Car Make and Model Recognition using Image Processing and Machine Learning

Hashir Yaqoob, Shaharyar Bhatti and Rana Raees Ahmed Khan.

***Abstract*—Advancement in automation, artificial intelligence and robotics has enabled development of intelligent transportation syustems for traffic monitoring and traffic surveillance systems. Car Make and Model recognition is an important part of such systems. Our project automatically and in real time, recognizes and classifies cars according to their make and model using machine learning, deep neural networks, and transfer learning to classify the cars, and image processing techniques to detect the cars in video feeds and in real time. Our project consists of most advanced systems including Convolutional Neural Networks, Bag of Features model and Support Vector Machines (SVM) classifiers. Moreover, this project has a numerous number of child projects, meaning that you can use any image, text or document database and train our system to do identification and classification according to the characterisitics defined by the user. Furthermore, the intention of this paper is to demonstrate our work and provide pathways to enable others to elaborate our work on this system.**

I. INTRODUCTION

CAR Make and Model Recognition, otherwise known as Make and Model Recognition (MMR) system is our project. This project incorporates image processing methods and machine learning techniques to recognize and classify vehicles according to their make and model.

The aim of this project is to develop a software system that takes an image or video feed as input; process it using image processing techniques, and then use machine learning to recognize and classify the make and model of the vehicle. For instance, when an image of a Honda Civic is loaded as an input to the system, it should classify the make as "*Honda"* and model as "*Civic",* with a satisfactory accuracy.

This research paper illustrates the methods and implementation of Car Make and Model Recognition CMMR system. Development platform was MATLAB, project included built-in MATLAB functions, and the User Interface (GUI) is implemented using MATLAB GUI developer "*GUIDE"*.

A number of image processing techniques are used for feature extraction, such as SURF (speeded up robust features), SIFT (scalar invariant feature transform), HOG (histogram of oriented gradients), cross correlation, peak correlation and edge detection. Various filters are also used to extract special information such as edges, intensities, color information and a lot more, for training a neural network.

The machine learning algorithms used in this project include Convolutional Neural Networks, Deep Neural Networks, Transfer Learning, Support Vector Machines (SVM) classifier, and a Bag-Of-Features model.

A. Overview

The principal objective of this project is to address the problem of recognition and classification of objects, according to certain desited characteristics. In our case, it is the classification of vehicles according to make and model. Our system incorporates latest techniques of machine learning for classification. A common approach for classification and recognition is given as:

- *Extracting Features from Images:* Feature extraction could be done by using image-processing techniques such as SURF or SIFT etc. The image is then represented using these features, which give a better description of the image, useful for classification.
- *Classification:* After feature extraction, the car iages are classified using machine learning technques. We have used deep neural networks using transfer learning and alsoo implemented Bag of Features model.

B. Bag of Features Method for MMR

Bag of features method uses SURF features to encode or represent the image. It then uses a machine learning classfifcation algorithm called the support vector machine SVM.

C. Convolutional Neural Network for MMR

Convolutional Neural Network or CNN is a type of neural network used widely for image classification systems. In this method, features are extracted using the convolutional layers and then classified using later layers of the network.

D. Data-sets used

The image data sets we used comprised of three image databases. First one was the COMPCARS[] Comprehensive Cars image database, another data base was collected from online internet resources, and the third data set was the one we collected ourselves, adding upto an approximate number of 10,000 images combined. The images were first split into categories according to the make and model of the vehicle, next the data set was split for training and testing images.

E. *Applications of VMMR systems*

The project has its applications in the following fields:

- Security and surveillance
- Toll plaza systems
- Car parking systems
- Traffic monitoring systems
- It can also enhance license plate recognition systems.

This project can also be used for data mining and data analysis purposes to observe related trends such as the color preferance, model preferance, etc.

II. Related Work

We studeid papers available on VMMR systems, we concentrated on those who used image processing and machine learning techniques for the implementation of VMMR systems. At the end we chose BOF model and CNN approaches for the implementation.

A. *Car Make Recognition Using Logo Extraction*

This approach uses the logo seen at the front or back of the vehicle and recognizes its make. The logo is first localized using SIFT key point and then is extracted and matched with the database. Mausam Jain and D. Tharun Kumar [1] used SIFT key points to detect the logo and extract it, then used template matching to match the logo with its respective class. The accuracy they reported is around 70%.

