On the Importance of Learning Aggregate Posteriors in Multimodal Variational Autoencoders

Chris Cremer CCREMER@CS.TORONTO.EDU
Nate Kushman NKUSHMAN@MICROSOFT.COM

Abstract

We study latent variable models of two modalities: images and text. A common task for these multimodal models is to perform conditional generation; for instance, generating an image conditioned on text. This can be achieved by sampling the posterior of the text then generating the image given the latent variable. However, we find that a problem with this approach is that the posterior of the text does not match the posteriors of the images corresponding to that text. The result is that the generated images are either of poor quality or don't match the text. A similar problem is also encountered in the mismatch between the prior and the marginal aggregate posterior. In this paper, we highlight the importance of learning aggregate posteriors when faced with these types of distribution mismatches. We demonstrate this on modified versions of the CLEVR and CelebA datasets.

1. Introduction

We explore variational autoencoders (VAEs) of images and text. We find that one difficulty when learning these types of multimodal models is that the aggregate of the image posteriors corresponding to the same text often does not match the posterior of the text. Written differently, the problem is that $p(z|y) \neq \mathbb{E}_{p_D(x_y)}[p(z|x)]$, where z is the latent variable, x is the image, y is the text, and $p_D(x_y)$ is the data distribution of images with the same text y. Fig. 1 demonstrates this mismatch by visualizing the true posterior of the text and the aggregate of the image approximate posteriors in a VAE with a 2D latent space. We can see that there are many holes in the aggregate posterior which indicates that if we sample the text posterior $p(z|y)$, then some generated images may be poor because we could be sampling within the holes of the aggregate image posterior.

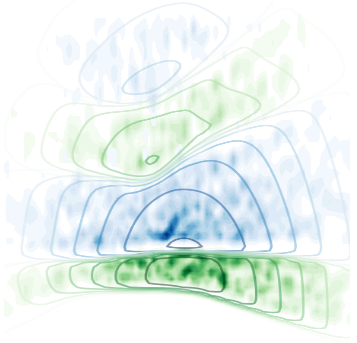

Figure 1: Distribution mismatch between true posterior and aggregate posterior. The large contours are the true posteriors of the text, $p(z|y)$. The smaller contours within are the aggregate approximate posteriors of the images corresponding to the text, $\mathbb{E}_{p_D(x_y)}[q(z|x)]$. Blue and green represent two different texts.

This problem is analogous to the prior mismatch problem: $p(z) \neq \mathbb{E}_{p_D(x)}[p(z|x)]$. It is well known that matching the marginal posterior and the prior is challenging (Rosca et al., 2018) and there have been steps towards improving this by either learning the prior during training (Tomczak and Welling, 2017) or learning more flexible posteriors (Jimenez Rezende and Mohamed, 2015; Makhzani et al., 2015; van den Berg et al., 2018;

Takahashi et al., 2018). In this paper, we highlight this distribution mismatch problem in multimodal latent variable models and explore the benefit of learning aggregate posteriors with flow distributions. We also relate this approach to current multimodal VAE objectives, such as JMVAE (Suzuki et al., 2016) and TELBO (Vedantam et al., 2017).

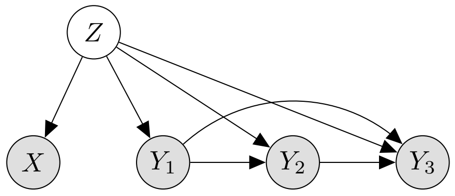

Figure 2: Graphical model of a Vision-Language VAE, where X represents images and Y represents text. We use three words (Y_1, Y_2, Y_3) to represent the auto-regressive text decoder.

2. Background

We refer to a VAE that generates images and text as a Vision-Language VAE (VLVAE). See Fig. 2 for the graphical model of a VLVAE. The model includes two decoders: one for images and one for text, where the text decoder is auto-regressive. We can learn the generative model of Fig. 2 by maximizing the following lower bound of the joint log-likelihood:

$$\log p(x,y) \geq \mathbb{E}_{q(z|x)}\left[\log\left(\frac{p(x|z)p(y|z)p(z)}{q(z|x)}\right)\right] \tag{1}$$

$$= \mathbb{E}_{q(z|x)}\left[\log\left(p(x|z)p(y|z)\right)\right] - \mathrm{KL}\left(q(z|x)||p(z)\right) \tag{2}$$

Notice that we've made a design choice to perform inference using only x. We chose this so that the latent variable does not include information specific to text, such as syntax. The latent variable ends up modelling the semantics in the text and leaves the auto-regressive decoder $p(y|z)$ to handle the syntax. See Fig. 7 in the appendix for an example of the difference between joint inference (using x and y) and marginal inference (using just x). This choice is similar to works which hinder the decoder in order to select the information that the latent variable encodes (Chen et al., 2016).

