Empirical Evaluation of Convolutional Neural Networks Prediction Time in Classifying German Traffic Signs

Joshua Fulco[1], Akanksha Devkar[2], Aravind Krishnan[1], Gregory Slavin[1] and Carlos Morato[1,3]
[1]Robotics Engineering Department, Worcester Polytechnic Institute, Worcester, MA 01609, U.S.A.
[2]Electrical and Computer Engineering Department, Worcester Polytechnic Institute, Worcester, MA 01609, U.S.A.
[3]Department of Mechatronics and Sensors, US Corporate Research Center, ABB Inc., CT 06002, U.S.A.

Keywords: Convolutional Neural Networks, Classification Time, Traffic Sign Classification.

Abstract: This paper discusses the use of Deep Learning and neural networks to identify images which contain road signs to aid in the navigation of autonomous vehicles. Images of 32x32 pixels and 128x128 pixels of the GTSRB dataset were used in training the existing neural network models as well as our novel models. Existing neural network models mentioned in the literature study validate that very high accuracies in image classification are already achieved. Different neural network model architectures were also reviewed to determine which architecture produced the highest accuracy within the most efficient time. Modifications to these architectures were made to produce valid results with a reduced image identification time. Our results of classifying a traffic sign image of 32x32 pixels in 0.6ms is very reliable for real time output. By looking at the image identification times for a 32x32 pixel image and a 128x128 pixel image we observed that size of the image is not the main factor in the increase of the prediction time.

1 INTRODUCTION

Deep learning is a form of machine learning which has recently re-emerged as a topic of great interest and advanced research around the world after a near decade fallout with researchers. The basic concept of machine learning also known as representative learning can be best described in an example where a machine (or system of machine processors) takes in raw data such as an audio clip or photograph and then uses extractor tools to discover patterns within the raw data. Features identified or discovered within the raw data by the machine are passed between the multiple layers or filters which are used to sort out features such that they are saved to memory. In theory, with a varying degree of accuracy, a system exercising the methods of machine learning should be able to recognize and/or identify identical or similar data to what was presented in a previous training. In many ways machine systems which are organized in a manner similar to how it is believed the human brain is organized to intake information. It is believed that the neurons in the human brain accept data or information through the dendrites. The nucleus sorts or identifies this information and then transmits other portions of the information through impulses sent out to other neurons through axons. With machine learning filters or activation functions which identify features in raw data are often organized in layers. These filters usually are trained to identify one particular feature, and pass other information between activation functions in the layer or transmit binary signals confirming or eliminating the presence of a feature on to the next layer or back to a previous layer. Methods with at least three layers of activation function filters are referred to Deep learning models.

Deep learning methods are making great strides recently in the field of autonomous automotive operations. Recent news has been made with record breaking deliveries for the Anheuser-Busch Corporation using autonomous vehicles (Menge, 2016). Deep learning methods of machine learning have also been used to aid in navigation of sedan vehicles on roads with varying terrain. As shown in source (Bojarski et al., 2017) a series continuously taken photographs of a vehicles path are used to determined wheel orientation. The photograph data is used to train a navigation model which can correlate to wheel position. In addition visual navigation from images along a path are hoped to mature as autonomous vehicle development continues. Street sign recognition through Deep Learning can also be used for vehicle velocity and navigation control, such that an autonomous vehicle will stop at a stop sign, yield to a pedestrian

Fulco, J., Devkar, A., Krishnan, A., Slavin, G. and Morato, C.
Empirical Evaluation of Convolutional Neural Networks Prediction Time in Classifying German Traffic Signs.
DOI: 10.5220/0006307402600267
In *Proceedings of the 3rd International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2017)*, pages 260-267
ISBN: 978-989-758-242-4

or other vehicles, as well as identify directional route signs and road placards on a highway. It is the subject of this paper to exploit the development of a Convolutional Neural Network Deep Learning model that could possibly be used to identify road signs for use in the navigation of autonomous vehicles.

