Mining self-similarity: Label super-resolution with epitomic representations
ECCV 2020
Supplementary material

Nikolay Malkin[1], Anthony Ortiz[2], and Nebojsa Jojic[3]

[1] Yale University, New Haven, CT 06520, USA
kolya.malkin@yale.edu
[2] Microsoft AI for Good Research Lab, Redmond, WA 98052, USA
[3] Microsoft Research, Redmond, WA 98052, USA
{anthony.ortiz,jojic}@microsoft.com

Table of Contents

1 Code

We provide sample code at `https://github.com/anthonymlortiz/epitomes_lsr`. The folder `data/` includes an example of a $2\times$ downsampled 2×2km tile of aerial imagery and the associated high-res and low-res labels. Run the notebook `train_epitome.ipynb` to visualize the iterations of epitome training on this area. Run `self_epitomic_sr.ipynb` to visualize single-image label super-resolution. (The corresponding static HTML files can also be used for viewing. Some outputs of the first notebook are shown in Fig. 1.)

The training code is short and clear, and we feel that it can serve in lieu of pseudocode. The reader can immediately run it, perhaps even with their own data.

2 The vanishing posterior problem in EM and SGD epitome learning

As derived in [10], the original EM algorithm for epitome learning optimizes the log likelihood

$$\sum_t \log \sum_{s=1}^{S} p(x^t|s)p(s) \tag{1}$$

by iterating two steps. First, the image patches x^t are mapped to the epitome using the posterior $p(s|x) \propto p(x|s)p(s)$, with

$$\log p(s_1, s_2|x) = \text{const} + \sum_{i,j,k} \frac{-1}{2}\left(\frac{x_{i,j,k}^2}{\boldsymbol{\sigma}^2_{s_1+i,s_2+j,k}} - \frac{2x_{i,j,k}\boldsymbol{\mu}_{s_1+i,s_2+j,k}}{\boldsymbol{\sigma}^2_{s_1+i,s_2+j,k}} + \frac{\boldsymbol{\mu}^2_{s_1+i,s_2+j,k}}{\boldsymbol{\sigma}^2_{s_1+i,s_2+j,k}} + \log \boldsymbol{\sigma}^2_{s_1+i,s_2+j,k} \right) + \log p(s_1, s_2). \tag{2}$$

Thus, the E step is efficiently performed with convolutions, creating the posterior $p(s|x)$ used in the M step which re-estimates epitome parameters by averaging the pixels from the patches based on their locations and posterior mapping. For example, the mean epitome is re-estimated as

$$\mu_{m,n,k} = \frac{\sum_t \sum_{(s_1,s_2)} x^t_{m-s_1,n-s_2,k} p(s_1, s_2|x^t)}{\sum_t \sum_{s_1,s_2} p(s_1, s_2|x^t)} \tag{3}$$

where inner summations over s_1, s_2 in numerator and denominator are performed over windows W_{s_1,s_2} containing the pixel m, n: for a $K \times K$ patch x^t, (s_1, s_2) goes from $(m - K + 1, n - K + 1)$ to (m, n). We readily recognize that both numerator and denominator consist of convolutions of the $K \times K$ patch x^t with the posterior $N \times N$ posterior map $p(s_1, s_2|x^t)$ over an $N \times N$ epitome.

The main numerical difficulty with epitome training is that the posterior map can be zero in many places, as the consequence of a large dynamic range of the values computed in Eq. 2, especially as the variances contract during learning. Upon exponentiation to obtain $p(s|x)$, some locations in the epitome may get their probability clipped to zero. If all data patches avoid certain pixels in the epitome, then those locations stop being updated. Note that this is not a local minimum problem, but purely a numerical precision issue. Avoiding division by zero by adding a small constant to the numerator and the denominator in Eq. 3 and subtracting the maximum in Eq. 2 before exponentiation do nothing to remedy this problem: The areas of the epitome that die off simply get filled with the mean of all pixels seen in training. In practice EM training usually has to be carefully coaxed out of these situations, for example by training first with larger patches and later with smaller ones. The larger patches hit more pixels, making it less likely that some epitome locations will never be part of the component s for which at least one data point x^t shows a measurable posterior

after exponentiation. For a regular mixture model the problem can be remedied by keeping track of the highest value of the posterior $p(s|x^t)$ over all data points. In epitomes, because of parameter sharing, this is not trivial. In [11], the authors introduce a solution which evaluates different epitome parts with different levels of precision. The number of precision levels is finite, and the complexity of the learning algorithm scales linearly with the number of levels, which makes the approach impractical for very large epitomes where we could have very large dynamic ranges in the posterior.

