A Generalist Framework for Panoptic Segmentation of Images and Videos

Ting Chen, Lala Li, Saurabh Saxena, Geoffrey Hinton†, David J. Fleet†
Google Deepmind
{iamtingchen,lala,srbs,geoffhinton,davidfleet}@google.com

Abstract

Panoptic segmentation assigns semantic and instance ID labels to every pixel of an image. As permutations of instance IDs are also valid solutions, the task requires learning of high-dimensional one-to-many mapping. As a result, state-of-the-art approaches use customized architectures and task-specific loss functions. We formulate panoptic segmentation as a discrete data generation problem, without relying on inductive bias of the task. A diffusion model is proposed to model panoptic masks, with a simple architecture and generic loss function. By simply adding past predictions as a conditioning signal, our method is capable of modeling video (in a streaming setting) and thereby learns to track object instances automatically. With extensive experiments, we demonstrate that our simple approach can perform competitively to state-of-the-art specialist methods in similar settings.†

1. Introduction

Panoptic segmentation [30] is a fundamental vision task that assigns semantic and instance labels for every pixel of an image. The semantic labels describe the class of each pixel (e.g., sky, vertical), and the instance labels provides a unique ID for each instance in the image (to distinguish different instances of the same class). The task is a combination of semantic segmentation and instance segmentation, providing rich semantic information about the scene.

While the class categories of semantic labels are fixed a priori, the instance IDs assigned to objects in an image can be permuted without affecting the instances identified. For example, swapping instance IDs of two cars would not affect the outcome. Thus, a neural network trained to predict instance IDs should be able to learn a one-to-many mapping, from a single image to multiple instance ID assignments. The learning of one-to-many mappings is challenging and traditional approaches usually leverage a pipeline of multiple stages involving object detection, segmentation,

†Equal advising.
†Code at https://github.com/google-research/pix2seq

Figure 1: We formulate panoptic segmentation as a conditional discrete mask (m) generation problem for images (left) and videos (right), using a Bit Diffusion generative model [12].

merging multiple predictions [30, 34, 66, 40, 14]. Recently, end-to-end methods [58, 17, 70, 16, 35, 68, 69, 33] have been proposed, based on a differentiable bipartite graph matching [7]; this effectively converts a one-to-many mapping into a one-to-one mapping based on the identified matching. However, such methods still require customized architectures and sophisticated loss functions with built-in inductive bias for the panoptic segmentation task.

Eshewing task-specific architectures and loss functions, recent generalist vision models, such as Pix2Seq [10, 11], OFA [60], UViM [31], and Unified I/O [43], advocate a generic, task-agnostic framework, generalizing across multiple tasks while being much simpler than previous models. For instance, Pix2Seq [10, 11] formulates a set of core vision tasks in terms of the generation of semantically meaningful sequences conditioned on an image, and they train a single autoregressive model based on Transformers [55].

Following the same philosophy, we formulate the panoptic segmentation task as a conditional discrete data generation problem, depicted in Figure 1. We learn a generative model for panoptic masks, treated as an array of discrete tokens, conditioned on an input image. One can also apply the model to video data (in an online/streaming setting), by simply including predictions from past frames as an additional conditioning signal. In doing so, the model can then learn to track and segment objects automatically.

Generative modeling for panoptic segmentation is very challenging as the panoptic masks are discrete/categorical...
and can be very large. To generate a $512 \times 1024$ panoptic mask, for example, the model has to produce more than $1M$ discrete tokens (of semantic and instance labels). This is expensive for auto-regressive models as they are inherently sequential, scaling poorly with the size of data input. Diffusion models [50, 23, 51, 52] are better at handling high dimension data but they are most commonly applied to continuous rather than discrete domains. By representing discrete data with analog bits [12] we show that one can train a diffusion model on large panoptic masks directly, without the need to also learn an intermediate latent space.

In what follows, we introduce our diffusion-based model for panoptic segmentation, and then describe extensive experiments on both image and video datasets. In doing so we demonstrate that the proposed method performs competitively to state-of-the-art methods in similar settings, proving a simple and generic approach to panoptic segmentation.

