Towards a large-scale category detection with a distributed hierarchical compositional model

Domen Tabernik[1], **Matej Kristan**[1], **Marko Boben**[1], **Aleš Leonardis**[1,2]

[1]*Faculty of Computer and Information Science, University of Ljubljana*
[2]*CN-CR Centre, School of Computer Science, University of Birmingham*
{domen.tabernik},{matej.kristan},{marko.boben},{ales.leonardis}@*fri.uni-lj.si*

Abstract

In this paper we evaluate a visual object detection system implemented on a distributed processing platform, presented in our previous work, with the goal of assessing the scalability of the system to a large-scale category detection. While state-of-the-art detection methods based on sliding windows may not be capable of scaling to a higher number of categories, we provide initial evidence that using a hierarchical compositional method called learned-hierarchy-of-parts (LHOP) may be capable of scaling to a higher number of categories. We show with the library trained on an MPEG-7 Shape database that the method is capable of scaling from a system with 5 categories and 6 second averaged response time to a system with 70 categories and averaged response time of 27 seconds.

1 Introduction

The continuous improvements in hardware has enabled availability of cheap and plentiful computational power, which in the field of computer vision has been utilized to enable large-scale image recognition. The latest benchmark datasets for visual object classification, such as ImageNet [2] with around 15 million images and 20.000 classes, would not have been possible to process without an increase in computational power and, in particularly, without an ability to process distributed over a cluster of machines. Implementing computer vision algorithms in a cloud environment has also enabled such algorithms for the use in web-services where new applications can exploit powerful computer vision algorithms. For instance, such popular web-services include Google Image Search[1] , TinEye[2] and Microglossa[3] with an ability to find images based on similar features, or Google Goggles[4] with its ability to identify text and logos among other properties.

However, such online services rely only on simple classification algorithms and do not utilize any advanced visual object detection or localization algorithms. Having an ability to do a more powerful object localization and detection would open up a possibility for new applications, such as a backend services for different robotic vision systems, where localization of objects is an important aspect. A main reason for the lack of localization in such systems can be contributed to a complexity of detection algorithm which would require a significant amount of computational power. For instance, in image classification one of the state-of-the-art approaches [6] relies on a 200.000 dimensional descriptor composed from HOG [1] and LBP [7]. Applying this to a localization with sliding windows would increase computational complexity by more then 100.000 fold, as each image would contain anywhere from 10.000 to more then 100.000 windows. In other state-of-the-art detection systems, such as deformable parts model [3], increased computational complexity can be mitigated with a trade-off between sliding windows and lowered feature dimensionality. However, such systems are still not fully scalable to higher number of categories as each category has to be searched

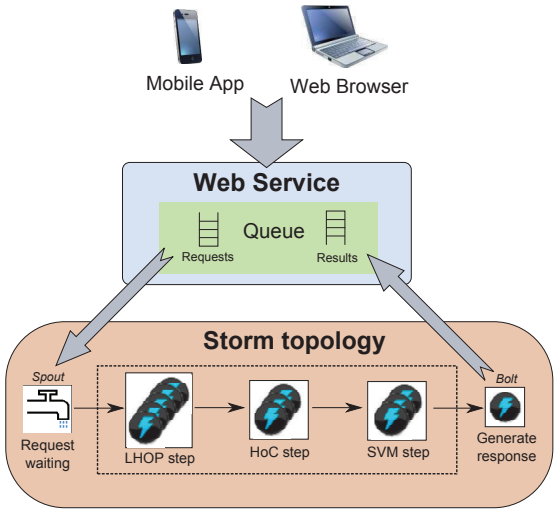

Figure 1: An overview of the system with front-end web-service and back-end implementation of the object detection on a distributed processing platform.

[1]http://images.google.com/
[2]http://www.tineye.com
[3]http://www.macroglossa.com
[4]http://www.google.com/mobile/goggles

