# Learning-based Sampling for Natural Image Matting

Jingwei Tang[1,2] Yağız Aksoy[2] Cengiz Öztireli[1] Markus Gross[1,2] Tunç Ozan Aydın[1]

[1] Disney Research [2] ETH Zürich

## Abstract

*The goal of natural image matting is the estimation of opacities of a user-defined foreground object that is essential in creating realistic composite imagery. Natural matting is a challenging process due to the high number of unknowns in the mathematical modeling of the problem, namely the opacities as well as the foreground and background layer colors, while the original image serves as the single observation. In this paper, we propose the estimation of the layer colors through the use of deep neural networks prior to the opacity estimation. The layer color estimation is a better match for the capabilities of neural networks, and the availability of these colors substantially increase the performance of opacity estimation due to the reduced number of unknowns in the compositing equation. A prominent approach to matting in parallel to ours is called sampling-based matting, which involves gathering color samples from known-opacity regions to predict the layer colors. Our approach outperforms not only the previous hand-crafted sampling algorithms, but also current data-driven methods. We hence classify our method as a hybrid sampling- and learning-based approach to matting, and demonstrate the effectiveness of our approach through detailed ablation studies using alternative network architectures.*

## 1. Introduction

Natural image matting is the estimation of the accurate soft transitions between a user-defined foreground and the background of an image. These soft transitions define the opacity of the foreground at each pixel, and the resulting *alpha matte* is one of the core elements in image and video editing workflows that is essential in compositing. Matting is a fundamental operation for various tasks during the post-production stage of feature films, such as compositing live-action and rendered elements together, and performing local color corrections.

Formally, the color mixtures in soft transitions between foreground and background are typically represented with the compositing equation

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \tag{1}$$

where $\alpha_i \in [0, 1]$ denotes the opacity of the foreground at pixel $i$. $I$ is the original image, and $F$ and $B$ represent the unknown color values of the foreground and the background respectively. Typically, user input is provided in the form of a *trimap*, which provides a label for every pixel as purely foreground (i.e. opaque, $\alpha = 1$), purely background (i.e. transparent, $\alpha = 0$), or of unknown opacity. Matting algorithms aim to estimate the unknown opacities by making use of the colors of the pixels in the known-opacity regions.

Data-driven matting algorithms [5, 14, 20] typically aim to estimate the alpha values directly. In this paper, we propose to estimate the other unknowns in (1), $F$ and $B$, using deep neural networks prior to the opacity estimation process. Our main motivation is to increase the matting accuracy by reducing the number of unknowns in (1). This goal of layer color estimation has many similarities with image inpainting, as it is essentially a scene completion process using partial observations. As demonstrated by the success of data-driven image inpainting algorithms [9, 11, 21, 22], estimating $F$ and $B$ is a more convenient task for deep networks than estimating $\alpha$ directly. We demonstrate through quantitative and qualitative evaluations that the matting performance can be substantially increased by first estimating the layer colors even when standard network architectures that have been originally designed for inpainting and direct $\alpha$ estimation are utilized. We also provide an extended data augmentation procedure for natural matting and demonstrate the effectiveness of each design choice through ablation studies.

Our approach has conceptual parallels with sampling-based matting methods, a prominent approach to matting where two color samples are selected from the foreground and background for each pixel to get an initial matte estimate using (1). Existing work in sampling-based matting typically selects a color pair by using the color line assumption and other metrics from spatial proximity of the samples among others. The inherent shortcoming of this approach is the lack of consideration of image structure and texture during the sample selection process. In contrast, the deep networks we train for this sample-selection process are able to make use of the color and texture characteristics of the whole image, and as a result, provide a much more reliable $F$ and $B$ estimations that is critical for the $\alpha$ estimation.

## 2. Related Work

Natural matting techniques are generally classified as affinity, sampling, and learning-based methods.

**Affinity-based** methods work by propagating opacity information from known pixels to unknown pixels according to some similarity measure, that is often formulated as a function of spatial proximity and color similarity. Seminal work by Levin et al. [12] introduced a closed-form solution to matting, where they compute affinity between two pixels by utilizing local color statistics. Later work represents every unknown pixel as a linear combination of a set of its non-local neighbors [4]. Likewise, KNN matting [3] enforces every pixel and their non-local neighbors to have similar alpha values. The information-flow matting method [1] showed that high-quality mattes can be produced by combining local and non-local affinities. While affinity-based methods can produce successful mattes, they typically suffer from high computational complexity and memory issues.

**Sampling-based** methods initially gather a set of foreground and background color samples from the regions marked as such in the trimap, and then select the best foreground-background color pair from them for each pixel. Given an unknown pixel, Robust Matting [19] favors sampling from known foreground and background regions that are spatially close. Shared Matting [7] recognizes that true samples may lie farther ahead, and gathers its samples from the trimap boundaries between the known and unknown regions. In contrast, Global Matting [8] samples all pixels at the trimap boundary, which decreases the probability of missing a true sample, but also increases the number of samples making the subsequent sample selection more expensive. Comprehensive Sampling [16] does not restrict the sampling domain to trimap boundaries, but instead varies sampling distance as a function of the proximity of the unknown pixel to known trimap regions. Subsequent work [10] proposes sparse sampling by assessing similarity at super-pixel level making use of a KL-Divergence based distance measure. These methods aim to find two single samples for each pixel in the unknown-opacity region. However, a recent analysis [2] shows that selecting multiple samples, even through a simple KNN search, outperforms sampling-based algorithms in a majority of natural scenes. Hence, recent work [6] using a multitude of sample pairs in a sparse coding framework shows favorable performance.

Notice that none of these strategies make use of the textural patterns observed in the original image or the high spatial correlation between neighboring pixels. Our method exploits this spatial correlation by estimating the per-pixel samples using the textures that exist in the foreground and background regions through the use of neural networks. We show that the foreground and background samples estimated by our method are vastly of higher-quality than the previous approaches to sampling-based matting.

