Classification of partially occluded road signs using Convolutional Neural Networks and Support Vector Machines

ANTONIA SCHROFF ALEXANDER SELIVANOV
{schroff, seliv}@kth.se

November 8, 2022

Abstract

While object recognition algorithms have reached commercial standards, occlusions of the object to be detected still pose a major problem. Sensors misinterpret their environment due to objects blocking the line of sight, making it more difficult for Artificial Intelligence to classify the picture. In our work we simulate occlusion by modifying a dataset of traffic signs and conduct a quantitative comparative analysis between a Convolutional Neural Network and a Support Vector Machine. Our goal is to determine the best model for traffic sign recognition in aggravated conditions. We compare the models on different levels of occlusion. Our research shows that Support Vector Machines generally perform better at the tested levels of occlusion, however accuracy decreases strongly for both models.

Contents

List of Acronyms and Abbreviations

CNN Convolutional Neural Network

FPS Frames Per Second

GHT Generalized Hough Transform

GPU Graphics Processing Unit

HOG Histogram of Oriented Gradients

RGB Red, Green, Blue

SRC Sparse Representation Classification

SVM Support Vector Machine

TSR Traffic Sign Recognition

YOLO You Only Look Once

1 Introduction

1.1 Background

While object recognition has made plenty of progress in the last couple of years ago, it is still immensely difficult to classify an object. Challenges in Computer Vision can be viewpoint variations, deformations, occlusions, illumination conditions and cluttered or textured backgrounds. In this paper, we want to focus on the problem of occlusions in the context of Computer Vision. As described in [1], occlusions can be separated into two kinds: there is self-occlusion, which means that an object occludes parts of itself. Inter-occlusion describes one object occluding another one. Occlusions occur constantly. In dense traffic situations, cars occlude each other, just as pedestrians do on a crowded street. Another big role plays into occlusion due to weather conditions, as precipitation interferes with measurements made by LIDAR, SONAR or camera sensors. As many try to solve the problem of object occlusion by optimizing hardware related parameters like the choice or the number of sensors, we want to look at the algorithm-related part.

In [2], occlusion is looked at in the context of a crowded mass of pedestrians. As soon as occlusion is detected, the target tracking is deactivated and later reactivated as soon as the object is occlusion free again. Similarly, in [3], the object detection process is separated into before-occlusion, during-occlusion and after-occlusion. Both papers assume that occlusion appears in a dynamic form, e.g. through people occluding each other. In [4], a relative discriminative histogram of oriented gradients based particle filter is used to improve robustness and accuracy, once again to track an object by using a video input. In [5], different CNN architectures are compared in the case of partial occlusions.

One of the major application areas of object detection and recognition is in the context of traffic and navigation. A technology already used in commercial vehicles is traffic sign detection and recognition, which classifies detected traffic signs in real time. Occlusions of traffic signs can be in the form of moving vehicles, but it can as well be a more permanent occlusion. Examples for permanent occlusion are snow covered traffic signs and soiled signs. Traffic sign detection and recognition is very suitable to keep the paper in an appropriate scope. In traffic sign detection and recognition it is not possible to adapt the camera angle in a more suitable manner, as the vehicle has to stay on the road. Occlusions often are permanent, which means that there is little use in tracking the signs to gather more information. Furthermore, traffic sign recognition and detection is performed with cameras as sensors, as LIDAR or SONAR wouldn't be able to detect the pictures shapes on a traffic sign. This allows us to use a dataset of images as an input.

In [6], the authors describe an end-to-end system to achieve real-time Traffic Sign Recognition (TSR). They show that using a Graphics Processing Unit (GPU) allows them to achieve computation of full HD 1920x1080 camera frames in a time span of 10 milliseconds. The extraction of the traffic sign location in the image was done using a modified version of the Generalized Hough Transform (GHT) algorithm. Some additional steps were used to improve the accuracy, for example they resorted to using the speed of the vehicle to estimate the location and size of the sign in neighboring frames. When it comes to the classification step, they tested template matching but established that performance was degraded due to variations in contrast, lighting and having different orientation of a sign in regard to the camera frame. To alleviate this, the authors propose the implementation of a Convolutional Neural Network (CNN) algorithm as a classifier. They achieved very good results for real-time classification with a total accuracy of 99.94% with a processing speed of 50 Frames Per Second (FPS). For future work, they propose the exploration of bad weather conditions which fits with the topic of this project.