To perform approximate inference with only text, we can learn $q_\phi(z|y)$ such that it approximates the true posterior $p(z|y)$. This is achieved by optimizing the following lower bound:

$$\log p(y) \geq \mathbb{E}_{q_\phi(z|y)}\left[\log\left(\frac{p(y|z)p(z)}{q_\phi(z|y)}\right)\right] \tag{3}$$

Importantly, when optimizing this lower bound, optimization is done wrt ϕ only, not the model parameters of $p(y|z)$. This is the approach taken in the TELBO objective of Vedantam et al. (2017).

3. Learning the Aggregate Posteriors

Rather than optimizing $q_\phi(z|y)$ to approximate the true posterior $p(z|y)$, we can train $q_\phi(z|y)$ to approximate the aggregate of the image posteriors corresponding to the same y:

$\mathbb{E}_{p_D(x_y)}[p(z|x)]$, where x_y is the set of images with the same caption y. Since we don't have access to $p(z|x)$, we will instead model the aggregate of the image *approximate* posteriors corresponding to the same y:

$$q^y(z) = \mathbb{E}_{p_D(x_y)}[q(z|x)] \tag{4}$$

We can learn to approximate $q^y(z)$ by minimizing the following divergence wrt ϕ: $\mathbb{E}_{p_D(x_y)}[KL(q(z|x)||q_\phi(z|y))]$. To see that this is correct, we will decompose the KL as is done in Hoffman and Johnson (2016) and Vedantam et al. (2017):

$$\mathbb{E}_{p_D(x_y)}\left[\underbrace{KL(q(z|x)||q_\phi(z|y))}_{\text{Posterior x to Posterior y}}\right] = \mathbb{E}_{p_D(x_y)}\left[\underbrace{KL(q(z|x)||q^y(z))}_{\text{Posterior x to Aggregate y}} + \underbrace{KL\left(q^y(z)||q_\phi(z|y)\right)}_{\text{Aggregate y to Posterior y}}\right] \tag{5}$$

From this decomposition, we see that minimizing $KL(q(z|x)||q_\phi(z|y))$ wrt ϕ effectively minimizes $KL(q^y(z)||q_\phi(z|y))$, which is what we desire. See section 5.6 for a derivation of Eqn. 5.

In the JMVAE model of Suzuki et al. (2016), their objective effectively combines Eqns. 1 and 5 and optimizes all parameters jointly. To some extend, the JMVAE objective causes $q_\phi(z|y)$ to model $q^y(z)$, however it also incentivizes $q(z|x)$ and $q_\phi(z|y)$ to be closer, which is likely undesirable if $q(z|x)$ and $q_\phi(z|y)$ are very different. In other words, the JMVAE objective applied to the VLVAE model will attempt to minimize KL $(q(z|x)||p(z|x))$ as well as KL $(q(z|x)||q_\phi(z|y))$.

In addition to approximating $q^y(z)$, we can also approximate the marginal approximate distribution: $q(z) = \mathbb{E}_{p_D(x)}[q(z|x)]$. We can approximate $q(z)$ with $q_\phi(z)$ by maximizing the following objective wrt ϕ: $\mathbb{E}_{q(z|x)}[\log q_\phi(z)]$. In our experiments, we use flexible flow distributions for $q_\phi(z)$ and $q_\phi(z|y)$. These flow distributions are further described in Section 5.1.1 of the appendix.

4. Experiments

Here we'd like to compare $q_\phi(z|y)$ when approximating the true posterior $p(z|y)$ versus when approximating the aggregate posterior $q^y(z)$. We compare them in terms of two metrics: correctness and conditional likelihood. Following Vedantam et al. (2017), the correctness is the fraction of attributes for each generated image that match those specified in the text's description. To compute correctness, we use a classifier, which was trained independently from the model, to predict the attributes for a given image. Thus, in order to achieve high correctness, the model must generate images of sufficient quality for the classifier to distinguish as well as generate images which match the text that it's conditioned on. Furthermore, to access the coverage of $q_\phi(z|y)$, we also estimate the conditional likelihood of held-out images. We do this by computing the following lower bound:

$$\log p(x|y) \geq \mathbb{E}_{q(z|x)}\left[\log\left(\frac{p(x|z)q_\phi(z|y)}{q(z|x)}\right)\right] \tag{6}$$

Thus, $q_\phi(z|y)$ will have lower conditional likelihood if it is not modelling the full diversity of images conditioned on the text.

We perform experiments on a modified version of the CLEVR dataset (Johnson et al., 2016), which we call Two Objects, where the image contains only two objects and the text is a description of the objects as well as the relation between them. We also use the CelebA dataset (Liu et al., 2015), where we've converted the attributes into a sequence of shuffled words. See Section 5.2 for further details of the datasets and Sections 5.7 and 5.8 for example dataset samples.

