One-shot Texture Segmentation

Ivan Ustyuzhaninov

University of Tübingen ivan.ustyuzhaninov@bethgelab.org

Wieland Brendel*

University of Tübingen wieland.brendel@bethgelab.org

Claudio Michaelis

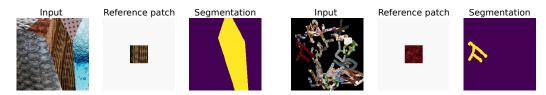
University of Tübingen claudio.michaelis@bethgelab.org

Matthias Bethge*

University of Tübingen matthias@bethgelab.org

Abstract

We introduce one-shot texture segmentation: the task of segmenting an input image containing multiple textures given a patch of a reference texture. This task is designed to turn the problem of texture-based perceptual grouping into an objective benchmark. We show that it is straight-forward to generate large synthetic data sets for this task from a relatively small number of natural textures. In particular, this task can be cast as a self-supervised problem thereby alleviating the need for massive amounts of manually annotated data necessary for traditional segmentation tasks. In this paper we introduce and study two concrete data sets: a dense collage of textures (CollTex) and a cluttered texturized Omniglot data set. We show that a baseline model trained on these synthesized data is able to generalize to natural images and videos without further fine-tuning, suggesting that the learned image representations are useful for higher-level vision tasks.



(a) Collage of natural textures

(b) 32 Omniglot characters with overlayed textures

Figure 1: Examples of texture segmentation tasks. In each example the three images correspond to input image, reference texture and target segmentation (from left to right).

1 Introduction

The human visual system is remarkably robust to many local image transformations. There is evidence that outside of the foveal region the visual system represents image patches as textures [44, 10] which are robust to local shifts and deformations. Such a compression can be extremely useful as it summarizes the mostly irrelevant fine-scale details (e.g. the individual blades of grass) but keeps the rough low-level semantic concepts (e.g. grass). This suggests that local texture features can form an important mid-level image representation.

To learn such representations we suggest the task of one-shot texture segmentation: given an image containing multiple textures as well as a reference texture, the task is to segment parts of the input

^{*}co-senior authors

image covered by the reference texture (see Figure 1). Since these parts can be highly variable in size and shape, models have to learn an extremely flexible and robust texture representation that can finely discriminate the local image statistics. Such a problem formulation allows for algorithmic data set generation which in turn allows to turn the underlying problem of perceptual texture grouping into a supervised learning task.

Our contributions are as follows:

- We introduce and openly release two benchmark data sets for one-shot texture segmentation as well as the code necessary to generate the data in a self-supervised fashion (Section 3).
- We introduce and train a strong baseline segmentation model for this task (Sections 4, 5 and 6.4).
- We show that one-shot texture segmentation requires computations of local image statistics of higher-orders. (Section 6.2).
- We demonstrate that the texture representations learned by our model generalize to natural images and can be used to obtain coarse semantic segmentations of natural images and videos without additional fine-tuning (Sections 6.6 and 6.7).

2 Related work

Segmentation Segmentation tasks can be roughly split into Semantic Segmentation [24, 6, 33] (including the recently introduced class-agnostic segmentation [25, 28]) and Instance Segmentation [30, 14]. Popular models for this task are encoder-decoder architectures with skip connections [4, 33] which are often based on pretrained DNNs like VGG [36] as feature extractors. Virtually all existing segmentation tasks [9, 22, 46] are based on manual human labels which are usually hard to get [46, 47, 2, 5] and concentrate on high-level semantic concepts. In contrast, we here focus on segmentations defined by mid-level texture cues for which tasks can be defined in a self-supervised way.

One-Shot Learning While the corpus of work on one-shot learning is quite extensive [20, 19, 43, 38, 40, 37, 32, 39], more complex tasks like one-shot segmentation [3, 17] have been established only recently. Our work is most similar to the Omniglot one-shot segmentation setting recently introduced by [26] in which cluttered Omniglot images are segmented given a reference shape. However, instead of segmenting based on shape similarity, we here focus on segmentations defined by the texture of the reference patch.