In [7], the authors created a new dataset comprised of 2728 photos of a variety of 24 Arabic traffic signs sampled in the eastern province of Saudi Arabia. They also implemented a CNN with the following architecture: two convolution layers, two max pooling layers, one dropout layer and three dense layers. With such an architecture they managed to achieve an accuracy of 100% when testing with their own dataset. In the future they plan on increasing the amount of images in their dataset and making it publicly

available for others to use. It would be interesting to compare the results we can obtain in our project by using an architecture similar to theirs when it comes to implementing the CNN.

In [8], the authors implemented a TSR system. To detect the presence of a traffic sign in an image they used color based segmentation where an image is separated into clusters based on the color in the image. For the classification step which is of higher interest to our research, they first used Histogram of Oriented Gradients (HOG) to extract and describe the features present in an image. Followed by a binary Support Vector Machine (SVM) to conduct the classification with which they obtained an accuracy of 98.0%. The detection step had an accuracy of 91%. They achieved processing of 5 FPS. They did an accuracy comparison and their method is just slightly behind the leading method they have chosen to compare to, which is CNN. They state that they are already working on the inclusion of deep learning techniques in combination with their method. It is therefore useful to conduct our research and compare how well SVM and CNN perform when the object we are detecting is partially occluded.

1.2 Problem Discussion

According to [5], adding partial occlusion to images significantly affects the performance of CNNs. The same goes for [9], which tries to improve a SVM in the application area of face recognition with partial occlusion. However, both managed to crucially improve their algorithm's accuracy by adapting them to the respective application case. In the research area of Object Detection, less work is available on TSR than on many other application areas. There is quite few work available on TSR when working with partial occlusions. In [10], TSR is performed under partial occlusion, comparing SVMs and Sparse Representation Classification (SRC). SVM combined with SRC turns out to have the highest accuracy in the case of occlusion. In our work we want to conduct a quantitative comparison of both SVM and CNN for traffic sign detection. Compared to previous research, our work is focused on longer term occlusions like snow build up on traffic signs.

1.3 Research questions, hypotheses

Which algorithm between CNN and SVM offers a higher general accuracy when classifying partially occluded objects?

2 Methods

2.1 Choice of research methods

A wide range of classifiers and learning algorithms are used in Computer Vision nowadays. Most commonly used algorithms are YOLO, SVM and CNN. As SVMs are a standard in Computer Vision and object recognition but at the same time have not been subject to papers working with occlusions, we chose SVM as one of the algorithms we want to test. SVMs are known for being memory-efficient, which means that during classification, they use a subset of training points. Furthermore they perform well in high-dimensional spaces, which is why we think that they are well suited for real-time object recognition. CNNs have been discussed in many papers regarding object recognition and have usually been performing the best, which is why we choose them as a benchmark to compare to the SVM. To limit the scope of the study, we will not include You Only Look Once (YOLO) in the comparison.

In practice, occlusion is often simulated to augment datasets and therefore prevent overfitting. Occlusion techniques are

- Random Erase (cutting out a random part of the picture and replacing it with noisy pixels)
- Cutout (rectangles are randomly cut out from the image)
- Hide and Seek (dividing the image into a grid and randomly cutting out some of the cells)
- Grid Mask (cut out squares arranged in a grid)

State of the art occlusion methods are MixCut and Mosaic. We consider Hide and Seek, Grid Mask, MixCut and Mosaic as unsuitable for the simulation of naturally occurring occlusion, which is why we choose Random Erase as well as Cutout as occlusion techniques. As we don't know which occlusion technique is more realistic, we perform the quantitative comparison on both occlusion types.

The data we use to train the models is modified depending on the model we use it for, as we want to optimize the CNN as well as the SVM and both have different properties that need to be taken into account.

Our KPI to indicate the performance of our models is their validation accuracy, we omit their training duration and classification speed. To assess their general accuracy, we run test runs several times and evaluate the mean average validation accuracy, as it has been done in [11].

