Mapping Tidal Salt Marshes

Nolan Walsh njwalsh
Nikhil Bhattasali nikhilxb
Freya Chay freyac

Abstract

This paper details the application of fully convolutional neural networks to the problem of land cover classification. Specifically, high resolution satellite imagery was used to train several models that performed image segmentation in order to map the location of tidal salt marshes. Included are detailed results of several experiments using a simple convolutional architecture, along with the results from a modified version of the more complex U-net architecture. While the more complex U-net far outperformed more basic models on the validation set, both model types performed similarly on an unseen test set.

1. Introduction

Tidal salt marshes are one of the most effective natural habitats for sequestering carbon [7, 21], yet the best estimate for global salt marsh area is 2.2-40 million hectares[3]. A more accurate estimate could facilitate political advocacy to preserve and restore salt marshes, support movements to incorporate them into carbon offset markets, and enable tracking of salt marsh area change over time.

By using pixel-wise image segmentation, our models can generate accurate maps of the locations of salt marshes along with generating an estimate of the percentage of land covered by salt marsh within an area.

For the sake of tractability, rather than pursuing a global mapping of salt marshes, we trained models that could identify California salt marsh. The hope is that this project can contribute to a global estimate somewhere down the line, or inspire others to take the next step and apply these models to a broader data set.

The input to our models is a multispectral Planet Labs satellite image. We use a convolutional neural net to produce a pixel-wise classification map: [1 - Salt Marsh, 0 - Other]. Thus, the output from our model is a binary image of the same two dimensional size as the input.

2. Related Work

In the past, most land cover classification has been done using random forest models[16], decision trees [17], hierarchical graphs [2] [14] and support vector machines [9]. People have also used one of several proprietary software packages that offer image classification and segmentation tools [13][14].

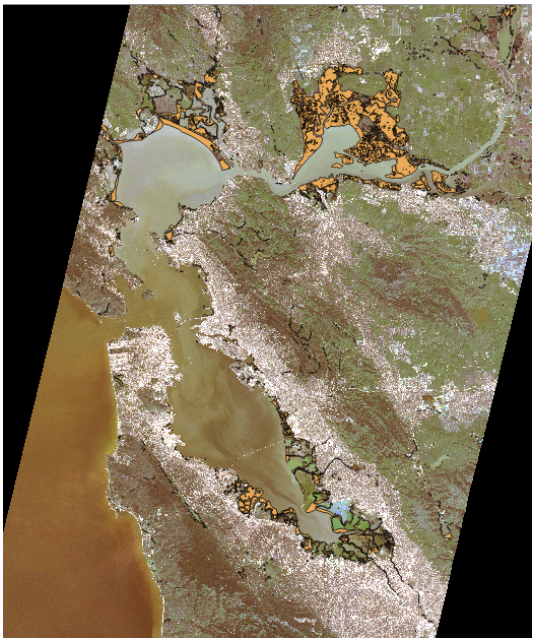

Figure 1. Map of San Francisco Bay Salt Marshes. Marsh colored in orange.

Recently, convolutional neural networks (CNNs) have produced excellent land cover classification results [5], demonstrating the efficacy of using convolution as a tool to interpret satellite imagery. However, we were interested in image segmentation rather than just image classification.

Recently, there has been a lot of recent activity in the realm of semantic segmentation using neural nets[11][22]. One interesting approach relies on Conditional Adversarial Nets to translate an input image into an output image of any type, including a semantic segmentation [10]. Another approach relies on fully convolutional neural networks (FCNNs) [20, 6, 12], in which a series of convolutional downsampling layers is followed by a series of upsampling layers to achieve an output with the same two-dimensional shape as the input. In the model proposed by Long et al. in 2014,

the researchers performed upsampling to the original image size at several layers among the standard convolutions. This let their model learn to combine coarse, high layer information with fine, low layer information, and identify different features for classification of the same target at distinct depths [20]. Since satellite imagery is captured from only one depth, the multiple upsamplings is less important, but the efficacy of an FCNN is applicable.

