Better Caption Embeddings Projections in PixelCNN++

Michael Sun[*]
Stanford University
Stanford, CA 94305
`msun415@stanford.edu`

Abstract

This project investigates different methods of projecting caption embeddings into the PixelCNN++ model, a variant of the conditional generative model PixelCNN, for the task of caption-to-image generation. All experiment models were trained with BERT word vectors and come in two tiers of granularity, the first being the category name for ImageNet ("dalmatian"), the second being a coarse WordNet label ("dog"). We first evaluate the bpd (bits per dimension) on baseline models with the default linear projection, trained on ImageNet32. Then, we investigate how model performance improves and generalizes, atop three other projection methods using common generative model metrics. The code to this project can be found on my GitHub (*github.com/shiningsunnyday*).

1 Introduction

In recent years, generative models have been a major source of innovation in the AI field. Although discriminative models have found the most direct applications in industry, generative models solve the more inspiring problem of modeling the data distribution itself. In particular, conditional generative models were introduced to model the joint distribution of the data conditioned on some latent representation of it, and this has led to exciting applications like caption-to-image generation.

Generative models model the data distribution in a variety of ways, but the most straightforward is via an autoregressive distribution

$$p(x_1, x_2, \ldots, x_n) = \prod_{i=1}^{n} p(x_i | x_{<i})$$

which is easily interpretable when given an ordering of the data dimensions, like words in a body of text.

The conditional PixelCNN, introduced by Van Der Oord et al. (2016b) [1] is an autoregressive model that models each conditional distribution with masked convolution layers. Inspired by the success of an originally proposed variant, PixelRNNs [2], PixelCNN++ [3] leverages gated activation unit to model more complex conditional dependencies.

2 Problem Statement

Given a dataset $\mathbb{D} = \{x^{(i)} : 1 \leq i \leq m, x^{(i)} \in \mathbb{R}^{1024}\}$ images, $x^{(i)}$ the $3 \times 32 \times 32$ pixels of the image, we observe $\mathbb{D}_Z = \{z^{(i)} : 1 \leq i \leq m\}$, the latent representations of each image. The task

[*]Michael is a sophomore and aspiring AI researcher studying Mathematical and Computational Science.

is to model the conditional distributions $p(x_1, x_2, \ldots, x_{1024}, z) = \prod_{i=1}^{1024} p(x_i|x_{<i}, h)$ and perform maximum likelihood estimation.

Baseline and preliminary experiments will be done on ImageNet32, a downsampled version of ImageNet.

3 Technical Approach

3.1 Background and Model Architecture

The architecture of the PixelCNN consists in modeling every conditional distribution $p(x_i|x_{<i}, h)$ with a convolutional neural network. Let us first recap how the PixelCNN works, then I will formalize the hyperparameters I varied for this paper.

3.1.1 Model Layers

The autoregressive property for the PixelCNN is satisfied by using masked convolutional filters, which produces feature maps in which the corresponding pixels depend only on pixels preceding it. For each layer, two separate types of filters are used - a vertical $n \times n$ stack with the bottom half masked, and a horizontal $1 \times n$ stack with pixels right to x_i one masked. This eliminates the previous problem of *blind spots*, pixels left out of the conditioning of x_i. Furthermore, each convolutional layer models the color channels in order, $p(x_{i,r}|x_{<i}, z)$, $p(x_{i,g}|x_{<i}, z, x_{i,r})$, and finally $p(x_{i,b}|x_{<i}, z, x_{i,r}, x_{i,g})$.

The convolutional layer output is combined with a non-linearity layer which, instead of a simple convolution followed by ReLU, consists of a gated activation unit which adds in the caption representation,

$$y = \tanh(W_{k,f} * x + V_{k,f}^T h) \odot \sigma(W_{k,g} * x + V_{k,g}^T h).$$

The gated activation is inspired from PixelRNN's superior results to PixelCNN (here $*$ represents the convolution operation described above). As illustrated in van der Oord et al. note that $W_{k,f}$ is for layer k and only involves the $n \times n$ vertical stack, whereas $W_{k,g}$ combines the output of the convolutional stacks.

