Block Annotation: Better Image Annotation with Sub-Image Decomposition Supplementary Material

Hubert Lin
Cornell University

Paul Upchurch
Cornell University

Kavita Bala
Cornell University

1. Introduction

This supplementary is organized as follows. First, we detail the parameters used in our experiments to reproduce our results (sections 2, 3). Second, we show some supplementary results which could not be included in the main paper (section 4). Third, we include visualizations of crowdsourced annotations and inpainted labels (section 5).

2. Deeplabv3+ and Mobilenetv2

In our experiments, we use the open-source implementation of Deeplabv3+ found at `https://github.com/tensorflow/models/tree/master/research/deeplab` . For our comparison against weakly supervised methods, we use the open-source implementation of Mobilenetv2 found at `https://github.com/tensorflow/models/tree/master/research/slim/nets/mobilenet`.

2.1. Architecture

For Deeplabv3+, we use Xception65 [2] as the backbone. ASPP atrous rates are set to 6,12,18 for output stride 16 and 12,24,36 for output stride 8 as in [1]. Training uses output stride 16 while evaluation uses output stride 8. Mobilenetv2 does not use ASPP or decoder.

2.2. Training Procedure

Training hyperparameters are based on [1]. All hyperparameters use settings in [1] unless otherwise noted here. Batch normalization parameters are not finetuned due to low batch size (GPU memory constraints). Batch normalization parameters are initialized and frozen with the pretrained checkpoint. All Deeplabv3+ experiments initialize the network with the official pretrained model on ImageNet + MSCOCO + Pascal VOC and are trained for 100K iterations. Experiments with Mobilenetv2 initialize the network with the official Mobilenetv2 (width 1.0, input resolution 224) ImageNet model. We use polynomial learning rate decay as in [1]. During training time, we ignore the loss for pixels whose predicted class has a softmax probability greater than 0.95. This can be considered a form of hard negative mining where samples (pixels) with high confidence are ignored. Our preliminary experiments suggest that this appears to improve rate of convergence.

In any experiments that use fewer than 100% of the images in the training dataset, the images are shuffled and the first N images are selected. The shuffling order is determined once and fixed across experiments.

Cityscapes. Deeplabv3+. Learning rate 0.0005. Batch size 2 with input crop size 769×769.

ADE20K. Deeplabv3+. Learning rate 0.001. Batch size 4 with input crop size 513×513.

Pascal VOC. Mobilenetv2. Learning rate 0.001. Batch size 16 with input crop size 513×513. Batch norm parameters are finetuned.

2.3. Evaluation Procedure

After training for a fixed 100K iterations (no early stopping), models are tested on the validation set. 100K iterations is chosen based on settings in [1] which uses 90K iterations for Cityscapes. For consistency, 100K iterations is also used for ADE20K and Pascal VOC. Evaluation is performed at single scale without flipping on full resolution images. For completeness, we report here the validation mIOU when the network is trained out-of-the-box with the full training data using the outlined procedure.

Cityscapes. Validation images are padded to 1025×2049. Validation mIOU is 77.7%.

ADE20K. Validation images are padded to 513×513. Validation mIOU is 37.39%.

Pascal VOC. Validation images are padded to 513×513. Validation mIOU is 69.6%.

3. Block-Inpainting Model

The block-inpainting model is based on Deeplabv3+ with some architectural and training modifications.

3.1. Architectural Modifications

The input to the block-inpainting model is a tensor of shape $h \times w \times (3 + K)$ where K is the number of classes in the dataset (see main paper). The weights of the first layer must be expanded to accommodate the K additional

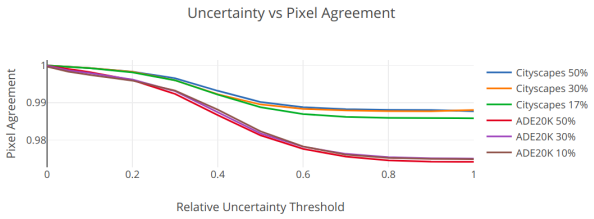

Figure 8: **Block-Inpainting Model** uncertainty versus human pixel-wise agreement for inpainted labels. Curves for different pixel budgets shown for comparison.

