1

Vehicle Counting in Crowded traffic pictures

Xi Huang
Yangyan Zhou

Abstract—**We propose a supervised learning framework for visual object counting tasks, such as estimating the number of vehicles in surveillance video frames. We focus on the practically-attractive case when the training images are annotated with dots (one dot per object). Our goal is to accurately estimate the count. We use the database TRANCOS, a novel database for extremely overlapping vehicle counting. We evade the hard task of learning to detect and localize individual object instances. Instead, we cast the problem as that of estimating an image density whose integral over any image region gives the count of objects within that region. Learning to infer such density can be formulated as a minimization of a regularized risk quadratic cost function. We introduce several ways of training function, which is well-suited for such learning, and at the same time can be computed efficiently via a maximum subarray algorithm. The learning can then be posed as a convex quadratic program solvable with cutting-plane optimization**

I. INTRODUCTION

The counting problem is the estimation of the number of objects in a still image or video frame. It arises in many real-world applications including cell counting in microscopic images, monitoring crowds in surveillance systems, and performing wildlife census or counting the number of trees in an aerial image of a forest. We take a supervised learning approach to this problem, and so require a set of training images with annotation. In this paper, we make use of the dataset e TRANCOS dataset offers is considerable, in sharp contrast to the rest of datasets. In this dataset the annotation of which is to specify the object position by putting a single dot on each object instance in each image. Figure 1 gives examples of traffic picture and the dotted annotation we use. Dotting (pointing) is the natural way to count objects for humans, at least when the number of objects is large. Overall, it should be noted that dotted annotation is less labor intensive than the bounding-box annotation, let alone pixel-accurate annotation, traditionally used by the supervised methods in the computer vision community [15]. Therefore, the dotted annotation represents an interesting and, perhaps, under-investigated case. This paper develops a simple and general discriminative learning-based framework for counting objects in images. Similar to global regression methods (see below), it also evades the hard problem of detecting all object instances in the images. However, unlike such methods, the approach also takes full and extensive use of the spatial information contained in the dotted supervision. The high-level idea of our approach is extremely simple: given an image I, our goal is to recover a density function F as a real function of pixels in this image.

the car detection task in the PASCAL VOC [1] benchmark has reached an Average Precision above 50%. Most of the winner methods in this dataset are based on detectors using Histogram of Oriented Gradients (HOG) features [5] and sliding window strategies. Figure 2 (a) depicts the results of a HOG based detector [6] in one of our images. It seems clear that novel solutions need to be explored. We show in this paper that this type of strategy, named counting by detection, does not improve the precision reported by counting by regression techniques, e.g. [7,8].

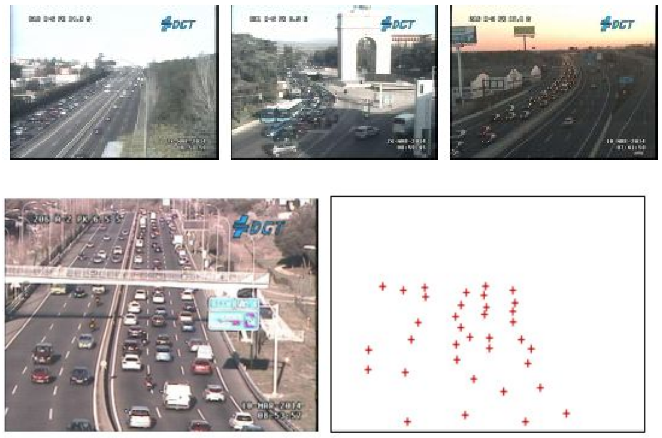


Figure1.TRaffic ANd COngestionS (TRANCOS) database images. The red cross picture are the ground truth label of the vehicles for the traffic picture on the left.

In this paper, we firstly tested the previous work Ricardo Guerrero-Gomez-Olmedo etc. has done by SIFT to do feature extraction and the counting by regression strategies to train model and their evaluation methods using the Grid Average Mean absolute Error (GAME) metric . Secondly, we used the mature method HOG : a counting by detection approach based on the HOG detector to do the feature extraction and used the supervised learning approach to train our model by SVM and KNN and done the evaluation.

