We thank all reviewers for giving us the insightful comments.

Relation to prior works (To reviewer #1 and #3) We have stated the similarities between our method and other relevant works in our paper (line 82-88). The differences between our work and other previous relevant category level self-supervised methods lie in two aspects: **(1)** the way to generate positive samples. We use a novel positive sample discovery method which is totally different from all other methods, such as Deep(er) Cluster, Local Aggregation, etc. Our positive sample discovery algorithm makes the network obtain additional prior information in the training process, i.e., the transitivity of semantic consistency. **(2)** the proposed hard sampling strategy. As far as we know, we are the first to design hard sampling strategy in self-supervised contrastive learning. The above two differences are exactly the two main components of our method.

The way to construct kNN graph (To reviewer #1) Indeed, the kNN graph is constructed by computing pairwise similarities incrementally. For each iteration, we calculate the nearst k neighbours of each anchor sample and record the indices of them. Then we collect all positive samples by a Breadth-First Search algorithm. This process is reflected as Eq (3) and Figure 1 in our paper.

Explanation of the Graph Distance (To reviewer #1) The Graph Distance means the shortest path between two vertices in the kNN Graph (the weight of each edge is set to 1). We use the graph distance to give another understanding to our positive sample discovery algorithm, which is that the positive samples discovered by our algorithm can be regarded as those samples whose graph distances from v_i are less or equal than l (l is the layer to propagate).

Other Ablations (To reviewer #1) We have found that only using InvP loss (without instance loss) will result in low speed of convergence, which is due to the unstable neighbours discovered by random initialized network. Besides, only using InvP loss has the phenomenon of training oscillation. For the impact of the number of positive samples or negatives samples, we have empirically tested several groups of hyperparameters, which shows that the number of positive samples is not very sensitive when the value is between 30 and 70. We will add these results to our paper. We thank review #1 for giving some detailed advices to improve the quality of our paper, such as the descriptions of some architechtures and typos, we will adopt these advices to further improve the paper.

The accuracy for predicting the positive and negatives samples (To reviewer #3) For the accuracy of predicting the positive samples and negatives samples. We have provided such statistics in Table 1 (in Supplementary Material) and Figure 2 (a) (in our paper). Table 1 in Supplementary Material directly reflects the accuracy of predicting positive samples. Concretely, using the nearst neighbours (can be regarded as the positive samples when the training process has converged) to predict the label of each sample reaches an accuracy of 61.3%, surpassing other results, which shows the high quality of the neighbours (positive samples). In Figure 2 (a), we also show the similarity distribution of positive samples and negative samples, which can reflect the good quality of positive samples and negative samples discovered by our algorithm. We also give the qualitative analysis in Figure 2 (b).

Baseline (To reviewer #3) About the baseline, our baseline is the KNN method, i.e. directly using the nearst K samples as positive samples. We have compared our algorithm with the KNN algorithm in Sec 4.2.

Connection with MoCo (To reviewer #3 and #4) As for the problem of MoCo, since we propose a general unsupervised learning algorithm, our goal is to find more accurate and useful positive and negative samples to help the network learn more useful features, while MoCo solves the problem of saving the sample features more effectively. Therefore, they are actually two different and orthogonal problems. MoCo can be applied to the proposed method by replacing the memory bank with the momentum queue (we pointed out this in line 187, page 6 of our paper). Due to the effectiveness of the momentum queue, we believe a better performance can be obtained. However, the improvements are not caused by our algorithm, and thus this setup does not reflect the performance of our method. In our paper, we use memory bank because it is simple and efficient. By contrast, using momentum queue requires two feedforward passes which makes the training time longer.

About Hyper-parameters (To reviewer #3) The proposed method relies on hyper-parameters such as k,l,P. We have evaluated several groups of hyper-parameters. The contrastive learning often takes much more time to converge than ordinary supervised learning. Therefore, it is hard to test all these hyper-parameters thoroughly. The current chosen hyper-parameters have worked fairly well, and we believe a further hyper-parameter search might lead to better results.

Wider Architechtures (To reviewer #4) As demonstrated in many relevant works, wider networks such as ResNet50(2x) or ResNet50(4x) can improve the downstream performance, and it will be interesting to see these results. However, we think this improvement does not derive from the proposed algorithm directly. In this paper, we mainly focus on evaluating the performance of the method itself.

Results with 200 epochs (To reviewer #4) For results with 200 epochs, we have reported in Table 5 with linear projection head. It can be seen form Table 5 that with 200 epochs, our result still outperforms all other methods appeared in Table 1 (200 epochs). We will add these results in Table 5 to Table 1 to give a more clear comparison.