Following the estimation of alpha values, sampling-based matting methods require an additional step for enforcing spatial smoothness in the alpha matte. This can be done by using extensions of affinity-based methods as proposed by Gastal et al. [7] or Aksoy et al. [1]. Instead of directly solving (1) for $\alpha$, which is the main cause of spatial smoothness issues, we use a final network that produces the alpha mattes which takes the foreground and background color samples as input.

**Learning-based** matting has recently received considerable attention. Early work in this direction explored combining alpha estimations from previous methods with the goal of improving overall matte quality [5]. Other authors presented results within restricted problem domains such as portrait matting [17]. With the introduction of a large-scale matting dataset by Xu et al. [20], different network architectures have been proposed [20, 14] that are purely data-driven. Our method is also data-driven, but in contrast to previous approaches, we decompose the matting problem into sub-problems that are easier for a neural network to learn in a similar way to sampling-based methods.

## 3. Method

The main idea behind our matting approach is to deconstruct the solution of (1) by first estimating the foreground and background color parameters in the equation, and by doing so reducing the difficulty of estimating the corresponding alpha value. Hence, as the starting point in our workflow, we need to reliably estimate the foreground and background colors, whose weighted combination by an unknown alpha value yields to the colors of the observed image.

As a consequence of the definition of the matting problem, the foreground and background have different characteristics. The background image can be thought as a fully opaque layer that is partially obscured by a foreground object. In other words, we may think as if behind the foreground there is a background image that has consistent structural and textural properties. In contrast, the foreground layer is spatially limited to the extent of non-opaque regions. Which means that the colors that participate in the mixtures with the background can be expected to have similar color distributions to the fully opaque pixels, but the structural and textural characteristics differ greatly in partial-opacity regions.

Following this observation, instead of estimating the foreground and background color pairs directly in a single shot for each pixel like previous sampling-based approaches, we first estimate the background image using its consistent structure, and then estimate the foreground colors using our background estimates. We then use our foreground and background color estimations as an input to a

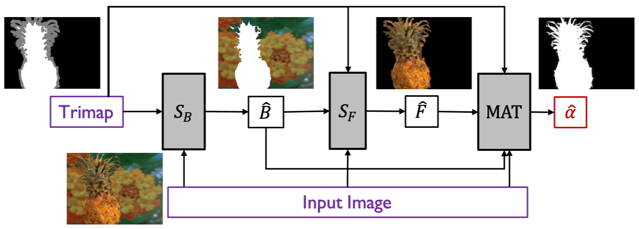

Figure 1. Our pipeline consisting of background $S_B$ and foreground $S_F$ sampling networks and a subsequent matting network MAT that utilizes the estimates from the sampling network pair.

network for the final estimation together with the input image and the trimap. Our pipeline is shown in Figure 1. We detail our approach to the estimation of each unknown in the compositing equation in this section.

## 3.1. Background sampling

It is easily noticeable that in particular, the background sampling has similarities with another problem in computer vision: image inpainting. Image inpainting aims to *hallucinate* a missing part of the image using the consistent structures and high-level understanding of the overall image. The biggest difference between the two problems is that in our case, the background image is partially observed in the input image hidden behind the partially opaque foreground with an unknown alpha matte.

Due to this similarity, we adopt a network architecture that has been originally designed for image inpainting and make several modifications to better fit the neural network to our purposes. We selected the state-of-the-art inpainting approach by Yu et al. [22] for this purpose. Note that we do not claim any novelty in the network design, and alternative architectures can potentially be adopted for this purpose with similar modifications.

We need to keep the network from hallucinating plausible structures in the unknown region, and instead train it to recognize the background structures that is hidden behind the foreground and estimate the samples accordingly. We achieve this by providing the input image as the input instead of only proving the fully transparent regions defined by the trimap, and define our loss function over only the unknown-opacity region:

$$\mathcal{L}_B = \frac{1}{|U|} \sum_{i \in U} |\hat{B}_i - B_i|, \tag{2}$$

where $\hat{B}$ and $B$ denote the predicted and ground-truth background colors, and $U$ is the image region labeled as unknown in the input trimap.

This seemingly small modification results in vastly different results as shown in Figure 2. With the additional input, the network in the end learns not to hallucinate regions but to use the partially obstructed background regions as a

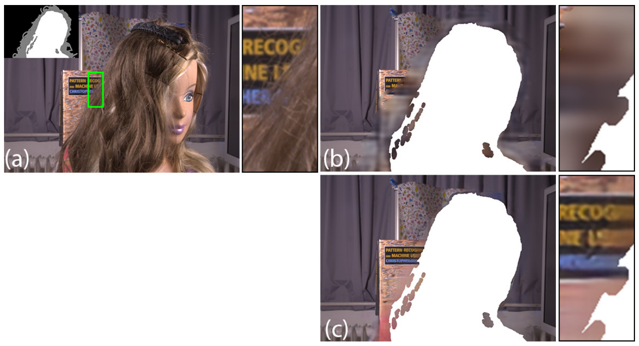

Figure 2. Predicted background colors of input image (a), by generic inpainting (b) versus the background sampling approach where the input image guides the image completion (c).

guide to estimate the high-frequency details. This difference is crucial for the purpose of sampling-based matting, as the color values for the background directly affects the matte quality through its use in the compositing equation.

Our loss function only includes the unknown-opacity region, as the background colors are invisible, and hence irrelevant, in the fully-opaque foreground regions.

## 3.2. Foreground sampling

Once the background samples are estimated, the foreground sampling problem boils down to choosing plausible colors from the known fully-opaque foreground regions that best represent the color mixtures observed in the input image. We adopt the same network architecture we used for background sampling, with a yet another additional input to the system, this time the background samples from the previous step.