The numerical difficulties arising from vanishing posteriors persist if the model is trained by stochastic gradient descent on the parameters of the model. For a mixture model $p(x) = \sum_s p(x|s)p(s)$, with the mixture components $p(x|s) = p(x;\theta_s)$ and the prior $p(s) \propto e^{\pi_s}$, the posterior factors into the gradient of the log-likelihood:

$$\frac{d\log p(x;\theta,\pi)}{d\theta_{s_1,s_2}} = p(s_1,s_2|x;\theta,\pi)\frac{d\log p(x;\theta)}{d\theta_{s_1,s_2}},$$
$$\frac{d\log p(x;\theta,\pi)}{d\pi_{s_1,s_2}} = p(s_1,s_2|x;\theta,\pi).$$

In the situation of epitomes, some θ_s are shared among different s. The gradients with respect to parameters at a given position vanish if all posteriors in a window around the position vanish. Furthermore, batch training and the posterior sampling described below can make it even more likely to loose parts of the epitome during training. As described in Section 3 of the main text, we use a location promotion strategy to solve this problem.

3 Large scale epitome training details

3.1 Training details and parameters

All of our epitome models were implemented using the PyTorch package [14] for efficient computation on the GPU; we also have efficient Matlab implementations for CPU. The epitomes are trained on variable-size patches, 11×11 to 31×31. For patches of size w, we smooth the probabilities $p(x^t|s)$ by temperature $(w/11)^2$. The means are initialized randomly, distributed as $0.5 + \text{unif}(0, 0.1)$. For stability, the variances are parametrized by their inverses $1/\sigma^2$, initialized at 10 and clipped between 1 and 100, and the priors are parametrized in the log domain, initialized at 0 and clipped between -4 and 4. They were trained for 50000 iterations with a batch size of 256 (aerial imagery) or 30000 iterations with a batch size of 64 (pathology imagery), using the Adam optimizer [12] with initial learning rate 0.003.

For the location promotion mechanism in training, we use a threshold of $10^{-8} \cdot$ (batch size). For the 499×499 aerial imagery epitomes, this amounts to $c \approx 0.64$, and for the 299×299 pathology epitomes, $c \approx 0.057$. The counter reset threshold is $\delta = 0.05$: after 0.95 of the epitome has hit its threshold, the counters are zeroed.

3.2 Evaluation by sampling

Here we explain the high-res label embedding and reconstruction method mentioned in Section 5.1. For a given patch x with high-res labels y, we compute the posterior distribution of epitome positions $p(s|x)$. To embed the labels y – that is, to add them, appropriately weighted, to the the counts that give $p(\ell|m,n)$, it would be necessary to convolve the 11×11 patch of *labels* with the posterior. It is more efficient, and equivalent in expectation, to sample several locations from the posterior distribution and to add the labels y to the counts surrounding those locations. We use 16 samples in our experiments. (Because the posterior distributions tend to be peaky, this does not affect the results greatly.) This also explains why there are unmapped areas in the middle column of Fig. 5: the eight epitomes were trained individually, but when they were combined as components of a uniform mixture, those areas were captured by the other epitomes and were never sampled. Similarly, in reconstruction, for a given patch x, we sample several positions s_n^* from the posterior distribution and sum the labels in windows around the s_n^* to form the output predictions for x.

3.3 Execution time

In this section we report the execution times for both U-Nets and epitome algorithms used in our land cover experiments. The EM algorithm for label super-resolution, which consists of iterative matrix multiplications and normalizations, runs in not more than a few seconds, and the main computational cost is incurred in training and evaluation.

The training of one set of four land cover epitomes and of a U-Net both take about 12 hours in our implementation. Note that the two models were implemented in different neural net toolkits: the U-Net implementation in the CNTK toolkit by the authors of [13,16] is two years old and ran on a different version of the CUDA library than our implementation of epitomes in Torch.

The average evaluation times per pixel with different methods are reported in Table 1. For the epitomes, the execution time does not include the label embedding (which can be done incrementally at training time) or the (fast) label super-resolution inference algorithm. On the other hand, for the self-epitomic LSR results, we sampled 5% of all 7×7 patches in the input tile for the low-resolution label embedding; this computation forms the bulk of the computation time, since the LSR algorithm immediately produces the target high-resolution labels without the need to reevaluate the posterior mappings.

