ADAPTIVE ALGORITHM SELECTION, WITH APPLICATIONS IN PEDESTRIAN DETECTION

Shu Zhang *

SONY Electronics Inc.
San Jose, CA, 95112, USA

Qi Zhu, Amit Roy-Chowdhury

University of California, Riverside
Riverside, CA, 92521, USA

ABSTRACT

Computer vision algorithms are known to be extremely sensitive to the environmental conditions in which the data is captured, e.g., lighting conditions and target density. Tuning of parameters or choosing a completely new algorithm is often needed to achieve a certain performance level. In this paper, we focus on this problem and propose a framework to automatically choose the "best" algorithm-parameter combination (often referred to as the best algorithm for simplicity in this paper) for a certain input data. This necessitates developing a mechanism to switch among different algorithms and parameters as the nature of the input video changes. Specifically, our proposed algorithm calculates a similarity function between a test video segment and a training video segment. Similarity between training and test dataset indicates the same algorithm can be applied to both of them. We design a cost function with this similarity measure and a constraint on the number of switches. In the experiments, we apply our algorithm to the problem of pedestrian detection. We show how to adaptively select among 7 algorithm-parameter combinations and provide promising results on 3 publicly available datasets.

Index Terms— Algorithm selection, adaptation, pedestrian detection

1. INTRODUCTION

Numerous algorithms have been developed for various computer vision applications. Also, many public datasets have been released to help researchers evaluate their algorithms. For instance, the datasets of CAVIAR [1] and TUD-Brussels [2] have been commonly used in the area of tracking. In many cases, an algorithm is able to achieve good performance on some datasets, while failing to beat other algorithms on other datasets. Besides, it is interesting to see that some algorithms perform well on parts of a dataset, but cannot achieve good results on some other parts. This is because algorithms are sensitive to the environmental conditions in each dataset or parts thereof. These observations raise an important question: can we automatically select the best algorithm to be used for a certain application domain?

The goal of this paper is the following. Given a set of existing computer vision algorithms and their parameters for a certain problem, can we automatically select the "best" algorithm-parameter combination (referred to as algorithm in the rest of the paper) for that problem domain? The answer, in most cases, will not lie in one specific algorithm but on an *adaptive mechanism for selecting among the set of algorithms*, since the conditions in the video will likely change over time. Conditions that could trigger the switch include the lighting in the video, the number of targets in the scene, the resolution of the targets, and so on - factors that are known to affect the performance of vision algorithms. In the experiments, we focus on the problem of pedestrian detection, since it is a fundamental low-level task that is crucial for higher-level ones such as recognition, and known to be sensitive to environmental factors. However, the proposed algorithm is a general solution that could be applicable to many other computer vision problems.

As illustrated in Fig. 1, the results of three pedestrian detectors are affected by the frequently-changing scales of objects, number of objects, and illumination condition. Each row shows representative image frames from a video recorded at different times of a day, and each column denotes the person detection results. Each pedestrian detector achieves desired results on some image frames. For instance, in the first frame, the desired detectors are 2 and 3, while in the second frame, the desired detectors change to 1 and 2. However, in the fourth frame, the illumination changes and the number of persons increases, and detector 3 achieves the best performance. Similar observations can be made in the rest of the frames. An example of an ideal detector is shown with green arrows. This example shows the importance of developing an adaptive switching mechanism among multiple detectors to minimize the detection error for each scenario.

1.1. Overview and Contributions

Motivated by Fig. 1, we propose a switching algorithm which adaptively selects the best available algorithm for each scenario. Our input consists of a set of existing algorithms. We

*Shu Zhang performed the work while at University of California, Riverside.

