Abstract

In this paper, we propose a transformer-based image matting model called MatteFormer, which takes full advantage of trimap information in the transformer block. Our method first introduces a prior-token which is a global representation of each trimap region (e.g. foreground, background and unknown). These prior-tokens are used as global priors and participate in the self-attention mechanism of each block. Each stage of the encoder is composed of PAST (Prior-Attentive Swin Transformer) block, which is based on the Swin Transformer block, but differs in a couple of aspects: 1) It has PA-WSA (Prior-Attentive Window Self-Attention) layer, performing self-attention not only with spatial-tokens but also with prior-tokens. 2) It has prior-memory which saves prior-tokens accumulatively from the previous blocks and transfers them to the next block. We evaluate our MatteFormer on the commonly used image matting datasets: Composition-1k and Distinctions-646. Experiment results show that our proposed method achieves state-of-the-art performance with a large margin. Our codes are available at https://github.com/webtoon/matteformer.

1. Introduction

Image matting is one of the most fundamental tasks in computer vision which is mainly used to separate a foreground object precisely for the purpose of image editing and compositing. Especially, the foreground not only includes complex objects like human hair and animal fur but also includes transparent objects like glass, bulb and water. Natural image can be represented as a linear combination of foreground $F \in \mathbb{R}^{H \times W \times C}$ and background $B \in \mathbb{R}^{H \times W \times C}$ with alpha matte $\alpha \in \mathbb{R}^{H \times W}$ as follows:

$$I_i = \alpha_i F_i + (1 - \alpha_i) B_i, \quad \alpha_i \in [0, 1],$$

where $H, W$ and $C$ denotes the height, the width and the number of channels (3 for a color image) respectively, and $i \in [HW]$ denotes the pixel index.

In image matting, estimating opacity value $\alpha$ given only observed image $I$, is a highly ill-posed problem if any extra information is not available. Thus, in many works, various types of additional user-inputs (e.g. trimap, scribble, binary mask, background image, etc.) are used, among which, especially trimap is the most common. Trimap is cost-intensive for users to draw but provide high-quality information about global context such as region information about foreground, background and unknown pixels. Consequently, it is natural to design a model to take a full advantage of this user-input and many works utilizing trimaps have been developed for image matting, most of which are based on convolutional neural networks (CNNs).

While CNNs have been very successful in computer vision tasks, for natural language processing (NLP) tasks, transformers earned great success and recently there have been many attempts to introduce transformers in downstream vision tasks as an alternative to CNNs. The seminal work of [11] proposed Vision Transformer (ViT) and showed impressive performances compare to CNN-based models, demonstrating potential on vision tasks. However, implementing the global self-attention in ViT needs high computational cost; it is quadratic to the number of patches.
To overcome this limitation, several general-purpose transformer backbones [10, 30, 52] have been proposed by reducing computational complexity with local self-attention methods. For example, [30] proposed a hierarchical transformer structure by introducing the self-attention within local windows to enable linear cost to the input size and the patch merging layer to reduce the number of spatial-tokens in deeper layers. Besides, they proposed the shifting window scheme to exchange information through neighboring windows. Unfortunately, since the shifting window technique slowly enlarges the receptive field, it is still hard to achieve a sufficiently large receptive field, especially in lower layers.

In this paper, we propose a transformer-based image matting model, named MatteFormer. We first define a prior-token, which represents global context feature of each trimap region; the foreground, background and unknown region as shown in Fig. 1. These prior-tokens are used as global priors and participate in the self-attention mechanism of each block. The encoder stage is composed of the PAST (Prior-Attentive Swin Transformer) blocks, which are based on the Swin Transformer block [30]. However, our PAST block is different from the Swin Transformer block in two aspects. First, it has the PA-WSA (Prior-Attentive Window Self-Attention) layer, where self-attention is computed not only with spatial-tokens but also with prior-tokens as shown in Fig. 1. Second, we introduce the prior-memory, memorizing all prior-tokens generated at each block. Through this, prior-tokens from the previous block can be utilized in the PA-WSA layer of the next block. We evaluate our MatteFormer on Composition-1k and Distinctions-646, which are the commonly used datasets in image matting. The results show that our method achieves state-of-the-art performance. We also conduct some extensive studies about the effectiveness of prior-tokens, visualization of self-attention maps, usage of ASPP, and the computational cost.