2. Preliminaries

Problem Formulation. Introduced in [30], panoptic segmentation masks can be expressed with two channels, $m \in \mathbb{Z}^{H \times W \times 2}$. The first represents the category/class label. The second is the instance ID. Since instance IDs can be perturbed without changing the underlying instances, we randomly assign integers in $[0, K]$ to instances every time an image is sampled during training. $K$ is maximum number of instances allowed in any image (0 denotes the null label).

To solve the panoptic segmentation problem, we simply learn an image-conditional panoptic mask generation model by maximizing $\sum_i \log P(m_i | x_i)$, where $m_i$ is a random categorical variable corresponding to the panoptic mask for image $x_i$ in the training data. As mentioned above, panoptic masks may comprise hundreds of thousands or even millions of discrete tokens, making generative modeling very challenging, particularly for autoregressive models.

Diffusion Models with Analog Bits. Unlike autoregressive generative models, diffusion models are more effective with high dimension data [50, 23, 51, 52]. Training entails learning a denoising network. During inference, the network generates target data in parallel, using far fewer iterations than the number of pixels.

In a nutshell, diffusion models learn a series of state transitions to transform noise $\epsilon$ from a known noise distribution into a data sample $x_0$ from the data distribution $p(x)$. To learn this mapping, we first define a forward transition from data $x_0$ to a noisy sample $x_t$ as follows,

$$x_t = \sqrt{1-t} x_0 + \sqrt{t} \, \epsilon,$$

where $\epsilon$ is drawn from standard normal density, $t$ is from uniform density on $[0,1]$, and $\gamma(t)$ is a monotonically decreasing function from 1 to 0. During training, one learns a neural network $f(x_t, t)$ to predict $x_0$ (or $\epsilon$) from $x_t$, usually formulated as a denoising task with an $\ell_2$ loss:

$$L_{x_0} = \mathbb{E}_{t \sim (0, T), \epsilon \sim N(0,1), x_0} \| f(x_t, t) - x_0 \|^2.$$  

To generate samples from a learned model, it starts with a sample of noise, $x_T$, and then follows a series of (reverse) state transitions $x_T \rightarrow x_{T-\Delta} \rightarrow \cdots \rightarrow x_0$ by iteratively applying the denoising function $f$ with appropriate transition rules (such as those from DDPM [23] or DDIM [51]).

Conventional diffusion models assume continuous data and Gaussian noise, and are not directly applicable to discrete data. To model discrete data, Bit Diffusion [12] first converts integers representing discrete tokens into bit strings, the bits of which are then cast as real numbers (a.k.a., analog bits) to which continuous diffusion models can be applied. To draw samples, Bit Diffusion uses a conventional sampler from continuous diffusion, after which a final quantization step (simple thresholding) is used to obtain the categorical variables from the generated analog bits.

3. Method

We formulate panoptic segmentation as a discrete data generation problem conditioned on pixels, similar to Pix2Seq [10, 11] but for dense prediction; hence we coin our approach Pix2Seq-D. In what follows we first specify the architecture design, and then the training and inference algorithms based on Bit Diffusion.

3.1. Architecture

Diffusion model sampling is iterative, and hence one must run the forward pass of the network many times during inference. Therefore, as shown in Fig. 2, we intentionally separate the network into two components: 1) an image encoder; and 2) a mask decoder. The former maps raw pixel data into high-level representation vectors, and then the mask decoder iteratively reads out the panoptic mask.

Pixel/image Encoder. The encoder is a network that maps a raw image $x \in \mathbb{R}^{H \times W \times 3}$ into a feature map in $\mathbb{R}^{H' \times W' \times d}$ where $H'$ and $W'$ are the height and width of the panoptic mask. The panoptic masks can be the same size or smaller than the original image. In this work, we follow [7, 10] in using ResNet [22] followed by transformer encoder layers [55] as the feature extractor. To make sure the output feature map has sufficient resolution, and includes features at different scales, inspired by U-Net [23, 45, 47] and feature pyramid network [38], we use convolutions with bilateral connections and upsampling operations to merge features from different resolutions. More sophisticated encoders are possible, to leverage recent advances in architecture designs [20, 41, 25, 71, 26], but this is not our main focus so we opt for simplicity.
Figure 2: The architecture for our panoptic mask generation framework. We separate the model into image encoder and mask decoder so that the iterative inference at test time only involves multiple passes over the decoder.

