Bag of Visual Words vs YOLO: a VOC2007 comparison of two generation of object recognition approaches

Lucas S. Althoff Student
ls.althoff@gmail.com
Teofilo de Campos Prof
teo.decampos@gmail.br

Departamento de Ciência da Comptutação
Universidade de Brasília
Campus Darcy Ribeiro, Asa Norte
Brasília-DF, CEP 70910-900, Brazil,

Abstract

This work re-visit the Visual Object Classification (VOC) challenge from 2007 to study the evolution of object detection approaches given two well know frameworks that had been state-of-the-art in 2008 and 2015. These methods still have lots of influence in newer model propositions within this visual task problem. The two frameworks are referenced as Bag of Visual Word (BoVW) and You Only Look Once (YOLO). The implementation of BoVW uses SIFT for local features extraction K means for feature clustering, and KNN as a classifier. For YOLO implementation, we selected the two simpler version of the model named tiny-YOLO and YOLO v1. The advance in precision given by deep learning CNN technics was a breakthrough in computer vision area.

1 Introduction

Objects are the semantic building block of visual content, the task devoted to find them is an essential computer vision task. Object recognition have a wide application field extending from security systems [2], to medical tools [6], automatic cinematography [5], geospacial land recognition [10] and autonomous driving [1].

This work is dedicated to analyse the evolution of object detection approaches, see Figure 1 for a recent overview over object recognition milestones. To do our comparison we selected one classical and one deep learning object detection model to be computed within PASCAL visual object classes (VOC) dataset from 2007.

PASCAL VOC is an open benchmark [3, 4] coherent basis of analysis to develop visual tasks solutions, this benchmark is composed with an unbiased dataset and a toolkit composed by padronized set of metrics and data structure. This benchmark are also associated with an annual challenge event to encourage research innovation, since every year a new dataset is made available to the public.

VOC dataset are built seeking to collect images in an unbiased way for object classification, detection and segmentation tasks. After been collection, image data are loaded with ground truth annotation given an categorization of 20 object classes.

Finally, our goal is to understand the evolution of object detection assessing the performance and limitations of two reference frameworks, a classical and one deep learning

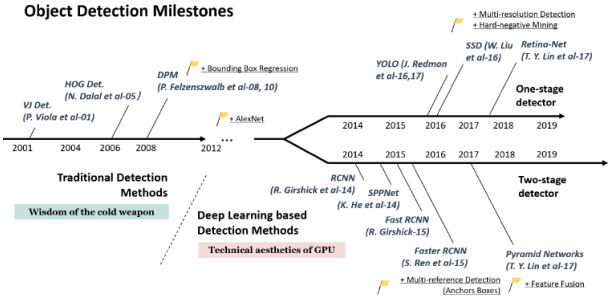

Figure 1: A road map of object detection showing the milestones for object detector divided into two generation, the first characterized, in general, by the exploitation of bag of word-like models, named traditional detection methods. Following, deep learning based models are splited in two main approachs which YOLO represents the one named one-stage detector. Image source:[13].

method: respectively Bag of Visual Word (BoVW) and You Only Look Once (YOLO). As especific goals we will explain the fundaments and principles of each referenced method trying to answer the research question: why is deep learning models so good for object detection?

1.1 Theoretical review

This section give an objective description of the principle methods and stages of BoVW and YOLO, reporting the reader to references with more details, this information are also useful to the analysis inside the discussion section.

1.1.1 Bag of visual word

Bag of visual word method is inspired by a natural language processing problem of retrieving text classification based in word frequencies named bag of words as stated in [7].

The visual version of bag-of-words method interprets image as a full text where some regions of the image, i.e. the visual words, are build from visual features organized in a visual vocabulary represented by a big set of feature clustered inside a kind of visual dictionary named codebook. This dataset codebook works like a reference to guide subsequent classification of single images. Figure 2 shows the pipeline and the principle stages of the process as well as the technics implemented to each one for this study.

Briefly, local features are extracted applying SIFT, then the codebook is trained using K-means clustering method. Then, the encoded visual vocabulary is fixed and applied to assign the descriptors response for a single image. Finally, the response assigned descriptors is fed into a classifier (KNN).