We see a number of optimizations to the epitome algorithms that are likely to speed up the evaluation. For example, some terms in the expression for the posterior are independent of the input patch and can be precomputed. In addition, our epitome code does not use the optimized multithreaded data loading methods that appear in the U-Net implementation.

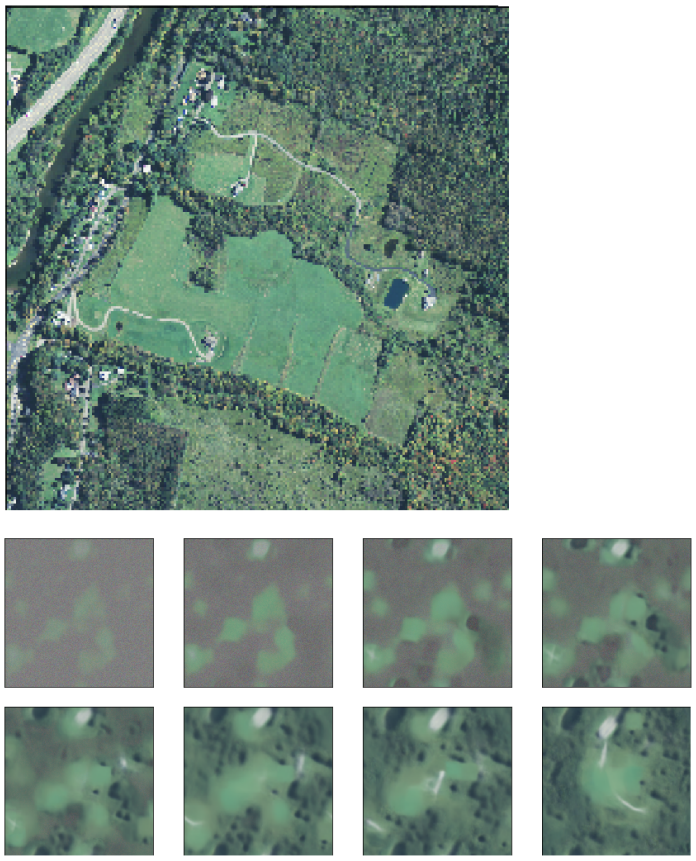

Fig. 1. The 1024×1024 image from our sample code (*above*) and 299×299 epitome $\boldsymbol{\mu}$ parameters at selected iterations of epitome training (after $32, 64, 128, \ldots, 4096$ batches of 256 patches), shown at the same scale (*below*)

Table 1. Execution times (per pixel) of various algorithms on the land cover data.

Method	parameters	time / thousand pixels
U-Net	256×256 input	3ms
All-tile epitomic LSR	11×11 windows, one sample	53ms
All-tile epitomic LSR	31×31 windows, one sample	29ms
Self-epitomic LSR	128×128 input	7ms
Self-epitomic LSR	256×256 input	8ms
Self-epitomic LSR	512×512 input	25ms
Self-epitomic LSR	1024×1024 input	94ms

4 On domain transfer: Comparison of the posterior distributions for land cover regions

Fig. 2 shows the eight components of the land cover epitome (Fig. 5 in the main text) and the posterior distributions in the **South** and **North** regions. Observe that the **South** data is more uniformly mapped. Also, notice the different distribution in forested areas: for example, the light-green forests are rare in **North**. The change in data distribution is in this way easily detectable and explain why U-Nets do not transfer from the **South** to the **North**. Our epitome training is based on the diversification criterion targeting only the distribution over the input patches x^t, but favoring the worst modeled patches. The forests of the **North** where U-Nets make errors can actually be found in the South, too, but they are just more rare. The diversification training learns these patterns, too, and the epitomic classification is not confused by these patterns in the North, resulting in 20% increase in accuracy (see the top of Table 1 in the main text).

