remote sensing

MDPI

Article

Weakly Supervised Deep Learning for Segmentation of Remote Sensing Imagery

Sherrie Wang [1,2,*], William Chen [3], Sang Michael Xie [3], George Azzari [2] and David B. Lobell [2]

[1] Institute for Computational and Mathematical Engineering, Stanford University, Stanford, CA 94305, USA
[2] Department of Earth System Science, Stanford University, Stanford, CA 94305, USA; gazzari@stanford.edu (G.A.); dlobell@stanford.edu (D.B.L.)
[3] Department of Computer Science, Stanford University, Stanford, CA 94305, USA; wic006@stanford.edu (W.C.); xie@cs.stanford.edu (S.M.X.)
* Correspondence: sherwang@stanford.edu

Received: 5 November 2019; Accepted: 29 December 2019; Published: 7 January 2020

Abstract: Accurate automated segmentation of remote sensing data could benefit applications from land cover mapping and agricultural monitoring to urban development surveyal and disaster damage assessment. While convolutional neural networks (CNNs) achieve state-of-the-art accuracy when segmenting natural images with huge labeled datasets, their successful translation to remote sensing tasks has been limited by low quantities of ground truth labels, especially fully segmented ones, in the remote sensing domain. In this work, we perform cropland segmentation using two types of labels commonly found in remote sensing datasets that can be considered sources of "weak supervision": (1) labels comprised of single geotagged points and (2) image-level labels. We demonstrate that (1) a U-Net trained on a single labeled pixel per image and (2) a U-Net image classifier transferred to segmentation can outperform pixel-level algorithms such as logistic regression, support vector machine, and random forest. While the high performance of neural networks is well-established for large datasets, our experiments indicate that U-Nets trained on weak labels outperform baseline methods with as few as 100 labels. Neural networks, therefore, can combine superior classification performance with efficient label usage, and allow pixel-level labels to be obtained from image labels.

Keywords: deep learning; image segmentation; weak supervision; agriculture; Landsat; land cover classification

1. Introduction

Automatic pixel-wise classification of remote sensing imagery enables large-scale study of land cover and land use on the Earth's surface, and is relevant to applications ranging from deforestation mapping [1] and development surveyal [2] to ice sheet monitoring [3] and disaster damage assessment [4]. In computer vision, pixel-wise classification is a classic task known as semantic segmentation, and has been tackled with increasing success in recent years due to the development of deep convolutional neural networks (CNNs) [5–9] and large labeled benchmark datasets on which to test architectures [10–13]. The advantage of CNNs over machine learning methods that take the features of a single pixel as input—such as random forests, support vector machines (SVMs), and logistic regression—is their ability to consider a pixel's context (that is, the pixels near that pixel) in addition to the pixel's own features when performing classification [14]. This context may be helpful when, for example, a pixel of grassland and a pixel of cropland share similar phenological and spectral features, but a wider view of cropland reveals that it is divided into rectangular parcels while grassland is not.

Traditional training of CNNs for segmentation, including the latest methods developed for remote sensing imagery (Section 2), provides the model with an image as input, and computes the loss

between the network's pixel-wise output and a segmented ground truth [15]. Here "segmented" means that every (or nearly every) pixel in the image has its own label. Segmented labels are dense in information but time-consuming and expensive to generate. In amassing 200,000 segmented labels, the creators of the Microsoft COCO (Common Objects in Context) dataset wrote that each image took on average 10 min to segment and only one-third of crowdsourced workers passed their accuracy criterion [11]. For some applications of remote sensing, there is the added challenge of labels requiring expert knowledge (e.g., identifying cropland in Sub-Saharan Africa) or on-the-ground surveys (e.g., measuring household wealth) to obtain [16]. This can be especially true in areas of the world where we know little about land cover and land use, and can benefit a great deal from additional knowledge. These challenges place many applications in a "small data" regime, with labels that do not resemble the typical fully segmented ground truth.

We explore in this work whether segmentation of remote sensing imagery can be achieved with high accuracy using (1) sparse pixel-level labels or (2) image-level labels, which we will call sources of *weak supervision*. We use the term *weak supervision* to refer to leveraging high-level or noisy input from subject matter experts or crowdsourced workers to perform supervised learning [17]. These ways of labeling data have a number of advantages: they already exist in abundance, are faster and cheaper to obtain per label de novo, may be easier to obtain from non-experts, and may be the only type of data obtainable on the ground. We study how segmentation performance varies with dataset size and how methods trained via weak supervision compare with those trained directly on segmented labels across both deep neural networks and other machine learning methods (random forest, SVM, logistic regression). Note that we will use the term *segmentation* to refer to pixel-level classification, and reserve *classification* for image-level tasks, following the lexicon of computer vision.

Our task of interest is cropland segmentation, though the methods developed are general enough to be applied to any segmentation task. We are motivated to assess weakly supervised methods in this domain because the ability to accurately locate and characterize cropland enables agricultural monitoring and food security assessment [18,19], but the ground truth labels available or easiest to collect in food-insecure regions are often suitable only for weak supervision of segmentation.

Given the lack of segmented labels in food-insecure regions for validation, we began our methodological development in the United States, where ample crop segmentation ground truth is available via the USGS's Cropland Data Layer (CDL) [20]. We simulated the weakly supervised setting (Figure 1) by training a CNN on either (1) a sparsely labeled segmentation problem or (2) an image classification problem. Inputs to the model are Landsat 8 annual median composite images with 7 optical bands. Our contributions can be summarized as follows.

1. With sparse pixel labels, we trained a CNN to perform cropland segmentation by masking out all but one pixel in each image on which to compute the loss. We show that randomization of this pixel's location is important for segmentation accuracy across the entire image and ability to use single-pixel classification as a proxy task for segmentation.
2. With image-level labels, we used class activation maps (CAMs) developed by Zhou et al. [21] to extract segmentation predictions from an intermediate CNN layer. The CAMs were converted to predictions via a thresholding algorithm that takes into account the distribution of image-level labels in the dataset.
3. We demonstrate that, while CNNs are already known to outperform other machine learning methods when trained on large datasets with high quality labels, they can also outperform random forest, SVM, and logistic regression models when trained on small numbers of weak labels. It is therefore possible to combine the high performance of deep neural networks with ground truth labels that are easy to obtain. The transfer of image labels to pixel labels also demonstrates a new possibility obtained by moving from established machine learning methods to deep learning.

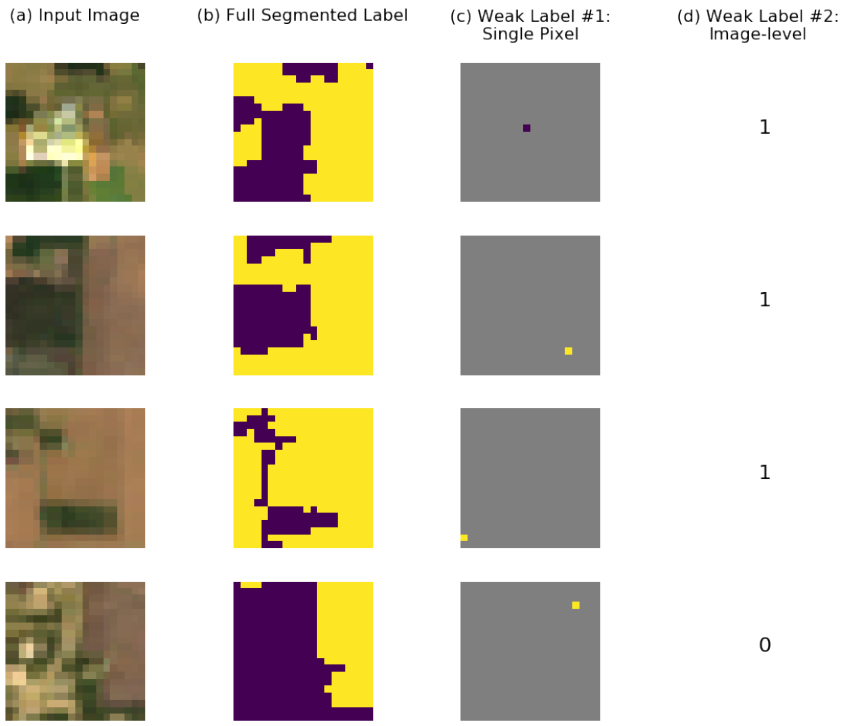


Figure 1. Examples of (**a**) Landsat images, (**b**) their corresponding full segmented labels, and (**c**,**d**) two types of weakly supervising labels. (**c**) Single pixel labels are available from datasets of geotagged points. Gray pixels' labels are not known. (**d**) Image-level labels provide high-level information about the image but labels of individual pixels are not known. We demonstrate methods to predict the full segmented label, given only one of the weakly supervising labels.

2. Related Work

The growing body of research that adapts deep neural networks created for natural image segmentation to remotely sensed imagery has largely focused on two areas. The first is creating architectures that allow CNNs and recurrent neural networks (RNNs) to adapt to the unique characteristics of satellite and aerial imagery. These works have successfully demonstrated the use of CNNs and RNNs for land cover classification [22,23], cloud masking [24,25], building footprint segmentation [26–29], ship segmentation [30], and road segmentation [31,32]. In each case, the ground truth used is densely segmented labels. A second group of pursuits has been to create large datasets, usually of very high resolution imagery, annotated with segmentation ground truth labels [33–36]. The most recent and largest of these include DeepGlobe 2018 [37] and BigEarthNet [38]. These datasets, especially when paired with competitions on platforms such as Kaggle and CrowdAnalytix, provide much-needed benchmarks and are catalysts for method development.

While these methods show us what is achievable on large, well-labeled datasets, there remains a mismatch between the datasets that are available or easy to collect in many applications and the tools built so far to perform segmentation. First, many deep neural networks contain vast numbers of trainable parameters and require large datasets to achieve high performance, and there are often only small quantities of ground labels available for training. To address the small data regime, some recent works have explored the use of transfer learning and semi-supervised learning techniques. Transfer learning makes use of labeled data in a setting similar to the problem of interest, which lacks labels. Kaiser et al. showed that accurate segmentation of a city's buildings and streets could be produced by training a CNN on large-scale, highly noisy labels from OpenStreetMap in other cities [39] , while Kemker et al. trained their CNN on synthetic aerial imagery before fine-tuning on real data [40]. Semi-supervised learning boosts performance when large quantities of unlabeled data can be leveraged to augment labeled data. For example, Kang et al. created pseudo-labeled samples

using non-deep learning methods to improve deep learning-based segmentation on small labeled datasets [41].

A second source of mismatch, and the one addressed in this work, is that segmentation predictions are often desired in settings where ground truth labels that are available or feasible to collect are at the point or image level. Methods to address this for natural images include work by Hong et al. that used "bridging layers" to share information between separate classification and segmentation networks [42], and work by Pinheiro and Collobert where a CNN trained on image classification contained intermediate input-sized layers that were aggregated to obtain a segmentation prediction [43]. Our approach for coupling image classification and segmentation is similar to the latter in concept, but we combine a U-Net architecture with class activation mapping developed by Zhou et al. [21]. U-Nets were designed for image segmentation on small- to moderate-sized biomedical datasets [7], and class activation maps (CAMs) allow models that are trained for classification tasks to localize class-specific image regions from the target image. To use single-pixel labels for segmentation, we masked out all other pixels of an image when computing the loss, similar to how pixels with the "void" class label are masked out when training on the Pascal VOC segmentation dataset [10].

