Final Project Report

Juan Carlos Cruz - ira406

ME 6543 Machine Learning and Data Analytics

Abstract

This paper implements and evaluates a traditional computer vision approach to object detection using Histogram of Oriented Gradients (HOG) features with Support Vector Machine (SVM) classification on a modern industrial safety dataset.

The SH17 dataset, containing 8,099 annotated images with 75,994 instances across 17 classes, is used to train and evaluate a person detector. A linear SVM with stochastic gradient descent optimization is implemented, processing 64,990 total features from 10,920 positive and 54,070 negative samples. The model's performance is evaluated against YOLO8, YOLO9, and YOLO10 models from the SH17 paper.

While the HOG-SVM approach achieved limited performance (mAP50: 0.4%, mAP50-95: 0.2%) compared to deep learning methods (best mAP50: 70.9%, mAP50-95: 48.7%), analysis of the detection boxes shows that our model's predictions tend to fall within ground truth areas, suggesting potential for improvement through enhanced post-processing and parameter tuning.

Introduction

Object detection is a core task in computer vision research and development. Modern deep learning approaches like convolutional neural networks (CNN) have been the standard approach to the object detection for RGB images ever since their capabilities for training with GPU hardware was demonstrated in 2012 [1]. Since then architectures such as YOLO [2] and Faster R-CNN [3] have been developed to improve the speed and accuracy of object detection tasks.

In this report my goal is to evaluate the performance of a more traditional object detection approach, the Histogram of Oriented Gradients (HOG) combined with a linear Support Vector Machine (SVM) classifier, on a recent dataset, the Safe Human dataset, a workplace safety dataset with 17 different objects [4]. The linear SVM approach is chosen as that is what is used in the original HOG paper [5], in addition because linear SVM approaches are considered to be fast at inference time while requiring less computational resources for training compared to deep learning approaches such as a CNN.

Literature Review

Object detection from images involves two tasks: object classification and object locating. Both of these tasks involve the extraction of meaningful features from an image. In modern CNN approaches to object detection, the feature extraction tends to be layers within the models architecture [6], while traditional computer vision approaches required engineered feature extraction methods.

One such feature extraction method is the Histogram of Oriented Gradients (HOG) method which captures local gradient structures from cells within an image [5], in fact in their paper the authors show "essentially perfect" classification of pedestrians in a MIT prediction test set. Other popular feature extraction methods at the time included the Scale-Invariant Feature Transform (SIFT) method which captures local invariant features from an image [7].

For the classification and detection tasks, different approaches to the learned Support Vector Machines (SVM) have been used. In [8], the authors detail a pedestrian detector based on a polynomial SVM using rectified Haar wavelets. A human limb classifier using a SVM and 1st and 2nd Gaussian features is developed in [9].

Other previously used learning based approaches include works such as [10], where the authors build a person detector using AdaBoost and Haar-like features.

Problem Description

The SH17 dataset contains 8,099 annotated images with 75,994 instances across 17 classes. From these classes, we aim to to develop a detection model for just the person class. The model will use a SVM classifier to classify features extracted with the HOG method as a person or not a person.

The performance of the model will be evaluated using the benchmarks described in

the SH-17 paper, including precision, recall, and mean average precision (mAP). Our model's performance will then be compared to benchmarks obtained in the SH17 paper for the YOLO8, YOLO9, and YOLO10 architectures.

Method

The code used for this project is based on the implementation code found in the following repository [11].

Included in this paper's submission files are a README, python pip requirements file, evaluation figures, the model itself, a training notebook, a validation notebook, a figure generation notebook, and a sample notebook with a portion of the dataset for the reader to test the code. If further testing is required, the README provides additional instructions for downloading the dataset and structuring the data.

Preprocessing

The dataset splits the images into training and validation data along with their respective annotations. The annotations are parsed so only the "person" annotation data is used. The images are loaded and converted to greyscale.

The preparation of training data requires both positive and negative samples. Positive samples are extracted from the annotated person bounding boxes and assigned a label value of 1, while negative samples are derived from image regions outside these boxes and assigned a label value of 0.

A sliding window approach generates feature extraction windows, using a fixed window size of (64, 128) pixels. Once a sampling window is calculated, we then use the skimage [12] HOG implementation to generate the feature data for that window. The process is repeated for all the person bounding boxes in the image and for non-person regions. An example of the HOG extraction window is shown in Figure 1.

An issue encountered by doing this is that without any limits on sample size we get a very large amount of negative samples for the dataset, greatly outnumbering the positive samples. This could potentiality lead to overfitting, but more practically this was causing the training notebook to crash as the more than 100 batch files were generated of a few GBs of size.

Initially a Linear Support Vector Classification (SVC) was implemented for this training, but when the data batches were passed to the model for fitting, the memory usage exceeded the capabilities of the model code. At that point, I investigate alternative optimizer

Figure 1: An example of the HOG feature extraction window during the evaluation process.

approaches and found [13], which suggested a SGD optimizer instead of the linear SVC due to the large amount of data. In addition to the change of optimizer, a ratio of 5 negative to every positive sample was added to the sample generation to limit the number of negative samples.

