Hierarchical Scene Coordinate Classification and Regression for Visual Localization —Supplementary Material—

Xiaotian Li[1] Shuzhe Wang[1] Yi Zhao[1] Jakob Verbeek[2] Juho Kannala[1]
[1]Aalto University [2]Facebook AI Research

In this supplementary material, we provide more details on network architecture, training procedure, and other experimental settings. Additional qualitative results are presented at the end.

A. Main Experiment Details

In this section, we present the experiment details on 7-Scenes, 12-Scenes, Cambridge Landmarks, and the combined scenes.

A.1. Network Architecture

We use a similar VGG-style [11] architecture as DSAC++ [2] as the base regression network, except we use ELU activation [4] instead of ReLU [7]. As mentioned in the main paper, we found that the plain regression network is faster to train with ReLU, while our network which has additional conditioning layers and classification branches works better with ELU. Conditioning layers and generators, as well as two additional classification branches, are added upon the base network for our 3-level hierarchical network which is used in the main experiments.

There are three convolutional layers with stride 2 in the regression base network. The output resolution of the regression branch is thus reduced by a factor of 8. Strided convolution, dilated convolution and upconvolution are used in the two classification branches to enlarge the receptive field and preserve the output resolution. The predicted classification labels are converted into one-hot format before being fed into the generators. If more than one label map used as input to a conditioning generator, the label maps are concatenated.

The detailed architecture is given in Fig. 1. For experiments on 7-Scenes, 12-Scenes, Cambridge Landmarks, we use the same network architecture. For experiments on the combined scenes, we increased the number of channels for certain layers and added two more layers in the first conditioning generator. The additional layers are marked in red, and the increased channel counts are marked in red, blue and purple for i7-Scenes, i12-Scenes and i19-Scenes, respectively. The more compact architecture for the experiments (*Ours capacity-* in Table 2 and Fig. 3 of the main paper) on the combined scenes is illustrated in Fig. 2. In the case we use different numbers of channels for a convolutional layer, the channel counts are marked in red, blue and purple for i7-Scenes, i12-Scenes and i19-Scenes respectively.

As in DSAC++ [2], our network always takes an input image of size 640×480. We follow the same practice to resize larger images as [2]. That is, the image is first rescaled to height 480. If its width is still larger than 640, it is cropped to width 640. Central cropping is used at test time, and random horizontal offsets are applied during training.

A.2. Network Training

For 7-Scenes and 12-Scenes, our network is trained from scratch for 300K iterations with an initial learning rate of 10^{-4} using Adam [6], and the batch size is set to 1. We halve the learning rate every 50K iterations for the last 200K iterations. For the Cambridge Landmark dataset, the dense reconstructions are far from perfect. The rendered ground truth scene coordinates contain a significant amount of outliers, which make the training difficult. Therefore, we train the network for 600K iterations for experiments on this dataset. For the combined scenes, the network is trained for 900K iterations.

As mentioned in the main paper, we found that the accuracy of the final regression predictions is critical to high localization performance. Therefore, a larger weight is given to the regression loss term. The weights for the classification loss terms w_1, w_2 are set to 1 for all scenes. The weight for the regression loss term is set to 100,000 for the three combined scenes and 10 for the other datasets.

For data augmentation, affine transformations are applied to each training image. We translate, rotate, scale, shear the image by values uniformly sampled from [-20%, 20%], $[-30°, 30°]$, [0.7,1.5], $[-10°, 10°]$, respectively. In addition, we also augment the images with additive bright-