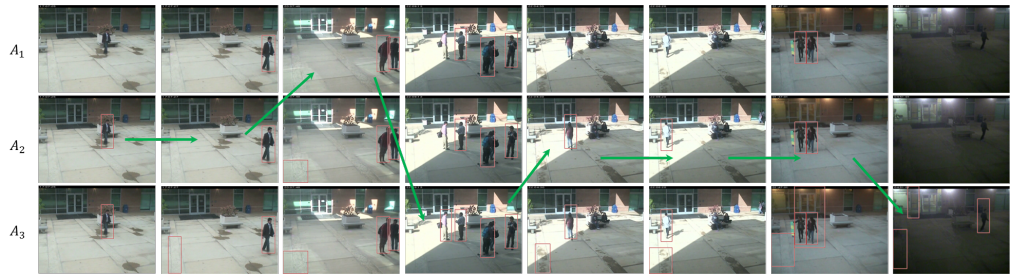

Fig. 1: Illustrations of pedestrian detection results by three algorithms, $\{A_1, A_2, A_3\}$, in a video sequence within a day. Each row denotes results of an algorithm,and each column represents an image frame. An example of the adaptive detector selection is shown with green arrows, which is $A_2 \rightarrow A_2 \rightarrow A_1 \rightarrow A_3 \rightarrow A_2 \rightarrow A_2 \rightarrow A_2 \rightarrow A_3$.

also have datasets on which these algorithms have been tested. Each available algorithm has image frames as inputs and performance results as outputs.

There are two operating phases in our proposed framework: *training* and *test*. In the training phase, every algorithm is applied to a unique scenario in the training dataset. The algorithms that provide the smallest detection error are then labeled as the best algorithms for this training scenario. In the test phase, we segment every video sequence in the test dataset into time windows. The goal of the proposed algorithm selection process is to choose the "best" algorithm for each video segment. The training segment that the test video is closest to determines the vision algorithm to be used (and hence the switching instant), subject to an additional constraint that the number of algorithm switches should be limited. We show how this can be framed as an optimization problem and how the global optimum is computed. We demonstrate the efficacy of the proposed approach on multiple pedestrian detection datasets. We apply 7 algorithm-parameter combinations on 3 public datasets [3, 4, 5]. It is proved that the proposed approach performs better than any single algorithm.

1.2. Related Work

Algorithm selection has been studied in recent years in a few works. In [6], image segmentation algorithms are selected on different images. Features are learned by SVM and the performance of each algorithm is mapped to a ranking vector based on the feature correlations. In [7], the goal is to segment pixels in an image into different regions that are suitable to different algorithms, where a random forest classifier is used.

Our work is different from these two approaches. We consider the problem of automatically switching the algorithm based on the scene similarity between a test time window and all the time windows in the training dataset. Our proposed algorithm does not learn which specific feature to be used for a dataset, and does not need manual analysis of the feature-performance correspondence. This is more general than [7], where the effects of the features on the training dataset are manually analyzed to obtain the correlations between features and algorithms, *e.g.* which feature has an impact on a specific data. The methodology of domain adaptation that we use finds the underlying correspondences among features. Moreover, we propose a global optimization methodology that automatically switches to the best algorithm but with a penalty on the number of switches. In [8], budget constraints are taken into account as the leverage rule between different algorithms in the context of handwriting recognition. The budget constraint can be added as an additional one in our proposed framework.

2. METHODOLOGY

2.1. Problem Description

We assume the availability of a number of algorithms for the problem, and the performance of every algorithm on some datasets is known. Our goal is to answer the following questions: 1) for every part of an unknown dataset, is it possible to automatically select an algorithm among all available algorithms that achieves the best result? 2) for the entire unknown dataset, what is the best strategy to switch between algorithms?

In our problem, the input is the set of K available algorithms $\mathcal{A} = \{A_1, \cdots, A_K\}$ and the dataset on which they are evaluated. We call this the training dataset $\mathcal{T} = \{T_1, \cdots, T_M\}$, where T_i is the i-th unique scenario in $\mathcal{T}$. We apply every algorithm A_k on each T_i and select the algorithm(s) that perform the best. The algorithm selection for T_i is denoted with a label Y_i. The unknown dataset is called the test dataset $\mathcal{R}$. In $\mathcal{R}$, we assume that there are N time windows in total. Every time window of images is denoted as $R_j, j = 1, \cdots, N$. The selection of algorithms for R_j is represented by L_j. Given the pairs (T_i, Y_i), the problem is how to find the unknown label L_j for each R_j that is in $\mathcal{R}$.