Texture modelling Computational modelling of textures has been a long-standing task in computer vision. Parametric models based on image statistics (*N*-th order joint histograms) were introduced by [16], followed by models based on other statistical measurements [15, 31]. Current state-of-the-art parametric texture models are based on DNN features [11, 23]. We build on these works and use DNNs as extractors of texture parameters to build our model. Another line of work in texture modelling focuses on non-parametric methods [8, 21], which are less suited for our approach since we rely on parametric representations to match textures in input and reference images.

Texture Perception Textures are well-suited for studying human perception as they are diverse, non-trivial stimuli. There is evidence that certain areas of the human brain are building texture representations to perform low-level visual inference [45, 27]. Within the substantial body of work on texture modelling [41], experimental settings very similar to ours have been introduced. For instance, in [42, 13] human subjects have to perform texture segmentation of images consisting of multiple textures, relying on local image statistics in a cluttered scene. In [35] a biologically motivated model of texture perception has been used to recognize objects. Finally, textures have also been used to study the difference between human perception and computational models of vision [12].

[†]https://github.com/ivust/one-shot-texture-segmentation

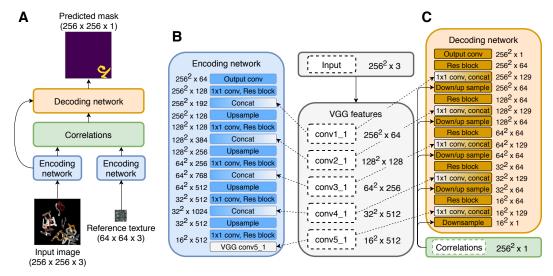


Figure 2: A: Overview of the architecture of the model, B: the architecture of the encoding network, C: the architecture of the decoding network. Residual blocks used in the encoder and decoder are identical (see text for details).

3 One-shot texture segmentation data sets

In the following section we describe two data sets for one-shot texture segmentation that can be algorithmically generated from a set of unlabelled natural texture images. We use textures from the Describable Textures Data set (DTD, 5640 textures) [7] and synthesized ones from a VGG texture model [11]. The input images are of the size 256×256 and contain multiple textures. One of these textures is presented as a 64×64 pixels reference image. The task for all data sets is to segment out the parts of the input image covered by the reference texture.

3.1 Collages of natural textures (CollTex)

Our first data set consists of collages of natural textures. Each input image is partitioned randomly into non-overlapping regions which are filled with randomly selected textures from a fixed training set of texture images (DTD, see examples in Figures 1 and 3). To generate random partitions, we uniformly sample N anchor points at random positions in the image, and then assign each image pixel to one of the N segmentation regions defined by the closest anchor point. We generate samples in real-time and reserve 100 randomly chosen DTD textures for testing.

3.2 Texturized cluttered Omniglot

The second data set we consider is based on texturized Omniglot characters [20]. It is generated by uniformly spreading texturized Omniglot characters on a black background. To do so, we take 105×105 characters from the Omniglot data set, scale them by a factor sampled from U[0.5,2] and rotate by an angle sampled from $U[0,2\pi)$. These transformed characters are used as masks for textures, which are then put into a 256×256 image at random locations with (x,y) coordinates being integers independently sampled from U[28,228]. Examples of the resulting images are in Figure 3.

Since Omniglot characters are usually composed of thin lines, we cannot use textures with long-range spatial dependencies (such as natural textures). Therefore we re-synthesize the DTD textures by matching Gram matrices in the first convolutional layer of the VGG network [11] and use these simpler textures to fill in the Omniglot characters (Figure 3). Resulting textures have shorter length scales not exceeding the typical size of an Omniglot character but contain patterns which cannot be modelled by simple colour matching as shown in Sec. 6.2. We reserve 100 randomly chosen textures and characters for testing.

A more challenging version of this data set is obtained by adding random background textures that are different from the textures of the Omniglot characters. The background textures are also re-synthesized DTD textures as described above. An example is given in Figure 3.

4 Model architecture

We solve the task of one-shot texture segmentation in three steps. First, we compute embeddings of an input image and a reference patch; second, we search for the reference texture in the embedding space to produce a rough segmentation mask; and, finally, we employ a decoding network to produce the output segmentation. The architecture is summarized in Figure 2.

