A Fixed-Point Approach to Unified Prompt-Based Counting

Wei Lin, Antoni B. Chan
Department of Computer Science, City University of Hong Kong
Tat Chee 83, Kowloon Tong, Hong Kong SAR, China
elonlin24@gmail.com, abchan@cityu.edu.hk

Overview
This supplemental material is organized as follows:
1. Results on cross-prompt and multi-prompt adaptation is discussed to show the advantages of the unified prompt-based counting;
2. More details about the three components in the proposed prompt-based counting model are presented;
3. The pseudo-code is presented to show the whole computation process of fixed-point loss in contrastive training.
4. The detailed results in ablation studies.
5. Additional visualizations of the proposed method are presented to showcase its advantages and disadvantages.

Cross/Multi-Prompt Adaptation
Converting different types of prompts into the same representation, prompt masks, offers two additional advantages:
(a) When new types of prompts are introduced, we can seamlessly transform them into prompts and directly apply them in our model without requiring training; (b) This approach enables us to combine multiple prompts for counting the object of interest.

To showcase the first advantage, we incorporate an additional prompt type, namely instance mask prompts, for prompt-based counting without requiring extra training. The instance mask is generated by converting the box annotations into instance masks using the SAM method (Kirillov et al. 2023). The results are presented in Table S1. The instance prompts yield comparable outcomes to box-guided training, as the prompts originate from box annotations. However, when using instance prompts, the estimation errors are slightly lower than the results obtained from box annotations, owing to the more precise representation provided by instance masks.

Regarding multiple prompts, we present the results in Table S2. Overall, the performance remains consistent with the results obtained using a single type of prompt. However, when the text prompt is combined with other prompts, such as box or point prompts, its performance improves. The original MAE and MSE of the text prompt are 16.81 and 105.83, respectively. These values decrease to 16.71 and 105.80 when the box prompt is added. Meanwhile, the MAE and MSE decrease to 16.78 and 105.79 when combined with point prompts.

Model Structure
Our prompt-based counting model consists of three main components: an image backbone, the cross-attention module, and the density predictor, as illustrated in Figure 2 of the main paper.

Backbone & Image Feature. The backbone utilize the first four blocks of ResNet-101 (He et al. 2016) as the backbone, the image feature can be represented as:
\[
F = G \left( \left[ F_{\frac{1}{3}}, \mathcal{U}_{2} \left( \frac{F_{\frac{1}{3}}}{2} \right) \right], \Theta_{G} \right),
\]
(S1)
where \( G \) is a \( \text{conv}_3 \times 3 \) layer with parameters \( \Theta_{G} \), and \( \mathcal{U}_2 \) represents bilinear interpolation with a scale factor of 2. The inputs to \( G \), denoted as \( F_{\frac{1}{3}} \) and \( \frac{F_{\frac{1}{3}}}{2} \), are feature maps extracted from the last two blocks in the backbone. Their subscripts indicate the downsampling rate compared to the input. \( F \) has the same resolution with \( F_{\frac{1}{3}} \).

Cross Attention & Density Feature. The aim of cross attention is to let the prompt token \( t \in \mathbb{R}^C \) communicate with the flattened feature \( F \in \mathbb{R}^{N \times C} \), and then generate density feature accordingly. Specifically, we use \( F \) to generate query
embedding, and use t to generate key and value embeddings:

\[ Q = FW_q, \quad k = W_k t, \quad v = W_v t \quad (S2) \]

in which \( W_q \in \mathbb{R}^{C \times C'} \), \( W_k \in \mathbb{R}^{C' \times C} \), and \( W_v \in \mathbb{R}^{C' \times C} \) are learnable parameters to perform cross attention. Prior attention models use softmax to compute the weights for each value embedding. However, there is only one token so that the result of softmax would always be one, contrast to the movetation of weight. Thus, we replace the softmax with Sigmoid function \( \sigma(x) = \frac{1}{1 + e^{-x}} \), resulting the following way to generate density features:

\[ F' = \mathcal{M}(S^\top v), \quad S = \sigma(Qk) \in \mathbb{R}^N, \quad (S3) \]

where \( \mathcal{M} \) is a linear projection to transform the attention results into density features, denoted as \( F' \). The visualized pipeline is displayed in Figure S1.