3.1.2 Loss Function

In the code behind this project's implementation, the loss function used is the negative of the log-likelihood, which is a discretized mixture of logistics - which is specified as

$$L(x, h) = -\log P(x, h)$$

$$P(x, h) = \sum_{i=1}^{k} a_i P_i(x, h)$$

where P_i is the i logistic distribution and a_i the (learned) linear combination coefficients.

For given x, model outputs include the k coefficients, and for each color channel, the predicted mean, variance and scale for each of the k mixtures. Given an input of $(N, N, 3)$, the output would be of size $(N, N, k + 3 * 3 * k)$.

3.2 Hyperparameter Changes

This paper seeks to explore more intelligent ways of incorporating in caption representations by investigating the set of hyperparameters associated with:

$$\{f(h)|y_k = \tanh(W_{k,f} * x + f(h)) \odot \sigma(W_{k,g} * x + f(h))$$

The motivation for this investigation is that one or both of two things may be true.

1) As captions become less diverse in semantic meaning (i.e. "dog" instead of "Dalmatian"), it may lead to mode collapse. It is then necessary to constrain the model's usage of captions, by decreasing its expressivity in the network by adding an activation.

2) As captions become coarser, there may be an increase in the model's bias, and it may prove effective to compensate by adding smoother, non-linear boundaries than that of the linear projection to better capture the nuances of caption-pixel interaction. Specifically, we explore non-linear alternatives to $f(h)$ described further below with motivation.

3.3 Non-linear Alternative 1: Gated Linear

$$f(h) = (W^T h)\sigma(V^T h + b)$$

The motivation is the same as that for the gated activation unit, which has proven successful [2] in modeling deep spatial dependencies.

3.4 Non-linear Alternative 2: Linear ReLU

$$f(h) = ReLU(V^T h + b)$$

The motivation is that it may prove an improved version of the linear projection. The ReLU activation may also prove to be more compatible with traditional CNN architecture.

3.5 Non-linear Alternative 3: Shallow Networks

$$f(h) = ReLU(W_2^T \sigma(W_1^T h + b_1) + b_2)$$

(In the experiments, $W_1 \in \mathbb{R}^{2\dim h, \dim h}$).

The motivation is similar to that of Linear ReLU, except this is more expressive, and may reduce model bias further. If it indeed does, the number of layers and layer sizes can become additional hyperparameters to tune.

4 Experiment Methodology and Results

4.1 Benchmark Models

Table 1: Benchmark Results

Dataset	Epochs	Conditioning	Projection Method	Validation bpd (bits-per-dimension)
ImageNet32	2	BERT	Linear	3.84
ImageNet32	3	one-hot	Linear	3.73
ImageNet32	4	zeros	Linear	3.75
CIFAR10	90	one-hot	Linear	3.04
CIFAR10	89	zeros	Linear	3.73

As expected, the conditioning alone can play a significant role in the bpd metric. On CIFAR10, one-hot conditioning reduced bpd by ≈ 0.69 bits. It can be interpreted that conditioning on a one-hot encoding rather than a zero encoding reduced training epochs by 1 (from 4) without sacrificing bpd. Similarly, conditioning on BERT further reduced training epochs by 1 with only a 0.11 increase in bpd.

However, the focus of this paper won't be on conditioning, so the above will only be reference. Nonetheless, we will return to discussing the outcomes of our experiments in lieu of these benchmark models.

4.2 Metrics Used

There're many helpful metrics for evaluating generative models, but this paper will use bits-per-dimension (bpd), Inception Score (IS) and the Frechet Inception Distance (FID).

Table 2: Baseline Results

Metric	Formula	Meaning
BPD	$L(x,h)/(\log 2 \cdot N^2)$	bits to capture loss per pixel
IS	$\exp\left(\mathbb{E}_{x\sim p}\left[\int c(y\|x)\left(\log \frac{c(y\|x)}{c(y)}\right) dy\right]\right)$	sharpness * diversity
FID	$\|\|\mu_T - \mu_G\|\|^2 + \mathrm{Tr}(\Sigma_T + \Sigma_G - 2(\Sigma_T\Sigma_G)^{1/2}$	similarity of feature representation

4.3 Training Logistics and Disclaimers

All following experiments were done on ImageNet32 with BERT conditioning. A maximum-GPU-compatible batch size of 32 was used with PyTorch, CUDA and numpy random seeds set to 12 in training with maximum CPU memory/workers. Metric evaluation was done over a random batch of samples (no seed). Due to memory requirement, shallow NN projection models require two GPUs to train (by the time I ran out of budget, I only managed to train one model).