2.2 Data set manipulation

The data set our work is based on is provided by [12]. It includes over 52,000 fully annotated high-resolution images of traffic signs from all over the world. The images vary in weather, season, time of day, camera and viewpoint. It provides the original images in JPEG format, JSON files with information about the dimensions of the image, the bounding boxes and the labels for the traffic signs located in the image as well as further information that is not relevant to our work. We create a copy of all the traffic signs and artificially occlude them, which means that we can control the level of occlusion to therefore compare the methods on different levels of occlusion. By occluding the signs, we simulate conditions like e.g. snow on the sign or other objects that interfere with an unobstructed view of the traffic signs.

As we skip the traffic sign detection step and only compare traffic sign recognition methods, we first use the provided annotations to crop the pictures around the area of the boundary boxes to obtain the area of the detected traffic sign. An example of a camera frame with the bounding boxes drawn over can be seen in figure 1. In the next step, we generate occlusion by selecting a random area of the image and setting the pixels inside that area to some color. To produce two occlusion variants, in one case the color is black and in another each pixel is of a random color. The occlusion is not limited to the area of the traffic sign and is rectangle shaped with varying proportions. The size of the occluded area in the image is be set by us and varies from 10% to 40% occlusion.

After cropping images in the initial dataset and ignoring cropped images with a dimension below 32x32 pixels, we obtained 93,901 road sign images spread over 401 different classes. Following this we looked at the class distribution and found out that this new dataset had a single class that contained over 50% of the images, more specifically 53,184 images were in class named "other-sign". We decided to exclude this class from the dataset that will be used for training both classification models. We also noticed that many classes have very few image samples. We therefore decided to ignore the classes that have less than 200 available samples as to not train the models on signs which we do not have sufficient data. The resulting dataset is composed of 19,020 images and 42 classes. The final class distribution can be seen in figure 2. It is clear that the dataset is not the most balanced in terms of classes.

Both algorithms (SVM and CNN) were trained using the same datasets, some examples of the images from the datasets can be seen in figure 4. These are composed of the images from the non-occluded road

Figure 1: The data set by [12] provides camera images, as seen on the left, as well as information about the location of the detected traffic signs.