4.1 Encoding network

The image embeddings that we are looking for should have two properties: (1) they should map different instances of the same texture to the same point in the embedding space, and (2) they should be local, i.e. for each spatial location encode only the texture in the neighborhood of this location. To ensure (1), we build our encoding network on top of the VGG features [36], which are known to be good texture models [11]. To ensure (2), the output of the encoding network has the same spatial size as the input, where the feature vector at each spatial position can be thought of as a representation of a local texture at this position.

To compute the embedding of an image (either an input one or a reference patch), we start by extracting the VGG features for this image (specifically, feature maps in layers conv1_1, conv2_1, conv3_1, conv4_1, conv5_1). Next, starting with the layer conv5_1, we repeatedly employ a computational unit consisting of a residual block, upsampling (bilinear interpolation), and concatenation of VGG features of the corresponding resolution to ultimately obtain an embedding of the same spatial resolution as the input. In particular, we use layer conv5_1 as an input to a residual block, consisting of three convolutional layers with 3×3 kernels (the first two of which are followed by ReLU non-linearities, while the third one is linear), and then up-sample the output of the residual block by a factor of 2 along each spatial dimension (bilinear interpolation). The resulting tensor has the same spatial dimensions as the VGG layer conv4_1, which we concatenate to it. Thereafter we repeat the same series of computations (residual block, upsampling, concatenation) until we obtain a tensor of the same spatial resolution as the input, to which we apply a 1×1 convolutional layer to obtain the final embedding.

The architecture of the encoding network and the exact numbers of feature maps are shown in Figure 2B.

4.2 Searching in the encoded space

Having encoded both the input image and the reference patch, we independently normalize the feature vectors at each spatial position to have norm equal to 1, and spatially convolve the two embeddings with each other. This corresponds to computing cosine distances (up to a constant factor) between them at each spatial position. The resulting feature map of the same spatial size as the input image shows locations with textures similar to the one of the reference patch (see the green boxes in Figure 2). It is then processed by the decoding network to produce the final segmentation.

4.3 Decoding network

The decoding network has a similar architecture to the encoding one (Figure 2C). The main difference is that while the architecture of the residual blocks remains the same, we concatenate not only the VGG features after the upsampling layers, but also the (downsampled) map of cosine distances obtained in the previous step (Section 4.2) Before concatenating the VGG features, we pass them through a 1×1 convolutional layer to reduce the number of feature maps to 64, which we found to be sufficient. After the final residual layer we apply a 1×1 convolution with sigmoid non-linearity to produce a single segmentation map $(256\times 256\times 1)$. The value of each pixel in this map can be interpreted as the probability that this pixel belongs to the reference texture.

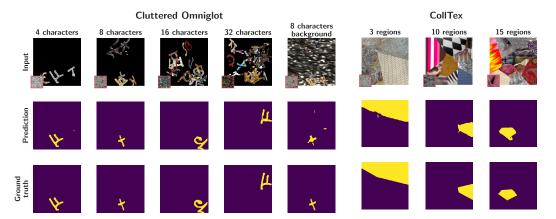


Figure 3: Qualitative results of one-shot texture segmentation for cluttered Omniglot images with different number of characters (left) and CollTex images with different numbers of segmentation regions (right). Input images are shown in the first row, predicted segmentation in the second, and ground truth segmentations in the third. Reference textures are shown in the bottom-left corner of the input image in a red frame.

5 Training and Evaluation

5.1 Loss function

We train using standard pixel-wise cross entropy loss [24] and apply a weighting term to compensate for the difference in foreground and background size. We denote the ground-truth segmentation mask for the input image \mathbf{x} as $S(\mathbf{x}) \in \{0,1\}^{256 \times 256}$, and the predicted one (i.e. the output of the decoding network, Section 4.3) as $\hat{S}(\mathbf{x}) \in [0,1]^{256 \times 256}$. To compare $S(\mathbf{x})$ and $\hat{S}(\mathbf{x})$, we use pixel-wise binary cross-entropy loss. Since most of the pixels belong to the background, we separately compute losses for foreground and background and scale them by the corresponding numbers of pixels. Specifically, if the total number of foreground $N_{\rm fg}$ and background $N_{\rm bg}$ pixels are given by

