Directly Training Joint Energy-Based Models for Conditional Synthesis and Calibrated Prediction of Multi-Attribute Data

Jacob Kelly [1] Richard Zemel [1] Will Grathwohl [1]

Abstract

Multi-attribute classification generalizes classification, presenting new challenges for making accurate predictions and quantifying uncertainty. We build upon recent work and show that architectures for multi-attribute prediction can be reinterpreted as energy-based models (EBMs). While existing EBM approaches achieve strong discriminative performance, they are unable to generate samples conditioned on novel attribute combinations. We propose a simple extension which expands the capabilities of EBMs to generating accurate conditional samples. Our approach, combined with newly developed techniques in energy-based model training, allows us to directly maximize the likelihood of data and labels under the unnormalized joint distribution. We evaluate our proposed approach on high-dimensional image data with high-dimensional binary attribute labels. We find our models are capable of both accurate, calibrated predictions and high-quality conditional synthesis of novel attribute combinations.

1. Introduction

Multi-attribute classification is a more general form of classification where each data example has a set of labels. Models for multi-attribute prediction can be implemented similarly to models for single-attribute prediction, but face additional difficulties. Some attributes may be rare, causing issues related to severe class-imbalance. Attributes may be missing or only a subset may be observed for each example in the dataset. In these settings, making calibrated predictions and quantifying uncertainty is especially important.

Energy-Based Models (EBMs) present a flexible approach for representing uncertainty. Multi-attribute classifiers can be interpreted as Joint Energy-Based Models (JEM) (Grathwohl et al., 2019). JEM re-purposes existing state-of-the-art classifier architectures to define an energy-based model of the joint distribution $p(x, y)$ of continuous images x and discrete, 1-dimensional class labels y. JEM gives improved calibration, out-of-distribution detection, and adversarial robustness while retaining strong discriminative performance. Grathwohl et al. (2019) exploit the structure of the energy-function to marginalize out the label y to uncover an EBM for $\log p(x)$ and a normalized $\log p(y|x)$ model. Then, the model can be trained to maximize the factorized likelihood $\log p(x, y) = \log p(x) + \log p(y|x)$. Techniques for training EBMs on continuous data are used to maximize the first term and the second term is optimized to minimize cross-entropy.

However, training JEM this way presents challenges in conditional sampling, especially if we want to condition on rare or novel attribute combinations. We propose a simple alternative approach to training JEM that directly maximizes the joint distribution $\log p(x, y)$. This requires sampling from the joint which can be challenging given the mixed continuous-discrete nature of the sampling problem. Leveraging recent improvements in EBM training (Nijkamp et al., 2020a; Du & Mordatch, 2019; Du et al., 2020b), we find that we are able to directly train the joint model of data and labels. Training in this way retains the benefits of joint modelling such as accurate discriminative performance and improved calibration, while also enabling these models to generate high-quality conditional samples of rare and novel attribute combinations.

2. Energy-Based Models

An EBM parameterizes a probability distribution as

$$p_\theta(x) = \frac{e^{f_\theta(x)}}{Z(\theta)} \tag{1}$$

where $f_\theta : \mathbb{R}^D \to \mathbb{R}$ fully specifies the model and $Z(\theta) = \int e^{f_\theta(x)} \mathrm{d}x$ is the normalizing constant.

While the flexibility of EBMs make them appealing, this flexibility comes with the cost of making sampling and likelihood evaluation difficult. EBMs are typically trained with

[1]University of Toronto & Vector Institute. Code available at github.com/jacobjinkelly/gibbs-jem. Correspondence to: Jacob Kelly <jkelly@cs.toronto.edu>.

Presented at the ICML 2021 Workshop on Uncertainty and Robustness in Deep Learning., Copyright 2021 by the author(s).

gradient-based optimization using the following estimator for the gradient of the maximum likelihood objective:

$$\nabla_\theta \log p_\theta(x) = \nabla_\theta f_\theta(x) - \mathbb{E}_{p_\theta(x')}[\nabla_\theta f_\theta(x')]. \quad (2)$$

Use of this estimator requires generating samples from $p_\theta(x)$. Since exact sampling is intractable, we resort to using approximate samples generated with MCMC (Tieleman, 2008). Fortunately, a host of techniques have been developed to make training with MCMC efficient when using deep neural networks to define the energy-function (Du & Mordatch, 2019; Du et al., 2020b; Nijkamp et al., 2019; 2020a; Xie et al., 2016).

The preferred sampling approach for continuous data is Langevin Dynamics (Welling & Teh, 2011) which updates samples with

$$x_{t+1} = x_t + \frac{\epsilon^2}{2\lambda}\nabla_x f_\theta(x) + \epsilon\alpha, \qquad \alpha \sim N(0, I) \quad (3)$$

where the step-size ϵ, and the temperature λ are hyperparameters. Common values for image data are $\epsilon = 0.01$, $\lambda = \frac{1}{20{,}000}$. The low temperature is necessary to generate samples quickly enough for efficient training.

3. Joint Energy-Based Models

We are given data in the form of pairs (x, y) where $x \in \mathbb{R}^D$ and $y \in \{0,1\}^K$. Multi-attribute prediction models parameterize the conditional distribution as $p_\theta(y|x) = \prod_{k=1}^K p_\theta(y_k|x)$. A parametric function $f_\theta(x) : \mathbb{R}^D \to \mathbb{R}^{K\times 2}$ defines

$$p_\theta(y_k|x) = \frac{\exp(f_\theta(x)[k][y_k])}{\exp(f_\theta(x)[k][0]) + \exp(f_\theta(x)[k][1])} \quad (4)$$

where $f_\theta(x)[k][0], f_\theta(x)[k][1]$ are the logits for attribute k taking values 0 and 1, respectively. Following Grathwohl et al. (2019), we can use this same function f_θ to define an unnormalized model of the joint distribution over data points x and attributes y:

$$p_\theta(x, y) = \frac{\prod_{i=1}^K \exp(f_\theta(x)[k][y_k])}{Z(\theta)} \quad (5)$$

where $Z(\theta)$ is the unknown normalizing constant. From this joint modeling interpretation, the form of $p_\theta(y|x)$ remains the same.

3.1. Factored Energy-Based Models

We can analytically marginalize out the labels y in Equation 5 to obtain the unconditional distribution $p_\theta(x)$. Factoring the joint probability as $p_\theta(x, y) = p_\theta(x)p_\theta(y|x)$, we can train the first term with Equation 2 and the second term with standard cross-entropy (Grathwohl et al., 2019). Ideally, training with this factorization or maximizing the joint log-likelihood directly would be identical, but since MCMC does not give exact samples, our gradient estimator is biased. This bias encourages the implicit distribution of the approximate MCMC sampler toward the data distribution (Nijkamp et al., 2020b). Thus, the distribution to which we apply Equation 2 will impact the final model.

In particular, using the factorization from (Grathwohl et al., 2019) impacts the final model by making the attribute-conditional distribution $p_\theta(x|y)$ hard to sample from directly. Grathwohl et al. (2019) circumvent this issue by relying on $p_\theta(x|y) \propto p_\theta(x)p_\theta(y|x)$ to generate attribute-conditional samples. While this approach works for low-dimensional y, it does not scale when y is high-dimensional or highly structured. This is problematic if we wish to condition on a y which is rare or, perhaps, does not appear in the training data, as we may never generate an x where y will be sampled from $p_\theta(y|x)$.

3.2. Directly Training the Joint Energy-Based Model

To avoid these issues we propose to train by applying Equation 2 directly to the joint distribution $p_\theta(x, y)$ as

$$\nabla_\theta \log p_\theta(x, y) = \nabla_\theta f_\theta(x, y) - \mathbb{E}_{p_\theta(x', y')}[\nabla_\theta f_\theta(x', y')]$$

We find that training this way allows us to directly sample $x \sim p_\theta(x|y)$, allowing us to condition on rare and even novel attribute combinations. Of course, joint sampling from high-dimensional, unnormalized distributions is an incredibly difficult task but recent advances in gradient-based sampling and EBMs (Welling & Teh, 2011; Du et al., 2020c; Grathwohl et al., 2019; 2021) have demonstrated that accurate samples can be generated from large-scale EBMs. The main difficulty lies in generating samples from the joint distribution, which we describe in the following section.

4. Training

When x is held fixed, $p_\theta(y|x)$ is tractable and can be sampled from exactly. When y is held fixed, notice that

$$\begin{aligned}\log p_\theta(x|y) &= \log p_\theta(x, y) - \log p_\theta(y) \\ &= f_\theta(x, y) - C(y)\end{aligned} \quad (6)$$

where $C(y)$ is a constant that does not depend on x. Thus, $p_\theta(x|y)$ is an EBM defined on continuous x given by evaluating $f_\theta(x, y)$ with y fixed to the conditioning value. In this setting Langevin Dynamics has been successfully applied to generate samples.

We now introduce our approach to sample from $p_\theta(x, y)$ which we call Langevin-Within-Gibbs (LWG). LWG works similarly to Gibbs sampling where we iteratively sample $y_{t+1} \sim p_\theta(y|x_t)$ and then $x_{t+1} \sim p_\theta(x|y_{t+1})$. We can update our current sample y_t exactly since $p_\theta(y|x_t)$ is tractable.

However, the attribute-conditional distribution $p_\theta(x|y_{t+1})$ is unnormalized and intractable. Instead of sampling exactly from this conditional, we update our current sample x_t using a Markov transition kernel applied to $p_\theta(x|y_{t+1})$, which we denote $\mathcal{T}_c(x_{t+1}|x_t, y_{t+1})$. In practice, this amounts to holding y_{t+1} fixed and apply one step of Langevin dynamics to sample x_{t+1}.

It is likely that more involved approaches (Zhou, 2019) could lead to improved performance or more efficient sampling but we found this simple approach to work well for our applications. Pseudo-code for our proposed joint sampler can be seen in Algorithm 1. We name our approach Gibbs-JEM, which involves sampling from JEM with LWG.