B. *Joint Car Make and Model Recognition*

In this approach, a particular car make and model is treated as a single category and recognized as such. Petrovic and Cootes [2] used front images of the car for make and model recognition. A region of interest is defined relative to license plate and features are extraced in it. Next nearest neighbout algorithm is used to classify.

Similarly Cheung and Ailee Chue [3] in the paper "Car make and model recognition" used two methods for feature extraction and matching. They also iused SIFT key points as features and descriptors for matching the query car image with its database images. The second approach used was Harris Corners for interest point detection and Fast Normalized Correlation for feature matching.

In the Bag-of-Features approach used by Abdul Jabbar Siddiqui, Abdelhamid Mammeri and Azzedine Boukerche in their paper "Real-Time Vehicle Make and Model Recognition based on a Bag-of-SURF-Features"[4], used the BOF model using SURF feature extractor. A vocabulary of words or features was formed by clustering SURF features of the images, next the classification was done using SVM classifier.

C. *VMMR using CNN (Deep Learning)*

Yiren Zhou et al. [5], in their paper "Image based Vehicle Analysis using Deep Neural Nets…" used a pre-trained Neural Network 'AlexNet". Although, the AlexNet originally is trained to recognize 1000 categories of different objects, the authors used it to distinguish between various categories of cars.

Two approaches were used, the first one was to extract feauters from higher layers of the CNN and then classify using SVM classifiers. The second approach was to fine tune the neural network to be used for VMMR system.

Another paper named "View Independent Car Make and Model and Color Recognition" by Afshin Dehgan and Syed Zain Masood [6], the authors trained a deep neural network for VMMR using a very large image data set consisting of millions of images. Their system is known as "Sighthound".

III. Proposed Methods

We implemented our MMR system using two approaches. The first approach used is the Bag-Of-Features model (BOF). In BOF, the SURF method was used for feature extraction and for classification, Multi-class Support Vector Machine (SVM) classifier was used. The proposed workflow is as follows:

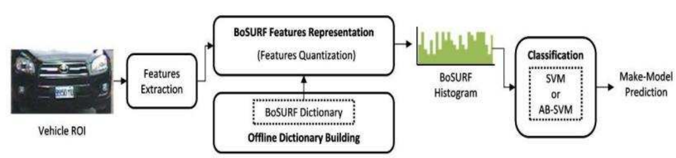

Figure 1: Bag of Feature model work flow

The second approach we used was the CNN neural network, which is trained for feature extraction and classification. We also used pre-trained neural network known as AlexNet, and used Transfer Learning to train AlexNet on our data set to classify vehicles in VMMR system.

IV. Procedure and Implementation

A. Data collection

First step to the VMMR system is the collection of data set large enough that your systems are trained on enough images to present a satisfactory accuracy. Our data set for vehicle images consisted of three data sets as mentioned earlier, the COMPCARS dataset, Internet Images, and personally collected data set, with raw images numbering over 10,000 images.

B. BOF Implementation

The BOF model is fairly and old model, and has been used widely for document classification and text recognition and classification, thus it has some constraints regarding the limit of size and capacity. BOF model is considered robust on the other hand due to its flexibility on the nature and orientation of the images. There is no size constraint and the feautre extraction process is fairly less time consuming than the Neural network training time.

Due to the size limit constraints and development machines constraints, we were forced to use a relatively small data set to train BOF model. We used approximately 4000 vehicle images, divided into 38 categories (models of the cars), and trained the BOF model on this data set. The method of training BOF is illustrated below:

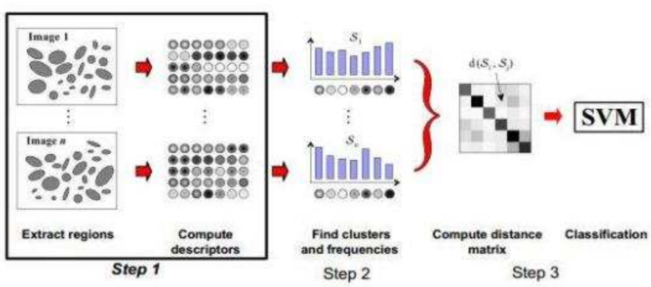

Figure 2: training a BOF model

- *Feature Extraction:* we extract special features from all the training images. Feature extraction is first achieved by locating SURF points and the defining SURF descriptors, we extract these points. An image with detected SURF points and their descriptors is shown below:

Figure 3: SURF features and SURF descriptors

- *Clustering:* using the SURF points, we create a vocabulary by clustering, it is the method to collect similar data points and collecting them in groups using via K-means clustering which uses the "*Euclidean Distance"* algorithm. "*K*" is the number of cluster centers (or words), which are the vocabulary of the BOF model. A cluster model is shown below with K=5:

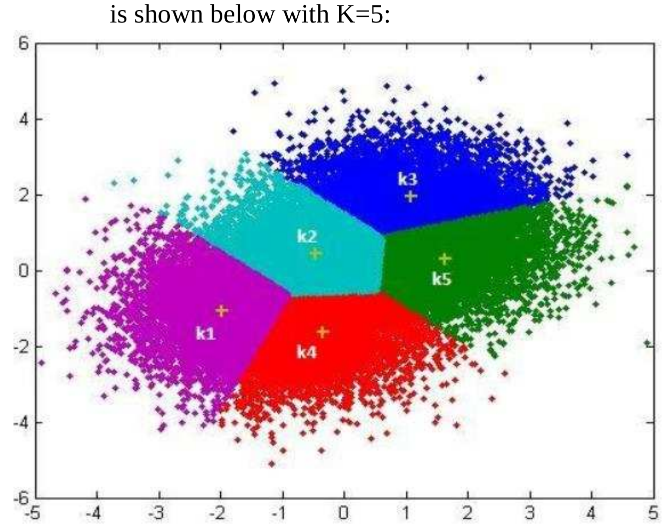

Figure 4: k-Means clustering for k=5

- *Encoding:* next we encoded all the training images using the clusters made in step 2 and we represent each image according to the vocabulary of the BOF model. Encoding procedure is shown below:

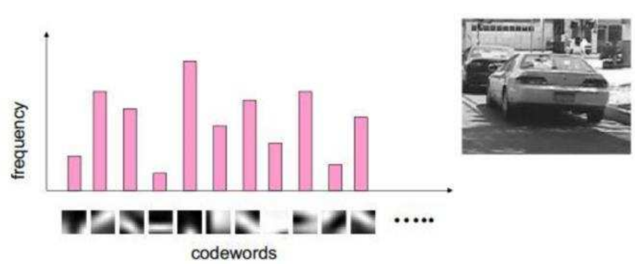

Figure 5: Encoding images with BOF Vocabulary

- *Classification:* we then use a multi class SVM classifier to classify the new test images, according to their make and model.

C. CNN using Transfer Learning

We used Transfer Learning to re-train a pre-trained neural network known as the AlexNet neural network. This network was trained on millions of images with more than a 1000 categories. To use AlexNet we first had to arrange for a large enough vehicle image data set to ensure the proper advantage of AlexNet in our system.

AlexNet is trained on images which are resized to [227 x 227] pixel size. Smaller image size means less data to extract out from the images, thus there was a need to train it on a large data set to ensure a satisfactory accuracy.

CNNs are better at image classification because it has additional layers besides the basic architecture layers of CNN, it also consists of nearly a 100 different types of filters, all of them extract unique information from the image, for example it has filters to extract noise, edges, colours, hidden details etc. A collage of all these filters outputs are shown below:

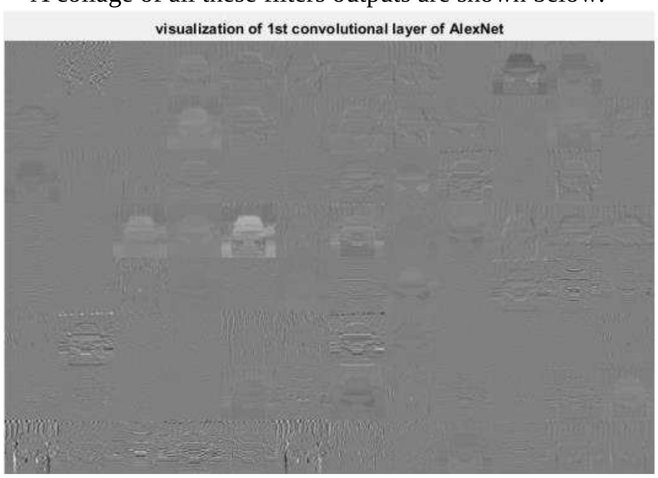

Figure 6: Output of 96 filters of 1st convolutional layer

Neural networks take a lot of time for training on a large data set. We used a GPU capable computer to train the CNN.