4.4 Tier 1 Baseline

The first step is to establish baselines with Linear projection on various epoch sizes.

Table 3: Baseline Results

Epochs	Projection	Val bpd (bits-per-dimension).	IS.	FID
2	Linear	3.840	1.484	423.58
1	Linear	3.819	1.461	423.66
1/2	Linear	3.948	1.467	428.66

4.5 Tier 2 Baseline

Due to severe Google Cloud budget constraints, Tier 2 models were trained for only a third of an epoch.

Table 4: Baseline Results

Epochs	Projection	Val bpd (bits-per-dimension).	IS.	FID
1/3	Linear	4.033	1.492	424.19

4.6 Tier 1 Results

Table 5: Tier 1 Projection Method Comparison

Tier	Epochs	Projection	Val bpd (bits-per-dimension).	IS.	FID
1	1	Linear	3.819	1.451	445.18
1	1	Gated Linear	3.800	1.493	439.58
1	1	Linear ReLU	3.782	1.481	454.83
1	1	Shallow NN	N/A	N/A	N/A
1	1/2	Linear	3.948	1.467	395.90
1	1/2	Gated Linear	3.924	1.507	421.90
1	1/2	Linear ReLU	3.918	1.486	416.75
1	1/2	Shallow NN	N/A	N/A	N/A

4.7 Tier 2 Results

Table 6: Tier 2 Projection Method Comparison

Tier	Epochs	Projection	Val bpd (bits-per-dimension).	IS.	FID
2	1/3	Linear	4.033	1.492	424.19
2	1/3	Gated Linear	4.028	1.486	429.79
2	1/3	Linear ReLU	4.000	1.447	433.89
2	1/3	Shallow NN	4.013	1.470	407.25

5 Observations

5.1 Bits-per-dimension (bpd)

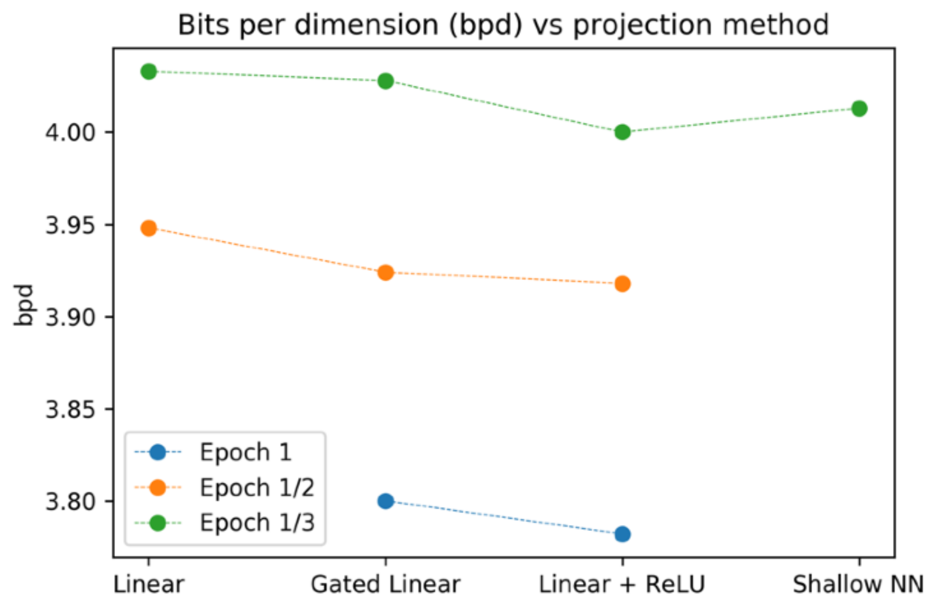
The linear + ReLU projection method produced the lowest validation bpd amongst both caption tiers and surpassed both the baseline linear-projection model trained for this project and the benchmark model. This may be surprising because the linear + ReLU projection isn't as expressive as the shallow NN. This suggests linear + ReLU may not only be the optimal projection method amongst the ones tried for this experiment, but also has a level of expressivity close to optimal (from a bias-variance tradeoff perspective) amongst all possible projection methods.