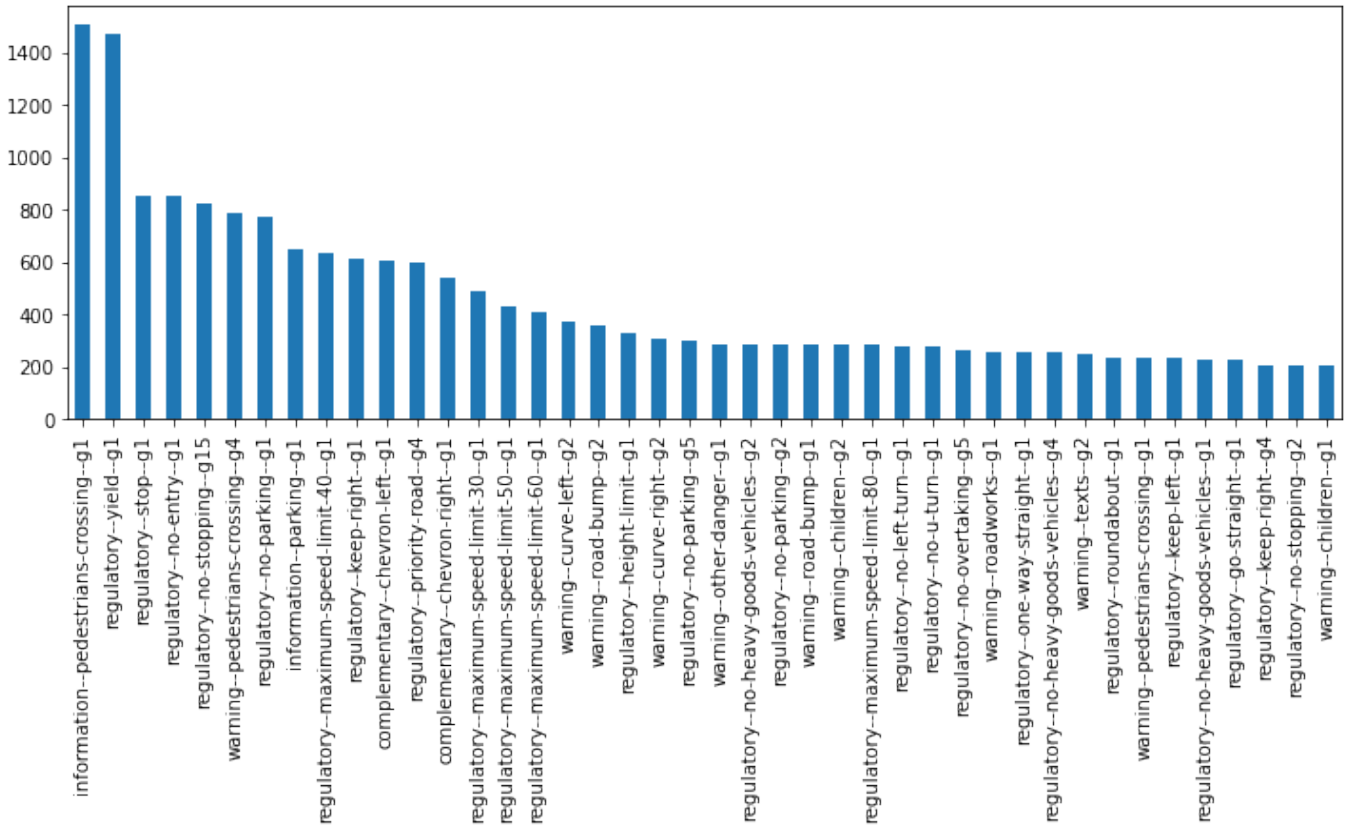

Figure 2: Dataset class distribution

signs ran through our occlusion method. An example of what is done to each image can be seen in figure 3. The resulting datasets are the following: original cropped images, 10%, 20%, 30% and 40% occlusion.

2.3 SVM

2.3.1 Data preparation

In preparation for the training of the SVM model. All the images were resized to a fixed dimension of 32x32 pixels. This was followed by the extraction of features from each image using HOG seen in figure 5. This feature descriptor allowed us to reduce the amount of features to less than the sum of all the Red, Green, Blue (RGB) components of each pixel in the image, if we were to use image colors as features.

Figure 3: Samples with 20% occlusion. Original, w/ black pixel occlusion, w/ colored pixel occlusion.

Figure 4: Samples of various occluded signs showcasing both variants.

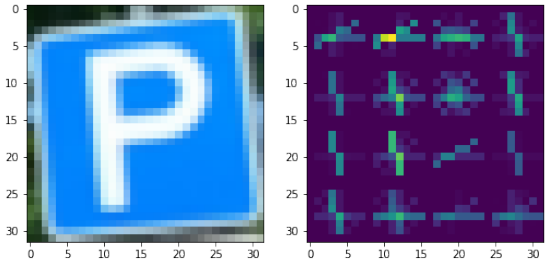

Figure 5: Example of the HOG features extracted from an image.

2.3.2 Training

For the training of the SVM model, we used the scikit-learn [13] library with which we first implemented the classifier using the default hyper-parameters provided. This was done to obtain a baseline for the model accuracy that we could then compare to when it comes to the tuning and optimization of the model. Second, we went forward with tuning of hyper-parameters by specifying a parameter distribution and using random search testing available in the aforementioned library. Once tuning was complete, we had a final accuracy baseline to which we could compare the subsequent models trained with images that contain partial occlusion.

2.3.3 Testing

After training the SVM models, we used a test set of 2853 images to confirm the respective performances of each model when it comes to the prediction of a dataset unknown to the training step.

2.4 CNN

A typical CNN's structure is made up of one or more Convolutional Layers, followed by a Pooling Layer. We try to find an appropriate architecture for the CNN as well as optimize the training parameters to maximize the CNN's accuracy.

2.4.1 Data augmentation

To generalise better and provide more images to the training of the model, we use data augmentation which can improve the accuracy of the model. As CNNs are not neutral to rotation and similar, we rescale the images, change their zoom range, their shear range and flip them horizontally.

Figure 6: On the left, the original image is shown. On the right side, the same image has been mirrored, zoomed in and distorted. It has also been rescaled by the factor 1/255.

