Supplementary Material for The Mapillary Traffic Sign Dataset for Detection and Classification on a Global Scale

Christian Ertler[0000−0001−6385−7907], Jerneja Mislej[0000−0002−6372−559X], Tobias Ollmann[0000−0002−8224−3692], Lorenzo Porzi[0000−0001−9331−2908], Gerhard Neuhold[0000−0002−1689−7284], and Yubin Kuang[0000−0002−4079−0962]

Facebook

1 Scene Classification for Image Selection

An important requirement during image selection for MTSD was to ensure high diversity of images with different image properties. Since the frequency of occurrence of certain traffic sign classes can be very different depending on the scene, we trained a neural network to predict the scene classes of the images and used the predicted labels to guide the image selection in order to diversify the scene classes in the final dataset. To train the scene classification network, we have used a subset of the scene classes of the BDD100K dataset [6]. After filtering BDD100K for images that have either the *residential*, *highway*, or *city street* class label, we trained a ResNet50 [5] that was pre-trained on ImageNet with a cross-entropy loss using stochastic gradient descent (SGD). The network was trained to convergence with an initial learning rate of 1×10^{-2} which was reduced by a factor of 0.1 until validation accuracy plateaued.

Figure 1 shows the distribution of scene classes within the supervised set of MTSD according to predictions of this model as targeted during our image selection scheme (see Section 2.1 of the main paper). We opted for a uniform distribution after treating *city street* and *residential* as a single class, since we found that these two classes (as annotated in BDD100K) are not always clearly distinguishable even for human. Given the large number of candidate images, this weakly-supervised image selection scheme facilitated increasing the diversity in scene classes.

2 Template Proposal Network

As described in Section 2.2 of the main paper, we used a neural network to predict similarities between sampled crops and grouped template images in order to assist the annotators in choosing a valid template during the annotation process. The predicted similarities were used to propose template images for each sign in a similarity-ordered way. Without such a mechanism, it would be extremely

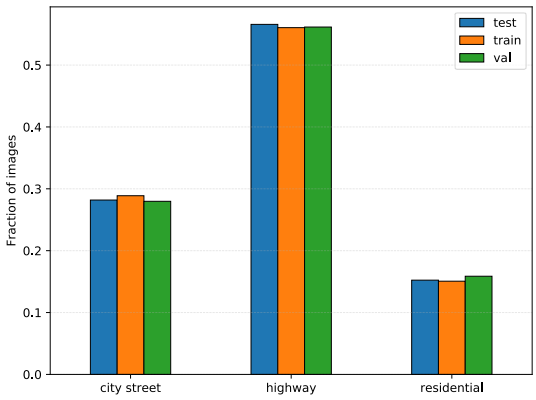

Fig. 1. Distribution of scene categories within MTSD as predicted by our scene prediction network.

time-consuming for the annotators to handle the large set of different template images that are available.

Note that the goal of the template proposal model is not necessarily to predict the correct class in terms of the most similar template but to have the matching template together with similar ones at least within the top-k predictions. In this way, the annotator can browse the template groups either ordered by similarity to the traffic sign crop under question, or ordered by the similarity between a selected template image and other templates. The latter ordering allows to browse through the template images in a semantically meaningful way if a matching template is not proposed in the first place. Further, we want to point out that this approach allowed us to add new missing templates to the UI on demand without the need of training data or re-training of the proposer network. Figure 2 shows a screenshot of the user interface using the described network to propose templates.

Besides the proposer based navigation, we additionally provided a text-based template search. This was necessary for cases where the proposer failed to provide good templates.

3 Classification Network

In this section, we provide details about the classification network we used for the baseline experiments.

As described in Section 5 of the main paper, the network consists of seven 3×3 convolution layers followed by global average pooling and a fully connected stage. The first two convolution layers operate on the input resolution of 40×40 pixels with 32 features each, followed by max-pooling to halve the size of the feature map. This is followed by four convolution layers with 64 features and again a max-pooling layer to halve the size in both x and y directions. The

Fig. 2. The classification UI used by the annotators. The traffic sign to be annotated is shown with its bounding box on the left. On the right, one can see the current selection (green bounding box in the 1st column) as well as the proposed templates. Each column starting with the second one shows a proposed template group based on the similarity of the real image crop and the templates as predicted by our proposal network; The other templates in the 1st column show proposals based on the currently selected template; note how this enables the annotator to find even more similar templates that are not proposed based on the image crop.

last convolution layer with 128 features is followed by global average pooling to flatten the feature map for the fully connected stage consisting of a hidden layer with 256 features and two prediction layers, one with a single sigmoid activation for foreground/background prediction and the second with a 401-way softmax activation head. All hidden layers have ReLU activations and batch normalization [3]. An illustration of the network and corresponding feature map sizes can be found in Figure 3.

4 Partial Annotation

In this section, we elaborate on how we automatically generated the partially annotated images using a structure from motion (SfM) pipeline.

For each fully-annotated image within the training set of MTSD, we query for a set of neighboring images from Mapillary that locate within a pre-defined distance in real-world coordinates to form an image cluster. Then, we recover the relative camera poses between images in the cluster using a pipeline based on OpenSfM [1]. To create tentative correspondences between annotated signs and automatic detections (by Mapillary) in the neighboring images, we rely on the class labels, *i.e.* a pair of signs with the same labels form a tentative correspondence. With such tentative correspondences, we further triangulate the 3D positions of the signs [2] and vote for the most geometrically feasible correspondences based on the estimated relative camera poses. Here, we triangulate the traffic signs as 3D points with the centers of corresponding 2D bounding boxes.