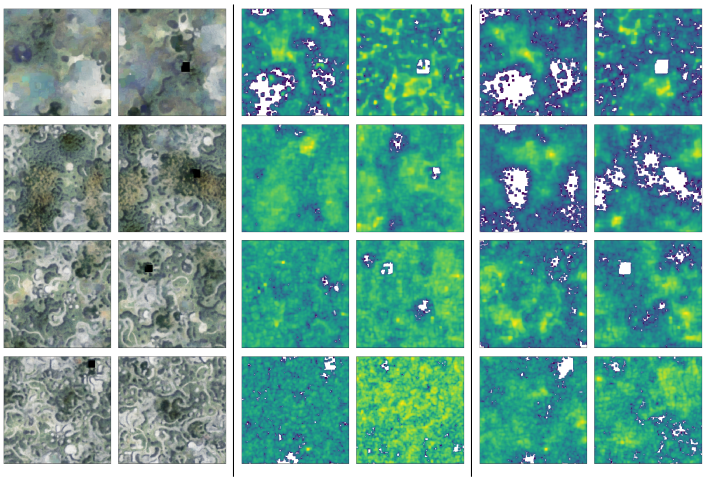

Fig. 2. Epitomes (total area $2 \cdot 10^6$ pixels) trained on imagery from the **South** region (*left*), the log posterior over positions for 31×31 patches in the **South** region (*middle*), the log posterior over positions for 31×31 patches in the **North** region (*right*)

5 Label super-resolution

In Fig. 5 we provide the standard National Land Cover Database (NLCD) class descriptions [8] and color legend in Fig. 5. We use the same colors in all NLCD visualizations here and in the main text. In Table 2 we also provide the $p(\ell|c)$ statistics from [13] which we used in our experiments.

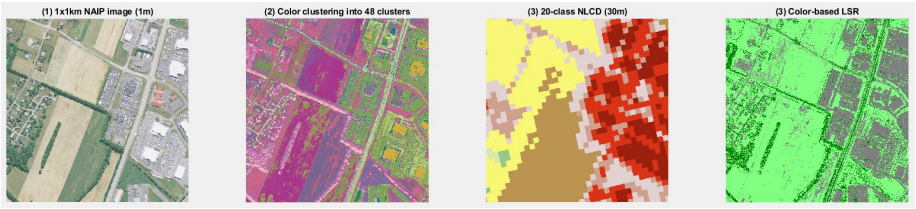

Fig. 3. Color LSR

In Fig. 4 we show a few more results of the single-tile LSR technique for visual inspection of both accurate and inaccurate predictions. But first, as promised in the main text, we address the possibility of performing LSR simply using pixel colors.

As discussed in the main text, our label super-resolution technique iterates equations

$$q_{\ell,c}(s) \propto p(\ell|s)p(s|c). \tag{4}$$

and

$$p(\ell|s) \propto \sum_c p(c)p(\ell|c)q_{\ell,c}(s). \tag{5}$$

Here, we assume that the texture components indexed by s are associated with classes c through $p(s|c)$, and the iterative procedure finds association between the components and high-res labels ℓ, i.e. the distribution $p(\ell|s)$. A straightforward interpretation of this is that components s are associated with image patches, and so $p(\ell|s)$ only tells us the (probability of) label ℓ of the whole prototype s, and therefore the patches mapped to it. Thus, without the epitomic reasoning in Eq. 12 in the main text, the granularity with which we can assign high-res labels would depend on the patch size, and to get to the highest possible resolution, we would have to assume that s indexes the prototypes of a *single* pixel size. In other words, s would simply refer to a color clustering of the image (precisely, a component of a mixture model on colors).

Fig. 3 illustrates such an approach. Using a single 1km^2 RGBI tile, we train a mixture of 48 4-dimensional diagonal Gaussians indexed by s and show the cluster assignments for each pixel to these 48 clusters using a random color palette. The term $p(s|c)$ is then computed simply by computing the 48×20 matrix of co-occurrences of clusters s and coarse classes c, which are also shown in the figure, and then normalizing the matrix appropriately. Iterating the two equations above now leads to the 4×48 mapping of labels ℓ to components s in form of the distribution $p(\ell|s)$. When the most likely label is assigned to each cluster, we obtain the LSR result shown in the last panel in the figure. The resulting HR predictions are remarkable given that only pixel colors were used, but they also, unsurprisingly, exhibit a scattering of predicted impervious pixels all over the image, and in general a fairly speckled result. In addition, the model confuses the main road in the patch with fields and forests. On the other hand, the epitomic pixel-wise reasoning described in the main text yields the result

shown in the first row of Fig. 4, where the road is accurately predicted and the speckles are generally suppressed, indicating that our technique reasons about larger patterns and the individual pixels within them to assign single-pixel-level labels.

