AUTOMATED OBJECT LABELING FOR CNN-BASED IMAGE SEGMENTATION

A. Novozámský ⋆ *D. Vít* † *F. Šroubek* ⋆ *J. Franc* † *M. Krbálek* † *Z. Bílkova* ⋆ *B. Zitová* ⋆

⋆The Czech Academy of Sciences, Institute of Information Theory and Automation, Prague, Czechia
{novozamsky, sroubekf, bilkova, zitova}@utia.cas.cz
†Czech Technical University in Prague, Faculty of Nuclear Sciences and Physical Engineering, Prague, Czechia
{vitdomin, jiri.franc, milan.krbalek}@fjfi.cvut.cz

ABSTRACT

Deep learning-based methods for classification and segmentation require large training sets. Generating training data is often a tedious and expensive task. In industrial applications, such as automated visual inspection of products in an assemble line, objects for classification are well defined yet labeled data are difficult to obtain. To alleviate the problem of manual labeling, we propose to train a convolutional neural network with an automatically generated training set using a naive classifier with handcrafted features. We show that when the naive classifier has high precision then the trained network has both high precision and recall despite the low recall of the naive classifier. We demonstrate the proposed methodology on real scenario of detecting a car coolant tank. However, the proposed methodology facilitates collection of train data for a wider type of CNN based methods such as near-duplicate image detection or segmenting tampered areas of images.

***Index Terms*—** CNN, SURF, U-net, automated object labeling, image segmentation

1. INTRODUCTION

Deep learning techniques have gained enormous popularity in recent years for their superior performance and versatility. Convolutional neural networks (CNNs) are the most frequently used tool for implementing the deep learning paradigm and their applications range from machine learning, through computer vision, to image/video processing. Machine learning tasks such as classification or segmentation, which can be regarded as a particular problem of classification, are typically solved via supervised learning, i.e. a mathematical model is built from a set of training data that contains both the inputs and the desired labels. Here, a common drawback of CNNs is the need for a large amount of annotated data. Generating such training sets involves a tedious data labeling process often carried out by human experts. Labeling can therefore be an expensive job.

Industrial applications are specific in the sense that the task in question is well defined, e.g. segmenting a particular object that does not change its appearance and shape (Fig. 1), however, it can become dynamic, e.g. objects to be segmented change because the production must react to the evolving market needs. Generating new labeled data, possibly manually, to train CNNs every time the object to be segmented is modified is impractical. One can return to traditional handcrafted features, such as SIFT [1], SURF [2], etc., that require less training data. Here, one object example (template) is often sufficient, especially if the object's appearance is constant. These traditional approaches exhibit a subpar performance and they can be even slower in inference (depending on the size of chosen CNNs). Both drawbacks make traditional approaches unsuitable in industrial applications, where real-time performance and high recall and precision are required.

In this work, we tackle the problem of missing training data for a segmentation CNN. The proposed methodology is based on the notion that CNNs generalize extremely well for objects that change their appearance only slightly. We claim that is it sufficient to generate labeled data automatically using a traditional classifier with handcrafted features. The key idea is that the classifier is designed to be conservative, with high precision and low recall, i.e. the segmented object can be often completely missed in the input data, yet the training set will contain a low number of false positives. Tuning the handcrafted features for new objects is assumed to be a relatively easy task and thus generating new labeled data is cheap. The labeled data are then used for training the segmentation CNN and the property of CNNs to generalize well helps the newly trained CNN to have both precision and recall high compared to the traditional classifier generating the training set.

It is important to note that the proposed methodology is applicable on collecting of train data for various types of CNNs. For instance, CNN-based methods performing near-duplicate image classification and search, typically, requires

This work was supported by Czech Science Foundation grant GA18-05360S, by the project No. TN01000024: National Competence Center – Cybernetics and Artificial Intelligence, from the Technology Agency of the Czech Republic, by the European Union's Horizon 2020 research and innovation program under grant agreement No 825227, and by the Praemium Academiae awarded by the Czech Academy of Sciences.

Fig. 1: Examples of inspected cars on the conveyor belt

a large number of image pairs representing original and the near-duplicated version. Analogically, collecting pixel-level labeled tampered images for training CNNs designed for manipulation detection is a challenging and expensive task. In both mentioned problems, the proposed method can help in increasing the train set size as well as precision/recall of the classifiers.