**Density Predictor.** We utilize the identical density predictor released in SPDCN (Lin et al. 2022), benefiting from its straightforward yet effective performance. As illustrated in Figure S2, the density predictor is composed of three combined layers of \texttt{conv.3x3} and \texttt{ReLU}. Before reaching the final convolutional layer, a \texttt{pixel_shuffle} layer with scale factor of 8 is employed to upscale the density features, aligning them with the resolution of the input image.

**Pseudo Code of Contrastive Training**

Algorithm S1 summarizes the procedure for computing the fixed-point loss under the contrastive training scheme. At the beginning, \( d_i^{(0)} \) and \( d_j^{(0)} \) are initialized with \( m_i \) and \( m_j \) respectively. Subsequently, the fixed-point estimation is performed iteratively.

In the \( t \)-th iteration, we first compute the prompt token \( t^{(t+1)} \) based on both the positive and negative prompt masks, \( d_i^{(t)} \) and \( d_j^{(t)} \). Following this, the positive and negative density map are updated via the function \( D \) with parameters \( \theta \), as described in (9) of the main paper. Here \( D \) comprises the cross-attention module in Figure S1 and the density predictor in Figure S2. Note that both density maps are detached from the computation graph before the final iteration.

To compute the fixed-point loss, we also predict the corresponding density map with the token at infinity, \( t^{(\infty)} \). The final loss function is computed for both the positive and negative samples. The distinction lies in that the prediction for the positive component should align with the ground truth, while the prediction for the negative component should approach an all-zero density map. It is worth noting that \( d_i^{(\infty)} \) is detached in the finite loss calculation to prevent degenerate solutions where both \( d_i^{(T)} \) and \( d_j^{(\infty)} \) converge towards zero (Chen and He 2021).

**Detailed Results in Ablation Studies**

In the main paper, we only present the average MAE/MSE across three types of prompts to conserve space. Here, we provide the detailed corresponding data. Table S3 compares the methods for generating text prompt masks; Table S4 presents the results while comparing fixed-point loss with L2 loss; Table S5 discusses the enhancements achieved by different components in fixed-points; and Table S6 provides...
Table S3: Comparison of text prompt mask generation.

<table>
<thead>
<tr>
<th>Mask</th>
<th>box (val)</th>
<th>text (val)</th>
<th>point (val)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cosine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
</tr>
<tr>
<td></td>
<td>18.55</td>
<td>69.32</td>
<td>20.59</td>
</tr>
<tr>
<td>softmax</td>
<td>16.87</td>
<td>59.42</td>
<td>16.92</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mask</th>
<th>box (test)</th>
<th>text (test)</th>
<th>point (test)</th>
</tr>
</thead>
<tbody>
<tr>
<td>cosine</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>MAE</td>
<td>MSE</td>
<td>MAE</td>
</tr>
<tr>
<td></td>
<td>16.68</td>
<td>105.08</td>
<td>16.81</td>
</tr>
<tr>
<td>softmax</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table S4: Comparison between MSE and fixed-point loss.

Table S5: Ablation study on different combinations of infinity and finity part in fixed-point loss.

Table S6: Ablation study on contrastive training.

The N-MAE and N-MSE values for different prompts.

More Visualization

In this section, we also present more visualizations of the proposed method. In Figure S3, we provide some counting examples with text prompts. It demonstrates that our model can generate accurate density maps even when the given image contains multiple types of objects. For convenience, we use $d^{(1)}$ to denote the prediction in the $t$-th iteration. The visualization shows that $d^{(1)}$ consistently predicts a lower count than $d^{(2)}$. When compared with $d^{(1)}$, the count of $d^{(2)}$ is closer to the ground truth.

In Figure S4, we illustrate some failure cases. In the first row of Figure S4, the minimum repetitive unit is a single lens. However, the object of interest, sunglasses, naturally has two lenses, doubling the repetitive unit. This discrepancy causes the model to predict a count that is twice the ground truth. In the second row of Figure S4, two types of pills with completely different appearances and shapes are presented. The model only counts one of them, while the other one is recognized as irrelevant object.

Figure S5 showcases two examples of counting with different prompts. These diverse prompts also exhibit varying levels of performance, but the estimation errors remain consistent across the entire dataset.

References


Figure S3: Examples of counting with text prompts.

Figure S4: Typical failure cases.
Figure S5: Examples of counting with different types of prompts.