Our loss function definition is composed of two terms in the foreground sampling case. The first term is the $L^1$ loss analogous to background sampling:

$$\mathcal{L}_{fg} = \frac{1}{|U|} \sum_{i \in U} |\hat{F}_i - F_i|. \tag{3}$$

As the second term, we introduce a *compositional loss* that penalizes deviations of an intermediate composite image by using (1) with predicted background and foreground colors, and ground-truth alpha mattes from the reference composite input image $I$:

$$\mathcal{L}_{comp} = \frac{1}{|U|} \sum_{i \in U} |\alpha_i \hat{F}_i + (1 - \alpha_i)\hat{B}_i - I_i|. \tag{4}$$

Note that we are making use of the ground-truth alpha values during training this network through the compositional loss. The alpha matte is not needed when estimating the foreground samples in the final system in forward passes.

The overall loss is defined simply as the sum of the $L^1$ and compositional losses:

$$\mathcal{L}_F = \mathcal{L}_{fg} + \mathcal{L}_{comp}. \tag{5}$$

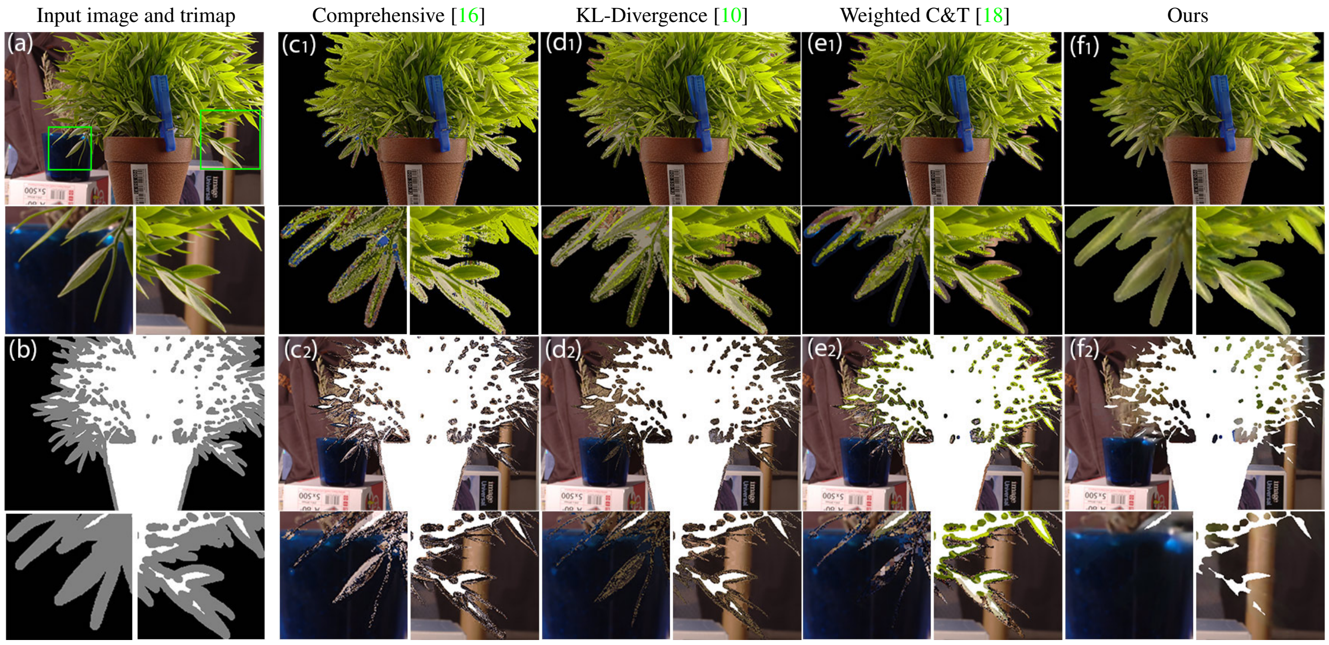

Figure 3. Comparison of color samples between current sampling-based methods and ours. In foreground color predictions ($c_1$) - ($f_1$), the foreground region of trimap (b) remains the same as the input image (a) while the background region is masked for presentation. Analogously we mask the foreground region while presenting background color predictions ($c_2$) - ($f_2$). Label expansion algorithm used in these methods to expand the unknown region of the trimap is disabled in order to have a fair comparison.

## 3.3. Generating the alpha matte

Figure 3 shows the foreground and background samples estimated by our method as well as by prominent sampling-based approaches in the literature. It is easily observed that our method is able to generate spatially smooth color predictions that match well with the actual colors that form the mixtures, while previous methods generate noisy sampling results. More importantly, due to the common use of the color line model in order to select samples, the competing methods incorrectly choose foreground samples that are very similar to the actual background color in transparent regions and vice versa. This results in low-quality alpha predictions as we show later in the experimental analysis (Section 4). While our method generates reliable color predictions, we observed that in order to generate a high-quality matte estimation, we need a neural network that takes the estimated colors as input and generates the matte instead of directly solving the compositing equation for the alpha values.

For this purpose, in our prototype implementation, we adopt a generative architecture proposed by Lutz et al. [14], but it should be noted that any architecture designed for alpha matting can replace the network of our choice at this step. The image, trimap and the predicted foreground and background colors are given to the network as input, and we propose a new loss function as explained below.

Alpha prediction loss $\mathcal{L}_{alpha}$ and compositional loss $\mathcal{L}_{comp}$ as used in Deep Matting [20] are both used as a part of the loss function $\mathcal{L}_{dm}$.

$$\mathcal{L}_{dm} = \frac{1}{2|U|}\sum_{i\in U}|\hat{\alpha}_i - \alpha_i| + \frac{1}{2|U|}\sum_{i\in U}|\hat{I}_i - I_i| \tag{6}$$

where $\hat{\alpha}$ and $\alpha_i$ are the predicted and ground-truth alpha matte respectively. $\hat{I}$ is the composited RGB image defined as $\hat{I} = \hat{\alpha}F + (1-\hat{\alpha})B$ with $F$ and $B$ being the ground-truth background and foreground colors. $I$ is the input RGB image.