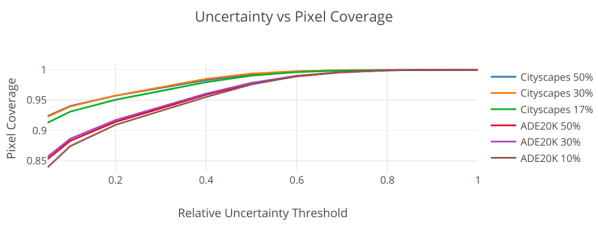

Figure 9: **Block-Inpainting Model** uncertainty versus pixel coverage for human checkerboard + automatic labels. x-axis truncated at 0.05 on left. Curves for different pixel budgets shown for comparison.

channels. These weights are initialized from a unit normal distribution. For each dataset, the weights are initialized once and then fixed for every experiment.

To compute uncertainty, dropout is added to the middle and exit flow blocks of the Xception65 backbone. The dropout keep probability is 0.8 as in [5]. No dropout is added to the decoder as preliminary experiments suggest that this will degrade performance.

3.2. Training Details

During training time, the block-inpainting model is trained with randomly drawn block hints from the set of block-annotated images (see main paper). The block-inpainting model is trained for 100K iterations following section 2. The entire set of block-annotated images are used as training targets.

3.3. Inference Details

At inference time, the block-inpainting model keeps dropout activated to estimate uncertainty [4]. The block-inpainting model outputs are averaged over 100 forward passes (100 is found to be sufficient in [3]) to form the final prediction. The uncertainty is computed by taking the sample variance of the softmax probabilities for the predicted class (see main paper).

3.4. Ablation Studies

Effect of Uncertainty Threshold vs Pixel Agreement. How does the uncertainty threshold affect pixel agreement for block-inpainting? In figure 8, we show the mean pixel agreement with human labels for varying thresholds. Lower uncertainty threshold for rejection results in higher pixel agreement. The pixel agreement with lower pixel budgets are shown for comparison. The pixel budget is the number of block-annotated pixels in the dataset with which the block-inpainting model is trained. All experiments use checkerboard block hints.

Effect of Uncertainty Threshold vs Pixel Coverage. How does the uncertainty threshold affect pixel coverage for block-inpainting? In figure 9, we show the mean pixel coverage for varying uncertainty thresholds as a fraction of maximum uncertainty. Lower uncertainty threshold for rejection results in lower pixel coverage. The pixel coverage with lower pixel budgets are shown for comparison. The pixel budget is the number of human block-annotated pixels in the dataset with which the block-inpainting model is trained. All experiments use checkerboard block hints.

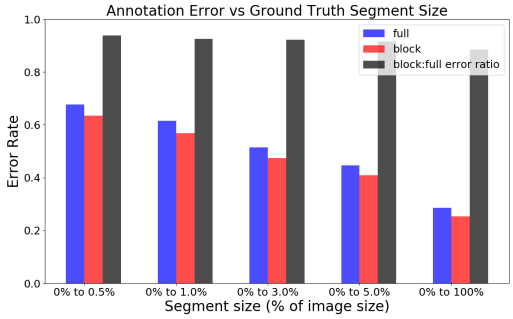

Figure 10: **Annotation error rate** for block and full annotation for differently sized ground truth segments. **Lower is better**.

4. Supplementary Results

We show some supplementary results which could not be included in the main paper.

4.1. Block Annotation Quality for Small Segments

How do workers perform when annotating small regions (which may be difficult to annotate)? We look at the annotation error rate of small segments in SUNCG. Figure 10 shows the annotation error in regions where the ground truth segments are within some range of sizes proportional to the full image size (480×640 px). Block annotations consistently have a lower error rate in these ranges, indicating that block annotation is advantageous regardless of scale of segments. For completeness, the rightmost bars show the error rate for segments of all sizes.