Table 2. Statistics $p(\ell|c)$ of the four HR labels in 15 of the 20 land cover classes that appear in our data. Statistics are from [13]. Our model did not use the available uncertainty in the statistics, but priors can easily be added in our model (see the future work section)

NLCD class	water	forest	field	imperv.
Open water	.97	.01	.01	.02
Developed, Open Space	.00	.42	.46	.11
Developed, Low	.01	.31	.34	.35
Developed, Medium	.01	.14	.21	.63
Developed, High	.01	.03	.07	.89
Barren Land	.09	.13	.45	.32
Deciduous Forest	.00	.92	.06	.01
Evergreen Forest	.00	.94	.05	.01
Mixed Forest	.01	.92	.06	.02
Shrub/Scrub	.00	.71	.26	.03
Grassland/Herbaceous	.01	.38	.54	.07
Pasture/Hay	.00	.11	.86	.03
Cultivated Crops	.00	.11	.86	.03
Woody Wetlands	.01	.90	.08	.00
Emergent Wetlands	.11	.07	.81	.01

6 Epitomic LSR in the 2020 IEEE-GRSS Data Fusion Contest

The IEEE Geoscience and Remote Sensing Society ran a competition on a task almost identical to the label super-resolution application on land cover described in the main paper. We show here that we can reach the top result in the competition through analysis *only* of the validation set (986 images), without ever looking at the very large training set of over 180k pairs of satellite images and their low-res labels. A detailed description of the winning method can be found in [17].

The goal of the 2020 IEEE GRSS data fusion competition [3] was to infer high-resolution (10m) labels in 8 classes (forest, shrubland, grassland, wetland, cropland, urban, barren, water) based on high-resolution (10m) 12-band Sentinel imagery and low-resolution (500m) MODIS labels (Fig. 7). To that end, a training set of around 180k 256×256 Sentinel images [18] and the corresponding MODIS labels [1] was provided, and the competition was performed on class average accuracy on a validation set (986 images) and a test set. We examine the

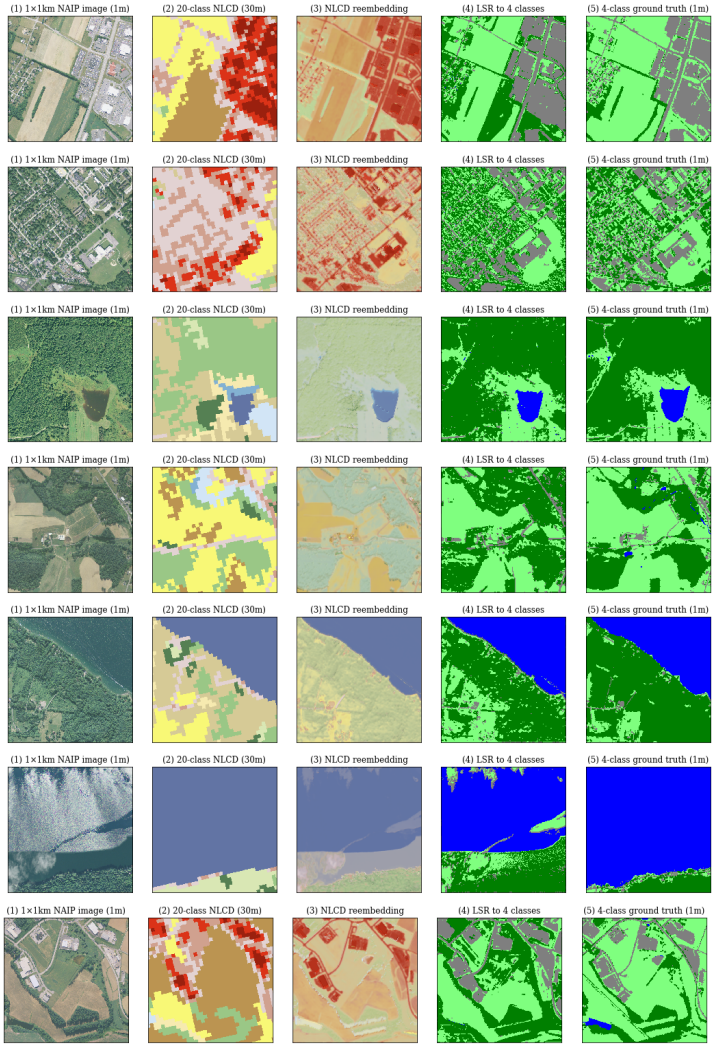

Fig. 4. Examples of single-tile (self-epitomic) LSR using 7×7 patches on $2\times$ downsampled images. The first row provides the analysis of the same tile shown super-resolved with a simple color model in Fig. 3. The rest are selected to illustrate both the success and failure modes of the single-tile approach

