UNIVERSITY of SOUTH FLORIDA

Hardware Accelerators for Machine Learning Department Of Computer Science and Engineering

Course Instructor: Dr.Dayane A.Reis
CIS 4930/6930

Project Title(Group 7)

Accelerating Pedestrian Detection with CPU Vs GPU Architecture

CPU(TEAM A)

Mythreye Pasula (U88527486)

Yaswanth Sai Ram Pavan Jami (U22367338)

Chaitanya Sai Dangeti (U77695453)

Vineeth Reddy Betham (U36652271)

GPU(TEAM B)

Sree Vardhani Iragavarapu (U13255029)

Lakshmi Venkata Chaitanya Raju Gedela(U42289643)

Kumar Ganesh Chowdary Naidu(U05647359)

Akshay Reddy Sudini(U44174638)

Description of Methods

Problem Description

The project aims to keep up with the existing issue of the applicability of advanced driver assistance systems (ADAS) that will enhance safety by incorporating pedestrian detection at the crossing points. Global deaths from road accidents are primarily due to vulnerable road users (VRUs), which are cyclists, pedestrians, and motorcyclists who commonly face the risk of traffic (Berberich & Doll, 2014). Our strategy is based on a multisensor system within the RSU (roadside unit), which warns vehicles that there are VRUs around. Our system's performance depends on the type of neural network employed.

Dataset: We want to run the pedestrian identification model on the central processing unit (CPU) using the INRIA benchmark dataset for training and testing purposes. The fact that the dataset contains a wide variety of photos, including those that were taken at crossings, makes it appropriate for use in situations that occur in the real world.

Implementation and the architecture in the CPU

Training Phase:

1. Feature Extraction - HOG features will be computed from images in the INRIA training dataset on the CPU. This involves dividing images into small cells, and creating histograms of gradient orientations.
2. Model Training: The SVM classifier will be trained on CPU processor using the HOG features extracted from the INRIA dataset. The SVM training process involves finding the optimal hyperplane to separate pedestrian and non-pedestrain examples.

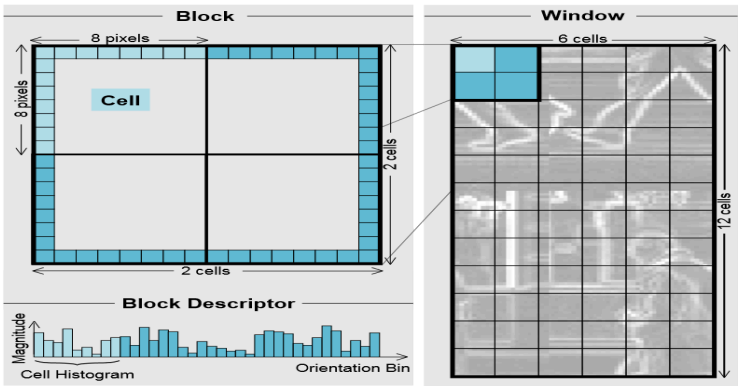
3. Validation: The trained SVM model will be validated on a separate part of the INRIA dataset to ensure its effectiveness and tune hyperparameters if necessary.

Inference Phase:

1. Preprocessing: For new images in the inference phase, similar preprocessing steps will be applied to resize and normalize the images on the CPU.
2. Feature Extraction: HOG feature will be extracted from the preprocessed images on the CPU, similar to the training phase.
3. Pedestrian Detection: The trained SVM model will be applied to extracted HOG features from new images for pedestrian detection on the CPU during the inference phase.
4. Post-processing : Techniques like non-maxima suppression may be applied to refine the detection results and eliminate redundant or overlapping detections on the CPU.

Detailed Method Description:

HOG-based Hypothesis Verification**:**The HOG descriptor and SVM classification technique, presented by Dalal and Triggs (2005), are essential to hypothesis verification, which separates genuine discoveries from spurious ones. We use a local spatial sliding window search at the hypothesis region of interest (ROI) instead of an exhaustive scan across all positions and scales. A kernel SVM classifier trained via supervised learning is used to compute and evaluate HOG features. This approach is anticipated to produce a peak with lower responses surrounding the proper object position. As a non-maxima suppression technique, we use mean shift to combine several neighbouring detections into a single ROI. The HOG descriptor scheme is shown in the following image. Block descriptors are produced by concatenating cell orientation histograms and weighting them according to the gradient magnitude.



Kernel SVM's Structure and methodology used

HOG descriptors, kernel Support Vector Machines (SVMs) are machine learning models designed to put image patches in pre-defined classes, either pedestrian or non- pedestrian (Kyrkou et al 2018). Since they can learn nonlinear decision boundaries in a high-dimensional feature space, these SVMs are helpful and important. By utilizing a SVM with a 2-norm soft margin kernel and a classification function f(x), our approach can perform the classification task.

Ai represents the positive Lagrange multipliers, the bias by b0, the class label by yi, the support vectors by si, the HOG instance by x, and the Gaussian kernel function by K(s,x). Grid search is used to find the kernel parameter, γ, and the slack variable weighting factor, C. The step-by-step process has been described thoroughly by Kyrkou (2018).