4.2. Semantic Segmentation Performance Random Blocks vs Checkerboard

How does the location of blocks sampled within an image affect semantic segmentation performance? The network trained on randomly sampled blocks achieves 77.1% mIOU while the network trained on checkerboard blocks achieves

77.7% mIOU. The increase in performance in checkerboard annotations over randomly sampled blocks is due to pixel diversity (pixels far apart in an image are less correlated than neighboring pixels). This is similar to the effect observed in the first experiment which shows that pixel diversity due to image diversity increases performance. These observations are aligned with our expectations (see block selection discussion in main paper). In this experiment, we used all images from Cityscapes and created: (a) random block annotations (sample 50% of blocks for each image) and (b) checkerboard block annotations. To ensure that results are not due to sampling bias, we create split (a) three times (i.e. sample 50% of blocks per image three times) and average the results over the three splits.

5. Visualizations

We show samples of crowdsourced annotations and block-inpainted labels.

5.1. Crowdsourced Annotations (SUNCG/CGIntrinsics)

Figure 11 shows a sample of five annotated images. This figure is best viewed in color on screen with high zoom. See main paper for annotation details.

For each image, segments from block annotation and segments from full annotation are shown. Synthetic labels are assigned using majority ground truth voting. Note that assigning synthetic labels in this way will cause detailed crowdsourced segmentations to be lost. See main paper for estimate of cost to assign labels. For regions without segments, "void" label is assigned (color is black). For comparison, the dataset ground truth is shown in the final row. With block annotation, notice that workers segment small regions (e.g. the stool in image 1, row 2; the chair back in image 2, row 2; and the faucets in image 4, row 2) and oversegment regions (e.g. the cushions on the couch in image 3, row 1). With full annotation, notice that workers miss large regions, perhaps due to fatigue (e.g. the right window in image 3, row 4).

5.2. Crowdsourced Annotations (Cityscapes)

Figure 12 shows a sample of three block-annotated images. This figure is best viewed in color on screen with high zoom. Regions that are not block-annotated are masked out in this visualization. At annotation time, the worker sees the entire image for context (see main paper). See main paper for annotation details.

For easier comparison against Cityscapes, crowdsourced segments are colored. Colors are random because class labels have not been assigned to crowdsourced segments (see main paper for estimate of cost to assign labels). Crowdsourced segments with synthetic class labels are also included. Synthetic labels are assigned by taking the majority class label for the expert-labelled pixels in the crowdsourced segment. Note that assigning synthetic labels in this way will cause detailed crowdsourced segmentations to be lost. These visualizations are included to provide a better sense of the crowdsourced segments across the entire image rather than within individual blocks. Cityscapes segments are colored by class with "void" labels masked out.

In the main text, we compare the number of segments to expert full-image annotation. To compare the number of segments, we compute the number of label connected-components in expert annotations. Blocks with more than 50% void expert labels are ignored for a fair comparison.

5.3. Block-Inpainted Labels

We show set of samples of automatically assigned labels by the block-inpainting model (trained and tested with Block-50%) in figures 13, 14. For comparison, we show the human expert labels and the agreement between the block-inpainting model labels and the human labels (agreement is in white). With uncertainty threshold 0.2 on Cityscapes and 0.4 on ADE20K, over 94% of the pixels in the images are labelled by the block-inpainting model.