One successful model architecture for image segmentation using an FCNN is the U-net [19]. This architecture consists of a contracting path of convolutional downsampling layers, and an expansive path of upsampling layers. Along the contracting path, outputs from each layer are passed to the corresponding layer on the expansive path. Last month, the U-net architecture was tweaked and applied to multi-class satellite image segmentation [15].

3. Methods

3.1. Convolutional Neural Networks

Convolutional neural networks (CNNs) are a class of feed-forward artificial neural networks that are especially designed to operate on visual inputs, i.e. images and video. CNNs make explicit 3 properties that are useful for the visual domain.

1. **3D volumes of neurons.** Neuronal layers are modeled as tensors of shape `(W, H, D)`, where `W` is layer width, `H` is layer width, and `D` is layer depth. For the input layer, `WxH` are the input image dimensions, and D is the number of channels (e.g. `D = 3` for standard RGB images).

2. **Local connectivity.** Tunable weights come in the form of *filters*, which are also modeled as tensors, of shape `(WW, HH, D)`, where `WW` is filter width and `HH` is filter width. A sliding dot product is taken between a filter and a region of layer activations, with the sliding (i.e. convolution) ocurring across the spatial dimensions. A number of different filters can be applied on a given layer, forming the `D` dimension for the next layer. Local connectivity enforced by this formulation promotes the extraction of spatial structure.

3. **Parameter sharing.** The same filter is applied across all regions of a layer, thereby dramatically reducing the number of parameters used (compared to a different weight for each neuron in the activation layer). Parameter sharing enforced by this formulation promotes spatial invariance, i.e. making the model more robust to slight positional changes in visual features.

There are several types of layers in CNNs, including `CONV` (convolution; the linear operation described above), `POOL` (pooling; reduces spatial dimensions), `RELU` (activation function; introduces nonlinearity), `CONVT` (convolution transpose; transpose of linear operation described above.), and `BATCHNORM` (batch normalization; normalizes layer activations). Micro-architectural decisions involve setting the hyperparameters of these layers: filter dimensions, filter count, stride, dialation, pool size, activation function, etc.

Convolution can be leveraged in architectures suited for different tasks. Configurations of layers are abstracted into modules, each with its own micro-architecture. Macro-architectural decisions involve connecting modules together: network depth and width, stacking, skip connections, auxiliary loss signal injection.

Currently, CNNs dominate on virtually all computer vision tasks compared to traditional approaches.

3.2. Image Segmentation

In an image classification problem, the task is to use a CNN to transform an image tensor of shape `(W, H, D)` into an score vector of shape `(C)` of normalized softmax scores over `C` object classes.

In an image segmentation problem, the task is to use a CNN to transform an image tensor of shape `(W, H, D)` into a label tensor of shape `(W, H)` that attaches labels to each pixel of the input. If there are more than 2 labels (multi-class), an intermediate label tensor of shape `(W, H, C)` is created, with normalized softmax scores over the `C` dimension used to produced a per-pixel argmax label. If there are only 2 labels (binary), the intermediate label tensor is of shape `(W, H)`, with a sigmoid activation function applied to translate the values to between 0 and 1, and a threshold applied to discretize.

For this project, the segmentation problem is a binary one: a value of 1 means a satellite image pixel corresponds to a saltmarsh, a value of 0 means it does not. Therefore, the loss function most appropriate for the task is binary cross entropy.

Let $\hat{\mathbf{y}}$ be the predicted label tensor of shape `(W, H)`, and let $\mathbf{y}$ be the ground-truth label tensor of the same shape. The binary cross entropy loss is:

$$\mathcal{L} = -\sum_{w}\sum_{h} (y_{wh}) \log (\hat{y}_{wh}) + (1 - y_{wh}) \log (1 - \hat{y}_{wh})$$

3.3. Optimization

Optimizing neural networks is a highly non-convex problem; as is standard practice, we optimize using a variant of Stochastic Gradient Descent (SGD).

We used the Adaptive Moment Estimation (Adam) varient, which is a method that computes adaptive learning rates for each parameter. In addition to storing an exponentially

decaying average of past squared gradients like Adadelta and RMSprop, Adam also keeps an exponentially decaying average of past gradients similar to momentum.

