Patch-wise Object Recognition for a Mobile Robot by means of a Convolutional Neural Network

* Solvi Arnold (Shinshu University) Kimitoshi Yamazaki (Shinshu University)

1. Introduction

This paper introduces our work on patch-wise classification of objects and surfaces by means of a convolutional neural network, intended for use in mobile robots. Convolutional neural networks have proven to be highly effective for object recognition. However, application to mobile robots brings with it a characteristic set of demands and limitations quite different from the conditions these techniques are usually being developed under. In particular, it requires real-time operation under visually challenging conditions, such as low lighting, motion blur, and image deformations introduced by wide lenses (cf. Figure 1, upper panel). Using an image dataset with these characteristics as our testbed, we explore the viability of a convolutional neural network-based approach.

2. Image classification using convolution nets

Recent years have seen convolutional neural networks take a central position in the field of visual recognition, achieving state-of-the-art results on many problems. A common application is image classification, where networks are trained to identify the object depicted in an image [1]. In this case network output is a vector giving for each class under consideration the estimated likelihood of the image belonging to that class. Recently, pixel-wise classification has been gaining attention [2][3]. Here nets are trained to guess the class-membership of every individual pixel in an image. The advantage of a pixel-wise approach is that it can handle scenes with multiple objects naturally, and captures spatial structure. An obvious disadvantage is computational cost: typically the output has even higher dimensionality than the input (namely image resolution multiplied by the number of classes). Also, as results for individual pixels tend to be jittery, post-processing to integrate pixel classifications over larger areas (e.g. super pixels) is necessary for smooth results, adding further complexity and computational cost to the system.

Many practical use cases would be best served with a solution that strikes a balance between these approaches. The mobile robots we target present such a case: object recognition ties into scene understanding and movement planning, hence spatial information is of great importance, but pixel-level precision is not. Furthermore, for real-time operation it is crucial that the eventual system can handle a video stream at good temporal resolution on systems with limited computational resources. With these considerations in mind, we explore an approach where classification is performed at an intermediary patch-size, much smaller than the full image, but still covering a large number of pixels.

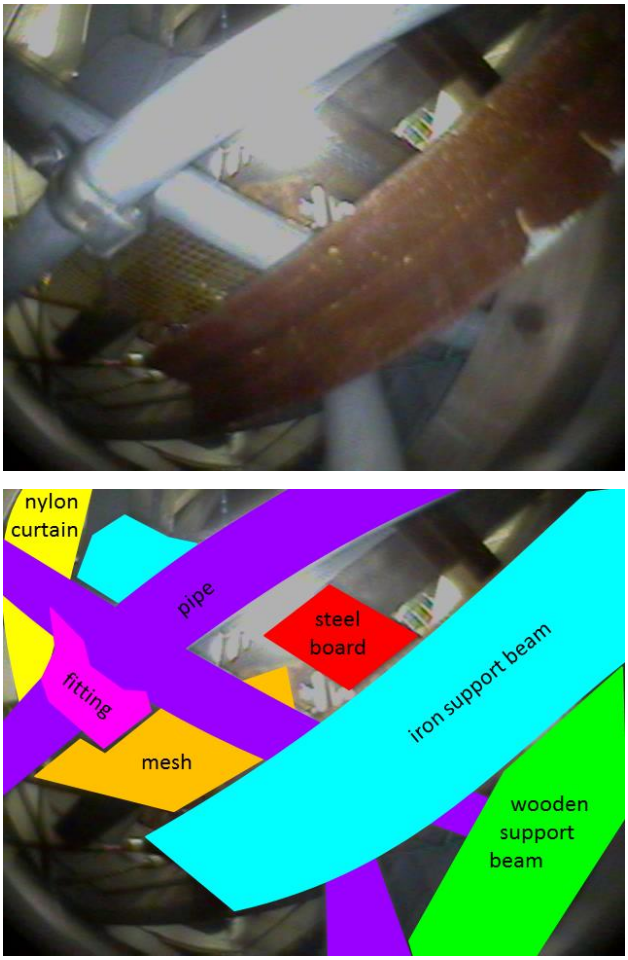

Figure 1. Example image and its (partial) annotation (polygon overlay method, see section 3). Note the blur and strong fisheye deformation.