**Algorithm 1** Pix2Seq-D training algorithm.

```python
def train_loss(images, masks):
    # images: [b, h, w, 3], masks: [b, h', w', 2].

    # Encode image features.
    h = pixel_encoder(images)

    # Discrete masks to analog bits.
    m_bits = int2bit(masks).astype(float)
    m_bits = (m_bits * 2 - 1) * scale

    # Corrupt analog bits.
    t = uniform(0, 1)  # scalar.
    eps = normal(mean=0, std=1)  # same shape as m_bits.
    m_crpt = sqrt(gamma(t)) * m_bits + sqrt(1 - gamma(t)) * eps

    # Predict and compute loss.
    m_logits, _ = mask_decoder(m_crpt, h, t)
    loss = cross_entropy(m_logits, masks)
    return loss.mean()
```

**Mask Decoder.** The decoder iteratively refines the panoptic mask during inference, conditioned on the image features. Specifically, the mask decoder is a TransUNet [8]. It takes as input the concatenation of image feature map from encoder and a noisy mask (randomly initialized or from previous step), and outputs a refined prediction of the mask. One difference between our decoder and the standard U-Net architecture used for image generation and image-to-image translation [23, 45, 48] is that we use transformer decoder layers on the top of U-Net, with cross-attention layers to incorporate the encoded image features (before upsampling).

### 3.2. Training Algorithm

Our main training objective is the conditional denoising of analog bits [12] that represent noisy panoptic masks. Algorithm 1 gives the training algorithm (with extra details in A), the key elements of which are introduced below.

**Algorithm 2** Pix2Seq-D inference algorithm.

```python
def infer(images, steps=10, td=1.0):
    # images: [b, h, w, 3].

    # Encode image features.
    h = pixel_encoder(images)

    m_t = normal(mean=0, std=1)  # same shape as m_bits.
    for step in range(steps):
        # Get time for current and next states.
        t_now = 1 - step / steps
        t_next = max(1 - (step + 1 + td) / steps, 0)

        # Predict analog bits m_0 from m_t.
        m_crpt, m_pred = mask_decoder(m_t, h, t_now)

        # Estimate m at t_next.
        m_t = ddim_step(m_t, m_pred, t_now, t_next)

    # Analog bits to masks.
    masks = bit2int(m_pred > 0)
    return masks
```

When constructing the analog bits, we can shift and scale them into $\{-b, b\}$. Typically, $b$ is set to be 1 [12] but we find that adjusting this scaling factor has a significant effect on the performance of the model. This scaling factor effectively allows one to adjust the signal-to-noise ratio of the diffusion process (or the noise schedule), as visualized in Fig. 3. With $b = 1$, we see that even with a large time step $t = 0.7$ (with $t \in [0, 1]$), the signal-to-noise ratio is still relatively high, so the masks are visible to naked eye and the model can easily recover the mask without using the encoded image features. With $b = 0.1$, however, the denoising task becomes significantly harder as the signal-to-noise ratio is reduced. In our study, we find $b = 0.1$ works substantially better than the default of 1.0.

**Softmax Cross Entropy Loss.** Conventional diffusion models (with or without analog bits) are trained with an $\ell_2$ denoising loss. This works reasonably well for panoptic segmentation, but we also discovered that a loss based on softmax cross entropy yields better performance. Al-
though the analog bits are real numbers, they can be seen as a one-hot weighted average of base categories. For example, ‘01’ = \(\alpha_0'00' + \alpha_1'01' + \alpha_2'10' + \alpha_3'11'\) where \(\alpha_1 = 1\), and \(\alpha_0 = \alpha_2 = \alpha_3 = 0\). Instead of modeling the analog bits in ‘01’ as real numbers, with a cross entropy loss, the network can directly model the underlying distribution over the four base categories, and use the weighted average to obtain the analog bits. As such, the mask decoder output not only analog bits (\(m_{\text{pred}}\)), but also the corresponding logits (\(m_{\text{logits}}\)), \(\hat{y} \in \mathbb{R}^{H \times W \times K}\), with a cross entropy loss for training; i.e.,

\[
L = \sum_{i,j,k} y_{ijk} \log \text{softmax}(\hat{y}_{ijk})
\]

Here, \(y\) is the one-hot vector corresponding to class or instance label. During inference, we still use analog bits from the mask decoder instead of underlying logits for the reverse diffusion process.