The method we used to train the CNN using transfer learning is stated below:

- *Feature Extraction:* we first extract features using the 96 image filters present in the first convolutional layer of AlexNet. Because we use these layers repeatedly and with different configuration everytime, the earlier layers extract low level features like edges and curves, while later layers extract high level information.
- *Classification:* the classification in CNN is achieved in the "fully connected layer" of the CNN, which is the last layer in the network.

The architecture of AlexNet is given as follows:

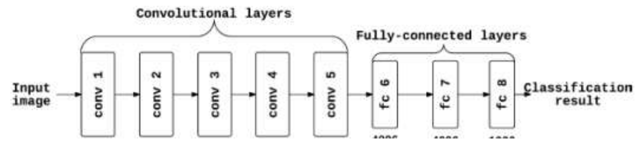

Figure 7: AlexNet layer architecture

The input image passes through 5 convolutional layers and 3 fully-connected layers before the classification results are shown.

D. Intensive Computational Requirements for CNN training and testing

As mentioned before, the training of the CNN requires a lot of computational resources and computing power to be trained on a very large scale data set. Using a computer with a nVidia Graphic Processor with CUDA driver capabilites was used to train and test the neural network on images and videos as well as live feed in real time.

Following is the training stage of the CNN AlexNet, which illustrates the time consumption of the training process:

Training on single CPU.
Initializing image normalization.

Epoch	Iteration	Time Elapsed (seconds)	Mini-batch Loss	Mini-batch Accuracy	Base Learning Rate
1	1	901.74	4.0271	1.56%	0.0010
2	50	45677.68	2.2180	29.69%	0.0010
3	100	94450.91	1.2687	60.94%	0.0010
4	150	140838.19	0.6772	82.81%	0.0010
5	200	188450.77	0.6051	79.69%	0.0010
7	250	237713.47	0.5628	81.25%	0.0010

Figure 8: AlexNet training process

The steps involved in the training of CNN are as follows:

- Data set collection: collection of over 10,000 images.
- *Resize, organize into classes and split for training and testing:* the data set is resized to the image layer input size for AlexNet, which is 227 x 227 pixels.
- *Modify AlexNet layers to use with our dataset:* modify the fully connected layers to adjust for the number of classes for our data set.
- *Retrain network:* retrain the network on our data set.
- *Classification:* after training is done, we pass a test image to the network and obtain the output of the classification result.\

The modifications applied to the network architecture of AlexNet are as follows (we modified layer 23 and layer 25 to do transfer learning training of the AlexNet:

1	'data'	Image Input	227x227x3 images with 'zerocenter' normalization
2	'conv1'	Convolution	96 11x11x3 convolutions with stride [4 4] and paddin
3	'relu1'	ReLU	ReLU
4	'norm1'	Cross Channel Normalization	cross channel normalization with 5 channels per eleme
5	'pool1'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
6	'conv2'	Convolution	256 5x5x48 convolutions with stride [1 1] and paddin
7	'relu2'	ReLU	ReLU
8	'norm2'	Cross Channel Normalization	cross channel normalization with 5 channels per eleme
9	'pool2'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
10	'conv3'	Convolution	384 3x3x256 convolutions with stride [1 1] and paddi
11	'relu3'	ReLU	ReLU
12	'conv4'	Convolution	384 3x3x192 convolutions with stride [1 1] and paddi
13	'relu4'	ReLU	ReLU
14	'conv5'	Convolution	256 3x3x192 convolutions with stride [1 1] and paddi
15	'relu5'	ReLU	ReLU
16	'pool5'	Max Pooling	3x3 max pooling with stride [2 2] and padding [0 0]
17	'fc6'	Fully Connected	4096 fully connected layer
18	'relu6'	ReLU	ReLU
19	'drop6'	Dropout	50% dropout
20	'fc7'	Fully Connected	4096 fully connected layer
21	'relu7'	ReLU	ReLU
22	'drop7'	Dropout	50% dropout
23	'fc8'	Fully Connected	1000 fully connected layer
24	'prob'	Softmax	softmax
25	'output'	Classification Output	crossentropyex with 'tench', 'goldfish', and 998 othe

Figure 9: Modified layers of AlexNet

E. *Classification using SVM Classifier*

The classification algorithm used in both BOF model and CNN network is the Support Vector Machine classifier.

We used multi class support vector machine classifiers to cover all the classes and categories of image data base. SVM classifer is a robust classification method which produces near to accurate results.

