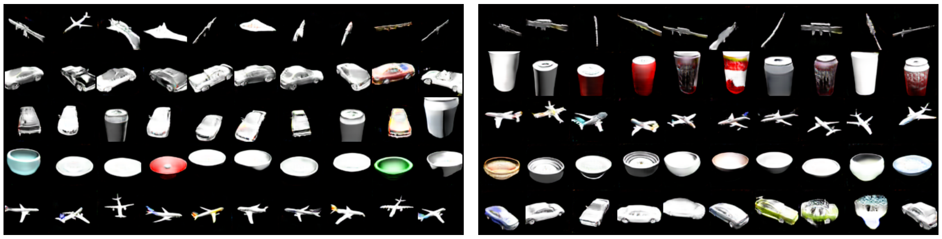
	MNIST		YTF		3D-cars		3D-chairs	
	NMI ↑	ENT ↓	NMI ↑	ENT ↓	NMI ↑	ENT ↓	NMI ↑	ENT ↓
JointVAE ($\beta = 10$)	0.536 ± 0.13	1.032 ± 0.29	0.372 ± 0.06	1.751 ± 0.03	**0.452 ± 0.24**	1.026 ± 0.43	0.392 ± 0.28	2.053 ± 0.46
JointVAE ($\beta = 20$)	**0.704 ± 0.08**	**0.661 ± 0.13**	0.421 ± 0.04	1.687 ± 0.02	0.441 ± 0.31	**1.022 ± 0.34**	0.431 ± 0.26	**1.817 ± 0.42**
JointVAE ($\beta = 30$)	0.680 ± 0.07	0.701 ± 0.13	0.447 ± 0.03	1.717 ± 0.02	0.391 ± 0.23	1.082 ± 0.53	0.448 ± 0.21	1.909 ± 0.44
JointVAE ($\beta = 40$)	0.676 ± 0.09	0.713 ± 0.14	**0.479 ± 0.02**	**1.662 ± 0.03**	0.324 ± 0.45	1.151 ± 0.51	**0.480 ± 0.24**	1.986 ± 0.41
JointVAE ($\beta = 50$)	0.649 ± 0.09	0.774 ± 0.15	0.435 ± 0.03	1.695 ± 0.02	0.376 ± 0.40	1.193 ± 0.25	0.377 ± 0.31	2.120 ± 0.30
InfoGAN + L_{ntxent}	0.838 ± 0.05	0.351 ± 0.09	0.712 ± 0.01	0.831 ± 0.02	0.617 ± 0.28	0.835 ± 0.52	0.438 ± 0.15	1.237 ± 0.32
Ours	**0.889 ± 0.04**	**0.213 ± 0.09**	**0.792 ± 0.01**	**0.636 ± 0.01**	**0.850 ± 0.07**	**0.303 ± 0.15**	**0.650 ± 0.08**	**0.765 ± 0.18**

Thank you for the helpful comments. We are encouraged that the reviewers found the problem setting important & unexplored (R2,3,4), and our solution effective & reasonable (R1,2,4) in overcoming issues of existing work (R1,3).

[R1] Clarifications regarding the VAE-based baseline We apologize for not including the hyperparameter details of JointVAE. We use the KL term for both continuous as well as discrete variables, to follow the standard normal ($\mathcal{N}(0,1)$) and uniform categorical distribution (Cat($p = 1/k$)), respectively. We use uniform categorical because of the unsupervised nature of the problem (L53-5, 83-4), and our full approach itself starts from uniform initialization (L127-30 supp). We use the same weight (β) for both KL loss terms (similar to JointVAE paper), the value of which was first decided empirically based on image reconstruction quality - we observed that a value in 100s (e.g. 100-300) resulted in poor reconstruction quality. We ultimately went with $\beta = 30$ (the value chosen by JointVAE for MNIST) for all datasets. We present an ablation study on the effect of strength of the KL term (β) on disentanglement in the table above. While we agree that a lesser weight on the KL term might imply lesser restriction for inference model to follow the uniform prior, it might result in reduced disentanglement as well. So, starting from a particular value (say $\beta = 30$), it is not clear whether increasing (towards uniform) or decreasing (towards less disentanglement) it would help from disentanglement's point of view. We observe this in the ablation study as well, where low and high values of the weight ($\beta = 10$ and $\beta = 50$ respectively) usually result in low disentanglement scores. We can, however, get slightly better results with alternate βs (different β for different datasets) than the ones reported in the main paper, and we'll update them with these in the final version. Note that for our approach we don't perform an exhaustive search for λ_2 (in L_{final}); it's set as 10 for all datasets (L124-5 in supp). Finally, comparison to FactorVAE is not directly applicable, as it can only capture continuous factors, and the paper itself mentions the inability to capture discrete factors as a limitation.

[R2] [R3] Concerns regarding technical novelty We agree we leverage existing techniques (L166-8). However, they have previously been used in orthogonal areas: Gumbel-softmax was introduced for differentiable sampling of one-hot like variables, and identity preserving transformations have been used as part of data augmentation, avoiding overfitting, representation learning, etc. In this work, we've integrated these techniques in a coherent framework to address an important problem in a *novel* setting of learning disentangled representations in class-imbalanced data (L169-70).

[R3] [R4] Concerns regarding datasets We'd like to point out that seminal works in learning disentangled representations (e.g. InfoGAN, β-VAE) present results on such small datasets, where it is relatively easy to account for, and capture the factors of variations. R3 states "..model tends to mode collapse on YTF" - we respectfully disagree: the categories in YTF consist of video frames of the same person, resulting in very similar *real images* themselves. Faithfully modeling such image distribution hence results in similar generated images. R3 further suggested to try the ShapeNet dataset, which is naturally imbalanced. The original dataset was too big to operate during the rebuttal phase (∼ 600k images, with 10 renderings per model), so we created a subset consisting of 5 categories - cars, airplanes, bowl, can, rifle - in a way which maintains the original imbalance between categories. Due to time constraints, we're only able to compare Uniform-InfoGAN and our final method. We can see that InfoGAN (NMI: 0.545, ENT: 0.687) mixes up different categories more frequently (rows 1/3) than our method (NMI: 0.781, ENT: 0.432), which is more consistent when grouping same category instances together. We'll include more analysis.


Uniform-InfoGAN

Ours

[R2] Clarifications regarding training components The first component, Gumbel-Softmax, should be thought of as making the latent distribution flexible, so that the model can capture *any* discrete factor having k modes (not necessarily object identity) present in any ratio. L_{ntxent}'s role is to push the discovered factor to better correspond with object identity (L162-3), in balanced or imbalanced case (as R2 points out). We include the results for Uniform-InfoGAN + L_{ntxent} in the table above. The results improve compared to Uniform-InfoGAN, but having the rigid uniform prior still results in worse performance compared to our approach. Hence, Gumbel-softmax alone *shouldn't* be thought of as an improvement, as it works best when used along with L_{ntxent} (L229-33). Furthermore, we cannot use class labels since this is an unsupervised task. Finally, we'll discuss the mentioned related works; we thank R2 for pointing them out.

[R4] Clarification regarding input for L_{ntxent} We agree and will discuss this in detail. *ntxent* is described in L159.