Attribution

The satellite images shown below are from Mapbox in the vicinity of Rosdorf/Göttingen, Germany, obtained as described in R.A., "Rendered maps with Python" on Medium. The grayscale label images are constructed from OpenStreetMap data, downloaded (2019-10-16) from the Overpass API filtered by `building=*` via the JOSM desktop app.

Problem statement & datasets

The task is to label house roofs in satellite images of rural areas roughly as follows:

given wanted

A possible application is identification of inhabited areas of interest for the solar panels market.

The "Dida" dataset contains

- 30 satellite (or similar) images of size $256 \times 256 \times$ RGBA, and
- 25 grayscale label images of unknown origin. One duplicate mislabel is omitted.

The the alpha channel is presumably used to indicate missing or sensitive data.

More specifically now, the aim is to construct a neural network image segmentation predictor and use it on the 5 unlabeled satellite images of the "Dida" dataset.

The "Mapbox/OSM" dataset, obtained as described in the beginning, contains 359 RGB images of size 128×128 with the corresponding grayscale label images. While Mapbox satellite images are well-aligned, they are outdated compared to OpenStreetMap building annotations. The images/label pairs have been hand-selected from a somewhat larger pool to make sure they are still compatible, i.e. refer to the same (on-the-)ground truth.

The datasets are downscaled to 128×128 and augmented 8-fold by random rotation and mirroring for training (resulting in a handful of duplicates even across training and validation sets).

Neural net architecture

Our neural network is essentially the MobileNetV2-derived encoder-decoder "U-Net" from the TensorFlow2 image segmentation tutorial

https://www.tensorflow.org/tutorials/images/segmentation.

The predictor is constructed by tapping into several progressively narrow layers of the pretrained and frozen image classifier MobileNetV2 (encoder) and by combining their activations/outputs using trainable convolutional upsampling layers (decoder). The MobileNetV2 encoder takes a $128 \times 128 \times$ RGB image; therefore, the alpha channel is split off before the "U-Net" and concatenated again with its output. The result is then passed through another convolution layer with two filters with a $128 \times 128 \times 2$ output. The channel dimension is interpreted as the predicted pixel label/class confidence (non-building vs. building). This is a departure from the tutorial that also includes the "object boundary" class. A graphical summary is shown in Fig. 3.

Results

We train the same model with the Adam optimizer on the pixelwise cross-entropy loss for 33 epochs, setting apart 20% of data for validation, either

- only on the "Dida" dataset (Scenario I), or
- on both "Dida" and "Mapbox/JOSM" datasets (Scenario II).

The behavior of the loss on the training and validation datasets during training is reasonable, see Fig. 1. Beyond 33 training epochs, the validation loss increases (not shown).

The predicted labels along with hand-drawn expected labels are shown in Fig. 2.