Given the estimate F of the density function and the query about the number of objects in the entire image I, the number of objects in the image is estimated by integrating F over the entire I. Furthermore, integrating the density over an image subregion S ⊂ I gives an estimate of the count of objects in that subregion. Our approach assumes that each pixel p in an image is represented by a feature vector xp and models the

1

density function as a linear transformation of xp: F(p) = w T xp. Given a set of training images, the parameter vector w is learnt in the regularized risk framework, so that the density function estimates for the training images matches the ground truth densities inferred from the user annotations (under regularization on w). The key conceptual difficulty with the density function is the discrete nature of both image observations (pixel grid) and, in particular, the user training annotation (sparse set of dots). As a result, while it is easy to reason about average densities over the extended image regions (e.g. the whole image), the notion of density is not well-defined at a pixel level. Thus, given a set of dotted annotation there is no trivial answer to the question: what should be the ground truth density for this training example. Consequently, this local ambiguity also renders standard pixel-based distances between density functions inappropriate for the regularized risk framework.

2. Related work

2.1 Training

There are works for vehicle detection in conditions of relatively high traffic in highways, and , the vehicle counting problem in traffic congestion images has not been previously systematically studied. Here are the most strategy they used:Using SIFT to do feature extraction. And training models using counting by detection: This assumes the use of a visual object detector, that localizes individual object instances in the image. Given the localizations of all instances, counting becomes trivial. However, object detection is very far from being solved [5], especially for overlapping instances. In particular, most current object detectors operate in two stages: first producing a real-valued confidence map; and second, given such a map, a further thresholding and non-maximum suppression steps are needed to locate peaks corresponding to individual instances [12]. More generative approaches avoid nonmaximum suppression by reasoning about relations between object parts and instances [6], but they are still geared towards a situation with a small number of objects in images and require time-consuming inference. Alternatively, several methods assume that objects tend to be uniform and disconnected from each other by the distinct background color, so that it is possible to localize individual instances via a Monte-Carlo process [3], morphological analysis [5] or variational optimization [2]. Methods in these groups deliver accurate counts when their underlying assumptions are met but are not applicable in more challenging situations

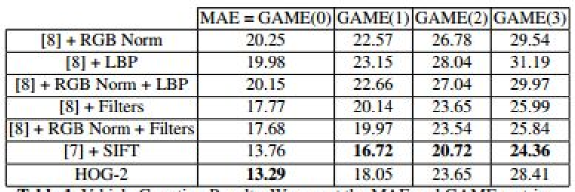
Counting by regression: These methods avoid solving the hard detection problem. Instead, a direct mapping from some global image characteristics (mainly histograms of various features) to the number of objects is learned. Such a standard regression problem can be addressed by a multitude of machine learning tools (e.g. neural networks [11, 17, 22]). This approach however has to discard any available information about the location of the objects (dots), using only its 1-dimensional statistics (total number) for learning. As a result, a large number of training images with the supplied counts needs to be provided during training. Finally, counting by segmentation methods [10, 28] can be regarded as hybrids of counting-by-detection and counting-by-regression approaches. They segment the objects into separate clusters and then regress from the global properties of each cluster to the overall number of objects in it

2.2 Evaluation

The MESA distance The distance D in (3) measures the mismatch between the ground truth and the estimated densities (the loss) and has a significant impact on the performance of the entire learning framework. There are two natural choices for D:

$$\mathcal{D}_{MESA}(F_1, F_2) = \max_{B \in \mathbf{B}} \left| \sum_{p \in B} F_1(p) - \sum_{p \in B} F_2(p) \right|$$

Here, $\mathbf{B}$ is the set of all box subarrays of I.



	MAE = GAME(0)	GAME(1)	GAME(2)	GAME(3)
[8] + RGB Norm	20.25	22.57	26.78	29.54
[8] + LBP	19.98	23.15	28.04	31.19
[8] + RGB Norm + LBP	20.15	22.66	27.04	29.97
[8] + Filters	17.77	20.14	23.65	25.99
[8] + RGB Norm + Filters	17.68	19.97	23.54	25.84
[7] + SIFT	13.76	**16.72**	**20.72**	**24.36**
HOG-2	**13.29**	18.05	23.65	28.41

Table 1. Vehicle Counting Results. We report the MAE and GAME metrics.