$$N_{\rm fg} = \sum_{ij} S_{ij}(\mathbf{x})$$
 and $N_{\rm bg} = \sum_{ij} 1 - S_{ij}(\mathbf{x}),$ (1)

we define the loss $\mathcal{L}(\mathbf{x})$ for an image \mathbf{x} as

$$\mathcal{L}_{fg}(\mathbf{x}) = -\frac{1}{N_{fg}} \sum_{ij} S_{ij}(\mathbf{x}) \log \hat{S}_{ij}(\mathbf{x}), \tag{2}$$

$$\mathcal{L}_{bg}(\mathbf{x}) = -\frac{1}{N_{bg}} \sum_{ij} (1 - S_{ij}(\mathbf{x})) \log (1 - \hat{S}_{ij}(\mathbf{x})), \tag{3}$$

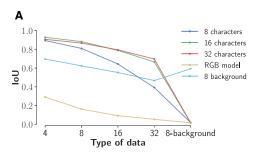
$$\mathcal{L}(\mathbf{x}) = \mathcal{L}_{fg}(\mathbf{x}) + \mathcal{L}_{bg}(\mathbf{x}). \tag{4}$$

5.2 Training

The training of the model consists of minimizing the mean loss $\mathbb{E}[\mathcal{L}(\mathbf{x})]$ over the training data set with respect to the parameters of the convolutional layers of the encoding and decoding networks while keeping the weights of the VGG feature extractor network fixed (see Section 4). We implement the networks in TensorFlow [1], estimate $\mathbb{E}[\mathcal{L}(\mathbf{x})]$ using mini-batches of 8 images, and use Adam optimizer [18] with all parameters set to default values. We set the learning rate to the initial value of 10^{-5} and decrease it by a factor of 2 every $4 \cdot 10^5$ steps.

5.3 Evaluation

We use the Intersection over Union (IoU) as a metric to evaluate the quality of the predicted segmentation masks (binarized using a threshold of 0.9, which we found to be optimal). All evaluations are done on the held-out textures and characters not used during training.



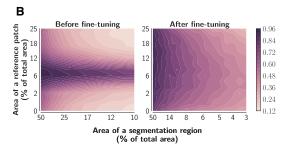


Figure 4: A: IoU scores on the test data for different types of cluttered Omniglot input images for models trained on Omniglot data sets with different numbers of characters. **B**: IoU scores for the model trained on collages of natural textures before and after fine-tuning on reference patches of variables sizes as a function of a reference patch size and an average area of a segmentation region.

6 Results

In this section we discuss our results on CollTex and texturized cluttered Omniglot, demonstrate the need for a sufficiently capable encoder, and finally, demonstrate two possible applications.

6.1 Proposed model is a strong baseline

To train the CollTex model, at each training iteration we draw a random integer $N \in [0, 10]$ and create a random collage of N textures (Section 3.1). After training this model achieves IoU scores between 94% (N = 2) and 76.5% (N = 10) on the held-out set of textures (see more data in Figure 4B(left) and Table 3).

For texturized cluttered Omniglot, we trained separate models on data sets of 8, 16, and 32 characters, as well as on a data set of 8 characters on a background texture. The performance on the images composed of the held-out characters and textures is between 92.9% (on images with 4 characters) and 69.7% (images with 32 characters) depending on the number of characters, and 58.9% for 8 characters with texture background (Figure 4A and Table 1).

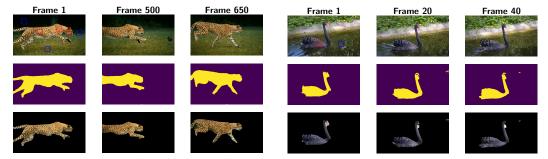
Similar to what was reported before [26], we find that the task becomes harder as we increase the amount of clutter by increasing the number of characters (Figure 4A). The background texture seems to create clutter uniformly throughout the image making the task even more challenging (see IoU scores in Figure 4A and compare the images in the second and fifth columns in Figure 3). A similar effect can be seen for collages of natural textures: here the performance decreases as the average area of a segmentation region gets smaller (Figure 4B and and Table 1).