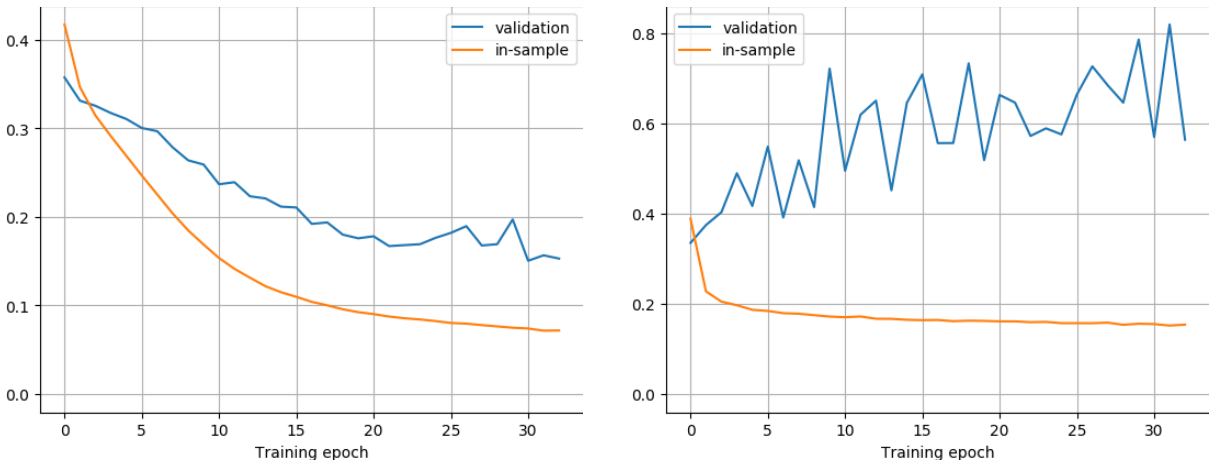

Figure 1: Cross-entropy loss, scenario I (left) and II (right).

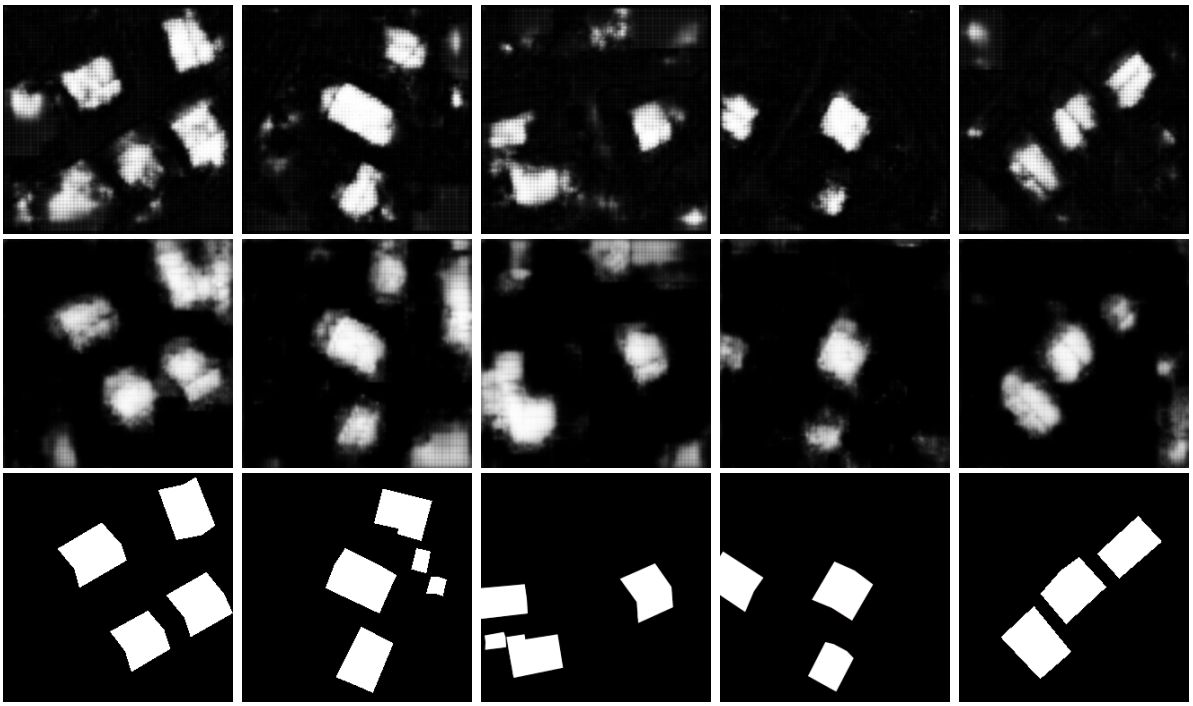

Figure 2: Missing Dida labels top-to-bottom: scenario I / scenario II / hand-made.

