalatha

Research Scholar, Department of ISE, MSRIT, Bengaluru, Karnataka, India

Associate Professor, Department of Management studies, CBIT, Proddatur, Andhra Pradesh, India

Received: 08 Feb 2021 **Accepted:** 11 Feb 2021 **Published:** 28 Feb 2021

ABSTRACT

Car insurance processing and insurance amount detection is an important area with large scope for advancements and automation. In this paper we look at the problem of car damage classification, where we look at the classification whether it is damaged or not and if it is damaged, we predict the insurance amount for that damage. We explore deep learning based techniques for this purpose. Initially, we try directly training a Convolutional Neural Network. However, due to small set of labelled data, it does not work well. Finally, we experiment with ensemble learning (Random Forest Regressor model). I achieved 66.7 % accuracy in amount prediction of the damaged car and overall prediction of 30 %.

KEYWORDS: *Car Damage Classification, CNN, Random Forest Regressor*

INTRODUCTION

In the present busy world, automation is what everyone is looking for in all sectors. The insurance sector is the place where the current knowledge of Artificial Intelligence can be used to take over the human interaction in the insurance claiming process [3]. The AI can filter the list of the claims and can reduce the work.

In this paper, I employed a Convolutional Neural Network for the classification of the cars. The dataset is taken from a Hacker earth challenge - Fast, Furious and Insured. The classification is a bit complex because all the images are of car and the main feature to be extracted is whether it is damaged or not. Due to this, the construction of the neural network is bit more complex.

LITERATURE REVIEW

In the present world, car accidents are continuously increasing, car insurance companies waste more money annually, due to false claims. The AI technology and deep learning can help problems such as analysing and processing data, detecting frauds, automating the claiming process in insurance industries [2].

Since car damaged assessment is a specific domain, it lacks the publicly available datasets for car damaged images with labelling. Training a model with small dataset is the most challenging. In this case,[5] demonstrated significant progress on how classification problems when the small dataset is not enough to train a CNN model. Using data augmentation by manually collection [1, 4, and 6].

DATASET DESCRIPTION

The data consisted of 1399 images of training images of which 99 are damaged and 1300 are not damaged. As the categories of binary classification needs to be balanced, this may not be a best dataset as one category of dataset is nearly 13 times high than the other and number of images to be trained is nearly 1400. For the solution the data was split into 90 %-10 % where 90 % was used for training and the rest of the 10% is used for validation.

The training data is then increased nearly 3-4 times by the keras's augmentation of the training images. This increased the accuracy of the output model.

TRAINING THE CNN

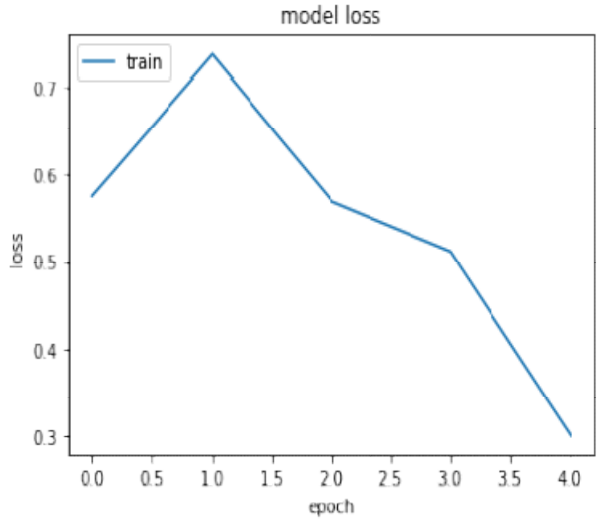
The images must be pre-processed before training as the neural network takes single input shape. The images must be resized to same dimensions.

The trained CNN model architecture consists of Conv2D, MaxPooling2D, Dense layers. The first part in the model is of 7 Conv2D layers with continuously increasing the number of neurons from 16 to 512 and the no of kernel size of 3x3 and the activation is "ReLU", in the Maxpooling2D layer the pool size of 2x2 and the rest of the parameters remaining the same. In the second phase of the model consists of 6 Dense layers with continuously decreasing number of neurons from 1024 to 64 with the ReLU activation and the last layer is of 1 neuron with "sigmoid" activation. The accuracy reached to 0.9275 from 0.8841 and remained in 0.9275 for further iterations and the loss reached 0.3002 from 0.5764 with a peak at 2nd epoch to 0.7377 and continuously decreased further. The figure 1 shows the loss vs epochs graph and figure 2 shows the accuracy vs epochs graph. The model is learnt till 2nd epoch and tried to decrease loss. This is due to the low validation data.

