SINGLE-SHOT SEMANTIC MATCHER FOR UNSEEN OBJECT DETECTION

V. Gorbatsevich[1]*, Y. Vizilter[1], V. Knyaz[1,2], A. Moiseenko[1,2]

[1] State Research Institute of Aviation System (GosNIIAS), 125319 Moscow, Russia – (gvs,viz,knyaz,moiseenkoas)@gosniias.ru
[2] Moscow Institute of Physics and Technology (MIPT), Russia

Commission II, WG II/5

KEY WORDS: Convolutional Neural Networks, Object Detection, Semantic Matching, Single-Shot Detector

ABSTRACT:

In this paper we combine the ideas of image matching, object detection, image retrieval and zero-shot learning for stating and solving the semantic matching problem. Semantic matcher takes two images (test and request) as input and returns detected objects (bounding boxes) on test image corresponding to semantic class represented by request (sample) image. We implement our single-shot semantic matcher CNN architecture based on GoogleNet and YOLO/DetectNet architectures. We propose the detection-by-request training and testing protocols for semantic matching algorithms. We train and test our CNN on the ILSVRC 2014 with 200 seen and 90 unseen classes and provide the real-time object detection with mAP 23 for seen and mAP 21 for unseen classes.

INTRODUCTION

Various image matching techniques are traditionally applied in stereo vision, 3D reconstruction, structure-from-motion, SLAM algorithms, multi-sensor fusion, landmark detection, object tracking and so on. However, classical image matching techniques cannot match images with essential shape or pose inter-frame changes, because image matching operates with pixel-level image fragments or mid-level shape templates, but not with semantic objects or object parts. On the other hand, object detection techniques presume the detection and localization of all objects of some given class (or classes). In the early years of computer vision object detection was implemented in nave manner based on direct matching of class samples. In the modern computer vision such nave detection-by-matching is totally substituted by learning-based detection. Currently all state-of-the-art detectors utilize the deep convolutional neural networks (CNN). Unfortunately, there is an important limitation of CNN-based detectors: they can detect only the objects of seen classes presented at CNN training stage. But sometimes we need to detect objects of previously unseen classes based on just one or too few sample images available at the execution stage. This unseen object detection problem is close to the image matching, but it requires the more sophisticated semantic matching. We propose to call the semantic matcher any procedure, which takes two images (test and sample) as input and returns the set of object detections (bounding boxes on test image) corresponding to semantic class represented by input sample image.

In this paper we combine the ideas of image matching, object detection, image retrieval and zero-shot learning for practical real-time solving of the semantic matching problem. In our single-shot semantic matcher CNN architecture we compare the deep features from the top hidden layer of single-shot detector with deep features representing the request (sample) image. Our CNN consists of three parts: Object Detector, Request Descriptor and Semantic Matcher. We perform the joint end-to-end training of these three CNN parts. Our implementation of semantic matching CNN architecture is based on GoogleNet (Szegedy et al., 2015) and YOLO/DetectNet (Redmon et al., 2016) CNN architectures. We train and test our net on the ILSVRC 2014 dataset (200 seen and 90 unseen classes) and provide the real-time object detection with mAP (mean average precision) 23 for seen and 21 for unseen classes.

The contributions of this paper are the following:
- new problem statement for semantic matching;
- original single-shot semantic matcher CNN architecture containing Object Detector, Request Descriptor and Semantic Matcher parts;
- experimental demonstration of possibility for real-time detection of unseen objects.

1. RELATED WORKS

1.1 Object Detection

Modern CNN based object detection methods are divided into two groups by computation scheme: region proposal based and single shot methods.

1.1.1 Proposal based methods are represented by R-CNN (Girshick et al., 2014), Fast RCNN (Girshick, 2015), Faster R-CNN (Ren et al., 2015), R-FCN (Dai et al., 2016), FPN (Lin et al., 2017a). R-CNN uses selective search method to generate region proposals in an image and then perform classification on the regions. R-CNN is very slow because the CNN must process each region separately. Fast R-CNN and Faster R-CNN improve the efficiency by using neural networks to generate the region proposals and share features between detector CNN and classifier CNN. R-FCN improves speed and accuracy by removing fully-connected layers for final detection. FPN uses pyramid CNN architecture based on ideas of SSD (Liu et al., 2016) and DenseNet (Huang et al., 2017) to improve accuracy. Deformable R-CNN (Dai et al., 2017) uses dynamic filters in convolutional layers to take into account for variability of view angle, object size and so on.