2.2. Solution Overview

The overview of our solution is shown in Fig. 2. In the training dataset $\mathcal{T}$, we obtain the label for each time window, which indicates the algorithm selection on T_i. In the

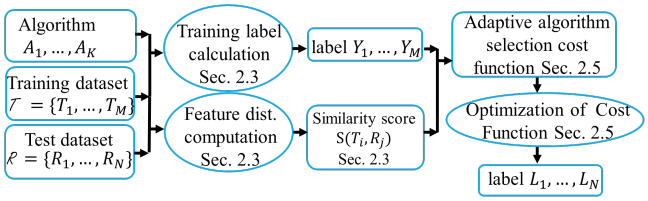

Fig. 2: Overall Methodology.

test dataset $\mathcal{R}$, the output of the algorithm, the label set $\mathcal{L} = \{L_1, \cdots, L_N\}$, is obtained by a cost function that is able to automatically select the best algorithm on a specific time window R_j. In this cost function, there are three terms that need to be obtained. The first term is the similarity score $S(T_i, R_j)$. The second term is the labeling function that is defined in Sec. 2.5. The third term is a regularization term that aims to reduce the total number of algorithm switches. This regularization term depends on two consecutive time windows.

2.3. Training Label Calculation

In the training dataset $\mathcal{T}$, we exhaustively apply every algorithm A_k on each time window T_i. The algorithm selection for T_i is denoted as the label $Y_i = \mathcal{Y}(T_i)$, where $\mathcal{Y}$ is a mapping function that satisfies $\mathcal{Y}(T_i) \rightarrow 2^{\mathcal{A}}$. The implementation of $\mathcal{Y}$ is provided in the experimental section. It is noted that Y_i can be either a single value or a vector because it is likely that multiple algorithms achieve the same performance for T_i. For instance, in the example in Fig. 1, the first and second algorithms obtain the same performance on the second frame.

2.4. Similarity Score Calculation

In the test dataset $\mathcal{R}$, every video sequence/image set is segmented into a sequence of non-overlapping time windows R_j. Our goal is to find an algorithm L_j for each R_j.

The criteria of algorithm selection L_j depends on the similarity between R_j and the scenarios in $\mathcal{T}$. Ideally, we want to compare R_j with every training scenario, and find the most similar training scenario to R_j. This training scenario shares similar characteristics with R_j, and thus the best algorithm(s) for it (identified during the training stage) should also provide the best performance for R_j. In practice, for overhead and stability consideration, we may not always choose the most similar training scenario, as explained later.

The similarity between R_j and T_i, denoted by $S(T_i, R_j)$, is calculated as

$$S(T_i, R_j) = e^{-d(T_i, R_j)} \tag{1}$$

where $d(T_i, R_j)$ represents the feature distance between T_i and R_j. It can be computed by the method in [9], where the key idea is to use the domain adaptation to project both the training data and each video segment of the test data into subspaces to learn domain-invariant features.

2.5. Adaptive Algorithm Selection

The ultimate goal of our proposed approach is to automatically select an algorithm L_j for a time window of images R_j in the test data. We formulate the selection of algorithms in the training dataset $\mathcal{R}$ as

$$\{L_1, \cdots, L_N\} = \max_k \{\sum_i \mathbb{I}(A_k, T_i) S(T_i, R_j) + \alpha h(j, j-1)\}, \tag{2}$$

where α is an coefficient, and $\mathbb{I}$ is a labeling function that is defined as

$$\mathbb{I}(A_k, T_i) = \begin{cases} 1, & \text{if } A_k \in Y_i, \\ 0, & \text{otherwise.} \end{cases} \tag{3}$$

$S(T_i, R_j)$ denotes the similarity between T_i and R_j. We use a matrix X to represent the output of $\mathbb{I}(A_k, T_i) S(T_i, R_j)$ for every j and k, where $X \in \mathbb{R}^{K \times N}$. We define k^j as the selected row for the column j of X, and define $f(k^j)$ as the label for the k^j-th row. In other words, $L_j \in f(k^j)$.

The second term $h(j, j-1)$ is a regularization term, which indicates the label similarity between two consecutive time windows R_{j-1} and R_j. We define $h(j, j-1)$ as

$$h(j, j-1) = \frac{len(f(k^j) \cap f(k^{j-1}))}{len(f(k^{j-1}))} \tag{4}$$

where $f(k^j) \cap f(k^{j-1})$ denotes the intersection between two vectors $f(k^j)$ and $f(k^{j-1})$, and len represents the length of a vector. $h(j, j-1)$ represents that if the selected rows of $j-1$ and j-th columns share similar algorithms, the regularization term $h(j, j-1)$ is high, leading to high overall cost.