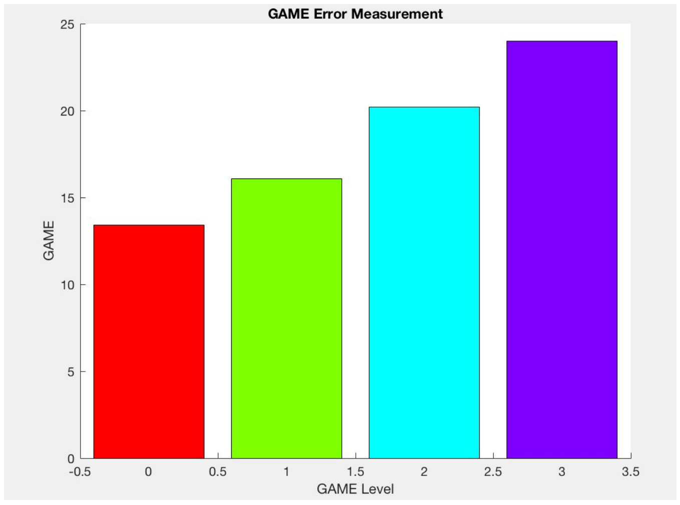


Figure 2. The results of the Vehicle counting results

Observe Table 1 GAME columns for a clear comparison of the different methods. First, as it was expected, the higher L the higher the error of the models. The best performance is now systematically obtained by [7]+SIFT. Furthermore, in Figure 4 we can see that it is the HOG-2 the one suffering the biggest increment of the error. Also, for L > 2 all the counting by regression models improve the results of HOG-2. Their results show that [7]+SIFT is the best approach counting and localizing the vehicles in the images

3 Experiments

3.1 Feature extraction

We are using HOG feature extraction ,here are the procedures how we do it.

HOG feature extraction algorithm

1. The color image is converted to grayscale
2. the luminance gradient is calculated at each pixel
3. To create a histogram of gradient orientations for each cell. The cell size is 48*64 pixel. So each picture is divided into 100 cells.
4. Normalization and Descriptor Blocks

Feature quantity becomes robust to changes in illumination

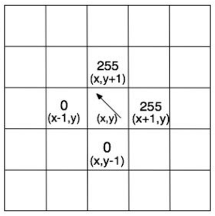
The luminance gradient is a vector with magnitude m and orientation θ represented by the change in the luminance.

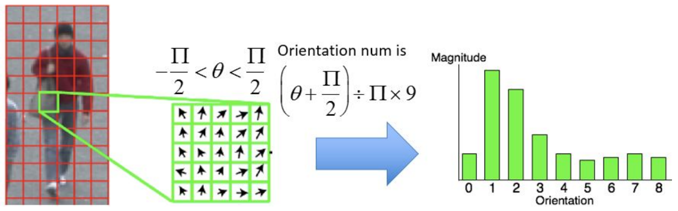
$$m(x,y)=\sqrt{(L(x+1,y)-L(x-1,y))^2+(L(x,y+1)-L(x,y-1))^2}$$

$$\theta(x,y)=\tan^{-1}\left(\frac{L(x,y+1)-L(x,y-1)}{L(x+1,y)-L(x-1,y)}\right)$$

$$-\frac{\Pi}{2}<\theta<\frac{\Pi}{2}$$

※L is the luminance value of pixel




$-\frac{\Pi}{2}<\theta<\frac{\Pi}{2}$
Orientation num is
$\left(\theta+\frac{\Pi}{2}\right)\div\Pi\times 9$
Magnitude
0 1 2 3 4 5 6 7 8
Orientation

3.2 Training

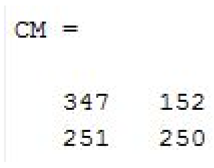
HOG can represent a rough shape of the object, so that it has been used for general object recognition, such as people or cars.In order to achieve the general object recognition, the classifier (eg SVM) is be used.

1. To teach the classifier, the correct image and the incorrect image.
2. Scan the classifier to determine whether there are vehicle in the detection window.

3.2.1 SVM

Train binary support vector machine classifier. We used fitcsvm trains or cross-validates a support vector machine (SVM) model for two-class (binary) classification on a low- through moderate-dimensional predictor data set. fitcsvm supports mapping the predictor data using kernel functions, and supports SMO, ISDA, or L1 soft-margin minimization via quadratic programming for objective-function minimization.