*Corresponding author

1.1.2 Single shot methods like YOLO (Redmon et al., 2016), SSD (Liu et al., 2016), DSOD (Shen et al., 2017), RetinaNet (Lin et al., 2017b) have recently been proposed for real-time detection. YOLO uses a single CNN to predict object class and location in one stage. There is not longer required to do per-region classification operation so that it is extremely fast. SSD improves YOLO in several aspects: using different layers for classification and bounding estimation; using different CNN layers for different scales; shape priors to improve IOU. DSOD is built upon the SSD framework but uses DenseNet-like (Huang et al., 2017) architecture to get better quality. RetinaNet based on FPN architecture that is transformed to single shot and special loss function – Focal Loss that keeps focus on hard cases during learning.

1.2 Image retrieval

Image retrieval is one of a fundamental task of machine learning. Modern algorithms are grown from handcrafted features and indexing algorithms (e.g, (Lowe, 2004), (Bay et al., 2006), (Dong et al., 2010)) to methods that uses CNN for global descriptor construction (for example (Ng et al., 2015). A variety of methods uses Deep features approach for image retrieval (for example, (Arandjelovic et al., 2016)). There based on the generalization properties of CNNs, CNNs were trained for classification on some big dataset (for example ILSVRC). (Babenko and Lempitsky, 2015) proposed to fine tune networks on a dataset similar or closed to target dataset. The same idea was widely used for different tasks. In (Torii et al., 2015) geo-tagged datasets were used for weakly supervised fine-tuning of a triplet network. A image pair with big distance used as non-matching, while matching pair are picked by closest images. Distance is defined by the current CNN representation. This approach performs end-to-end fine-tuning for image retrieval. In (Radenovic et al., 2016) hard negative mining was used to improve accuracy. In (Cao et al., 2017) original hashing method was used for effective feature coding. In (Zepeda and Perez, 2015) exemplar SVM was proposed for final classification.

2. MAIN IDEA AND CNN ARCHITECTURE

This paragraph evolves and describes the ideas of semantic matching, which were briefly outlined in Introduction.

As we mentioned above, image matching problem refers to the comparison of two images test and sample. Matching process usually consists in a search of correspondence between sample image and test image (or some part of test image). The result of image matching is a set of sample-to-test correspondence parameters, which describe the localization of corresponding part of test image and (if required) the transformation of this image fragment relative to sample image. Image matching algorithms need to be robust with respect to major lighting/color and minor pose/shape changes. However, classical image matching techniques cannot match images with major shape or 3D pose changes like those demonstrated in Figure 1, because image matching operates with pixel-level image fragments or mid-level segmented shape templates, but not with semantic objects or object parts.

On the other hand, object detection problem presumes the detection and localization of all objects of some given class in the observed (test) image. Currently all state-of-the-art object detectors are based on deep convolutional neural networks (CNN). These CNNs take test image as input and return the list of object proposals (bounding boxes) with some belief estimates for recognized object class. The training of such CNN-based detectors requires thousands of examples for each class. The main natural limitation of such common learning-based object detection scheme is that we can detect only objects of seen classes (presented in our training set).

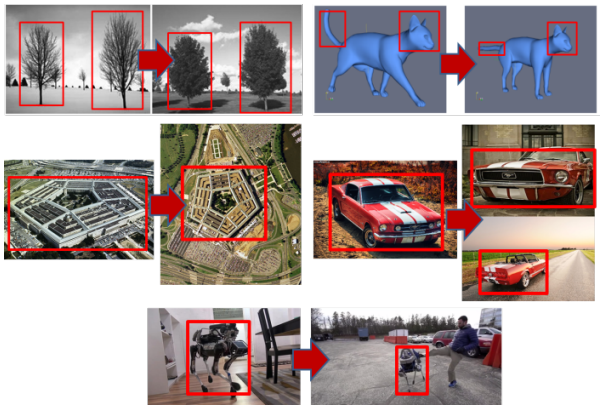


Figure 1. Semantic matching as image matching with major shape and/or pose changes