1.1.2 Scale Invariant Feature Transform

SIFT is a algorithm for local feature selection, an image description is done locating points which are invariant to translation, rotation, scaling and lightening variation. SIFT if com-

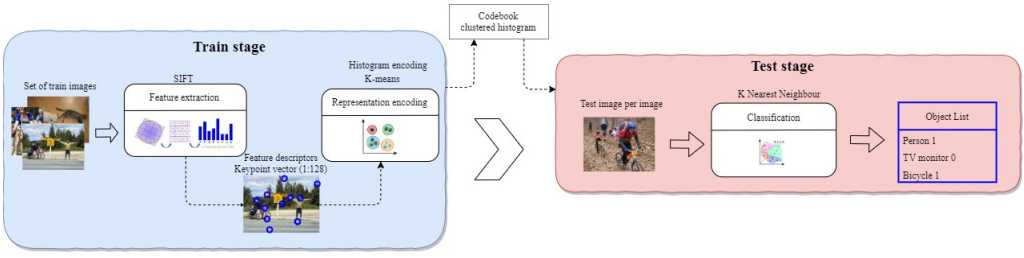

Figure 2: The typical pipeline of the bag of visual word model. First, the visual descriptor space of a set of images lead to the creation of a visual vocabulary. Then, each image feature are assigned inside the vocabulary to be subsequently classified.

posed of four stages 1) detect scale space, 2) localize keypoints, 3) assign orientation, 4) describe points. To apply SIFT in a optmized way we used OpenCV methods *cv2.xfeatures2d.SIFT_* and *sift.detect*().

The first step is done using a scale-space filtering, i.e apply Gaussian filter in image (G) with various σ values with gaussian multiples of σ ($k\sigma$), then compute all the difference of Gaussian (D) images.

$$D(x,y,\sigma) = G((x,y,k\sigma) - G(x,y,\sigma)) * I(x,y) \tag{1}$$

Where $I(x,y)$ is the analysed image x, y represents pixel coordinate and $*$ convolution operation. Gaussian image is given by

$$G(x,y,\sigma) = \frac{1}{2\pi\sigma}\exp[-\frac{x^2+y^2}{2\sigma^2}] \tag{2}$$

This process is done for different image scales building a Gaussian Pyramid. Then is applied a neighbor pixel-per-pixel search for local extrema over scale and space. The extrema points and scale are stored as potential keypoints. In the next stage a refinement on the list of keypoints is done, low contrast region are filtered and edge points are thresholded using Hessian matrix. For orientation assignment, the gradient magnitude and direction is computed and for each keypoint and all orientations above 80% stored.

SIFT descriptor are build computing the gradient magnitude and orientation for the 16x16 neighbourhood around a keypoint. To do so the neighbourhood are divided in 16 sub-blocks of 4x4 size, for each sub-block an orientation histogram of 8 bin is stored as a vector. So, SIFT descriptor is a 128 elements vector. Extra refinements of keypoints are done to give improvements against illumination variation.

1.1.3 K-means

K-means refer to an iterative unsupervised clustering of data. In BoVW, K-means reduce knowledge base for the classification, instead of dealing with milions of keypoints into K (usually thousands) visual words. The set of visual words or cluster are called codebook or visual vocabulary.

K-means algorithm initiate K points randomly over the descriptor space (128 dimensions) then every keypoint is classified with the closer visual word, a centroid of the cluster

Model Variant	Technical Description	Metrics (FPS - mAP)
Tiny YOLO	A simpler version of YOLO with only 9 conv layers for even faster computation	207 – 57.1 (VOC 2007 + 12)
YOLO [10]	A real-time object detector trained over categories with 20, deep learning arch based on GoogLenet [12]	45 – 63.4 (VOC 2007 + 12)
YOLO v2 [8]	A much more dense network based on Darknet-19, adding 9 new from batch normalization in the training, dimension cluster with k-means	40 – 48.6 (COCO test-dev)
YOLO 9000 [8]	The same model as YOLO v2 but with wordtree dataset combination expanding to over 9000 object categories with hierarchical classification.	
YOLO v3 [9]	YOLO v2 with a new backbone architecture the Darknet-53 and changes in cost function calculation and predictions using now the Feature Pyramid Networks (FPN)	30 - 57.9 (COCO test-dev)

Table 1: Summarization of YOLO variants, for time limitation we only implemented tiny and YOLO version.

is computed and the position of cluster center is updated, again all keypoints are assigned and a new centroid is computed until converging to the "best" K positions given the initialization, as this minimization is not global it depends in K initilization.

1.1.4 K Nearest Neighbor

The K Nearest Neighbor (KNN) is a nonparametric classifier, is one of the simplest of classification algorithms available for supervised learning, the algorithm use unlabeled image features (F1) to compare to labeled features (F2), the similarity of features are calculated with a distance function inside the hyperspace of 128 dimension of SIFT descriptor, in this cases, the Euclidean distance:

$$Dist(F_1, F_2) = \sqrt{\sum_{i=1}^{128} (F_{1,i} - F_{2,i})^2} \tag{3}$$

Where i is the feature dimension in the feature space. The classification of KNN is simply based on the a pre-defined number of nearest neighbor to take into account for classification. Opencv also have implemented KNN methods $cv2.ml.KNearest\_create()$, $train(trainData, cv.ml.ROW_SAMPLE, responses)$ and $findNearest(newcomer, K)$

1.1.5 You only look once

Contrasting with BoVW method, YOLO recognize objects and also detect its position. Briefly, YOLO is based on deep learning CNN architecture previously trained over a range of classes designed to achieve end-to-end real-time object recognition in videos [10].