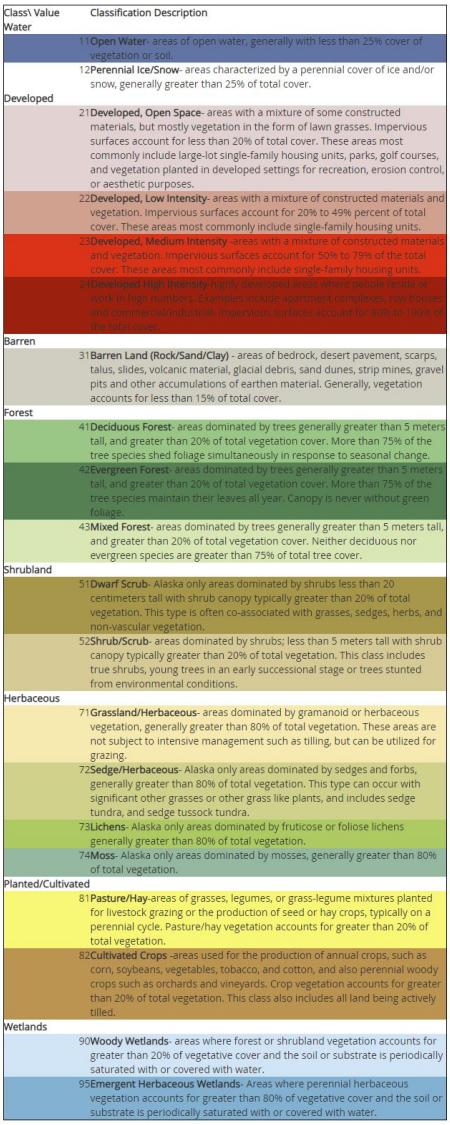

Fig. 5. National land cover database (NLCD) legend: descriptions and the color code [8]

validation set stage of the competition, in which 70 teams officially participated. The ground truth high-resolution labels for the validation set have since been made publicly available [4], making the following analysis possible.

The baseline methods [19] used both random forests and deep CNNs, which all achieved average accuracy not more than 54.1%. The top 10 teams' average accuracies on the validation set ranged between 68% and 71%. While we do not know how much data these teams used and which methods were tested by the teams in the competition, given the number of participants, it is probably safe to assume that convolutional neural networks and random forests were used in a variety of ways, as they were by the other winning teams in both this [5] and last year's [2] contests. Yet, our approach is a straightforward mixture model as described here and in the main text.

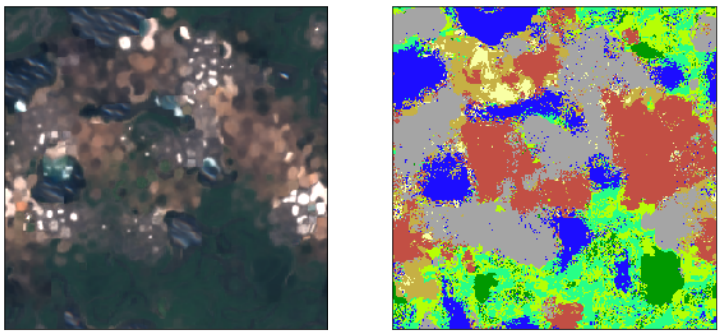

Fig. 6. 299×299 epitome of Sentinel imagery (RGB channels shown) and the output of the LSR algorithm, coded using the standard 10-class color scheme from [19]

In particular, we start with a simple color clustering model similar to the one in Sec. 5, equivalent to learning an epitome with patch size 1×1, because Sentinel imagery has many more bands than NAIP imagery, making separation of certain classes easier. For example, richer spectral information simplifies separating water from urban/built surfaces, as well as both of those from vegetation[4]. However, classifying the rest of the classes (forests, grassland, shrubland, barren, and croplands) requires analyzing spatial patterns, so we built an 299×299 epitome model with 7×7 patches (Fig. 6). LSR is performed in each model, using as $p(\ell|c)$ the statistical table provided by Fig. 3 in [19]. Then, we ensemble these predictions.

The color-only model (1×1 patches) achieves an average accuracy of 65.4%. The epitome model (7×7 patches) yields 65.5%. Either score would rank as 14/70. We ensemble these by trusting the color-only model's predictions on the

[4] This color clustering algorithm is augmented with a *bag of clusters* latent variable that makes the model sensitive to nearby colors. That approach is mostly independent of this work, so we direct the reader to [17] for the details.