Algorithm 1 Joint Sampling

```
**Input:** EBM $p_\theta(x, y) \propto e^{f_\theta(x,y)}$, initial distribution $p_0(x)$, number of steps $T$
**Output:** Approximate samples $x_T, y_T \sim p_\theta(x, y)$
Initialize samples $x_0 \sim p_0(x)$
$t = 0$
**while** $t < T$ **do**
    Sample $y_{t+1} \sim p_\theta(y_{t+1}|x_t)$ {Categorical}
    Sample $x_{t+1} \sim \mathcal{T}_c(x_{t+1}|x_t, y_{t+1})$ {Langevin, Eq. 3}
    $t = t + 1$
**end while**
Return $x_T, y_T$
```

5. Experiments

We train models on 64×64 images of shoes from UT Zappos50K (Yu & Grauman, 2014; 2017) and faces from CelebA (Liu et al., 2015). We train a supervised baseline with cross-entropy, and a JEM baseline trained as in Grathwohl et al. (2019). Both baselines and Gibbs-JEM use the same architecture; the models only differ in how they are trained. Training details are available in Appendix B.

5.1. Prediction

In Table 1 we examine the predictive performance of our model. We find that JEM and Gibbs-JEM achieve competitive accuracy and F1 scores while giving notably more calibrated predictions, and superior AUROC and AUPRC. We plot calibration diagrams and the Receiver Operating Characteristic and Precision-Recall curves of our model against the supervised baseline in Figures 1 and 2 respectively. Additional evaluations and explanation of micro and macro averages are available in Appendix C.

5.2. Conditional Synthesis

In addition to predictive performance, we find our models are also capable of high-quality conditional synthesis. We use a modified LWG sampler for conditional generation on a subset of the possible attributes, described in Appendix A.

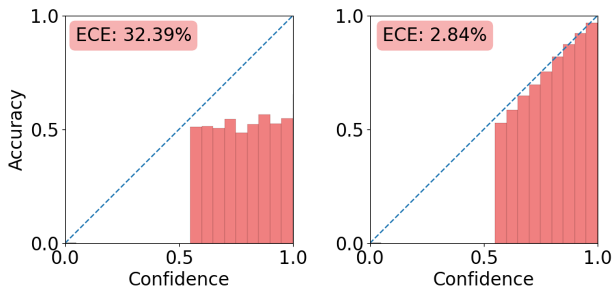


Figure 1. Calibration (micro-averaged) on CelebA. Supervised (left) vs. Gibbs-JEM (right). ECE is Expected Calibration Error (Guo et al., 2017).

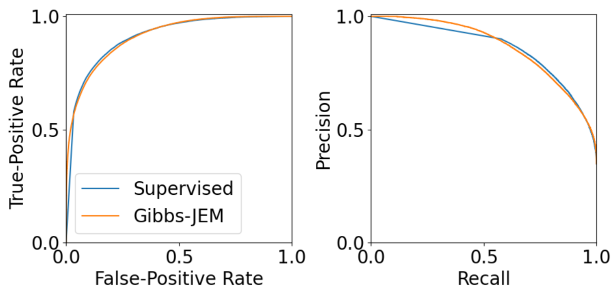


Figure 2. Micro-averaged Receiver Operating Characteristic (left) and Precision-Recall (right) curves on CelebA.

In Figures 3 and 4, we demonstrate the quality of our model's samples when conditioning on successively more attributes. We compare the quality of these samples to those from the JEM baseline and examine the quality of other samples in more detail in Appendix D.

Next, we trained a Gibbs-JEM model on CelebA while holding out several attribute combinations. In Figure 5, we plot samples from this model, conditioning on the held out attribute combinations. While sample quality has degraded, we find that the model is able to synthesize conditional samples of attributes it has seen separately, but never together, during training. A more thorough investigation of the model samples on held out attribute combinations is available in Appendix D.

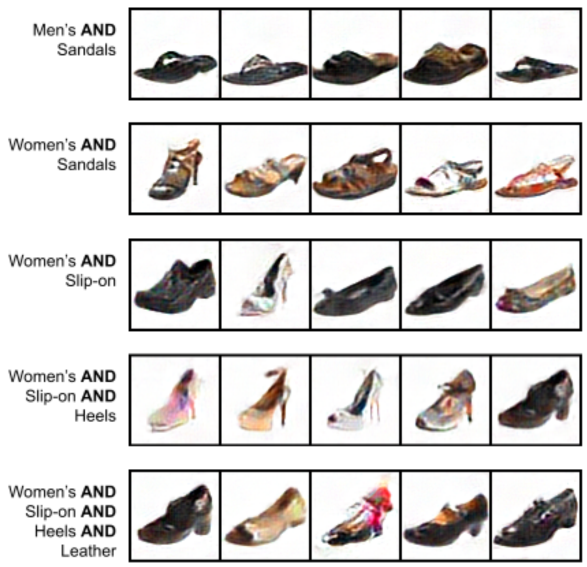
6. Related Work

Structured prediction problems have been a key application of EBMs. In this setting we wish to make predictions of highly structured y given inputs x. To capture complex correlations, this is often phrased in an energy-minimization framework (Gygli et al., 2017; Belanger et al., 2017) which has many similarities to the MCMC sampling we use. We believe our approach could be applied to these applications to add many of the benefits reported in Grathwohl et al. (2019).

Next are works that explore the unique capabilities of EBMs for challenging tasks such as continual learning (Li et al.,

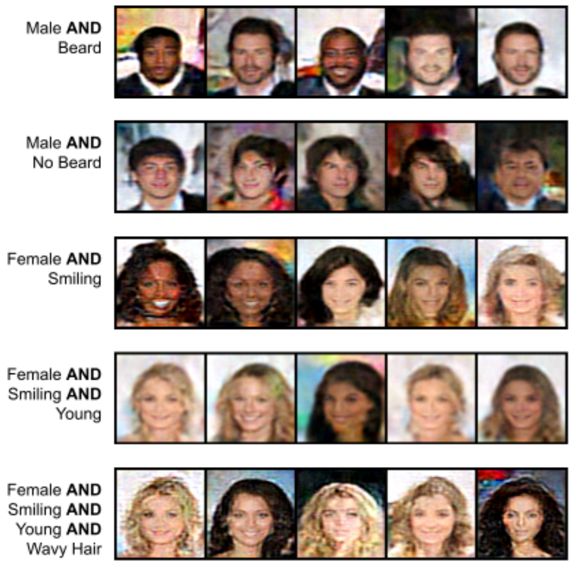
	UTZappos					CelebA				
	Accuracy	F1	AUPRC	AUROC	ECE	Accuracy	F1	AUPRC	AUROC	ECE
Supervised	**91.87**	**82.74**	88.19	95.97	26.39	84.53	77.07	82.77	90.89	32.39
JEM	90.05	78.68	86.73	94.32	7.428	**85.35**	**77.29**	**88.06**	**92.27**	**0.6987**
Gibbs-JEM	**91.81**	82.07	**91.53**	**96.60**	**0.2503**	84.14	74.64	86.39	91.17	2.838

Table 1. Predictive performance of Gibbs-JEM versus baselines. AUPRC is computed using Average Precision. ECE is Expected Calibration Error. F1, AUPRC, AUROC, and ECE are micro-averaged over attributes; macro-averages have the same rank-order and are available in Appendix C.



Men's **AND** Sandals
Women's **AND** Sandals
Women's **AND** Slip-on
Women's **AND** Slip-on **AND** Heels
Women's **AND** Slip-on **AND** Heels **AND** Leather

Figure 3. Conditional samples on UT Zappos50K.



Male **AND** Beard
Male **AND** No Beard
Female **AND** Smiling
Female **AND** Smiling **AND** Young
Female **AND** Smiling **AND** Young **AND** Wavy Hair

Figure 4. Conditional samples on CelebA.

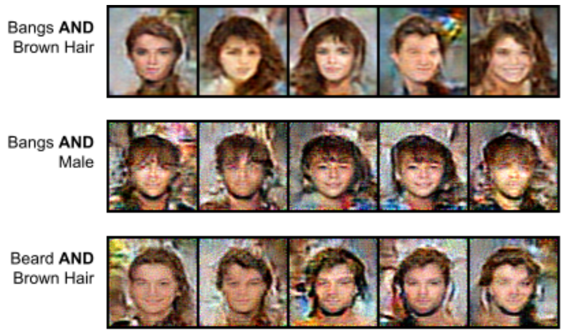


Figure 5. Samples of novel attribute combinations on CelebA.

2020) and compositional generation (Du et al., 2020a). We believe the techniques and architectures presented in this work could extend the range of problems these ideas can be applied to.

7. Conclusion

In this work we developed an approach to model the joint distribution of data and high-dimensional supervision using EBMs. We have demonstrated that our approach simultaneously achieves accurate and calibrated predictions and can perform high-quality conditional sampling, including of novel attribute combinations. Next steps include extending our approach to new data domains and settings with limited or partial supervision.

Acknowledgements

Resources used in preparing this research were provided, in part, by the Province of Ontario, the Government of Canada through CIFAR, and companies sponsoring the Vector Institute.

References

Belanger, D., Yang, B., and McCallum, A. End-to-end learning for structured prediction energy networks. In *International Conference on Machine Learning*, pp. 429–439. PMLR, 2017.

Du, Y. and Mordatch, I. Implicit generation and generalization in energy-based models. *arXiv preprint arXiv:1903.08689*, 2019.

Du, Y., Li, S., and Mordatch, I. Compositional visual generation and inference with energy based models. In *NeurIPS 2020*, 2020a.

Du, Y., Li, S., Tenenbaum, J., and Mordatch, I. Improved contrastive divergence training of energy based models. *arXiv preprint arXiv:2012.01316*, 2020b.

Du, Y., Meier, J., Ma, J., Fergus, R., and Rives, A. Energy-based models for atomic-resolution protein conformations. *arXiv preprint arXiv:2004.13167*, 2020c.

Elfwing, S., Uchibe, E., and Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *Neural Networks*, 107:3–11, 2018.