**Loss Weighting.** Standard generative models for discrete data assign equal weight to all tokens. For panoptic segmentation, with a loss defined over pixels, this means that large objects will have more influence than small objects. And this makes learning to segment small instances inefficient. To mitigate this, we use an adaptive loss to improve the segmentation of small instances by giving higher weights to mask tokens associated with small objects. The specific per-token loss weighting is as follows:

\[
w_{ij} = 1/c_{ij}^p, \quad \text{and} \quad w'_{ij} = H \times W \times w_{ij}/\sum_{ij} w_{ij},
\]

where \(c_{ij}\) is the pixel count for the instance at pixel location \((i, j)\), and \(p\) is a tunable parameter; uniform weighting occurs when \(p = 0\), and for \(p > 0\), a higher weight is assigned to mask tokens of small instances. Fig. 4 demonstrate the effects of \(p\) in weighting different mask tokens.

### 3.3. Inference Algorithm

Algorithm 2 summarizes the inference procedure. Given an input image, the model starts with random noise as the initial analog bits, and gradually refines its estimates to be closer to that of good panoptic masks. Like Bit Diffusion [12], we use asymmetric time intervals (controlled by a single parameter \(t_d\)) that is adjustable during inference time. It is worth noting that the encoder is only run once, so the cost of multiple iterations depends on the decoder alone.

### 3.4. Extension to Videos

Our image-conditional panoptic mask modeling with \(p(m|x)\) is directly applicable for video panoptic segmen-
to set refinement steps adaptively across video frames. For inference on DA VIS, we use 32 sampling steps for the first frame and 8 steps for subsequent frames. We set \( t_{d} = 2.0 \) yields near optimal results. We discard instance predictions with fewer than 80 pixels.

Training. MS-COCO is larger and more diverse than Cityscapes and DAVIS. Thus we mainly train on MS-COCO, and then transfer trained models to Cityscapes and DAVIS with fine-tuning (at a single resolution). We first separately pre-train the image encoder and mask decoder before training the image-conditional mask generation on MS-COCO. The image encoder is taken from the Pix2Seq [10] object detection checkpoint, pre-trained on objects365 [49]. It comprises a ResNet-50 [22] backbone, and 6-layer 512-dim Transformer [55] encoder layers. We also augment image encoder with a few convolutional up-sampling layers to increase its resolution and incorporate features at different layers. The mask decoder is a TransUNet [8] with base dimension 128, and channel multipliers of \( 1 \times 1 \times 2 \times 2 \times \), followed by 6-layer 512-dim Transformer [55] decoder layers. It is pre-trained on MS-COCO as an unconditional mask generation model without images.

Directly training our model on high resolution images and panoptic masks can be expensive as the existing architecture scales quadratically with resolution. So on MS-COCO, we train the model with increasing resolutions, similar to [53, 54, 24]. We first train at a lower resolution (256×256 for images; 128×128 for masks) for 800 epochs with a batch size of 512 and scale jittering [21, 65] of strength [1.0, 3.0]. We then continue train the model at full resolution (1024×1024 for images; 512×512 for masks) for only 15 epochs with a batch size of 16 without augmentation. This works well, as both convolution networks and transformers with sin-cos positional encoding generalize well across resolutions. More details on hyper-parameter settings for training can be found in Appendix B.

Inference. We use DDIM updating rules [51] for sampling. By default we use 20 sampling steps for MS-COCO. We find that setting \( t_{d} = 2.0 \) yields near optimal results. We discard instance predictions with fewer than 80 pixels.