In short, our contributions can be summarized as follows:

- We propose MatteFormer, the first transformer-based architecture for the image matting problem.
- We introduce prior-tokens which imply global information of each trimap region (foreground, background and unknown) and use them as global priors in our proposed network.
- We design the PAST (Prior-Attentive Swin Transformer) block, a variant of the Swin Transformer block, which includes the PA-WSA (Prior-Attentive Window Self-Attention) layer and the prior-memory.
- We evaluate MatteFormer on Composition-1k and Distinctions-646, showing that our method achieves state-of-the-art performance with a large margin.

2. Related Works

2.1. Natural Image Matting

Traditional methods. As natural image matting is considered as an ill-posed problem, most of matting algorithms make use of user-input as an additional prior. The trimap is frequently used which provides information about foreground, background and unknown regions. Traditional methods predict the alpha matte by primarily utilizing color features. They are mainly divided into sampling-based and propagation-based methods, according to the way of using the color features.

Sampling-based methods [8, 13, 15, 37, 45] make use of color similarities between unknown regions and known (foreground and background) regions to estimate the alpha matte with foreground and background color statistics. On the other hand, propagation-based methods [6, 16, 19–21, 38], also known as affinity-based methods, propagate the alpha values from a known region (foreground and background) to the unknown region to estimate alpha matte via the affinities of neighboring pixels.

Learning-based methods. Since traditional approaches highly depend on color features, these may produce artifacts due to lack of semantic information. Like deep-learning has achieved great success on many other vision tasks, image matting performance has also been significantly improved through CNNs. Meanwhile, because drawing a trimap requires expensive labor cost, approaches without trimap also have been studied.

For the trimap-based methods, [51] proposed a two-stage architecture and released the Composition-1K dataset. [33] utilized a GAN (generative adversarial network) framework to improve performance. [40] mixed the sampling-based method with deep learning method. [18] designed two encoders for local feature and global context, estimating both foreground and alpha matte. [31] proposed an index-guided encoder-decoder structure with the concept of learning to index. [22] developed a guided contextual attention module propagating high-level opacity globally based on low-level affinity. [53] proposed a patch-based method for high-resolution inputs with a module addressing cross-patch dependency and consistency issues between patches. [27] designed the model with the textural compensate path to enhance fine-grained details. [39] used the class of matting patterns and proposed the semantic trimap.

For the trimap-free methods, [34, 57] predicted alpha matte with a single RGB only. [57] proposed structure of two decoders for classifying foreground and background, and fused them in the extra network. [34] designed hierarchical attention structure and proposed Distinctions-646 dataset. [25, 36] proposed the method of using an additional background image instead of the trimap. [54] used binary mask as additional input and proposed a method to
progressively refine the uncertain regions through the decoding process.

Since trimap plays a role as a strong hint, the trimap-based methods generally perform better than the trimap-free methods. However, many algorithms have used the trimap by simply concatenating it with input RGB channels, which do not fully utilize the potential of trimap. In this respect, we follow the trimap-based method and propose method of fully making use of the trimap in networks via prior-tokens.

2.2. Transformer on Vision Tasks

Solely based on self-attention mechanism, transformer [43] has shown great success on NLP (natural language processing) tasks and has been a base structure for most language models. Besides, [11] applied the transformer to the image classification task demonstrating that transformer-based architectures can achieve competitive performance as an alternative to CNN.

With the potential of ViT [11] and its follow-ups [7, 14, 41, 55], researchers have studied applying transformers to typical vision tasks. For example, numerous transformer-based models have been proposed on image classification [10, 23, 28, 30, 35, 42, 48, 49, 56], object detection [1, 3, 30, 47], semantic segmentation [30, 50, 58], image completion [44] and low-level vision tasks [4, 24, 32].