2. RELATED WORK

Since the algorithm based on CNNs won ImageNet Classification challenge in 2012 [3], similar approaches have not only become very popular, but they have improved to the point where they can outperform a layman in image classification tasks. Segmentation as a part of classification techniques, has many applications in different research areas such as autopilot in self-driving cars (localizing other vehicles, traffic signs, pedestrians, etc.) [4] [5], medical imaging (localizing tumors and other pathologies, nuclei segmentation in the microscopy images, detection of the lips and tongue movements, etc.) [6] [7] [8], or interpreting satellite images and understanding real-world urban scenes [9] [10].

2.1. Segmentation

Segmentation tasks can be divided into three groups – *object localization* where the goal is to find bounding box coordinates for given objects in the image, *semantic segmentation* where each pixel has to be assigned to an appropriate class (sometimes called *pixel-level classification*), and *instance segmentation* where each pixel is assigned not only to a class but also to an individual object [11] [12]. Although our localization solution creates a label box around the object, the used U-net CNN method belongs to semantic segmentation group [13], [14], [15]. Most of these techniques rely on deep networks, where the amount of training data is crucial.

Data labeling is a highly non-trivial and often iterative process as shown for example in self-driving car neural networks training [4]. Thus the main problem in such segmentation approach is to get trusted and well-labeled data set. The manual annotation is the most common way how to perform data labeling. There are many platforms and software tools to create effectively bounding boxes, such as LabelImg, RectLabel, LabelMe, or VIA (VGG Image Annotator). Another possibility is reinforcement learning that enables to train machine learning models by the trial-and-error. Unfortunately reinforcement learning or any public labeled data repository

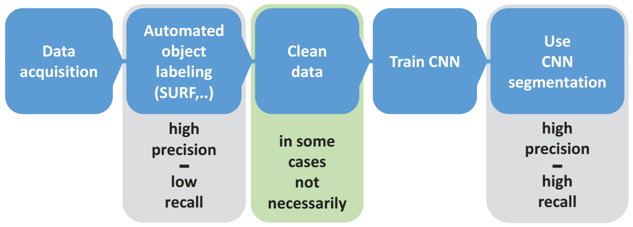
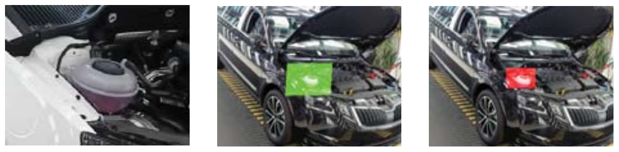


Fig. 2: Our framework for automated object labeling for image segmentation.



(a) template image (b) SURF detection (b) mask for CNN

Fig. 3: Object detection with SURF method.

is not applicable in our problem, where we need to localize very special car components.

2.2. Features for object recognition

There are various approaches to object description used in feature-based image recognition [16]. Their popularity has declined slightly due to CNNs, which tend to be faster and work with higher precision and variability of use. On the other hand, their irreplaceable significance will prove in the case of object detection from one single image template. In addition, it ought to be mentioned that each of the presented methods can be beneficial in different applications.

The most commonly used methods include: SIFT [1]; MSER [17]; FAST [18]; SURF [2]; BRIEF [19]; and BRISK [20]. In our detection of engine parts we achieved the best results with the SURF method, especially thanks to its invariance to scale and rotation.

3. OUR FRAMEWORK

Our framework consists of five phases, as depicted in Fig. 2. The process starts with the data acquisition, a crucial part which can influence the outcome of the whole segmentation. In the second step an appropriate type of features is selected (here the SURF [2] were used), evaluated, and objects are automatically labeled using these features. *Clean data* phase is applied only when needed, i.e. there are object occlusions or other types of unwanted artifacts. In these cases, such images are omitted from further processing. The latter two steps follow a common CNN segmentation solution, where CNN is trained using data from the second phase and then used for the segmentation.