In addition to loss and accuracy, we also evaluated our models using *precision*, *recall*, and the *F1 score*. Precision is the proportion of predicted positives that are actual:

$$\text{precision} = \frac{\text{true positive}}{\text{true positive + false positive}}$$

Recall is the proportion of actual positive that are predicted:

$$\text{recall} = \frac{\text{true positive}}{\text{true positive + false negative}}$$

The F1 score is a weighted average of the precision and recall, and is attractive because it gives a single number that can be compared across models:

$$\text{F1} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

3.4. Experimental Environment

We built our models using Keras [8], an open-source modular wrapper on top of Tensorflow [1]. Using Keras allowed us to write high-level, modularized Python code to define our model and run experiments. At the same time, it afforded us the ability to take advantage of efficient, GPU-optimized backend implementations of our model during training and validation.

We use instances on Google Cloud, with 1 attached GPU to perform our experiments. Each experiment is encapsulated in a separate Jupyter Notebook, which allowed us rich scripting, visualization, and documentation functionality.

4. Dataset and Features

Raw data for this problem comes from the Planet Labs Open California dataset[18]. Planet labs has many satellites capturing images of California on a daily basis. Since each satellite type captures distinct imagery, we exclusively used RapidEye data: 5x5m per pixel resolution with 5 information bands (Blue, Green, Red, Red Edge, and Near Infrared). For reference, Red Edge and Near Infrared bands are particularly useful for capturing information about vegetation and its characteristics[23].

In the Open California dataset, no composite images are available, so mitigating noise due to cloud cover was a matter of image selection. Images chosen had no visible cloud cover and less than 10% cloud cover according to the Planet Lab filter. We expected all 5 information bands to change based on the season, so images chosen were captured by the satellite in the middle of March of 2017.

The data was delivered from Planet Labs as sets of GeoTiffs with a 25*25 km area and 1 km of overlap with neighboring images in every direction. GeoTiffs were stitched into mosaics, large continuous images, using gdal. Labels for this mosaic came from the California Department of Fish and Wildlife Marine Resources GIS map of California Coastal Wetlands[4].

Images and labels for 4 locations on the California Coast (San Francisco Bay, Port Hueneme, Morro Bay, and Chula Vista) were read into numpy arrays, fractured into smaller tiles, and split into train, validation, and test sets. Tiles containing any null data (appearing black in Figure 1 were discarded. Since we were interested in a model's ability to generalize to regions of unmapped salt marsh, SF, Hueneme, and Chula Vista were split 80:20 into the train and validation sets. Morro Bay in its entirety was held out for testing.

Dataset	Num Pixels	% Salt Marsh
train	374,210,560	0.036
validation	93,569,024	0.035
test	1,572,864	0.063

Table 1. Train, val, and test data metrics.

5. Experimental Analysis

Unless otherwise specified, all experiments described in this section were performed with the following model: a tile of size 128x128 is fed into two successive convolutional layers with ReLu activation. In between these two layers is a batch normalization layer and a max pooling layer. Next, there are two up sampling (up_conv) layers so that the model's output is of the same two-dimensional size as the input. Finally we perform a single convolutional output layer (a convolution with a single filter) in order to obtain our final output. This "basic model" is drawn out in figure 2.

All precision and recall numbers in the following section were computed using a cutoff of 0.5 (any predicted score equal to or above 0.5 was considered a prediction of 1 and any predicted score below 0.5 was considered a prediction of 0).

5.1. Training Set Sub-sampling

Train Set	% Salt Marsh (Train)	Val F1
all tiles	0.04	0.320
50% salt marsh tiles	0.13	0.327
100% salt marsh tiles	0.16	0.372

Table 2. Training set sub-sampling experiments.

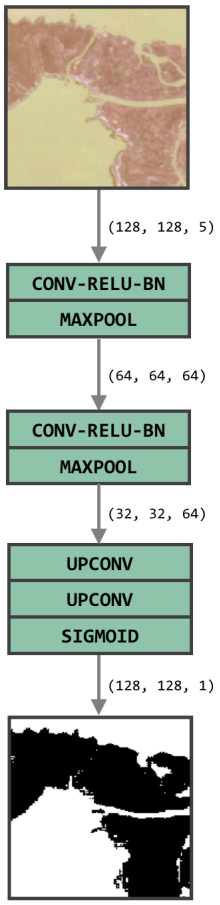

Figure 2. The "basic model" used in many of our experiments. See appendix for more details.