4.1. Aggregate vs True Posterior

For this experiment, we first train a VLVAE by optimizing the objective of Eqn. 1, then subsequently train $q_\phi(z|y)$. We use $q_{True}(z|y)$ to refer to $q_\phi(z|y)$ trained to approximate the true posterior by optimizing Eqn 3. Similarly, we use $q_{Agg}(z|y)$ to refer to $q_\phi(z|y)$ trained to approximate the aggregate posterior by optimizing Eqn 5. Table 1 shows the comparison of $q_{True}(z|y)$ and $q_{Agg}(z|y)$ in terms of correctness and conditional likelihood on the validation set. On the Two Object dataset, we see that learning $q_{Agg}(z|y)$ significantly improves the correctness of the generated images, while also having higher conditional likelihood, indicating that it also maintains image diversity. On CelebA, there is little difference in the correctness, while $q_{Agg}(z|y)$ has higher conditional likelihood.

	Two Objects		CelebA	
	$q_{True}(z\|y)$	$q_{Agg}(z\|y)$	$q_{True}(z\|y)$	$q_{Agg}(z\|y)$
Correctness (%)	72.75	87.01	80.83	80.32
$\log p(x\|y)$	75716.48	75739.79	-6776.27	-6637.08

Table 1: Comparison of the aggregate of the image posteriors for a given text $q_{Agg}(z|y)$ and the approximate posterior for the text $q_{True}(z|y)$ on the validation set. Learning the aggregate generally helps improve the correctness of samples and conditional likelihood.

4.2. Aggregate vs Prior

The VLVAE model from the experiment above was trained with a standard normal distribution for the prior, $p(z) = N(0, 1)$. Here, we take the trained VLVAE and approximate $\mathbb{E}_{p_D(x)}[q(z|x)]$ with $q_\phi(z)$ by maximizing $\mathbb{E}_{q(z|x)}[\log q_\phi(z)]$ wrt ϕ. We then compute the likelihood of the validation set under each prior: $p(z)$ and $q_\phi(z)$. Table 2 shows that $q_\phi(z)$ improves the correctness of the samples from the model. Since the validation set likelihoods are also higher under $q_\phi(z)$, this suggests that $q_\phi(z)$ maintains the diversity of the samples.

	Two Objects		CelebA	
	$p(z)$	$q_\phi(z)$	$p(z)$	$q_\phi(z)$
Correctness (%)	65.62	83.81	80.25	81.14
$\log p(x)$	75715.36	75726.15	-6686.31	-6636.30
$\log p(y)$	-19.43	-17.40	-53.44	-11.78

Table 2: Comparison of original prior, $p(z) = N(0, 1)$, and learned aggregate prior, $q_\phi(z)$, on the validation set. Learning the aggregate helps improve the correctness of samples and likelihoods.

References

X. Chen, D. P. Kingma, T. Salimans, Y. Duan, P. Dhariwal, J. Schulman, I. Sutskever, and P. Abbeel. Variational Lossy Autoencoder. *ArXiv e-prints*, November 2016.

L. Dinh, J. Sohl-Dickstein, and S. Bengio. Density estimation using Real NVP. *ArXiv e-prints*, May 2016.

M. Hoffman and M. Johnson. Elbo surgery: yet another way to carve up the variational evidence lower bound. *In NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016.

D. Jimenez Rezende and S. Mohamed. Variational Inference with Normalizing Flows. *ArXiv e-prints*, May 2015.

J. Johnson, B. Hariharan, L. van der Maaten, L. Fei-Fei, C. L. Zitnick, and R. Girshick. CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning. *ArXiv e-prints*, December 2016.

Z. Liu, P. Luo, X. Wang, and X. Tang. Deep learning face attributes in the wild. *In Intl. Conf. on Computer Vision*, 2015.

A. Makhzani, J. Shlens, N. Jaitly, I. Goodfellow, and B. Frey. Adversarial Autoencoders. *ArXiv e-prints*, November 2015.

M. Rosca, B. Lakshminarayanan, and S. Mohamed. Distribution Matching in Variational Inference. *ArXiv e-prints*, February 2018.

M. Suzuki, K. Nakayama, and Y. Matsuo. Joint Multimodal Learning with Deep Generative Models. *ArXiv e-prints*, November 2016.

H. Takahashi, T. Iwata, Y. Yamanaka, M. Yamada, and S. Yagi. Variational Autoencoder with Implicit Optimal Priors. *ArXiv e-prints*, September 2018.

J. M. Tomczak and M. Welling. VAE with a VampPrior. *ArXiv e-prints*, May 2017.

R. van den Berg, L. Hasenclever, J. M. Tomczak, and M. Welling. Sylvester Normalizing Flows for Variational Inference. *ArXiv e-prints*, March 2018.