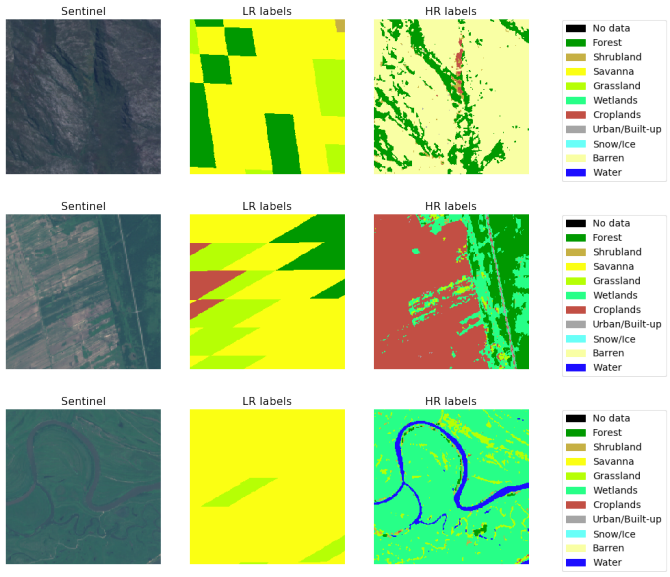

Fig. 7. Examples of high-res imagery, low-res labels, and high-res labels from the IEEE GRSS competition dataset

easily confused *shrubland* and *barren* classes to get average accuracy 68%, placing the model at 11/70.

However, it is important to note that the ground truth labels themselves are not entirely accurate, as they were created in a semi-manual manner, rather than fully manually [19]. The artifacts in the ground truth labels (Fig. 7) suggest that this 'ground truth' is actually based on a convolutional network's predictions (probably on lots of HR data).

To verify this, we trained a small neural network (a 5-layer fully convolutional network with two 3×3 convolutions and three 1×1 convolutions) on the predictions of our best ensemble model to introduce similar inductive biases, and this network reaches 70.7%, the top score on the leaderboard. Similar models, applied to the test set, also reached the highest accuracy of 57.5% in the main competition track.

In summary, using the simple statistical models described in this paper, it is possible to match the top score in an international competition in weakly supervised land cover mapping. Furthermore, our unsupervised learning approach is much more data-efficient than most modern-day supervised learning techniques, as evidenced by the fact that our results needed only 986 out of over 180k images in training.

7 Future work

We and other researchers interested in the epitomic representations discussed here have many possible follow-up research directions to explore.

As our models are easy to interpret, they can be used in scenarios where the predictions or just their errors have to be explained. The errors of the model as a predictor of high-res labels can be tracked not only to individual patches in the epitome, but through the epitome back to other patches in the data. In fact, as we discussed above, the visualization in Fig. 2 can be used to understand the domain shift of other models, such as U-Nets.

Traceability back to the training patches can also be used to further improve models through efficient hierarchical matching where the input patch is ultimately compared with an individual training patch, rather than its compressed version in the epitome.

It is also interesting to think about encoding various types of invariances as latent variables in the model, e.g. global illumination or local deformation variables.

The results in Fig. 4 all use the same statistics in Table 2 in inference, even though, clearly, the actual statistics of occurrence of different land cover labels in different NLCD classes will vary from image to image, indicating that the model is fairly robust to errors in these statistics. It would be interesting to see just how robust the approach is, and in particular, if the simplified summary statistics described in Fig. 5 could be used, or if it is possible to super-resolve a user's estimate of label percentage in a user-defined area in human-in-the-loop schemes.

While our label super-resolution formulation only uses the average frequencies of labels ℓ in different classes c shown in Table 2 and applies these on the entire dataset, or an entire single tile, the recent LSR work [13] also used the uncertainty estimates (variances on the frequencies across different 30m blocks). In their approach, a training cost is defined so that it uses both means and variances to match the statistics over predicted HR labels in each block, rather than summarily across the entire image. It is possible to use this information in our method, too. E.g., the label frequency variance can be used to set an appropriate Dirichlet prior on the $p(\ell|c)$ for each tile or even an individual 30m block, and then treat the parameters $p(\ell|c)$ as latent.

To infer labels for an image using the epitomes and $p(\ell|s)$ maps in Fig. 5 of the main text, we need to sample a large number of patches, covering the entire image, and compare them with the epitomes. In our current implementation, the inference time is within an order of magnitude of the U-Net's computational cost. However, the inference can be sped up through experimentation with sampling strategies. It would also be interesting to study approximate, e.g., coarse-to-fine, epitome mapping techniques, or even learnable indexing techniques that would speed up the label inference.

It would be interesting to test epitomic representations on recognition tasks.