3. Dataset

Here we describe the preparation of the dataset. We obtained 67 minutes of raw footage from a remote-controlled indoor robot. We extracted still-frames at 10-second intervals for annotation. Pruning of near-duplicates (at times the robot sat still for extended periods of time)

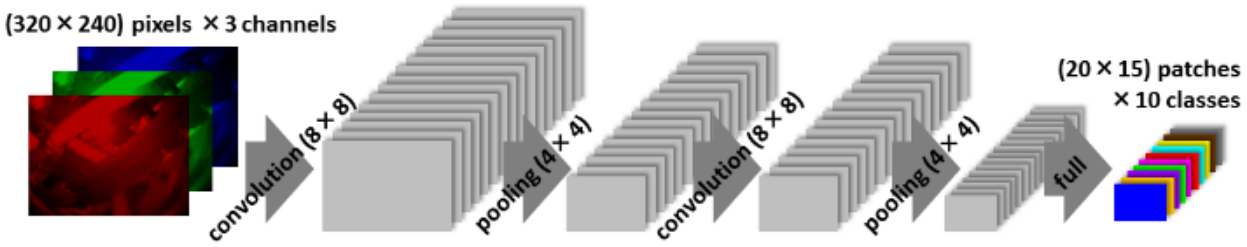

Figure 2. Convolutional neural network architecture.

left 203 images. We added horizontally flipped duplicates of every image to increase the dataset size[1], and split the resulting set of 406 images into a training set of 346 images and a test set of 60 images. We determined 10 object/surface classes, shown in table 1. Each image was annotated with either for the following methods:

1) Superpixel marking: After segmenting the image into superpixels using the turbo pixels algorithm [4], we manually assigned a class to every superpixel deemed to belong to one of our target classes.
2) Polygonal overlay: We marked objects by overlaying them with coloured polygons (with different colours mapping to different classes).

1	inner surface of pipe
2	iron mesh
3	iron pipe
4	wooden support beam
5	pipe fitting
6	steel board
7	iron support beam
8	nylon curtain
9	ballast (ground surface)
10	concrete

Table 1. Objects & surfaces targeted for classification.

Next we computed learning targets from the annotated images as follows:

1) We divide the image (grid-wise) into 20x15 equal-sized patches.
2) Every patch is given a vector, with each value in the vector corresponding to one class. We iterate over the pixels in the patch, and for each pixel that was assigned a class in the annotation, we add +1 to the vector element corresponding to that class.
3) We divide the vector element-wise by the size of the patch (in pixels). The resulting vector gives a distribution over classes for the patch (e.g. 0.61 part nylon curtain and 0.27 part mesh). Note that the elements of a vector need not sum to one, as we allow for non-classified pixels.
4) The resulting 3D matrix (= 2D grid of class vectors) is used as the target for learning on this image. Note that this implies that the target for learning is not a classification of the patch as belonging to one particular class, but instead the class distribution on the patch.

We used an intermediate patch size of 256 pixels (1/300 of the image), but consider what would happen as we change this scale. If we were to match patch size to image size (i.e. one big patch) we would be learning class distributions over whole images, with no spatial information. Conversely, if we would set the patch size to cover just a single pixel, we would learn pixel-wise singular classifications (as individual pixels belong to at most one class, their class distribution reduces to a versor of the class space, or a zero vector for pixels lacking annotation).

4. Convolutional neural network

We used a four-layer neural network, consisting of a 3-dimensional (width × height × RGB) input layer receiving the input image, two convolution layers using a rectified linear activation function, and finally a fully connected output layer using a sigmoid activation function. Both of the convolutional layers also implement a pooling operation on their output. Figure 2 illustrates the network structure. Images were rescaled to the net's input layer resolution of 320 by 240 (the original footage varied between 640 by 480 and 720 by 540 pixels). We use stochastic gradient descent as our learning algorithm, with a learning rate of 0.05. We let the net learn for 5,000,000 image presentations (although performance levels off well before the end of training).

The neural net was implemented using the Pylearn2 machine learning library [5], and was run on a single NVidia GTX970 GPU.

[1] When dealing with an actually symmetric task it is of course more elegant to impose symmetric linking on the neural network's connection weight vectors, but we do not assume task symmetry in our goal application.