R. Vedantam, I. Fischer, J. Huang, and K. Murphy. Generative Models of Visually Grounded Imagination. *ArXiv e-prints*, May 2017.

J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks. *ArXiv e-prints*, March 2017.

5. Appendix

5.1. Model Architecture

Overview of model: The model will involve different components depending of which modalities it uses for inference and which modalities to generate. For instance, in Eqn. 1, the model performs inference with only images and then generates images and text. In this case, the model begins by encoding the image into a vector. If we were performing inference with both modalities, both modalities would be encoded and the vectors would be concatenated. Next, the encoding is passed to another network which outputs the parameters of a Gaussian distribution over the latent space. The distribution is then sampled and the images and text are decoded. To decode the latent vector samples, the samples begin by being decoded by a network which is shared by both modalities. Then this vector is passed to an image decoder and text decoder. In the following, we will provide more detail of each of those components.

Image encoder: We use networks similar to the ones used in CycleGAN (Zhu et al., 2017). Broadly, we use a 3-block Resnet with instance norm then flatten the representation and output a 200 dimensional vector. We build off this implementation:
https://github.com/aitorzip/PyTorch-CycleGAN/blob/master/models.py.

Text encoder: The text is encoded with a GRU with a hidden state size of 200. The final hidden state is used as the text encoding

Shared encoder network: The encodings of each modality are concatenated then passed through three fully-connected layers with batchnorm and a residual connection. It outputs the means and variances of the Gaussian for the latent variable z.

Shared decoder network: The latent variable z is passed through five fully-connected layers with two residual connections and batchnorm. The resulting vector has dimensionality of 200.

Image decoder: The vector from the shared decoder network is linearly transformed into a 1000 dimensional vector then reshaped into 2D representation with 10 channels. Similar to the image encoder, we use a 3-block Resnet with instance norm to output the parameters of the image likelihood distribution. See Section 5.1.2 for details of the image likelihood.

Text decoder: The text decoder is a GRU with a 200 dimensional hidden state. Each hidden state is dependent on the previous word, previous hidden state, and the shared decoder network output. The hidden state for each word is put through three fully-connected layers with a residual connection, batchnorm, and dropout, and outputs the distribution over the vocabulary.

5.1.1. Flow Distributions

In this paper, we use flows in two cases: 1) to model a distribution given text $q_\phi(z|y)$ and 2) to model the aggregate approximate posterior of the dataset $q(z)$, which can be used to replace the prior $p(z)$. In both cases, we transform a Gaussian distribution into a more complex distribution. We will first describe the flow used for 2) then we will describe how its modified for 1).

The flow transformation that we employ is similar to the transformations of Real NVP (Dinh et al., 2016). We partition the latent variable z into two, z_1 and z_2, then perform the following transformations:

$$z'_1 = z_1 \circ \sigma_1(z_2) + \mu_1(z_2) \tag{7}$$

$$z'_2 = z_2 \circ \sigma_2(z'_1) + \mu_2(z'_1) \tag{8}$$

where $\sigma_1, \sigma_2, \mu_1, \mu_2 : \mathbb{R}^n \to \mathbb{R}^n$ are differentiable mappings parameterized by neural nets and $\circ$ takes the Hadamard or element-wise product. We partition the latent variable by simply indexing the elements of the first half and the second half. The determinant of the combined transformation's Jacobian is $\left|\det\left(\frac{\partial z'}{\partial z}\right)\right| = \left(\prod_{i=1}^{n} \sigma_1(z)_i\right)\left(\prod_{j=1}^{n} \sigma_2(v')_j\right)$. We employ six of these flows for each sample.

When using a flow to model a distribution which is conditioned on text $q_\phi(z|y)$, we pass the text to each flow transformation. Thus, $\sigma_1(z_2)$, $\mu_1(z_2)$, $\sigma_2(z_2)$, $\mu_2(z_2)$ become $\sigma_1(z_2, y)$, $\mu_1(z_2, y)$, $\sigma_2(z_2, y)$, $\mu_2(z_2, y)$. Here we also chain six of these flows for each sample.

5.1.2. Image Likelihood Distribution

The images are preprocessed to be continuous values between 0 and 1. For our image likelihood $p(x|z)$, we use a constrained Beta distribution. Specifically, the image decoder outputs the α parameter of the Beta distribution and we set the β parameter to $1 - \alpha$, thus the distribution is constrained to α and β values which sum to 1. We also scale both α and β by a factor of 100 in order to adjust the variance of the distribution. We chose a Beta distribution because it is a distribution over continuous values between 0 and 1, unlike other commonly used distributions such as Gaussian or Laplace.