A simple SVM classifier is a binary classifier and works on only two classes separated by a separating hyperplane, which defines the boundary between the two classes, as shown below:

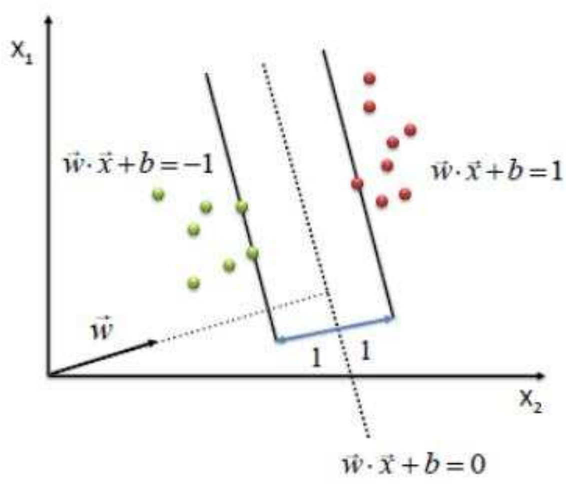

Figure 10: separating hyperplane

The optimal hyperplane is calculated using the following equations to best separate the data points of the classes:

$$\frac{w}{\|w\|}\cdot(x_2-x_1)=\text{width}=\frac{2}{\|w\|}$$

$$w\cdot x_2+b=1$$
$$w\cdot x_1+b=-1$$
$$w\cdot x_2+b-w\cdot x_1-b=1-(-1)$$
$$w\cdot x_2-w\cdot x_1=2$$
$$\frac{w}{\|w\|}(x_2-x_1)=\frac{2}{\|w\|}$$

Equation 1: Equations to find the optimum separating hyperplane between classes

A multi class classifier uses different data quantization methods and multiple classifiers and hyperplanes for multiple classes. The hyperplanes in multi-class classifeirs are not linear. They use kernels to linearize non-linear data and apply non-linear hyperplanes to distinguish between classes, such a hyperplane is shown below, which linearize data and hyperplane to make it a binary classifer to better classify test images:

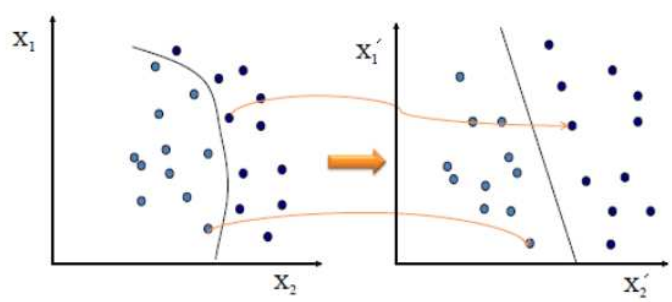

Figure 11: linearization of data and hyperplane

As mentioned earlier, we use kernels to linearize the data, dollowing are some of the kernels which are applied on the multi class SVM classifier:

$$K(\mathbf{X}_i,\mathbf{X}_j)=\begin{cases}\mathbf{X}_i\cdot\mathbf{X}_j & \text{Linear}\\ (\gamma\mathbf{X}_i\cdot\mathbf{X}_j+\mathrm{C})^d & \text{Polynomial}\\ \exp(-\gamma\,|\mathbf{X}_i-\mathbf{X}_j|^2) & \text{RBF}\\ \tanh(\gamma\mathbf{X}_i\cdot\mathbf{X}_j+\mathrm{C}) & \text{Sigmoid}\end{cases}$$

Equation 2: SVM Kernel functions

V. GRAPHICAL USER INTERFACE

Our system has an easy to understand and use GUI, which is specifically designed to display image classification output, video classification output, and classification in real time.

It has the option to ask the user to load an image, video or start a live feed, and select what classification methods the user wants to use, and then start the classification process by clicking the Classify button.

Furthermore, we also have incorporated the training option in the GUI, where the user can define the data set location and start the training of selected classification models by the user.

Moreover, our GUI also has the option and functions to extract logo of the car, and the license plate of the car, which further can be read using OCR-optical character recognition (part of future development). GUI is displayed below:

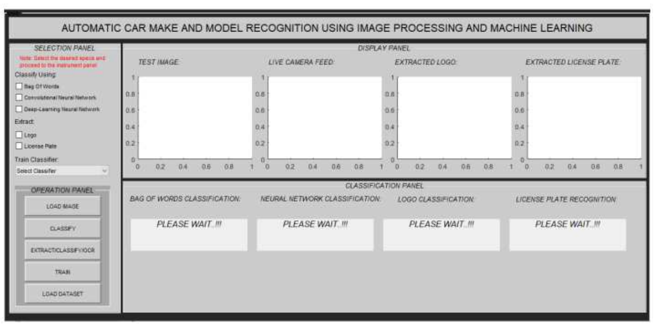

Figure 12: Project GUI

VI. EXPERIMENTAL RESULTS

A. Bag-Of-Features

BOF model was trained on around 4000 images which contributed to make a total of 49 classes, or categories of cars.