Grathwohl, W., Wang, K.-C., Jacobsen, J.-H., Duvenaud, D., Norouzi, M., and Swersky, K. Your classifier is secretly an energy based model and you should treat it like one. *arXiv preprint arXiv:1912.03263*, 2019.

Grathwohl, W., Swersky, K., Hashemi, M., Duvenaud, D., and Maddison, C. J. Oops i took a gradient: Scalable sampling for discrete distributions, 2021.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. On calibration of modern neural networks. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1321–1330. PMLR, 06–11 Aug 2017. URL http://proceedings.mlr.press/v70/guo17a.html.

Gygli, M., Norouzi, M., and Angelova, A. Deep value networks learn to evaluate and iteratively refine structured outputs. In *International Conference on Machine Learning*, pp. 1341–1351. PMLR, 2017.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del R'ıo, J. F., Wiebe, M., Peterson, P., G'erard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2. URL https://doi.org/10.1038/s41586-020-2649-2.

Hendrycks, D. and Gimpel, K. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Li, S., Du, Y., van de Ven, G. M., Torralba, A., and Mordatch, I. Energy-based models for continual learning. *arXiv preprint arXiv:2011.12216*, 2020.

Lipton, Z. C., Elkan, C., and Narayanaswamy, B. Thresholding classifiers to maximize f1 score, 2014.

Liu, Z., Luo, P., Wang, X., and Tang, X. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

Nijkamp, E., Hill, M., Zhu, S.-C., and Wu, Y. N. Learning non-convergent non-persistent short-run MCMC toward energy-based model. *arXiv preprint arXiv:1904.09770*, 2019.

Nijkamp, E., Hill, M., Han, T., Zhu, S.-C., and Wu, Y. N. On the anatomy of MCMC-based maximum likelihood learning of energy-based models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5272–5280, 2020a.

Nijkamp, E., Pang, B., Han, T., Zhou, L., Zhu, S.-C., and Wu, Y. N. Learning multi-layer latent variable model via variational optimization of short run MCMC for approximate inference. In *European Conference on Computer Vision*, pp. 361–378. Springer, 2020b.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., and Chintala, S. Pytorch: An imperative style, high-performance deep learning library. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32*, pp. 8024–8035. Curran Associates, Inc., 2019.

Ramachandran, P., Zoph, B., and Le, Q. V. Swish: a self-gated activation function. *arXiv preprint arXiv:1710.05941*, 7:1, 2017.

Tieleman, T. Training restricted boltzmann machines using approximations to the likelihood gradient. In *International Conference on Machine Learning*, pp. 1064–1071, 2008.

Welling, M. and Teh, Y. W. Bayesian learning via stochastic gradient langevin dynamics. In *Proceedings of the 28th international conference on machine learning (ICML-11)*, pp. 681–688. Citeseer, 2011.

Xie, J., Lu, Y., Zhu, S.-C., and Wu, Y. A theory of generative convnet. In *International Conference on Machine Learning*, pp. 2635–2644. PMLR, 2016.

Yu, A. and Grauman, K. Fine-grained visual comparisons with local learning. In *Computer Vision and Pattern Recognition (CVPR)*, Jun 2014.

Yu, A. and Grauman, K. Semantic jitter: Dense supervision for visual comparisons via synthetic images. In *International Conference on Computer Vision (ICCV)*, Oct 2017.

Zhou, G. Mixed hamiltonian monte carlo for mixed discrete and continuous variables. *arXiv preprint arXiv:1909.04852*, 2019.

A. Joint Energy-Based Models

A.1. Conditional Sampling

After holding fixed the attributes we'd like to condition on, we are left with the problem of drawing joint samples of the data and the remaining "free" attributes. We refer to this as semi-conditional sampling. Semi-conditional sampling can be seen as a form of unconditional sampling once we've fixed the attributes to be conditioned on. There are two approaches to unconditional sampling from our joint models. Thus there are two approaches to semi-conditional sampling. The first approach involves drawing joint samples x, y and then discarding the unneeded attributes y. We refer to this approach as "resampling", since the attributes y are being re-sampled. This form of sampling is how the model is trained. The other approach involves marginalizing out y and drawing unconditional samples from $p_\theta(x)$. We refer this approach as "marginalizing" since the attributes y are marginalized out. When applying these two approaches to unconditional sampling and semi-conditional sampling, the only difference lies in which attributes are "free", as those are the ones which can be resampled or marginalized out.

In the following, let c be the set of attribute indices we'd like to condition on, $\overline{c}$ its complement, y_c the attribute vector y with conditioned attributes c, and $y_{\overline{c}}$ the attribute vector y with free attributes $\overline{c}$. For example, suppose we'd like to condition on attributes 1 and 3 set to 1 and attribute 4 set to 0, and there are 5 attributes in total. Then $c = \{1, 3, 4\}, \overline{c} = \{2, 5\}$ and $y_c[1] = y_c[3] = 1$, $y_c[4] = 0$, and $y_c[i]$ is undefined for any $i \notin c$, $y_{\overline{c}}$ is undefined for any $i \in c$.

Resampling In Algorithm 2, we describe in more detail the procedure for semi-conditional sampling via resampling.

Algorithm 2 Semi-conditional Sampling (Resampling)

```
Input: EBM $p_\theta(x, y) \propto e^{f_\theta(x,y)}$, initial distribution $p_0(x)$, number of steps $T$, conditioning information $y_c$
Output: Approximate samples $x_T, y_T \sim p_\theta(x, y_{\overline{c}}|y_c)$
Initialize samples $x_0 \sim p_0(x)$
$t = 0$
while $t < T$ do
  for $i \in c$ do
    Copy $y_{t+1}[i] = y_c[i]$ {Conditioning attributes}
  end for
  for $i \notin c$ do
    Sample $y_{t+1}[i] \sim p_\theta(y_{t+1}[i]|x_t)$ {Free attributes}
  end for
  Sample $x_{t+1} \sim \mathcal{T}_c(x_{t+1}|x_t, y_{t+1})$ {Langevin, Eq. 3}
  $t = t + 1$
end while
Return $x_T, y_T$
```

Marginalizing Instead of resampling the attributes y which aren't being conditioned on, we can instead marginalize them out. We sample from the following distribution:

$$p(x, y_{\overline{c}}|y_c) \propto \left(\prod_{i \in c} \exp(f_\theta(x)[i][y_c[i]]) \right) \cdot \left(\prod_{i \notin c} \exp(f_\theta(x)[0]) + \exp(f_\theta(x)[i][1]) \right) \tag{7}$$

where the constant of proportionality is the normalizing constant. We write $\overline{\mathcal{T}}_c(x_{t+1}|x_t)$ for the Markov transition kernel which updates our current sample x_t with one step of Langevin Dynamics according to the energy in Equation 7. We outline the sampling procedure for semi-conditional sampling via marginalizing in Algorithm 3.

Algorithm 3 Semi-conditional Sampling (Marginalizing)

```
Input: EBM $p_\theta(x, y) \propto e^{f_\theta(x,y)}$, initial distribution $p_0(x)$, number of steps $T$, conditioning information $y_c$
Output: Approximate samples $x_T, y_T \sim p_\theta(x, y_{\overline{c}}|y_c)$
Initialize samples $x_0 \sim p_0(x)$
$t = 0$
while $t < T$ do
  Sample $x_{t+1} \sim \overline{\mathcal{T}}_c(x_{t+1}|x_t)$ {Langevin, Eq. 3}
  $t = t + 1$
end while
Return $x_T, y_T$
```

A.2. JEM for Multiple Attributes

We introduce JEM (Grathwohl et al., 2019) for classifiers of multiple binary attributes. We used a softmax parameterization for ease of notation, but this can be extended to use a single logit for each binary attribute instead of the two that we use.

Following Equation 5, we analytically marginalize out the binary attributes y to obtain the unconditional distribution:

$$\begin{aligned}
p_\theta(x) &= \sum_{y_1,\dots,y_K} p(x, y_1, \dots, y_K) \\
&\propto \sum_{y_1,\dots,y_K} \prod_{k=1}^{K} \exp f_\theta(x)[k][y_k] \\
&\propto \sum_{y_1} \cdots \sum_{y_K} \exp f_\theta(x)[1][y_1] \cdot \dots \cdot \exp f_\theta(x)[n][y_K] \\
&\propto \sum_{y_1} \exp f_\theta(x)[1][y_1] \cdot \dots \cdot \sum_{y_K} \exp f_\theta(x)[n][y_K] \\
&\propto \prod_{k=1}^{K} \sum_{y_k} \exp f_\theta(x)[k][y_k]
\end{aligned}$$

where the constant of proportionality is $Z(\theta)$ from Equation 5.

We can check that this joint interpretation gives the same parameterization of $p_\theta(y|x)$.

$$\begin{aligned}
p_\theta(y|x) &= \frac{p_\theta(x, y)}{p_\theta(x)} \\
&= \prod_{k=1}^{K} \frac{\exp f_\theta(x)[k][y_k]}{\sum_{y_k} \exp f_\theta(x)[k][y_k]} \\
&= \prod_{k=1}^{K} p_\theta(y_k|x)
\end{aligned}$$

This is the same as is given in Equation 4, so we are done.

B. Experimental Details

B.1. Training

We use the Adam optimizer with default parameters $\beta_1 = 0.9, \beta_2 = 0.999$ (Kingma & Ba, 2014) throughout our experiments. We used a learning rate of 10^{-4} throughout our experiments. We found that the standard hyperparameters of $\epsilon = 0.001$ and $\lambda = \frac{1}{20{,}000}$ worked well across datasets.

Sampling During training, we use Persistent Contrastive Divergence (PCD) and a replay buffer with a standard size of 10000 and replacement rate of 5% throughout all of our experiments (Du & Mordatch, 2019). At each sampling step, we clamp the sample to remain in the unit interval where the data lies, following Du et al. (2020b).

Sampling Noise The initial distribution for Langevin dynamics is uniform with bounds equal to the per-dimension minimum and maximum values of one batch of data.