However, as noted by Aksoy et al. [1] and Lutz et al. [14], these loss functions fail to properly promote sharpness in the final alpha mattes and as a result, produce blurred results. To address this issue, we use a loss defined over alpha gradients defined as the $L^1$ distance between the spatial gradient of predicted and ground-truth alpha mattes:

$$\mathcal{L}_{grad} = \frac{1}{|U|}\sum_{i\in U}|\hat{G}_i - G_i|, \tag{7}$$

where $\hat{G}$ and $G$ denote the gradient magnitude of predicted and ground-truth alpha mattes respectively. Our final matting loss is then defined as:

$$\mathcal{L}_{matting} = \mathcal{L}_{dm} + \mathcal{L}_{grad}. \tag{8}$$

This loss promotes sharp results that effectively capture high-frequency alpha changes. We investigate the effect of the gradient term in our ablation study in Section 4.4.

## 4. Results and Discussion

We discuss our extensive data augmentation scheme in Section 4.1, provide details about our network implementations in Section 4.2, present an ablation study in Section 4.3, show quantitative and qualitative evaluations in Section 4.4, and finally show compositing examples in Section 4.5.

### 4.1. Dataset augmentation

We use the 431 unique foreground images and corresponding alpha mattes in Adobe matting dataset [20] for training. In contrast to Xu et al. [20], where the authors generate the training dataset by compositing each foreground with 100 fixed backgrounds beforehand, we use a dynamic data augmentation scheme where we create new composite images on the fly by many randomizations, as described below.

Firstly, we extend the number of distinct ground-truth mattes by compositing two random foreground images on top of each other with a probability of 0.5. The corresponding trimaps are then defined, again on the fly, by dilating the foreground by a random number of pixels ranging from 1 to 19. Random horizontal flipping is then applied on each training input with probability 0.5. This creates greater variability in terms of ground-truth mattes and trimaps, and a better generalizability as images with sharp opacity boundaries and large transparent regions are randomly combined to create hybrid scenes.

After selecting and mixing the foreground images, we select 8 random background images from the MS COCO dataset [13] and create composite images, which are fed to the network as a batch. Using the same foreground with different backgrounds in a batch increases the invariance of the matte estimation with respect to the background image. Selecting the background images randomly at each iteration instead of defining a pre-determined set results in the network seeing completely new input images throughout the training process, which also helps with the generalizability of the final network.

Finally, we apply random scaling to the input images before feeding them to the network. In addition to cropping $320 \times 320$, $480 \times 480$, or $640 \times 640$ patches from the original 1080p images similar to Lutz et al. [14] and Xu et al. [20], we randomly resize the image to $480 \times 480$ or $640 \times 640$ before cropping the patches. This way, the network does not only see zoomed-in versions of the input images, but also input images as a whole. It is therefore able to better generalize in terms of the image scale.

We show the effectiveness of our data augmentation strategy in the ablation study in Section 4.3.

During the training of foreground sampling network, we utilize an additional preprocessing step to further enlarge the foreground set, where each foreground image is enhanced before composition by randomly changing its brightness, contrast and saturation within the range $[0.5, 1.5]$ according to a truncated Gaussian distribution with $\mu = 1, \sigma = 0.2$. The hue of the foreground image is also randomly shifted within the range $[-0.2, 0.2]$ according to another truncated Gaussian distribution with $\mu = 0, \sigma = 0.1$.

### 4.2. Implementation details

The training processes of the foreground and background sampling networks start from pretrained model provided by Yu et al. [22] on *Places2* [23] dataset. To be consistent with the pretrained model, we use input size of $256 \times 256$. Therefore, the preprocessing crops the original image into sizes of $256 \times 256$, $384 \times 384$ and $512 \times 512$ and resize back to size $256 \times 256$.

The learning rates used in all trainings are all set to $10^{-5}$ and remain constant. We use Adam optimizer with $\beta_1 = 0.9$ and $\beta_2 = 0.999$ for back propagation. Network architectures of the sampling and matting networks are further detailed in the supplementary material.

### 4.3. Ablation study

In order to investigate the effects of various components we introduced in previous sections we performed a series of ablation studies. To this end we benchmarked various configurations of our method on the Adobe dataset [20], where we selectively deactivated the training dataset augmentation as discussed in Section 4.1 (*Aug*), excluded the background (*BG*) and foreground (*FG*) color predictions from the inputs to our final matting network, and ignored the gradient term ($\mathcal{L}_{grad}$) from Section 3.3 in our training loss. Additionally, in order to test if the use of background and foreground color predictions improves matting quality consistently across different matting network architectures, we ran tests using both our original architecture as described in Section 3.3 (*AlphaGAN*) as well as the Deep Matting architecture from Xu et al. [20] (*DM*). The results are shown in Table 1.

Our ablation study shows that the quality of alpha mattes can be significantly improved according to both MSE and SAD metrics using background color predictions (b,c).

Table 1. Results of our ablation study. See text for details.

| Cfg | Aug. | BG | FG | Arch. | $\mathcal{L}_{grad}$ | SAD | MSE |
|---|---|---|---|---|---|---|---|
| (b) | ✓ | | | AlphaGAN | ✓ | 47.13 | 0.0146 |
| (c) | ✓ | ✓ | | AlphaGAN | ✓ | 39.91 | 0.0115 |
| (d) | | ✓ | ✓ | AlphaGAN | ✓ | 42.70 | 0.0126 |
| (e) | ✓ | ✓ | ✓ | AlphaGAN | | 40.26 | 0.0107 |
| (f) | ✓ | ✓ | ✓ | AlphaGAN | ✓ | 40.35 | 0.0099 |
| | ✓ | | | DM | ✓ | 53.56 | 0.0187 |
| | ✓ | ✓ | | DM | ✓ | 45.47 | 0.0131 |
| | ✓ | ✓ | ✓ | DM | ✓ | 42.01 | 0.0114 |

