We thank reviewers for the constructive comments. We are encouraged that our paper presents very efficient search algorithm [**R1**,**R2**,**R4**], our performance is significant [**R1**,**R2**,**R4**], the proposed search space is well-formulated and uniquely well-suited [**R2**,**R3**], our ablations are well-executed [**R1**,**R2**,**R3**], NAS for multi-component scenarios is worth investigating [**R4**], and our codes are complete with good reproducibility [**R1**,**R2**,**R4**]. All comments are addressed and technical details are provided below. We will release all models and further polish the documents.

Inter-modular search space [**R1**] We design a fine-grained inter-modular search space mainly because (i) there is a large discrepancy on the distribution between features of different levels from both branches and a level-to-level attention mechanism enables **input-specific** and **semantic-aware** feature calibration across branches, (ii) the proposed operation is highly efficient and incurs minor increase in parameters. To verify this, we keep only **vector** $a_{b\rightarrow f}$ and $a_{f\rightarrow b}$ (shared among levels) via Eq.1 after upsample the multi-level features in both branches, and repeat the overall searching process. The result on COCO val set is 44.0 PQ (+0.1 PQ vs no inter-modular, -0.7 PQ vs Auto-Panoptic).

Cityscapes[**R1**] Due to time limit, we reuse DetNAS backbone (Params/Flops same as **R50**) and execute **partial search** and achieves 59.8 PQ, which is satisfactory and is expected to obtain more performance gain in the full version, i.e., overall search with proper training protocols/hyper-parameters and larger backbone. We do not observe degradation under UPSNet setting (12K iter, imagenet pretrain) and **do not** use tricks like extra long training (EfficientPS, Panoptic-FPN, Seamless), more powerful pretrain (EfficientPS), delicate post-process (AUNet,BANet) and ms testing.

Longer training [**R1**] Under 3x schedule, our Auto-Panoptic achieves 45.2 PQ, while DetNAS backbone with 5x5 DF conv in both heads achieves 44.8 PQ, which is 0.4 PQ lower and is much slower due to the heavy head. This demonstrates the superior performance of Auto-Panoptic is not because 'it is slightly faster to train'.

Contribution to NAS literature[**R2**,**R3**] We want to emphasize our work aims to overcome immense amount of computation and severe convergence inefficiency when extending the current single-task NAS to the more complicated multi-component scenarios (extra large search space 6.7e33 vs DetNAS's 1.2e24). To achieve this, we propose customized search space (including problem-oriented inter-modular search space that enables flexible alignment across branches) and a highly efficient Path-Priority Policy ($\times$16.7 speedup with better performance).

Comparison to random baseline[**R2**] The error bars of the random baseline for 5 trials is (40.46$\pm$0.67). We do not have multiple trials due to the cost and will add it in the revised version. For eliminating the effect of search space, the Hand-crafted model in Table 4 clearly demonstrates the effectiveness of the search algorithm.

Hyperparamter[**R2**, **R3**] $E = 12$ is the least common multiple of $|P_1|, |P_2|, |P_3|$. Enumerating 12 models per cycle can ensure each path in different component has the same number of evaluation. $T = 5$ makes the enumerated models in Path-Priority roughly equals to that of one generation in EA (60 vs 50).

Alg.1/Fig.2 Clarification[**R3**,**R4**] Statement '$|P_1| \mid i$ **and** $\cdots$': i is divisible by $|P_i|_{i=1:3}$. This ensures each path in each component is trained equally. RANK function: we rank and score the paths according to their fitness, i.e., PQ. The scatter in Fig.2b plots the performance of enumerated models before retraining, while the best model found is retrained (c.f. Sec.4.1), thus these enumerated models have lower PQ since they are not fully trained.

Path-Priority (abbr. PP)[**R4**] The evaluate/rank process will not affect the sampling process. Consider a layer with 6 paths, EA uses uniform sampling and the probability that a path is not sampled after N samplings is $(\frac{5}{6})^N$. When $N = 6$, PP based on **fair sampling without replacement** can ensure all paths are evaluated exactly once, while in EA the probability of a path for not being sampled is 33.5%. Thus PP can cover the whole search space with much reduced computation. We use the PQ of the model as the performance indicator of the paths based on a greedy assumption: when the model is good, then its paths are good as well. Thus we assign the score of the paths based on $score_i = K - rank_i$ and **accumulate the scores over** T **cycles**. The paths with the highest score are picked to build up the best model.

Plain/Hand-crafted baseline[**R1**,**R4**] For Auto-Panoptic-Plain, we replace all layers in the searched head with 3x3 conv, which is to verify the effectiveness of the proposed search space for head. Regarding the Hand-crafted baseline, we follow R4's suggestion to fix DetNAS backbone and search for head architecture, which achieves 43.4 PQ on COCO val set. This result together with Table 3 demonstrate the effectiveness of Multi-Component Search.

Searching FPN/RPN/Detection Head [**R4**] For FPN, all layers are 1x1 conv and are used to adjust the channel number, they act more like fc layer slide over the feature map. There is only 1 conv in RPN and no conv in detection head (2fc box head). Adding them to the search space may bring limited gain and we do not search them for simplicity.

Inference time/Wall clock time[**R1**,**R2**] Inference: Auto-Panoptic(186ms) vs DetNAS-backbone-with-searched-head (209ms) vs UPSNet (171ms). Wall clock: supernet fine-tuning: 2 days, Path-Priority: 0.4 days (on 8 GPUs).

Experiment setup[**R1**] DF conv is not added to Panoptic-FPN (c.f. Table 2). We build inter-modular search space between the 5-th RPN level between the last level of semantic branch. We do not freeze BN during retraining.