The confusion matrix created after the training of the BOF model is shown below:

* The confusion matrix for this test set is:

KNOWN	CHEVORLET MATIZ	DAIHATSU COURE	DAIHATSU MIRA	HONDA ACCORD	HONDA CITY	HONDA CIVIC
CHEVORLET MATIZ	0.00	0.00	0.00	0.00	0.00	0.00
DAIHATSU COURE	0.00	0.36	0.00	0.00	0.00	0.00
DAIHATSU MIRA	0.00	0.00	0.00	0.00	0.00	0.00
HONDA ACCORD	0.00	0.10	0.00	0.00	0.00	0.00
HONDA CITY	0.00	0.16	0.00	0.00	0.00	0.00
HONDA CIVIC	0.00	0.13	0.00	0.00	0.00	0.00
HONDA CRZ	0.00	0.13	0.00	0.00	0.00	0.00
HONDA FIT	0.00	0.09	0.00	0.00	0.00	0.00
HONDA INSIGHT	0.00	0.06	0.00	0.00	0.00	0.00
HONDA VEZEL	0.00	0.06	0.00	0.00	0.00	0.00
HYUNDAI SANTRO	0.00	0.50	0.00	0.00	0.00	0.00
MAZDA CORAL	0.00	0.00	0.00	0.00	0.00	0.00
NISSAN SUNNY	0.00	0.00	0.00	0.00	0.00	0.00
SUZUKI ALTO	0.00	0.07	0.00	0.00	0.00	0.00
SUZUKI ALTO JAPANESE	0.00	0.08	0.00	0.00	0.00	0.00
SUZUKI BALENO	0.00	0.06	0.00	0.00	0.00	0.00
SUZUKI BOLAN	0.00	0.09	0.00	0.00	0.00	0.00

Figure 13: BOF confusion matrix

The visual vocabulary created after the clustering in the BOF model is shown below, we set the vocabulary size to 600 words, and this is the total word count against the occourence:

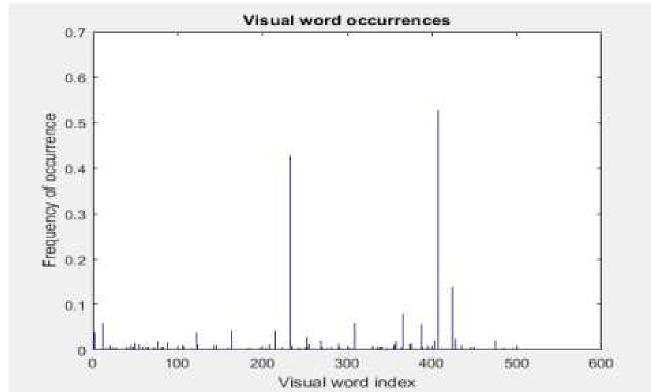

Figure 14: Visual vocabulary histogram

To test the accuracy of the BOF model, we test it by passing a test image of a “HONDA CIVIC” car to the BOF model using our GUI, which yields the following results:

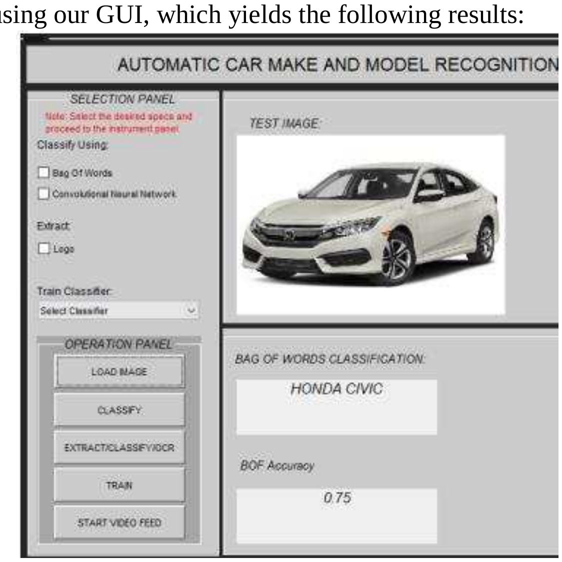

Figure 15: BOF classification output

The BOF model presented a satisfactory accuracy, ranging from 70% to 78% on average, evaluated by our accuracy formula.