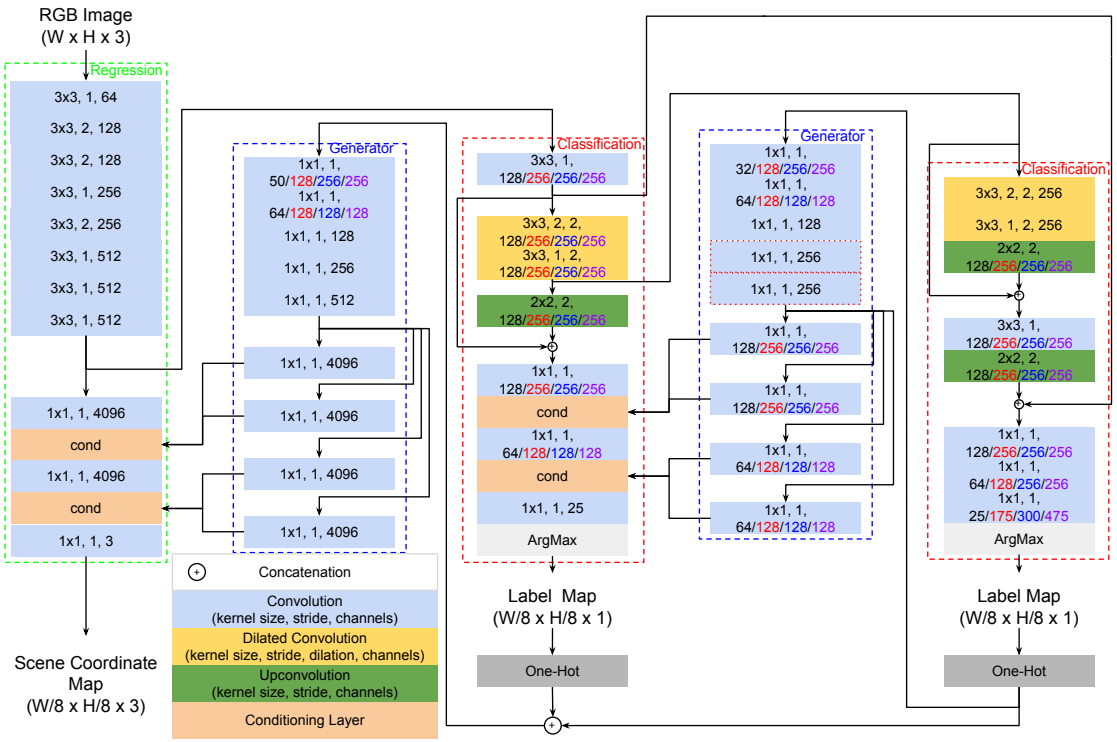

Figure 1. Detailed network architecture.

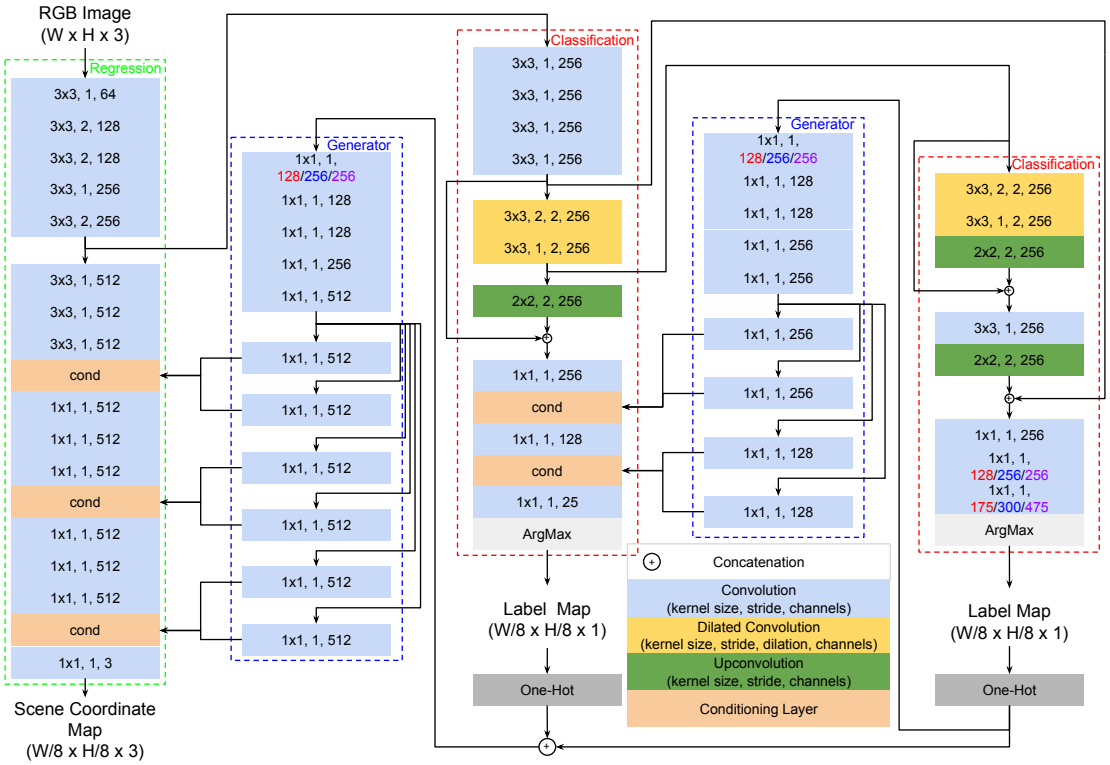

Figure 2. The more compact architecture of *Ours capacity-*.

ness changes uniformly sampled from [-20, 20]. When training without data augmentation, as with [2], we randomly shift the image by -4 to 4 pixels, both horizontally and vertically, to make full use of data, as the output resolution is reduced by a factor of 8.