Previous work on epitomic representations recognized that patch models are powerful in modeling local patterns larger than the patch size, as the patches

effectively get quilted into larger patterns, and just a few longer range relationships can go a long way towards grounding the quilting or capturing relationships useful in recognition [7,15]. Through a similar type of reasoning in models based on large scale epitomes it may be possible to further improve segmentation and recognition performance, e.g., in case of long features such as rivers and roads.

Recent work such as [6] has shown that the current state of the art in recognition is highly dependent on texture features, rather than image shapes. But, models we describe here can also be built on mask (shape) patches, as was shown in the paper that introduced epitomes [10], and also used in early co-segmentation work [9].

Beyond inspiration from those epitome works, we can use a variety of generative modeling work (before GANs and autoencoders), as discussed in the conclusions of the main paper, to build hierarchical models. But, using neural models of texture in combination with statistical reasoning is also possible.

References

1. Modis/terra+aqua land cover type yearly l3 global 500m. http://doi.org/10.5067/MODIS/mcd12q1.006
2. 2019 ieee grss data fusion contest results. http://www.grss-ieee.org/community/technical-committees/data-fusion/2019-ieee-grss-data-fusion-contest-results/ (2020)
3. 2020 ieee grss data fusion contest. https://competitions.codalab.org/competitions/22289#results (2020)
4. 2020 ieee grss data fusion contest. http://www.grss-ieee.org/community/technical-committees/data-fusion (2020)
5. 2020 ieee grss data fusion contest results. http://www.grss-ieee.org/community/technical-committees/data-fusion/2020-ieee-grss-data-fusion-contest-results/ (2020)
6. Brendel, W., Bethge, M.: Approximating CNNs with bag-of-local-features models works surprisingly well on imagenet. In: International Conference on Learning Representations (ICLR) (2019)
7. Cheung, V., Jojic, N., Samaras, D.: Capturing long-range correlations with patch models. In: 2007 IEEE Conference on Computer Vision and Pattern Recognition. pp. 1–8. IEEE (2007)
8. Homer, C., Dewitz, J., Yang, L., Jin, S., Danielson, P., Xian, G., Coulston, J., Herold, N., Wickham, J., Megown, K.: Completion of the 2011 national land cover database for the conterminous united states–representing a decade of land cover change information. Photogrammetric Engineering & Remote Sensing **81**(5), 345–354 (2015)
9. Jojic, N., Caspi, Y.: Capturing image structure with probabilistic index maps. In: Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004. vol. 1, pp. I–I. IEEE (2004)
10. Jojic, N., Frey, B.J., Kannan, A.: Epitomic analysis of appearance and shape. In: ICCV. vol. 3, p. 34 (2003)
11. Jojic, N., Perina, A., Murino, V.: Structural epitome: a way to summarize one's visual experience. In: Advances in neural information processing systems. pp. 1027–1035 (2010)
12. Kingma, D., Ba, J.: Adam: A method for stochastic optimization. International Conference on Learning Representations (12 2014)
13. Malkin, K., Robinson, C., Hou, L., Soobitsky, R., Czawlytko, J., Samaras, D., Saltz, J., Joppa, L., Jojic, N.: Label super-resolution networks. International Conference on Learning Representations (2019)
14. Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., Lerer, A.: Automatic differentiation in PyTorch. In: NIPS Autodiff Workshop (2017)
15. Perina, A., Jojic, N.: Spring lattice counting grids: Scene recognition using deformable positional constraints. In: European Conference on Computer Vision. pp. 837–851. Springer (2012)
16. Robinson, C., Hou, L., Malkin, K., Soobitsky, R., Czawlytko, J., Dilkina, B., Jojic, N.: Large scale high-resolution land cover mapping with multi-resolution data. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. pp. 12726–12735 (2019)
17. Robinson, C., Malkin, K., Hu, L., Dilkina, B., Jojic, N.: Weakly supervised semantic segmentation in the 2020 IEEE GRSS Data Fusion Contest. Proceedings of the International Geoscience and Remote Sensing Symposium (2020)

18. Schmitt, M., Hughes, L.H., Qiu, C., Zhu, X.X.: Sen12ms–a curated dataset of georeferenced multi-spectral sentinel-1/2 imagery for deep learning and data fusion. ISPRS Journal of Photogrammetry and Remote Sensing (2020)
19. Schmitt, M., Prexl, J., Ebel, P., Liebel, L., Zhu, X.X.: Weakly supervised semantic segmentation of satellite images for land cover mapping–challenges and opportunities. arXiv preprint arXiv:2002.08254 (2020)