2 OBJECTIVE

It's about time that we have autonomous cars taking on the roads and the technology today has certainly evolved to the extent of realising robotic cars unleash the Earth's road networks. Lux Research calculates that the revenue opportunity from Advanced Driver Assistance Systems (ADAS) features will grow from $2.4 billion today to $102 billion in 2030. For such an autonomous car to perform efficiently, it needs to perceive the road visually. These cars need to have a sharp vision that detect all types of road conditions - traffic lights, road signs and take corrective measures accordingly. Also, we have recently seen that Tesla's semi-autonomous cars are performing great with their Autopilot mode and this assistance relieves drivers stress.

Semi-autonomous cars are excelling in keeping the driver alert by suggesting corrective actions and making sure the driver doesn't miss any on-road information. We focus on reducing the space and time complexity of classification of German road signs using one of the Deep Learning techniques - Convolutional Neural Networks(CNN) for the purpose of driver assistance and making an autonomous car more efficient.

Staggering results are achieved in the offline classification reaching 99.65% (Mao et al., 2016) accuracy, and when these results are compared with the real-time classification results, they are low, particularly due to the computational complexity in time and hardware requirements. It is varying anywhere between 80% to 93% (Chen et al., 2012). After localization and detection, the detected image of a road sign can have problems like

Scaling - leads to image quality reduction making it hard to recognize/discern.

Rotation - a damaged/ rotated sign may be interpreted as a different sign altogether, for instance when a left arrow somehow rotated 90 degrees clockwise can simply mean that it is a straight arrow and not make any sense. It is much harder to understand if that rotated arrow was pointing right or left originally.

Projection distortion - a distorted sign board can lead to wrong classification or no classification, for instance the sign which indicates the merging of left/right lane can be classified completely wrong due to the distortion in the top portion of the sign.

There are few other problems while detecting the road signs during the video recording phase such as road signs may be subjected to dust deposition, or may have faded with time, or be victim to graffiti, stickers subjecting to occlusion and even getting damaged in some cases. All these lead to increase in the error rates of detection, prediction and classification process. So we believe, Deep Learning with CNN on a good hardware with reduced algorithmic complexity can ease the process and improve accuracy for the real-time output. Another aspect to consider is the selection of a dataset or datasets. Most of the Deep Learning road sign identification experiments we found are based on a German set of road signs known as the German Traffic Sign Recognition Benchmark (GTSRB) (Igel,) which has become somewhat of a standard bearer.

As mentioned above, real time road sign detection is an important feature on ADAS, but in order to be truly effective it needs to be both, accurate and efficient. Specifically for road sign classification, Deep Learning models based on Neural Networks are known to be very accurate, achieving rates above human detection abilities (according to our preliminary research) but not necessarily fast enough (yet) to be used in real time systems where (embedded) processing power is limited. On the other side, more conventional computer vision techniques can be used to achieve real time processing speed, but fell short in accuracy when compared to Deep Learning methods. Our objective is to empirically evaluate different Convolutional Neural Networks with multiple variations of the hyper-parameters and make it classify the road signs more efficiently for the real time output.

3 REVIEW OF LITERATURE

Dan Cires et al. avoided the cumbersome computation of handcrafted features and achieved a 98.73% recognition rate with the best of their CNNs for the German road sign recognition benchmark. They further improved these results up to 99.15% recognition rate using a committee of a CNN and a Multi Layer Perceptron(MLP) where the CNNs are trained on raw pixel intensities, although error rates of the best MLPs are 3-4% above those of the best CNNs, a committee consistently outperforms the individual classifiers. (Ciresan et al., 2011)

Jack Greenhalgh et al. proposed a novel real-time system for the automatic detection and recognition of traffic symbols. Candidate regions are detected as

MSERs. This detection method is significantly insensitive to variations in illumination and lighting conditions. Traffic symbols are recognized using HOG features and a cascade of linear SVM classifiers. A method for the synthetic generation of training data was proposed, which allows large datasets to be generated from template images, removing the need for hand labeled datasets. Their system can identify signs from the whole range of ideographic road signs currently in use in the U.K. which form the basis of our training data. This system retains a high accuracy at a variety of vehicle speeds and achieved recognition accuracy of 89.2% for white road signs and 92.1% for color signs (Greenhalgh, 2012)

