Origins of Creativity in Attention Based Diffusion Models

Emma Finn EFINN@COLLEGE.HARVARD.EDU
T. Anderson Keller T.ANDERSON.KELLER@GMAIL.COM
Manos Theodosis ETHEODOSIS@G.HARVARD.EDU
Demba E. Ba DEMBA@SEAS.HARVARD.EDU
CRISP Lab, Harvard John A. Paulson School of Engineering and Applied Sciences,
and Kempner Institute for the Study of Natural and Artificial Intelligence,
Harvard University, Cambridge, MA

Abstract

As diffusion models have become the tool of choice for image generation and as the quality of the images continues to improve, the question of how 'creativity' originates in diffusion has become increasingly important. The score matching perspective on diffusion has proven particularly fruitful for understanding how and why diffusion models generate images that remain visually plausible while differing significantly from their training images. In particular, as explained in (Kamb & Ganguli, 2024) and others, e.g., (Ambrogioni, 2023), theory suggests that if our score matching were optimal, we would only be able to recover training samples through our diffusion process. However, as shown by Kamb & Ganguli, (2024), in diffusion models where the score is parametrized by a simple CNN, the inductive biases of the CNN itself (translation equivariance and locality) allow the model to generate samples that globally do not match any training samples, but are rather patch-wise 'mosaics'. Despite the widespread use of UNet architectures with self-attention as the score backbone in diffusion models, the theoretical role of attention in score networks remains largely unexplored. In this work, we take a preliminary step in this direction to extend this theory to the case of diffusion models whose score is parametrized by a CNN with a final self-attention layer. We show that our theory suggests that self-attention will induce a globally image-consistent arrangement of local features beyond the patch-level in generated samples, and we verify this behavior empirically on a carefully crafted dataset.

1. Introduction

Diffusion models have become the premier tool for image generation in the last decade [9]. Their capacity to generate visually plausible images that generalize beyond the training dataset makes them extremely useful, but this capacity is not well understood [2]. Diffusion models operate by performing a series of transformations that map the underlying distribution of images to a centered, multivariate Gaussian and then learning the reverse process. One of the most fruitful approaches for understanding creativity in diffusion models has been the score matching perspective, where a relatively small neural network is trained to approximate the derivative of the log-likelihood of the underlying image distribution [7]. However, a large body of work has demonstrated that if this score-approximation is exact, a diffusion model can only return training samples: it is not creative at all [4, 6]. Kamb and Ganguli in [4] offered an important first step in understanding why diffusion models are able to generalize extremely well despite theory suggesting that well-trained diffusion models should memorize [1]. In particular, they provide a complete theory for diffusion models whose score approximator is a convolutional neural network (CNN). Because CNNs have two in-

ductive biases, translational equivariance and locality, they solve analytically for a 'score machine' with those two inductive biases. They demonstrated that for CNN-backed diffusion models, their theory "partially predicts the results" of pre-trained diffusion models [4]. In practice, however, state-of-the-art diffusion models use a much more complex score estimator network. In particular, most these models use a U-Net structure with self-attention blocks throughout [2, 10], though recent papers have explored a fully transformer-based score network [8]. Both architectures violate the local assumption strongly and the translational equivariance assumption weakly.

In this work, we propose a theory for CNNs with a single self-attention layer at the very end, which provides a first step towards bridging the gap between the existing theory and state-of-the-art models. In particular, we derive a simple theoretical example that suggests that self-attention may play the role of enforcing global self-consistency in the other-wise local patch-mosaic construction of Kamb and Ganguli [4]. Empirically, we then validate this intuition on a simple toy dataset, showing that samples are far more globally self-consistent with attention than without. We propose this as a first step towards understanding the role of attention in the creativity of diffusion models.

2. Equivariant Score Machine with Attention

To gain intuition for the form of the optimal score function with attention, we first will analyze a model with full attention over all patches. Then, to provide a tractable closed form solution, we will make the additional assumption that attention is 'top-1', and show that this intuition holds.

2.1. Simple CNN with Full Attention

We begin with the following notation, following [4]: let ϕ be an arbitrary image in the diffusion process, and for each pixel location $x \in \Lambda$, we write ϕ_x for the pixel value of ϕ at location x. Let $\Omega_x \subset \Lambda$ be the set of pixels in the patch centered at x. Let Φ be an arbitrary patch. Let π_t be the distribution over (noisy) images at time t. The true score at x is $s_t[\phi](x) = \nabla_{\phi_x} \log \pi_t(\phi)$.