Meanwhile, many researches [10, 30, 46, 49, 52, 56] have been conducted on designing general-purpose transformer backbone for downstream vision tasks. Especially, [10, 30, 52] focus on variation of self-attention for reducing computational cost and on hierarchical structures for extracting multi-scale features. For example, Swin Transformer [30], which is used as our baseline, performs self-attention in a local window with the shifted window scheme allowing for cross-window connections. However, it has a limitation of still-insufficient receptive field, especially in lower layers. Our approach addresses this issue via prior-tokens and achieves state-of-the-art performance on image matting problem.

3. Methodology

3.1. Network Structure

As shown in Fig. 2, our proposed network called MatteFormer has a typical encoder-decoder structure with shortcut connections. Each encoder stage is composed of a PAST (Prior-Attentive Swin Transformer) block which is a variant version of the Swin Transformer block, and the patch merging layer which reduces the number of tokens. Our proposed PAST block compensates for the insufficient receptive field, originating from the self-attention within local windows, via prior-tokens. In the decoder, we use a quite simple structure as used in [54] with a few convolution and upsample layers. Intermediate features in the encoding layers are delivered to the corresponding decoder layer in a direct way by shortcut connections.

3.2. Prior-token

As a trimap includes definite region information, it is natural to design the network to make full use of this powerful hint, for achieving better results. We first generate prior-tokens of three query regions; foreground, background and unknown areas, by averaging all tokens belonging to the corresponding query region. For example, in Fig. 1, the foreground prior-token is the mean feature of all spatial-tokens lying on the foreground area (white trimap region in the figure). Specifically, the prior-token $p^q$ can be formulated as

$$p^q = \frac{1}{N_q} \sum_{i=1}^{N} r_i^q \cdot z_i, \quad q \in \{fg, bg, uk\},$$

where $q$ is the query which takes a value among foreground (fg), background (bg) or unknown (uk), $i$ is the to-
ken index and \( z_i \) denotes the feature of a spatial-token. \( r_i^q \) is a binary value depending on whether the \( i \)-th token is on the corresponding query region (\( r_i^q = 1 \)) or not (\( r_i^q = 0 \)). For example, \( r_i^q = 1 \) if the \( i \)-th spatial-token is in the foreground region. \( N_q = \sum_i^N r_i^q \) is the number of tokens belonging to the region \( q \), indicating how many spatial-tokens are in the corresponding query region. \( N \) denotes the total number of spatial-tokens. For the sake of easy computation, if the area corresponding to a spatial-token has more than one query region, i.e., the area is in the boundary, the spatial-token is allocated to the dominant query region. Finally, a prior-token \( p^q \) is assumed to have informative representation as a global prior for the corresponding query region.

### 3.3. Prior-Attentive Swin Transformer Block

In this work, we use Swin Transformer [30] as our base model, which showed great promise on mainstream vision tasks such as image classification, object detection and semantic segmentation with a hierarchical architecture design and the shifted window scheme. It has a great efficiency with two aspects. One is that local self-attention on non-overlapping windows enables model to cover large image size with efficient computation. The other is that shifted window partitioning allows for cross-window connection with enhancing long-range dependency. However, it has a limitation that shifted windows enlarge the receptive field slowly, which leads to an insufficient attention region, especially in lower layers.

To address the issue, we propose a Prior-Attentive Swin Transformer (PAST) block as shown in Fig. 3. Our PAST block is based on the Swin Transformer block but different from it in two aspects: First, in the self-attention layer, a token can attend not only to spatial-tokens in the local window but also to global features represented by prior-tokens. Specifically, three prior-tokens are used in our method; foreground, background and unknown prior-tokens. Second, it has a prior-memory, which stores prior-tokens generated from the previous block. Accumulated prior-tokens are used in the next block as informative priors when calculating the self-attention.