Tam T. Le et al. presented a real-time processing method of traffic-sign detection to apply in autonomous driving system. Their proposed method utilized linear SVM to classify color by a low complexity (average 23 ms per frame). After that shape matching was applied to eliminate positive errors. They achieved 92.91 percent of detection accuracy and it was applied on real-time autonomous driving system with the processing speed of 20fps, where the maximum speed of car was limited at 30 km per hour.(Le et al., 2010)

Yujun Zeng et al. proposed a novel architecture for road sign recognition, where CNN acts as a feature extractor and Extreme Learning Machines(ELM) trained on CNN-learnt features as the classifier, so that the discriminative Deep Convolutional features could match well with the generalization performance of ELM classifier, leading to a satisfactory recognition accuracy without using more complex CNNs, ensemble features, or data augmentation. In contrast with state-of-art methods, the proposed method could achieve competitive results (99.40%, without any data augmentation and preprocessing like contrast normalization) with a much simpler architecture that relieves the time-consuming training procedure a lot. Yet, the fact that most errors are mainly due to motion blur implies that the performance may be further improved if the CNN is equipped with some layers that could learn blur-invariant features.(Zeng et al., 2015)

Pierre Sermanet et al. presented a Convolutional Network architecture with state-of-art results on the GTSRB road sign dataset implemented with the EBLearn open-source library. During phase I of the GTSRB competition, this architecture reached 98.97% accuracy using 32x32 colored data while the top score was obtained by the IDSIA team with 98.98%. The first 13 top scores were obtained with ConvNet architectures, 5 of which were above human performance (98.81%).

Subsequently to this first phase, they established a new record of 99.17% accuracy by increasing networks capacity and depth and ignoring color information. This contradicted prior results with other methods suggesting that colorless recognition, while effective, was less accurate. They also demonstrated the benefits of multi-scale features in multiple experiments. Additionally, they report very competitive results (97.33%) using random features.(Sermanet, 2011)

Junqi Jin et.al designed a TSR system using a CNN, which is a special kind of deep neural network. The model had the ability to learn both features and classifiers. The learned features detect specific local patterns that are better than hand-coded features. They have proposed an Hinge Loss Stochastic Gradient Descent (HLSGD) method to train CNNs. After they tested their algorithm on the GTSRB and compared results with other competitors, their experiments showed that HLSGD gave faster and more stable convergence and a state-of-art recognition rate of 99.65%(Jin et al., 2014)

4 DATASET

With any Deep Learning model a compiled set of raw data is necessary to train, test, and validate a model. The more samples within a dataset, the more features a model can recognize during training. (Provided the samples belong to a small number of classes.) The more features recognized by the model, the better the chance of high accuracy sample recognition during validation and actual use. Due to access and time the availability of compiled datasets is limited. It is not uncommon for researchers to modify existing datasets to achieve a large number of samples. In the case of an image dataset it is not uncommon for images to be cropped, contracted to a smaller size or specific area, or reduced to gray scale if in color.

Traditionally half of a dataset is used for the purposes of training. The remaining half is again split into quarters. One quarter is used as part of a test set, which may be mixed with the training set to determine system accuracy following training. If the model does not recognize the test set with the accuracy desired additional exposure to the training set may occur until the desired accuracy is achieved. The last quarter of a dataset, is often referred to as the validation set. This set of data is not used until training is complete. The validation set is exposed to the model for identification to serve as a final proof of model accuracy.

For this project, to train a Deep Learning model that can identify road signs, an image dataset was required. Two image datasets were found to be in existences that are used by commercial groups in pursuit of autonomous vehicle navigation. The first dataset is from the Laboratory for Intelligent and Safe Automobiles (LISA) at the University of San Diego California, a group funded by a grant from several auto manufacturers and US government agencies. Dataset (Trivedi,) is derived from several videos which were recorded from car dashboard cameras. Some sample datasets were parsed into individual still images. As a result 47 types of US road signs are recorded in images with the natural scenic background. The images range in size from 640x480 to 1024x522 pixels. 6610 images are in the set with 7855 annotations that identify the content of the images. Annotation content includes not only road sign descriptions but descriptions of other items seen in the frame.