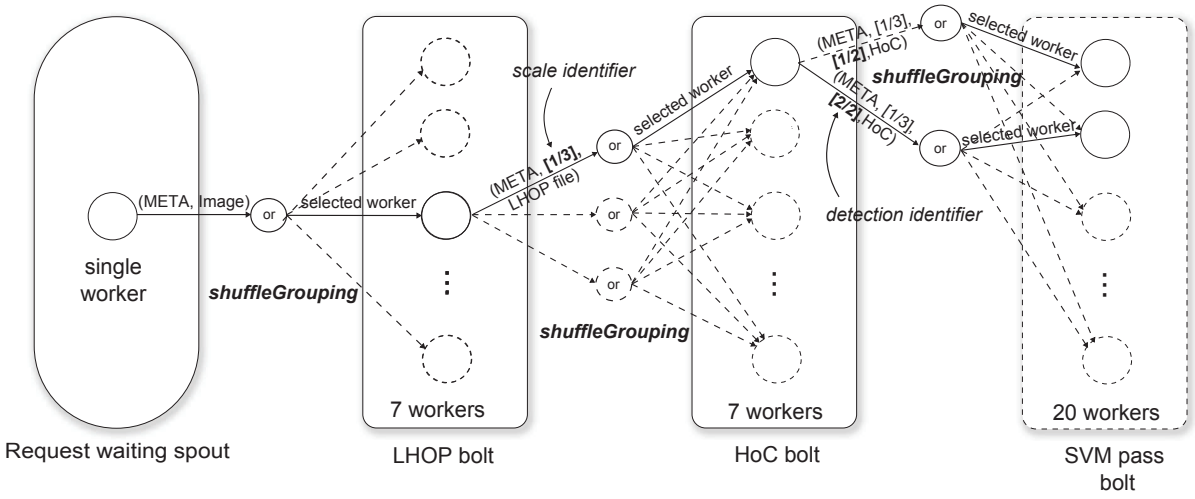

Figure 2: Overview of the first two steps in the topology: the LHOP and the HoC steps.

independently. Scaling to more then 1000 categories would require 1000 fold increase in computational time, with even more time required if different viewpoints are treated as separate classes.

The issue of computational complexity of detection algorithm was addressed in our previous work [9, 8], where a web-service running on a distributed processing platform was proposed. To enable detection on real-time distributed processing platform and a scalability to a higher number of categories, a hierarchical compositional method, called learned-hierarchy-of-parts [4] (LHOP), was implemented instead of a computationally expensive sliding window methods. Additionally, to handle many false positive detections known to occur in such hierarchical methods, a hypothesis verification with HoC descriptor was proposed. This added additional computational time as a non-linear support vector machine was used to perform hypothesis verifications, however, the hypothesis verification was performed only on regions corresponding to specific detections and did not present any significant performance bottlenecks. The presented web-service was shown to produce a response time of around 2 seconds for image classification with 100 categories. For object detection a 10 seconds response time was achieved, however, only two categories were used in that evaluation and an ability to scale was not shown.

In this work we evaluate the distributed processing platform for object detection proposed in our previous work [8], however, as opposed to our previous work, we focus on evaluating a scalability of such system to a higher number of categories to show a possibility of handling large-scale image category detection. We evaluate the system by measuring the response time of the library trained on 70 categories of MPEG-7 Shapes [5] dataset. Additionally, our contribution can also be found in the modification of a hypothesis verification step to accommodate object detection for higher number of categories.

The remainder of this paper is structured as follows: in Section 2 an LHOP implemented on a distributed processing platform with modifications to hypothesis verification is presented, in Section 3 the evaluation is performed and in Section 4 a conclusion is provided.

2 A distributed implementation of LHOP

In this section we detail the implementation of the LHOP detection algorithm in a distributed processing platform. The algorithm is implemented on top of an Apache Storm[5] platform with its computational model consisting of a directed graph, called a topology. As in our previous work [8], we build our topology from three main steps: (i) an LHOP step where we process the image and produce hierarchical compositions, (ii) an HoC step where we extract detected regions and produce HoC descriptor for each detected object and (iii) an SVM step where we classify each descriptor into pre-trained categories (see, Fig. 1 for overview). However, the HoC and the SVM steps were further adjusted to allow for a large-scale category detection.

2.1 The LHOP step

The initial LHOP step remains the same as in [8]. The input generating spout is implemented with the Request waiting spout, which communicates with the web-service through the Beanstalk[6] queuing system. The initial request, being forwarded from the web-service, comes in a form of a meta-data and a query image $(Metadata, Image)$, with meta-data containing additional information about the request. Due to a small workload only one worker is assigned for this spout.

The initial processing of the request is performed by the LHOP bolt where each request, emitted by the Request