Exponential Moving Average We keep an exponential moving average (EMA), as in Du et al. (2020b), of the training parameters θ being optimized by applying the following update after each training iteration:

$$\hat{\theta} = \mu \cdot \hat{\theta} + (1 - \mu) \cdot \theta.$$

We initialize $\hat{\theta}_0 = \theta_0$ and set $\mu = 0.999$ in our experiments. At test time we use the parameters $\hat{\theta}$.

Augmenting Sampling Chains Following Du et al. (2020b), we apply data augmentations to buffer samples when sampling during training. We applied blurring when training on UT Zappos50k, and blurring and flipping transforms on CelebA. We use the same transforms with the same parameters and probability of application as in Du et al. (2020b).

At test-time, for CelebA we apply augmentations every 60 steps. For UT Zappos50K we did not find data augmentations affected test-time sampling.

Missing Gradient Term in Contrastive Divergence We add the missing gradient term in contrastive divergence, as described in Du et al. (2020b). We estimate this loss term following Du et al. (2020b) by backpropagating through the final step of Langevin dynamics. We weight the KL loss term with the standard weighting of 0.3. We found that including this loss term greatly increases the diversity and quality of samples, especially when sampling from noise.

Reservoir Buffer Instead of the standard replay buffer, we use the reservoir buffer from Du et al. (2020b). Samples from the reservoir buffer approximate uniform samples from the model over the course of its entire training, in contrast to the replay buffer, whose samples are biased towards more recent training iterations. We used the reservoir buffer when training on UT Zappos50K. We found that it increased the stability of sampling at the cost of slowing down learning.

Joint Persistent Contrastive Divergence Persistent Contrastive Divergence initializes sampling chains from a reservoir of past samples. Since we are training our models with joint samples, we likewise maintain a buffer of joint samples. This involves maintaining a parallel buffer which tracks the predicted attribute configurations of each corresponding sample in the buffer. We can then use this buffer of joint samples to conditionally initialize sampling chains at test-time to the desired attribute combination.

Number of Langevin Steps in Langevin-Within-Gibbs We found that increasing the number of Langevin steps used with Langevin-Within-Gibbs significantly increased the stability of joint sampling during training, at the cost of slightly decreasing discriminative performance. In particular, we used 2 steps instead of 1 step when training on CelebA, and reduced the number of Langevin-Within-Gibbs steps by half so as to keep the total number of Langevin steps constant. We used 40 Langevin steps for both UT Zappos50K and CelebA.

Initializing Langevin-Within-Gibbs We track the attributes in our replay buffer and initialize samples from buffer over the joint of images and labels. We also found it very important for stability to first sample the attributes conditioned on the data. Sampling the data conditioned on the random attributes we found to be significantly less stable.

Stability Training these models was still less stable than ideal. Regularizing the norm of the energies of data and generated samples has been found to be useful for improving stability and allowing for the use of larger step sizes past (Du & Mordatch, 2019; Du et al., 2020b). In our experiments we were unable to regularized the norm of the energies of data and generated samples without severely impairing the performance of the model. We suspect that energy norm regularization prevents the discriminative performance of the model from improving. We were also unable to use large step sizes as used in Du et al. (2020b), as this caused our models to diverge very fast, even when regularizing the energy norms. For this reason, we used a step size of 1 throughout our experiments. We believe that figuring out how to make the use of larger step sizes more stable is crucial for improving the quality of our model's samples.

Optimization We trained models and kept the weights which had the largest accuracy on a held-out set. For JEM and Gibbs-JEM models, we additionally checked that sampling had not diverged. Interestingly, we found that occasionally model accuracy would improve after sampling had diverged, and the model themselves would diverge shortly thereafter. We had limited success restarting Gibbs-JEM models with lower learning rates or increased number of steps after divergence. Interestingly, in contrast to results from Grathwohl et al. (2019), we found JEM models to be quite stable in training. We suspect this is due to the new training techniques we applied that were developed in Du et al. (2020b).

Architecture We use a variant of the convolutional architecture with 64 filters for 64x64 images from Nijkamp et al. (2020b). We modified the final layer to be fully-connected and output logits according to the number of attributes. We replaced Leaky ReLU activations with the Swish activation (Hendrycks & Gimpel, 2016; Elfwing et al., 2018; Ramachandran et al., 2017), which is named `SiLU` in PyTorch. We used the following architecture for CelebA. The same architecture is used for UTZappos, except there are 38 outputs in the final layer (since UTZappos has 19 attributes, while CelebA has 23).

```
(cnn): Sequential(
  (0): Conv2d(3, 64,
        kernel_size=(3, 3),
        stride=(1, 1),
        padding=(1, 1))
  (1): SiLU(inplace=True)
  (2): Conv2d(64, 128,
        kernel_size=(4, 4),
        stride=(2, 2),
        padding=(1, 1))
  (3): SiLU(inplace=True)
  (4): Conv2d(128, 256,
        kernel_size=(4, 4),
        stride=(2, 2),
        padding=(1, 1))
  (5): SiLU(inplace=True)
  (6): Conv2d(256, 512,
        kernel_size=(4, 4),
        stride=(2, 2),
        padding=(1, 1))
  (7): SiLU(inplace=True)
  (8): Conv2d(512, 512,
        kernel_size=(4, 4),
        stride=(2, 2),
        padding=(1, 1))
  (9): SiLU(inplace=True)
  (10): Conv2d(512, 64,
        kernel_size=(1, 1),
        stride=(1, 1))
  (11): Flatten(start_dim=1,
        end_dim=-1)
)
(mlp): Sequential(
  (0): Linear(in_features=1024,
            out_features=128,
            bias=True)
  (1): SiLU(inplace=True)
  (2): Linear(in_features=128,
            out_features=128,
            bias=True)
  (3): SiLU(inplace=True)
  (4): Linear(in_features=128,
            out_features=46,
            bias=True)
)
```

Compute We used PyTorch (Paszke et al., 2019) and NumPy (Harris et al., 2020) throughout our experiments. We used NVIDIA T4 GPUs throughout our experiments.

B.2. Data Processing

We applied standard dequantization

$$\tilde{x} = \frac{x * 255 + u}{256}, u \sim \mathcal{U}[0, 1]$$

followed by adding Gaussian noise with standard deviation of 0.001, and clamping to the interval $[0, 1]$ to both UT Zappos50K and CelebA. We split 5000 random examples into a held out validation set for evaluation on both datasets.

On UT Zappos50K we filtered to attributes which had at least 10% frequency of positivie examples in the dataset, while in CelebA we filtered to 13%. On UT Zappos50K we also dropped the attributes “HeelHeight”, “Insole”, and “ToeStyle” as they had too many missing attributes. After dropping these attributes, attributes “Gender”, “Material”, “Closure”, “Category”, and “SubCategory” had some missing values. We dropped any examples in the dataset which had missing values for these attributes. Many attributes in UT Zappos50K are categorical. We binarized these attributes by treating a $1 - \text{of} - K$ one-hot categorical attribute as K binary attributes.

For the held-out combinations experiment on CelebA, we held-out 11 paired-attribute combinations. This resulted in a held out set of 42,803 examples. The held out combinations were as follows:

- Male AND Bangs
- Male AND Blond Hair
- Beard AND Bangs
- Beard AND Brown Hair
- Bangs AND Old
- Bangs AND Black Hair
- Bangs AND Brown Hair
- Bangs AND Blond Hair
- Old AND Black Hair
- Old AND Brown Hair
- Old AND Blond Hair

C. Evaluation Metrics

C.1. Averaging Across Attributes

We used two approaches to average metrics across attributes. Macro-averaging computes the metric over each attribute separately, and then computes the average over attributes. Micro-averaging treats all attributes as the same attribute, and computes the metric as if there were `number of examples` $\times$ `number of attributes` examples. See Lipton et al. (2014) for further discussion.

C.2. Additional Evaluations

We report macro-averaged metrics for both baselines and Gibbs-JEM in Table 2.

C.3. Additional Plots

We show calibration diagrams, Receiver Operating Characteristic (ROC) curves and Precision-Recall (PR) curves for individual attributes on CelebA and UT Zappos50K, and micro-averagred on UT Zappos50K.

We found that the supervised baselines were notably less calibrated across attributes and datasets. We hypothesize that it is much easier for the supervised baseline to be overconfident in its predictions. We tuned the model performance by taking the weights with the best accuracy on the validation set. We found that supervised baselines needed to be trained for significantly more iterations to converge to the best validation accuracy. We also tried weighting the cross-entropy loss according to the frequencies of attributes in the training set, but this decreased accuracy.

We also find that the ROC and PR curves for supervised baselines are slightly misleading in some regions. Many predictions of the supervised baseline have confidence numerically 1, as is expected given the poor calibration of these models. Thus, in the PR curves, using a threshold very close to 1, for example $1 - 10^{-7}$, does not result in the model achieving a recall of 0. In these cases, the plots linearly interpolate the PR curve in the region between where recall is 0 and the recall where the largest threshold less than 1 is used. Since there are many predictions with confidence 1, this recall value is often very large (we’ve observed values greater than 0.5). Thus, where this occurs in some plots care should be taken in how they are interpreted.

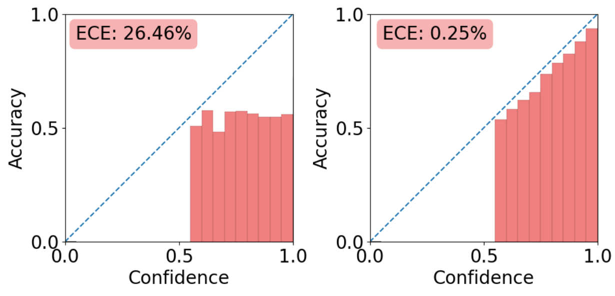


Figure 6. Calibration (micro-averaged) on UT Zappos50K. Supervised (left) vs. Gibbs-JEM (right).

	UTZappos				CelebA			
	F1	AUPRC	AUROC	ECE	F1	AUPRC	AUROC	ECE
Supervised	**78.85**	82.16	93.54	25.31	**69.93**	72.35	86.73	32.03
JEM	72.00	76.86	90.61	11.53	66.84	**75.64**	**88.16**	**1.421**
Gibbs-JEM	75.84	**84.14**	**94.09**	**0.9106**	60.50	74.60	87.24	3.391

Table 2. Macro-averaged metrics.