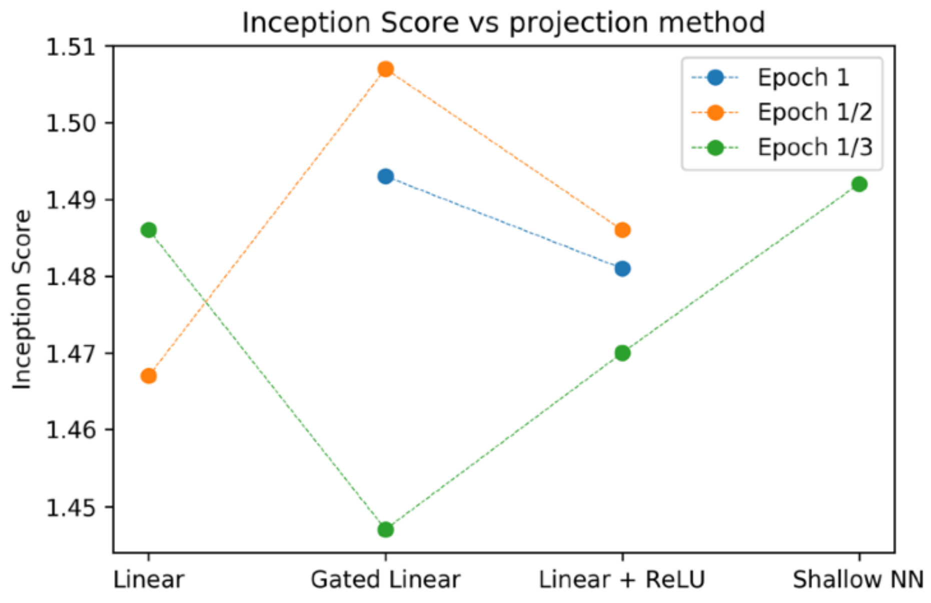
5.2 Inception Score (IS)

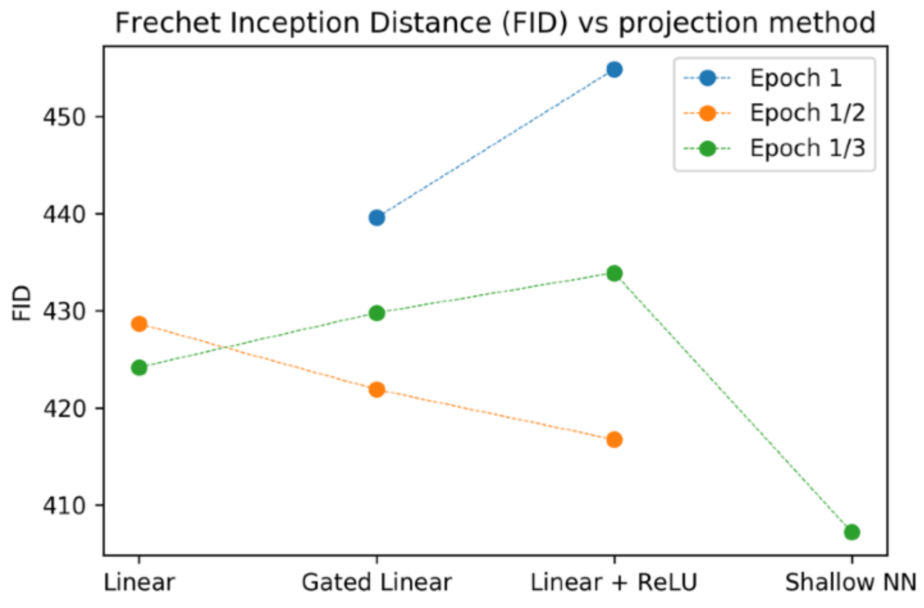
The gated linear projection method had the highest IS score, suggesting its "gate" functionality achieves the best of both worlds - sharpness and expressivity. This shares the same sentiment as the choice to use the gated activation unit in PixelCNN++ due to its superior ability to model spatial dependencies through many layers.