The model is then compiled with Adam optimizer, the loss function is set to binary_crossentropy as it is binary classification problem and for evaluation, metrics is set to accuracy. The model is trained then trained with 10% validation from training set. The augmented images are generated using the keras's Image Data Generator; these are directly trained without creating the images with the flow method of the object returned from the Image Data Generator.

```
model.compile(
        loss='binary_crossentropy',
        optimizer='adam',
        metrics=['accuracy']
)
datagen=tf.keras.preprocessing.image.ImageDataGenerator(
          horizontal_flip=True,
          validation_split=0.1,
          vertical_flip=True,
          fill_mode='constant', # constant makes the background as black
          rotation_range=30,
          rescale=1/255.  #rescale to make them in range(0,255) and 255. Is necessary to make
#them as decimal
)
image_generator=datagen.flow(x_train,y_train,batch_size=32,shuffle=True,subset='validation')
history =model.fit(image_generator,epochs=5)
```

Sample Images



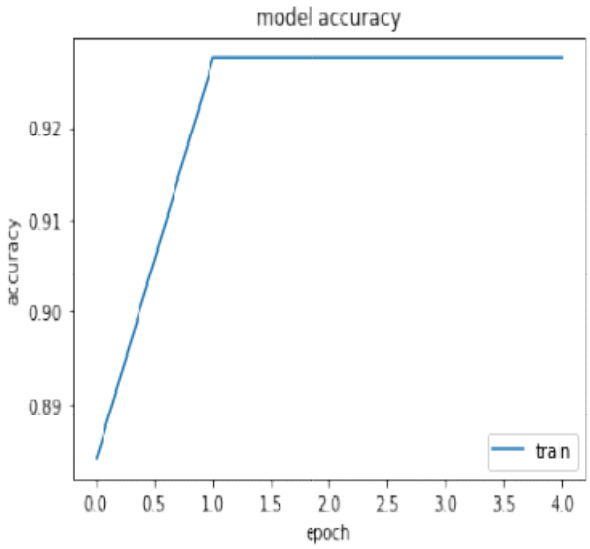


Figure 1: Model Loss.

Figure 2: Model Accuracy.