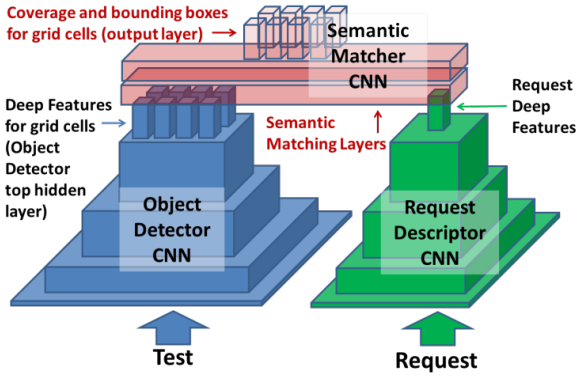
But sometimes we need to detect semantic objects of previously unseen class based on one or too few new sample images available at the execution stage. This unseen object detection problem is close to the image matching problem, but it requires the more sophisticated semantic matching. Semantic matcher should take two images (test and sample) as input and return the list of object proposals (bounding boxes on test image) with some belief estimates, which correspond not to concrete input sample, but to whole (previously unseen) semantic class represented by this sample image. The original single-shot detectors are trained for detection of some fixed seen classes, but we propose to apply the retrieval by deep feature trick for fusion of single-shot detection and image retrieval ideas. In our single-shot semantic matcher CNN architecture we compare the deep features from the top hidden layer of single-shot detector (see Figure 2) with deep features representing the request class-sample image.

So, our CNN consists of three parts: Object Detector, Request Descriptor and Semantic Matcher (Figure 2). We perform the joint end-to-end training of these three CNN parts. Thus we obtain the real-time semantic object detection-by-request. Our CNN architecture (Table 1) is based on the GoogleNet-V1 CNN that implemented in Caffe (Jia et al., 2014) framework and pre-trained on the ILSVRC 2012 image base.

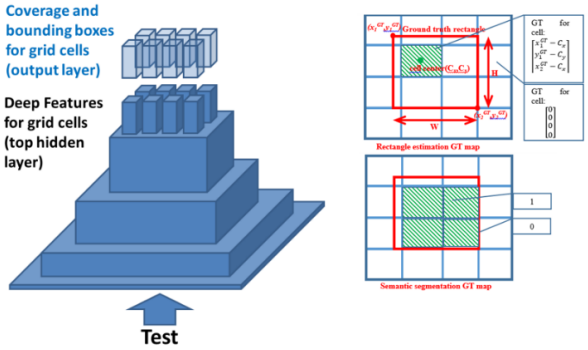
<table>
<tr><td>Input image</td><td>Test (512 × 512)</td><td></td><td>128 × 128</td></tr>
<tr><td>Object Detector CNN</td><td>GoogleNet FCN</td><td>Request descriptor CNN</td><td>GoogleNet FCN</td></tr>
<tr><td colspan="4">Concatenation</td></tr>
<tr><td rowspan="3">Semantic Matcher CNN</td><td colspan="3">Conv 1x1 (1024)</td></tr>
<tr><td colspan="3">Conv 1x1 (1024)</td></tr>
<tr><td>Conv 1x1 (1) (Segmentation)</td><td colspan="2">Conv 1x1 (4) (Bboxes)</td></tr>
</table>

Table 1. Our CNN architecture

We select the GoogleNet as a basic CNN model due to its very wide spreading in a machine learning society, which provides the guaranteed repeatability of our results with the use of any



(a) single-shot object detector



(b) single-shot semantic matcher

Figure 2. From single-shot object detector (a) to our single-shot semantic matcher CNN architecture (b).

deep learning framework. For Object Detector CNN and Request Descriptor CNN we use the GoogleNet-V1. So Object Detector CNN contains 2 convolution layers, 9 inception modules, and 3 pooling. Request Descriptor CNN has a similar architecture with the exception of 2 pooling. Semantic matcher CNN contains two 1×1 convolutional layers serially and two convolutional layers for bounding box prediction and semantic segmentation. The output grid cell is 16×16 pixels of size.

To train the CNN we use the complex loss function that is the linear combination of loss functions for objects semantic segmentation and objects bounding boxes estimation:

$$L = L_{semantic} + L_{rectangle} \tag{1}$$

For semantic segmentation we use the following $L2$-loss function:

$$L_{semantic} = \sum_{k=1}^{N} \sum_{x=1}^{W} \sum_{y=1}^{H} (label_k(x,y) - GTmap_k(x,y))^2 \tag{2}$$

Here
N – number of samples;
W, H – horizontal and vertical grid sizes;
$label_k(x,y)$ – predicted class label map for k-th sample;
$GTmap_k(x,y)$ – ground truth class label map for k-th sample.