The second dataset is the German Traffic Sign Recognition Benchmark (GTSRB) dataset, containing 40 classes of German traffic signs, and 50,000 images ranging in size from 15x15 to 250x250 pixels.(Stallkamp et al., 2012) The dataset was first used as a benchmark for the 2011 IJCNN computer vision competition, and is provided by the real time computer vision research group. The dataset has both single and multi-annotated images of road signs only. The road signs are photographed in natural environments, but cropped so show only the signs. Natural environments do include various lighting and weather conditions.

We decided to train the model used for this experiment with the LISA dataset but after realizing that the dataset was very small relative to other image datasets that were used for successful identification with Deep Learning, a plan to combine the GTSRB with the LISA dataset was compiled. During the early phases of the effort it was realized that the two datasets only have one class of sample image in common. Stop signs are the same in both the United States as well as in Germany. Simply increasing the number of classes with a small number of images in a dataset (as would result from a simple combination of both datasets) will not yield more accurate results at training, as not enough features will be discovered to improve identification accuracy.

It was also realized that the datasets contain image samples that were different in focus and size. The GTSRB samples have only a ten percent border around the sign, while the LISA samples and more generically pictures with road signs in the image frame. As such to combine the datasets the stop signs in the LISA dataset would need to be reduced and cropped to have a matching size to the other dataset. It was determined that the effort of combining the two datasets was not an efficient use of time as the images gained in the new dataset would be less than three percent of the images already contained in the GTSRB dataset. As such only the GTSRB dataset was used for image identification in the project. All references to the dataset will reference the GTSRB. While the breakdown of the dataset for training, test, and validation as noted above is the accepted standard, processor capability limited the ability to work with whole sets. Sample images from the dataset were loaded to work with the model in batches containing 4000 to 6400 samples accordingly.

5 IMPLEMENTATION DETAILS

Three main Python scripts were used to produce the bulk of the work. The first script was used to build a workable version of the dataset to be used for training, based on the original GTSRB dataset. The path and names of all the images with their corresponding class were collected from the annotation file provided. The localization and dimensions of the traffic signs in each image were again obtained from the annotation files. For convenience, 43 separate folders were created and the respective traffic sign (also called classes) images were resized to predetermined size of 32x32 and 128x128 pixels and stored respectively.

The second script was used to train the GTSRB dataset using a combination of Convolutional Neural Network models and image sizes. An augmented version of this GTSRB dataset was developed by rotation of the images in various angels. In preparation for the training, a list of all the image filenames were created and shuffled for randomness. This increased the robustness of the network. The process of training was performed in batches where the image files were loaded into memory, divided as training, validation and testing sets with a factor of , and respectively. At the very end, the model with its weights were saved to a .h5 file. For plotting purposes, the history and the scores' results were saved to .npy files.

The third script performs the actual class prediction time measurements. Just like the training script (second script), the process starts by creating the random list of image files and loading the test set to be used for prediction. The file saved by the training script is used to load the desired model and then each sample, or a collection of samples from the test set are then submitted for prediction having their execution time collected using the timeit Python library. Time results are saved in a .npy file to be processed later. This process is then repeated for each trained model selected and we had six different models.

6 MODEL TRAINING (INTERMEDIATE RESULTS)

As part of our study of prediction times for real-time applications, for instance in detection and classification of traffic signs for an autonomous vehicle, we used a variety of convolutional neural network models with distinct characteristics. We were specifically interested in gathering information about the prediction time itself.