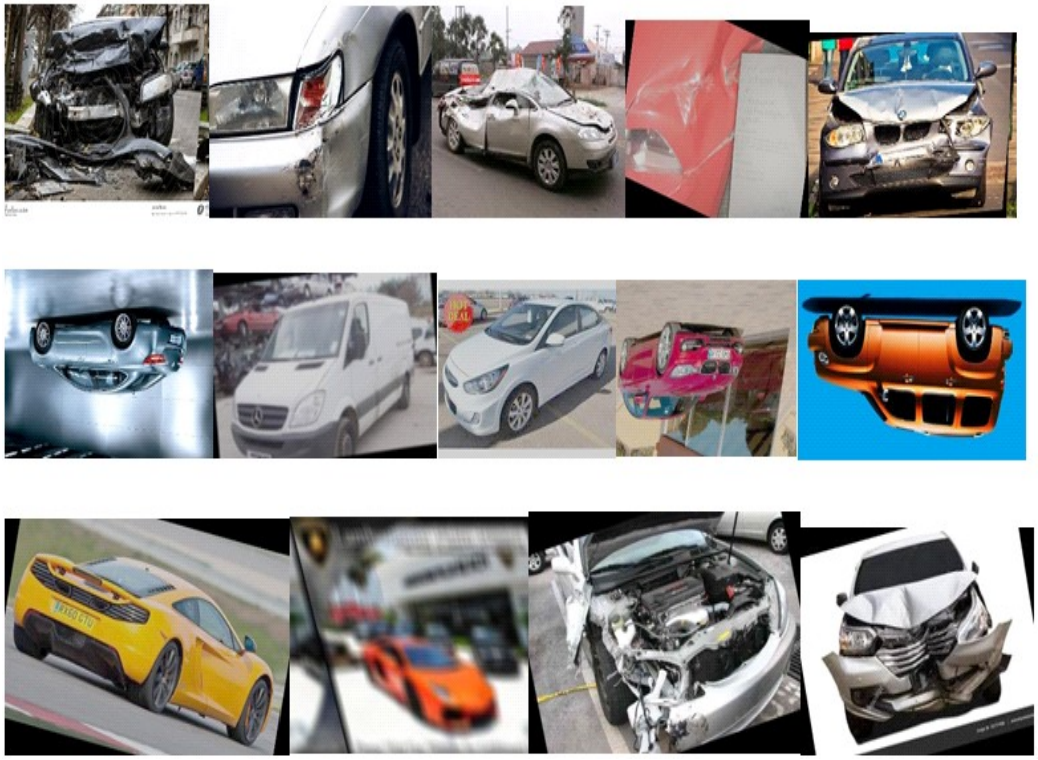


Figure 3: Random Images from Both Train and Test Dataset.s

Table 1: CNN Model Summary

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 398, 398, 16)      448
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 199, 199, 16)      0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 197, 197, 32)      4640
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 98, 98, 32)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 96, 96, 64)        18496
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 48, 48, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 46, 46, 256)       147712
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 23, 23, 256)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 21, 21, 512)       1180160
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 10, 10, 512)       0
_________________________________________________________________
conv2d_5 (Conv2D)            (None, 8, 8, 512)         2359808
_________________________________________________________________
max_pooling2d_5 (MaxPooling2 (None, 4, 4, 512)         0
_________________________________________________________________
conv2d_6 (Conv2D)            (None, 2, 2, 512)         2359808
_________________________________________________________________
max_pooling2d_6 (MaxPooling2 (None, 1, 1, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 1024)              525312
_________________________________________________________________
dropout (Dropout)            (None, 1024)              0
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800
_________________________________________________________________
dropout_1 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_2 (Dense)              (None, 512)               262656
_________________________________________________________________
dropout_2 (Dropout)          (None, 512)               0
_________________________________________________________________
dense_3 (Dense)              (None, 256)               131328
_________________________________________________________________
dropout_3 (Dropout)          (None, 256)               0
_________________________________________________________________
dense_4 (Dense)              (None, 128)               32896
_________________________________________________________________
dropout_4 (Dropout)          (None, 128)               0
_________________________________________________________________
dense_5 (Dense)              (None, 64)                8256
_________________________________________________________________
dense_6 (Dense)              (None, 1)                 65
=================================================================
Total params: 7,556,385
Trainable params: 7,556,385
Non-trainable: 0params
_________________________________________________________________
```

TRAINING THE ENSEMBLE MODEL

After the images are classified into respective categories, the data of the damaged cars is taken into pandasdata frame. This data consists of the car cost, Expiry data, min coverage, max coverage and the amount given if the car is damaged. As the amount is continuous data, Random Forest Regressor model of the ensemble is used for this regression. The number of estimators is set to 1000 and rest of params are set to default. And the regressor model summary is as follows

The advantage of the Random Forest Regressor algorithm is that the input featured data need not be scaled. The accuracy reached to 0.6706.

```
RandomForestRegressor (bootstrap=True, ccp_alpha=0.0, criterion='mse',
max_depth=None, max_features='auto', max_leaf_nodes=None,
max_samples=None, min_impurity_decrease=0.0,
min_impurity_split=None, min_samples_leaf=1,
min_samples_split=2, min_weight_fraction_leaf=0.0,
n_estimators=1000, n_jobs=None, oob_score=False,
random_state=None, verbose=0, warm_start=False)
```

CONCLUSIONS

In this paper, I proposed a deep learning based solution for the car damage classification and the Random Forest Regressor Model for the amount of insurance to be claimed if the car is classified as damaged. I experimented with multiple Regressor models such as SVM, Linear Regression models. I observed that the Random Forest Regressor modelhad reached the best accuracy among the major models. I also noted that the model is not much effective in damage classification as it reached 28.3% of test accuracy. It indicates that the features which are to be considered as damaged are not dominant compared to other features of the car model.

REFERENCES

1. *Connor Shorten and Taghi M Khoshgoftaar. 2019. A survey on image data augmentation for deep learning.Journal ofBig Data6, 1 (2019), 60*
2. *NajmeddineDhieb, Hakim Ghazzai, Hichem Besbes, and Yehia Massoud. 2019. Extreme Gradient Boosting MachineLearning Algorithm For Safe Auto Insurance Operations. In2019 IEEE International Conference of Vehicular Electronicsand Safety (ICVES). IEEE, 1–5.*
3. *Jeffrey de Deijn. 2018. Automatic Car Damage Recognition using Convolutional Neural Networks. (2018).*
4. *Kalpesh Patil, Mandar Kulkarni, Anand Sriraman, and Shirish Karande. 2017. Deep learning based car damageclassification. In2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 50–54.*
5. *JasonYosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. 2014. How transferable are features in deep neuralnetworks?. InAdvances in neural information processing systems. 3320–3328.*
6. *Mahavir Dwivedi, Malik Hashmat Shadab, SN Omkar, Edgar Bosco Monis, Bharat Khanna, and Satya Ranjan. [n.d.].Deep Learning Based Car Damage Classification and Detection. ([n. d.]*
7. *Kalpesh Patil Mandar Kulkarni Shirish Karande,Deep Learning Based Car Damage Classification,TCS Innovation Labs, Pune, India*
8. *An Introduction to Machine Learning with Python by Andreas C. Müller and Sarah Guido (O'Reilly).*
9. *https://www.tensorflow.org/api_docs/python/tf/keras/preprocessing/image/*
10. *https://keras.io/api/preprocessing/image/*
11. *https://www.hackerearth.com/challenges/competitive/hackerearth-machine-learning-challenge-vehicle-insurance-claim/*