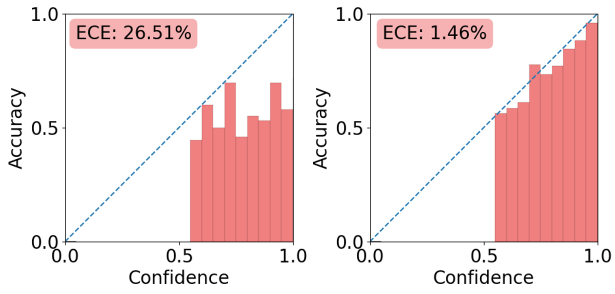


Figure 7. Calibration on attribute "Men's" on UT Zappos50K. Supervised (left) vs. Gibbs-JEM (right).

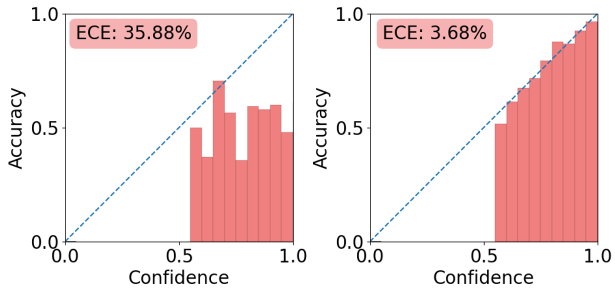


Figure 9. Calibration on attribute "Arched eyebrows" on CelebA. Supervised (left) vs. Gibbs-JEM (right).

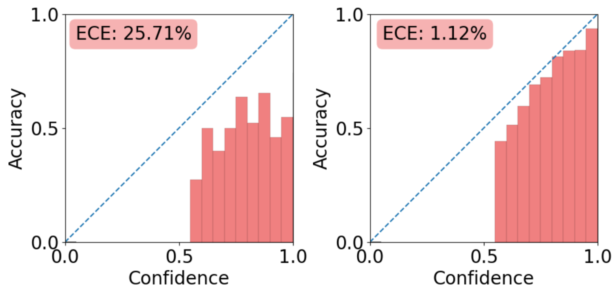


Figure 8. Calibration on attribute "Girls'" on UT Zappos50K. Supervised (left) vs. Gibbs-JEM (right).

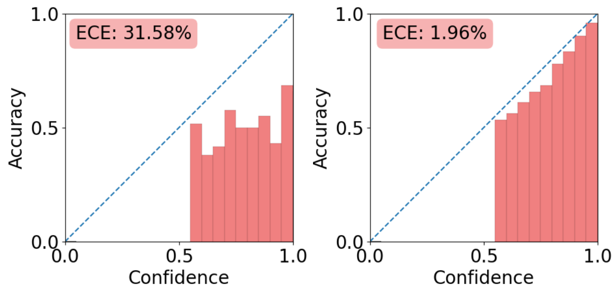


Figure 10. Calibration on attribute "Bags under eyes" on CelebA. Supervised (left) vs. Gibbs-JEM (right).

D. Additional Samples

D.1. Data Samples

In Figures 16 and 17, we plot random data samples from both datasets used in our experiments.

D.2. Samples from the Buffer vs. from Noise

During training with Persistent Contrastive Divergence (PCD), sampling chains are initialized from a replay buffer of chains. At test-time, we have the option of initializing samples from fresh noise, or from samples in the replay buffer used during training.

In Figures 18 through 21, we plot unconditional samples from our model from fresh noise and the buffer. We find that samples from fresh noise are of slightly lower quality and slightly less diverse than samples from the buffer, especially in the backgrounds of the images. We found that the techniques from Du et al. (2020b) significantly improved the quality and diversity of samples from noise. In Figures 22, 23, we compare fresh and buffer samples when conditioning on attributes. One approach we found useful for mitigating diverged samples was to filter samples using the model's likelihood. In particular, for a desired batch size N of samples, we generate $10N$ samples, then score each sample using the conditional $p_\theta(y_c|x)$ of our model, where y_c are the conditioning attributes. We then take the top N samples with the highest scores.

An important question to consider when initializing chains from the buffer for conditional sampling is whether there are samples of the desired attribute combination are available. We use a buffer of size $10,000$ throughout our experiments here, but CelebA has many more unique attribute combinations than there are samples in the buffer. Thus we cannot always rely upon initializing from the buffer to performing conditional sampling. On the other hand, samples generated from the buffer tend to be more diverse and of higher quality. so one question to ask is whether we can initialize unconditionally from the buffer for conditional sampling. That is, we take samples from the buffer without filtering to

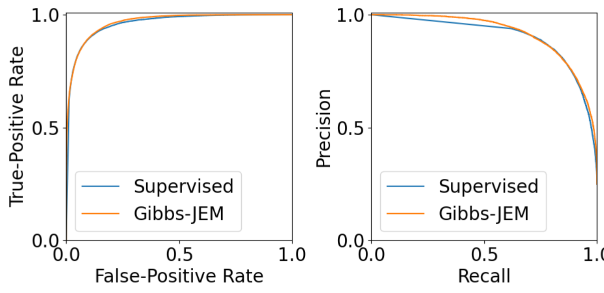


Figure 11. Micro-averaged Receiver Operating Characteristic (left) and Precision-Recall (right) curves on UT Zappos50K.

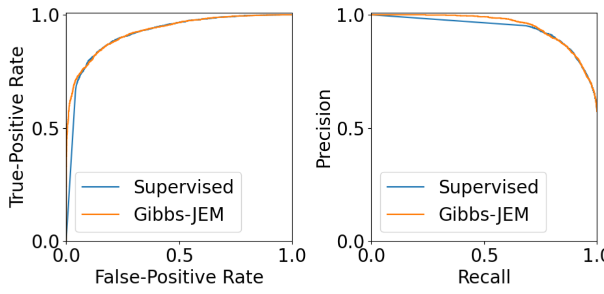


Figure 12. Receiver Operating Characteristic (left) and Precision-Recall (right) curves on UT Zappos50K for attribute "Women's".

samples with the conditioned attributes. This way, we can retain the benefits of improved sample quality while also conditioning on attributes not in the buffer. To accomplish this, we would need a model and sampling procedure which effectively mixes between modes with different attribute settings.

In Figures 24, 25. we examine conditional sampling when sampling conditionally and unconditionally from the buffer, respectively. We find that sampling is unable to consistently to mix well between modes with different attributes.

D.3. Conditional Sampling via Resampling vs. Marginalization

As discussed in Appendix A, we can draw conditional samples either by resampling or marginalizing out the free attributes y. In Figures 27, 26, 29, 28, we compare each sampling method on both fresh noise and resampling. One surprising result is that both sampling methods gives decent sample quality. In particular it is surprising that the bias in MCMC sampling as reported in Nijkamp et al. (2020b) does not cause sampling via marginalization to diverge, despite resampling being the sampling approach used during training. It is apparent however that marginalizing and resampling give qualitatively different samples. We found both sampling methods to work well for our application, and used both methods in different conditioning settings. In contrast, in Figure 30, we find that JEM is unable to generate conditional samples using marginalizing. We observed similar results when using resampling. This confirms our earlier

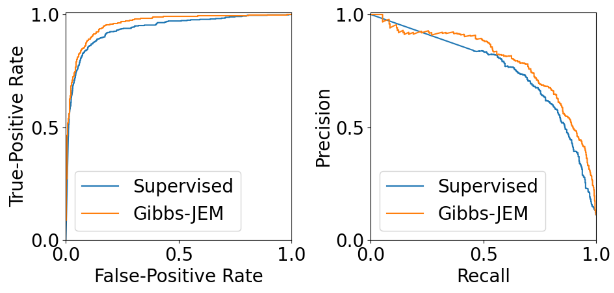


Figure 13. Receiver Operating Characteristic (left) and Precision-Recall (right) curves on UT Zappos50K for attribute "Mesh Material".

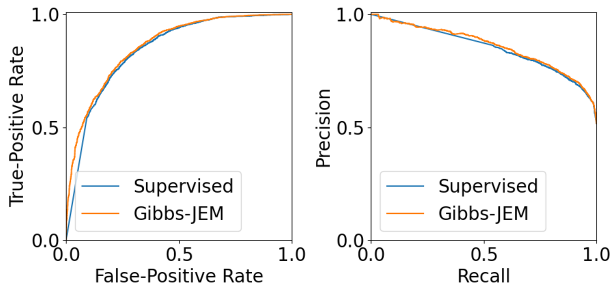


Figure 14. Receiver Operating Characteristic (left) and Precision-Recall (right) curves on CelebA for attribute "Attractive".

claim that the only reliable method to generate conditional samples is via the factorization $p_\theta(x|y) \propto p_\theta(x)p_\theta(y|x)$. That is, we can generate conditional samples from JEM only by sampling unconditionally from the model, and then classifying the samples using the $p_\theta(y|x)$ model.

D.4. How accurate is conditional sampling?

One sanity check for conditional sampling is to ensure that conditional samples agree with the conditional $p_\theta(y|x)$. That is, if we generate samples conditioned on attributes y_c via $p_\theta(x|y_c)$, how do we know those samples have the attribute y_c? The main method we used for determining this was visual inspection of the samples. A slightly better approach would be to use the supervised baseline to classify conditional samples, and measure the accuracy. Here we verify that the conditional samples are internally consistent. That is, we verify that when we generate samples via $p_\theta(x|y_c)$, our model classifies the samples as belong to y_c when using classifier of the same model $p_\theta(y_c|x)$. In particular, when conditioning on pairs of attributes, our model's conditional samples were classified correctly according to the model with an accuracy of $> 97\%$ over 100 samples for each pair conditioning. We found this result held across sampling from noise and from the buffer, and using resampling or marginalization for sampling.