The most immediate problem with this data set was the extreme class imbalance. As can be seen Table 1, only 3.6% of our pixels are labeled as Salt Marsh pixels. In order to help our model learn to actually output predictions of Salt Marsh, we decided to experiment with training on a sub-sample of the available data. Table 2 details the results of this experiment. Each model was trained using a different sub-sample of the total available training data. The model trained using "50% salt marsh tiles" was trained using all available tiles containing at least one pixel of salt marsh, and another equally sized set of tiles containing no salt marsh. The model trained using "100% salt marsh tiles" was trained using only the tiles that contained at least one pixel of salt marsh. All models were evaluated on a single validation set, containing all available validation data (i.e. containing 3.5% salt marsh pixels).

This experiment shows that training on a sub-sample of the available tiles increases the performance of the model, even when the model is evaluated using the initial distribution of salt marsh tiles. This is likely because using a smaller subset of the training data helps eliminate the class imbalance, while still preserving enough negative examples. Even among just the tiles containing salt marsh, only 16% of the total pixels were labeled as salt marsh.

5.2. Patch Size

Experimenting with patch size served two purposes: - in combination with the tile sub-sampling method described above, it addressed class imbalance by capturing the salt marsh pixels with more or less of the surrounding area. - it allowed us to adjust our dataset to the breadth of information the model needed to make it's predictions.

We experimented with four tile sizes, using the basic model described above to evaluate each. Training data was comprised of 100% salt marsh tiles (as described above). Results from this experiment are shown in Table 3. Experimental results marked by a * were not run with a train/val split consistent with the rest of the experiments in this section.

Patch Size	% Salt Marsh (Train)	Val F1
64x64	0.40	0.226
128x128	0.26	0.372
256x256	0.17	0.404*
512x512	0.02	0.286*

Table 3. Patch size experiments.

Although Table 3, indicates that we should tile size of 256x256, the inconsistency between the 128x128 and the 256x256 data set splits and the relatively small improvement in F1 score led us to prioritize training time over F1 score. With 128x128 tiles, a single epoch training the Simple Conv-net took 37s, while an epoch took 63s for the 256x256 tiles.

5.3. Model Depth

Num Layers	Val Precision	Val Recall	Val F1
3	0.275	0.343	0.305
5	0.368	0.375	0.372
7	0.302	0.409	0.347

Table 4. Model depth experiments.

We also experimented with the complexity of our model. Table 4 contains the results of training models with different numbers of convolutional layers. We found that our initial architecture containing five convolutional layers performed the best.

5.4. Data Set Diversity

All experiments described above were performed just using data from the Bay Area. In order to test how well these

Train Dataset	Test precision	Test recall	Test F1
Bay Area Only	0.0	0.0	0.0
Three Regions	0.321	0.598	0.418

Table 5. Data diversity experiment. The "Three Regions" data set includes data from the Bay Area, Port Huenene, and Chula Vista. The test set includes only data from Morro Bay

models could generalize, we created a test set using data from a separate region of California (Morro Bay). Using a model trained entirely on data from the Bay Area led to extremely poor performance on the Morro Bay test set. This model failed to recognize, or even report, a single pixel of Salt Marsh.

In order to remedy the model's inability to generalize, we decided to train a new model on a more diverse data set. Using data from three separate regions of California (Bay Area, Port Huenene, and Chula Vista), we trained a model with the same architecture as before. This model performed considerably better on the Morro Bay test set, achieving an F1 score of nearly 0.42.

5.5. U-Net

Model	Val Precision	Val Recall	Val F1
U-Net	0.742	0.667	0.703
Base-Net	0.368	0.375	0.372

Table 6. Validation results for "U-Net" model.