A.3. Pose Optimization

At test time, we follow the same PnP-RANSAC pipeline and parameter settings as in [2]. The inlier threshold is set to $\tau = 10$ for all the scenes. The softness factor is set to $\beta = 0.5$ for the soft inlier count [2]. A set of 256 initial hypotheses are sampled, and the refinement of the selected hypothesis is performed until convergence for a maximum of 100 iterations.

A.4. Run Time

The network training takes ≈12 hours for 300K iterations on an NVIDIA Tesla V100 GPU, and ≈18 hours on an NVIDIA GeForce GTX 1080 Ti GPU.

At test time, it takes ≈100ms for our method to localize an image on an NVIDIA GeForce GTX 1080 Ti GPU and an Intel Core i7-7820X CPU. Scene coordinate prediction takes 50-65ms depending on the network size. Pose optimization takes 30-60ms depending on the accuracy of the predicted correspondences.

B. Experiments on the Aachen Dataset

In this section, we provide the experimental details on the Aachen dataset.

B.1. Ground Truth Labels

Similar to the experiments on the other datasets, to generate the ground truth location labels, we run hierarchical k-means clustering on the sparse point cloud model used in [8], which is built with COLMAP [9, 10] using SuperPoint [5] as local feature detector and descriptor. For this dataset we adopt a 4-level classification-only network. We also experimented with two classification-regression networks, but the 4-level classification-only network works better (see Table 5in the main paper). For the 4-level classification-only network, we use four-level hierarchical k-means with the branching factor set to 100 for all levels. This results in ≈685K valid clusters at the finest level, with each of them containing only a single 3D point. For the experiments with the 4-level classification-regression network and the 3-level classification-regression network, we use three-level and two-level hierarchical k-means with the same branching factor setting (100 for all levels), respectively.

B.2. Network Architecture

As stated in the main paper, for the experiments on the Aachen dataset, we use a list of sparse features as input to the network, rather than a regular RGB image. Due to the sparse and irregular format of the input, we use 1×1 convolutional layers in the network. We add a dummy spatial dimension to the input, *i.e*. we use a descriptor map of size N$\times$1$\times$256 as input. In addition, there are no shared layers between different levels. To use image-level contextual information, every output layer including the first one is also conditioned on an image ID. To achieve this, the encoded image ID is concatenated with the label maps (if available) and then fed into the conditioning parameter generators. As mentioned in the main paper, during training, we use the ID of the training image. At inference time, we adopt NetVLAD [1] for global image retrieval, and we use the ID of a retrieved image. The detailed architecture of the 4-level classification-only network is given in Fig. 3. For the 4-level classification-regression network, we simply change the last classification layer to a regression output layer. For the 3-level classification-regression network, one classification level is further removed.

B.3. Network Training

The network is trained from scratch for 900K iterations with an initial learning rate of 10^{-4} using Adam [6], and the batch size is set to 1, similar to the previous experiments. We halve the learning rate every 50K iterations for the last 200K iterations. As in [3, 8], all images are converted to grayscale before extracting the descriptors. Random affine transformations, brightness and contrast changes are also applied to the images before the feature extraction. During training, we ignore the interest point detection, and a descriptor is extract from the dense descriptor map if it has an available corresponding 3D point in the spare 3D model. Following [8], before extracting the NetVLAD [1] and SuperPoint [5] features, the images are downsampled such that largest dimension is 960. At test time, for SuperPoint, Non-Maximum Suppression (NMS) with radius 4 is applied to the detected keypoints and 2K of them with the highest keypoint scores are used as the input to our network.

B.4. Pose Optimization

We follow the PnP-RANSAC algorithm as in [8] and the same parameter settings are used. The inlier threshold is set to $\tau = 10$, and at most 5,000 hypotheses are sampled if no hypotheses with more than 100 inliers are found. Note that the pose optimization is applied independently for all the top-10 retrieved database images.