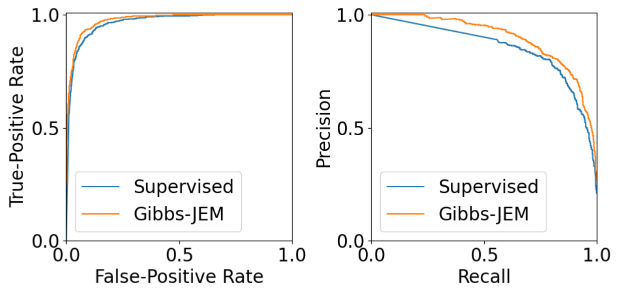


Figure 15. Receiver Operating Characteristic (left) and Precision-Recall (right) curves on CelebA for attribute "Blond Hair".

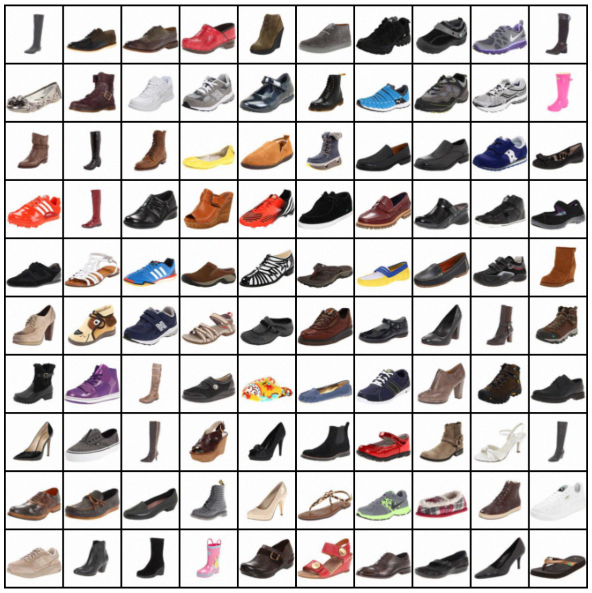


Figure 16. Unconditional data from UT Zappos50K.

D.5. Synthesizing Novel Attribute Combinations

One of the main motivations for our work is the ability of our model to synthesize novel attribute combinations. Here we investigate in more detail the performance of our model on this task.

As discussed previously, traditional Factored JEM models can only generate conditional samples by sampling unconditionally and then classifying the resulting samples. This method becomes very inefficient when we wish to sample attribute combinations not seen during training. On the 11 paired attribute combinations we held out during training, all novel attribute combinations occurred in less than 1% of samples. Similarly, we found that the held out attribute combinations occurred in less than 1% of samples in the buffer. While rare, we still have samples of novel attribute combinations in the buffer. We find however that these samples are not useful for initializing sampling chains and lead to poor sampling quality. We suspect this is because the samples from the buffer with the novel attributes have either diverged or are incorrectly classified by the model.

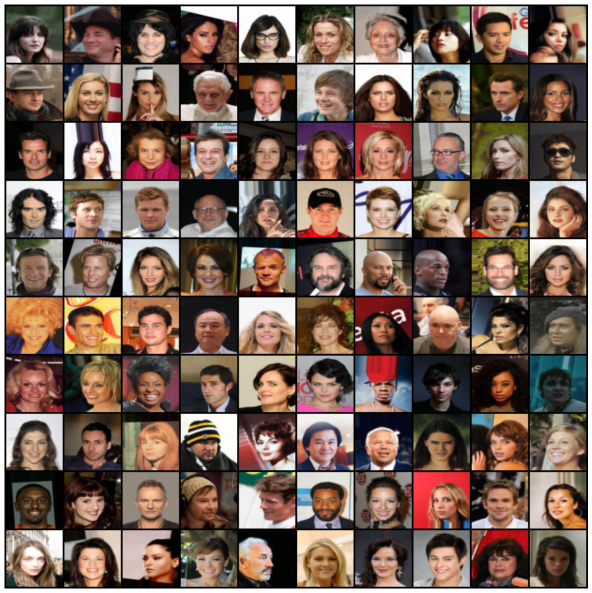


Figure 17. Unconditional data from CelebA.

We refer to CelebA with the held-out attribute combinations as Celeba-novel. In Figures 31 through 34 we show samples for the individual attributes whose combinations are held-out for test-time. We found that sample quality differed more significantly here between resampling and marginalization, and chose the option which looked the best for each attribute combination. In Figures 35 through 37, we show samples on novel attribute combinations.

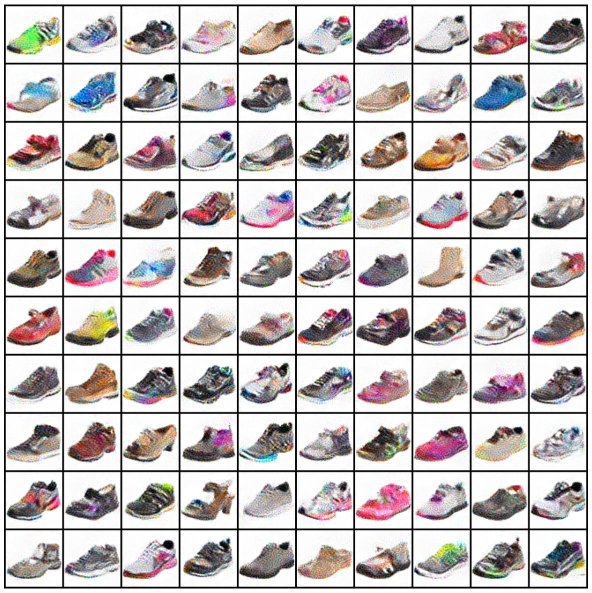


Figure 18. Unconditional samples from noise on UT Zappos50K.

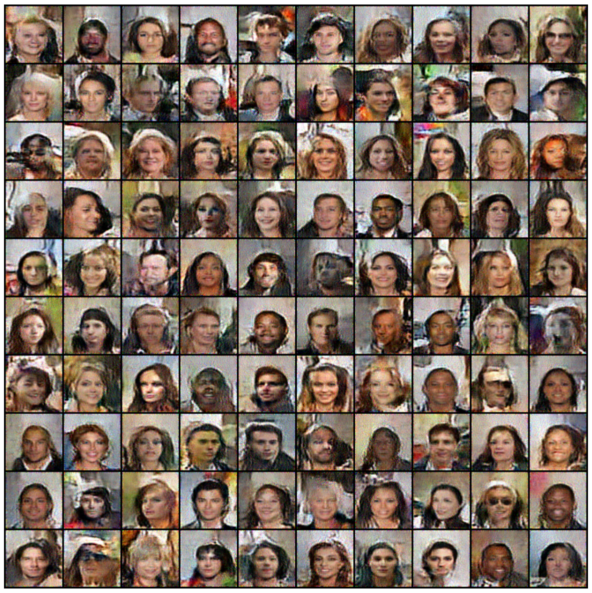


Figure 20. Unconditional samples from noise on CelebA.

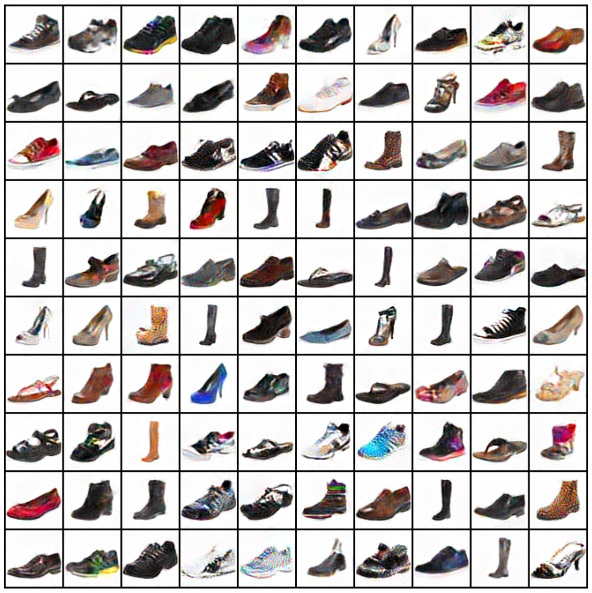


Figure 19. Unconditional samples from buffer on UT Zappos50K.

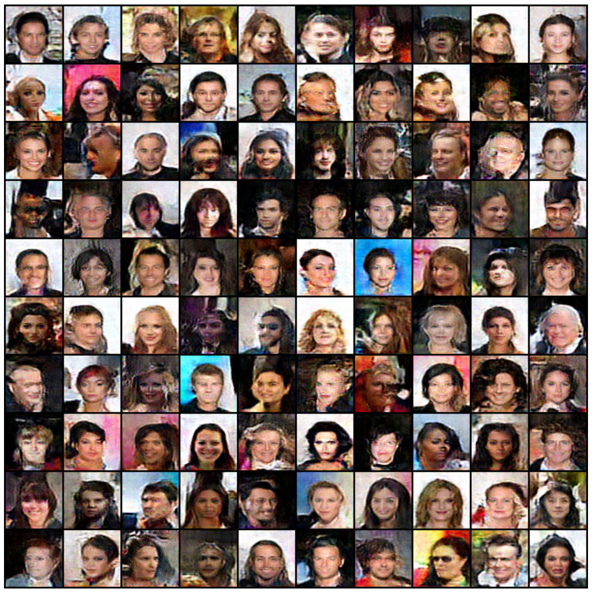


Figure 21. Unconditional samples from buffer on CelebA.

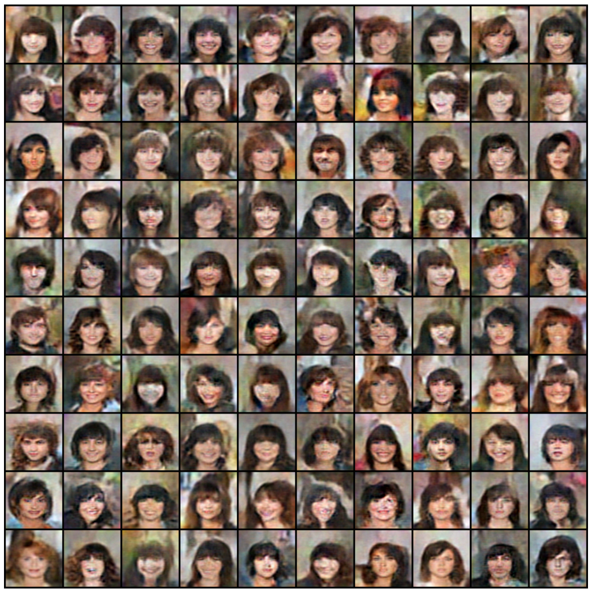


Figure 22. Conditional samples from noise on the attribute “Bangs” on CelebA.