4.2. Main Results

We compare with two families of state-of-the-art methods, i.e., specialist approaches and generalist approaches. Table 1 summarizes results for MS-COCO. Pix2Seq-\( D \) achieves competitive Panoptic Quality (PQ) to state-of-the-art methods with the ResNet-50 backbone. When compared with other recent generalist models such as UViM [31], our model performs significantly better while being much more efficient. Similar results are obtained for Cityscape, the details of which are given in Appendix C.

Table 2 compares Pix2Seq-\( D \) to state-of-the-art methods on unsupervised video object segmentation on DAVIS, using the standard \( J_{F} \) & \( F \) metrics [46]. Baselines do not in-
<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th># of Params</th>
<th>PQ</th>
<th>PQ$^{bing}$</th>
<th>PQ$^{euff}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialist approaches:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MaskFormer [17]</td>
<td>ResNet-50</td>
<td>45M</td>
<td>46.5</td>
<td>51.0</td>
<td>39.8</td>
</tr>
<tr>
<td>K-Net [70]</td>
<td>ResNet-50</td>
<td>-</td>
<td>47.1</td>
<td>51.7</td>
<td>40.3</td>
</tr>
<tr>
<td>CMT-DeepLab [68]</td>
<td>ResNet-50</td>
<td>-</td>
<td>48.5</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Panoptic SegFormer [35]</td>
<td>ResNet-50</td>
<td>51M</td>
<td>49.6</td>
<td>54.4</td>
<td>42.4</td>
</tr>
<tr>
<td>Mask2Former [16]</td>
<td>ResNet-50</td>
<td>44M</td>
<td>51.9</td>
<td>57.7</td>
<td>43.0</td>
</tr>
<tr>
<td>kMaX-DeepLab [69]</td>
<td>ResNet-50</td>
<td>57M</td>
<td>53.0</td>
<td>58.3</td>
<td>44.9</td>
</tr>
<tr>
<td>DETR [7]</td>
<td>ResNet-101</td>
<td>61.8M</td>
<td>45.1</td>
<td>50.5</td>
<td>37.0</td>
</tr>
<tr>
<td>Mask2Former [13]</td>
<td>Swin-L</td>
<td>216M</td>
<td>57.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>kMaX-DeepLab [69]</td>
<td>ConvNeXt-L</td>
<td>232M</td>
<td>58.1</td>
<td>64.3</td>
<td>48.8</td>
</tr>
<tr>
<td>MasK DINO [33]</td>
<td>Swin-L</td>
<td>223M</td>
<td>59.4</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Generalist approaches:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>UViM [31]</td>
<td>ViT</td>
<td>939M</td>
<td>45.8</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pix2Seq-$\mathcal{D}$ (steps=5)</td>
<td>ResNet-50</td>
<td>94.5M</td>
<td>47.5</td>
<td>52.2</td>
<td>40.3</td>
</tr>
<tr>
<td>Pix2Seq-$\mathcal{D}$ (steps=10)</td>
<td>ResNet-50</td>
<td>94.5M</td>
<td>49.4</td>
<td>54.4</td>
<td>41.9</td>
</tr>
<tr>
<td>Pix2Seq-$\mathcal{D}$ (steps=20)</td>
<td>ResNet-50</td>
<td>94.5M</td>
<td>50.3</td>
<td>55.3</td>
<td>42.9</td>
</tr>
<tr>
<td>Pix2Seq-$\mathcal{D}$ (steps=50)</td>
<td>ResNet-50</td>
<td>94.5M</td>
<td>50.2</td>
<td>55.1</td>
<td>42.8</td>
</tr>
</tbody>
</table>

Table 1: Results on MS-COCO. Pix2Seq-$\mathcal{D}$ achieves competitive results to state-of-the-art specialist models with ResNet-50 backbone.