2.4.2 CNN Structure

The CNN is built from six layers. The first layer is a Convolution layer. The main goal of convolution is to extract features from the input image. By learning image attributes with tiny squares of input data, convolution maintains spatial connectivity between pixels. The second layer is a Pooling layer, which reduces the dimensionality of the feature map. The third layer is a Flattening layer, which turns the output into a linear array. The last three layers are all Density layers, which are supposed to turn our so far convolutional network to a neural network. The first two layers use ReLu as an activation function, which is today's most popular activation function. "The biggest advantage of ReLu is indeed non-saturation of its gradient, which greatly accelerates the convergence of stochastic gradient descent compared to the sigmoid / tanh functions."[14] The activation function of the last layer is a *Softmax* function with 42 units, according to our 42 classes. *Softmax* is used for multi-class classification tasks.

2.4.3 Training

Parameters that we tune for training the model are the batch size and the number epochs. The batch size can be between 1 and the total number of images in the training set and represents the number of samples per batch. "It has been observed in practice that when using a larger batch there is a significant degradation in the quality of the model, as measured by its ability to generalize."[15] To keep the quality of the model high but also taking the running time into concern, we set the batch size to 32. The number of epochs determines how many times the complete data set gets passed through the CNN. The higher the number of epochs, the higher the model's accuracy. In practice, the number of epochs can be determined by observing at which number of epochs the accuracy does not change anymore. To limit the running time of the training, we set the number of epochs to 100. As to be expected, the accuracy is dependent on occlusion level, this can be seen in figure 7.

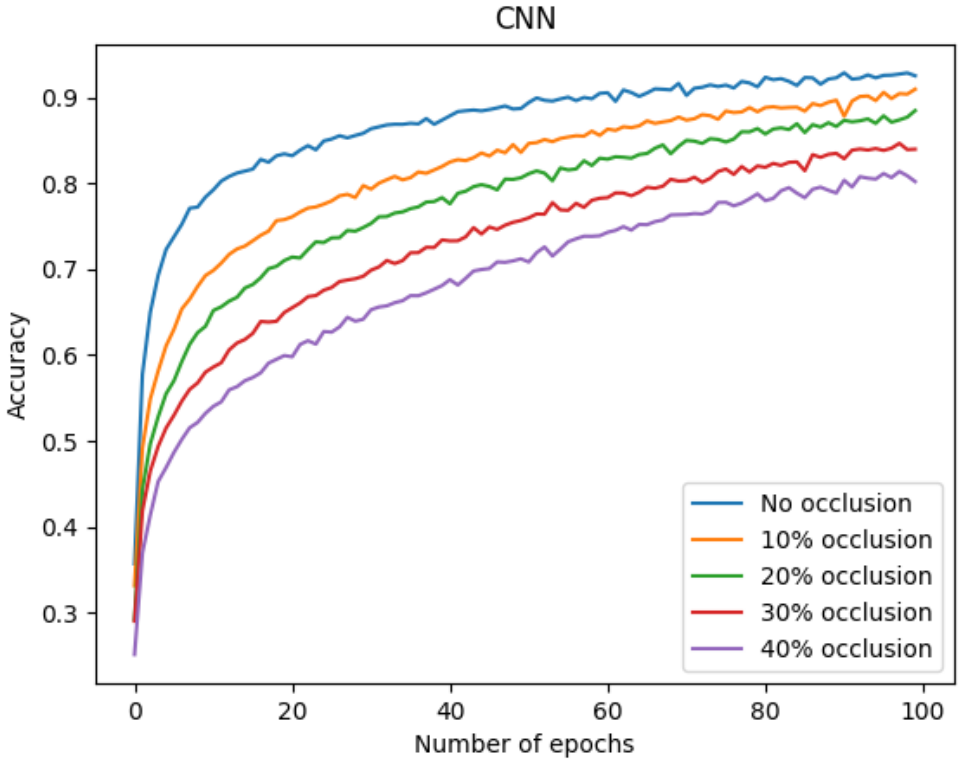

Figure 7: The marginal growth of accuracy decreases with an increasing number of epochs. Training with cutout occlusion method.