References

[1] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 1

[2] François Chollet. Xception: Deep learning with depthwise separable convolutions. *arXiv preprint*, pages 1610–02357, 2017. 1

[3] Yarin Gal and Zoubin Ghahramani. Bayesian convolutional neural networks with bernoulli approximate variational inference. *arXiv preprint arXiv:1506.02158*, 2015. 2

[4] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 2

[5] Christian Leibig, Vaneeda Allken, Murat Seçkin Ayhan, Philipp Berens, and Siegfried Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Scientific reports*, 7(1):17816, 2017. 2

Figure 11: SUNCG/CGIntrinsics Annotation Samples. Top to bottom: (Row 1) Crowdsourced blocks (boundaries). (Row 2) Crowdsourced blocks (synthetic labels). (Row 3) Crowdsourced full (boundaries). (Row 4) Crowdsourced full (synthetic labels). (Row 5) Ground truth. NOTE: Synthetic labels are the majority ground truth label for pixels in each segment. This means finely segmented crowdsourced segments (such as cushions on couches) will be lost in visualization. White dotted boxes highlight examples where block annotation qualitatively outperforms full annotation.

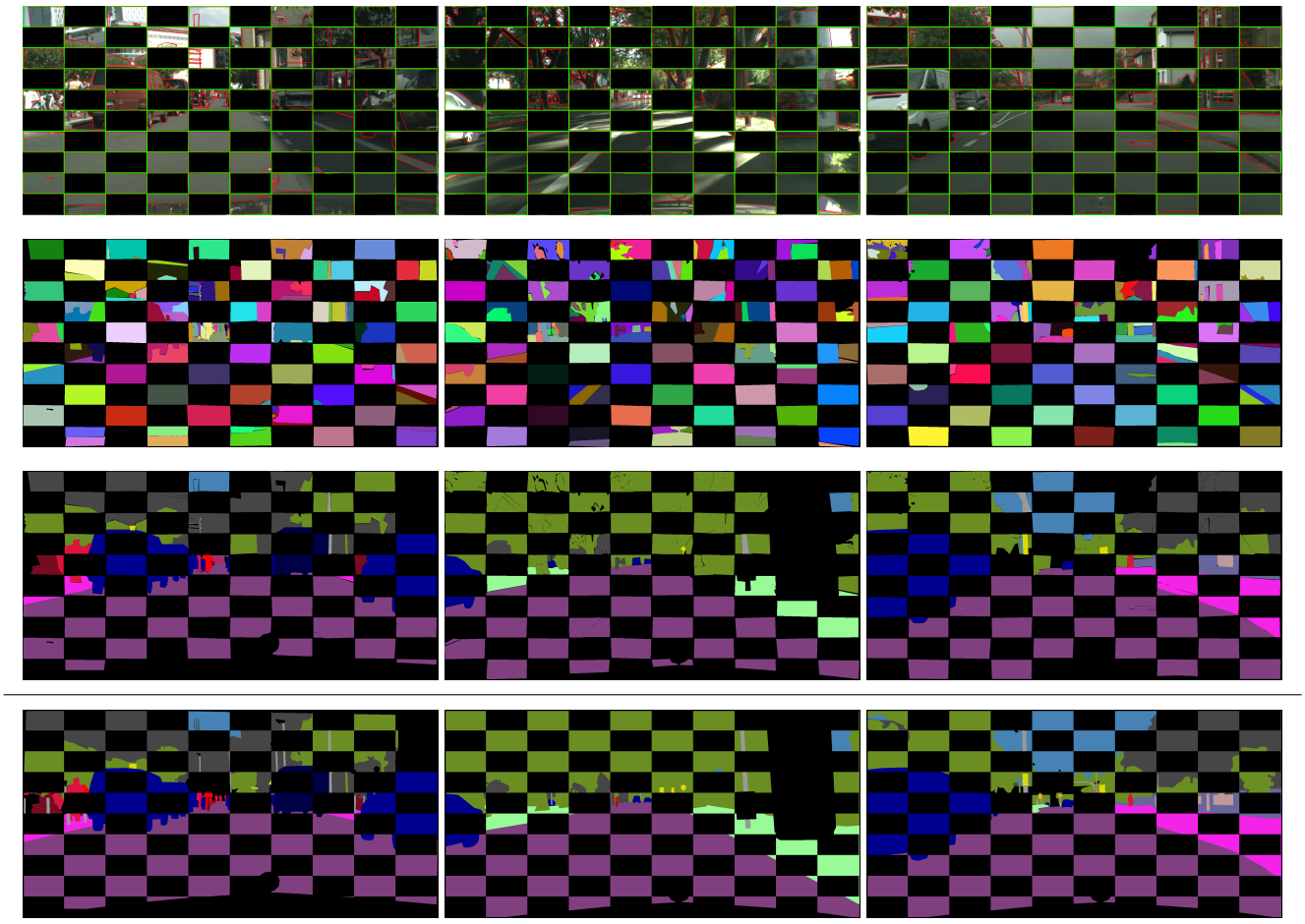

Figure 12: Cityscapes Annotation Samples. Top to bottom: (Row 1) Crowdsourced (boundaries). (Row 2) Crowdsourced (randomly colored). (Row 3) Crowdsourced (synthetic labels). (Row 4) Expert Cityscapes. NOTE: Synthetic labels are the majority expert label for pixels in each segment. This means finely segmented crowdsourced segments (such as sky between leaves) will be lost in visualization.