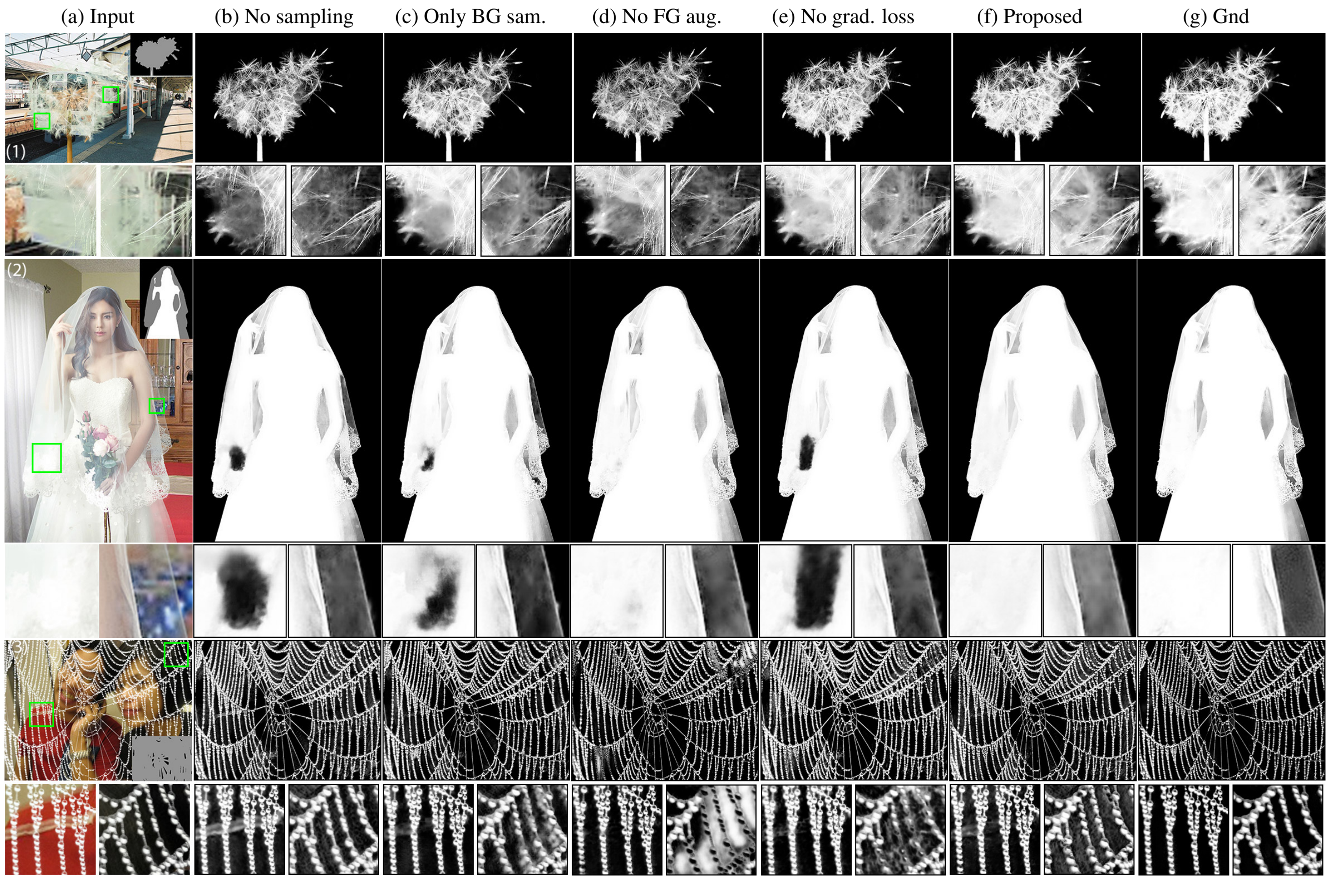

Figure 4. Some examples from Adobe dataset [20] along with corresponding trimaps are shown in (a) for our ablation study. Alpha mattes are computed from models using configurations from Table 1, i.e. model that does not use sampling networks (b), only use background sampling network (c), does not augment foreground set (d), does not use alpha gradient loss (e) and our best model (f). Corresponding ground-truth alpha mattes (g) are also shown. All models use the same network architecture described in Section 4.2.

Additionally using foreground color predictions results in a more modest MSE improvement over using solely background color predictions (c,f). The training dataset augmentation (Section 4.1) makes a clear difference in both metrics (d,f). Using the gradient loss term results in a slightly better MSE score (e,f). This first part of our ablation shows the merit of each component we tested, and thus we use configuration (f) as our final method.

A second important conclusion we make from the last three rows of Table 1 is that even when we change the final matting network architecture, utilizing the background and foreground color predictions still results in significant improvements in the output matte quality. This suggests that our core idea of using sampling networks in the context of matting has merit independent of the specific network architectures used for matting. For further investigation, in Figure 4 we provide examples for a visual comparison of the various configurations from Table 1.

Comparisons of configurations (b), (c) and (f) on all three examples show that noticeable errors can be avoided by using background or both background and foreground color predictions. The improvement is particularly clear in the region where foreground colors are similar to background colors because without predicted colors' help, the matting network can easily confuse background and foreground. Using alpha gradient loss has the effect of eliminating undesirable blurriness, as we observe by comparing configurations (e) and (f) in example (3). Alpha gradient loss can also help with capturing image structure and avoid erroneously creating a hole in the alpha matte, which can be seen in example (2). These are just some examples of matte quality improvements that may not be evident from quantitative evaluation, but nevertheless provides additional motivation for choosing configuration (f) over others.

## 4.4. Comparisons against the state-of-the-art

At the time of this submission, our technique outperforms current methods in the alpha matting benchmark [15]. Table 2 shows the SAD and MSE scores of top-performing methods as well as top-performing sampling-based methods. Our method only takes 0.6 seconds of computation time on average for the images in the benchmark.