3.1. U-net architecture design

U-net, presented in 2015 by O. Ronneberger et al. [21] was designed to perform fast and precise segmentation. It is based on the fully convolutional network [22], but apart from the contracting path to capture contexts it has also a symmetric expanding path for precise localization. Unlike the Ronneberger's group, we do not use cropping in the contracting step and every 3x3 convolution produces the same size as their inputs *(padding=same)*. Therefore we get the mask at the output equally large as an input image, which is 512x512 pixels. Our modified U-net architecture is illustrated in Figure 5. The activation function ReLU, the 2x2 max pooling operation, and the total of 27 convolution layers are preserved according to the original design. Our U-net network was built in Keras, the high-level API in TensorFlow, and it was trained on NVIDIA Quadro P6000 graphics card. Obtaining data for training and validation are described in Chapter 4.2.

4. REAL EXPERIMENT

The data were collected under the project for Škoda-auto aimed at the automatic inspection of vehicle production. The requested goal of the project was to monitor the level of coolant in the tank of a car engine. On the control block KB8, the cars are parked manually on the conveyor strand by employees, the bonnet is lifted and the whole car is thoroughly visually inspected for defects such as misaligned interior padding, not functioning indicators on a dashboard, discharged battery or other problems that might have occurred during assembly and have been missed during previous inspections. One of such problems is the mentioned lack of coolant, which frequently occurs in cars that stayed in the factory longer than expected, e.g. due to a replacement of faulty parts.

4.1. Data acquisition

We had to locate the spot for camera montage as close as possible to the inspected car while not obstructing any activities of employees. We found such a place on the left side panel at approximately eye height and pointed the camera slightly forwards so the inside of the engine space could be visible for about 20 seconds. We used an ordinary webcam *Logitech BRIO 4K Stream Edition*. A continuously running script for data acquisition worked for several weeks with 1 fps in FullHD resolution and saved data in 10-minute intervals. All frames were recorded in motion JPEG (M-JPEG) and after that encoded in HEVC h265.

We avoided recording duplicate frames, when the conveyor was stopped for a break or a shift change, by automatically stopping recording in the script at regularly scheduled breaks. There were a few instances when the line was unexpectedly stopped and one car remained under our cameras for half an hour, while people were also passing in front of it. We had to account for these unusual cases and build our processing system sufficiently robust.

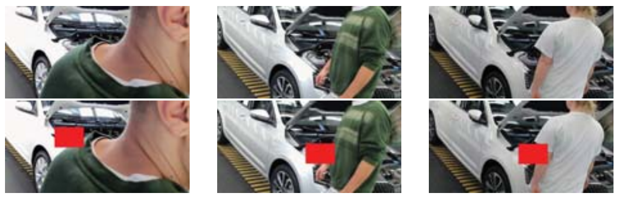


Fig. 4: Examples of occlusion of the coolant in the wild training data. The mask for CNN is marked with the red rectangle, see Section 4.2.

4.2. Automated data labeling and cleaning process

Using SURF features for detecting only the coolant tank yielded poor results as the cropped area is very small with low resolution. The other algorithms mentioned in Section 2.2 work even less satisfactory in such small areas. Therefore, we extended the template to the size shown in Fig. 3. This extended region of interest is used for counting reference points and Fig. 3b demonstrates the output of SURF in one frame of analyzed videos. If the matching algorithm detects the template in an image, we check the size and shape of the marked area. In the next step, we exploit the fact that the coolant has a strongly distinguishable purple color. Using the HSV space, we are able to threshold the hue and separate purple areas from the rest of the image. Lastly, we use morphological operation to join areas together. After the coolant localization, we count the center of gravity and the final labeled segment that enters the training process of the CNN is a cropped area around the liquid centroid: 10 pixels below, 50 pixels above and 40 pixels on each side; see the red rectangle in Fig. 3c.

This way we generated a complete set of 26,000 images, from which we manually removed faulty cases and obtained a clean set of 20,500 images. Most of the faulty data were caused by occlusion and accounted for 14.58% of the total number of automatically generated images, examples are given in Fig. 4. Than we divided randomly the clean set to 18,000 images for testing *(DTest)*, 2,000 images for training *(DTrain)*, and 500 images for validation *(DVal)*. Moreover we randomly chose from the complete set of 26,000 images another set of 2,000 for training *(DTrainW)* and 500 for validation *(DValW)*. We refer to this training set that contains also faulty cases as the *wild data*. To validate both algorithm independently, we selected and labeled manually 1,000 images as *ground-truth data*.