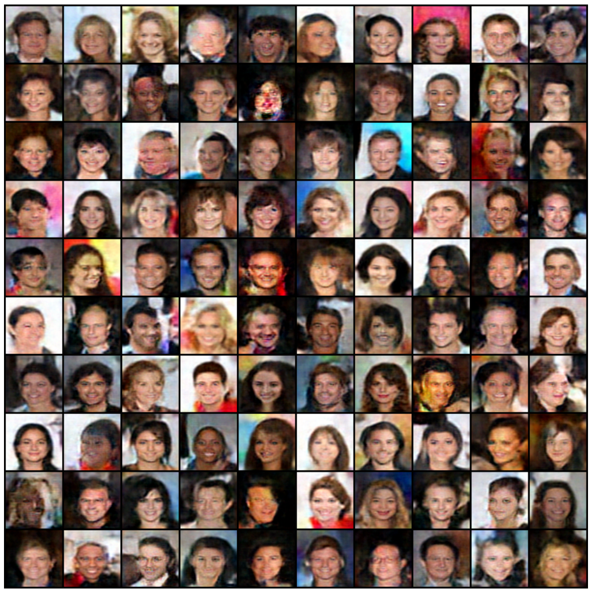


Figure 24. Conditional samples initialized conditionally from buffer on the attribute “Smiling” on CelebA.

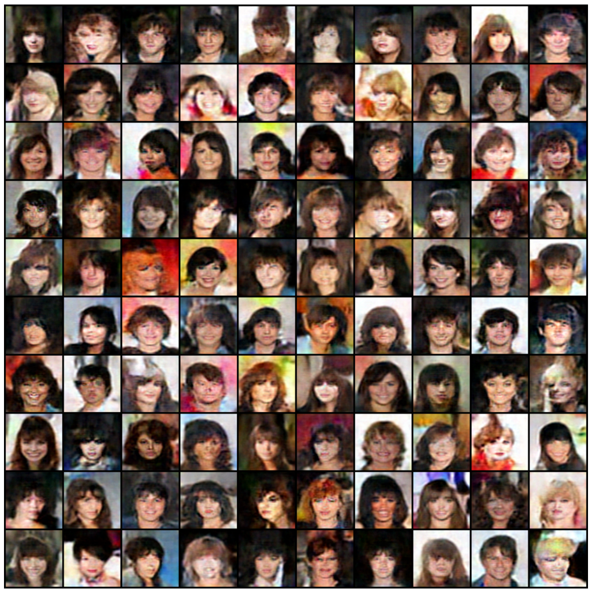


Figure 23. Conditional samples from buffer on the attribute “Bangs” on CelebA.

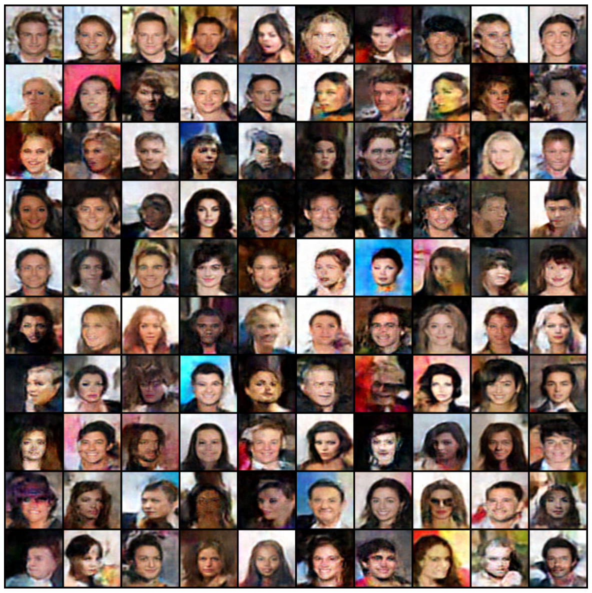


Figure 25. Conditional samples initialized unconditionally from buffer on the attribute “Smiling” on CelebA.

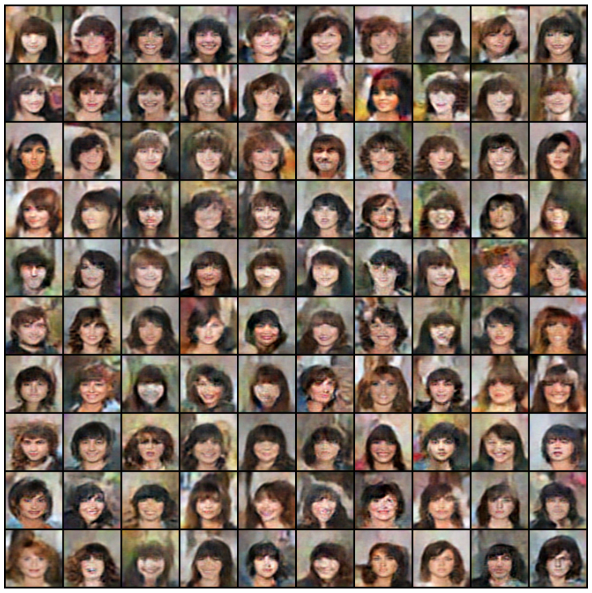


Figure 26. Conditional samples via resampling from noise on the attribute “Bangs” on CelebA.

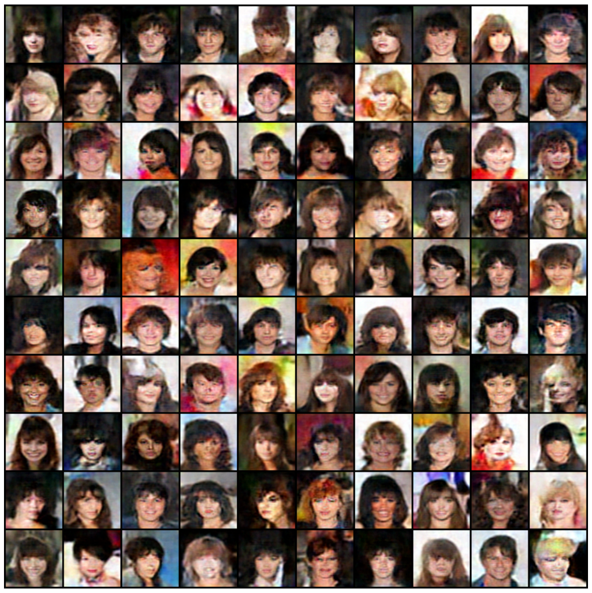


Figure 28. Conditional samples via resampling from buffer on the attribute “Bangs” on CelebA.

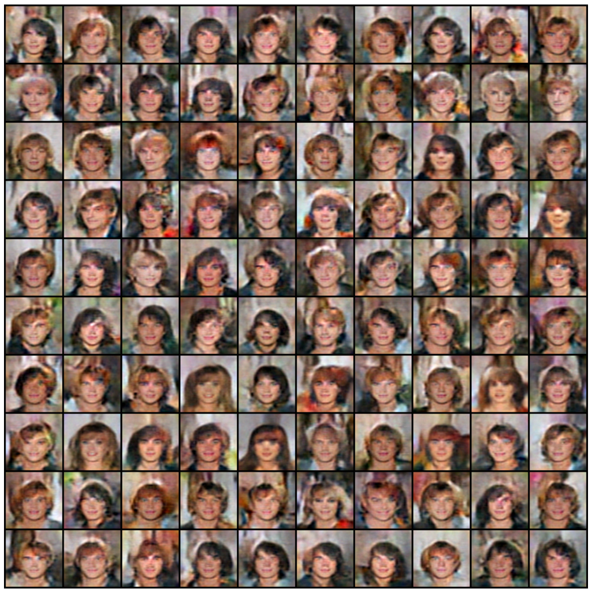


Figure 27. Conditional samples via marginalization from noise on the attribute “Bangs” on CelebA.

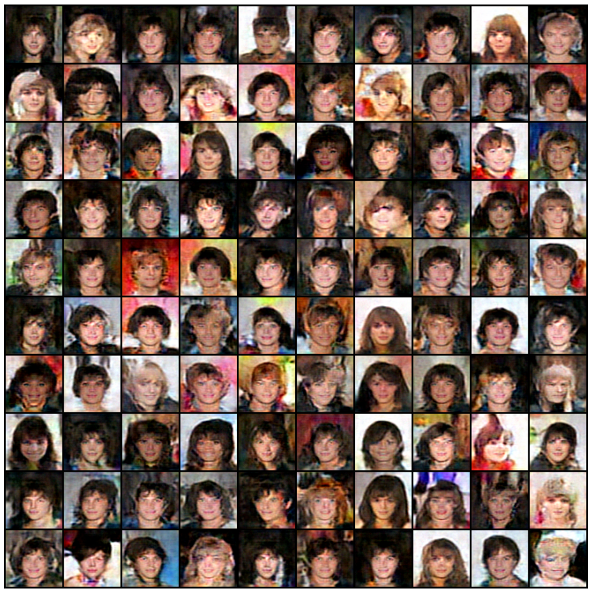


Figure 29. Conditional samples via marginalization from buffer on the attribute “Bangs” on CelebA.