While traditional object detection systems exploit descriptors and classifiers to perform detection using the overall statistic of regions that could represent objects, end-to-end models like YOLO divide image into a grid to compute in a single loss function the region candidates and the class probability map that together outputs object class and position.

In other words, traditional strategy convolves image at multiple locations and scales searching for high scoring regions that are considered detections. In its turn, YOLO saves

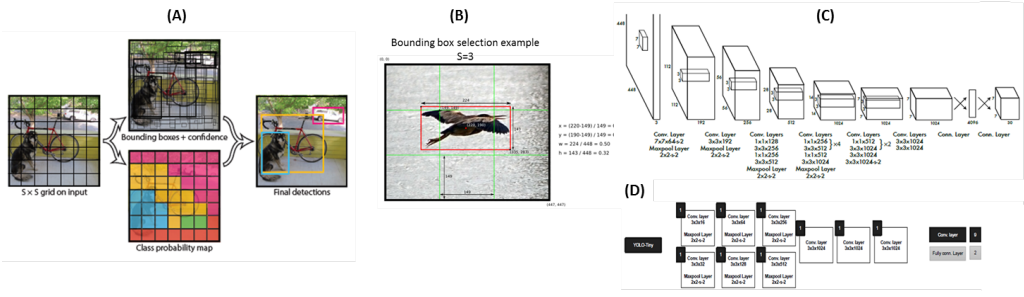

Figure 3: YOLO framework A) steps to detect objects B) Example of bounding box selection with extraction of normalized parameters C) YOLO neural network architecture D) Tiny Yolo architecture simplified for fast tests.

effort introducing the full image into a single neural network in an end-to-end manner [10]. Figure 3 exhibit the network stages, i.e grid distribution for bounding boxes prediction and class segmentation of each grid cell.

At this moment, YOLO model has at least 5 oficial variants, since its first proposal in 2015 improvements were done in the overall stages and elements of the method that define YOLO, Table 1 these five variants and a brief technical description of them.

In terms of accuracy, YOLO lags behind other recent object detection systems. Even though, YOLO is one of the fastest methods of object detection in literature, that is why it is a good option when dealing with real-time object detection, it has also flexibility to be used in generalized situations like artwork and games.

1.2 YOLO's unified detection

As YOLO has a unified detection the central issue on how YOLO detect objects is to understand how the output of the network is encoded from an input image to the list of objects and positions.

First of all YOLO divides the input image into a grid, for each cell of the grid is computed the probability of an object being inside, the probability is stated by five normalized parameters: $(x, y, w, h, confidence)$. The (x, y) coordinates represent the center of the box candidate, relative to the grid cell location, the (w, h) represent the box width and height with respect to the whole image. In its turn, $confidence$ reflects the presence or absence of an object of any class inside that cell.

$$Pr(Class_i|Object) * Pr(Object) * IoU_{pred}^{Truth} = Pr(Class_i) * IoU_{pred}^{Truth} \quad (4)$$

YOLO states that only one object can be predicted for each cell, in other words, the cell where the object center is responsible for that object. This rule limits how close detected objects can be, but it forces the loop to continue over the entire image.

BoVW	
Training Algorithm	**Testing Algorithm**
Input (VOC2007 dataset images) Output (k - clusters, k - visual word)	Input (k - visual word, test image) Output (labeled image)
Step 1: Collect set of images for each class of interest (using VOCdev toolkit). Step 2: Apply BoVW on collected images. BoVW consists of three main steps: 1. Extract keypoints from images using SIFT feature detection. 2. Create descriptor for each extracted keypoints. 3. Clustering features using k-means clustering algorithm (Create visual vocabulary using vector quantization of descriptor space) and store the resulting "visual words".	Step 1: Get unlabeled test image with VOCdev toolkit. Step 2: Extract features of test image using SIFT. Step 3: Extract visual word (centroid) for testing image. Step 4: Calculate the nearest neighbor using Euclidean distance between visual word of tested image and visual words of training images. Step 5: Take the decision: compare extracted features of unlabeled image with visual words extracted in training stage.

Table 2: Sequence of steps for applying BoVW

2 Methodology

In this section will present the overall algorithm of each object detector methodology to give light to the performance analyse of the models as well as the limitation aspects of them.

Because time limitation, YOLO could not be trained from scratch, we only extracted results samples for pre-trained tiny YOLO and YOLO v1.