B.5. Run Time

The network training takes 2-3 days on an NVIDIA Tesla V100/NVIDIA GeForce GTX 1080 Ti GPU. On an NVIDIA GeForce GTX 1080 Ti GPU and an Intel Core i7-7820X CPU, it takes ≈1.1/1.4s (Aachen Day/Aachen Night) for our method to localize an image. It takes

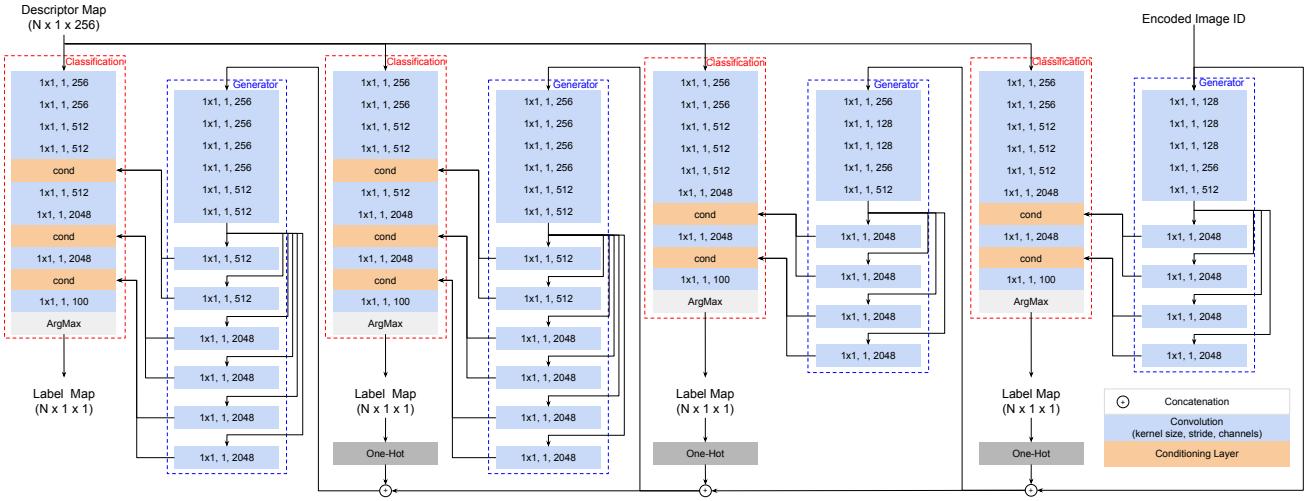

Figure 3. Detailed network architecture of the 4-level classification-only network for the Aachen dataset experiments.

≈170ms to extract the global and local descriptors. Scene coordinate prediction takes ≈280ms (10×28ms) and pose optimization takes ≈600/900ms (10×60/90ms) (Aachen Day/Aachen Night). The time needed for global descriptor matching and the simple pre-RANSAC filtering is negligible.

C. Additional Qualitative Results

We show in Fig. 4 the quality of scene coordinate predictions for test images from 7-Scenes/i7-Scenes, and compare our method to the regression-only baseline. The scene coordinates are mapped to RGB values for visualization.

We show in Fig. 5 the quality of scene coordinate predictions for the Aachen dataset experiments. The scene coordinate predictions are visualized as 2D-2D matches between the query and database images. We show only the inlier matches.

References

[1] Relja Arandjelović, Petr Gronat, Akihiko Torii, Tomás Pajdla, and Josef Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 3

[2] Eric Brachmann and Carsten Rother. Learning less is more - 6D camera localization via 3D surface regression. In *CVPR*, 2018. 1, 3

[3] Eric Brachmann and Carsten Rother. Expert sample consensus applied to camera re-localization. In *ICCV*, 2019. 3

[4] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). *arXiv:1511.07289*, 2015. 1

[5] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. SuperPoint: Self-supervised interest point detection and description. In *CVPR Workshops*, 2018. 3

[6] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv:1412.6980*, 2014. 1, 3

[7] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010. 1

[8] Paul-Edouard Sarlin, Cesar Cadena, Roland Siegwart, and Marcin Dymczyk. From coarse to fine: Robust hierarchical localization at large scale. In *CVPR*, 2019. 3

[9] Johannes Lutz Schönberger and Jan-Michael Frahm. Structure-from-motion revisited. In *CVPR*, 2016. 3

[10] Johannes Lutz Schönberger, Enliang Zheng, Marc Pollefeys, and Jan-Michael Frahm. Pixelwise view selection for unstructured multi-view stereo. In *ECCV*, 2016. 3