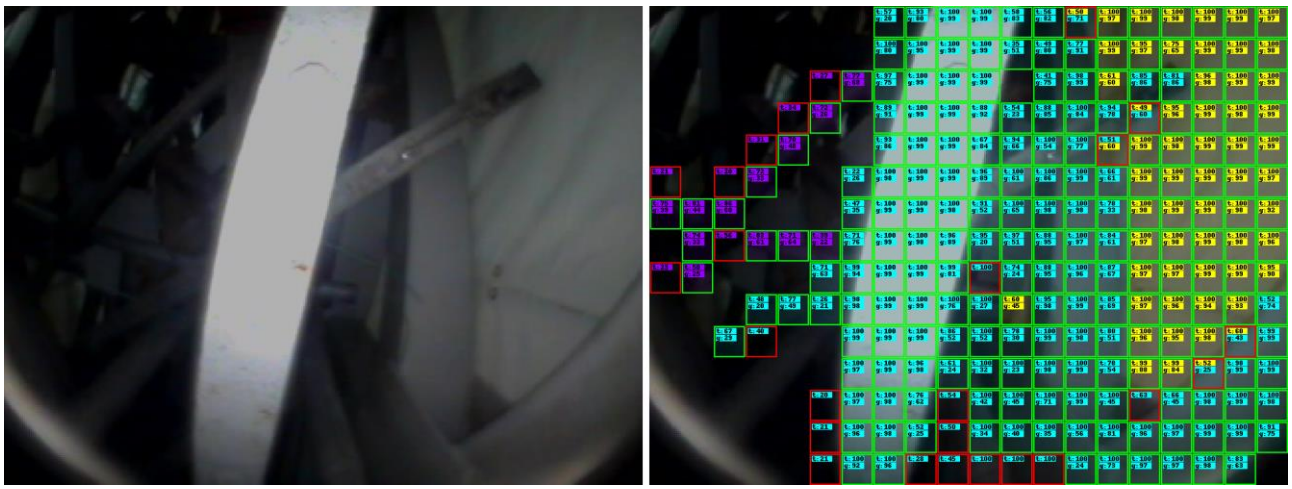

Figure 3. Example input (left) and its patch-wise classification by the neural net (right). The upper box in each patch indicates its dominant class in the annotation. The lower box indicates the dominant class in the net's output for that patch. See Table 2 for colour legend. Dominant values below 0.2 were discarded. Patches outlined in green (red) indicate a match (mismatch) between dominant class in annotation and network output.

5. Results

We tracked two performance measures, which we will call *class-distribution accuracy* and *best-guess accuracy*, for both data sets (training data and test data). Results are given in Table 2. The class-distribution accuracy measure simply shows how close the net's output is to the learning target, (i.e. the patch-wise class-distribution). The best-guess accuracy measure is slightly more lenient, showing the rate at which the dominant class in the net's output for a given patch matches the dominant class in the actual distribution for that patch (i.e. the classification accuracy after we force output and target into a one-class-per-patch format). The latter score ignores patches without annotation as targets are undefined there. Scores on the training set verify that the system is successfully learning to replicate the training targets. Results on the test set show how well the net generalizes to new images. Figure 3 shows a representative example of classification on an image form the test set. The performance discrepancy between the sets suggests that the net is overfitting on the training set. However, the net still achieves decent scores on the test set, showing ability to generalize to new images. We expect these scores to go up further as we increase the dataset size and tune the system's parameters.

	Training set	**Test set**
Class-distribution accuracy	96%	63%
Best-guess accuracy	98%	67%

Table 2. Performance measurements

6. Conclusions & future work

We have presented a patch-wise classification approach for mobile robots using a convolutional neural network. Despite the early stage of development and the limitations of the dataset, we obtained results that support the viability of this approach. We are considering various avenues for further improvement of the system, in particular the use of temporal coherence [6]. As we are targeting the visual stream from a mobile robot, we can exploit the similarity of subsequent images to build up accurate classifications over multiple frames, and we can exploit the difference between subsequent frames as a source of information about scene structure.

Acknowledgements

This work was partly funded by ImPACT Program of the Council for Science, Technology and Innovation (Cabinet Office, Government of Japan). We appreciate Prof. Tadokoro and Prof. Konyo for providing image data.

References

[1] Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E. Hinton. "Imagenet classification with deep convolutional neural networks." *Advances in neural information processing systems*, 25: 1097-1105, 2012.

[2] Camille Couprie, Clément Farabet, Laurent Najman, Yann LeCun: " Indoor Semantic Segmentation using depth information", International Conference on Learning Representations (ICLR2013), arXiv:1301.3572v2, 2013.

[3] Clement Farabet, Camille Couprie, Laurent Najman and Yann LeCun: "Learning Hierarchical Features for Scene

Labeling", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8): 1915-1929, 2013.

[4] Alex Levinshtein, Adrian Stere, Kiriakos N. Kutulakos, David J. Fleet, Sven J. Dickinson, and Kaleem Siddiqi: "TurboPixels: Fast Superpixels Using Geometric Flows". IEEE Transactions on Pattern Analysis and Machine Intelligence, 31(12): 2290-2297, 2009.

[5] Ian J. Goodfellow, David Warde-Farley, Pascal Lamblin, Vincent Dumoulin, Mehdi Mirza, Razvan Pascanu, James Bergstra, Frédéric Bastien, and Yoshua Bengio: "Pylearn2: a machine learning research library", *arXiv:1308.4214*, 2013.

[6] Hossein Mobahi, Ronan Collobert, and Jason Weston: "Deep learning from temporal coherence in video", *Proceedings of the 26th Annual International Conference on Machine Learning* (ICML '09), pp.737-744, 2009. DOI=10.1145/1553374.1553469