Here is the confusion matrix we got for our data:



```
CM =

   347   152
   251   250
```

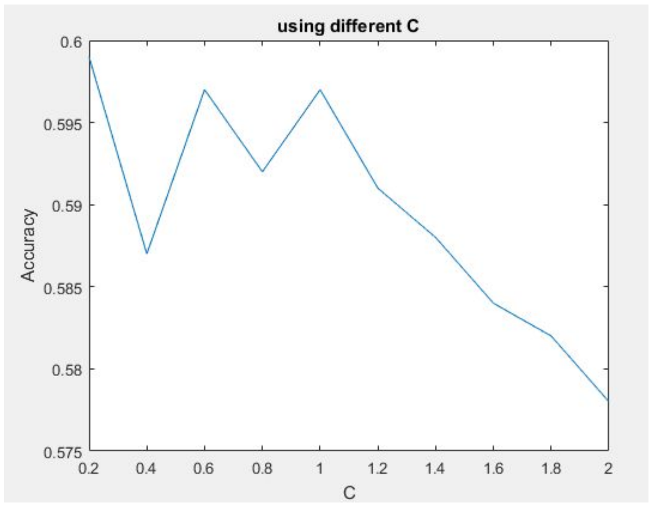
Accuracy= 0.5970;

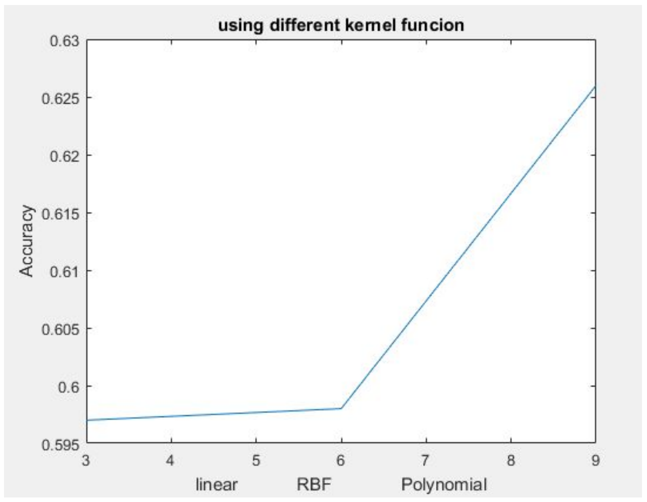
per-class recall rate:

arrR = 0.6954 0.4990

per-class prediction rate:

arrP = 0.5803 0.6219





3.2.2 KNN

In pattern recognition, the k-Nearest Neighbors algorithm (or k-NN for short) is a non-parametricmethod used for classification and regression.[1] In both cases, the input consists of the k closest training examples in the feature space. The output depends on whether k-NN is used for classification or regression:

With two clusters of 1 and 0, we use knn method to predict the result of testing sample, the error rate as follows:

```
error rate =    0.4470
Confusion matrix:
   498   100
   248   154
```

Recall rate: 0.8328/ 0.3831
Precision rate: 0.6676/ 0.6063

3.2.3 Logistic Regression

logistic regression, or logit regression, or logit model[1] is a regression model where the dependent variable (DV) is categorical. This article covers the case of binary dependent variables—that is, where it can take only two values, such as pass/fail, win/lose, alive/dead or healthy/sick. Cases with more than two categories are referred to as multinomial logistic regression, or, if the multiple categories are ordered, as ordinal logistic regression.

Since we only have two category of labels, we use Logistic regression to train the parameter of every single features. And the result as follows.

```
error rate  =  0. 512
Confusion mat:
   277   191
   321   211
```

Recall rate: 0.5919/0.3966
Precision rate: 0.5919/0.3966

3.2.4 Naive Bayes
Naive Bayes is a classification algorithm that applies density estimation to the data.

The algorithm leverages Bayes theorem, and (naively) assumes that the predictors are conditionally independent, given the class. Though the assumption is usually violated in practice, naive Bayes classifiers tend to yield posterior distributions that are robust to biased class density estimates, particularly where the posterior is 0.5 (the decision boundary) [1].

Naive Bayes classifiers assign observations to the most probable class (in other words, the maximum a posteriori decision rule). Explicitly, the algorithm:

Estimates the densities of the predictors within each class.
Models posterior probabilities according to Bayes rule. That is, for all k = 1,...,K,

$$\hat{P}(Y=k|X_1,..,X_P)=\frac{\pi(Y=k)\prod_{j=1}^{P}P(X_j|Y=k)}{\sum_{k=1}^{K}\pi(Y=k)\prod_{j=1}^{P}P(X_j|Y=k)},$$

1. where:

Y is the random variable corresponding to the class index of an observation.