6.2 Encoding network learns non-trivial texture embeddings

To test whether the networks learn to compute higher order texture statistics, we replaced our encoding network (see Section 4.1) with a simpler version consisting of only one linear convolutional layer (with 1×1 filters), which can only learn to perform a color transformation separately on each pixel. On texturized Omniglot this model (called the "RGB model" in Figure 4(a)) performs very poorly compared to the full encoding network. This suggests that the task is not easily solvable in pixel space and the encoding network computes non-trivial and useful texture representations. This result holds for CollTex as well (see Tables 4 and 6).

6.3 Pre-training the VGG features extractor is not necessary

The VGG features pre-trained on the ImageNet are known to be good texture models [11], therefore we build our model on top of them. However, we would like to test if such pre-training is necessary to solve our task. To do so we randomly initialize the VGG weights, and optimize them jointly with the encoding and decoding networks. We find that a network trained in such a way performs similarly to the one using pre-trained VGG (Tables 3 and 5), suggesting that the proposed architecture allows to learn texture representations sufficient for the task without any supervision.



(a) An excerpt from a YouTube video.

(b) The "blackswan" video from DAVIS 2016 [29].

Figure 5: Examples of one-shot texture segmentation applied to videos. From the first frame multiple patches containing the texture of an object (shown in red) and background (shown in blue) are extracted. Segmentations for subsequent frames are produced by combining the generated segmentations for each of these patches.

6.4 Training on more difficult data sets helps obtain better models

We evaluated all of our Omniglot models on data sets with 4, 8, 16 and 32 characters. Figure 4(a) shows that the models trained on data sets containing more characters generalize to a smaller number of characters, but not vice versa (best IoU scores are achieved with models trained on 16 and 32 characters). A background texture introduces a different kind of clutter compared to Omniglot characters, making models trained on Omniglot unable to generalize to this case.

6.5 Model performance is independent of reference patch size

As described in Section 3, we used fixed-size (64×64) reference patches for training. That results in size-dependant models with dropping performance as we use reference patches of different sizes (Figure 4(b)). However, fine-tuning the model on reference patches with a randomly chosen size allows to recover the performance for a wide range of patch sizes (Figure 4(b), right panel). We observed the same effect on texturized Omniglot images. This suggests that the proposed model generalizes to different patch sizes and can perform texture matching for patches as small as 16×16 .

6.6 Demo: Segmenting objects by their textures in videos

In Figure 5 we present qualitative results for natural video object segmentation using the model trained on CollTex. From the first frame we extract both the object textures (shown in red) and the background textures (shown in blue). Using them we obtain the segmentation mask of subsequent frames by combining the mask \hat{S}_{obj} for the object texture and the one \hat{S}_{bg} for the background: $\hat{S} = \hat{S}_{\text{obj}} \cdot (1 - \hat{S}_{\text{bg}})$. The video in Figure 5B was taken from the DAVIS 2016 data set [29], which provides the ground-truth segmentations. Using only two texture patches our model reaches an IoU of 84.8% despite being trained exclusively on the synthetic CollTex data set. In comparison, the current state-of-the-art semi-supervised and unsupervised models (which are designed and trained explicitly for this data set) reach the corresponding IoU scores of 96.3% and 90.4%. Further discussion of these results can be found in Section 7.

6.7 Demo: Classifying objects by their textures

We extracted images of five different classes (leopard, salamander, zebra, monarch butterfly, hedgehog) from the ImageNet data set [34] to demonstrate that our model trained only on synthetic collages of textures can be used for quick object classification in natural images. For each class we extract the 100 most predictive texture patches (32×32 pixels) using a deep bag-of-features model (manuscript submitted to NIPS, see example patches in Figure 6(a)). Using these we perform a one-vs-all classification by selecting those images in which more than 30% of the image is covered by the target

[†]https://youtu.be/B4nd9GF1dRg

^{*}Results from http://davischallenge.org/davis2016/soa_compare.html



patches for discriminating leopards from other animals. in panel (a) (incorrectly classified: area $\leq 30\%$).