In cropland mapping, most work to date has performed segmentation using features at an individual pixel level [18–20,44,45]. These methods, which include random forests and SVM, have become easy to implement at large scale due to the development of platforms such as Google Earth Engine. However, unlike CNNs, they do not automatically take into account the spatial context of each pixel, which can lend a great deal of information about whether that pixel is cropland or not. Methods have been created to fuse "object-based" features at larger spatial scales with pixel features to improve random forest- and SVM-based cropland maps [45,46], but an advantage of CNNs is that the network learns how to use spatial context to aid segmentation and is not limited by hand-engineering.

3. Dataset

We describe our study area in the Midwestern United States, the remote sensing dataset used for classification and segmentation, and how we obtained image-level and pixel-level labels.

3.1. Study Area

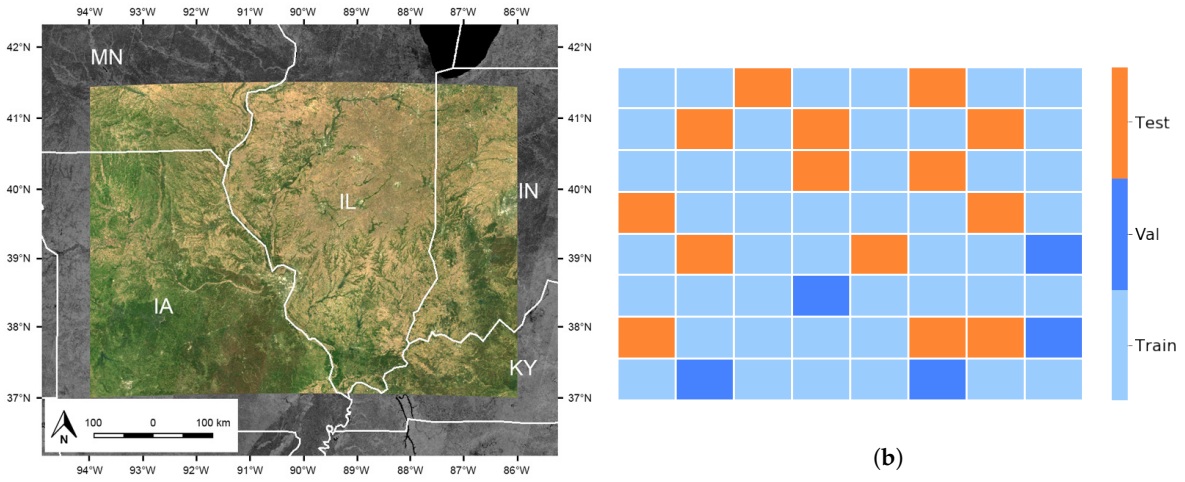
The study area is shown in Figure 2a, spanning from 37° N to 41°30′ N and from 94° W to 86° W. It covers an area of over 450,000 km^2 in the United States Midwest, intersecting the states of Illinois, Iowa, Indiana, Missouri, and Kentucky. We chose this region because the United States Department of Agriculture (USDA) maintains high quality pixel-by-pixel land cover labels across the US, allowing us to evaluate the quality of our segmentation. Furthermore, we applied our methods to a large area to show that they can scale spatially. Land cover-wise, the study region is 44% cropland and 56% non-crop (mostly temperate forest).

3.2. Remote Sensing Data

The Landsat Program is a series of Earth-observing satellites jointly managed by the USGS and NASA. Landsat 8 provides moderate-resolution (30 m) satellite imagery in seven surface reflectance bands—ultra blue, blue, green, red, near infrared, shortwave infrared 1, and shortwave infrared 2 [47]—designed to serve a wide range of scientific applications. Images are collected on a 16-day cycle.

Using Google Earth Engine, we found all the Landsat 8 Surface Reflectance Tier 1 images that intersect the study area and were taken between 1 January 2017 and 31 December 2017. We then computed a single composite image from this image collection by taking the median value at each pixel and band. Since Landsat imagery, and satellite imagery more broadly, is affected by different types of contamination, such as clouds, snow, and shadows [48], we used the quality assessment band delivered with the Landsat 8 images to mask out clouds and shadows prior to computing the median composite. The resulting seven-band image spans 4.5 degrees latitude and 8.0 degrees longitude and

contains just over 500 million 30-by-30 meter pixels. To prepare the imagery to be input to a CNN, we divided the composite into approximately 200,000 tiles each of dimension 50 × 50 pixels.



(**a**) (**b**)

Figure 2. (**a**) Landsat 8 median composite showing our study area in the Midwestern United States for the year 2017. (**b**) Spatial split of the dataset into training, validation, and test sets for method evaluation. The geographic split reduces spatial correlations that may lead to inflated validation and test set accuracies. The non-test set is split into 10 folds for cross-validation; only one fold (darker blue) is visualized here.

3.3. Pixel-Level Labels

The Cropland Data Layer (CDL) is a raster geo-referenced land cover map collected by the USDA for the entire continental United States [20]. It is offered at 30 m resolution, so that each Landsat 8 pixel has a corresponding CDL label. CDL includes 132 detailed classes spanning field crops, tree crops, developed areas, forest, and water. In our dataset across the corn belt, we observe 78 CDL classes. The four most common classes—deciduous forest, corn, soybean, and grassland/pasture—account for 85% of the dataset. The remaining classes are each less than 5% of the dataset. For our classification task, we aggregated all crop classes into a single "cropland" class, and non-crop classes into a single "non-cropland" class.

For the remainder of our study, we treat CDL labels as ground truth and use them to evaluate the performance of our methods. The quality of our evaluation therefore depends on the quality of the CDL labels. CDL is created yearly using imagery from Landsat 8 and the Disaster Monitoring Constellation (DMC) satellites, and a decision tree algorithm is trained and validated on ground samples. The accuracy of CDL labels varies by class; for the top classes in our dataset, accuracies detailed in the CDL metadata generally exceed 90% [20]. Since we simplify the CDL labels into $\{0,1\}$ for non-crop and cropland, and non-crop and cropland discrimination is easier than crop type discrimination, the binary labels used to supervise our image classification are likely even more accurate than CDL is for individual crop classes.

3.4. Image-Level Labels

Since our goal is to evaluate the possibility of generating segmentation labels from a CNN trained on image-level labels, image-level labels are needed for our dataset. For each 50 × 50-pixel tile (each covering 2.25 km^2), we computed a binary label $\in \{0,1\}$ based on whether the majority of pixel-level CDL labels in that tile are crop pixels or not. The label 1 indicates that the tile is "more than 50% cropland" and the label 0 indicates that the tile is "less than 50% cropland". The class balance of the dataset is shown in Table 1. This labeling scheme was chosen because it is quick for humans to assess, and is therefore a realistic label to crowdsource or ask domain experts to generate de novo. We leave

for future work the exploration of other labeling thresholds or schemes, such as mere presence of a class of interest.

Table 1. Summary statistics for the US Midwest dataset.

Dataset Split	# Tiles	# Pixels	% Tiles Majority Cropland	% Pixels Cropland
Training	≤100,000	≤250,000,000	40% ± 2%	41% ± 2%
Validation	15,170	37,925,000	40% ± 17%	41% ± 14%
Test	42,476	106,190,000	54%	53%
Total	194,176	485,440,000	43%	44%

3.5. Training, Validation, and Test Splits

Since land cover type and characteristics vary smoothly across space, adjacent tiles may contain parts of the same crop field, forest, city, etc. A random split of the 200,000 tiles into training, validation, and test sets will therefore result in a test set performance that overestimates how well the model generalizes to new areas within the study region.

To reduce the performance-inflating effect of spatial auto-correlation, we split the study region into 64 rectangles geographically, and randomly assigned 50 rectangles to a training and validation set and 14 rectangles to a test set. Within the training and validation set, the 50 rectangles were split into 10 folds of 5 rectangles each. One such split is shown in Figure 2b; the other nine are shown in Figure A1 (Appendix A.1). With this geographic train-validation-test split, the test set metrics are an estimate of model performance when applied to a new area within the study region (but not a measure of model generalization to other regions in the US or the world).

To tune machine learning hyperparameters, we trained the model on 9 folds (45 rectangles) and validated on 1 fold (5 rectangles). Using k-fold cross-validation ($k = 10$) allows us to obtain error estimates when tuning hyperparameters and evaluating the performance of different methods. Note that all splits remained the same across experiments done with different training set sizes; a training set size of 1000 is a sub-sample of the tiles in the corresponding full training set.

4. Methods

In this section, we describe the methods used to (1) train a CNN on dense segmentation labels, (2) train a CNN on single-pixel labels for segmentation, and (3) transfer a CNN trained on image classification to the task of segmentation. We also describe data augmentation techniques used to expand our effective dataset size and baseline models such as random forests.

An overview of the CNN architectures trained with single-pixel labels and image labels are depicted in Figures 3 and 4, respectively.

4.1. Data Augmentation

At training time, we employed random rotations and flips to increase our effective dataset size, with the assumption that image-level and segmentation labels are invariant to rotations and flips. We do not employ stretching or rotations that are not a multiple of 90°, since doing so may alter the image-level label of a tile.

4.2. Convolutional Neural Network Architecture

The deep learning models discussed in this paper share the same core U-Net architecture, illustrated in Figures 3 and 4. The U-Net is a convolutional neural network designed originally for segmenting biomedical imagery [7], intended to perform well with a relatively small number of training images and to yield segmentation at the same resolution as the input image. An input image with C channels, height H, and width W (denoted as dimension $C \times H \times W$) is "encoded" by layers in

the first half of the network to yield a low dimensional representation. This representation contains high-level information on the image being segmented. The second half of the network then "decodes" the representation back to the original image height and width, with $K-1$ channels parameterizing the categorical probability distribution over K classes at each pixel.

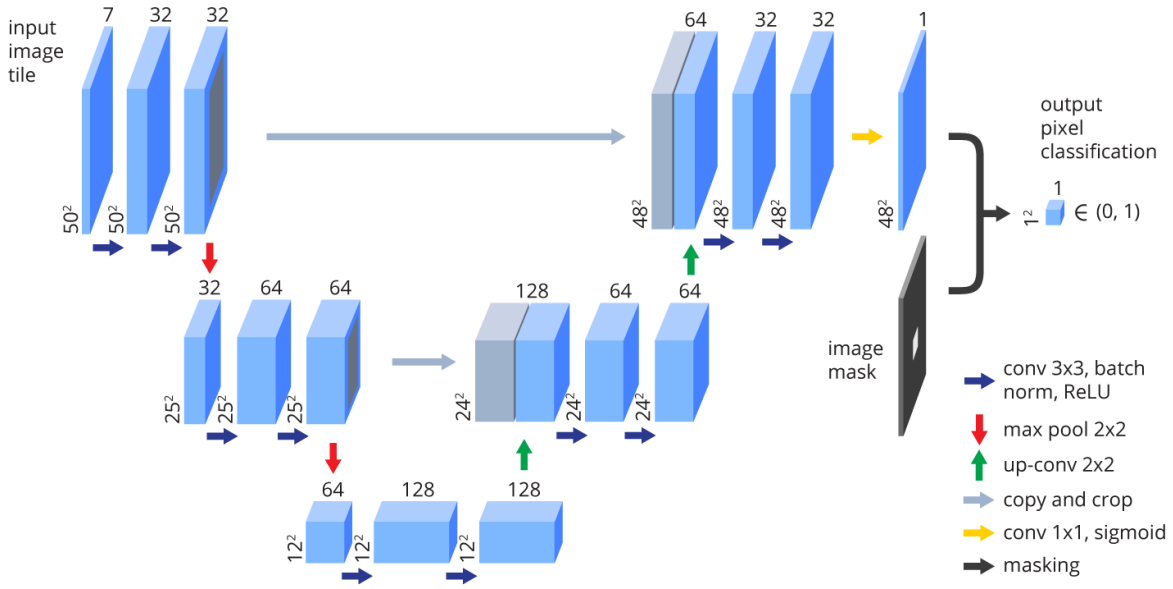


Figure 3. A U-Net (a type of CNN) with two down-convolutional blocks and two up-convolutional blocks is shown here. A block is comprised of two convolutional-batch norm layers followed by a max pool or up-convolutional layer. We used masking at the loss computation step to train a U-Net on single pixel labels. The pre-masking output is the network's segmentation prediction.