Model	Test Precision	Test Recall	Test F1
U-Net	0.29	0.77	0.419
Base-Net	0.321	0.598	0.418

Table 7. Test results for "U-Net" model.

We also decided to experiment with a simplified version of the U-net architecture (see Figure 3). This model far outperformed our base model on both the "Three Region" and "Bay Area" train and validation sets. Despite these impressive results, the U-net based model did not outperform the basic model in all metrics on the Morro Bay test set.

The test results indicate that the U-Net architecture might be sufficiently complex such that it could overfit to the three separate train/val locations and not generalize well to the unseen train location. To continue with U-net, we anticipate needing a larger, more diverse dataset.

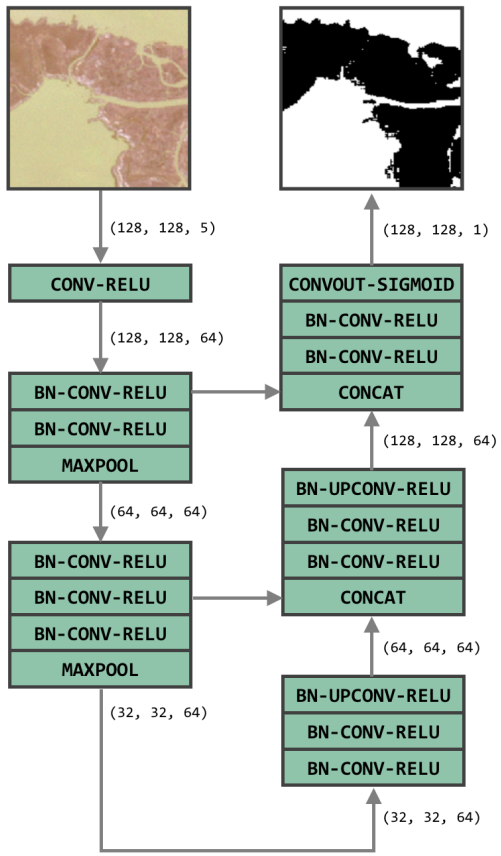

Figure 3. The "U-Net" model used for our experiments. See appendix for more details.

5.6. Coarse Ground Truth Labels

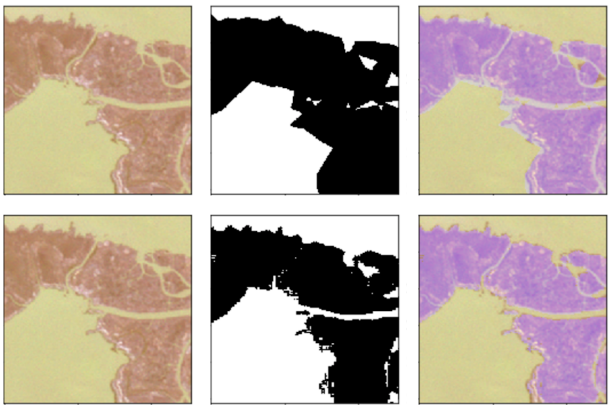

Figure 4. Top: ground truth labels. Bottom: basic model predicted labels. From let to right: satellite image, labels (salt marsh in black), composite (salt marsh in purple).

One unexpected challenge we encountered was discovering that the ground truth labels labels[4] were not as truthful as we had hoped. When examining results from the Sim-