**Self-attention with prior-tokens.** The detailed structure of our PAST block is as follows: First, prior-tokens generated in Sec. 3.2 are concatenated with spatial-tokens in a local window. With these global prior-tokens and local spatial-tokens, self-attention is performed in the Prior-Attentive Window Self-Attention (PA-WSA) layer with matrix multiplications of queries, keys and values in a multi-head manner. Specifically, the self-attention mechanism is as follows:

\[
\text{Attention}(Q, K, V) = \text{SoftMax}(QK^T \times s + B)V \quad (3)
\]

where \( K, V \in \mathbb{R}^{(M^2+N_p) \times d} \), \( Q \in \mathbb{R}^{M^2 \times d} \) are key, value and query matrices respectively, \( d \) is the feature dimension of key, value and query, \( M \) is the window size, thus \( M^2 \) is the number of tokens in a local window, \( N_p \) is the number of prior-tokens and \( s \) is a scaling factor. \( B \) is a relative position bias slightly modified from [30]. Between spatial-tokens in a local window, relative positions lies on range \([-M+1, M-1]\) both along x- and y-axis. In our case, as we use extra prior-tokens, we set an auxiliary matrix \( \hat{B} \in \mathbb{R}^{(2M-1)^2+N_p} \). Values of \( B \in \mathbb{R}^{M^2 \times (M^2+N_p)} \) are taken from \( \hat{B} \) and the relative position bias \( B \) adjusts the values of attention map according to the relative position. In this way, a query spatial-token can attend to both local spatial-tokens and global prior-tokens.

**Prior-memory.** Prior-tokens generated from different PAST blocks have different representations for the corresponding query region. That is, the prior-tokens from the previous blocks can deliver informative context to the current block. To implement this, we first define a prior-memory. Like local spatial-tokens, prior-tokens sequentially pass through the PA-WSA, normalization and MLP layer. After that, the prior-tokens are appended to the prior-memory. In the next block, accumulated tokens are used as priors in the PA-WSA layer, i.e., the later block has larger \( N_p \) than the former blocks. More specifically, the \( b \)-th block in each of the four stages in Fig. 2 has \( 3 \times b \) prior-tokens in the corresponding prior-memory.

### 3.4. Training Scheme

The total loss function is defined as a weighted sum of three loss functions; \( L_{dl} \) loss, composition loss [51] and Laplacian loss [18], like in [54].

\[
L_{\text{total}} = L_{dl} + L_{\text{comp}} + L_{\text{lap}} \quad (4)
\]

where \( L_{dl} \) is the absolute difference between the ground truth alpha and the predicted alpha. \( L_{\text{comp}} \) indicates the absolute difference between the ground truth image and the composited image, which is calculated from Eq. (1) with the ground truth foreground, background and predicted alpha.
mattes. $L_{lap}$ measures the differences of Laplacian pyramid representations of the alpha maps and captures the local and global difference.

In the decoding process, we use PRM (Progressive Refinement Module) [54] to generate a precise output map as a coarse-to-fine manner. First, the decoder outputs three alpha mattes from different intermediate layers whose output size is 1/8, 1/4 and 1/1 of an input resolution respectively, and then resized to input resolution. Next, through the PRM, outputs are selectively fused and uncertain regions are progressively refined. Specifically, for a current output index $l$, a refined alpha map $\alpha_l \in \mathbb{R}^{H \times W}$ is calculated with raw matting output $\alpha'_l \in \mathbb{R}^{H \times W}$ and self-guidance mask $g_l \in \mathbb{R}^{H \times W}$ as follows:

$$\alpha_l = \alpha'_l \odot g_l + \alpha_{l-1} \odot (1 - g_l),$$  \hspace{1cm} (5)

$$g_l(x, y) = \begin{cases} 1, & \text{if } 0 < \alpha_{l-1}(x, y) < 1 \\ 0, & \text{otherwise} \end{cases},$$  \hspace{1cm} (6)

where $\odot$ denotes element-wise multiplication. The self-guidance mask $g_l$ is obtained from the previous alpha map $\alpha_{l-1}$. It is defined to have 0 if predicted pixel is definite region (foreground or background), and 1 if transparent region. The non-confident pixels in the previous output $\alpha_{l-1}$ are replaced with pixels of the current output $\alpha'_l$ according to $g_l$. Meanwhile, the confident pixels of $\alpha_{l-1}$ are not updated. In this way, confident regions are preserved and the current output can focus only on refining the non-confident region.