5.2. Dataset Details

Two Object CLEVR

We modified the CLEVR dataset (Johnson et al., 2016) so that the image contains only two objects and the text is a description of the objects as well as the relation between them. We call this dataset the Two Object dataset. See Fig. 3 for example images and text. The images have dimensions: [112,112,3]. The text consists of nine words and the format of the text is as following: $O_1^{size}, O_1^{colour}, O_1^{material}, O_1^{shape}, R, O_2^{size}, O_2^{colour}, O_2^{material}, O_2^{shape}$, where O_1 and O_2 are used to refer to each object in the image and R refers to the relational word. We will now list the vocabulary for each attribute. O^{size}: small, large. O^{colour}: cyan, red, gray, blue, yellow, purple, brown, green. $O^{material}$: rubber, metal. O^{shape}: sphere, cylinder, cube. For the realtional word, the possible words are: right, left, front, behind. Thus, there is a total of 2*8*2*3*4=384 possible different texts. The dataset consists of 100k image-text pairs, where 90k are used for training and 10k for validation.

CelebA

We use the CelebA dataset (Liu et al., 2015), where we've converted the attributes into a sequence of shuffled words. See Fig. 5 for examples of samples from the dataset. The images have dimensions: [64,64,3]. The max length of the text is 9 words, but the median length is 4 words. Following Vedantam et al. (2017), we use a subset of the attributes. The vocabulary encompasses 20 different tokens: Bushy Eyebrows, Male, Female, Mouth Slightly Open, Smiling, Bald, Bangs, Black Hair, Blond Hair, Brown Hair, Eyeglasses, Gray

Hair, Heavy Makeup, Mustache, Pale Skin, Receding Hairline, Straight Hair, Wavy Hair, Wearing Hat, and '-' to indicate blank word. There are a total of 202599 image-text pairs, and 180k are used for training and 22598 are used for validation.

5.3. Training Details

We trained the model for 400k steps with the ADAM optimizer with learning rate $4 * 10^{-4}$. We used a batch size of 20. The size of the latent variable was 50 dimensions. We employed warmup to the objective for the first 20k steps. More specifically, we annealed the weight of the KL term of the objective from 0 to 1 over the first 20k steps.

5.3.1. Likelihood Term Weights

Given that the dimensionality of the modalities differ widely, the evidence lower bound objective does not align with the real importance of each modality. For instance, we have images of 112x112x3=37632 dimensions, compared to text with around 9 dimensions. Thus there is much greater weight put on the image compared to the text. To compensate, we down-weight the likelihood term of the images $p(x|z)$ and we up-weight the text likelihood $p(y|z)$ in the objective during training.

5.4. Classifier Details

To measure the correctness of the samples from the model, we train classifiers to predict the text corresponding to a given image. The architecture of the classifiers is nearly the same as the image encoder. For the correctness of the prior samples of Table 2, we compare generated images to generated text. For the correctness of the conditional samples of Table 1, we compare generated images to the given real text.

For the two object CLEVR dataset, there is ambiguity regarding the object ordering in the text as well as the chosen relational word (right, left, front, back). Due to this, we pass the relational word to the classifier so that the output text is deterministic. The correctness is simply the fraction of matching words between the sample from the model and the classifier prediction. For the CelebA dataset, since the word ordering is random, the correctness measure ignores ordering. In this case, the classifier takes as input an image and outputs the probability of each word being associated with that image. Given some text, the correctness of the text is computed based on the fraction of the words that have a higher than 50% probability of being from that image (based on the classifier).

5.5. 2D Visualization Details

For Fig. 1, we trained a VLVAE model with a latent size of 2 dimensions. It was trained on a simplified version of the CLEVR dataset, where each image contains only a single object and where the object only varies in its colour and size. More specifically, the object in the image can only be blue or red and large or small. Once trained, to plot the true posterior for the text $p(z|y)$, we computed $p(y|x)p(z)$ for each z in the grid then normalized. To plot the aggregate distributions, we encoded the images that correspond to the text y then normalized the mixture of the image approximate posteriors.

5.6. Analysis of Aggregate Posterior Objective

$$\mathcal{L} = \mathbb{E}_{p(x)} \left[\underbrace{\mathbb{E}_{q(z|x)} \left[\log p(x|z) \right]}_{\text{Reconstruction}} - \underbrace{KL(q(z|x)||p(z))}_{\text{Posterior to Prior}} \right] \tag{9}$$