(a) Best (top row) and worst (bottom row) leopard (b) Leopard image segmented using the top-left patch



(c) Leopard image segmented using the top-left patch (d) Different class image segmented using the top-left in panel (a) (correctly classified: area > 30%). patch in panel (a) (correctly classified: area $\leq 30\%$).

Figure 6: Examples of classification of ImageNet images of leopards against images of 4 other ImageNet classes (see details in the text) by their textures.

texture. For the leopard class this leads to a test performance of 85%. In comparison, a linear model on pixel values achieves around 70%. In addition to classifying images using the texture patches from the bag-of-features model, we can use the classification accuracy of our model to identify the most and least informative patches (see Figure 6(a)). We do so by using each of the extracted patches as a reference patch for the texture segmentation model, and employ the accuracy of a classifier based on these segmentations as a numerical value for how informative the patch is. Upon visual inspection there is little apparent variation among the most informative patches, while the colour noticeably changes across the least informative ones.

Discussion 7

We introduced the task of one-shot texture segmentation, which we believe is a powerful formalization of texture-based perceptual grouping in terms of a self-supervised learning objective. It allows us to learn useful low- and mid-level representations, which we found to generalize to natural images and videos.

We studied this task on two data sets: (1) a dense collage of multiple textures (CollTex) and (2) texturized cluttered Omniglot. We introduced a strong baseline model to solve these tasks, and demonstrated competitive performance even on task configurations that may be challenging for humans (e.g. 32 characters and 15 regions in Figure 3).

As reported in Sections 6.6 and 6.7, our CollTex model generalizes to segmenting objects in natural images and videos, demonstrating its ability to form useful image representations. Even though the performance of our model does not match the current state-of-the-art, it achieves competitive scores without any fine-tuning to the specific task, highlighting its superior generalization properties.

We believe that this work is not only an important addition to self-supervised and one-shot learning, but it also offers new insights into the generalization capabilities provided by the texture representations. We therefore expect the proposed one-shot texture segmentation task to become an important learning target for mid-level representations.

References

- [1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Largescale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
- [2] David Acuna, Huan Ling, Amlan Kar, and Sanja Fidler. Efficient interactive annotation of segmentation datasets with polygon-rnn++. In *CVPR*, 2018.
- [3] Zhen Liu Irfan Essa Byron Boots Amirreza Shaban, Shray Bansal. One-shot learning for semantic segmentation. In *BMVC*, 2017.
- [4] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39(12):2481–2495, Dec 2017.
- [5] A. Barriuso and A. Torralba. Notes on image annotation. ArXiv e-prints, October 2012.
- [6] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In Proceedings of the 4rd International Conference on Learning Representations (ICLR), 2015.
- [7] M. Cimpoi, S. Maji, I. Kokkinos, S. Mohamed, , and A. Vedaldi. Describing textures in the wild. In *Proceedings of the IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2014.
- [8] A. A. Efros and T. K. Leung. Texture synthesis by non-parametric sampling. In *Proceedings of the Seventh IEEE International Conference on Computer Vision*, volume 2, pages 1033–1038 vol.2, 1999.
- [9] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.
- [10] Jeremy Freeman and Eero P Simoncelli. Metamers of the ventral stream. *Nature Neuroscience*, 14:1195.
- [11] Leon A. Gatys, Alexander S. Ecker, and Matthias Bethge. Texture synthesis using convolutional neural networks. In *Advances in Neural Information Processing Systems* 28 (NIPS), 2015.
- [12] Stephen Grossberg and Ennio Mingolla. Neural dynamics of perceptual grouping: Textures, boundaries, and emergent segmentations. *Perception & Psychophysics*, 38(2):141–171, Mar 1985.
- [13] S.J. Harrison and D.R.T. Keeble. Within-texture collinearity improves human texture segmentation. *Vision Research*, 48(19):1955 1964, 2008.
- [14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 2980–2988, 2017.
- [15] D. J. Heeger and J. R. Bergen. Pyramid-based texture analysis/synthesis. In *Proceedings.*, *International Conference on Image Processing*, volume 3, pages 648–651 vol.3, Oct 1995.
- [16] Bela Julesz. Visual pattern discrimination. IRE Transactions on Information Theory, 8(2):84–92, February 1962.
- [17] Trevor Darrell Alyosha Efros Sergey Levine Kate Rakelly, Evan Shelhamer. Conditional networks for few-shot semantic segmentation. In *ICLR workshop*, 2018.
- [18] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings* of the 3rd International Conference on Learning Representations (ICLR), 2014.
- [19] Gregory Koch. Siamese neural networks for one-shot image recognition. In *ICML Deep Learning workshop*, 2015.
- [20] Brenden M. Lake, Ruslan Salakhutdinov, and Joshua B. Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