Our data augmentation setting is set similar to that of [22] and [54]. We first perform an affine transformation with a random degree, scale and flip. Then we randomly crop both the image and the trimap to a fixed size. After that, random color jittering is applied. Finally, the augmented crop both the image and the trimap to a fixed size. After that, random color jittering is applied. Finally, the augmented

\section{4. Experiments}

In this section, We evaluate our MatteFormer on two public datasets, Composition-1k [51] and Distinctions-646 [34], which are commonly used in the image matting task. We first describe the experimental environments; datasets, evaluation metrics and implementation details. Next, we compare the results of our MatteFormer to those of other state-of-the-art works. Finally, we conduct some ablation studies on our proposed method.

\subsection{4.1. Datasets and Evaluation}

Composition-1k provides unique 50 foreground images with the corresponding ground truth alpha mattes as the test set. Background test images are pre-defined from PASCAL VOC2012 [12]. By compositing the foreground and background images, the number of test samples is 1,000 in total. The train set consists of 431 foreground object images with ground truth alpha mattes. Contrast to the test set, train background images are sampled from MS COCO [26].

Distinctions-646 is comprised of 646 distinct foreground images and it has more versatility and robustness than Composition-1k. Foreground samples are divided into 596 train and 50 test samples. Like Composition-1k, test background samples are pre-defined with PASCAL VOC2012. Foreground images are synthesized with background images following the same composition rule in [51]. Unfortunately, as distinctions-646 does not release official trimaps unlike Composition-1k, a fair comparison with the previous works is hard.

We evaluate our MatteFormer using four major quantitative metrics in image matting: sum of absolute differences (SAD), mean square error (MSE), slope (Grad), and connectivity (Conn). We use official evaluation code provided by [51]. Note that a model with lower metric values can predict more precise alpha mattes.

\subsection{4.2. Implementation Details}

We implement PAST block based on the Swin Transformer block. We introduce prior-tokens to participate in the self-attention mechanism with local spatial-tokens and feed them to other layers in the same way as spatial-tokens. A prior-memory is put on each stage, memorizing prior-tokens from only the PAST blocks in the same stage for simplicity.

Our encoder is first initialized with the Tiny model of Swin Transformer pretrained on ImageNet [9], then trained on the image matting dataset in an end-to-end manner. Since we use a trimap and an RGB image as network input, the number of input channels is 6 which is different from that of the pretrained model. Thus, we bring the weight of the pretrained patch-embedding layer only to the first 3 channels (RGB) of our patch-embedding layer. As the size of our relative position bias table $B$, is bigger than that of the pretrained model due to prior-tokens, we bring the pretrained bias table to the front of our bias table weight. The shortcut layers which deliver the encoder features to the decoder layers are composed of $3 \times 3$ convolutions with normalization layers. As a decoder, we simply followed [54] which use a simple CNN-based structure with $3 \times 3$ convolution layers and upsample layers. Both shortcut and decoder layers are randomly initialized.

When training, we set the network input size to $512 \times 512$ and the batch size to 20 in total on 2 GPUs. The learning rate is initialized to $4 \cdot 10^{-4}$. We use Adam optimizer with $\beta_1 = 0.5$ and $\beta_2 = 0.999$. 

11700
Table 1. Results on Composition-1k test set. * denotes the baseline of comparison which is our reproduction of MG Matting with trimap input. Original MG Matting is based on ResNet-34 and MG Matting-trimap,res50* uses ResNet-50.