We can rewrite the KL term above in terms of the aggregate posterior $q(z)$:

$$\begin{aligned}
\mathbb{E}_{p(x)} \left[KL(q(z|x)||p(z)) \right] =& \mathbb{E}_{p(x)q(z|x)} \left[\log \left(\frac{q(z|x)}{p(z)} \right) \right] && (10)\\
=& \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{q(z|x)}{p(z)} \right) \right] && (11)\\
=& \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{q(z|x)}{p(z)} \right) * \frac{p(x)}{p(x)} \right] && (12)\\
=& \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{q(z,x)}{p(z)p(x)} \right) \right] && (13)\\
=& \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{q(z)}{p(z)} \right) \right] + \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{p(x|z)}{p(x)} \right) \right] && (14)\\
=& KL\left(q(z)||p(z)\right) + \mathbb{E}_{q(z)q(x|z)} \left[\log \left(\frac{q(z|x)p(x)}{q(z)p(x)} \right) \right] && (15)\\
=& KL\left(q(z)||p(z)\right) + \mathbb{E}_{p(x)q(z|x)} \left[\log \left(\frac{q(z|x)}{q(z)} \right) \right] && (16)\\
=& KL\left(q(z)||p(z)\right) + \mathbb{E}_{p(x)} \left[KL(q(z|x)||q(z)) \right] && (17)
\end{aligned}$$

So the lower bound $\mathcal{L}$ can be written as:

$$\mathcal{L} = \mathbb{E}_{p(x)} \left[\underbrace{\mathbb{E}_{q(z|x)} \left[\log p(x|z) \right]}_{\text{Reconstruction}} - \underbrace{KL(q(z|x)||q(z))}_{\text{Posterior to Aggregate}} - \underbrace{KL\left(q(z)||p(z)\right)}_{\text{Aggregate to Prior}} \right] \tag{18}$$

In Vedantam et al. (2017), they showed that a similar decomposition is applicable to the KL term of the JMVAE (Suzuki et al., 2016) objective. Following the same derivation as above, the KL term of JMVAE can be expressed as:

$$\mathbb{E}_{p(x_y)} \left[KL(q(z|x)||q_\phi(z|y)) \right] = \mathbb{E}_{p(x_y)} \left[\underbrace{KL(q(z|x)||q^y(z))}_{\text{Posterior x to Aggregate x}} + \underbrace{KL\left(q^y(z)||q_\phi(z|y)\right)}_{\text{Aggregate x to Posterior y}} \right] \tag{19}$$

5.7. Two Objects Samples

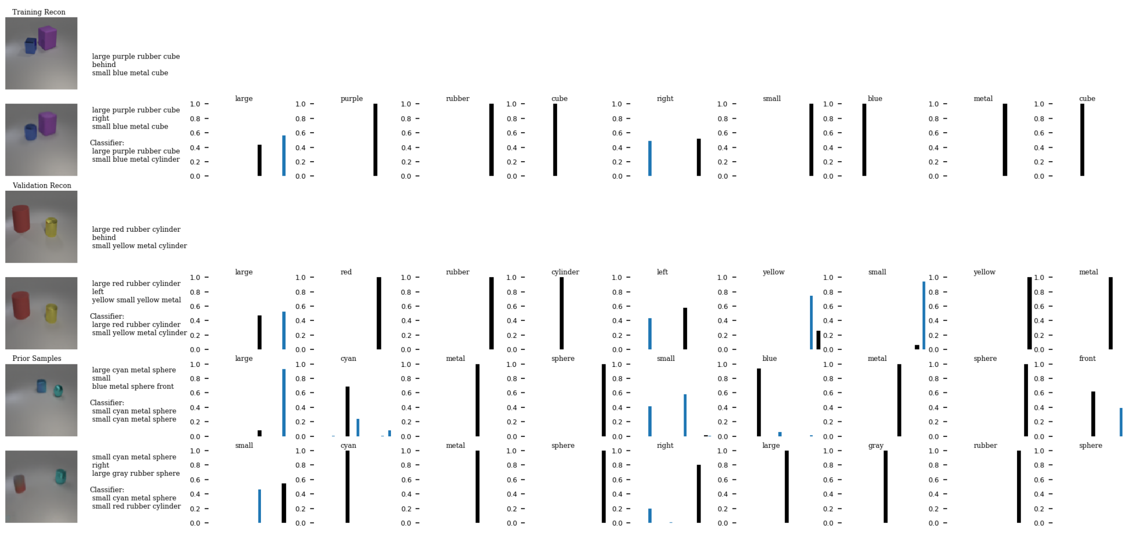

Figure 3: Row 1: Image from training set along with corresponding text. Row 2: Reconstruction of training image and text. The classier text is the text the classifier predicted given the generated relation. The blue histograms are the text decoding distributions. The black bar in the histogram indicates the sampled word which is also printed above the histogram. We can see that since the latent space does not contain information specific to the text, the first one is uncertain as well as the relation. Row 4 and 5: Same as above but on the validation set. Row 6 and 7: Samples from the prior.