In addition, we use the $L1$-loss function normalized by the geometric mean of ground truth width and height of object in order to save balance between objects that belong in the wide size range:

$$\begin{aligned} L_{rectangle} = \sum_{k=1}^{N} \sum_{x=1}^{W} \sum_{y=1}^{H} & |x_1^k(x,y) - x_{GT_1}^k(x,y)| \cdot K_{norm}^k \\ & + |y_1^k(x,y) - y_{GT_1}^k(x,y)| \cdot K_{norm}^k \\ & + |x_2^k(x,y) - x_{GT_2}^k(x,y)| \cdot K_{norm}^k \\ & + |y_2^k(x,y) - y_{GT_2}^k(x,y)| \cdot K_{norm}^k \end{aligned} \tag{3}$$

Here
$x_1^k(x,y), y_1^k(x,y), \ldots$ – prediction of relative coordinates for k-th sample in cell $C_{x,y}$;
$x_{GT_1}^k(x,y), y_1^k(x,y), \ldots$ ground truth relative coordinates for k-th sample in cell $C_{x,y}$;
$K_{norm}^k = \frac{1}{\sqrt{W_k \cdot H_k}}$ – scale normalization coefficient for k-th sample;
W_k, H_k – ground truth width and height of k-th sample.

Note that scale normalization is implemented in our architecture as a special output-like mask layer with $4 \times W \times H$ cells containing the scale normalization coefficients if these cells cover object appropriated input reference image, and 0 values - otherwise. At the training stage the element-wise multiplication of CNN output and this mask is performed and this way the requirement of scale normalization of coordinate errors in back propagation as well as ignoring the cells without objects in the learning for bounding box prediction is fulfilled.

So training dataset consist of pairs of images – test image and request image. Those pairs can be negative (there is no object of requested class on test image) or positive (there is some object of requested class on test image).

Finally, lets note that our CNN consists of three parts: Object Detector, Request Descriptor and Semantic Matcher, which process different input data. Object Detector processes the test image, Request Descriptor processes the request (sample) image representing the requested class and Semantic Matcher processes the activations of top layers of Object Detector and Request Descriptor CNNs. Due to this we can decrease the amount of computations via preliminary calculation, tabular storage and further reuse of deep descriptors. Such implementation allows preserving the high processing speed of single-shot detector even if we deal with more than one sample per class or multiple classes, because we run Object Detector once, take requests descriptions from memory storage and execute only the Semantic Matcher each time for each proper analyzed request.

3. EXPERIMENTAL RESULTS

We use the ILSVRC 2014 dataset (training subset) for our CNN training. It contains 456567 images annotated with the 200 object categories and contained objects bounding boxes. For training we use 512x512 fixed size images cropped from ILSVRC images without scaling and 128x128 fixed size images cropped from same dataset for reference image.

3.1 Detection-by-Request learning and testing

We propose the new detection-by-request testing protocols on ILSVRC 2014 for semantic matching algorithms. These protocols presume the estimation of AP (average precision) characteristics (with bounding box Intersection-over-Union $>$ 0.5) for each sample of each class being taken as request (one leaf out). Based on this exemplar-wise information we calculate the total AP and mAP scores for semantic matching algorithm - average and top. The top score is based on the account of detection results of only one most representative request for each class (estimated via testing on the validation set).

In seen classes case it is possible to compare proposed method to classical single shot detectors like YOLO. In this case we cant use one leaf out technique, so we use test subset for request images ad whole testing subset for testing. Our model can run more than 30 FPS on GeForce 1080Ti, so computational speed is comparable to single shot detectors.

3.2 Seen Object Detection Results

For seen object detection testing, we use ILSVRC 2014 dataset and official code for evaluation. Our result is 0.23 mAP. Our mAP is much lower than modern CNN-based object detectors (about 0.7 based on result table for ILSVCR 2017). This relatively low level can be explained by two reasons:
- During training we dont use any types of OHEM (online hard examples mining) or other techniques for result improvements.
- Our CNN based on first generation Google net.

So the results are comparable with first generation of single shot detectors like YOLO. We think with modern CNN and object detection learning techniques like OHEM, Focal loss and modern CNNs like DenseNet semantic matcher can reach state of art mAP on seen objects.

