1. Appendix A.

Effect of multi-stage training during rectifying process. In our target-guided uncertainty rectifying, we refine high-uncertain classes using multiple stages. As shown in Table 1, we provide the performance of different refinery stages for the GTA5 $\rightarrow$ Cityscapes task. We can observe that the refinery stage can improve the performance with a large margin. However, when we refine the model on the 5th stage, the performance drops the mIoU by 0.9 due to the over-fitting.

	1	2	3	4	5
mIoU	46.1	46.7	47.4	48.8	47.9

Table 1. The performance of different refinery stages. Here we only report the performance for the GTA5 $\rightarrow$ Cityscapes task.

The effect of hyper-parameters α and μ during soft-balanace sampling . In our soft-balance sampling strategy, we use α and μ to smooth the shape of sampling probability. To testify the influence of these parameters, we conduct experiments on GTA5 $\rightarrow$ Cityscapes, and we only report the performance for the 1st refinery stage.

α	μ	mIoU
30	0.07	44.6
40	0.05	44.9
60	0.03	45.8
80	0.02	46.1

Table 2. The influence of various hyper-parameters. We report the performance for the 1st refinery stage in the GTA5 $\rightarrow$ Ciyscapes task.

Explanation for pseudo labels assignment. In our implementation of uncertainty-aware pseudo labels assignment, we assume that the input pixel-level entropy follows a bimodal distribution, estimated by a Gaussian Mixture Model (GMM). Precisely, the left mode corresponds to pixels with correctly predicted labels, and the right mode to pixels with incorrectly predicted labels. Intuitively, this assumption can be easily violated in practice when multiple entropy modes occur. In practice, within a single image and a single class, the distribution is relatively simple, and the multi-modes problem can be generally avoided. We also experimentally confirmed this two modes assumption in Fig. 1. This qualitative performance is measured on the dataset GTA5 $\rightarrow$ Cityscapes, whereas the other dataset shows similar results.