For our binary cropland versus non-cropland classification task, K is 2, so the U-Net output is of dimension $H \times W$. Since spatial information is lost during encoding, features from the first half of the network are concatenated to those of the second half to re-introduce spatial information and allow for precise segmentation. In our U-Net, the input image is of dimension $7 \times 50 \times 50$, and the output is 48×48 due to max-pooling and up-convolutional layers operating in multiples of 2 (Figures 3 and 4). We compared the innermost 48×48 pixels in $\mathbf{y}$ to the output when computing segmentation accuracy.

Since neural network performance depends on a number of tunable hyperparameters, we used cross-validation with grid search to select the U-Net network depth, number of filters, L_2 regularization strength, learning rate, and batch size. The details of hyperparameter search, deep learning frameworks, and hardware used are described in Appendix A.3. For the remainder of this paper, we will show the performance of U-Net and U-CAM models with optimized hyperparameters shown in Table A4. We found that a U-Net with 4 encoding and 4 decoding blocks, with 64 initial filters, L_2 regularization of strength 10^{-4} to 10^{-3}, learning rate of 10^{-3}, and batch size of 32 performed the best during cross-validation.

The U-Nets were trained across 20+ epochs; to calculate test set metrics, we chose the epoch with the highest task validation accuracy to evaluate on the test set. This was done for each training set fold to obtain standard errors.

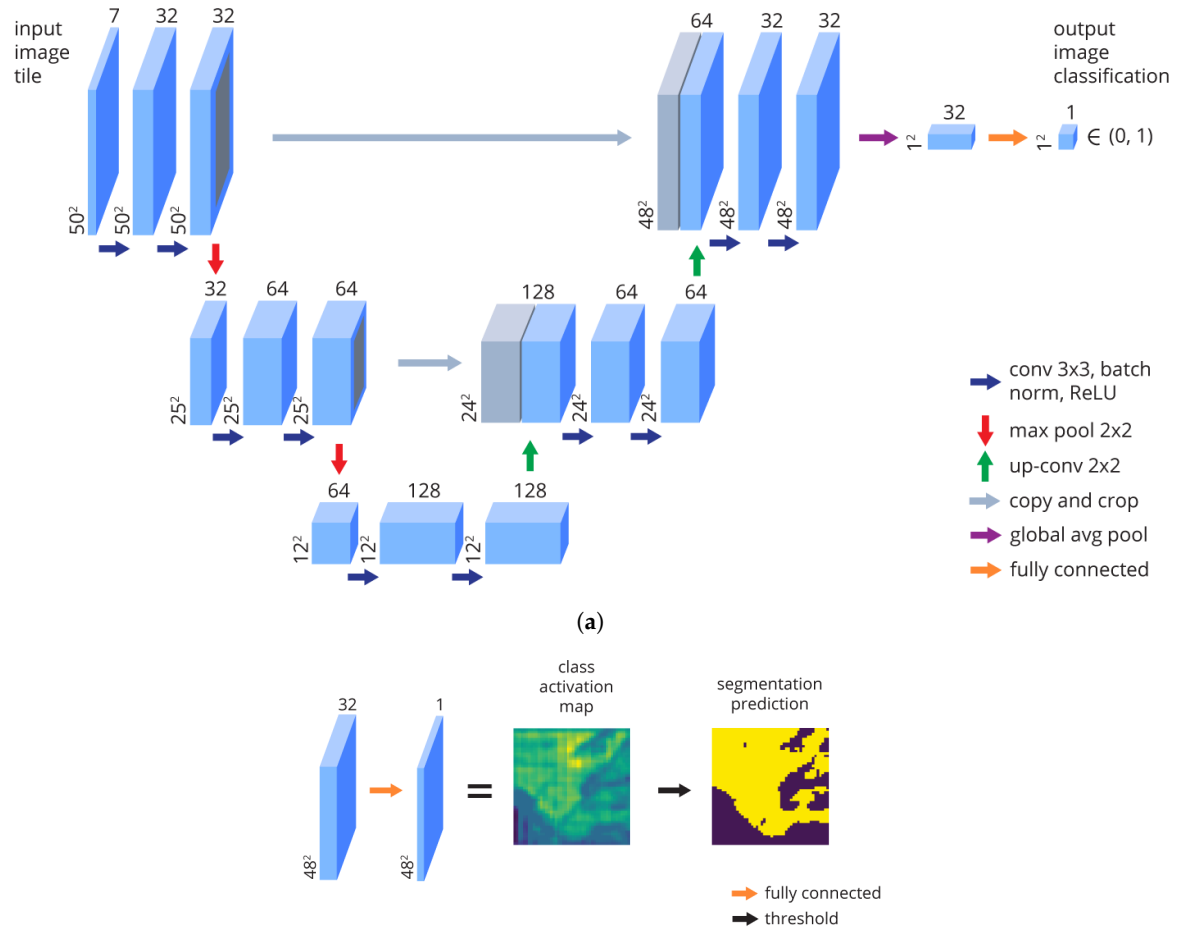


Figure 4. (**a**) We combined the U-Net and class activation map (CAM) to create a new architecture for weakly supervised segmentation via image classification. (**b**) A CAM is obtained via a weighted sum across the filter dimension of the last convolutional layer output, where the weights come from the fully connected layer. The CAM is then thresholded to obtain a hard segmentation prediction.

4.3. End-to-End Segmentation Using Dense Labels

To get an upper bound on how well U-Nets can perform on cropland segmentation, we performed end-to-end training using dense labels. Training minimized the binary cross entropy loss, defined as a function of each sample as

$$\ell(\mathbf{y}, \hat{\mathbf{y}}) = -\frac{1}{WH} \sum_{j=1}^{H} \sum_{k=1}^{W} \left[\mathbf{y}_{jk} \cdot \log \hat{\mathbf{y}}_{jk} + (1 - \mathbf{y}_{jk}) \cdot \log(1 - \hat{\mathbf{y}}_{jk}) \right], \quad (1)$$

for model predictions $\hat{\mathbf{y}}$ of dimension $H \times W$ and segmentation label $\mathbf{y}$ of dimension $H \times W$. The notation $\mathbf{y}_{jk}$ denotes the pixel at the (j, k) spatial location of $\mathbf{y}$. Note that $\hat{\mathbf{y}}$ is a function of the input image $\mathbf{x}$ and model parameters θ, and each pixel's prediction is a real number in the range $(0, 1)$ representing the probability of that pixel being cropland. For applications with more than 2 classes, the U-Net can easily be adapted to output more prediction classes per pixel, and training would minimize a categorical cross entropy loss.

4.4. End-to-End Segmentation Using Sparse Labels

Though our dataset has cropland labels at all pixels, we simulated the sparse label setting by sampling one pixel per input image to be the labeled one; all other pixel labels were masked out and

not seen by the U-Net. More formally, for each input image, we sampled uniformly at random one of the 2500 pixels in $\mathbf{y}$, whose spatial location we denote (j^*, k^*), to be the only label in the image whose binary cross entropy is computed in the loss.

For an input sample, the masked U-Net loss can be written

$$\ell(\mathbf{y}, \hat{\mathbf{y}}) = -\left[\mathbf{y}_{j^*k^*} \cdot \log \hat{\mathbf{y}}_{j^*k^*} + (1 - \mathbf{y}_{j^*k^*}) \cdot \log(1 - \hat{\mathbf{y}}_{j^*k^*})\right] \tag{2}$$

We emphasize that, although the position of the labeled pixel is different across tiles, it is fixed for the same tile during every epoch of training and evaluation. On a dataset actually comprised of single-pixel labels, there is no need for random sampling of the label position. Rather, one would sample a satellite image tile that contains the labeled point at a random position.

To obtain dense segmentation predictions from the trained U-Net, we simply skipped the masking step. The unmasked output $\hat{\mathbf{y}}$ from the network was compared against the dense segmentation label $\mathbf{y}$ to obtain full segmentation accuracies.

4.5. Obtaining Segmentation from Image Classification

4.5.1. Image Classification

We modified the U-Net to perform image classification by replacing the last 1×1 convolution with a global average pooling layer followed by a fully connected layer that outputs a single number $\in (0, 1)$. The global average pooling layer computes the mean value across all spatial dimensions of the input and is used to recover a class activation map (Section 4.5.2). A diagram of a 2-layer U-Net modified for image classification is shown in Figure 4a.

The classification task is to detect whether the majority ($\geq$50%) of pixels in an image are in the "cropland" category. Recall that, for our US Midwest dataset, segmentation labels from CDL were converted into binary labels (Section 3.4). To train the model, we used an image-level binary cross entropy loss, which is defined for each input as

$$\ell(y, \hat{y}) = -\left[y \log \hat{y} + (1 - y) \log(1 - \hat{y})\right], \tag{3}$$

where $\hat{y}$ is the image-level model prediction and y is the image-level binary label.

4.5.2. Class Activation Maps

To derive segmentation from a network trained for image classification, we used class activation maps (CAMs) following the work of Zhou et al. [21]. CAMs arose from the discovery that intermediate layers of CNNs detect objects despite no supervision on the location of objects at the time of training.

A diagram of how to compute a CAM is shown in Figure 4b; it is the weighted sum of the last convolutional layer's outputs, where the weights are from the fully connected layer. Mathematically, the CAM is defined as

$$\text{CAM} = \sum_c (w_c f_c + b_c) \tag{4}$$

for the last convolutional layer output f, fully connected layer weights w, and fully connected layer biases b. The sum is over the filter dimension c.

Notice that if f is of dimension $C \times H \times W$, then the CAM has dimension $H \times W$. Intuitively, the CAM shows how much each pixel of the last convolutional output was "activated" for cropland. We discuss how this activation is converted to a valid probability in the next section.

4.5.3. Segmentation Threshold

The values of a CAM can in theory take on any real value, with a higher value indicating a higher activation for cropland in the corresponding pixel. In practice, we observed CAM values falling in the interval $[-10, 10]$. To convert the CAM to a segmentation prediction that is 0 or 1 at each pixel, we set a single threshold activation value, above which we will predict a value of 1 and below which we will predict a value of 0.

Notice that the threshold value cannot simply be assumed to be 0. Although a value of 0 evaluates to a cropland probability of 0.5 when passed through the last layer of the U-CAM network (sigmoid layer), the sigmoid layer's input is obtained via a weighted sum over filter dimensions and average over spatial dimensions. Since the sigmoid of a sum does not equal the sum of sigmoids, the threshold of 0 does not correspond to a probability of 0.5 at each CAM pixel. In practice, we observed optimal thresholds that deviated from 0, generally within the range $[-1, 1]$.