We evaluate reactions of our matcher on different request of one class. We fix a threshold and build precision/recall graph for different requests. As you can see on Figure 3 the quality of request is fundamentally important. In our example precision/recall grown from 0.66/0.19 for worst request to 0.95/0.9 for best request. We have computed average precision and recall with fixed threshold on the 200 seen classes mean by request average precision 0.7, mean by request average recall 0.58. The results of this evaluation are shown in Figure 4.

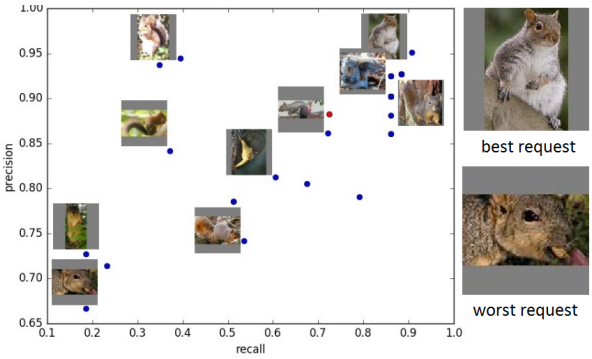


Figure 3. Precision and recall (for fixed threshold) for different requests for class squirrel

If we use best requests for each class according to the F_1-measure (Figure 5) we receive best request average precision 0.77 (more than 10% gain), best request average recall 0.68 (about 15% gain).

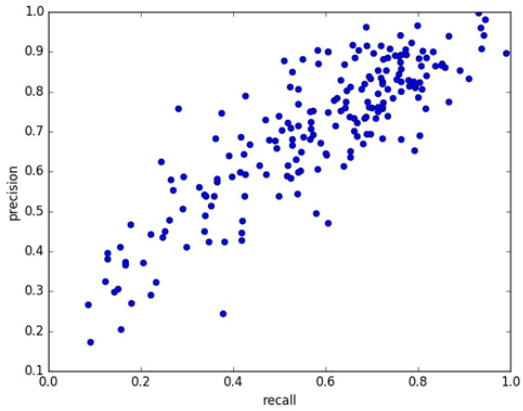


Figure 4. Mean (by request) precision and recall (for fixed threshold) for different classes. Each point represent one of 200 seen classes

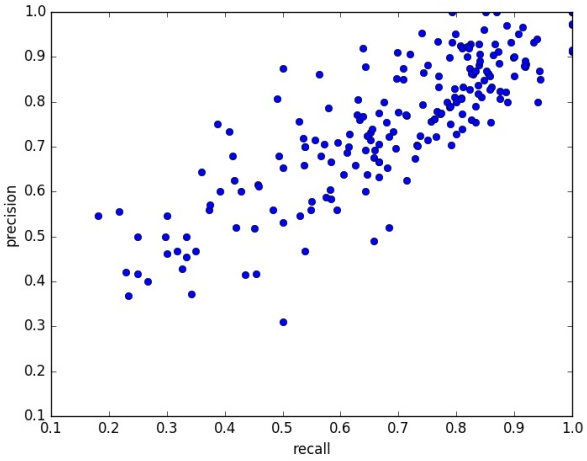


Figure 5. Best (by request) precision and recall (for fixed threshold) for different classes. Each point represent one of 200 seen classes

3.3 Unseen Object Detection Results

We conducted similar studies for unseen classes of images. We selected 90 classes from extended version of ILSVRC 2014 dataset (3632 classes) that were not used at the training stage. First we evaluate reactions of our matcher on different request of one class. To evaluate difference between reactions on seen and unseen classes we build similar graphs as in seen case (Figure 6, 7). In our example precision/recall grown from 0.42/0.24 for worst request to 0.72/0.56 for best request.

We have computed average precision and recall with fixed threshold on the 90 unseen classes mean by request average precision 0.686, mean by request average recall 0.57.

If we use best requests for each class according to the F_1-measure we receive best request average precision 0.73 (more than 6% gain), best request average recall 0.63 (more than 10% gain). For 90 unseen classes we receive 0.21 mAP following our testing protocol. As you can see from results, our CNN has small quality drop for unseen classes in comparison to seen case. For fixed threshold mean by request average precision drops by approx. 2%, mean by request average recall drops also by 2%. Some qualitative examples shown on Figure 8.