Training

For training, an sklearn [14] pipeline is created using stochastic gradient descent (SGD) learning method and fit with a linear SVM.

As described in the preprocessing section, the large amount of samples resulted in a exceeding the memory capabilities in the original implementation. Given the large amount of sample data available, the SGD optimizer method is chosen as recommended in the sklearn documentation [15].

The default parameters are used for training including a tolerance of 0.001 and a max iteration of 1000. During the model fitting step the max iterations were met, thus convergence was not achieved in our training. Since SGD is sensitive to the feature scaling, the model input scaling in included in the model pipeline using the StandardScaler from sklearn.

In total 6479 images were loaded as training data with the following output:

- Total features: 64990
- Positive samples: 10920
- Negative samples: 54070
- Feature dimension: 3780

Implementation

For the evaluation pipeline, the sliding window approach is again used but incorporates a image scaling pyramid to handle size variations of the target objects similar to that used in [11]. In the sliding window the HOG feature extractor method is again used with the same parameters as in the training step.

The trained SVM model provides confidence scores for each window using decision function method, which returns the signed distance from the separating hyperplane, we then keep those above a certain threshold as detections.

Without post processing, the model outputs a large amount of detection boxes which overlap and leads to multiple detections of the same object when using the Intersection over Union (IoU) metric as a threshold for positive and false positive detections.

To account for this, post processing methods are added. First, boxes are merged using a density-weighted averaging approach. Then the boxes are adjusted to maintain a reasonable aspect ratio. Finally, a non-maximum suppression step is applied to keep only the highest scoring boxes that don't significantly overlap. The results of a post processing step can be seen in Figure 2.

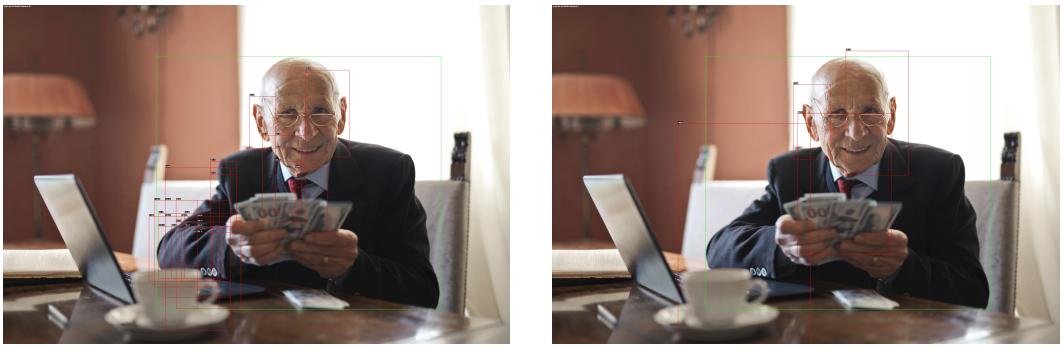

Figure 2: Example of post-processing results. Left: Raw detection windows with confidence scores above threshold. Right: Final detections after post processing.

For the evaluation code, the author found a confidence threshold of 1000 to be suitable for accepting a prediction as a positive detection. It's worth noting that since the model used a scaler in it's pipeline, features with scores above 0.0 are considered within the "person" classification, while those below 0.0 are negative.

Results

Evaluation Metrics

The SH17 paper uses the following metrics to benchmark their model's performances:

$$Precision = \frac{True\ Positive}{True\ Positive + False\ Positive} \tag{1}$$

$$Recall = \frac{True\ Positive}{True\ Positive + False\ Negative} \tag{2}$$

The Mean Average Precision (mAP) metric is also used in the SH17 paper. The mAP metric is calculated by averaging the average precision (AP) of each class, which in the COCO evaluation system [16] is calculated at different Intersection over Union (IoU) thresholds.

The Intersection over Union (IoU) metric is used to gauge the accuracy of object localization. It calculates the overlap between the ground truth bounding boxes (b_g) and the model's predicted bounding boxes (b_{pred}) as follows:

$$IoU = \frac{Area(b_{pred} \cap b_g)}{Area(b_{pred} \cup b_g)} \tag{3}$$

where b_g represents the ground truth bounding box, and b_{pred} denotes the bounding box predicted by the model.

Model Performance

The results of the developed model and the YOLO models trained in SH-17 paper are displayed in Table 1. Our model is listed on the top row of the table as HOG-SVM, and the YOLO models are listed below it. The top performing model is bolded.