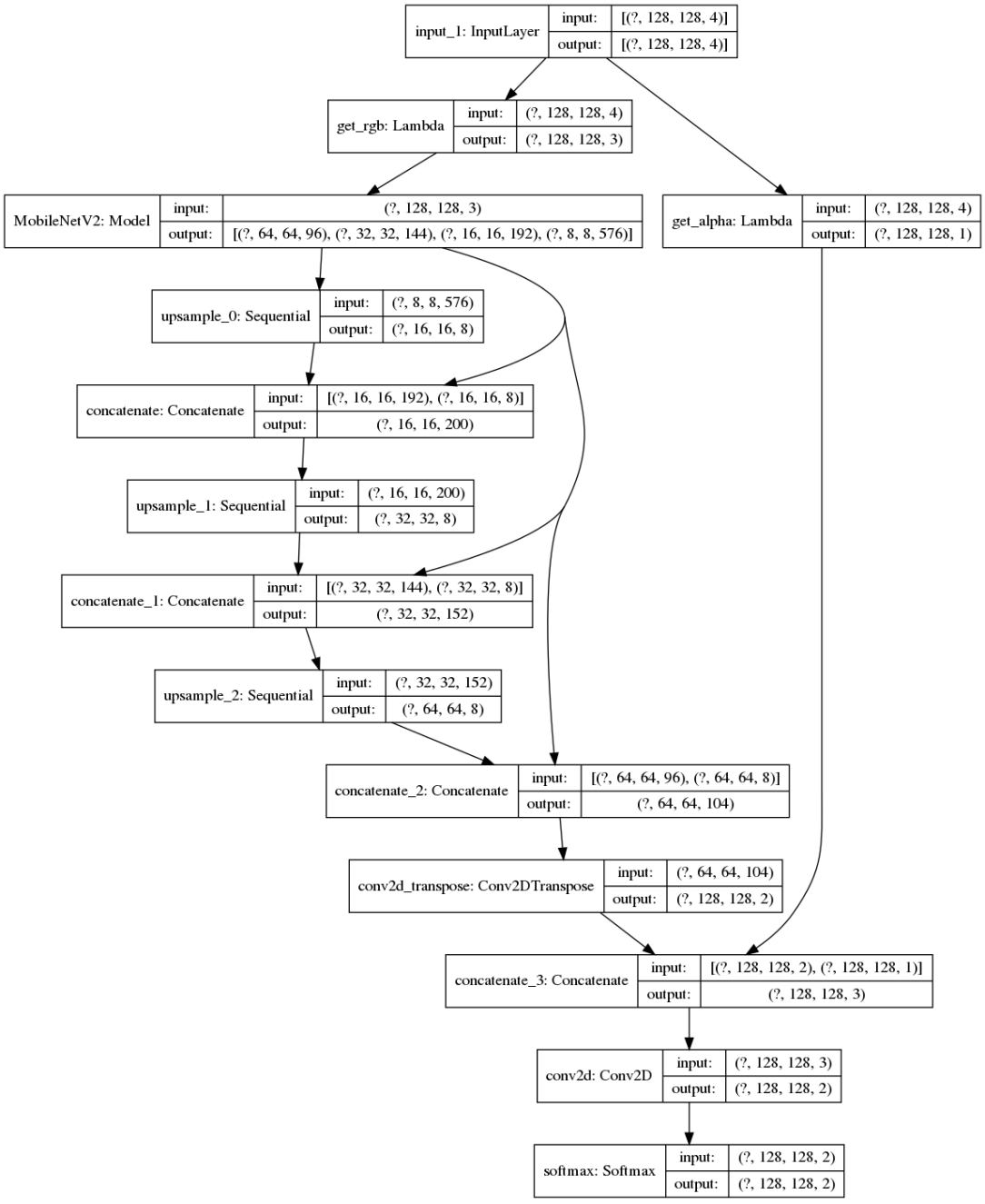

Figure 3: Neural net architecture on top of MobileNetV2.