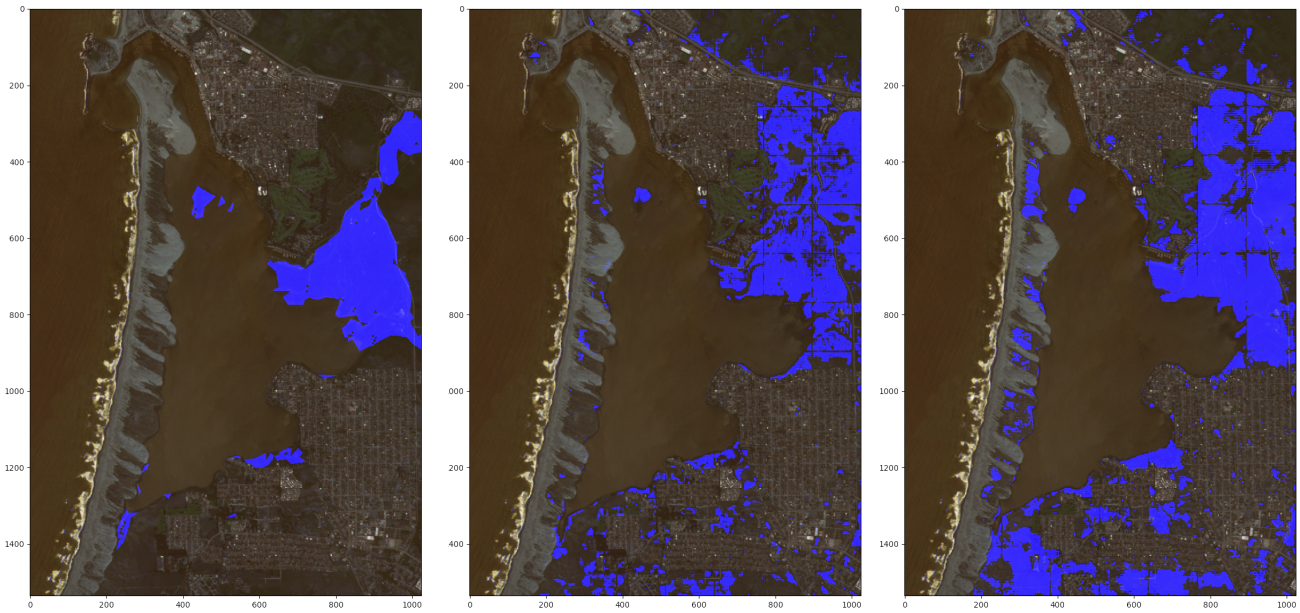

Figure 5. From left to right: ground truth labels, basic model predictions, and U-net predictions for Morro Bay.

ple Conv-net, it became clear that the ground truth labels were not matching up with the satellite imagery on a pixel-by-pixel scale. Probably, this is because the labels were not generated by evaluating high resolution data, but rather by some method of geo-referencing polygons that represent mapped salt marshes. As shown in Figure 4, this means that even for predictions that seem qualitatively solid, we take small hits in precision and recall metrics.

5.7. Best Area Estimation: Morro Bay

	Base Model	**U-net**	**Ground Truth**
Salt Marsh	4.09 km^2	6.12 km^2	2.49 km^2

Table 8. Salt marsh area estimates on Morro Bay test set.

We tested our model's ability to accurately estimate the total area covered by tidal salt marshes on the Morro Bay test set. We generated a naive estimate of the total area covered by salt marsh by taking the product of the number of pixels classified as salt marsh, and the real-world area covered by each pixel ($25m^2$). Sadly, neither of our best performing models generated an accurate estimate for the area covered by salt marshes in Morro Bay. However, this was largely expected given the relatively poor F1 scores for both models on this test set.

From both Table 8 and 9, it's clear that we are over-predicting salt marsh, and it appears the more complex the model, the more likely we are to pick out salt marsh in a satellite image.

	Pred Marsh	**Pred non-Marsh**
True Marsh	80408	19062
True non-Marsh	164513	1308881

Table 9. U-net confusion matrix of pixels in the Morro Bay test set.

6. Conclusion and Future Work

Overall, we are pleased with these results. Using a modified version of the U-net architecture, we were able to achieve an F1 score of just over 0.7 on our validation set. This is a promising result, especially considering the problems of coarse labels and imbalanced classes inherent in this data set.

However, in spite of with these promising results on our validation set, more work must be done before these models can be used to accurately estimate the total global coverage of tidal salt marshes.

- We would also like to compare our model's performance to a human labeled data set. This would give us a better idea of our current performance given the noise present in the current data set.
- Find better ground truth labels: the coarseness of our labels led to uncertainty in our training and evaluation. Additionally, there are ongoing efforts to restore salt marsh and coastal development can easily obliterate salt marsh, so a closer match in time between ground truth label creation and satellite imagery capture could be impactful.