[5] https://storm.incubator.apache.org/
[6] http://kr.github.com/beanstalkd/

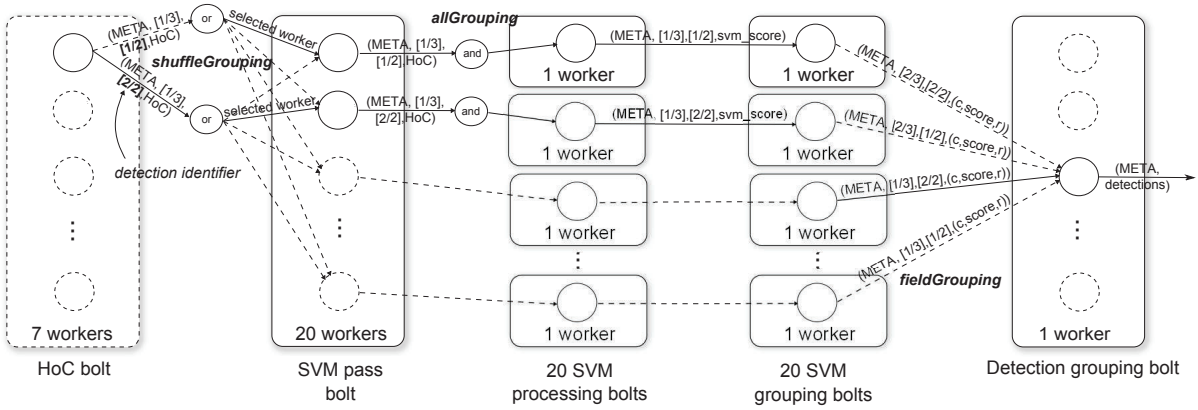

Figure 3: The overview of the last step containing SVM related bolts.

waiting spout, is received as $(Metadata, Image)$ tuple by the LHOP bolt. To ensure even distribution of the workload a shuffle grouping is used as a connection between the input spout and the LHOP workers (see, Fig. 2). The work for LHOP bolt consists of computing LHOP compositions from the input image and emitting results for each scale to the next bolt. Results are emitted as multiple new tuples, one for each scale, to ensure detections of each scale can be further processed in parallel. While in previous work the LHOP processing had 50 workers assigned, the large-scale category detection implementation has main performance bottleneck in the SVM processing and has therefore the most workers assigned. The LHOP bolt now has only 7 workers remaining.

2.2 The HoC step

The second step mostly remains unchanged compared to [8], however, additional output information is returned and overhead, when 1000 or more descriptors are emitted, is eliminated. As part of the core HoC operation, regions of all the detected objects are extracted from each input LHOP detection found at the particular scale and each detected region is used to generate a HoC descriptor to pass it along to the next step.

The output is further optimized by returning a batch of 100 descriptors packed into one output tuple instead of outputting each descriptor individually. Additionally, the output is modified to attach the detected object category to each HoC descriptor. This value will be used in the SVM step to optimize the required processing. The output of the HoC bolt now returns multiple tuples, each containing meta-data, scale identifier, detection identifier and a batch of HoC descriptors with bounding box and a category information. Due to low computational requirements this bolt has only 7 workers assigned.

2.3 The SVM step

With the last step in our topology we perform classification of the detection using a Support Vector Machine. The topology configuration for the SVM step is depicted in Figure 3. The SVM bolts implementing this have been modified to enable large-scale category detection. The algorithm used in our previous work [8] has verified each detection with all the possible categories and only the best category has been used as correct one. However, verification of all categories is not scalable to a large-scale category detection as the computational time would increase linearly with the number of added categories. Instead, we can perform hypothesis verification only for the category that was detected, and avoid verification for any other categories. Additionally, as each bolt, as implemented in [8], was originally intended to perform verification of each detection for multiple categories a parallel workers in one bolt were utilized to each handle only a subset of categories. While at the same time each detection was sent to all the workers so that all the categories could be verified. While this mechanism is still enabled for any web-service requiring classification of the whole image, this has been effectively disabled in the detection process by using only one worker per bolt. However, as the number of detections outputted by the LHOP and HoC bolts can substantially increase, we modified the topology of the SVM step to better handle such increase of the workload by adding a multiple SVM processing bolts to process detections in parallel. A new additional bolt, named SVM-pass, was also added between the HoC bolt and the SVM-processing bolt. The main task of this bolt is to only re-route the individual detections into the specific SVM-processing bolt. To handle a significant increase in number of detections 20 bolts were assigned. The same number of workers were also assigned to the SVM-pass bolt.

The remainder of this step is implemented similarly to [8], with the SVM-grouping bolt performing the task of grouping multiple SVM scores from one detections. One bolt is created for each SVM processing bolt, however, as only one worker is assigned to each SVM processing bolt, the work of this bolt is reduced to only passing the results to the next bolt.

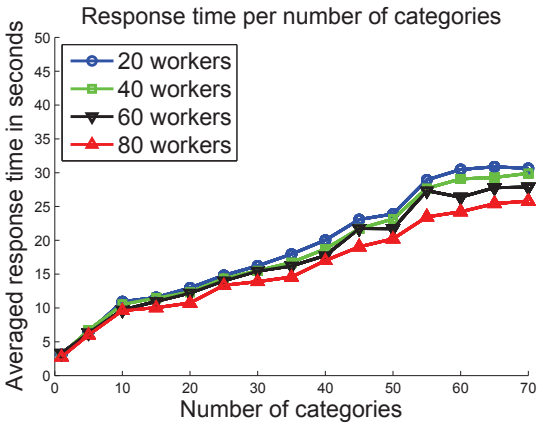

Figure 4: An averaged response time for the object detection performed on a cluster with different number of workers and with different number of categories.