Table 2. Our scores in the alpha matting benchmark [15] together with the top-performing published and closely related methods. $S$, $L$ and $U$ denote the three trimap types, small, large and user, included in the benchmark. Bold and blue numbers represent the best scores obtained among methods in the benchmark at the time of submission*.

| | Average Rank | | | | Troll | | | Doll | | | Donkey | | | Elephant | | | Plant | | | Pineapple | | | Plastic bag | | | Net | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *Overall* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* | *S* | *L* | *U* |
| Sum of Absolute Differences | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ours | 3.5 | 2.5 | 2.8 | 5.3 | **9.1** | **9.7** | 9.8 | **4.3** | **4.8** | **5.1** | 3.4 | 3.7 | 3.2 | **0.9** | 1.1 | 2.0 | **5.1** | 6.8 | 9.7 | 2.5 | 4.0 | **3.7** | 18.6 | 19.3 | 19.1 | 20.0 | 21.6 | 23.2 |
| Deep Matting [20] | 5.3 | 6.5 | 4.3 | 5.3 | 10.7 | 11.2 | 11.0 | 4.8 | 5.8 | 5.6 | **2.8** | **2.9** | **2.9** | 1.1 | **1.1** | 2.0 | 6.0 | 7.1 | 8.9 | 2.7 | **3.2** | 3.9 | 19.2 | 19.6 | 18.7 | 21.8 | 23.9 | 24.1 |
| IFM [1] | 6.1 | 7.4 | 5.8 | 5.3 | 10.3 | 11.2 | 12.5 | 5.6 | 7.3 | 7.3 | 3.8 | 4.1 | 3.0 | 1.4 | 2.3 | 2.0 | 5.9 | 7.1 | 8.6 | 3.6 | 5.7 | 4.6 | 18.3 | 19.3 | 15.8 | 20.2 | 22.2 | 22.3 |
| AlphaGAN [14] | 8.7 | 9.5 | 8.4 | 8.1 | 9.6 | 10.7 | 10.4 | 4.7 | 5.3 | 5.4 | 3.1 | 3.7 | 3.1 | 1.1 | 1.3 | 2.0 | 6.4 | 8.3 | 9.3 | 3.6 | 5.0 | 4.3 | 20.8 | 21.5 | 20.6 | 25.7 | 28.7 | 26.7 |
| KL-Divergence [10] | 18.5 | 17.5 | 17.5 | 20.4 | 11.6 | 17.5 | 14.7 | 5.6 | 8.5 | 8.0 | 4.9 | 5.3 | 3.7 | 1.5 | 3.5 | 2.1 | 5.8 | 8.3 | 14.1 | 5.6 | 9.3 | 8.0 | 24.6 | 27.7 | 28.9 | 20.7 | 22.7 | 23.9 |
| Comprehensive [16] | 20.5 | 17.8 | 20.4 | 23.4 | 11.2 | 18.5 | 14.8 | 6.5 | 9.5 | 8.9 | 4.5 | 4.9 | 4.1 | 1.7 | 3.1 | 2.3 | 5.4 | 9.8 | 13.4 | 5.5 | 11.5 | 7.4 | 23.9 | 22.0 | 22.8 | 23.8 | 28.0 | 28.1 |
| Mean Squared Error | | | | | | | | | | | | | | | | | | | | | | | | | | | | |
| Ours | 5.0 | 2.9 | 5.0 | 7.1 | **0.3** | **0.3** | **0.3** | **0.1** | **0.2** | **0.2** | 0.2 | 0.2 | 0.2 | **0.0** | **0.0** | 0.1 | 0.4 | 0.6 | 1.2 | **0.1** | 0.3 | 0.3 | 1.1 | 1.1 | 1.2 | 0.7 | 0.8 | 0.8 |
| Deep Matting [20] | 7.5 | 6.4 | 6.5 | 9.6 | 0.4 | 0.4 | 0.4 | 0.2 | 0.3 | 0.3 | **0.1** | **0.1** | **0.2** | 0.0 | 0.0 | 0.2 | 0.5 | 0.6 | 1.0 | 0.2 | **0.2** | 0.4 | 1.1 | **1.1** | 1.1 | 0.8 | 0.9 | 1.0 |
| IFM [1] | 7.6 | 9.8 | 6.6 | 6.5 | 0.3 | 0.4 | 0.5 | 0.3 | 0.4 | 0.5 | 0.3 | 0.3 | 0.2 | 0.1 | 0.1 | 0.1 | 0.4 | 0.4 | **0.6** | 0.2 | 0.3 | 0.3 | 1.3 | 1.2 | 0.8 | 0.8 | 0.8 | 0.9 |
| AlphaGAN [14] | 11.4 | 11.9 | 12.1 | 10.3 | 0.3 | 0.3 | 0.4 | 0.2 | 0.2 | 0.3 | 0.2 | 0.2 | 0.2 | 0.0 | 0.0 | 0.1 | 0.6 | 0.9 | 1.1 | 0.3 | 0.5 | 0.4 | 1.3 | 1.3 | 1.2 | 1.1 | 1.4 | 1.1 |
| KL-Divergence [10] | 18.0 | 17.4 | 16.9 | 19.8 | 0.4 | 0.9 | 0.7 | 0.3 | 0.5 | 0.5 | 0.3 | 0.4 | 0.3 | 0.1 | 0.2 | 0.1 | 0.4 | 0.4 | 1.2 | 0.4 | 0.6 | 0.6 | 1.7 | 2.0 | 2.1 | 0.8 | 0.8 | 0.9 |
| Comprehensive [16] | 19.7 | 18.3 | 19.6 | 21.1 | 0.4 | 1.2 | 0.8 | 0.3 | 0.6 | 0.6 | 0.3 | 0.3 | 0.3 | 0.1 | 0.2 | 0.1 | **0.2** | 0.5 | 0.9 | 0.3 | 0.9 | 0.5 | 1.6 | 1.5 | 1.5 | 1.0 | 1.2 | 1.3 |

* Some columns do not have a bold number when the best-scoring algorithm for that particular image-trimap pair is not among the top-ranking methods included here.