3 Results

To evaluate object detection algorithms precision, you usually use the Average Precision or mAP(Mean Average Precision). This metrics are based on precision and recall calculation of a predictive model.

$$Precision = \frac{t_p}{t_p + f_p} \tag{5}$$

Where where tp = true positive, fp = false positive, fn = false negative. Measuring the proportion of correct positive predictions (in the context of object detection true negative has no useful meaning) with respect to all positive predictions.

$$Recall = \frac{t_p}{t_p + f_n} \tag{6}$$

For N classes the mean average precision is given by

$$mAP = \frac{1}{N} \sum_{i=1}^{N} AP_i \tag{7}$$

So, mAP for is the average of the AP calculated for all the classes. AP is calculated by taking the area under the precision-recall curve, after interpolating the PR curve into regular steps. A simple way to do this is dividing PR curve into 11 rectangles.

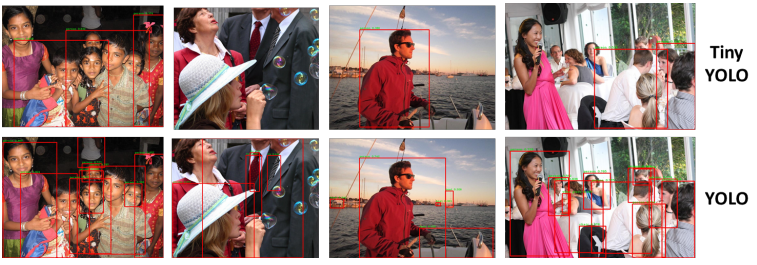

Figure 4: Comparison between the both YOLO simpler versions tested on 20 images. Showing the expected impact on the accuracy of prediction because the absence of 4 layers and the difference over the training dataset (MS COCO for YOLO and VOC 07 + 2012 for tiny YOLO). However, the Tiny version still captures correct bounding boxes.

SIFT has a big computer cost, so for performance improvement we limited the number of SIFT vectors extracted from each image, although all images was analysed to guarantee a vocabulary diversity.

4 Final Remarks

BoVW depends on a vocabulary that is not fixed, for every initialization of K-means a new vocabulary is constructed and it impacts the precision of the classifier.

These improvements rely on the great results neural networks based in convolution operations have in extracting visual features in supervised training contexts of object/region recognition.

To accurately detect or recognize objects in images/videos models have to overcome some challenging situations because of the vastness of object presentation in an image like the variety of object luminosities, angle of view, occlusion and the variety of objects of the same class. In machine learning context, one way to handle these vastness are data augmentation, Essentially, the more data used for training a object detection model more diversity in object representation will be considered. Benchmarkings on object detection like VOC are are responsible for for model improvements. A big set of labeled organized data are essential for classification technics.

4.1 Future work

This work lacks on quantitative results due to the time limitations to conduct the full methodology, so for future work we will compare the mAP results for BoVW and YOLO giving more insights on the advantages and limitation of the both methods.

References

[1] Eduardo Arnold, Omar Y Al-Jarrah, Mehrdad Dianati, Saber Fallah, David Oxtoby, and Alex Mouzakitis. A survey on 3d object detection methods for autonomous driving applications. *IEEE Transactions on Intelligent Transportation Systems*, 2019.

[2] Jesus Benito-Picazo, Enrique Domínguez, Esteban J Palomo, and Ezequiel López-Rubio. Deep learning-based security system powered by low cost hardware and panoramic cameras. In *International Work-Conference on the Interplay Between Natural and Artificial Computation*, pages 317–326. Springer, 2019.

[3] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (VOC) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010. Available from http://host.robots.ox.ac.uk/pascal/VOC/pubs/everingham10.pdf.

[4] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, January 2015. Available from http://host.robots.ox.ac.uk/pascal/VOC/pubs/everingham15.pdf.

[5] Ijaz Ul Haq, Khan Muhammad, Tanveer Hussain, Soonil Kwon, Maleerat Sodanil, Sung Wook Baik, and Mi Young Lee. Movie scene segmentation using object detection and set theory. *International Journal of Distributed Sensor Networks*, 15(6): 1550147719845277, 2019.

[6] Zhuoling Li, Minghui Dong, Shiping Wen, Xiang Hu, Pan Zhou, and Zhigang Zeng. Clu-cnns: Object detection for medical images. *Neurocomputing*, 350:53–59, 2019.

[7] Taurius Petraitis, Rytis Maskeliunas, Robertas Damasevicius, Dawid Polap, Marcin Wozniak, and Marcin Gabryel. Environment recognition based on images using bag-of-words. In *IJCCI*, pages 166–176, 2017.

[8] Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[9] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

[10] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[11] Gilbert Rotich, Sathyanarayanan Aakur, Rodrigo Minetto, Mauricio Pamplona Segundo, and Sudeep Sarkar. Using semantic relationships among objects for geospatial land use classification. In *2018 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pages 1–7. IEEE, 2018.

[12] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[13] Zhengxia Zou, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *arXiv preprint arXiv:1905.05055*, 2019.