To determine the optimal threshold, we found the threshold that maximizes image-level prediction accuracy on tiles in the training set. This algorithm proceeds as follows. At epoch t,

1. Compute the CAM for each training tile as described in Section 4.5.2.
2. Enumerate a possible set of threshold values V.
3. For each training tile and possible threshold value $v \in V$,
   (a) Let the prediction at pixel (j, k) be $s_v(j, k) = \mathbb{1}\{\text{CAM}(j, k) \geq v\}$. That is, if the CAM value is equal to or exceeds the threshold, predict that the pixel is cropland.
   (b) Compute the image-level prediction $\hat{y}_v$ from the segmentation prediction s_v in the same way that image-level labels were determined from the segmented ground truth (or human labeling):

$$\hat{y}_v = \mathbb{1}\left\{\frac{1}{WH}\sum_{(j,k)} s_v(j, k) \geq 0.5\right\}$$

   In other words, an image whose segmented prediction has a majority of pixels ($\geq$50%) predicted to be cropland would be labeled 1; otherwise 0.
4. Find the threshold that maximizes the accuracy of image-level predictions across all training tiles, i.e.,

$$v^* = \arg\max_{v \in V} \sum_{i=1}^{m} \mathbb{1}\left\{\hat{y}_v^{(i)} = y^{(i)}\right\}$$

5. Return the segmentation prediction s_{v^*} for each training and validation image.

We point out that this way of determining a threshold and creating segmented predictions required another loop through the training set, which increased training time. If segmentation labels are available for some tiles in the training set, they can be used to find the threshold instead.

4.6. Baseline Models

We compared the masked U-Net and U-CAM methods against a few commonly used machine learning baselines: logistic regression, support vector machines (SVM), and random forests. All have been used in the field of remote sensing to classify land cover. For each method, we optimized over its hyperparameters across dataset sizes to provide the highest performing baseline possible; the best hyperparameters are shown in Tables A1–A3. Descriptions of the three baselines and the hardware we used to run them can be found in Appendix A.2.

The same training, validation, and test set splits were used for the baseline models as for the deep neural networks. For comparison against the masked U-Net, the center pixel of each image and its

label were provided to the baseline models for training and validation. For comparison against the U-CAM model, the same image-level label ("more than 50% cropland" or "less than 50% cropland") was used to label all pixels in the image ($50 \times 50 = 2500$ pixels) in the training set fed to baselines. In other words, all pixels in an image labeled "more than 50% cropland" are labeled as "cropland" and all pixels in a "less than 50% cropland" image are labeled as "non-cropland". Evaluation on the validation and test sets in this setting was, however, still performed using pixel-level labels.

5. Results

Here we summarize the results of the (1) U-Net trained on dense segmentation labels, (2) masked U-Net trained on single-pixel labels, and (3) U-CAM transferring image classification to segmentation. For a description of baseline model results, see Appendix A.5.

5.1. U-Net Oracle Trained on Dense Segmentation Labels

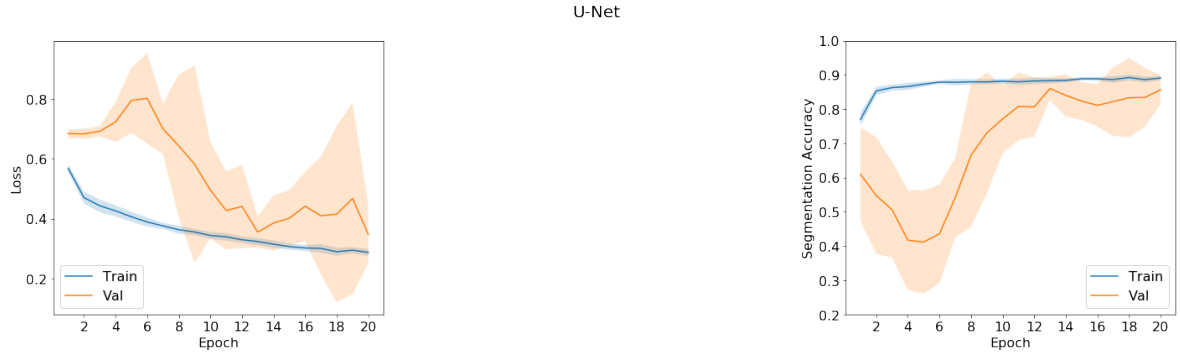
Figure 5a shows fully supervised U-Net loss and segmentation accuracy across 20 epochs of training, averaged for the 10 training folds. Early in training, cross-entropy loss decreases rapidly and segmentation accuracy increases steeply. The model begins to perform well on the training set after only one epoch of training, while performance on the validation set slowly improves after more epochs (around 10 for $n = 200$). We viewed the fully supervised U-Net as an oracle that provides an upper bound on how well we can expect the U-Net architecture to perform at segmentation given the best possible labels. Note in Figure 6 that, at 100,000 training samples, U-Net test set accuracy reaches 92%, which is approaching the accuracy of CDL, our ground truth.

5.2. Obtaining Segmentation from Sparse Pixel Labels

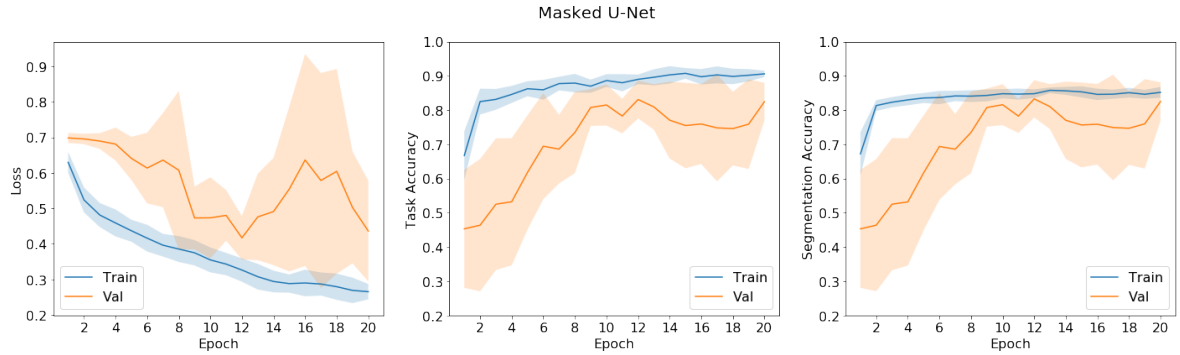
When using one labeled pixel per image to supervise a U-Net for segmentation, we first observe that decreasing cross-entropy loss and increasing task accuracy (single-pixel classification) corresponds to increasing segmentation accuracy as the model trains (Figure 5b). The closer the correspondence between task accuracy and segmentation accuracy, the more we can use the task accuracy to select the best training epoch for segmentation and be confident in the implied segmentation accuracy.

We trained the masked U-Net model on tiles with either randomized or constant label positions, and found that randomness in the position of the labeled pixel across tiles was important for avoiding overfitting and achieving high correlation between task accuracy and segmentation accuracy (Figure A2 and Appendix A.4). Because the task-segmentation correlation in the case of randomized label positions is close to 1.0 on the validation set (Figure 7a), a model with high task validation accuracy is nearly guaranteed to also yield a high segmentation accuracy.

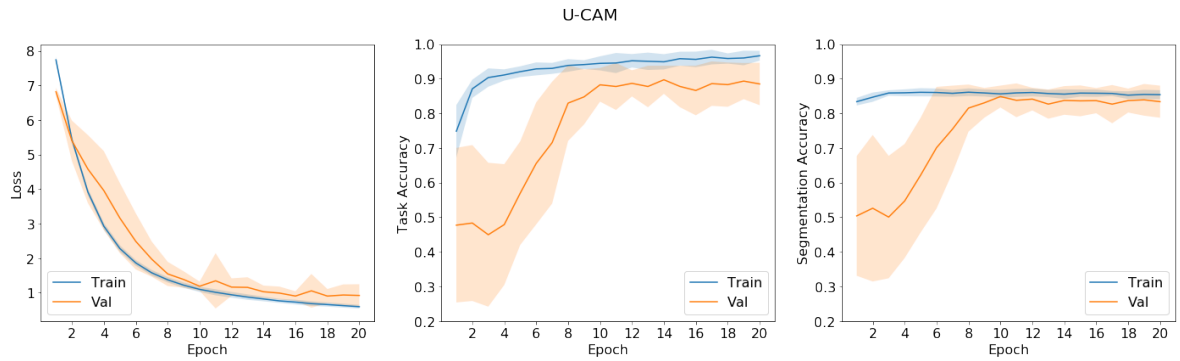
Figure 6a compares the test set accuracy of the masked U-Net against baseline and oracle methods across training set sizes from 10 to 100,000. Because our label classes are fairly balanced (Table 1), we primarily report our findings using the accuracy metric; the findings are similar for precision, recall, and F1-score metrics, shown in Table 2 for $n = 100$ and $n = 1000$. At training sizes below $n = 100$, the masked U-Net has lower accuracies for cropland classification than all three baselines. This suggests that it is difficult to learn the large number of parameters in the U-Net well with under 100 labeled pixels. At training sets larger than $n = 100$, however, the advantage of seeing a pixel's context—even without their labels—allows the masked U-Net to outperform the baselines. At $n = 1000$, the masked U-Net achieves a segmentation accuracy of 0.88, compared to SVM at 0.85, random forest at 0.84, and logistic regression at 0.81. The masked U-Net accuracy continues to increase with training size and approaches the performance of the U-Net upper bound; even at n = 100,000 the model still benefits from more training samples.



(**a**)



(**b**)



(**c**)

Figure 5. Training and validation set loss, task accuracy, and segmentation accuracy averaged across 10 runs of 20 epochs at $n = 200$ for the (**a**) U-Net, (**b**) masked U-Net, and (**c**) U-CAM models. The "task" refers to single pixel classification for the masked U-Net and image classification for the U-CAM model. The U-Net does not perform a proxy task, so it has no middle panel. One standard deviation error bars are shown in the shaded area.

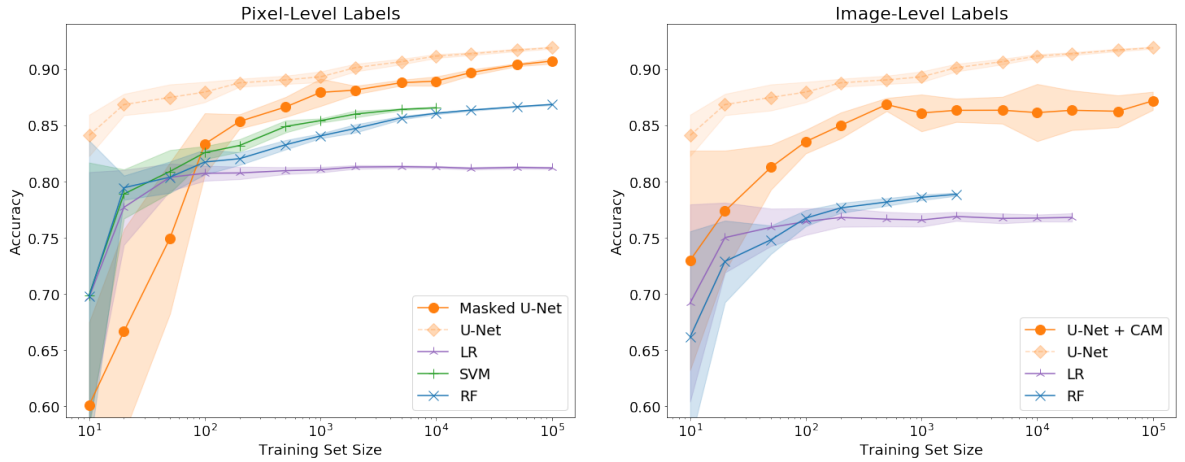


Figure 6. Test set cropland segmentation accuracy versus training set size given (**a**) single pixel labels and (**b**) image labels. Baseline methods include logistic regression (LR), support vector machine (SVM), and random forest (RF). Upper bound for the performance of a U-Net architecture is given by the U-Net with fully segmented labels. Deep learning methods shown are (**a**) U-Net trained on masked loss and (**b**) U-Net with class activation map (U-CAM) to transfer image classification to segmentation. Standard errors result from runs on 10 training folds. SVM is not present and RF/LR are not shown for large sample sizes in panel (**b**) due to prohibitively large computational runtimes.