Six models were used, with the total number of layers ranging from 3 to 31, more specifically from 1 to 10 convolutional layers. The training used two versions of the dataset with images previously resized to 32x32 pixels and 128x128 pixels. The dataset was divided into 50% for the training, 25% for the validation and 25% for the testing. All the six models were trained on a Intel i7 processor with NVIDIA GPU 1060 series. the physical RAM was about 16 gigabytes. All the models converged for the normal dataset(without augmentation). Typically the deeper models took more than 24 hours to complete the training. The shallowest model 1 took about 6 hours to finish training. The final, trained, models had a number of parameters varying from a little more than 300 thousand up to almost 70 million. A brief summary of the models used is listed in Table 1, where the number of convolutional layers is the most important aspect.

Table 1: Convolutional Network Models.

Model	Layers		Parameters	
	Total	Convolutional	32 Pixels	128 Pixels
1	3	1	1,239,339	28,846,315
2	10	2	319,979	7,397,867
3	31	10	28,463,723	59,921,003
4	16	4	1,482,091	17,210,731
5	16	4	5,835,435	68,749,995
6	17	3	3,788,907	54,120,555

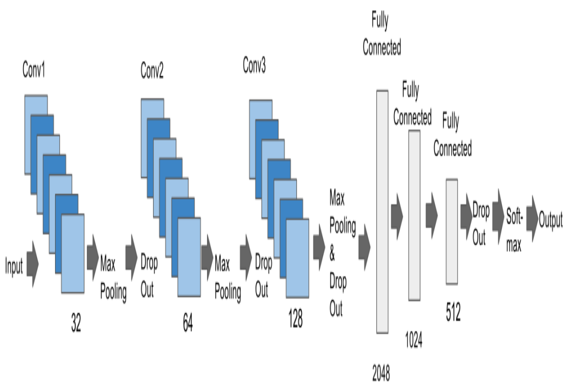

Figure 1: Model 6 Architecture.

Fig. 1 shows the architecture of our best model-the model 6 and Fig. 2 shows the instances from GTSRB dataset. We considered one of the traffic signs, 'Speed Limit of 20kmph' as shown in Fig. 3 and obtained visualizations per layer for model 6. Fig. 4 shows the first layer i.e. the Convolutional Layer with 32 filters. Convolutional layer computes the output of neurons that are connected to local regions in the input, each computing a dot product between their weights and a small region they are connected to in the input volume.

Figure 2: Instances from GTSRB dataset.

Figure 3: Traffic Sign - Speed Limit of 20kmph.

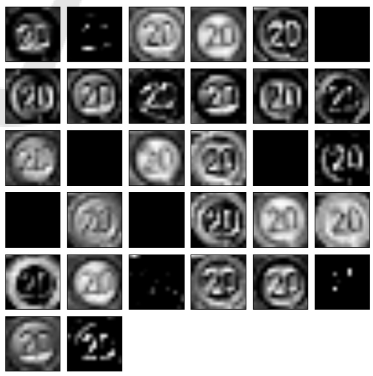

Figure 4: First Layer: Convolutional Layer with 32 filters.

Fig. 5 shows the second layer i.e. Max Pooling layer that performs a downsampling operation along the spatial dimensions (width, height), and fig. 6 shows the eighth layer that applies Max Pooling after the third and final convolutional layer with 128 filters Fig. 7 shows validation accuracy results for a combination of models and image sizes, where a rate of 99%

was achieved in a few cases. Training with a higher number of epochs also showed a slight improvement in overall accuracy. Additionally, an augmented version of the dataset, by means of small rotation on the original images, was also used for training, hoping for better accuracy. In the experiment, image rotations of -4o,-2o,+2o and +4o were added to the dataset at training time. Two of the best models in respect to accuracy, were then selected for training this augmented version of the dataset. Improved accuracy results are shown in Fig. 8.

Figure 5: Second Layer: Max Pooling.

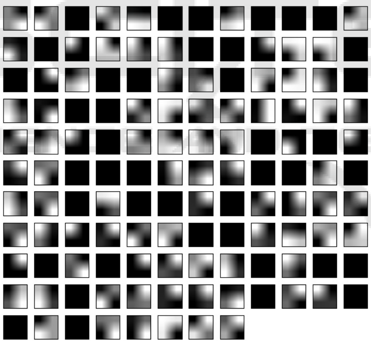

Figure 6: Eighth Layer: Max Pooling.