4.3. Results on Image Matting Datasets

**Composition-1k.** We first compare our MatteFormer with state-of-the-art models on Composition-1k dataset. Tab. 1 tabulates the quantitative results of recent methods and shows that our method outperforms others achieving a new state-of-the-art performance. We set [54] as our strong baseline for comparison because we follow many of its experimental details. However, in original paper, it use a binary mask as an extra input instead of trimap. For fair comparison, we retrain the same model and more larger model (with ResNet-50 [17] backbone) with a trimap and set them as baselines for comparison (marked as * in Tab. 1). Some reasons we do not take [39] and [29] as our baseline are that [39] used classes of matting patterns as additional semantic information and [29] showed slightly less performance than our base model, a reproduction of MG Matting with the trimap input. Fig. 4 shows visual comparison results on Composition-1k among different methods demonstrating the effectiveness of our method.

**Distinctions-646.** In the case of Distinctions-646, it is hard to fairly compare with previously reported results because it does not offer official trimaps for the test set. We first produce the trimap from the ground truth alpha matte by binarizing the foreground with a threshold and randomly dilating it. We train both the baseline (MG Matting marked as *) described above and our MatteFormer on Distinctions-646 test set. * denotes the baseline of comparison which is our reproduction of MG Matting with trimap input.
Method  | SAD (10^{-3}) | MSE  | Grad  | Conn  
---|---|---|---|---
Baseline (no prior-token)  | 26.43 | 5.20 | 9.57 | 21.89 
Baseline + GAP prior-token  | 25.30 | 4.72 | 9.63 | 20.61 
Baseline + uk prior-token  | 24.70 | 4.46 | 9.10 | 19.73 
Baseline + uk/bg prior-token  | 24.19 | 4.05 | 8.72 | 19.19 
Baseline + prior-memory (MatteFormer)  | 23.80 | 4.03 | 8.68 | 18.90 

Table 3. Ablation study on the usage of prior-tokens and prior-memory. Baseline uses a Swin Transformer Tiny model as its encoder without prior-token. Results are on the Composition-1k.

646, and then evaluate them with the same testing environment. The results are shown in the last two rows of Tab. 2. The first five methods in the table are from [34] as references, which are trained on Distinctions-646 train set. The results show that MatteFormer has superior performance compared to the baseline model.

4.4. Ablation Study

Prior-tokens and Prior-memory. In MatteFormer, the building block of each encoder stage is the PAST block which is a variant version of Swin Transformer block, as described in Sec. 3.3. To show how the PAST block contributes to improving performance, we conduct an ablation study. We start from a baseline model in which we set the encoder as a pure Swin Transformer Tiny model without prior-token. The decoder and shortcuts of the baseline are the same as MatteFormer.

In the PA-WSA (Prior-Attentive Window Self-Attention) layer, a token in the local window can attend not only to the in-window spatial-tokens but also to the global prior-tokens. As a global prior, we first use an averaged token of all spatial-tokens (Global Average Pooling (GAP)-token). In this setting, we do not use any trimap information in self-attention layer. Next, we use unknown prior-token as a global prior. Further, we use all 3 prior-tokens; foreground, background and unknown prior-token. Finally, our proposed MatteFormer model uses all prior-tokens and introduces prior-memory which make it possible to access all prior-tokens generated from the previous blocks.

The results are shown in Tab. 3. The baseline which uses no prior-token performs worst in all metrics as expected. When using unknown prior-token, the metric values were lower than when using GAP-token as a global prior. This implies that the prior-token generated via the trimap delivers more useful information than the simple GAP-token. In the case of using 3 prior-tokens (foreground, background, unknown), it shows better performance than using 1 prior-token (unknown prior-token) only. This suggests that a token in a local window can refer all three prior-tokens properly in the self-attention layer. Introducing prior-memory also shows a improvement of performance. Prior-tokens from the previous blocks contribute to making better representation in the PA-WSA layer of the current block.

Visualization of attention maps in PA-WSA layer. In this subsection, we demonstrate that the PA-WSA layer indeed has a prior-attentive property. In Fig. 5, we visualize the mean attention maps, multi-head attention maps and the mean attention ratios between local spatial-tokens and global prior-tokens. We study both of them on 50 test images with different foregrounds. We averaged the attention maps over all windows and samples. For simplicity, we conduct the ablation study on the model without prior-memory.