Examples of the masked U-Net's segmentation predictions on the test set are shown in Figure 8, along with the random forest predictions on the same images. Across all samples, the masked U-Net produces predictions that are more spatially coherent—i.e., neighboring pixels are more correlated in label—than the random forest predictions. The U-Net also notably does not classify urban vegetation as cropland where the random forest does, illustrating the utility of seeing a pixel in its context.

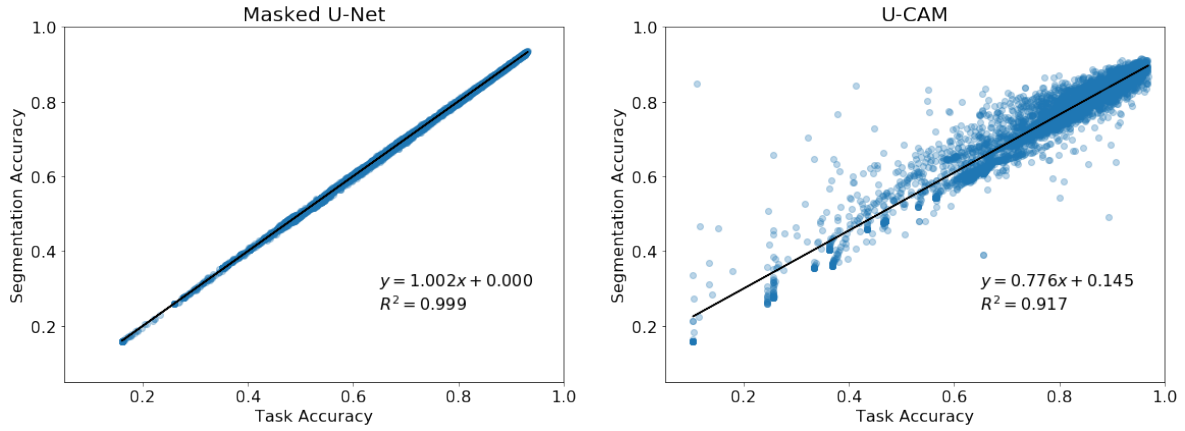


Figure 7. (**a**) Segmentation accuracy and task accuracy under the Masked U-Net model are correlated with an R^2 of 0.999. (**b**) Segmentation accuracy and task accuracy under the U-CAM model are correlated with an R^2 of 0.917. Validation set task accuracy and segmentation accuracy pairs are plotted across all dataset sizes, runs, and epochs, for a total of 7600 points for each method.

Table 2. Test set accuracy, precision, recall, and F1 scores for the masked U-Net and baselines trained on pixel labels at training set size $n = 100$ and $n = 1000$. Bolded numbers display the highest value in each column.

$n = 100$				
Method	**Overall Accuracy**	**Precision**	**Recall**	**F1 Score**
Masked U-Net	$\mathbf{0.833 \pm 0.029}$	$\mathbf{0.855 \pm 0.032}$	$\mathbf{0.826 \pm 0.067}$	$\mathbf{0.839 \pm 0.034}$
Random Forest	0.817 ± 0.011	0.758 ± 0.036	0.814 ± 0.047	0.783 ± 0.012
SVM	0.826 ± 0.006	0.784 ± 0.027	0.789 ± 0.027	0.785 ± 0.008
Logistic Regression	0.807 ± 0.007	0.771 ± 0.029	0.750 ± 0.044	0.759 ± 0.012
$n = 1000$				
Method	**Overall Accuracy**	**Precision**	**Recall**	**F1 Score**
Masked U-Net	$\mathbf{0.875 \pm 0.003}$	$\mathbf{0.886 \pm 0.014}$	$\mathbf{0.876 \pm 0.015}$	$\mathbf{0.881 \pm 0.003}$
Random Forest	0.840 ± 0.002	0.809 ± 0.013	0.795 ± 0.022	0.801 ± 0.006
SVM	0.854 ± 0.002	0.834 ± 0.015	0.799 ± 0.015	0.816 ± 0.006
Logistic Regression	0.810 ± 0.002	0.800 ± 0.010	0.710 ± 0.017	0.752 ± 0.006

5.3. Obtaining Segmentation from Image Classification

Figure 5c shows the training and validation set performance of the U-CAM model across epochs. As loss on the image classification task decreases and accuracy increases, segmentation accuracy increases as well, despite the model never seeing any pixel labels. The correlation between image classification accuracy and segmentation accuracy is 0.91 on the validation set (Figure 7b), indicating that models that perform well on image classification generally perform well on segmentation as well, but there are outliers. This strong but incomplete correspondence between the two tasks suggests that locating the cropland pixels in an image is one way the model can tell whether an image is majority cropland, but it is not the only way. The presence of certain features—for example, a few densely clustered buildings—may be a strong enough signal for the model to classify an image as non-cropland or cropland without looking at the other parts of the image.

Nevertheless, using image classification accuracy on the validation set to select the model for segmentation led us to choose U-CAM models that outperform the baselines and achieve segmentation accuracies exceeding 85% on the test set (Figure 6b). Our baseline machine learning methods are not designed to extract pixel-level information from image labels, so we modified their input data to be individual pixels labeled with the image label. Image labels add significant noise to pixel-level training, and the accuracies of the random forest and logistic regression baselines are 4–6% lower than their counterparts trained on pixel labels. The U-CAM model performs better than the image-level baselines in segmentation accuracy at all dataset sizes, and also performs better than the baselines trained on pixel labels at $n \geq 100$. At $n = 1000$, the U-CAM method achieves a segmentation accuracy of 0.86, compared to random forest at 0.79 and logistic regression at 0.77. Results for precision, recall, and F1-score metrics are shown in Table 3 for $n = 100$ and $n = 1000$; we observe that, while accuracy and precision of the U-CAM model are comparable to those of the masked U-Net, recall is significantly lower.

Figure 8 shows examples of U-CAM segmentation predictions for the test set and the corresponding cropland activation maps extracted from the network. Like the masked U-Net predictions, the U-CAM segmentation is more spatially coherent than the random forest segmentation, which is very noisy due to the many incorrect training labels.

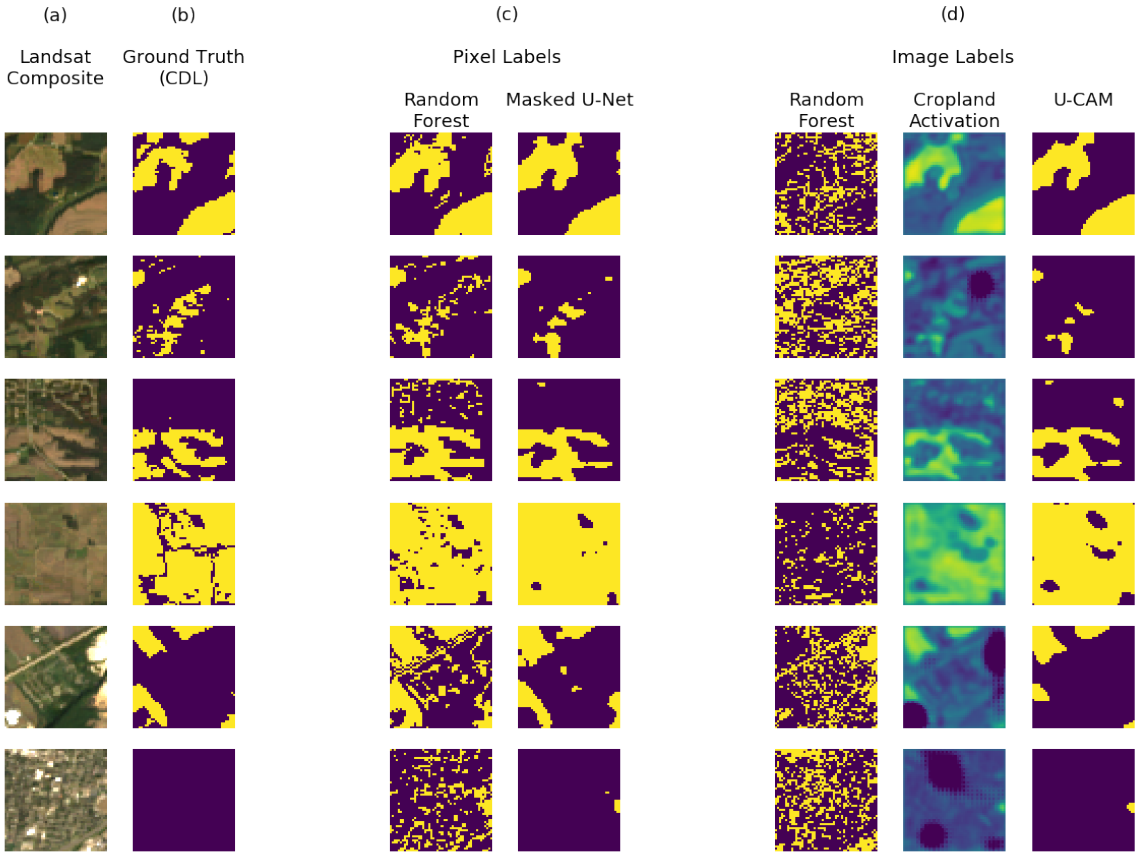


Figure 8. Six examples of (**a**) Landsat composite images, (**b**) their corresponding CDL labels, (**c**) segmentation predictions by models trained on sparse pixel labels, and (**d**) segmentation predictions by models trained on image labels. Yellow pixels are predicted to be cropland, and purple pixels are predicted to be non-cropland. Panel (**d**) shows the class activation map (CAM) extracted from the U-CAM model; yellow corresponds to high cropland activation and dark blue corresponds to low cropland activation. All models were trained on 1000 labels, and predictions are shown for samples in the test set.

Table 3. Test set accuracy, precision, recall, and F1 scores for the U-CAM and baselines trained on image labels at training set size $n = 100$ and $n = 1000$. Bolded numbers display the highest value in each column.

$n = 100$				
Method	**Overall Accuracy**	**Precision**	**Recall**	**F1 Score**
U-CAM	$\mathbf{0.835 \pm 0.010}$	$\mathbf{0.805 \pm 0.035}$	$\mathbf{0.769 \pm 0.051}$	$\mathbf{0.772 \pm 0.020}$
Random Forest	0.767 ± 0.007	0.692 ± 0.015	0.608 ± 0.046	0.646 ± 0.021
Logistic Regression	0.764 ± 0.012	0.821 ± 0.018	0.724 ± 0.050	0.768 ± 0.021
$n = 1000$				
Method	**Overall Accuracy**	**Precision**	**Recall**	**F1 Score**
U-CAM	$\mathbf{0.861 \pm 0.016}$	$\mathbf{0.863 \pm 0.036}$	$\mathbf{0.771 \pm 0.067}$	$\mathbf{0.800 \pm 0.033}$
Random Forest	0.786 ± 0.003	0.706 ± 0.004	0.631 ± 0.020	0.666 ± 0.009
Logistic Regression	0.766 ± 0.006	0.822 ± 0.006	0.725 ± 0.020	0.770 ± 0.009

6. Discussion

6.1. Weakly Supervised Segmentation

The methods assessed in this paper show that CNNs can be trained for segmentation using small datasets comprised of pixel or image labels. The masked U-Net and U-CAM models can achieve segmentation accuracies of over 85% with modest dataset sizes in the hundreds of labels, outperforming commonly used pixel-based methods like logistic regression, SVM, and random forest. These simple modifications allow the advantages of CNNs, namely their ability to account for spatial context and learn nonlinear transformations, to be combined with datasets that are easy and feasible in quantity for domain experts or crowdsourced workers to generate.