- Augment current dataset: One issue that came to light during our test set experiments was that many of our models had trouble accurately predicting the class of pixels on or near the edges of tiles. A potential experiment in response would be to train and evaluate a model using overlapping tiles in order to alleviate the problems associated with lack of data near the edges of images. Other future directions that we would like to explore include training the models described in this paper using a much larger data set, and augmenting that data set using the horizontal and vertical flips of each tile.

- Expand dataset: It is clear that more training data leads to a more robust model. There are excellently mapped salt marshes in the UK and in regions of South America. It would be interesting to see how drastic diversity in the training set would affect test set predictions.

- Re-experiment with sub-sampling: Our experiments with sub-sampling were performed before we had diversified our dataset. The main problem encountered in the test set predictions is over predicting salt marsh. Although our original experiments indicated we should exclude all non-salt-marsh-containing tiles, it seems as if including more diverse, non-marsh examples would be one way alleviate this problem.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[2] R. Alshehhi and P. R. Marpu. Hierarchical graph-based segmentation for extracting road networks from high-resolution satellite images. *ISPRS Journal of Photogrammetry and Remote Sensing*, 126:245–260, 2017.

[3] Blue Carbon Initiative. Where is it? http://thebluecarboninitiative.org/ accessed June 2017.

[4] California Department of Fish and Wildlife Marine Resources Region. (2006). California coastal wetlands. Available at: http://purl.stanford.edu/dq706fk1136.

[5] M. Castelluccio, G. Poggi, C. Sansone, and L. Verdoliva. Land use classification in remote sensing images by convolutional neural networks. *arXiv preprint arXiv:1508.00092*, 2015.

[6] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[7] G. L. Chmura, S. C. Anisfeld, D. R. Cahoon, and J. C. Lynch. Global carbon sequestration in tidal, saline wetland soils. *Global biogeochemical cycles*, 17(4), 2003.

[8] F. Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[9] C. Huang, L. Davis, and J. Townshend. An assessment of support vector machines for land cover classification. *International Journal of remote sensing*, 23(4):725–749, 2002.

[10] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. *arXiv preprint arXiv:1611.07004*, 2016.

[11] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. *arXiv preprint arXiv:1611.09326*, 2016.

[12] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[13] M. B. Lyons, S. R. Phinn, and C. M. Roelfsema. Long term land cover and seagrass mapping using landsat and object-based image analysis from 1972 to 2010 in the coastal environment of south east queensland, australia. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:34–46, 2012.

[14] D. I. Moody, D. E. Bauer, S. P. Brumby, E. D. Chisolm, M. S. Warren, S. W. Skillman, and R. Keisler. Land cover classification in fused multisensor multispectral satellite imagery. In *Image Analysis and Interpretation (SSIAI), 2016 IEEE Southwest Symposium on*, pages 85–88. IEEE, 2016.

[15] A. Nowaczynski. Deep learning for satellite imagery via image segmentation. https://deepsense.io/deep-learning-for-satellite-imagery-via-image-segmentation/ accessed June 2017.

[16] M. Pal. Random forest classifier for remote sensing classification. *International Journal of Remote Sensing*, 26(1):217–222, 2005.

[17] M. Pal and P. M. Mather. An assessment of the effectiveness of decision tree methods for land cover classification. *Remote sensing of environment*, 86(4):554–565, 2003.

[18] Planet Labs. Open california. Available at: https://www.planet.com/explorer/.

[19] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 234–241. Springer, 2015.

[20] E. Shelhamer, J. Long, and T. Darrell. Fully convolutional networks for semantic segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(4):640–651, 2017.

[21] J. Siikamäki, J. N. Sanchirico, S. Jardine, D. McLaughlin, and D. Morris. Blue carbon: coastal ecosystems, their carbon storage, and potential for reducing emissions. *Environment: Science and Policy for Sustainable Development*, 55(6):14–29, 2013.

[22] F. Visin, M. Ciccone, A. Romero, K. Kastner, K. Cho, Y. Bengio, M. Matteucci, and A. Courville. Reseg: A recurrent neural network-based model for semantic segmentation.

In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 41–48, 2016.