In Fig. 5a, we first visualize the mean attention maps of
PA-WSA layers. Since local window size is set to 7, y-axis represents 49 query spatial-tokens in a local window. Meanwhile, x-axis represents 49 spatial-tokens and 3 prior-tokens (unknown, foreground and background token in order). We can observe that the prior-token regions (last three columns) in attention maps are activated. It means that one token can attend to the prior-tokens in a self-attention layer to refer to global priors as we intended.

Fig. 5b shows the attention maps of each head. We can see that each head has a different pattern of attention, especially in terms of prior-token usage. For example, head 4 and 5 mainly use the unknown prior-token and head 6 focuses on both unknown and background prior-tokens. Meanwhile, head 0 attends more on the spatial-tokens rather than the prior-tokens. Head 3 uses all three prior-tokens together with the spatial-tokens.

In Fig. 5c, we visualize mean attention ratios on local spatial-tokens and global prior-tokens over all blocks, quantitatively showing that how much attention is on each prior-token. There are four lines in Fig. 5c, one is the summation of attention ratios over local spatial-tokens, and the three others are the ratios of prior-tokens. We observe that prior-tokens are used in the PA-WSA layers across all blocks as we have expected. Tokens tend to attend more on the unknown prior-token than the known prior-tokens (foreground and background), which means that the unknown region is more informative to predict the alpha matte.

**ASPP (Atrous Spatial Pyramid Pooling).** Many state-of-the-art image matting models [25, 29, 34, 39, 54] and many semantic segmentation methods use ASPP (Atrous Spatial Pyramid Pooling) [5] between encoder and decoder to grow receptive fields for global representation. ASPP employs multiple atrous convolution filters with various atrous rates to capture spatially far-away context features.

However, we discover that ASPP is rather hinder the performance in our model. We evaluate MatteFormer on the Composition-1k both with ASPP and without ASPP in Tab. 4. All models with no ASPP show better performances than those with ASPP. Since our encoder is based on the transformer, MatteFormer already has a larger global receptive field than CNN-based models. As forcibly increasing receptive field with pre-defined atrous convolution, ASPP is redundant in our model. So, we do not use ASPP in our model unlike other recent methods.

**Parameters and FLOPs.** In Tab. 5, we compare the number of parameters and FLOPs of our MatteFormer to those of baseline models, for demonstrating the effectiveness of our method. Input image size is assumed to be 512 for calculating FLOPs. SAD and MSE are calculated on the Composition-1k test set for comparison. Note that the baseline (no prior-token) uses a pure Swin Transformer Tiny model as its backbone with shortcut layers and uses no prior-token. Also, it does not use the ASPP module unlike MG Matting-trimap* and MG Matting-trimap,res50*.

First, our baseline model has fewer parameters/FLOPs than MG Matting-trimap,res50* model, but shows better performance. It means that transformer-based architecture works effectively on the image matting problem. Next, comparing the baseline and our MatteFormer, we can observe a quite large gap on evaluation scores with a little increase of parameters and FLOPs. Consequently, Tab. 5 suggests that our proposed method, using prior-tokens and prior-memory, effectively works on the our transformer-based architecture.

### 5. Conclusion

In this work, we propose MatteFormer, a simple yet effective model with a modified Swin Transformer block called PAST (Prior-Attentive Swin Transformer) block for the image matting problem. We introduce prior-tokens which represent the context of global regions separated by the given trimap. In our PAST block, prior-tokens are used as global priors with local spatial-tokens, participating at the self-attention mechanism in the PA-WSA (Prior-Attentive Window Self-Attention) layer. So, one token can attend to both local spatial-tokens and global prior-tokens. We evaluate MatteFormer on the common datasets of the image matting problem. The experimental results show that our method achieves state-of-the-art performance. We hope that our MatteFormer can serve as a strong baseline for future image matting models based on a transformer architecture. One limitation of our work is that it mainly focuses on the encoder structure and the trimap-based approach. In the future work, we would like to design a fully transformer-based model with prior-tokens and also extend our model to trimap-free methods.
References