<table>
<thead>
<tr>
<th>Method</th>
<th>Backbone</th>
<th>$\mathcal{J} &amp; \mathcal{F}$</th>
<th>$\mathcal{J}$-Mean</th>
<th>$\mathcal{F}$-Recall</th>
<th>$\mathcal{J}$-mean</th>
<th>$\mathcal{F}$-Recall</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialist approaches:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RVOS [56]</td>
<td>ResNet-101</td>
<td>41.2</td>
<td>36.8</td>
<td>40.2</td>
<td>45.7</td>
<td>46.4</td>
</tr>
<tr>
<td>STEM-Seg [3]</td>
<td>ResNet-101</td>
<td>64.7</td>
<td>61.5</td>
<td>70.4</td>
<td>67.8</td>
<td>75.5</td>
</tr>
<tr>
<td>MAST [32]</td>
<td>ResNet-18</td>
<td>65.5</td>
<td>63.3</td>
<td>73.2</td>
<td>67.6</td>
<td>77.7</td>
</tr>
<tr>
<td>UnOVOSt [44]</td>
<td>ResNet-101</td>
<td>67.9</td>
<td>66.4</td>
<td>76.4</td>
<td>69.3</td>
<td>76.9</td>
</tr>
<tr>
<td>Propose-Reduce [37]</td>
<td>ResNeXt-101</td>
<td>70.4</td>
<td>67.0</td>
<td>-</td>
<td>73.8</td>
<td>-</td>
</tr>
<tr>
<td>Generalist approaches:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pix2Seq-$\mathcal{D}$ (ours)</td>
<td>ResNet-50</td>
<td>68.4</td>
<td>65.1</td>
<td>70.6</td>
<td>71.7</td>
<td>77.1</td>
</tr>
</tbody>
</table>

Table 2: Results of unsupervised video object segmentation on DAVIS 2017 validation set.

<table>
<thead>
<tr>
<th>Input scaling</th>
<th>PQ</th>
<th>$\ell_2$ Regression</th>
<th>Cross Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.03</td>
<td>40.8</td>
<td>43.9</td>
<td>38.7</td>
</tr>
<tr>
<td>0.1</td>
<td></td>
<td></td>
<td>21.3</td>
</tr>
</tbody>
</table>

Table 3: Ablation on input scaling.

<table>
<thead>
<tr>
<th>Loss function</th>
<th>PQ</th>
<th>$\ell_2$ Regression</th>
<th>Cross Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>PQ</td>
<td>41.9</td>
<td></td>
<td>43.9</td>
</tr>
</tbody>
</table>

Table 4: Ablation on loss function.

<table>
<thead>
<tr>
<th>Loss weight $p$</th>
<th>PQ</th>
<th>$\ell_2$ Regression</th>
<th>Cross Entropy</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>40.4</td>
<td>43.9</td>
<td>43.7</td>
</tr>
<tr>
<td>0.4</td>
<td></td>
<td></td>
<td>41.3</td>
</tr>
</tbody>
</table>

Table 5: Ablation on loss weighting.

4.3. Ablations on Training

Ablations on model training are performed with MS-COCO dataset. To reduce the computation cost while still be able to demonstrate the performance differences in design choices, we train the model for 100 epochs with a batch size of 128 in a single-resolution stage (512×512 image size, and 256×256 mask size).

**Input Scaling of Analog Bits.** Table 3 shows the dependence of PQ results on input scaling of analog bits. The scale factor of 0.1 used here yield results that outperform the standard scaling of 1.0 in previous work [12].

**Loss Functions.** Table 4 compares our proposed cross entropy loss to the $\ell_2$ loss normally used by diffusion models. Interestingly, the cross entropy loss yields substantial gains over $\ell_2$.

**Loss weighting.** Table 5 shows the effects of $p$ for loss weighting. Weighting with $p = 0.2$ appears near optimal and clearly outperforms uniform weighting ($p = 0$).

4.4. Ablations on Inference

Figure 6 explores the dependence of model performance on hyper-parameter choices during inference (sampling),

clude other generalist models as they are not directly applicable to the task. Our method achieves results on par with state-of-the-art methods without specialized designs.
namely, the number of inference steps, time differences and threshold on the minimum size of instance regions, all on MS-COCO. Specifically, Fig. 6a shows that inference steps of 20 seems sufficient for near optimal performance on MS-COCO. Fig. 6b shows that the \( t_{\text{d}} \) parameter as in asymmetric time intervals [12] has a significant impact, with intermediate values (e.g., 2-3) yielding the best results. Fig. 6c shows that the right choice of threshold on small instances leads to small performance gains.