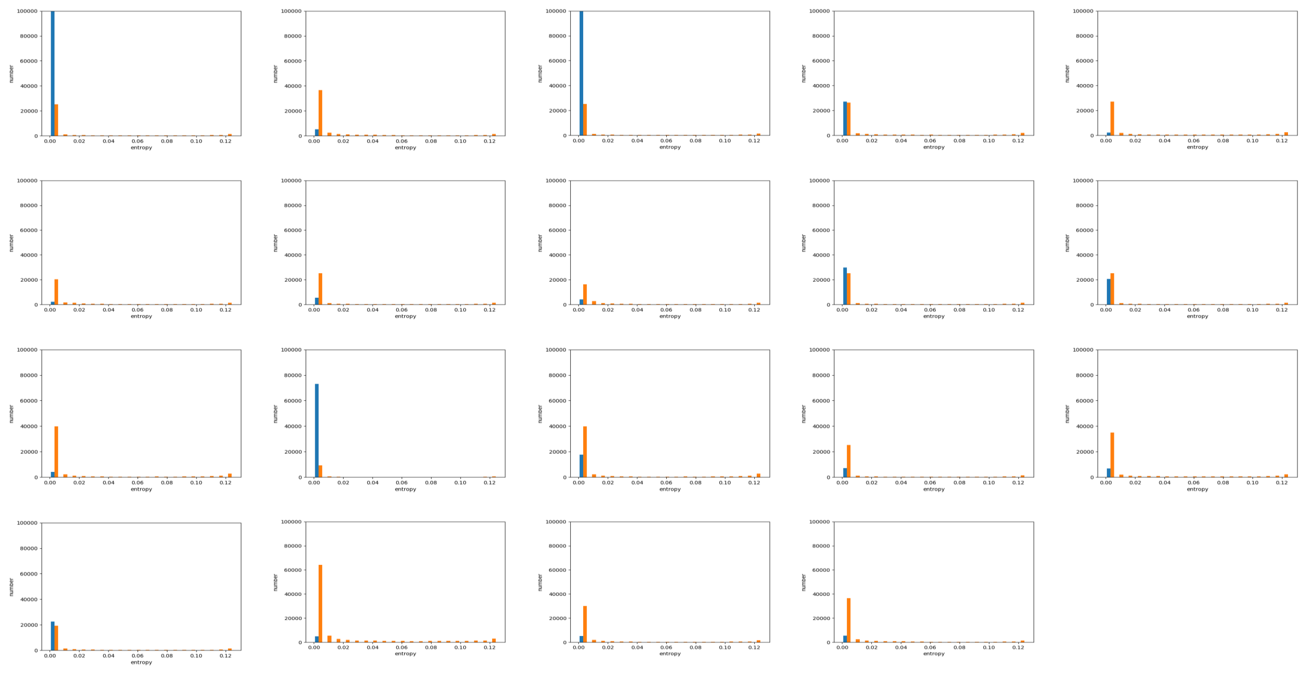

Figure 1. The distributions of positive (blue) and negative (orange) predictions on the whole 19 classes for a random image.

2. Appendix B. Algorithm

The training produce of the proposed UncerDA is summarized in Algorithm 1, which consists two stages. The first stage is related to *target-guided uncertainty rectifying*, which enhances feature alignment during adversarial adaptation process. In the second stage, we use *uncertainty-aware pseudo labels assignment* strategy to select positive pseudo labels and refine the model on the target data. For detailed equations and loss functions, please refer to main paper.

Algorithm 1: UncerDA

```
Data: training dataset: $(\mathcal{X}_s, \mathcal{Y}_s, \mathcal{X}_t)$; soft-balance parameters: $\lambda, \alpha, \mu$; uncertainty class number: $k$;
Result: the output model $F_{\theta_f}$
initialization;
Warmup: $F_\theta \leftarrow (\mathcal{X}_s, \mathcal{Y}_s, \mathcal{X}_t)$ according to [31];

for $stage \leftarrow 1$ to $s$ do
    Generate hard pseudo labels for the target data: $\hat{\mathcal{Y}}_t \leftarrow F_{\theta_f}(\mathcal{X}_t; \theta_f)$;
    for $c \leftarrow 0$ to $C$ do
        Calculate category-level entropy for the target dataset $\mathcal{I}^c_{\mathcal{X}_t}$;
    end
    Sort $\mathcal{I}^c_{\mathcal{X}_t}$ on the whole $C$ classes and select top-$k$ classes as a subset $S_k$ ;
    Calculate instance-level soft-balance sampling probability for the source data $\hat{p}_i(x_s)$;
    for $m \leftarrow 0$ to $epochs$ do
        for $i \leftarrow 0$ to $len(\mathcal{X}_t)$ do
            Get target images $x_t^{(i)}$ according to random sampling strategy;
            Get source images $x_s^{(i)}$ according to soft-balance sampling probability $\hat{p}_i(x_s)$;
            Train the model $F_{\theta_f} \leftarrow (x_s^{(i)}, y_s^{(i)}, x_t^{(i)}; \theta_f)$ using the adversarial adaptation (AA) method according to
              [31];
        end
    end
end

$\mathcal{P}$ is a set of positive prediction samples; $\mathcal{N}$ is a set of negative prediction samples;
Generate hard pseudo labels for the target data: $\hat{\mathcal{Y}}_t \leftarrow F_{\theta_f}(\mathcal{X}_t; \theta_f)$;
for $i \leftarrow 0$ to $len(\mathcal{X}_t)$ do
    for $c \leftarrow 0$ to $C$ do
        Calculate pixel-level entropy $I^c_{x_t^{(i)}}$ for the target image $x_t^{(i)}$ ;
        $\mathcal{B}, \mathcal{F} \leftarrow$ GMM($I^{(i)}_{x_t}$, 2) {$\mathcal{B}, \mathcal{F}$: Probabilities of two Gaussians for $I^{(i)}_{x_t}$};
        $\mathcal{P}, \mathcal{N} \leftarrow$ SeparateCategoryLevelEntropy($I^{(i)}_{x_t}$,$\mathcal{B}, \mathcal{F}$) {Separate entropy according to Fig.4} ;
        Get positive pseudo labels $\hat{\mathcal{Y}}_t^{(p)} \leftarrow \hat{\mathcal{Y}}_t \cap \mathcal{P}$, and negative predictions $\mathcal{N}$ are ignored;
    end
end
Train the model $F_{\theta_f} \leftarrow (\mathcal{X}_t, \hat{\mathcal{Y}}_t^{(p)}; \theta_f)$ using the self-training (ST) method.
```