The selection of the best U-Net for segmentation using only weak labels requires high performance on the weakly supervising task to correspond to high segmentation performance. We showed that this is true when the position of labeled pixels is random across the training set ($R^2 \approx 1.0$), while the relationship is not as strong when the labeled pixel is always in the center of the image ($R^2 = 0.79$). Under random labeling, the model can perform well on the task either by (1) classifying all pixels in each image correctly or (2) memorizing the locations of the labels in all training tiles and classifying those pixels correctly. The near-perfect correlation between task and segmentation accuracy on the validation set indicates that the U-Net accomplished the former. In contrast, when the center pixel is always the labeled one, the model does not have to correctly classify the other pixels in the image. This indicates that, if one is given a dataset of point labels, a random crop of remote sensing imagery should be extracted around each point, rather than tiles with the label always at a fixed position.

Meanwhile, performance on the image classification task has a correlation with segmentation performance of $R^2 = 0.91$. We hypothesize that this is because the global average pooling layer encourages the U-CAM model to perform classification via segmentation. In other words, the model performs well on image classification if its pixel-level predictions are correct on average. Furthermore, the skip connections of the U-Net enable spatial information from the input image to be kept and used to localize pixel labels. Ultimately, this high correlation makes it possible to pick the best model for segmentation using only image-level labels, an important proxy when there are few or no segmentation labels available.

6.2. Trade-Offs between Label Types

In light of our findings, researchers obtaining ground truth labels for segmentation de novo have their choice of dense labels (e.g., geospatially referenced polygons, densely segmented rasters), point labels (e.g., geospatially referenced points, pixel labels), or image labels. Our results shed some light on the trade-offs involved in choosing the label type. Figure 6 shows that the fully supervised U-Net performs segmentation well at extremely small dataset sizes; given only ten segmented training samples, the U-Net segments cropland at 84% accuracy. In comparison, the masked U-Net achieves a similar mean accuracy and variance after seeing between 100 to 200 pixel labels. This ratio of 10:1 to 20:1 single pixel labels to densely segmented labels holds across the curves in Figure 6, and suggests that pixel-labels are preferable to segmented labels if they are less than 10–20 times as costly to obtain. Here cost should take into account not only compensation for crowdsourced workers or researchers, but also the difficulty of the labeling task, the complexity of designing the annotation instructions for training, and the likelihood that labels will meet the quality standard.

With the U-CAM model, a similar equivalence of 10:1 to 20:1 between image labels and densely segmented labels exists, until the performance of the U-CAM model flattens out after 500 image labels at 0.87. Additional labels beyond 500 do not help the model better localize the precise location of cropland pixels. Therefore, to achieve the highest segmentation accuracies, image labels may need to be augmented with pixel labels or densely segmented labels. One can imagine pre-training on a large number of image labels and fine-tuning on a small number of segmented labels. More research is needed to improve the localization of segmentation predictions transferred from image classification.

6.3. Method Limitations and Future Directions

By training U-Nets on an annual, median composite of the first seven Landsat bands, our work does not leverage the temporal nature of satellite imagery or commonly-used vegetation indices (VIs) like NDVI. Since the timing of plant growth and senescence helps distinguish different types of vegetation, features that capture variation in time should be an improvement over the annual median. Future work can explore segmentation using weakly supervised CNNs or RNNs with temporal features, especially in ways that are label-efficient. As for the use of vegetation indices, nonlinear methods like neural networks and random forests should in theory be able to recover them if given enough data, though adding VIs may still improve performance, especially at small training set sizes. The goal of this study is not to create the best possible cropland map, but to demonstrate that CNNs can perform segmentation of remote sensing imagery with weak labels, which have traditionally been used only to train pixel-based machine learning methods.

While we have shown that deep learning methods can achieve state-of-the-art accuracies on segmentation using weak labels, the application of CNNs to remote sensing tasks still contains trade-offs relative to more established machine learning methods (i.e., our baselines). First, training CNNs on remote sensing datasets and applying them at a large scale currently requires the user to move large quantities of data between GIS platforms (in our case, Google Earth Engine) and deep learning frameworks (TensorFlow, PyTorch, etc.). Further integration of these platforms will alleviate the manual manipulation of geospatial data and go a long way toward enabling the application of deep neural networks in this domain.

Second, deep learning models still suffer from a shortage of tools that enable users to qualitatively understand the relationship between inputs and the model's prediction. In applications where machine predictions feed into human decision-making, this lack of interpretability decreases trust in neural networks and may make them less suitable than highly interpretable models like logistic regression. More visualization tools and theory are needed to improve the transparency of deep learning; in the meantime, performance and interpretability should continue to be viewed as a trade-off when selecting between machine learning algorithms.

7. Conclusions

In this paper, we showed that the U-Net model, designed for end-to-end segmentation, can segment cropland in Landsat composite imagery over the US Midwest using small quantities of weakly supervising labels. The masked U-Net, trained on pixel labels, and the U-CAM model, trained on image labels, achieve segmentation accuracies exceeding 85% on training set sizes in the hundreds of labels. They outperform traditional machine learning baselines trained on the same quantities of labels (above $n = 100$), and show greater spatial coherence in their predictions.

Our work demonstrates that CNNs can be trained to perform accurate segmentation with weak supervision, using ground truth labels that contain less information per label than densely segmented ones but are easier to obtain in large quantities. This enlarges the possibilities of methods that can be used with existing point or image labels, plus future such datasets generated from fieldwork or crowdsourcing. Further work is needed to bridge the gap between the data requirements of state-of-the-art machine learning methods and the data availability in many remote sensing applications, as well as integrating GIS data platforms with deep learning frameworks in order to apply these methods at large scale.

Author Contributions: Conceptualization, S.W. and D.B.L.; methodology, S.W., W.C., and S.M.X.; software, S.W., W.C., and S.M.X.; validation, S.W. and W.C.; formal analysis, S.W. and W.C.; investigation, S.W. and W.C.; resources, D.B.L.; data curation, S.W. and G.A.; writing—original draft preparation, S.W. and W.C.; writing—review and editing, S.W. and D.B.L.; visualization, S.W.; supervision, D.B.L. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Acknowledgments: We thank Nick Guo for providing technical support on the Google Cloud Platform and data storage.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
CAM	Class Activation Map
SVM	Support Vector Machine
CDL	Cropland Data Layer

Appendix A

Appendix A.1. Training, Validation, Test Set Splits

Since satellite readings and cropland labels are highly correlated in space, metrics like accuracy will be inflated if individual samples are split into training, validation, and test sets at random. We therefore split the study region into 64 rectangles geographically, and randomly assigned 50 rectangles to a training and validation set and 14 rectangles to a test set. Within the training and validation set, the 50 rectangles were split into 10 folds of 5 rectangles each. Nine of the ten geographic training, validation, and test set splits are shown in Figure A1.

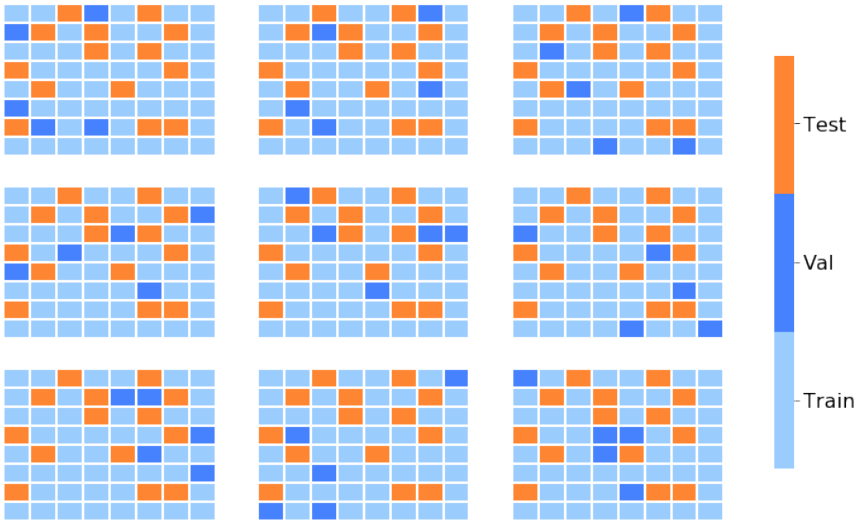


Figure A1. Nine of the ten folds showing training, validation, and test set splits. Folds are defined geographically to reduce performance inflation due to spatial autocorrelation. The first fold is shown in Figure 2b.

Appendix A.2. Baseline Model Descriptions and Implementation Details

- **Logistic regression.** Logistic regression is a commonly used classification method that uses a logistic function to model a binary outcome. The predictors are assumed to have a linear relationship with the logarithm of the outcome odds. Since it can only learn linear decision boundaries between classes, logistic regression performs poorly when class boundaries are highly non-linear but well when they are approximately linear, and can outperform non-linear methods when predictor dimensionality is high relative to number of data points.
- **Support-vector machine (SVM).** SVMs are a class of models capable of performing non-linear classification. They do this by constructing hyperplanes that separate the training set into classes

in high or infinite dimensional space with the largest margins possible. They have been used with success in remote sensing to perform land cover classification and crop mapping [49].

- **Random forest.** Random forests are an ensemble machine learning method in which many decision trees are aggregated to perform classification or regression [50]. They are used frequently in the field of remote sensing to perform land cover classification and crop mapping [51,52], and have been shown to yield higher accuracies than maximum likelihood classifiers, support vector machines, and other methods for crop mapping [49,53,54].

We used Python's `scikit-learn` [55] implementation of logistic regression with penalty, support vector machines, and random forest classifiers. At each dataset size, we performed a 10-fold cross-validation to find the best hyperparameters. The hyperparameters that yielded the highest mean validation accuracy for each method and dataset size are shown in Tables A1–A3.

All baseline models were run on a Google Compute Engine virtual machine with 4 Intel Broadwell vCPUs and 52GB RAM, running Ubuntu 16.04. We used Python 3.7.3 and `scikit-learn` 0.21.3.

Table A1. Hyperparameters for penalized logistic regression yielding the highest validation accuracy across dataset sizes.

Dataset Size	Hyperparameter	
	Penalty	λ
10	L_1	10^{-5}
20	L_2	10^{-4}
50	L_2	10^{-4}
100	L_2	10^{-4}
200	L_2	10^{-4}
500	L_1	10^{-2}
1000	L_2	10^{-3}
2000	L_2	10^{-2}
5000	L_2	10^{-2}
10,000	L_1	10^{0}
20,000	L_1	10^{0}
50,000	L_2	10^{-1}
100,000	L_1	10^{1}

Table A2. Hyperparameters for SVM yielding the highest validation accuracy across dataset sizes.