2.6. Overall Optimization

Eq. 2 can be rewritten as

$$\{L_1, \cdots, L_N\} = \max_k \{X_{k,j} + \alpha h(j, j-1)\} \tag{5}$$

We solve this equation by using the dynamic programming methodology [10], where the global optimal solution is guaranteed to be obtained.

3. EXPERIMENTS

3.1. Experimental Setup

We demonstrate the effectiveness of our approach through a set of experiments. There are 7 state-of-the-art available algorithms: Cascades [11], HOG [3], ACF [12], PartBased [13], KSVM [14], MultiFtr [15], and HOG-LBP [16]. The public datasets that are used in this paper are: INRIA [3], ETHMS [4], and TUD-Brussels [5]. We also compare our results with the algorithm selection methodology in [7], named as RanFor. We extract four different features that are used for distance calculation in the experiments: HOG features [3], SIFT

	10^{-1}	10^{0}	Ave
HOG	0.498	0.231	0.333
MultiFtr	0.283	0.109	0.161
KSVM	0.445	0.221	0.302
HOG-LBP	0.426	0.190	0.272
Cascades	0.244	-	0.182
PartBased	0.538	0.199	0.302
ACF	0.173	0.078	0.103
RanFor	0.379	0.197	0.257
Proposed	0.238	0.115	0.152

(a) INRIA-Test

	10^{-1}	10^{0}	Ave
HOG	0.679	0.386	0.517
MultiFtr	0.651	0.378	0.499
KSVM	0.771	0.456	0.598
HOG-LBP	0.569	0.336	0.437
Cascades	0.653	0.513	0.569
PartBased	0.710	0.540	0.609
ACF	0.665	0.504	0.569
RanFor	0.674	0.483	0.565
Proposed	0.633	0.364	0.483

(b) ETHMS

	10^{-1}	10^{0}	Ave
HOG	0.829	0.563	0.675
MultiFtr	0.748	0.555	0.636
KSVM	0.877	0.583	0.714
HOG-LBP	0.867	0.616	0.714
Cascades	0.892	0.756	0.810
PartBased	0.911	0.814	0.851
ACF	0.865	0.763	0.804
RanFor	0.879	0.724	0.789
Proposed	0.744	0.553	0.635

(c) TUD-Brussels

Table 1: The miss rate at FPPI = 10^{-1}, 10^{0} and the average miss rates between 10^{-1} and 10^{0}. The best results and the second best results are highlighted.

features [17], gradient features [18], and texture features [19]. We group all the training data into 20 unique scenarios for the training dataset $\mathcal{T}$. The results are plotted by false-positives-per-image (FPPI) vs. miss rate.

We also investigate the effect of algorithm switches in every dataset. In the training phase, we estimate the classification threshold of each algorithm that leads to FPPI $= 1$. In the test dataset, we keep the detections with the scores greater than this threshold for each algorithm. We evaluate the detections for each time window, where the number of miss detections is used as the evaluation metrics. We set the parameter α in Eq. 2 to 0.3. We have conducted a set of experiments and have observed that for $\alpha < 1$ the results are similar (when $\alpha > 1$, the performance of the proposed algorithm decreases significantly as it is unlikely to switch between algorithms even when it may provide significant improvement).

3.2. Performance Evaluation

In this section, we apply the proposed adaptive selection strategy to public datasets. In Fig. 3, we show the average FPPI vs. miss rate curves of the three datasets (INRIA-Test, ETHMS, and TUD-Brussels), where the red curves represent the results of the proposed method. *It is shown that the average performance of the proposed approach across all datasets is the best*. This result proves the effectiveness of developing an adaptive algorithm selection scheme.

In Table. 1, we report the miss rates of each algorithm at FPPI = 10^{-1} and 10^{0}. We also show the average miss rates between FPPI = 10^{-1} and FPPI = 10^{0}. In INRIA-Test dataset, the proposed method has low miss rates at both FPPI = 10^{-1} and FPPI = 10^{0}. In ETHMS dataset, HOG-LBP performs the best, but our proposed algorithm selection strategy obtains similar results. In TUD-Brussels, our proposed approach performs very similar to MultiFtr. Both ETHMS and TUD-Brussels datasets have low feature variances within them. The performance of our proposed switching method is similar to the best existing algorithm. This shows that the proposed method can automatically choose one particular algorithm that would perform the best.