3 Results and Analysis

Table 1 and figure 8 show the results of the obtained accuracy for both algorithms. Each algorithm has been evaluated for an occlusion level of 0%, 10%, 20%, 30% and 40%. At a level of 0% occlusion, the SVM model outputs an accuracy of ≈90%. With the cutout occlusion method, at only 10% image occlusion, the accuracy drops by about 9% and at 40% occlusion, the accuracy has decreased by approximately 38%. A similar trend is seen when we look at the results for the random erase occlusion method.

CNN on the other hand delivers generally lower accuracy across various occlusion levels. With the cutout method at 0% occlusion we get a accuracy of ≈73% which is already quite lower than SVM. With each following occlusion level the model accuracy drops approximately by 5%. At the highest occlusion of 40% CNN delivers a ≈57% accuracy which is quite on par with SVM's ≈56%.

Based on the results of both occlusion methods, it would seem that using the cutout occlusion method results in higher accuracy, at least with our method and the dataset we worked with.

An additional interesting pattern is that the higher the occlusion level the closer the accuracy for both algorithms becomes. We have a ≈2% difference when testing with the cutout method and just over 1% difference with random erase.

Algorithm	Occlusion type and percentage								
		Cutout				Random erase			
	0%	10%	20%	30%	40%	10%	20%	30%	40%
SVM	**90.40**	**81.77**	**75.15**	**66.77**	55.73	**80.48**	**69.96**	**59.55**	50.19
CNN	83.37	73.15	68.60	62.53	**57.03**	73.19	63.99	56.19	**51.54**

Table 1: Validation accuracy for each algorithm depending on the level of occlusion and occlusion type.

4 Conclusion and Future Work

Concluding it becomes clear that in most of our runs, SVM performs better for both the Cutout and the Random erase occlusion technique. An interesting observation is that CNN takes over in terms of accuracy with an increasing level of occlusion however overall SVM offers a higher general accuracy when classifying partially occluded objects.

In future work, the models should be tested with data having higher levels of occlusion to validate the hypothesis that CNN might offer a higher accuracy in that scenario. Furthermore, YOLO could be included in the comparison. Also, it would be interesting to compare additional occlusion techniques that were not tested in our work but which could offer a more realistic occlusion. This includes two methods: blurring a section of an image and overlapping another image over a section of the image.

Our results can be used for further research that aims at improving object detection at the software implementation level.

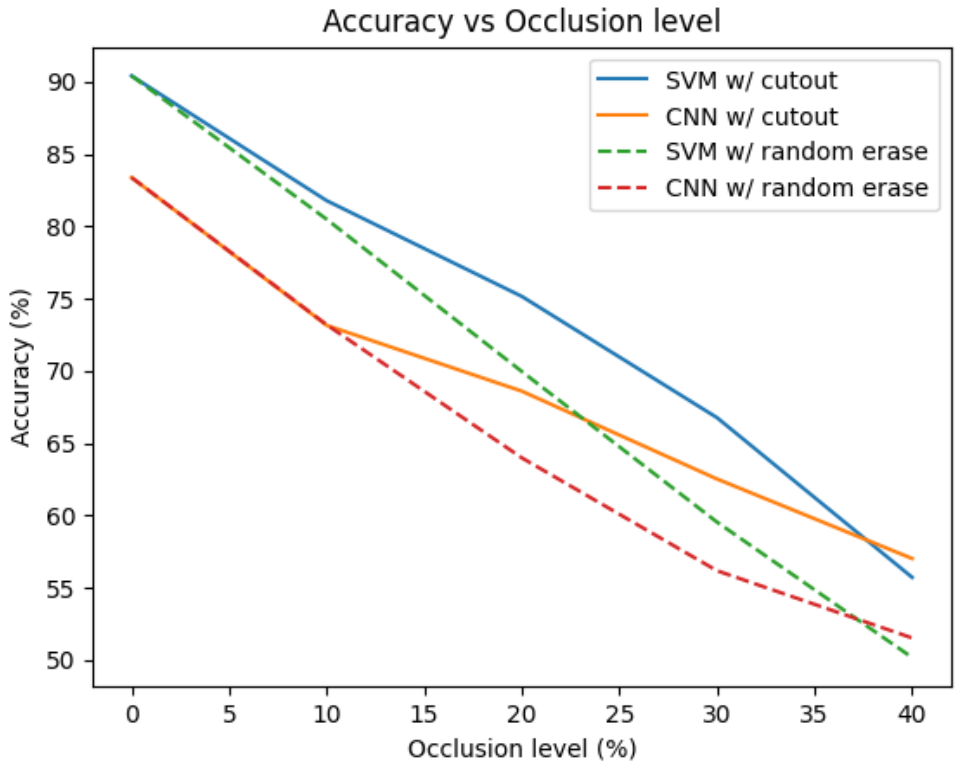

Figure 8: Accuracy of the models with increasing level of occlusion.