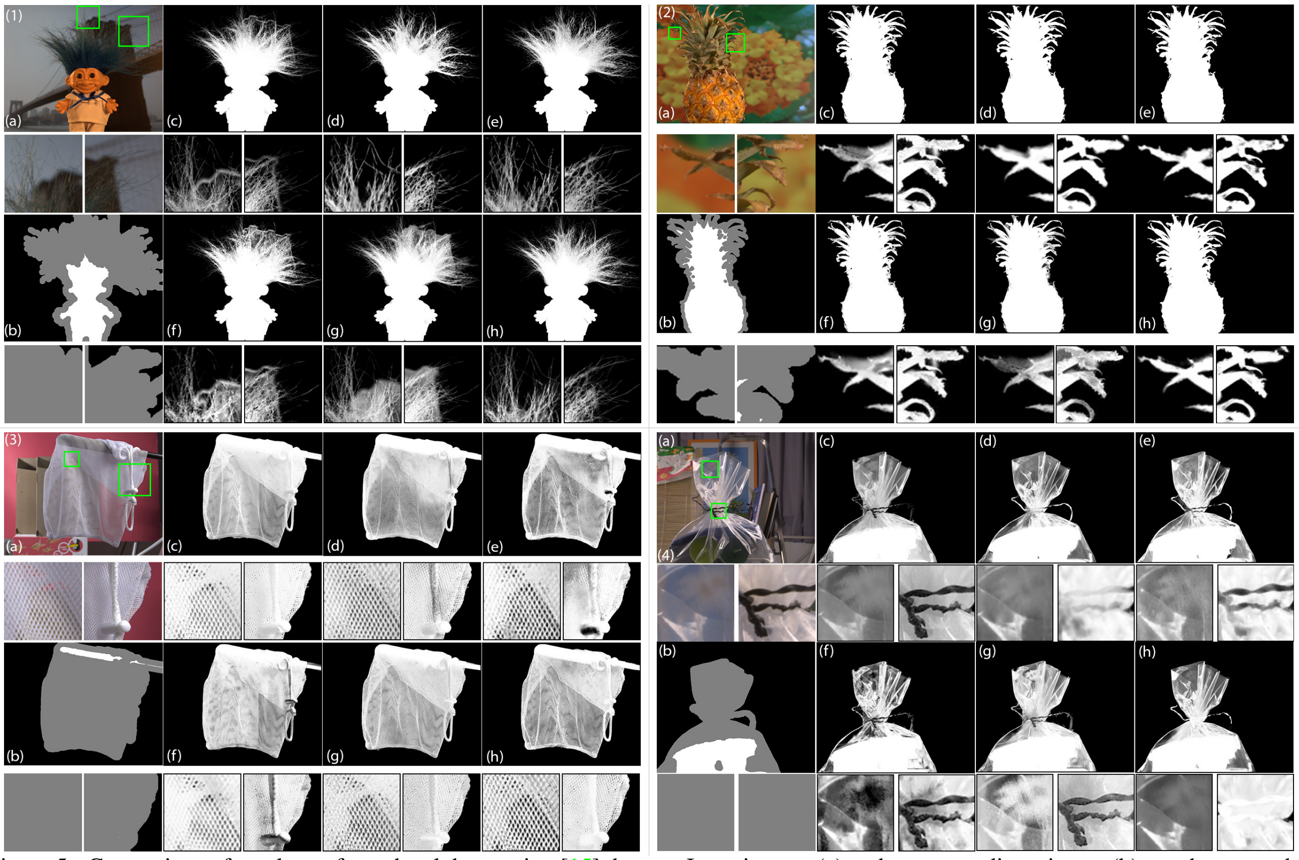

Figure 5. Comparison of results on from the alpha matting [15] dataset. Input images (a) and corresponding trimaps (b) are shown on the left. Alpha mattes are computed from Information-flow Matting [1] (c), Deep Matting [20] (d), AlphaGAN [14] (e), KL-Divergence Based Sparse Sampling [10] (f) and Comprehensive Sampling [16] (g) and our model (h).

We also report lower errors on Adobe dataset (Table 1) compared to both [14] and [20]. However we note that due to potential differences in the trimap generation process the numbers from Adobe dataset may not be directly comparable.

We present qualitative comparisons between our method and the state-of-the-art in natural matting as well as related methods using sampling-based approaches in Figure 5. The hand-crafted sampling-based strategies fail to give satisfactory results in most of the examples, while the affinity-based

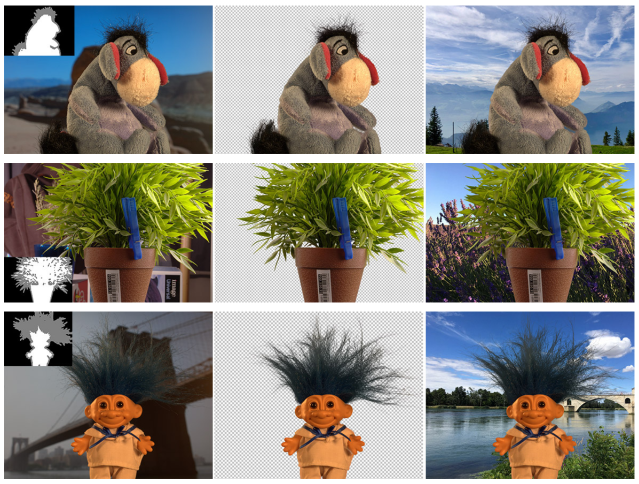
Figure 6. Basic compositing examples using foreground colors and alpha mattes predicted by our method.