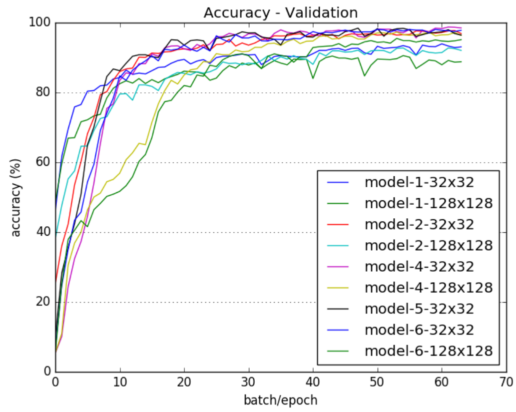

Figure 7: Accuracy for Validation Set.

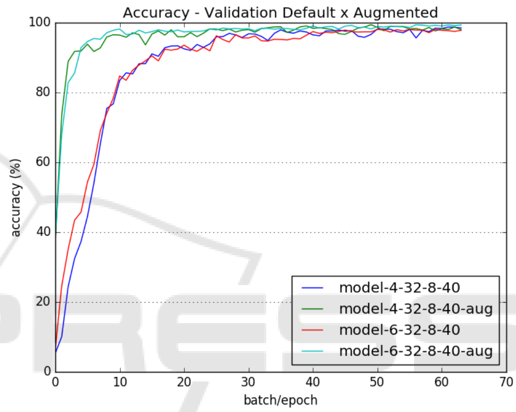

Figure 8: Augmented x Non-Augmented Accuracy.

7 RESULTS - PREDICTION AND MEASUREMENT

After having trained multiple models, in different scenarios, we started performing class prediction time measurements. Using the trained models and a subset of the test set, the prediction time was measured, at first, sample by sample.

The results from all 6 models are shown in Fig. 9. The left side of the image shows the prediction times for samples with 32x32 pixels and samples with 128x128 pixels on the right. Prediction times varied from as low as 0.6ms for the first model up to about 11.5ms for the 3rd model in the case of images with 32x32 pixels. If we consider a scenario where each video frame has an average of 3 traffic signs, they are equivalent to processing 500fps down to 30fps. In the case of 128x128 pixel images, when compared to the 32x32 pixel images, prediction times increased to 2.3ms for the first model and up to 21.0ms for the 3rd model, or in video processing terms, 140fps down to 15fps. The increase in time was about 1.3 times for the best case, up to 7.5 times for the worst case. The model 3 happens to be the shallowest of the 6 models. One important observation made here is, in most cases the prediction time increase was less than 2.5 times when compared to the increase in image size of 16 times in the number of pixels. It shows that the image size is not the main factor behind the increase in prediction times.

Up to this point, all time measurements (and training) were performed on a system based on a NVIDIA GTX 1060 GPU card with 6GB of memory. To better understand the variations in hardware performance, additional experiments were performed with other GPU and CPU based systems. Fig. 10 shows

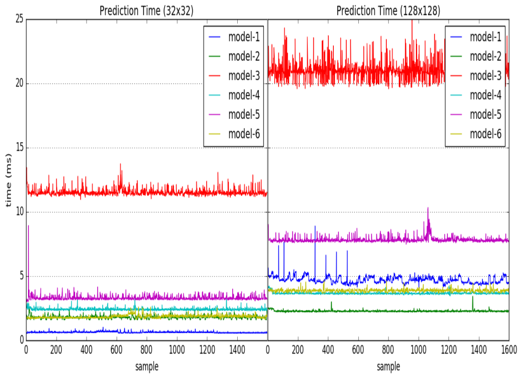

Figure 9: Prediction Times - 32x32 pixels against 128x128 pixels.

the results for prediction times measured on 4 different systems. Both a faster and a slower GPU were used as well a CPU based system for comparison. In this present work we were not able to perform measurements with an embedded solution, more applicable in a real-time system, which will have to be left for future work.