4.3. Results

The experiment consists of three parts. First, we randomly mixed and divided the data in *DTrain* into 20 subsets each

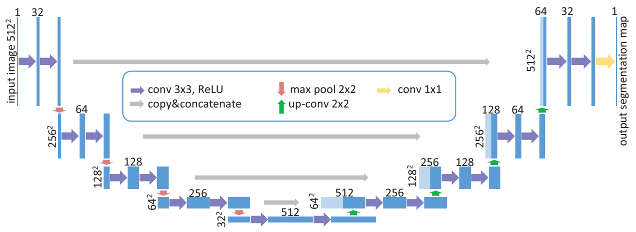


Fig. 5: U-net architecture inspired by Ronneberger et al. [21]. In our case, the U-net is symmetric ant the output image is the same size like input image, namely 512x512 pixels.

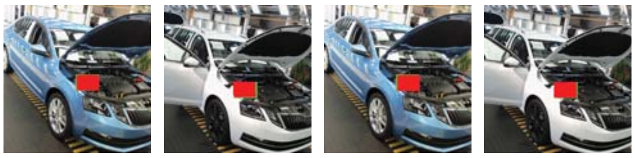


Fig. 6: A comparison of SURF (green) and CNN (red) segmentation in *DTest* with training on *DTrain* (first two images) and *DTrainW* (second two).

with 100 images. Similarly, we randomly divided the validation data *DVal* into subsets of 25 images. This way we simulate a situation when we do not have enough labeled data from the feature-base algorithm. Than we train the U-net with the subsets and got 20 models for segmentation. For evaluation, we used the whole *DTest* set in all the cases. The obtained precision and recall [23] summarized in Table 1 shows that the excellent performance is quickly reached and for the training sets of size 800 and more there is practically no improvement.

Second, the same process we repeated with *DTrainW*, where faulty data were included, and the precision and recall were counted on the same data as before, i.e. *DTest*. The output accuracy is again summarized Table 1. The results are almost the same meaning that the CNN is stable even if we train with the *wild data* in *DTrainW* and *DValW*. In all the cases, we set the maximum number of training epochs to 200 and employed an early-stopping criterion when the value of the loss function stopped changing.

In the third part, we compare both algorithms on the ground-truth data, see Table 2.

Table 1: Results of our framework for CNN segmentation. N = size of the training set; n = number in the validation set; P, R = precision and recall of the clean data; P_w, R_w = precision and recall of the wild data.

N	n	P	P_w	R	R_w
400	100	96.38	96.82	80.55	94.36
800	200	98.42	97.99	99.13	99.51
1200	300	98.26	98.18	99.42	99.29
1600	400	98.85	98.01	98.67	98.63
2000	500	98.69	97.22	99.27	99.53

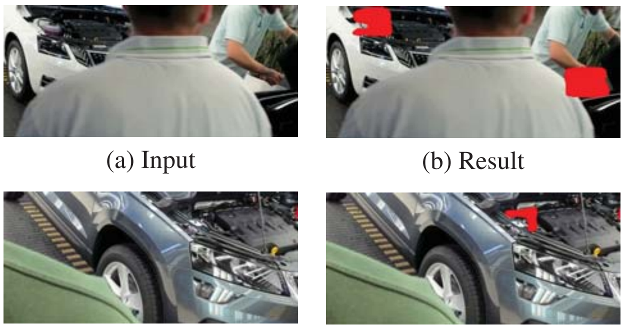


Fig. 7: False positives from CNN segmentation

Table 2: A comparison of feature-base methods and our framework on 1000 manually labeled images - *ground-truth*.

	BRISK	FAST	MSER	SURF	CNN
Precision [%]	100.00	100.00	62.4	100.00	**100.00**
Recall [%]	0.80	0.70	1.50	1.20	**99.50**

5. CONCLUSION

In the proposed paper we described an approach which helps to overcome the lack of annotated data in the CNN-based segmentation. The combination of the low recall & high precision feature-based labeling and adapted U-net CNN formed a segmentation algorithm with high precision and high recall (see Section 4). The robustness of the labelling method was further improved by using the larger size of ROIs for feature evaluation and object detection, followed by object thresholding in the HSV color space to extract the best possible object representation for CNN training.