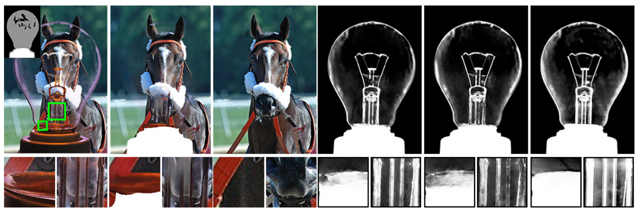

Figure 7. Example of erroneous color predictions affecting alpha matte prediction. Input image and trimaps are shown in (a). Alpha mattes are computed from the model that deactivates sampling networks (d) and the one uses both sampling networks (e), i.e. our best model. Comparing with ground-truth alpha matte (f), the alpha matte is worse in (e), especially in the regions shown below the images. This is due to the erroneous background color predictions (b) in these regions, which can be seen clearly when comparing to ground-truth background colors (c).

information-flow matting [1] has problems when the background or foreground colors in the unknown region does not clearly appear in the known regions of the trimap, as the troll and plastic bag examples demonstrates. Deep matting [20] suffers from smoothness issues (pineapple and troll), and both deep matting and AlphaGAN shows erroneous low-alpha values in high-transparency images (net, plastic bag and pineapple). We are able to fix these issues in the previous methods and create reliable estimates even when the colors in the unknown region are not seen in the known regions, such as the black knot in the plastic bag example. This robustness demonstrates the effectiveness of our sampling-based approached combined with the representational power of the neural networks.

### 4.5. Compositing

In order to use an alpha matte for compositing, the foreground layer colors should be estimated. As the natural matting methods in the literature typically aim to only estimate the alpha matte, the layer color estimation is done as a post-processing by specialized methods that take the original image and the estimated alpha matte as input [2, 3, 12]. The foreground colors in our method, however, are already estimated prior to the alpha estimation. Hence, our results can be directly used for compositing without the need of an additional color estimation method. In Figure 6 we present proof-of-concept results for several challenging images using the foreground colors from our sampling network.

## 5. Limitations

In rare cases, it is possible for an erroneous background or foreground color prediction to have an adverse effect on the accuracy of the predicted alpha matte. Figure 7 shows such an example, where including color predictions as inputs to the matting network yields worse results. Training the sampling networks with a more diverse dataset could potentially be helpful with such cases. Fortunately, unlike the ground-truth training data for matting, the data necessary for training the sampling networks can be generated trivially by removing regions of arbitrary images.

## 6. Conclusion

In this work, we proposed to divide the matting problem into sub-problems that are more convenient for neural networks. The core idea of our method is to utilize the representational power of neural networks for image texture and spatial coherence for decreasing the number of unknowns in the compositing equation. By using network architectures originally designed for image inpainting, we demonstrated that colors for background and foreground layers can be reliably estimated. We showed through quantitative and qualitative evaluation that the opacity estimation performance can be substantially increased when these estimated layer colors are provided as an additional input to different network architectures for natural matting.

## References

[1] Y. Aksoy, T. O. Aydın, and M. Pollefeys. Designing effective inter-pixel information flow for natural image matting. In *Proc. CVPR*, 2017. 2, 4, 7, 8

[2] Y. Aksoy, T. O. Aydın, and M. Pollefeys. Information-flow matting. `arXiv:1707.05055 [cs.CV]`, 2017. 2, 8

[3] Q. Chen, D. Li, and C.-K. Tang. KNN matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 35(9):2175–2188, 2013. 2, 8

[4] X. Chen, D. Zou, Q. Zhao, and P. Tan. Manifold preserving edit propagation. *ACM Trans. Graph.*, 31(6):132:1–132:7, 2012. 2

[5] D. Cho, Y.-W. Tai, and I. S. Kweon. Natural image matting using deep convolutional neural networks. In *Proc. ECCV*, 2016. 1, 2

[6] X. Feng, X. Liang, and Z. Zhang. A cluster sampling method for image matting via sparse coding. In *Proc. ECCV*, 2016. 2

[7] E. S. Gastal and M. M. Oliveira. Shared sampling for real-time alpha matting. In *Computer Graphics Forum*, volume 29, pages 575–584. Wiley Online Library, 2010. 2

[8] K. He, C. Rhemann, C. Rother, X. Tang, and J. Sun. A global sampling method for alpha matting. 2011. 2

[9] S. Iizuka, E. Simo-Serra, and H. Ishikawa. Globally and locally consistent image completion. *ACM Trans. Graph.*, 36(4):107:1–107:14, 2017. 1

[10] L. Karacan, A. Erdem, and E. Erdem. Image matting with KL-divergence based sparse sampling. In *Proc. ICCV*, 2015. 2, 4, 7

[11] R. Köhler, C. Schuler, B. Schölkopf, and S. Harmeling. Mask-specific inpainting with deep neural networks. In *Proc. GCPR*, 2014. 1

[12] A. Levin, D. Lischinski, and Y. Weiss. A closed-form solution to natural image matting. *IEEE Trans. Pattern Anal. Mach. Intell.*, 30(2):228–242, 2008. 2, 8

[13] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014. 5

[14] S. Lutz, K. Amplianitis, and A. Smolic. Alphagan: Generative adversarial networks for natural image matting. *arXiv preprint arXiv:1807.10088*, 2018. 1, 2, 4, 5, 7

[15] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott. A perceptually motivated online benchmark for image matting. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 1826–1833. IEEE, 2009. 6, 7

[16] E. Shahrian, D. Rajan, B. Price, and S. Cohen. Improving image matting using comprehensive sampling sets. In *Proc. CVPR*, 2013. 2, 4, 7

[17] X. Shen, X. Tao, H. Gao, C. Zhou, and J. Jia. Deep automatic portrait matting. In *Proc. ECCV*, 2016. 2

[18] E. S. Varnousfaderani and D. Rajan. Weighted color and texture sample selection for image matting. *IEEE Transactions on Image Processing*, 22(11):4260–4270, 2013. 4

[19] J. Wang and M. F. Cohen. Optimized color sampling for robust matting. In *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, pages 1–8. IEEE, 2007. 2

[20] N. Xu, B. Price, S. Cohen, and T. Huang. Deep image matting. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 1, 2, 4, 5, 6, 7, 8

[21] C. Yang, X. Lu, Z. Lin, E. Shechtman, O. Wang, and H. Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proc. CVPR*, 2017. 1

[22] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang. Generative image inpainting with contextual attention. *arXiv preprint*, 2018. 1, 3, 5

[23] B. Zhou, A. Lapedriza, A. Khosla, A. Oliva, and A. Torralba. Places: A 10 million image database for scene recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1452–1464, 2018. 5