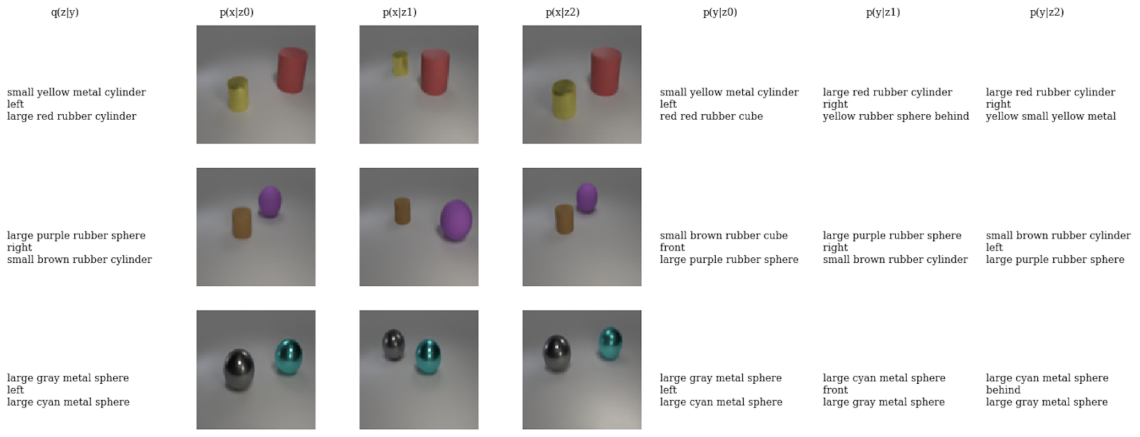
q(z|y)
p(x|z0)
p(x|z1)
p(x|z2)
p(y|z0)
p(y|z1)
p(y|z2)
small yellow metal cylinder
left
large red rubber cylinder
small yellow metal cylinder
left
red red rubber cube
large red rubber cylinder
right
yellow rubber sphere behind
large red rubber cylinder
right
yellow small yellow metal
large purple rubber sphere
right
small brown rubber cylinder
small brown rubber cube
front
large purple rubber sphere
large purple rubber sphere
right
small brown rubber cylinder
small brown rubber cylinder
left
large purple rubber sphere
large gray metal sphere
left
large cyan metal sphere
large gray metal sphere
left
large cyan metal sphere
large cyan metal sphere
front
large gray metal sphere
large cyan metal sphere
behind
large gray metal sphere

Figure 4: Column 1: Text provided to the text inference network $q_\phi(z|y)$. Columns 2,3,4: Image decoding of three samples $z0, z1, z2$ from $q_\phi(z|y)$. Columns 5,6,7: Text decoding of three samples $z0, z1, z2$ from $q_\phi(z|y)$.