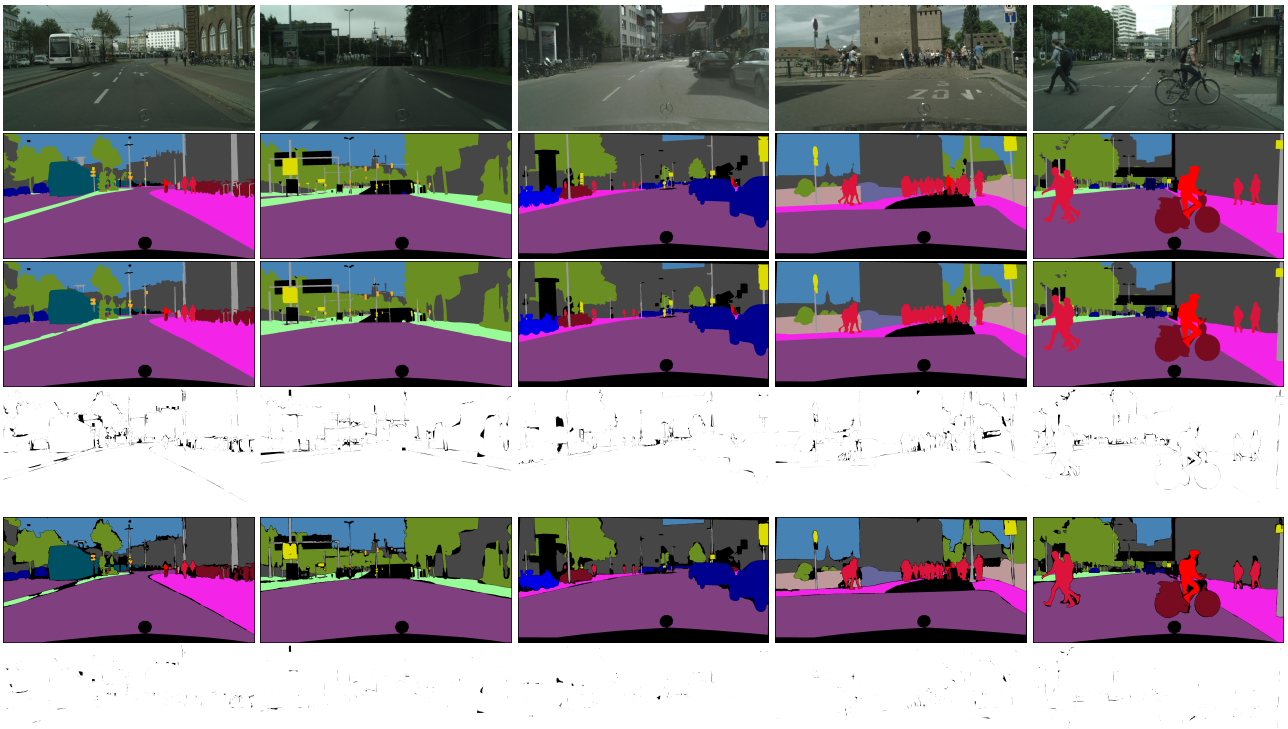

Figure 13: Block-Inpainting Cityscapes Samples. Top to bottom: (Row 1) Original image. (Row 2) Human labels. (Row 3) Inpainted labels (all). (Row 4) Agreement (row 3 vs row 2). (Row 5) Inpainted labels (<20% relative uncertainty). (Row 6) Agreement (row 5 vs row 2). Void labels and rejected inpainted labels are masked out.