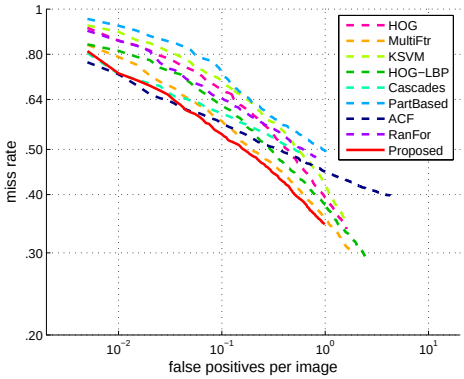

Fig. 3: The average performance of each algorithm on all the datasets. The red curve in each sub-figure represents the performance of the proposed approach.

4. CONCLUSION

In this paper, we present a novel approach to adaptively select the "best" algorithm among existing algorithms for each segment of a video sequence. We calculate the feature similarity on the manifold that is shared between training and test data. The more similar they are, the higher the possibility that the same algorithm can perform well on both of them. We propose a cost function with an additional constraint on the number of algorithm switches, where the global optimal solution is guaranteed to be obtained. We show the efficacy of the proposed method on the application of pedestrian detection, where 7 algorithm-parameter combinations are considered. Our promising results have shown that adaptively selecting algorithms can outperform a fixed algorithm selection. In the future, we would like to work on an end-to-end algorithm selection algorithm, *e.g.* deep neural network, which does not need hand-crafted features that are used in the current method.

Acknowledgment

This work was partially supported under NSF grant CPS-1544969.

5. REFERENCES

[1] "CAVIAR dataset," http://homepages.inf.ed.ac.uk/rbf/CAVIARDATA1/.

[2] M. Andriluka, S. Roth, and B. Schiele, "People-tracking-by-detection and people-detection-by-tracking," in *CVPR*, 2008.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *CVPR*, 2005.

[4] A. Ess, B. Leibe, K. Schindler, and L. V. Gool, "A mobile vision system for robust multi-person tracking," in *CVPR*, 2008.

[5] C. Wojek, S. Walk, and B. Schiele, "Multi-cue onboard pedestrian detection," in *CVPR*, 2009.

[6] X. Yong, D. Feng, Z. Rongchun, and M. Petrou, "Learning-based algorithm selection for image segmentation," *Pattern Recognition Letters*, vol. 26, no. 8, pp. 1059–1068, 2005.

[7] O. M. Aodha, G. J. Brostow, and M. Pollefeys, "Segmenting video into classes of algorithm-suitability," in *CVPR*, 2010.

[8] J. Wang, T. Bolukbasi, K. Trapeznikov, and V. Saligrama, "Model selection by linear programming," in *ECCV*, 2014.

[9] B. Gong, Y. Shi, F. Sha, and K. Grauman, "Geodesic flow kernel for unsupervised domain adaptation," in *CVPR*, 2012.

[10] T. H. Cormen, C. Stein, R. L. Rivest, and C. E. Leiserson, *Introduction to algorithms*, McGraw-Hill Higher Education, 2 edition, 2001.

[11] H. Cevikalp and B. Triggs, "Efficient object detection using cascades of nearest convex model classifiers," in *CVPR*, 2012.

[12] P. Dollár, S. Belongie R. Appel, and P. Perona, "Fast feature pyramids for object detection," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 36, no. 8, pp. 1532–1545, 2014.

[13] P. F. Felzenszwalb, R.B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 32, no. 9, pp. 1627–1645, 2010.

[14] S. Maji, A. Berg, and J. Malik, "Classification using intersection kernel svms is efficient," in *CVPR*, 2008.

[15] S. Walk, N. Majer, K. Schindler, and B. Schiele, "New features and insights for pedestrian detection," in *CVPR*, 2010.

[16] X. Wang and T. X. Han, "An hog-lbp human detector with partial occlusion handling," in *ICCV*, 2009.

[17] D. Lowe, "Object recognition from local scale-invariant features," in *ICCV*, 1999.

[18] A. Oliva and A. Torralba, "Modeling the shape of the scene: a holistic representation of the spatial envelope," *International Journal on Computer Vision*, vol. 42, no. 3, pp. 145–175, 2001.

[19] "GLCM," http://www.fp.ucalgary.ca/mhallbey/tutorial.htm.