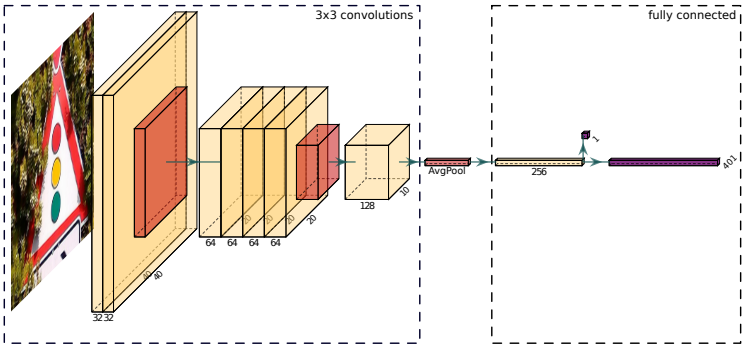

Fig. 3. Network architecture of the baseline classifier. The blocks illustrate the output feature maps of the corresponding layers. Yellow boxes are convolution or fully connected layers; Orange boxes are pooling layers.

Table 1. Statistics of annotator interactions

	Count	Mean
Images worked on	52,608	-
Signs worked on	266,238	-
Originated from detection	128,601	0.52
IoU with original detection		- 0.76
New signs per image		- 2.63

To this end, we have established geometrically and semantically consistent correspondences between the annotated signs and automatic detections. The correspondences are then utilized to generate the partial annotated dataset (as described in Section 2.4 of the main paper) by propagating the human verified class labels of the corresponding traffic sign instances in the fully annotated training set to the automatically generated ones.

5 Annotator Interactions

To gather some insights about the work of our annotators, we analyzed their interactions with the bounding boxes fetched from the *Mapillary* API as described in Section 2 of the main paper. Results are presented in Table 1.

We found that 52 % of the bounding boxes annotated in MTSD originated from an automatic detection. Each detection bounding box was manually refined by our annotators to tightly fit the corresponding traffic sign. All missing bounding boxes were manually added by our annotators. The class labels have been assigned by our annotators in a purely manual way, *i.e.* we did not initialize with automatically predicted class labels. When comparing the final boxes within MTSD to the original detection, we find an overlap of only 76 % in terms of IoU,

showing the amount of manual refinement to the bounding boxes. Additionally, we found that the annotators on average added approximately 3 completely new bounding boxes that were missing before in each image.

In terms of timings, we found that the average time spent on carefully localizing all traffic signs present in an image was 2.4 min. This includes manually scanning the entire image and drawing bounding boxes around each traffic sign. For the classification of a single traffic sign crop we measured an average of 19.6 s including the selection of a traffic sign template (if applicable) and assigning attributes.

6 Qualitative Examples

In the following we show additional examples of annotated MTSD images in Section 6.1. Further, we show results of our transfer learning experiments on TT100K [7] and MVD [4] in Section 6.2. For qualitative comparisons of detections, we make sure that we choose score thresholds so that either recall or precision are comparable.

6.1 Examples in MTSD

We show some examples of annotated images from the MTSD training set in Figure 4. MTSD covers a broad range of capture settings including cities, highways, residential areas, and rural areas with different lighting and weather conditions from varying view points. This variety makes MTSD the most diverse traffic sign dataset available.

6.2 Impact of Transfer Learning

To illustrate the gains of our baseline on TT100K by pre-training the model on MTSD, we show qualitative comparisons of detections in Figure 5. The model pre-trained on MTSD is able to detect more traffic signs in many cases while preserving a high precision. For fair qualitative comparison, both models operate on the same level of precision (0.95), however, the model pre-trained on MTSD achieves a higher recall (0.91 *vs.* 0.81).

A similar qualitative comparison for MVD is shown in Figure 6. Again, both models operate on the same precision level of 0.8, while the model pre-trained on MTSD obtains a higher recall of 0.67 compared to 0.61 for the model trained solely on MVD. Besides the higher recall, the pre-trained model has less confusion with billboards and other planar objects that are similar to traffic signs.

Fig. 4. Examples of annotated images from the MTSD training set, covering diverse lighting and weather conditions

Fig. 5. Qualitative comparisons between our baseline trained on TT100K only (left), and our baseline pre-trained on MTSD and fine-tuned on TT100K (right). The score thresholds are chosen such that both models operate on the same level of precision.

Fig. 6. Qualitative comparisons between our binary baseline detector trained on MVD only (left), and our baseline pre-trained on MTSD and fine-tuned on MVD (right). The score thresholds are chosen such that both models operate on the same level of precision.

References

1. Opensfm. https://github.com/mapillary/OpenSfM, accessed: 2019-11-11 3
2. Hartley, R., Zisserman, A.: Multiple View Geometry in Computer Vision. Cambridge university press (2003) 3
3. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: Proceedings of the International Conference on Machine Learning (ICML). pp. 448–456 (2015) 3
4. Neuhold, G., Ollmann, T., Rota Bulo, S., Kontschieder, P.: The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2017) 5
5. Ren, S., He, K., Girshick, R., Sun, J.: Faster r-cnn: Towards real-time object detection with region proposal networks. In: Proceedings of the Conference on Neural Information Processing Systems (NIPS). pp. 91–99 (2015) 1
6. Yu, F., Xian, W., Chen, Y., Liu, F., Liao, M., Madhavan, V., Darrell, T.: Bdd100k: A diverse driving video database with scalable annotation tooling. arXiv preprint arXiv:1805.04687 (2018) 1
7. Zhu, Z., Liang, D., Zhang, S., Huang, X., Li, B., Hu, S.: Traffic-sign detection and classification in the wild. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) (2016) 5