The achieved precision & recall values (see Tables 1-2) demonstrate that the U-net was a good choice for a segmentation task when objects to be detected change their appearance just slightly. In similar industry applications only small rotations and/or negligible affine degradations can be assumed.

6. REFERENCES

[1] D. G. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, Sep. 1999, vol. 2, pp. 1150–1157 vol.2.

[2] H. Bay, A. Ess, T. Tuytelaars, and L.V. Gool, "Speeded-up robust features (surf)," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346 – 359, 2008, Similarity Matching in Computer Vision and Multimedia.

[3] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural net-

works," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., pp. 1097–1105. Curran Associates, Inc., 2012.

[4] A. Karpathy, "Multi-task learning in the wilderness," 06 2019.

[5] K. Behrendt, L. Novak, and R. Botros, "A deep learning approach to traffic lights: Detection, tracking, and classification," 05 2017, pp. 1370–1377.

[6] C. Chen, X. Liu, M. Ding, J. Zheng, and J. Li, "3d dilated multi-fiber network for real-time brain tumor segmentation in mri," in *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019, Lecture Notes in Computer Science*, Cham, 10 2019, Springer International Publishing.

[7] Z. Bílková, A. Novozámský, A. Domínec, Š. Greško, B. Zitová, and M. Paroubková, "Automatic evaluation of speech therapy exercises based on image data," in *Image Analysis and Recognition*, Cham, 2019, pp. 397–404, Springer International Publishing.

[8] P. Mlynarski, H. Delingette, A. Criminisi, and N. Ayache, "3d convolutional neural networks for tumor segmentation using long-range 2d context," *Computerized Medical Imaging and Graphics*, vol. 73, pp. 60–72, 04 2019.

[9] Z. Zhengxin and L. Qingjie, "Road extraction by deep residual u-net," *IEEE Geoscience and Remote Sensing Letters*, vol. PP, 11 2017.

[10] C. Marius, O. Mohamed, R. Sebastian, R. Timo, E. Markus, B. Rodrigo, F. Uwe, R. Stefan, and S. Bernt, "The cityscapes dataset for semantic urban scene understanding," 06 2016, pp. 3213–3223.

[11] H. Kaiming, G. Georgia, D. Piotr, and G. Ross, "Mask r-cnn," 10 2017, pp. 2980–2988.

[12] W. Chen, X. Gong, X. Liu, Q. Zhang, Y. Li, and Z. Wang, "Fasterseg: Searching for faster real-time semantic segmentation," 12 2019.

[13] R. Li, W. Liu, L. Yang, S. Sun, W. Hu, F. Zhang, and W. Li, "Deepunet: A deep fully convolutional network for pixel-level sea-land segmentation," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. PP, 09 2017.

[14] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," 07 2018, Lecture Notes in Computer Science, vol 11045. Springer.

[15] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, pp. 843–848, 2018.

[16] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, no. 27, pp. 1615–1630, 2005.

[17] J. Matas, O. Chum, M. Urban, and T. Pajdla, "Robust wide-baseline stereo from maximally stable extremal regions," *Image and Vision Computing*, vol. 22, no. 10, pp. 761 – 767, 2004, British Machine Vision Computing 2002.

[18] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision – ECCV 2006*, Aleš Leonardis, Horst Bischof, and Axel Pinz, Eds., Berlin, Heidelberg, 2006, pp. 430–443, Springer Berlin Heidelberg.

[19] M. Calonder, V. Lepetit, C. Strecha, and P. Fua, "Brief: Binary robust independent elementary features," 09 2010, vol. 6314, pp. 778–792.

[20] S. Leutenegger, M. Chli, and R. Y. Siegwart, "Brisk: Binary robust invariant scalable keypoints," in *2011 International Conference on Computer Vision*, Nov 2011, pp. 2548–2555.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, Cham, 2015, pp. 234–241.

[22] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2015, pp. 3431–3440.

[23] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861 – 874, 2006, ROC Analysis in Pattern Recognition.