Dataset Size	Hyperparameter		
	Kernel	Penalty C	Kernel Coefficient γ
10	Linear	1000	10^{-4}
20	RBF	1000	1.0
50	RBF	1000	1.0
100	RBF	1000	1.0
200	RBF	100	10.0
500	RBF	1000	10.0
1000	RBF	1000	10.0
2000	RBF	1000	10.0
5000	RBF	1000	10.0
10,000	RBF	1000	10.0
20,000	—	—	—
50,000	—	—	—
100,000	—	—	—

Table A3. Hyperparameters for random forest yielding the highest validation accuracy across dataset sizes. Note `sqrt` is the default number of features used per tree, which for 7 features rounds down to 2.

Dataset Size	Hyperparameter			Max Features
	# Trees	Min Samples Split	Min Samples Leaf	
10	10	2	1	4
20	500	5	1	`sqrt`
50	100	10	1	7
100	500	10	1	`sqrt`
200	50	10	1	`sqrt`
500	500	2	2	4
1000	500	2	2	`sqrt`
2000	500	2	1	4
5000	500	10	1	4
10,000	500	10	2	4
20,000	500	10	5	4
50,000	500	10	2	4
100,000	500	10	2	4

Appendix A.3. U-Net Implementation and Hyperparameter Search Details

Neural networks were trained on the same Google Compute VM as the baselines, using Nvidia K80 GPUs. The densely supervised and masked U-Nets were implemented in PyTorch 1.2.0, and the U-CAM model was implemented in TensorFlow 1.4.1. We used an Adam optimizer with learning rate 10^{-3}, $\beta_1 = 0.9$, $\beta_2 = 0.999$, and batch size of 32. Batch normalization was used after each convolutional layer with batch norm momentum of 0.9. The U-Nets were trained for 20 epochs to allow convergence on training set sizes $n \geq 1000$, and trained for 200 epochs at $n = 10$, 100 epochs at $n = 20$ and $n = 50$, and 50 epochs at $100 \leq n < 1000$.

To curb overfitting, we added an L2 regularization term to our cross entropy losses so that our final training loss was

$$J(\theta, \lambda) = \mathcal{L}(\theta) + \lambda ||\theta||_2^2 \tag{A1}$$

Note that only model weights were penalized; biases were not.

Optimal U-Net hyperparameters were found via grid search and are shown in Table A4. The hyperparameters we searched over are the number of encoding and decoding blocks l, the number of filters f, and regularization strength λ.

Table A4. Hyperparameters for U-Net yielding the highest validation accuracy across dataset sizes.

Dataset Size	Hyperparameter			Learning Rate
	Layers	Initial Filters	L_2 Regularization	
10	4	64	10^{-3}	10^{-3}
20	4	64	10^{-3}	10^{-3}
50	4	64	10^{-3}	10^{-3}
100	4	64	10^{-3}	10^{-3}
200	4	64	10^{-3}	10^{-3}
500	4	64	10^{-3}	10^{-3}
1000	4	64	10^{-3}	10^{-3}
2000	4	64	10^{-3}	10^{-3}
5000	4	64	10^{-4}	10^{-3}
10,000	4	64	10^{-4}	10^{-3}
20,000	4	64	10^{-4}	10^{-3}
50,000	4	64	10^{-4}	10^{-3}
100,000	4	64	10^{-4}	10^{-3}

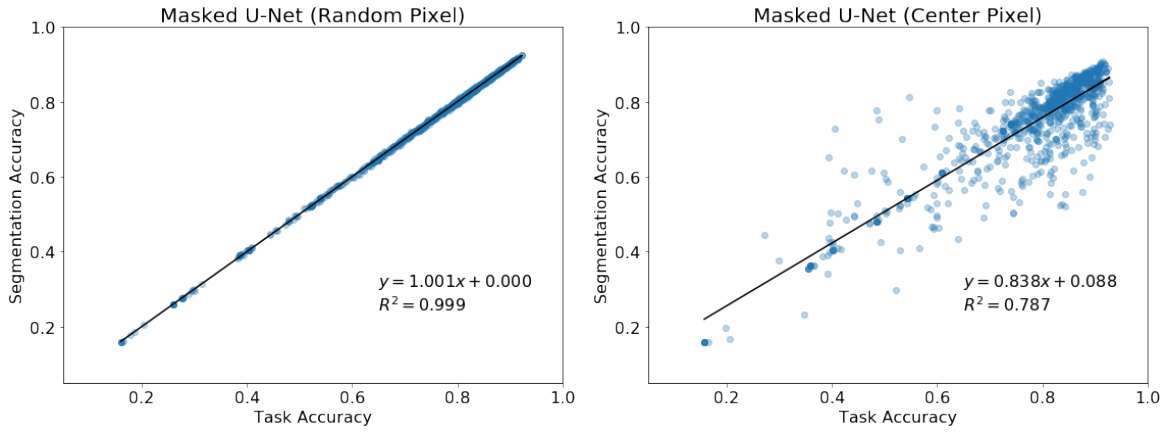
An ℓ-block U-Net has ℓ down-convolutional blocks and ℓ up-convolutional blocks, and an f-filter U-Net starts with f filters in the first convolutional block and doubles the number of filters in each subsequent block. We searched exhaustively over $\ell \in \{3,4,5\}$, $f \in \{16,32,64,128,256\}$, and regularization strength $\lambda \in \{10^{-5},10^{-4},10^{-3},10^{-2},10^{-1},10^{0}\}$; we also tried some models with learning rate $\in \{1\times10^{-4},3\times10^{-4},1\times10^{-3},3\times10^{-3},1\times10^{-2}\}$ and batch size $\in \{8,16,32,64\}$. The network weights were initialized using Xavier initialization [56]. For each dataset size, we chose the hyperparameters that yielded the highest validation set accuracy across the ten folds for the U-CAM model, as the U-CAM model was more sensitive to hyperparameters while the end-to-end U-Nets were more robust.

We observed that model depth and width did not significantly affect end-to-end segmentation, suggesting that the task of mapping cropland using a Landsat composite is simple enough to be performed well with the smallest of these models (3 blocks, 16 filters in the first block). For transferring image classification to segmentation, however, deeper 4 or 5 block U-Nets with 32 to 64 starting filters achieved the highest segmentation accuracy. Optimal L_2 regularization strength varied from 10^{-4} to 10^{-3} depending on training size.

Appendix A.4. Random vs. Deterministic Masking

Randomness in the position of the labeled pixel was important for achieving high correlation between validation task accuracy (single pixel classification) and segmentation accuracy.

Figure A2 shows the correlation between validation set task accuracy and segmentation accuracy for the two types of labels. While random label position achieves an R^2 close to 1.0, a label position that is always in the center of the tile achieves $R^2 = 0.787$. A lower R^2 means there is less of a guarantee that a model that classifies the center pixel correctly also segments an entire tile correctly, making it more difficult to select a good model during cross-validation.



(**a**) (**b**)

Figure A2. Scatter plots and corresponding least squares fit between validation set task accuracy and segmentation accuracy for (**a**) randomly located pixel labels and (**b**) pixel labels always at the center of the tile for the masked U-Net model. Points are shown for training set sizes of $n \in \{100, 1000, 10{,}000\}$ across 10 runs of $[50, 20, 20]$ epochs, respectively.

Appendix A.5. Baseline Model Results

Figure 6 summarizes the performance of baseline and oracle methods across a wide range of training set sizes. Hyperparameters of the baseline models were tuned for each training set size and are listed in Tables A1–A3.

Of the three pixel-based baseline methods (Figure 6a), SVM achieved the highest classification accuracies consistently across different values of n (85.4% mean accuracy at $n = 1000$), while random forest accuracies were close behind (84.0% mean accuracy at $n = 1000$). Accuracies for both methods increase with training set size up to the largest size of $n = 10^5$, though the increase slows at larger n. In contrast, logistic regression performs significantly worse than the nonlinear baselines and reaches its highest accuracy of 81% by $n = 2000$. The downside of SVM is that its $O(n^2)$ computational complexity makes it prohibitively time-consuming to train, so we do not report SVM accuracies at $n > 10^4$.

Due to high SVM runtime, we only evaluated logistic regression and random forest baselines for image-level labels, where 1 image label corresponds to 2500 pixels. Memory constraints also limited these two methods to training set sizes of up to 20,000 images (50 million pixels) and 2000 images (5 million pixels) respectively. Figure 6b shows their performance as the number of image labels increases. While random forest accuracies are on average worse than logistic regression when the training size is very small, the forest begins to capture nonlinearities and surpass logistic regression when shown 100 or more images.

References

1. Hansen, M.C.; Potapov, P.V.; Moore, R.; Hancher, M.; Turubanova, S.A.; Tyukavina, A.; Thau, D.; Stehman, S.V.; Goetz, S.J.; Loveland, T.R.; et al. High-Resolution Global Maps of 21st-Century Forest Cover Change. *Science* **2013**, *342*, 850–853. [CrossRef] [PubMed]
2. Esch, T.; Heldens, W.; Hirner, A.; Keil, M.; Marconcini, M.; Roth, A.; Zeidler, J.; Dech, S.; Strano, E. Breaking new ground in mapping human settlements from space – The Global Urban Footprint. *ISPRS J. Photogramm. Remote Sens.* **2017**, *134*, 30–42. [CrossRef]
3. Bindschadler, R. Monitoring ice sheet behavior from space. *Rev. Geophys.* **1998**, *36*, 79–104. [CrossRef]
4. Amit, S.N.K.B.; Shiraishi, S.; Inoshita, T.; Aoki, Y. Analysis of satellite images for disaster detection. In Proceedings of the 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 10–15 July 2016; pp. 5189–5192. [CrossRef]
5. Farabet, C.; Couprie, C.; Najman, L.; LeCun, Y. Learning Hierarchical Features for Scene Labeling. *IEEE Trans. Pattern Anal. Mach. Intell.* **2013**, *35*, 1915–1929. [CrossRef]
6. Noh, H.; Hong, S.; Han, B. Learning Deconvolution Network for Semantic Segmentation. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Las Condes, Chile, 11–18 December 2015; pp. 1520–1528. [CrossRef]
7. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*; Navab, N., Hornegger, J., Wells, W.M., Frangi, A.F., Eds.; Springer International Publishing: Cham, Switzerland, 2015; pp. 234–241.
8. Chen, L.; Zhu, Y.; Papandreou, G.; Schroff, F.; Adam, H. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In *Computer Vision—ECCV 2018*; Ferrari, V., Hebert, M., Sminchisescu, C., Weiss, Y., Eds.; Springer International Publishing: Cham, Switzerland, 2018; pp. 833–851.
9. Chen, L.; Papandreou, G.; Schroff, F.; Adam, H. Rethinking Atrous Convolution for Semantic Image Segmentation. *arXiv* **2017**, arXiv:1706.05587.
10. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The Pascal Visual Object Classes (VOC) Challenge. *Int. J. Comput. Vis.* **2010**, *88*, 303–338. [CrossRef]
11. Lin, T.Y.; Maire, M.; Belongie, S.; Hays, J.; Perona, P.; Ramanan, D.; Dollár, P.; Zitnick, C.L. Microsoft COCO: Common Objects in Context. In *Computer Vision—ECCV 2014*; Fleet, D., Pajdla, T., Schiele, B., Tuytelaars, T., Eds.; Springer International Publishing: Cham, Switzerland, 2014; pp. 740–755.
12. Cordts, M.; Omran, M.; Ramos, S.; Rehfeld, T.; Enzweiler, M.; Benenson, R.; Franke, U.; Roth, S.; Schiele, B. The Cityscapes Dataset for Semantic Urban Scene Understanding. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.
13. Neuhold, G.; Ollmann, T.; Bulo, S.; Kontschieder, P. The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes. In Proceedings of the 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017; IEEE Computer Society: Los Alamitos, CA, USA, 2017; pp. 5000–5009. [CrossRef]