Machine Learning Problem solved:In the case of the CPU the the execution in only done sequential,whereas when we employ the GPU architecture we can use the Parallelization of the GPU's for faster result, using th R-cnn can be used for better feature extraction.

Development Environment and Frameworks Used for the ML model:Tensor flow and Pytorch(for GPU Implementation)

Implementation and the architecture in the GPU

In the GPU implementation we use the public dataset INRIA pedestrian Dataset, we are using the following datasets as it contains a wide range of images of pedestrian crossings and the kind the resolution that the dataset provides for the easier classification.

1.Preprocess the dataset INRIA(we can also collect the data from the VR and FIR wavelength data through the real time analysis). The preprocess steps will be implemented to optimize the classification results.

2.Extract the important features from the data such as the edges, shape etc. This will be done when the images are sent via the convolution phase of R-CNN in the form of feature maps.

The parameters such as image size, format of the annotations are adjusted through the proper training of the dataset.

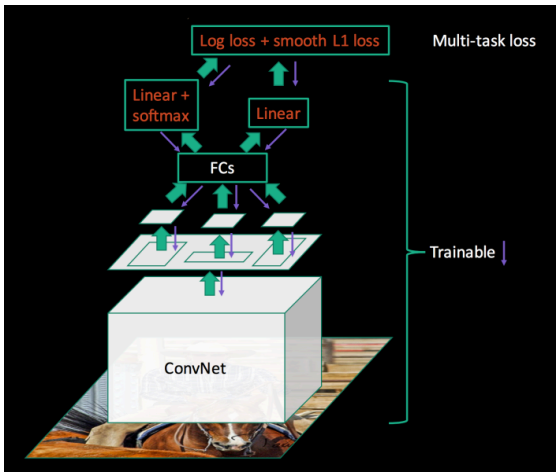
We are trying to use the R-CNN for our neural network, which would be much more suitable for the image type data, we can use the R-CNN for much better accuracies than the other neural networks whereas we have to decide on the tradeoff of the speed of the model.

The reason we use R_CNN is to utilize the aspect of GPU, which is parallel processing. The matrix multiplications involved and other such operations would be hastened with GPU, due to parallel computing. The main drawback of R_CNN, the speed of computing, will be hastened when using GPU, rather than CPU which would lead to sequential computing

3. Use the CNN to learn the features from the data, which involves backpropagation and optimizing the network weights, after that feed the extracted features from the CNN to the kernel SVM, which separates the non VRU's after that the hyperparameters(batch size and all) are modified in accordance to the performance.

Feature extraction using the CNN model after feeding the extracted features to a fully connected layer. The no of hidden layers in the Fully Connected Layer phase, will be decided to optimize performance result .

We will be using pre-trained R_CNN, hence, we would be implementing transfer learning in this context. For the transfer learning, we would be considering the cases of either freezing the weight learning of initial conv layers or only unfreezing the fully connected layers during the training phase, to observe and balance the trade off between speed and accuracy for the case of GPU.The total flow for using R_CNN can be observed in the diagram below:In the Inference we can either use the SVM based on the Model Performance we are looking to modify it.



Metrics for CPU and GPU Performance

Accuracy - Measure the accuracy of the pedestrian detection model on the CPU/GPU by comparing predicted labels with ground truth labels

False Alarm rate: Evaluate false alarm rate to assess the model's ability to avoid misclassifying non-pedestrian objects,

Processing Speed: Measure the time taken for the entire detection process, including feature extraction and classification, to ensure real-time performance.

References:

Afifi, S. M., GholamHosseini, H., & Poopak, S. (2015). Hardware implementations of SVM on FPGA: A state-of-the-art review of current practice.

Berberich, M., & Doll, K. (2014). Highly flexible FPGA architecture of a support vector machine. In *MPC workshop* (pp. 25-32).

Dalal, N., & Triggs, B. (2005). Histograms of oriented for human detection. In *Proc. IEEE Computer Society Conf. on Computer Vision and Pattern Recognition* (pp. 886–893).

Kyrkou, C., Theocharides, T., Bouganis, C. S., & Polycarpou, M. (2018). Boosting the hardware efficiency of cascade support vector machines for embedded classification applications. *International Journal of Parallel Programming*, *46*(6), 1220–1246.

Maggiani, L., Bourrasset, C., Quinton, J. C., Berry, F., & Sérot, J. (2018). Bio-inspired heterogeneous architecture for real-time pedestrian detection applications. *Journal of Real-Time Image Processing*, *14*, 535-548.

Weimer, D., Köhler, S., Hellert, C., Doll, K., Brunsmann, U., & Krzikalla, R. (2011, June). GPU architecture for stationary multisensor pedestrian detection at smart intersections. In *2011 IEEE Intelligent Vehicles Symposium (IV)* (pp. 89-94). IEEE.

Wasala, M., & Kryjak, T. (2022, June). Real-time HOG+ SVM-based object detection using SoC FPGA for a UHD video stream. In *2022 11th Mediterranean Conference on Embedded Computing (MECO)* (pp. 1-6). IEEE.

Zhou, Y., Chen, Z., & Huang, X. (2015, May). A pipeline architecture for traffic sign classification on an FPGA. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)* (pp. 950–953). IEEE.