Figure 7 shows how performance varies with the number of inference steps for the first frame, and for the remaining frames, on DAVIS video dataset. We find that more inference steps are helpful for the first frame compared to subsequent frames of video. Therefore, we can reduce the total number of steps by using more steps for the first frames (e.g., 32), and fewer steps for subsequent frames (e.g., 8). It is also worth noting that even using 8 steps for the first frame and only 1 step for each subsequent frame, the model still achieves an impressive \( J & F \) of 67.3.

4.5. Case study

Figure 8, 9 and 10 show example results of Pix2Seq-\( D \) on MS-COCO, Cityscape, and DAVIS. One can see that our model is capable of capturing small objects in dense scene well. More visualizations are shown in Appendix E.

5. Related Work

Image Panoptic Segmentation. Panoptic segmentation, introduced in [30], unifies semantic segmentation and instance segmentation. Previous approaches to panoptic segmentation involve pipelines with multiple stages, such as object detection, semantic and instance segmentation, and the fusion of separate predictions [30, 34, 66, 40, 14, 7]. With multiple stages involved, learning is often not end-to-end. Recent work has proposed end-to-end approaches with Transformer based architectures [58, 17, 70, 16, 35, 68, 69, 33], for which the model directly predicts segmentation masks and optimizes based on a bipartite graph matching loss. Nevertheless, they still require customized architectures (e.g., per instance mask generation, and mask fusion module). Their loss functions are also specialized for optimizing metrics used in object matching.

Our approach is a significant departure to existing methods with task-specific designs, as we simply treat the task as image-conditional discrete mask generation, without reliance on inductive biases of the task. This results in a simpler and more generic design, one which is easily extended to video segmentation with minimal modifications.

Video Segmentation. Among the numerous video segmentation tasks, video object segmentation (VOS) [46, 61] is perhaps most canonical task, which entails the segmentation of key objects (of unknown categories). Video instance segmentation (VIS) [67] is similar to VOS, but requires category prediction of instances. Video panoptic segmentation (VPS) [28] is a direct extension of image panoptic segmentation to the video domain. All video segmentation tasks involve two main challenges, namely, segmentation and object tracking. And like image segmentation, most existing methods are specialist models comprising multiple stages with pipe-lined frameworks, e.g., track-detect [67, 57, 36, 6, 44], clip-match [3, 5], propose-reduce [37]. End-to-end approaches have also been proposed recently [62, 29], but with a specialized loss function.

In this work we directly take the Pix2Seq-\( D \) model, pretrained on COCO for panoptic segmentation, and fine-tune it for unsupervised video object segmentation (UVOS), where it performs VOS without the need for manual initialization. Model architectures, training losses, input augmentations and sampling methods all remained largely unchanged when applied to UVOS data. Because of this, we believe it is just as straightforward to apply Pix2Seq-\( D \) to the other video segmentation tasks as well.

Others. Our work is also related to recent generalist vision models [10, 11, 60, 31, 43] where both architecture and loss functions are task-agnostic. Existing generalist models are based on autoregressive models, while our work is based on Bit Diffusion [12, 23, 50, 51]. Diffusion models have
been applied to semantic segmentation, directly [1, 64, 27]
or indirectly [4, 2]. However none of these methods model segmentation masks as discrete/categorical tokens, nor are their models capable of video segmentation.

6. Conclusion and Future Work

This paper proposes a simple framework for panoptic segmentation of images and videos, based on conditional generative models of discrete panoptic masks. Our approach is able to model large number of discrete tokens ($10^6$ in our experiments), which is difficult with other existing generative segmentation models. We believe both the architecture, modeling choices, and training procedure (including augmentations) we use here can be further improved to boost the performance. Furthermore, the required inference steps can also be further reduced with techniques like progressive distillation. Finally, we want to note that, as a significant departure to status quo, we acknowledge that our current empirical result is still behind compared to well-tuned pipelines in existing systems (though our results are still competitive and at a practically usable level). However, with the simplicity of the proposed approach, we hope it would spark future development that drives new state-of-the-art systems.
Acknowledgements

We specially thank Liang-Chieh Chen for the helpful feedback on our initial draft.

References