3 Evaluation and results

The evaluation process was performed on a cluster of three machines: one machine with 16 CPU cores and two machines with 32 CPU cores. Combined together 80 workers were allocated for the Storm topology. Note, that all the experiments were performed with the OpenMP enabled. We generated LHOP library with up to 70 different visual categories from the MPEG-7 Shapes [5] dataset. Each category was incrementally trained, separately from others. Intermediate libraries with accumulated categories were saved to obtain testing libraries for different number of categories. Evaluation was performed with the sample image of 512x512 pixels with 7 scales and 10 repeated requests.

The results are reported in Figure 4. With all 70 categories combined in the library we achieve response time of around 27 seconds, while up to 5 categories can be detected in 6 seconds. Focusing on an ability to scale, we can see a slow increase of the response time when new categories are being added. While pure logarithmic increase of the response time cannot be claimed yet, a linear increase with a small slope can be observed. To confirm a possible logarithmic increase an evaluation with 100, 500 and 1000 would have to be performed.

We can also observe a modest improvement in the response time when additional workers are added. The improvements are more distinct with the higher number of categories, however, even in such cases doubling the number of workers does not reduce the response time in half. This can be partially explained by the LHOP processing which is not being distributed over the cluster, but it may also be an indication of under-utilization of the system and further improvements could still be gained by optimizing the topology.

4 Conclusion

In this work we have performed an evaluation of the object detection algorithm, running on a distributed processing platform. The main goal of the evaluation was to asses the scalability of the hierarchical compositional methods to a large-scale category detection problem. We have relied on a learned-hierarchy-of-parts [4] (LHOP) running as a topology on a Apache Storm platform to provide a detection algorithm more suitable for detecting higher number of categories than state-of-the-art sliding window methods. Our evaluation shows a favorable results for the hierarchical compositional method as the added categories are only modestly increasing the averaged response time of the detection.

In our future work we will further increase a the number of categories to confirm a possible logarithmic increase of the response time. We also plan to compare such system to the state-of-the-art sliding windows method, such as Deformable Parts Model [3], to further show a superiority of hierarchical compositional method in the large-scale category detection problem. We also plan to further optimize the topology as certain aspects of the results point to the under-utilization of the whole system.

Acknowledgments. This work was supported in part by ARRS research program P2-0214 and ARRS research projects J2-4284, J2-3607 and J2-2221.

References

[1] Navneet Dalal and Bill Triggs. Histograms of Oriented Gradients for Human Detection. In *CVPR*, pages 886–893, 2005.

[2] Jia Deng, AC Berg, Kai Li, and L Fei-Fei. What does classifying more than 10,000 image categories tell us? *11th European Conference on Computer Vision*, 5:71–84, 2010.

[3] Pedro F Felzenszwalb, Ross B Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE transactions on pattern analysis and machine intelligence*, 32:1627–1645, 2010.

[4] Sanja Fidler and Ales Leonardis. Towards Scalable Representations of Object Categories: Learning a Hierarchy of Parts. In *CVPR*. IEEE Computer Society, 2007.

[5] Longin Jan Latecki, Rolf Lak, and Ulrich Eckhardt. Shape Descriptors for Non-rigid Shapes with a Single Closed Contour. *CVPR*, pages 424–429, 2000.

[6] Yuanqing Lin, Fengjun Lv, Shenghuo Zhu, Ming Yang, and Timothee Cour. Large-scale image classification: fast feature extraction and svm training. *Computer Vision and Pattern Recognition*, (October), 2011.

[7] Tim Ojala, Matti Pietikhenl, David Harwood, and Laws Texture Measures. Performance evaluation of texture measures with classification based on Kullback discrimination of distributionsi. *Proceedings of the 12th IAPR International Conference on Pattern Recognition (ICPR 1994)*, 1:582–585, 1994.

[8] Domen Tabernik, Luka Čehovin, Matej Kristan, Marko Boben, and Aleš Leonardis. A web-service for object detection using hierarchical models. In *The 9th International Conference on Computer Vision Systems*, 2013.

[9] Domen Tabernik, Luka Čehovin, Matej Kristan, Marko Boben, and Aleš Leonardis. ViCoS Eye - a webservice for visual object categorization. In *Proceedings of the 18th Computer Vision Winter Workshop*, 2013.