[23] Weichelt, Rosso, Marx, Sandra, Reigber, Douglass, Heynen. The rapideye red edge band. https://apollomapping.com/wp-content/user_uploads/2012/07/RapidEye-Red-Edge-White-Paper.pdf.

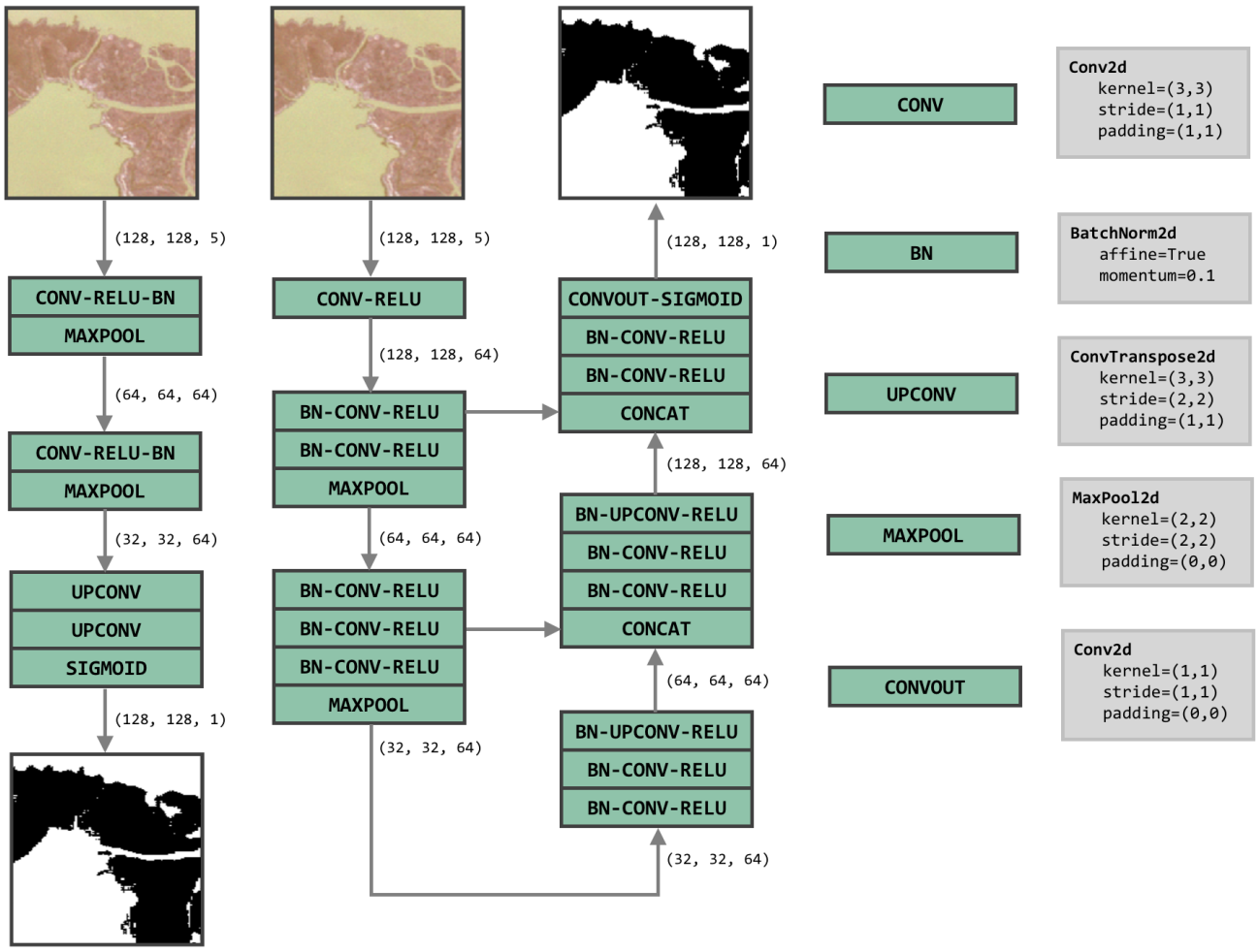

Figure 6. Details for "Basic" and "U-Net" models.