5.3 Frechet Inception Distance (FID)

The baseline method (linear) retains the lowest FID scores, although the only model trained with shallow NN exceptionally outperforms its linear counterpart. It seems linear and shallow NN projections are most compatible as deep NN features used to evaluate FID. Gated Linear and Linear + ReLU, by contrast, may produce features better suited for other model families like CNNs/RNNs, rather than that found in the pre-final layer of InceptionV2.


Bits per dimension (bpd) vs projection method
bpd
4.00
3.95
3.90
3.85
3.80
Epoch 1
Epoch 1/2
Epoch 1/3
Linear
Gated Linear
Linear + ReLU
Shallow NN


Inception Score vs projection method
Inception Score
1.51
1.50
1.49
1.48
1.47
1.46
1.45
Epoch 1
Epoch 1/2
Epoch 1/3
Linear
Gated Linear
Linear + ReLU
Shallow NN


Frechet Inception Distance (FID) vs projection method
FID
450
440
430
420
410
Epoch 1
Epoch 1/2
Epoch 1/3
Linear
Gated Linear
Linear + ReLU
Shallow NN

6 Conclusion

These observations reveal clear insight into how designing the method of projection contribute to the robustness of the model. In each epoch group, every projection method surpassed the val bpd of the baseline, which corroborates my hypothesis that increasing the expressivity of the projection reduces model bias. However, an increase in expressivity only helps to the point it causes the model to overfit, which was the case with shallow NN. I believe this is because label-based word2vec carries only so much semantic meaning which is suitable for projection methods up to a certain expressivity.

Although budget constraints prevented me from experimenting with Tier 3 captions, it's reasonable to presume there would be no performance gains on the val bpd training atop shallow NN projections on Tier 3 captions.

I also gained insights into how we can use the projection method to optimize for different, common and interpretable evaluative metrics. The FID observations also remind us it's not enough to consider only the expressivity, but also how the projection fits in as a feature to the model architecture. It's reasonable to presume the Linear + ReLU projection method may have worked better on the original PixelCNN model that didn't use gated activation units.

7 Future Directions

7.1 Pretrained Embedding

As mentioned in the previous section, an increase in the projection method's expressivity is only useful up to fully extracting the semantic information available in a caption embedding. Throughout this project, only BERT embeddings were used. Although BERT has led to breakthroughs in many language modelling tasks, it may be suboptimal as a way of conditioning on image pixels. In fact, it's questionable how the benchmark models trained with only one-hot and zero conditioning outperformed the one trained with BERT. Though one may argue the BERT model only trained for 2 epochs, our experiments suggest, if anything, 1 epoch is already sufficient in achieving comparable val bpd's.

It may be of interest to investigate other available word vector embeddings, or for the more ambitious, training embeddings directly on image caption collections rather than text documents.

7.2 Alternative Conditioning

Throughout this project, only conditioning on class labels were used. It would be interesting to obtain more semantically meaningful caption phrases like "dog sitting upright on grass" then explore more "invasive" conditioning methods like transformers (with attention used to both encode extended caption phrases and to decode image pixels).

The results of this project make me optimistic that the method of projection can become a design choice in determining model robustness for future models of the PixelCNN family.

References

[1] Aaron van den Oord, Nal Kalchbrenner, Oriol Vinyals, Lasse Espeholt, Alex Graves, and Koray Kavukcuoglu. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1606.05328*, 2016c.

[2] Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. Pixel Recurrent Neural Networks. *arXiv preprint arXiv:1601.06759*, 2016c.

[3] Tim Salimans, Andrej Karpathy, Xi Chen, Diederik P. Kingma. Conditional image generation with pixelcnn decoders. *arXiv preprint arXiv:1701.05517*, 2017.