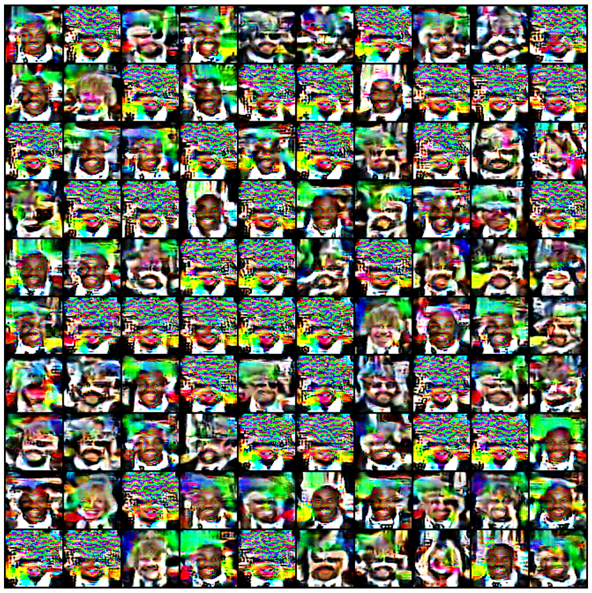


Figure 30. JEM conditional samples via marginalization from noise on the attribute “Smiling” on CelebA.

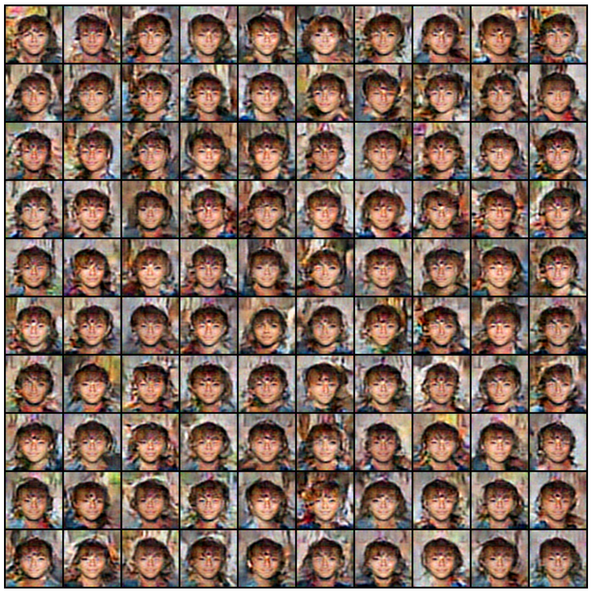


Figure 31. Conditional samples via marginalization from noise on the attribute “Bangs” on CelebA-novel.

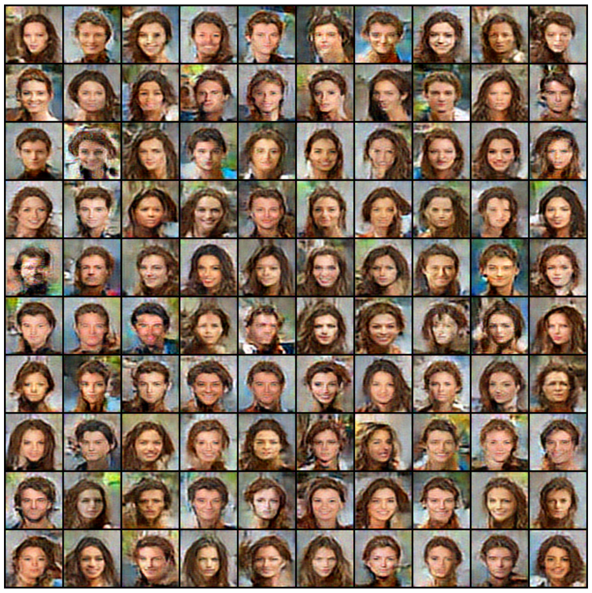


Figure 32. Conditional samples via resampling from noise on the attribute “Brown Hair” on CelebA-novel.

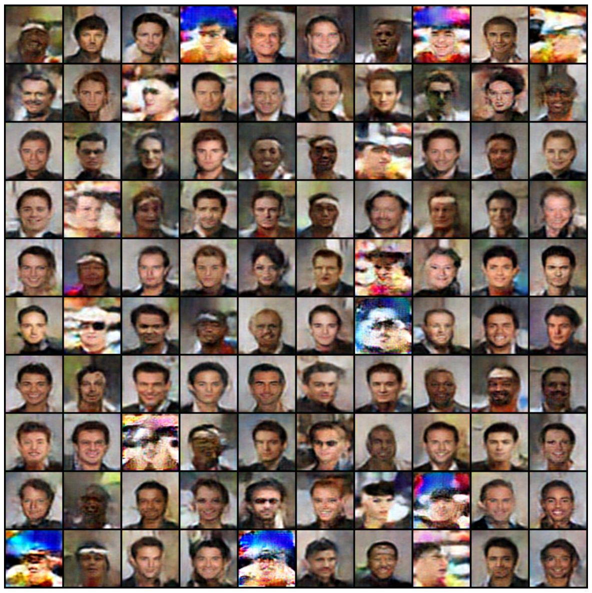


Figure 33. Conditional samples via marginalizing from noise on the attribute “Male” on CelebA-novel.

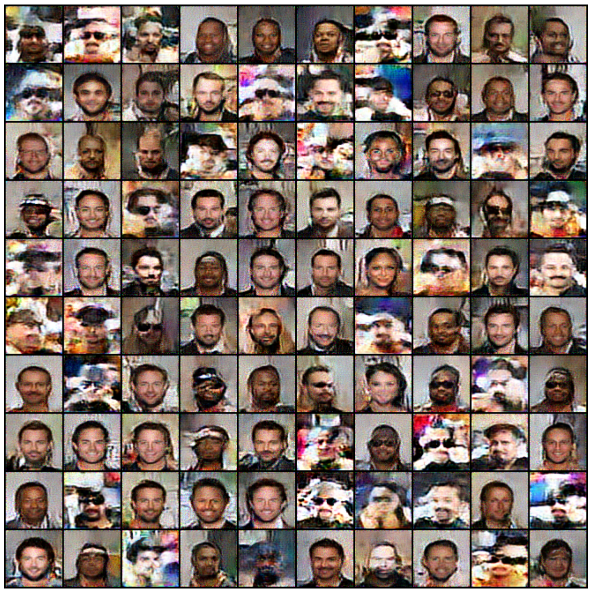


Figure 34. Conditional samples via marginalizing from noise on the attribute "Beard" on CelebA-novel.

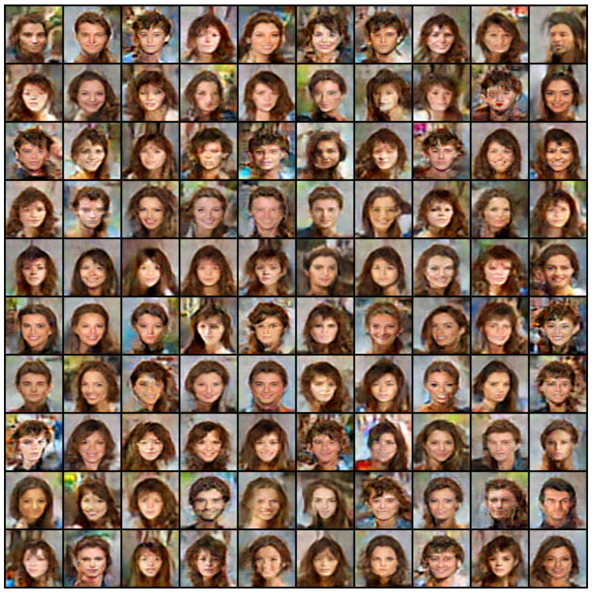


Figure 35. Conditional samples via resampling from noise on the held-out attributes "Bangs" and "Brown Hair" on CelebA-novel.

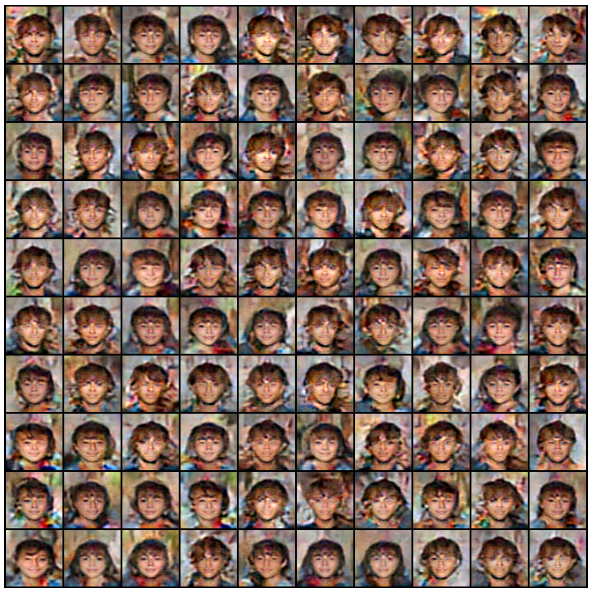


Figure 36. Conditional samples via marginalizing from noise on the held-out attributes "Bangs" and "Male" on CelebA-novel.

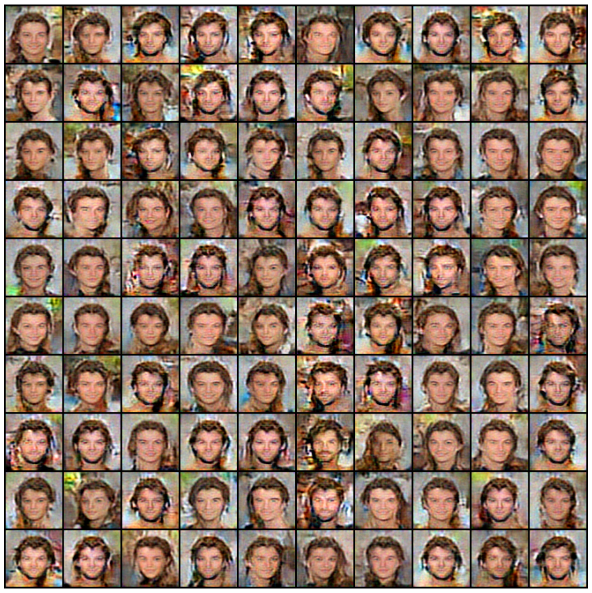


Figure 37. Conditional samples via marginalizing from noise on the held-out attributes "Beard" and "Brown Hair" on CelebA-novel.