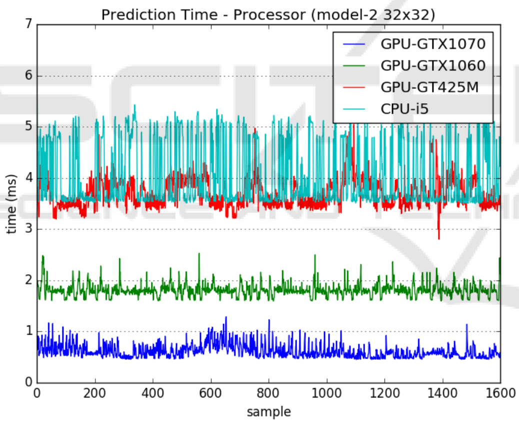

Figure 10: Prediction Times per Processor.

Considering that in a real-time application for traffic sign classification, one particular frame from the video stream can contain multiple detected signs, which can be fed to the classification algorithm all at once for possible improved performance. Another experiment was performed where multiple samples were trained for comparison and the graph in Fig. 11 shows that the overall performance was not affected by the number of samples to be predicted at the same time. We tested with 1, 3 and 5 samples. The time in the graph represents the average time per sample and the behavior was confirmed by similar data collected from other trained models.

A final experiment was performed using the trained models with the augmented dataset in order

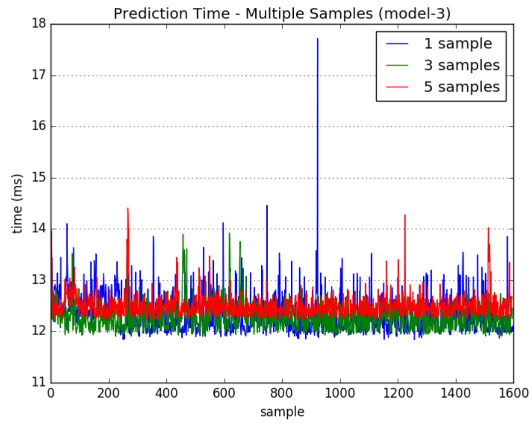

Figure 11: Prediction Times for Multiple Samples.

to determine if the larger dataset (that gave us better accuracy) would affect the overall classification time. Fig. 12 shows an increase of about 16% in prediction time for a dataset 5 times larger than the original.

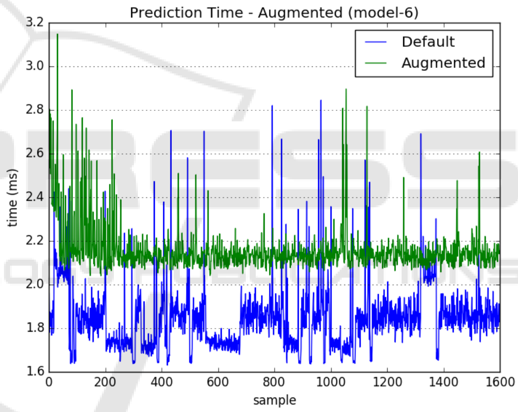

Figure 12: Prediction Times for Augmented Dataset.