Model	Params (M)	Images	Instances	P (%)	R (%)	mAP50 (%)	mAP50-95 (%)
HOG-SVM	-	1620	-	3.4	1.9	0.4	0.2
Yolo-8-n	3.2	1620	15358	67.5	53.6	58.0	36.6
Yolo-8-s	11.2	1620	15358	81.5	55.7	63.7	41.7
Yolo-8-m	25.9	1620	15358	77.1	60.5	66.6	45.7
Yolo-8-l	43.7	1620	15358	76.7	62.9	68.0	47.0
Yolo-8-x	68.2	1620	15358	77.1	63.1	69.3	47.2
Yolo-9-t	2.0	1620	15358	75.0	52.6	58.5	37.5
Yolo-9-s	7.2	1620	15358	73.6	60.2	65.3	42.9
Yolo-9-m	20.1	1620	15358	77.4	62.0	68.6	46.5
Yolo-9-c	25.5	1620	15358	79.6	60.8	67.7	46.5
Yolo-9-e	**58.1**	**1620**	**15358**	**81.0**	**65.0**	**70.9**	**48.7**
Yolo-10-n	2.3	1620	15358	66.8	53.2	57.2	35.9
Yolo-10-s	7.2	1620	15358	75.8	57.0	62.7	40.9
Yolo-10-m	15.4	1620	15358	71.4	61.4	65.7	43.8
Yolo-10-b	19.1	1620	15358	77.7	59.1	65.8	45.1
Yolo-10-l	24.4	1620	15358	76.0	61.8	67.4	46.0
Yolo-10-x	29.5	1620	15358	76.8	62.8	67.8	46.7

Table 1: Model performance comparison between the HOG-SVM model and the YOLO models from the SH17 paper [4].

Discussion

From evaluating a few samples of the model's predictions, it was observed that our model would fall below the 0.5 IoU threshold, which is the minimum threshold for a detection to be considered a true positive, thus the model's performance was extremely poor on using COCO evaluation metrics.

As described in the previous section, steps were taken in the implementation to merge the multiple prediction boxes into a more representative single box per person class, which would potentially improve the IoU value with respect to the ground truth boxes. In improving the IoU value, we could see more true positives being detected, which would improve the precision and recall values of the model. In addition hyper parameter tuning for the SGD optimizer could be done in training, but these improvements are beyond the scope of this project.

Conclusion

In this report, we compared traditional machine learning approaches to more modern deep learning methods for person detection in images. A linear SVM classifier with a SGD optimizer was chosen in this case, as linear SVM have previously been used in many successful applications of person object detection. Features from the images were extracted using the HOG approach.

The final model performed extremely poorly when evaluated using the same metrics as the SH17 paper. From the evaluation, it was observed that the model's predictions would fall below the 0.5 IoU threshold, which is the minimum threshold for a detection to be considered a true positive. Thus more work is needed to improve the model's performance which could be accomplished with improved prediction window merging and hyperparameter tuning.

AI Use Disclaimer

Claude 3.5 Sonnet was used in code generation and LaTeX formatting for this paper. Code was edited and verified by the author.

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[2] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," 2016. [Online]. Available: https://arxiv.org/abs/1506.02640

[3] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," 2016. [Online]. Available: https://arxiv.org/abs/1506.01497

[4] H. M. Ahmad and A. Rahimi, "Sh17: A dataset for human safety and personal protective equipment detection in manufacturing industry," 2024. [Online]. Available: https://arxiv.org/abs/2407.04590

[5] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, 2005, pp. 886–893 vol. 1.

[6] V. Janapa Reddi, *Machine Learning Systems: Principles and Practices of Engineering Artificially Intelligent Systems*. Harvard University, 2024, online open-source book, Last updated: November 19, 2024. [Online]. Available: https://harvard-edge.github.io/cs249r_book/

[7] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[8] C. Papageorgiou and T. Poggio, "A trainable system for object detection," *International Journal of Computer Vision*, vol. 38, no. 1, pp. 15–33, 2000.

[9] R. Ronfard, C. Schmid, and B. Triggs, "Learning to parse pictures of people," in *Proceedings of the 7th European Conference on Computer Vision*, vol. IV, Copenhagen, Denmark, 2002, pp. 700–714.

[10] Viola, Jones, and Snow, "Detecting pedestrians using patterns of motion and appearance," in *Proceedings Ninth IEEE International Conference on Computer Vision*, 2003, pp. 734–741 vol.2.

[11] S. Plvs, "Object detection via hog-svm," 2023. [Online]. Available: https://github.com/SamPlvs/Object-detection-via-HOG-SVM

[12] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, "scikit-image: Image processing in Python," *PeerJ*, vol. 2, p. e453, 2014.

[13] sascha, "Answer to "scikit-learn: what is the difference between svc and sgd?"," https://stackoverflow.com/a/48022399, 2017, stack Overflow answer posted on December 29, 2017. [Online]. Available: https://stackoverflow.com/a/48022399

[14] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[15] Scikit Learn Contributors, "Choosing the right estimator - machine learning map," https://scikit-learn.org/1.4/tutorial/machine_learning_map/index.html, 2024.

[16] T.-Y. Lin, M. Maire, S. Belongie, L. Bourdev, R. Girshick, J. Hays, P. Perona, D. Ramanan, C. L. Zitnick, and P. Dollár, "Coco - common objects in context," 2015, accessed: 2024-12-06. [Online]. Available: https://cocodataset.org/