B. CNN and Transfer Learning results

As expected, Neural networks are supposed to produce more accurate results and robust outputs for classification.

We trained the AlexNet on 50 classes consisting of a total of 10,000 images, which consisted of the COMPCARS data set, and our own collected images from the premises of the university parking.

Following is the screenshot of the training stage of the CNN on nVidia CUDA capable computer:

```
|          11 |          950 |     10500.67 |       0.0631 |       98.00% |       0.0001 |
|          12 |         1000 |     11027.15 |       0.0258 |       99.00% |       0.0001 |
|          13 |         1050 |     11560.32 |       0.0215 |      100.00% |       0.0001 |
|          13 |         1100 |     12091.58 |       0.0290 |      100.00% |       0.0001 |
|          14 |         1150 |     12619.01 |       0.0798 |       97.00% |       0.0001 |
|          14 |         1200 |     13151.69 |       0.0881 |       96.00% |       0.0001 |
|          15 |         1250 |     13685.95 |       0.0219 |      100.00% |       0.0001 |
|          15 |         1300 |     14209.47 |       0.0850 |       96.00% |       0.0001 |
|          16 |         1350 |     14734.82 |       0.0831 |       99.00% |       0.0001 |
|          17 |         1400 |     15067.21 |       0.1088 |       96.00% |     1.00e-05 |
|          17 |         1450 |     15371.76 |       0.0620 |       98.00% |     1.00e-05 |
|          18 |         1500 |     15678.88 |       0.0120 |      100.00% |     1.00e-05 |
|          18 |         1550 |     15990.78 |       0.0098 |      100.00% |     1.00e-05 |
|          19 |         1600 |     16305.51 |       0.0514 |       98.00% |     1.00e-05 |
|          19 |         1650 |     16619.96 |       0.0650 |       98.00% |     1.00e-05 |
|          20 |         1700 |     16940.75 |       0.0642 |       97.00% |     1.00e-05 |
|          20 |         1740 |     17194.09 |       0.0529 |       99.00% |     1.00e-05 |
|=========================================================================================|

accuracy =

    0.9362

Parallel pool using the 'local' profile is shutting down.
```

Figure 16: CNN training process, average accuracy

As shown above, the accuracy of the evaluation set and the training set is about 93%.

Next, we passed a series of images to the CNN, and observed above 90% accuracy in classification. Following is a snapshot of an image of "HONDA CIVIC", passed as an input test image and the ouput is shown below, all processed in GUI:

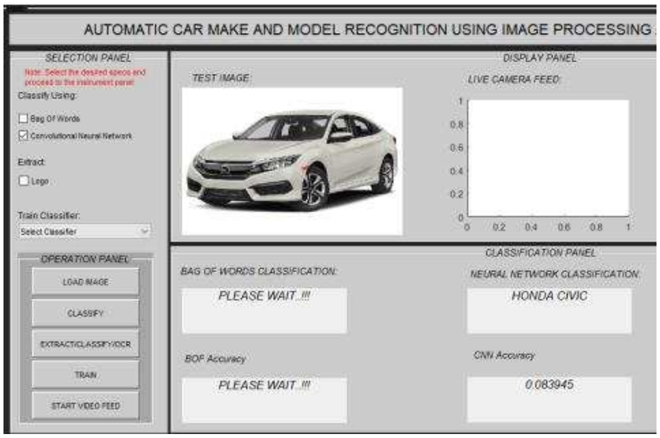

Figure 17: CNN classification output

C. LOGO Detection and classification

We also implemented template matching using peak correlation method, to localize and extract the logo from front and back of the car. For this, we created a template data set of logo images, as shown below:

Figure 18: car logo image templates

These templates are used to classify the extracted logos from the vehicle images, this function was implemented as an extra and the accuracy of logo localization was not up to the mark and needs more work and fine-tuning. However, a true localization and classification match is shown below which was carried out in the GUI:

Figure 19: LOGO detection and classification output

D. Evaluation Metric

This section emphasizes on the outcomes of our project, and the accuracy it yielded for both models. The accuracy evaluation formula we use is as follows:

$$Accuracy = \frac{true\ matches}{total\ test\ images} \times 100\%$$

Where *true matches* is the number of times the classifer truly classifed a vehicle image.