$X_1,...,X_P$ are the random predictors of an observation.

2. Classifies an observation by estimating the posterior probability for each class, and then assigns the observation to the class yielding the maximum posterior probability.

Here we got the experiment results. Confusion matrix:

```
CM =

   348   250
   165   237
```

accuracy =0.5850

arrR =0.5819   0.5896

arrP =0.6784   0.4867

Assess the in-sample performance of Model by estimating the misclassification error.

isGenRate =0.2830

The in-sample misclassification rate is 28.3%.

If you specify a character vector, then the software models all the features using that distribution. If you specify a 1-by-P cell array of character vectors, then the software models feature j using the distribution in element j of the cell array.

By default, the software sets all predictors specified as categorical predictors (using the Categorical Predictors name-value pair argument) to 'mvmn'. Otherwise, the default distribution is 'normal'.

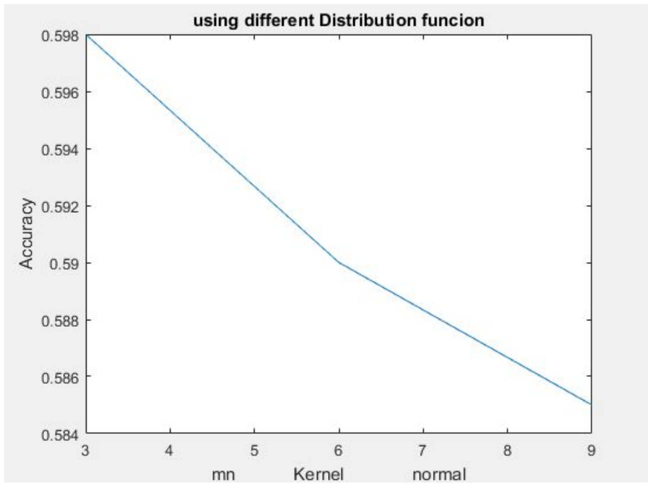


Figure Using different Distribution function to get different accurcy

II. Conclusion

We have presented the general framework for learning to count Vehicles in traffic images of the database TRANCOS. While our ultimate goal is the counting accuracy over the entire image with a lot of overlapping of the vehicles, during the learning our approach is optimizing the loss. This loss involves counting accuracy over multiple subarrays of the entire image (and not only the entire image itself). We demonstrate that given limited amount of training data, such an approach achieves much higher accuracy than optimizing the counting accuracy over the entire image directly). At the same time, we used several different ways to do the training samples and evaluated each way by calculating the prediction results.

Acknowledgment

This work was done under the instruction of Prof. Liu . The detailed each step was discussed and processed by Xi Huang and Yangyan Zhou together. Huang did the data preprocessing and training method logistic regression and KNN. Zhou did the part of feature extraction and training method SVM and naive Bayes. The evaluation and the report was finished by Zhou and Huang together

References

[1]https://en.wikipedia.org/wiki/Logistic_regression

[2]https://en.wikipedia.org/wiki/K-nearest_neighbors_algorithm

[3]https://www.mathworks.com/help/stats/fitcnb.html

[4]https://www.mathworks.com/help/stats/naivebayes-class.html

[5]Everingham, M., Van Gool, L., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC)challenge. IJCV 88(2) (2010) 303–338

[6]Guerrero-Gomez-Olmedo, R., Lopez-Sastre, R.J., Maldonado-Bascon, S., FernandezCaballero, A.: Vehicle tracking by simultaneous detection and viewpoint estimation. In: IWINAC. (2013)

[7]Extremely Overlapping Vehicle Counting. R. Guerrero-Gómez-Olmedo, B. Torre-Jiménez, R. J. López-Sastre, S. Maldonado-Bascón and Daniel Oñoro-Rubio. IbPRIA 2015

[8]http://www.robots.ox.ac.uk/~vgg/research/counting/index.html

[9]P. D. Kovesi. MATLAB and Octave functions for computer vision and image processing. School of Computer Science & Software Engineering, The University of Western Australia. Available from: http://www.csse.uwa.edu.au/~pk/research/matlabfns/

[10]S. K. Nath, K. Palaniappan, and F. Bunyak. Cell segmentation using coupled level sets and graph-vertex coloring. MICCAI (1), pp. 101–108, 2006.