We would like to thank all reviewers evaluating the paper, and will fully address all the review concerns in the revision.

Re R#1: The given method improves upon Projection cGAN (PcGAN) in only some cases. From the current IS and FID results, we are only inferior to PcGAN on the ImageNet dataset. Especially, our method is much better than PcGAN on the VGG face dataset. We have recently tested VGG face using 2000 classes. The IS score of TAC-GAN and PcGAN are 109.04$\pm$2.44 and 79.51$\pm$1.03. The FID score of TAC-GAN and PcGAN are 13.79 and 22.42. The results suggest that our method is advantageous on fine-grained datasets in which classes are more close to each other.

Projection cGAN is simpler than the proposed method. Since the four players share the convolutional layers, our method only adds a FC layer to AC-GAN. Compared to projection cGAN, our method only has a slight increase in computational load because of calculation of two additional losses.

pacGAN + ACGAN. Thanks for the nice suggestion. As suggested by R#1, we combine pacGAN with AC-GAN and our TAC-GAN, and the results are reported in Table 1 and Fig 1. pacGAN is a great method that significantly increases the performance of AC-GAN, though the performance is still lower than our method in terms of both scores and visual quality. This indicates that the drawbacks in AC-GAN loss cannot be fully addressed by pacGAN. We can see that combining pacGAN and TAC-GAN increases the performance, suggesting that pacGAN and TAC-GAN are compatible.

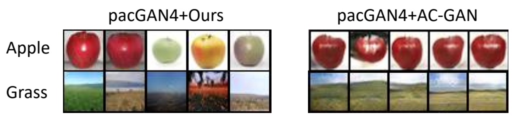

Figure 1: Generated Images

Metric \ Method	Ours	pacGAN4+Ours	AC-GAN	pacGAN4+AC-GAN
IS	9.34 $\pm$ 0.077	**9.85 $\pm$ 0.116**	5.37 $\pm$ 0.064	8.54 $\pm$ 0.143
FID	7.22	**6.79**	82.45	20.94

Table 1: IS and FID scores

Re R#2: We would like to thank R#2 for very detailed comments. **About Difference between ACGAN and cGAN.** We consider AC-GAN as a particular type of cGAN, as suggested also in the PcGAN paper. This is because the generator of AC-GAN and usual cGANs models the conditional distribution $p(x|y)$. The difference is how the discriminators match joint distributions of (X, Y) between generated and real data. This can be done in several ways as suggested in Figure 1 in the PcGAN paper (The tile of this figure is " Discriminator models for conditional GANs"). The usual cGAN concatenates X (or features of X) with Y and then use standard GAN loss to match joint distributions $p(x, y)$ (real) and $q(x, y)$ (fake). AC-GAN and PcGAN make use of the factorization $p(x, y) = p(y|x)p(x)$ to match the two factors separately, but AC-GAN has a imperfect loss that fails to match $p(y|x)$ with $q(y|x)$.

Low intra-class diversity for cGAN. The intra-class diversity of usual cGANs (concatenation) cannot be explained by the theorems in our paper. Our Theorems addresses the problems in AC-GAN loss. The usual cGANs have theoretically correct losses, and there are no clear answers to their bad performance. One hypothesis is that directly match the joint distributions of (x, y) by concatenation is hard. Both PcGAN and our method suggest that using the factorization $p(x, y) = p(y|x)p(x)$ and taking advantages of special structures in $p(y|x)$ is more effective.

What if not using biased batch sampling. If not using biased sampling, mutual information and JSD are not computationally equivalent, which will degrade the performance.

How is FID computed. We used the scripts in the BigGAN repository to calculate the scores. The FID scores were calculated on the entire dataset. We have also provided FID scores of each class in the Supplement.

Repeat experiments in Table 1 Following the procedures in PcGAN and the state-of-the-art BigGAN method, we did not repeat the experiments. We agree with R#2 that repeating the experiments is definitely much better to compare different methods, but we have limited HPC resources to repeat the experiments during a short rebuttal period.

VAE-GAN The generative & inference networks in bidirectional GANs (eg, VAE-GAN) construct the cycle-consistency term, which provides a bound to increase the entropy of the generated samples, thus improving (intra-class) diversity, as shown in Lemma 1 and Figure 3 of [1]. Cycle-consistency is often considered in unsupervised learning, while we propose the auxiliary classifier to explicitly improve intra-class diversity in class-conditional generation.

Re R#3: why choose KL over JSD. We choose KL because its nice connection to the cross-entropy loss. Using KL will make the algorithm much simpler because we only need classification losses to match the conditional distributions.

More clear explanation of the missing conditional entropy term. Yes, when updating the classifier C, the conditional entropy can be considered as a constant term. But when updating the generator G, it cannot be considered as constant because G is involved in this term. We will discuss these two situations in more detail in the updated version.

Regarding complexity and stability Please refer to the second answer to R#1 for explanationo of complexity. In all our experiments, we did not specifically tune the hyperparameter λ_c and we found our method pretty stable, as shown in Fig 6 in the main paper.

[1] Chunyuan Li, Ke Bai, Jianqiao Li, Guoyin Wang, Changyou Chen, and Lawrence Carin. Adversarial learning of sampler based on an unnormalized distribution.arXiv preprint arXiv:1901.00612, 2019.