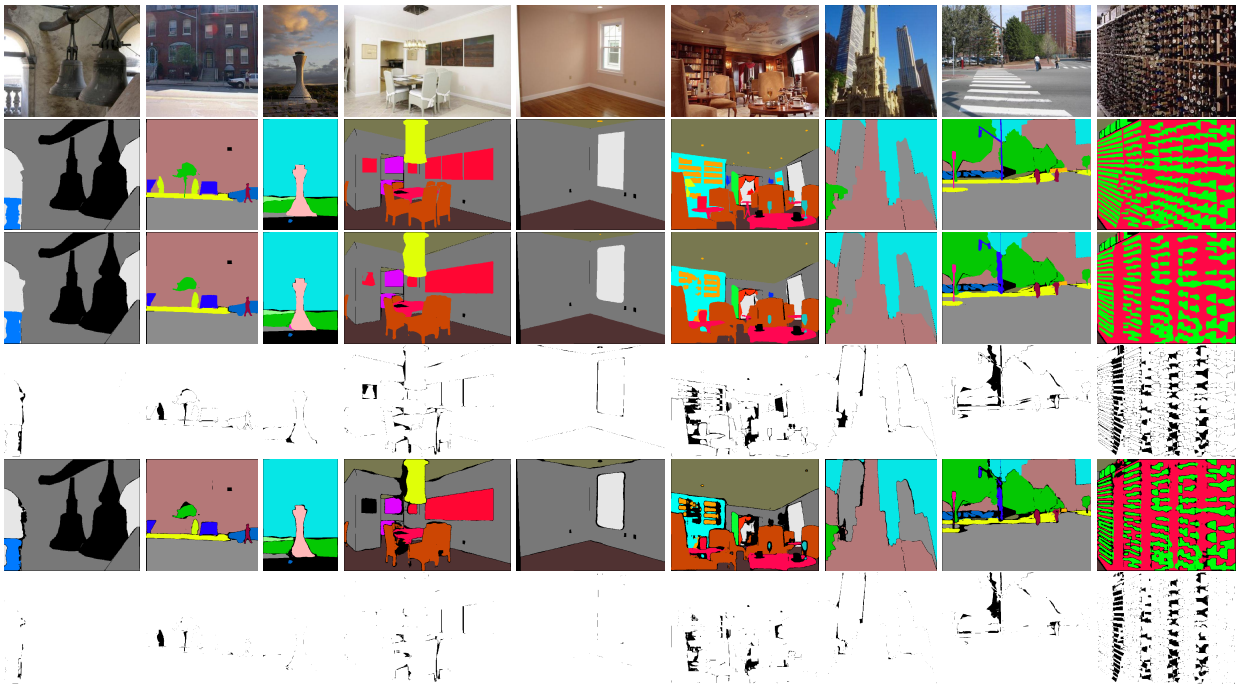

Figure 14: Block-Inpainting ADE20K Samples. Top to bottom: (Row 1) Original image. (Row 2) Human labels. (Row 3) Inpainted labels (all). (Row 4) Agreement (row 3 vs row 2). (Row 5) Inpainted labels (<40% relative uncertainty). (Row 6) Agreement (row 5 vs row 2). Void labels and rejected inpainted labels are masked out.