- [21] Elizaveta Levina and Peter J. Bickel. Texture synthesis and nonparametric resampling of random fields. *Ann. Statist.*, 34(4):1751–1773, 08 2006.
- [22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision ECCV 2014, pages 740–755, Cham, 2014. Springer International Publishing.
- [23] Gang Liu, Y. Gousseau, and G. S. Xia. Texture synthesis through convolutional neural networks and spectrum constraints. In 2016 23rd International Conference on Pattern Recognition (ICPR), pages 3234–3239, Dec 2016.
- [24] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3431–3440, June 2015.
- [25] K.K. Maninis, S. Caelles, J. Pont-Tuset, and L. Van Gool. Deep extreme cut: From extreme points to object segmentation. 2018.
- [26] C. Michaelis, M. Bethge, and A. S. Ecker. One-Shot Segmentation in Clutter. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*. PMLR, 2018.
- [27] Isamu Motoyoshi, Shin'ya Nishida, Lavanya Sharan, and Edward Adelson. Image statistics and the perception of surface qualities. 447:206–9, 06 2007.
- [28] Dimitrios Papadopoulos, Jasper Uijlings, Frank Keller, and Vittorio Ferrari. *Extreme clicking for efficient object annotation*. 7 2017.
- [29] F. Perazzi, J. Pont-Tuset, B. McWilliams, L. Van Gool, M. Gross, and A. Sorkine-Hornung. A benchmark dataset and evaluation methodology for video object segmentation. In *Computer Vision and Pattern Recognition*, 2016.
- [30] Pedro O Pinheiro, Ronan Collobert, and Piotr Dollar. Learning to segment object candidates. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 28, pages 1990–1998. Curran Associates, Inc., 2015.
- [31] Javier Portilla and Eero P. Simoncelli. A parametric texture model based on joint statistics of complex wavelet coefficients. *International Journal of Computer Vision*, 40(1):49–70, Oct 2000.
- [32] Ting Yao Chenggang Yan Tao Mei Qi Cai, Yingwei Pan. Memory matching networks for one-shot image recognition. In CVPR, 2018.
- [33] O. Ronneberger, P.Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. (available on arXiv:1505.04597 [cs.CV]).
- [34] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015.
- [35] T. Serre, L. Wolf, S. Bileschi, M. Riesenhuber, and T. Poggio. Robust object recognition with cortex-like mechanisms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(3):411–426, March 2007.
- [36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. CoRR, abs/1409.1556, 2014.
- [37] Wei Shen Alan Yuille Siyuan Qiao, Chenxi Liu. Few-shot image recognition by predicting parameters from activations. In *CVPR*, 2018.
- [38] Jake Snell, Kevin Swersky, and Richard Zemel. Prototypical networks for few-shot learning. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, Advances in Neural Information Processing Systems 30, pages 4077–4087. Curran Associates, Inc., 2017.
- [39] Nikos Komodakis Spyros Gidaris. Dynamic few-shot visual learning without forgetting. In CVPR, 2018.