References

[1] H. Chandel and S. Vatta, "Occlusion detection and handling: A review," *International Journal of Computer Applications*, vol. 120, pp. 33–38, 06 2015. doi: 10.5120/21264-3857

[2] Y. Guan, X. Chen, D. Yang, and Y. Wu, "Multi-person tracking-by-detection with local particle filtering and global occlusion handling," in *2014 IEEE International Conference on Multimedia and Expo (ICME)*, 2014. doi: 10.1109/ICME.2014.6890149 pp. 1–6.

[3] T. Yang, Q. Pan, J. Li, and S. Li, "Real-time multiple objects tracking with occlusion handling in dynamic scenes," 07 2005. doi: 10.1109/CVPR.2005.292. ISBN 0-7695-2372-2 pp. 970– 975 vol. 1.

[4] B.-F. Wu, C.-C. Kao, C.-L. Jen, Y.-F. Li, Y.-H. Chen, and J.-H. Juang, "A relative-discriminative-histogram-of-oriented-gradients-based particle filter approach to vehicle occlusion handling and tracking," *Industrial Electronics, IEEE Transactions on*, vol. 61, pp. 4228–4237, 08 2014. doi: 10.1109/TIE.2013.2284131

[5] A. Kortylewski, Q. Liu, A. Wang, Y. Sun, and A. Yuille, "Compositional convolutional neural networks: A robust and interpretable model for object recognition under occlusion," *International Journal of Computer Vision*, vol. 129, no. 3, p. 736–760, 2020. doi: 10.1007/s11263-020-01401-3

[6] A. Shustanov and P. Yakimov, "Cnn design for real-time traffic sign recognition," vol. 201. Elsevier Ltd, 2017. doi: 10.1016/j.proeng.2017.09.594. ISSN 18777058 pp. 718–725.

[7] D. A. Alghmgham, G. Latif, J. Alghazo, and L. Alzubaidi, "Autonomous traffic sign (atsr) detection and recognition using deep cnn," vol. 163. Elsevier B.V., 2019. doi: 10.1016/j.procs.2019.12.108. ISSN 18770509 pp. 266–274.

[8] S. Ardianto, C. J. Chen, and H. M. Hang, "Real-time traffic sign recognition using color segmentation and svm." IEEE Computer Society, 6 2017. doi: 10.1109/IWSSIP.2017.7965570. ISBN 9781509063444. ISSN 21578702

[9] H. Jia and A. M. Martinez, "Support vector machines in face recognition with occlusions," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009. doi: 10.1109/CVPR.2009.5206862 pp. 136–141.

[10] C. Liu, F. Chang, and Z. Chen, "High performance traffic sign recognition based on sparse representation and svm classification," in *2014 10th International Conference on Natural Computation (ICNC)*, 2014. doi: 10.1109/ICNC.2014.6975818 pp. 108–112.

[11] C. Nguyen, Y. Wang, and H. N. Nguyen, "Random forest classifier combined with feature selection for breast cancer diagnosis and prognostic," *Journal of Biomedical Science and Engineering*, vol. 06, no. 05, p. 551–560, 2013. doi: 10.4236/jbise.2013.65070

[12] C. Ertler, J. Mislej, T. Ollmann, L. Porzi, and Y. Kuang, "Traffic sign detection and classification around the world," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2020.

[13] "scikit-learn," https://scikit-learn.org/stable/index.html, accessed: 2021-10-08.

[14] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks." [Online]. Available: http://code.google.com/p/cuda-convnet/

[15] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang, "On large-batch training for deep learning: Generalization gap and sharp minima," 2017.