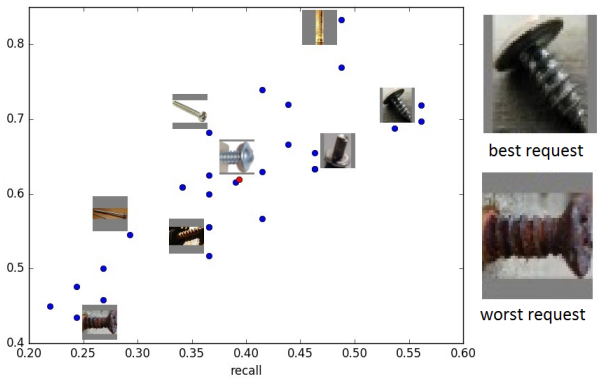


Figure 6. Precision and recall (for fixed threshold) for different requests for unseeen class screw

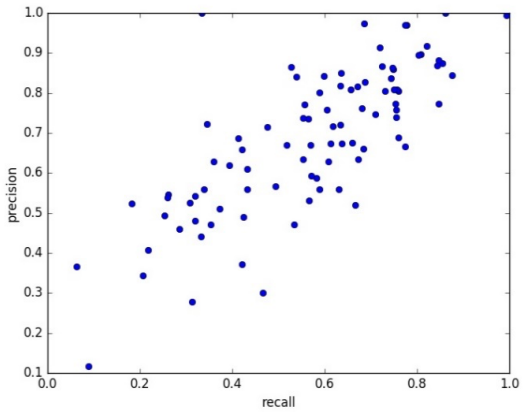


Figure 7. Mean (by request) precision and recall (for fixed threshold) for different classes. Each point represent one of 90 unseen classes

4. CONCLUSION

In this paper, we focus on stating and solving the semantic matching problem. Semantic matching algorithm takes two images (test and request) as input and returns the list of detected objects (bounding boxes) on test image corresponding to semantic class represented by request (sample) image. Such semantic matchers could be of use in the image matching, sensor fusion, landmark detection, change detection, object tracking and other computer vision tasks, which require the (re)identification and localization of 2D image or 3D scene parts under their major shape and pose changes. On the other hand, such semantic matching could provide the object detection for objects of unseen classes represented by only one or too few samples.

We consider the test image as a collection of rectangular fragments and reduce the semantic matching problem to the fragment retrieval-by-request task. Then we apply the ideas of single-shot object detectors like YOLO and SSD in order to achieve the real-time semantic matching tool. In our single-shot semantic matcher CNN architecture the deep features from the top hidden layer of single-shot detector are compared with deep feature vector, which represents the request image. Object detection and request description parts are trained together end-to-end for seen class object detection/localization by request and then applied to both seen and unseen classes.

Our implementation is based on GoogleNet and YOLO/DetectNet

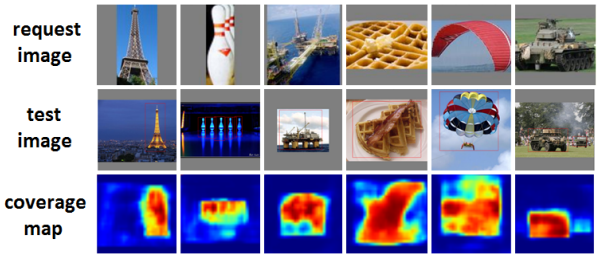


Figure 8. Qualitative Examples for the unseen classes detection: request image, test image and coverage map

(Redmon et al., 2016) CNN architectures. We train our net on the ImageNET-200 dataset (200 seen classes, 90% images for learning, 10% - for testing) and test both on the testing part of ImageNET-200 dataset and on ImageNET-90 dataset (90 unseen classes). We propose the detection-by-request training and testing protocols on ILSVRC 2014 for semantic matching algorithms. These protocols presume the estimation mean average precision characteristics (with bounding box Intersection-over-Union >0.5) for each sample of each class being taken as request. Based on this exemplar-wise information we calculate the total mAP scores for semantic matching algorithm - average and top. The top score is based on the account of detection results of only one most representative request for each class (estimated via testing on the validation set). Our semantic matcher CNN operates at the rate of 30 fps with CPU Core i7 and GPU GeForce GTX 1080 Ti. This computational speed is comparable, for example, to multi-class single shot detectors. So, we can conclude that our semantic matcher can be used as a real-time multi-class object detector with reasonable detection rates for seen classes and possibility for detection of unseen classes.