14. Goodfellow, I.; Bengio, Y.; Courville, A. *Deep Learning*; MIT Press: Cambridge, MA, USA, 2016.
15. Garcia-Garcia, A.; Orts-Escolano, S.; Oprea, S.; Villena-Martinez, V.; Martinez-Gonzalez, P.; Garcia-Rodriguez, J. A survey on deep learning techniques for image and video semantic segmentation. *Appl. Soft Comput.* **2018**, *70*, 41–65. [CrossRef]
16. Jean, N.; Burke, M.; Xie, M.; Davis, W.M.; Lobell, D.B.; Ermon, S. Combining satellite imagery and machine learning to predict poverty. *Science* **2016**, *353*, 790–794. [CrossRef]
17. Zhou, Z.H. A brief introduction to weakly supervised learning. *Natl. Sci. Rev.* **2017**, *5*, 44–53. [CrossRef]
18. Jin, Z.; Azzari, G.; You, C.; Tommaso, S.D.; Aston, S.; Burke, M.; Lobell, D.B. Smallholder maize area and yield mapping at national scales with Google Earth Engine. *Remote Sens. Environ.* **2019**, *228*, 115–128. [CrossRef]
19. Xiong, J.; Thenkabail, P.S.; Gumma, M.K.; Teluguntla, P.; Poehnelt, J.; Congalton, R.G.; Yadav, K.; Thau, D. Automated cropland mapping of continental Africa using Google Earth Engine cloud computing. *ISPRS J. Photogramm. Remote Sens.* **2017**, *126*, 225–244. [CrossRef]
20. USDA National Agricultural Statistics Service Cropland Data Layer. Published Crop-Specific Data Layer. 2017. Available online: https://nassgeodata.gmu.edu/CropScape/ (accessed on 1 March 2018).
21. Zhou, B.; Lapedriza, A.; Xiao, J.; Torralba, A.; Oliva, A. Learning Deep Features for Scene Recognition using Places Database. In *Advances in Neural Information Processing Systems 27*; Ghahramani, Z., Welling, M., Cortes, C., Lawrence, N.D., Weinberger, K.Q., Eds.; Curran Associates, Inc.: Dutchess County, NY, USA, 2014; pp. 487–495.
22. Rußwurm, M.; Körner, M. Multi-Temporal Land Cover Classification with Sequential Recurrent Encoders. *ISPRS Int. J. Geo-Inf.* **2018**, *7*. [CrossRef]
23. Rustowicz, R.; Cheong, R.; Wang, L.; Ermon, S.; Burke, M.; Lobell, D.B. Semantic Segmentation of Crop Type in Africa: A Novel Dataset and Analysis of Deep Learning Methods. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops, Long Beach, CA, USA, 16–20 June 2019.
24. Jeppesen, J.H.; Jacobsen, R.H.; Inceoglu, F.; Toftegaard, T.S. A cloud detection algorithm for satellite imagery based on deep learning. *Remote Sens. Environ.* **2019**, *229*, 247–259. [CrossRef]
25. Drönner, J.; Korfhage, N.; Egli, S.; Mühling, M.; Thies, B.; Bendix, J.; Freisleben, B.; Seeger, B. Fast Cloud Segmentation Using Convolutional Neural Networks. *Remote Sens.* **2018**, *10*, 1782. [CrossRef]
26. Zhang, A.; Liu, X.; Gros, A.; Tiecke, T. Building Detection from Satellite Images on a Global Scale. *arXiv* **2017**, arXiv:1707.08952.
27. Sun, Y.; Zhang, X.; Zhao, X.; Xin, Q. Extracting Building Boundaries from High Resolution Optical Images and LiDAR Data by Integrating the Convolutional Neural Network and the Active Contour Model. *Remote Sens.* **2018**, *10*, 1459. [CrossRef]
28. Yi, Y.; Zhang, Z.; Zhang, W.; Zhang, C.; Li, W.; Zhao, T. Semantic Segmentation of Urban Buildings from VHR Remote Sensing Imagery Using a Deep Convolutional Neural Network. *Remote Sens.* **2019**, *11*, 1774. [CrossRef]
29. Diakogiannis, F.I.; Waldner, F.; Caccetta, P.; Wu, C. ResUNet-a: A deep learning framework for semantic segmentation of remotely sensed data. *arXiv* **2019**, arXiv:1904.00592.
30. Hwang, J.I.; Jung, H.S. Automatic Ship Detection Using the Artificial Neural Network and Support Vector Machine from X-Band SAR Satellite Images. *Remote Sens.* **2018**, *10*, 1799. [CrossRef]
31. Henry, C.; Azimi, S.M.; Merkle, N. Road Segmentation in SAR Satellite Images With Deep Fully Convolutional Neural Networks. *IEEE Geosci. Remote Sens. Lett.* **2018**, *15*, 1867–1871. [CrossRef]
32. Panboonyuen, T.; Jitkajornwanich, K.; Lawawirojwong, S.; Srestasathiern, P.; Vateekul, P. Road Segmentation of Remotely-Sensed Images Using Deep Convolutional Neural Networks with Landscape Metrics and Conditional Random Fields. *Remote Sens.* **2017**, *9*, 680. [CrossRef]
33. ISPRS Test Project on Urban Classification, 3D Building Reconstruction and Semantic Labeling. 2018. Available online: http://www2.isprs.org/commissions/comm3/wg4/tests.html (accessed on 30 April 2019).
34. Maggiori, E.; Tarabalka, Y.; Charpiat, G.; Alliez, P. Can Semantic Labeling Methods Generalize to Any City? The Inria Aerial Image Labeling Benchmark. In Proceedings of the IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Fort Worth, TX, USA, 23–28 July 2017.

35. Dstl Satellite Imagery Feature Detection. 2017. Available online: https://www.kaggle.com/c/dstl-satellite-imagery-feature-detection (accessed on 30 April 2019).
36. CrowdAnalytix Agricultural Crop Cover Classification Challenge. 2018. Available online: https://www.crowdanalytix.com/contests/agricultural-crop-cover-classification-challenge (accessed on 30 April 2019).
37. Demir, I.; Koperski, K.; Lindenbaum, D.; Pang, G.; Huang, J.; Basu, S.; Hughes, F.; Tuia, D.; Raska, R. DeepGlobe 2018: A Challenge to Parse the Earth through Satellite Images. In Proceedings of the 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), Salt Lake City, UT, USA, 19–21 June 2018; pp. 172–17209. [CrossRef]
38. Sumbul, G.; Charfuelan, M.; Demir, B.; Markl, V. Bigearthnet: A Large-Scale Benchmark Archive for Remote Sensing Image Understanding. In Proceedings of the IEEE International Conference on Geoscience and Remote Sensing Symposium, Yokohama, Japan, 28 July–2 August 2019; pp. 5901–5904.
39. Kaiser, P.; Wegner, J.D.; Lucchi, A.; Jaggi, M.; Hofmann, T.; Schindler, K. Learning Aerial Image Segmentation From Online Maps. *IEEE Trans. Geosci. Remote Sens.* **2017**, *55*, 6054–6068. [CrossRef]
40. Kemker, R.; Salvaggio, C.; Kanan, C. Algorithms for semantic segmentation of multispectral remote sensing imagery using deep learning. *ISPRS J. Photogramm. Remote Sens.* **2018**, *145*, 60–77. [CrossRef]
41. Kang, X.; Zhuo, B.; Duan, P. Semi-supervised deep learning for hyperspectral image classification. *Remote Sens. Lett.* **2019**, *10*, 353–362. [CrossRef]
42. Hong, S.; Noh, H.; Han, B. Decoupled Deep Neural Network for Semi-supervised Semantic Segmentation. In Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, QC, Canada, 7–12 December 2015; pp. 1495–1503.
43. Pinheiro, P.O.; Collobert, R. From image-level to pixel-level labeling with Convolutional Networks. In Proceedings of the 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Boston, MA, USA, 7–12 June 2015; pp. 1713–1721. [CrossRef]
44. Teluguntla, P.; Thenkabail, P.S.; Oliphant, A.; Xiong, J.; Gumma, M.K.; Congalton, R.G.; Yadav, K.; Huete, A. A 30-m landsat-derived cropland extent product of Australia and China using random forest machine learning algorithm on Google Earth Engine cloud computing platform. *ISPRS J. Photogramm. Remote Sens.* **2018**, *144*, 325–340. [CrossRef]
45. Xiong, J.; Thenkabail, P.S.; Tilton, J.C.; Gumma, M.K.; Teluguntla, P.; Oliphant, A.; Congalton, R.G.; Yadav, K.; Gorelick, N. Nominal 30-m Cropland Extent Map of Continental Africa by Integrating Pixel-Based and Object-Based Algorithms Using Sentinel-2 and Landsat-8 Data on Google Earth Engine. *Remote Sens.* **2017**, *9*, 1065. [CrossRef]
46. Belgiu, M.; Csillik, O. Sentinel-2 cropland mapping using pixel-based and object-based time-weighted dynamic time warping analysis. *Remote Sens. Environ.* **2018**, *204*, 509–523. [CrossRef]
47. Roy, D.P.; Wulder, M.A.; Loveland, T.R.; Woodcock, C.E.; Allen, R.G.; Anderson, M.C.; Helder, D.; Irons, J.R.; Johnson, D.M.; Kennedy, R.; et al. Landsat-8: Science and product vision for terrestrial global change research. *Remote Sens. Environ.* **2014**, *145*, 154–172. [CrossRef]
48. Whitcraft, A.K.; Vermote, E.F.; Becker-Reshef, I.; Justice, C.O. Cloud cover throughout the agricultural growing season: Impacts on passive optical earth observations. *Remote Sens. Environ.* **2015**, *156*, 438–447. [CrossRef]
49. Inglada, J.; Arias, M.; Tardy, B.; Hagolle, O.; Valero, S.; Morin, D.; Dedieu, G.; Sepulcre, G.; Bontemps, S.; Defourny, P.; et al. Assessment of an Operational System for Crop Type Map Production Using High Temporal and Spatial Resolution Satellite Optical Imagery. *Remote Sens.* **2015**, *7*, 12356–12379. [CrossRef]
50. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
51. Gislason, P.O.; Benediktsson, J.A.; Sveinsson, J.R. Random Forests for land cover classification. *Pattern Recognit. Lett.* **2006**, *27*, 294–300. [CrossRef]
52. Azzari, G.; Lobell, D. Landsat-based classification in the cloud: An opportunity for a paradigm shift in land cover monitoring. *Remote Sens. Environ.* **2017**, *202*, 64–74. [CrossRef]
53. Ok, A.O.; Akar, O.; Gungor, O. Evaluation of random forest method for agricultural crop classification. *Eur. J. Remote Sens.* **2012**, *45*, 421–432. [CrossRef]
54. Gomez, C.; White, J.C.; Wulder, M.A. Optical remotely sensed time series data for land cover classification: A review. *ISPRS J. Photogramm. Remote Sens.* **2016**, *116*, 55–72. [CrossRef]

55. Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; et al. Scikit-learn: Machine Learning in Python. *J. Mach. Learn. Res.* **2011**, *12*, 2825–2830.
56. Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. In Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, Sardinia, Italy, 13–15 May 2010; pp. 249–256.