8 CONCLUSION

Even though the primary goal for this project was not to achieve the best possible accuracy, we were able to obtain satisfactory results bordering the 99% mark, with slightly better numbers for the augmented dataset. About the prediction time results, when we combine all the performed experiments together, the best observation we can make is, the average time for classification is somewhat proportional to the number of convolutional layers in the respective model. On the contrary, results are not necessarily proportional to the actual total number of layers in the model or even the number of parameters in the model. Overall, the choice of a particular convolutional model plays a bigger role than the change in image size or an augmentation of the dataset. Also, when comparing prediction times for a single sample versus multiple samples, we saw no differences at all, giving the possibility of parallel classification of multiple signs per video frame, instead of just one sign. In all cases, the data collected is hardware dependent, as demonstrated, but with the current state-of-the-art GPUs, real-time capability is indeed achievable, even though, from our experiments, some of the deepest models did not show fast enough results, especially when we consider the fact that we are simply talking about classification time and not account for the actual traffic sign localization within the video frame captured. Several different processors ran six different CNN Deep Learning models. The models were trained and verified to identify German traffic sign images with varying orientations and augmentations from the GTSRB benchmark dataset with a high accuracy. With some variations to the dataset, identification of images occurred in less than one millisecond on one of the processors. In others cases, with deeper networks, more specifically with multiple convolutional layers, the results were not as good in comparison with shallower networks along with a dependency on the processor used. Our model 6 has just three convolutional layers and it gave the best accuracy. After a certain depth, the features that the network looks for may not make sense for the classification purposes and we believe this could possibly be one of the reasons for the multiple convolutional layers not performing better. We aim to further improve prevention of gradient vanishing and use deep residual learning with deeper networks to observe their behavior. As future work, we can modify the dataset by converting the images to gray-scale, known to be helpful to improve accuracy as it was the case with the winner in the GTSRB competition. Another aspect to consider is the distribution of the images within the classes which are not homogeneous. Most common traffic signs have up to 3 thousand images, while the least common ones have only a few hundred. In this case, a reduced version of the dataset, by removing classes where the number of images is less than 1000, could give us a more homogeneous dataset, and hopefully an increase in overall accuracy, by not relying on classes with too few trained samples. And yet another aspect to be considered is to fine-tune the models used in the experiment, in terms of activation and loss functions and its parameters. In terms of hardware, this experiment was limited to desktop or notebook PCs, and so, it did not consider some of the embedded platforms that are intended to real-time applications like traffic sign classification. We intend to implement these networks on NVDIA Tegra X1 development board and achieve great increment in the classification speed for real-time applications.

REFERENCES

Bojarski, M., Del Testa, D., Dworakowski, D., Firner, B., Flepp, B., Goyal, P., Jackel, L. D., Monfort, M., Muller, U., and Zhang, J. e. a. (2017). End to end learning for self-driving cars. *Arxiv.org*.

Chen, L., Li, Q., Li, M., Zhang, L., and Mao, Q. (2012). Design of a multi-sensor cooperation travel environment perception system for autonomous vehicle. *Sensors*, 12(12):12386–12404.

Ciresan, D., Meier, U., Masci, J., and Schmidhuber, J. (2011). A committee of neural networks for traffic sign classification. *The 2011 International Joint Conference on Neural Networks*.

Greenhalgh, JackMirmehdi, M. (2012). Real-time detection and recognition of road traffic signs. *IEEE Transactions on Intelligent Transportation Systems*, 13(4):1498–1506.

Igel, C. German traffic sign recognition benchmark gtsrb dataset available at: http://benchmark.ini.rub.de/?section=home.

Jin, J., Fu, K., and Zhang, C. (2014). Traffic sign recognition with hinge loss trained convolutional neural networks. *IEEE Transactions on Intelligent Transportation Systems*, 15(5):1991–2000.

Le, T. T., Tran, S. T., Mita, S., and Nguyen, T. D. (2010). Real time traffic sign detection using color and shape-based features. *Intelligent Information and Database Systems*, pages 268–278.

Mao, X., Hijazi, S., Casas, R., Kaul, P., Kumar, R., and Rowen, C. (2016). Hierarchical cnn for traffic sign recognition. *2016 IEEE Intelligent Vehicles Symposium (IV)*.

Menge, Anheuser-Busch, O. (2016). Complete first self-driving truck delivery of beer. *Neural Networks*.

Sermanet, PierreLeCun, Y. (2011). Traffic sign recognition with multi-scale convolutional networks. *The 2011 International Joint Conference on Neural Networks*.

Stallkamp, J., Schlipsing, M., Salmen, J., and Igel, C. (2012). Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323–332.

Trivedi, M. M. Lisa dataset. available at: http://cvrr.ucsd.edu. *Cvrr.ucsd.edu*.

Zeng, Y., Xu, X., Fang, Y., and Zhao, K. (2015). Traffic sign recognition using deep convolutional networks and extreme learning machine. *Lecture Notes in Computer Science*, pages 272–280.