VII. CONCLUSION

VMMR systems are useful in traffic monitoring and surveillance, toll plaza systems and parking systems. Currently work is being carried out by researchers to develop accurate and robust techniques for car make and model recognition. Our system includes advanced machine learning algorithms such as CNN and deep learning and state of the art AlexNet neural network.

However the BOF model is ruled out as less effective and accurate classification method as compared to the neural networks efficiency and accuracy. We obtained above 90% accuracy using neural networks as compared to the 75% average accuracy of BOF model.

The accuracy can be further improved by using a larger data sets having images numbering in millions, to ensure a proper learning of the classes, which contains images taken in every kind of different situations, so that the learning is universal and more effective.

Furthermore, this system has a numerous number od child projects in the domain of object and pattern recognition and classification.

Moreover, this system could be used for data mining and trend analysis, for example, we can use the calssification in real time and observe the trends of most preferred colour of vehicles, most preferred models of the cars, in any region. This analysis can be very helpful for vehicle manufacturing companies to make their business more profitable.

Appendix

- **SURF:** Speeded up robust features.
- **SIFT:** Scalar Invariant Feature Transform.
- **HOG:** Histogram of oriented gradients.
- **Template Matching:** Match an object in an image using the templates in the template database.
- **Peak Correlation:** Match two images using peaks created by taking the discrete fourier transform of both images.
- **KNN:** k nearest neighbour algorithm.
- **BOF:** Bag of Features
- **CNN:** Convolutional Neural Network.
- **SVM:** Support Vector Machine.
- **Transfer Learning:** Method of re-training a pre-trained neural network using its layers weights and your own data set.
- **VMMR:** Vehicle Make and Model Recognition.
- **MMR:** Make and Model Recognition.

Acknowledgment

This project would not have been possible without the support, supervision and sincere guidance of our supervisor, Ms. Sumayya Haroon, Assistant Professor, faculty of Electrical Engineering department of COMSATS Institute of Information Technology, Islamabad. She guided us effectively to make this project reach its completion.

We are thankful to our seniors, and faculty members, who helped us out in every difficulty and obstacle that occoured during the development of this project.

We would very humbly pay our gratitude to our family members, for their support and prayers.

In the end, we would like to thank our group members for putting in their sincere efforts and knowledge.

This work is dedicated to our respected supervisor, to the university and our friends and family.

References

[1] Mausam Jain, D. Tharun Kumar, "*Car Make and Model Recognition",* IIT Hyderabad, ODF, Yeddumailaram – 502205, 2015.

[2] V. S. Petrovic and T. F. Cootes, "*Analysis of features for rigid structure vehicle type recognition",* in Proc. British Machine Vision Conference (BMVC'04), pp. 587-596, Kingstone UK, September 2004.

[3] Sparta Cheung, Alice Chu, *"Make and Model Recognition of Cars",* CSE 190A Projects in Vision and Learning, Final Report, 2008.

[4] Abdul Jabbar Siddiqui et. Al *"Real-Time Vehicle Make and Model Recognition Based on a Bag of SURF Features", IEEE Transactions on Intelligent Transportation Systems, VOL. 17, NO. 11, November, 2016.*

[5] *Yiren Zhou et. Al. Ímage-Base Vehicle Analysis using Deep Neural Network: A Systematic Study", arXiv:1601.01145v2[cs.CV], August, 2016.*

[6] Afshin Dehghan et. Al. "View Independent Vehicle Make, Model and Color Recognition using Convolutional Neural Network", Computer Vision Lab, Sighthound Inc., Winter Park, FL. arXiv:1702.01721v1[cs.CV], 6 Feb 2017.

[7] A Krizhevsky, et. Al. "*Imagenet Classification with Deep Convolutional Neural Networks:*, Advances in Neural Information Processing Systems, 2012.

[8] Derrick Liu, Yushi Wang, *"Image Classification of Vehicle Make and Model using Convolutional Neural Networks and Transfer Learning",* Stanford Uni.

[9] Dr. Kazi A. Kalpoma et. Al. "*Logo Recognition using SURF Features and kNN Search Tree",* International Journal of Scientific and Engineering Research, Volume 6, Issue 9, September-2015. ISSN 2229-5518.

[10] pp. 876—880. Available: http://www.halcyon.com/pub/journals/21ps03-vidmar