- [40] Eleni Triantafillou, Richard Zemel, and Raquel Urtasun. Few-shot learning through an information retrieval lens. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 30, pages 2255–2265. Curran Associates, Inc., 2017.
- [41] Jonathan D. Victor, Mary M. Conte, and Charles F. Chubb. Textures as probes of visual processing. *Annual Review of Vision Science*, 3(1):275–296, 2017. PMID: 28937948.
- [42] Jonathan D. Victor, Daniel J. Thengone, Syed M. Rizvi, and Mary M. Conte. A perceptual space of local image statistics. *Vision Research*, 117:117 135, 2015.
- [43] Oriol Vinyals, Charles Blundell, Tim Lillicrap, koray kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems* 29, pages 3630–3638. Curran Associates, Inc., 2016.
- [44] T. S. A. Wallis, C. M. Funke, A. S. Ecker, L. A. Gatys, F. A. Wichmann, and M. Bethge. A parametric texture model based on deep convolutional features closely matches texture appearance for humans. *Journal of Vision*, 17(12), 2017.
- [45] Yunguo Yu, Anita M. Schmid, and Jonathan D. Victor. Visual processing of informative multipoint correlations arises primarily in v2. *eLife*, 4, 2015.
- [46] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.
- [47] Yan Zhu, Yuandong Tian, Dimitris Metaxas, and Piotr Dollar. Semantic amodal segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

A Summary of IoU results for cluttered Omniglot and CollTex

		Trained on (number of characters)					
		8	16	32	32 RGB	8-back	
Evaluated on	4	0.894	0.929	0.907	0.291	0.697	
	8	0.810	0.883	0.867	0.162	0.624	
	16	0.644	0.789	0.795	0.092	0.553	
	32	0.393	0.666	0.697	0.053	0.467	
	8-back	0.018	0.017	0.017	0.017	0.593	

Table 1: IoU scores for the cluttered Omniglot data set with models trained and evaluated on different number of characters (columns correspond to the training data, rows correspond to the evaluation data).

		Type of a model (10 segmentation regions)					
		Fine-tuned					
			on variable	No VGG			
		Regular	patch sizes	pre-training	RGB		
	16×16	0.153	0.524	0.111	0.172		
Reference patch size	32×32	0.184	0.644	0.150	0.187		
	64×64	0.765	0.692	0.693	0.169		
	96×96	0.209	0.700	0.098	0.171		
	128×128	0.131	0.576	0.098	0.167		

Table 2: IoU scores for models trained on CollTex images with 10 segmentation regions. The columns correspond to different models: regular one, the one fine-tuned on reference patches of variable sizes, the one not using pre-trained VGG, and the RGB model using a pixel-wise color transform for encoding.

B Detailed IoU results for CollTex

		Number of segmentation regions			
		2 5 10 20			
Reference patch size	16×16	0.556	0.254	0.153	0.086
	32×32	0.661	0.322	0.184	0.113
	64×64	0.940	0.856	0.765	0.603
	96×96	0.699	0.391	0.209	0.010
	128×128	0.590	0.256	0.131	0.062

Table 3: IoU scores for a model trained on CollTex images (using 64×64 reference patches).

		Number of segmentation regions			
		2 5 10 20			
Reference patch size	16×16	0.829	0.658	0.524	0.432
	32×32	0.870	0.774	0.644	0.484
	64×64	0.913	0.819	0.692	0.543
	96×96	0.917	0.845	0.700	0.509
	128×128	0.894	0.762	0.576	0.371

Table 4: IoU scores for a model trained on CollTex images (using 64×64 reference patches) and fine-tuned on reference patches of variable sizes.

		Number of segmentation regions			
		2	5	10	20
Reference patch size	16×16	0.528	0.215	0.111	0.067
	32×32	0.623	0.280	0.150	0.095
	64×64	0.911	0.805	0.693	0.546
	96×96	0.498	0.200	0.098	0.050
	128×128	0.474	0.198	0.098	0.050

Table 5: IoU scores for a model trained on CollTex images (using 64×64 reference patches) without pre-training the VGG features extractor.

		Number of segmentation regions			
		2 5 10 20			
Reference patch size	16×16	0.607	0.320	0.172	0.095
	32×32			0.172	0.093
	64×64			0.169	0.106
	96×96	0.615	0.313	0.171	0.101
	128×128	0.620	0.306	0.167	0.089

Table 6: IoU scores for a model trained on CollTex images (using 64×64 reference patches) using the encoding network consisting of a single linear 1×1 convolutional layer.