[11] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv:1409.1556*, 2014. 1

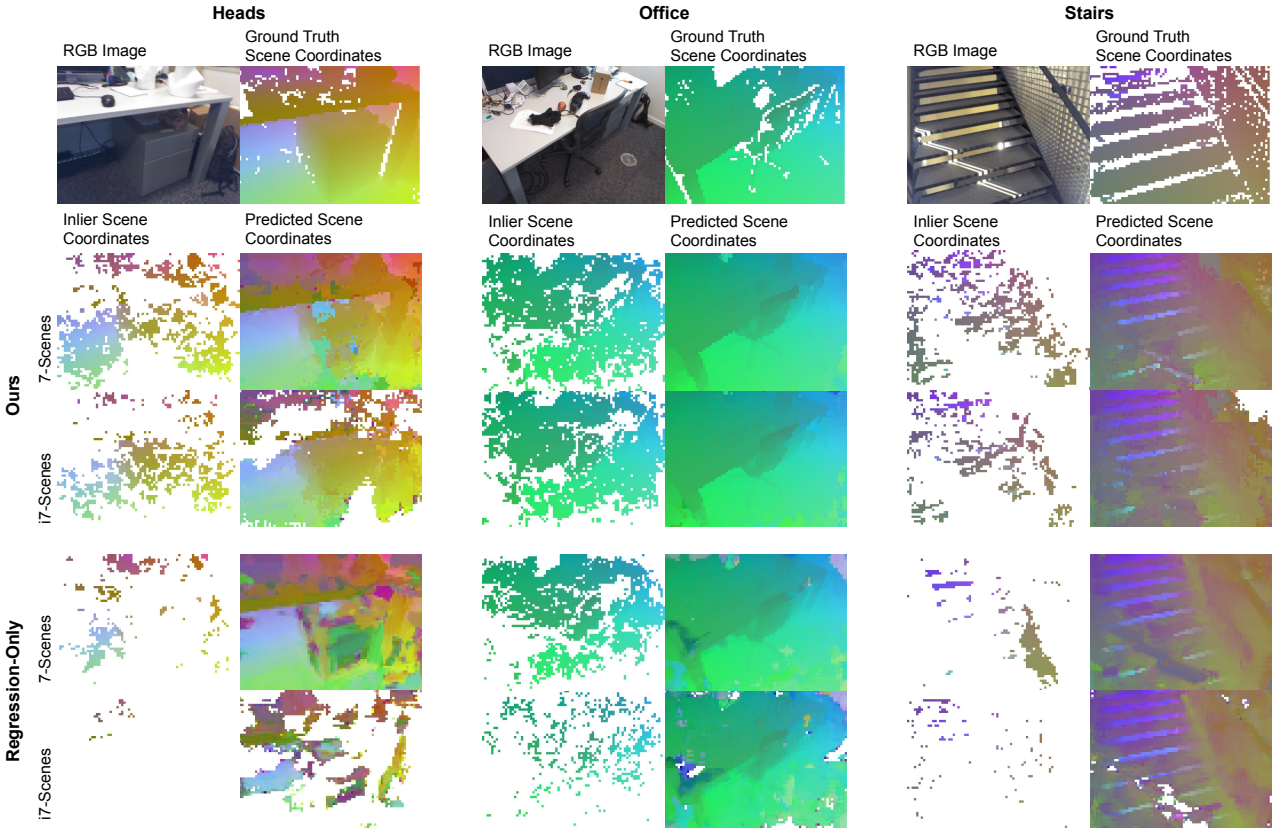

Figure 4. We visualize the scene coordinate predictions for three test images from 7-Scenes/i7-Scenes. The XYZ coordinates are mapped to RGB values. The ground truth scene coordinates are computed from the depth maps, and invalid depth values (0, 65535) are ignored. Should a scene coordinate prediction be out of the scope of the corresponding individual scene, the prediction is treated as invalid and not visualized. We also visualize the inlier scene coordinates retained after the pose optimization (PnP-RANSAC) stage. On both 7-Scenes and i7-Scenes, our method produces consistently better scene coordinate predictions with more inliers compared to the regression-only baseline.

Figure 5. We show the scene coordinate predictions for the Aachen dataset experiments. The scene coordinate predictions are visualized as 2D-2D matches between the query (left) and database (right) images. For each pair, the retrieved database image with the largest number of inliers is selected, and only the inlier matches are visualized. We show that our method is able to produce accurate correspondences for challenging queries (left column). Failure cases are also given (right column).