5.8. CelebA Samples

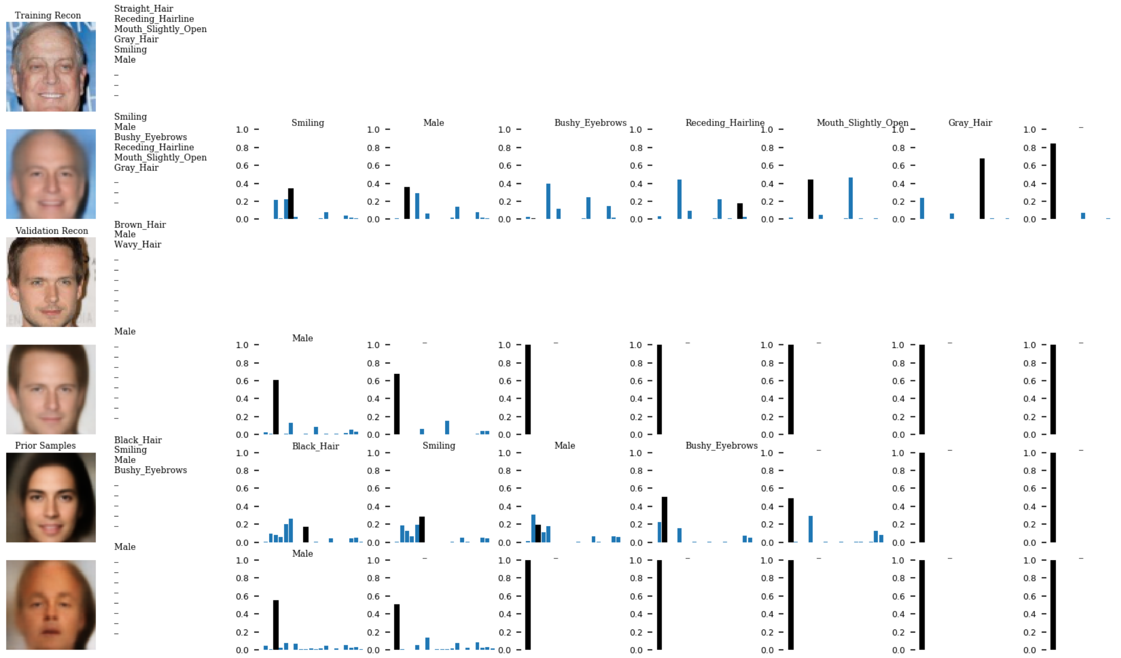

Figure 5: Row 1: Image from training set along with corresponding text. Row 2: Reconstruction of training image and text. The blue histograms are the text decoding distributions. The black bar indicates the sampled word which is also printed above the histogram. We can see that since the latent space does not contain information specific to the text, there is uncertainty in the ordering of the attributes. Row 4 and 5: Same as above but on the validation set. Row 6 and 7: Samples from the prior.

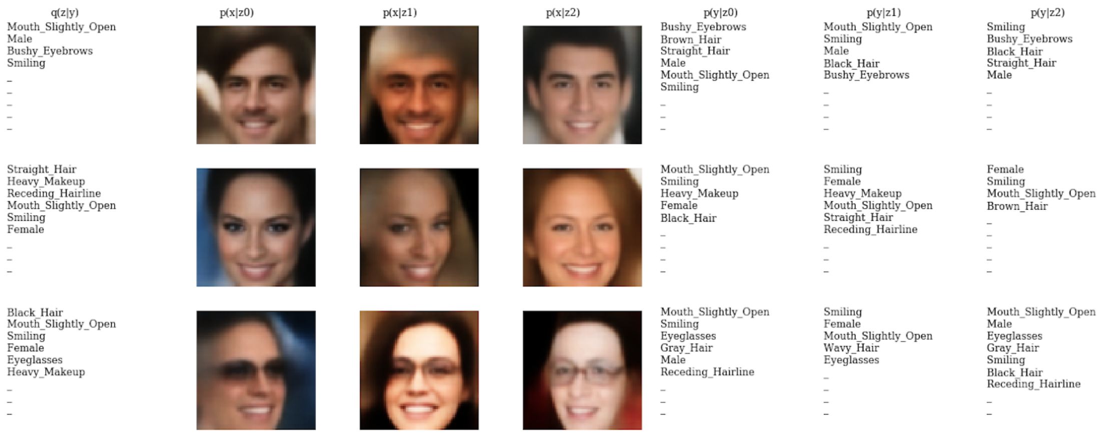

Figure 6: Column 1: Text provided to the text inference network $q_\phi(z|y)$. Columns 2,3,4: Image decoding of three samples $z0, z1, z2$ from $q_\phi(z|y)$. Columns 5,6,7: Text decoding of three samples $z0, z1, z2$ from $q_\phi(z|y)$.

6. Joint vs Marginal Inference

Comparing marginal $q(z|x)$ and joint inference $q(z|x, y)$, we can see from Fig. 7 that joint inference stores information specific to text in the latent space since it has no uncertainty in its word distributions. In contrast, the marginal inference model uses the auto-regressive text decoder to model the uncertainty in the words. Specifically, there's uncertainty in the order of objects and which relation is used.

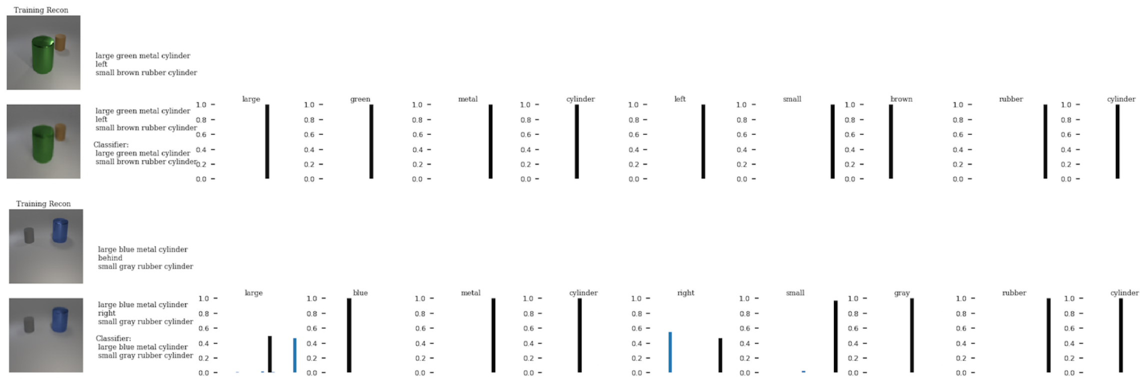

Figure 7: Reconstructions from joint inference (top) and marginal inference (bottom). Joint inference stores info specific to text in the latent space.