Model for Caffe framework is available on our GitHub page: https://github.com/NIIAS3050/SingleShortSemanticMatcher. In the future we will try to improve detection rates via OHEM,focal loss and other techniques. We will implement our architecture based on more modern and powerful CNN architectures like DenseNet (Huang et al., 2017) in the schemes of SSD (Liu et al., 2016) or FPN (Lin et al., 2017a) detectors (for non-real-time applications).

ACKNOWLEDGEMENTS

The reported study was funded by Russian Science Foundation (Project No. 16-11-00082) and Russian Foundation for Basic Research (Project 17-29-03185).

REFERENCES

Arandjelovic, R., Gronat, P., Torii, A., Pajdla, T. and Sivic, J., 2016. Netvlad: Cnn architecture for weakly supervised place recognition. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5297–5307.

Babenko, A. and Lempitsky, V., 2015. Aggregating local deep features for image retrieval. In: *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1269–1277.

Bay, H., Tuytelaars, T. and Van Gool, L., 2006. *SURF: Speeded Up Robust Features*. Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 404–417.

Cao, Y., Long, M., Wang, J. and Liu, S., 2017. Deep visual-semantic quantization for efficient image retrieval. In: *2017*

IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 916–925.

Dai, J., Li, Y., He, K. and Sun, J., 2016. R-fcn: Object detection via region-based fully convolutional networks. In: D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon and R. Garnett (eds), *Advances in Neural Information Processing Systems 29*, Curran Associates, Inc., pp. 379–387.

Dai, J., Qi, H., Xiong, Y., Li, Y., Zhang, G., Hu, H. and Wei, Y., 2017. Deformable convolutional networks. In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 764–773.

Dong, L., Yu, X., Li, L. and Hoe, J. K. E., 2010. Hog based multistage object detection and pose recognition for service robot. In: *2010 11th International Conference on Control Automation Robotics Vision*, pp. 2495–2500.

Girshick, R., 2015. Fast r-cnn. In: *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, IEEE Computer Society, Washington, DC, USA, pp. 1440–1448.

Girshick, R., Donahue, J., Darrell, T. and Malik, J., 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In: *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*, CVPR '14, IEEE Computer Society, Washington, DC, USA, pp. 580–587.

Huang, G., Liu, Z., van der Maaten, L. and Weinberger, K. Q., 2017. Densely connected convolutional networks. *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* pp. 2261–2269.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. B., Guadarrama, S. and Darrell, T., 2014. Caffe: Convolutional architecture for fast feature embedding. pp. 675–678.

Lin, T., Dollár, P., Girshick, R. B., He, K., Hariharan, B. and Belongie, S. J., 2017a. Feature pyramid networks for object detection. In: *CVPR*, IEEE Computer Society, pp. 936–944.

Lin, T., Goyal, P., Girshick, R. B., He, K. and Dollár, P., 2017b. Focal loss for dense object detection. In: *ICCV*, IEEE Computer Society, pp. 2999–3007.

Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y. and Berg, A. C., 2016. Ssd: Single shot multibox detector. In: B. Leibe, J. Matas, N. Sebe and M. Welling (eds), *Computer Vision – ECCV 2016*, Springer International Publishing, Cham, pp. 21–37.

Lowe, D. G., 2004. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision* 60(2), pp. 91–110.

Ng, J. Y. H., Yang, F. and Davis, L. S., 2015. Exploiting local features from deep networks for image retrieval. In: *2015 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 53–61.

Radenovic, F., Tolias, G. and Chum, O., 2016. CNN image retrieval learns from bow: Unsupervised fine-tuning with hard examples. In: *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, pp. 3–20.

Redmon, J., Divvala, S. K., Girshick, R. B. and Farhadi, A., 2016. You only look once: Unified, real-time object detection. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pp. 779–788.

Ren, S., He, K., Girshick, R. and Sun, J., 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In: C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama and R. Garnett (eds), *Advances in Neural Information Processing Systems 28*, Curran Associates, Inc., pp. 91–99.

Shen, Z., Liu, Z., Li, J., Jiang, Y., Chen, Y. and Xue, X., 2017. DSOD: learning deeply supervised object detectors from scratch. In: *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pp. 1937–1945.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S. E., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A., 2015. Going deeper with convolutions. In: *CVPR*, IEEE Computer Society, pp. 1–9.

Torii, A., Sivic, J., Okutomi, M. and Pajdla, T., 2015. Visual place recognition with repetitive structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37(11), pp. 2346–2359.

Zepeda, J. and Perez, P., 2015. Exemplar svms as visual feature encoders. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3052–3060.