We embed each patch via $g(\phi_{\Omega_x}) \in \mathbb{R}^d$ where g is a convolutional embedding network, which can be thought of as the first portion of our score-approximation network. Our learnable parameters are in g. We then define the full score function estimator $\tilde{g}[\phi](x)$, which corresponds to a single layer of attention on top of a CNN embedding.

$$\tilde{g}[\phi](x) = g(\phi_{\Omega_x}) + \sum_y \alpha_{xy}\, g(\phi_{\Omega_y}), \quad \text{where} \quad \alpha_{xy} = \frac{\exp\big(\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle\big)}{\sum_{y'} \exp\big(\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle\big)}, \quad (1)$$

so that each location x 'attends' to all patches in the image, where the sum over y runs over all other patch centers $y \in \Lambda$ in the image ϕ. In particular, we apply the simplest kind of attention over top of a convolutional neural network. The standard attention framework involves three learnable projection matrices W_Q, W_K and W_V that embeds our input set of tokens, X into queries Q, keys, K, and values, V. Attention weights are then given by Attention(Q,K,V) $=$ softmax $\left(\frac{QK^T}{\sqrt{d}}\right) V$. In our case, we downgrade W_Q, W_K, and W_V from learnable parameters into identity matrices, and recover the form

$$\text{Attention(Q,K,V)} = \text{softmax}\left(\frac{XX^T}{\sqrt{d}}\right) V \quad (2)$$

which gives us exact the entry-wise formulation above, which says that $\alpha_{xy} = \frac{\exp(x_i x_j)}{\sum_{k=1}^{n} \exp(x_i \cdot x_k)}$. We drop the scalar $\frac{1}{\sqrt{d}}$ for simplicity here.
Adopting this functional form, the corresponding score matching loss becomes

$$\mathcal{L} \;=\; \sum_{x} \mathbb{E}_{\phi\sim\pi_t} \big\| \tilde{g}[\phi](x) \;-\; s_t[\phi](x) \big\|^2.$$

We mirror the derivation in [4] and write this expectation as an integral over π_t and impose the stationary condition w.r.t. g (since the attention layer has no parameters), yielding:

$$0 = \frac{\delta \mathcal{L}}{\delta g(\Phi)} \;= \sum_{x} \frac{\delta}{\delta g(\Phi)} \int \pi_t(\phi)[||\tilde{g}[\phi](x) - s[\phi](x)||^2] d\phi \quad \forall\, \Phi. \tag{3}$$

See Appendix A for details on the derivation. Eventually we find that, setting $\mu_x := \sum_z \alpha_{xz}(\phi)\, g(\phi_{\Omega_z})$:

$$\begin{aligned} 0 = 2 \sum_{x} \int \pi_t [g(\phi_x) + \sum_{y} \alpha_{xy} g(\phi_{\Omega_y}) - s[\phi](x)]^T \delta(\phi_{\Omega_x} - \Phi)\; d\phi \\ + \int \pi_t [g(\phi_x) + \sum_{y} \alpha_{xy} g(\phi_{\Omega_y}) - s[\phi](x)]^T \sum_{y} \alpha_{xy} \left(I + (g(\phi_{\Omega_x}) - \mu_x) \delta(\phi_{\Omega_y} - \Phi) \right) d\phi \end{aligned} \tag{4}$$

Intuitively, the first term in our sum corresponds to a when the patch of interest Φ is the query, and the second term corresponds to when the patch is the key or the value. In particular, the $\delta(\phi_{\Omega_x} - \Phi)$ term matches exactly the CNN-case, and so encourages $g(\phi_{\Omega_x})$ to match the true score of ϕ_{Ω_x}, if it were considered a purely local, de-contextualized patch. The second term corresponds to the sum of gradients from every other patch ϕ_{Ω_y} that attended to x, which "encourages" $g(\phi_{\Omega_x})$ to provide more information about the image at position y. If the weight α_{xy} is large, then, we should see that if there's error when we evaluate at position x, our gradient should push us to change the value of $g(\phi_{\Omega_y})$ so that it becomes even more useful in reconstructing the image at position x. Thus, our score function is a weighted average of the CNN score function evaluated at that patch's location and the score function evaluated at every other patch that is informative about the patch at position x. This will encourage "copy-and-paste" behavior of patches that often appear together in a given image since the score will move the image in the direction of self-consistency because of the second term. While the general functional-gradient expression offers strong intuition for our empirical findings, deriving a closed-form solution in full generality proves intractable. We therefore specialize to an informative, tractable case in the next section that partially matches our experimental setup and provides more direct insight into the behavior of the score machine.

2.2. Simple CNN with Top-1 Attention

For simplicity, we assume that our attention is a "winner-take-all" regime, meaning that only the most attended to patch contributes to the sum. In particular, we have

$$\sum_{y} \alpha_{xy}\, g(\phi_{\Omega_y}) \;\longrightarrow\; g\big(\phi_{\Omega_{y^*(x)}}\big), \quad y^*(x) = \arg\max_{y} \langle g(\phi_{\Omega_x}),\, g(\phi_{\Omega_y}) \rangle. \tag{5}$$

We also assume "patch-independence" under the distribution π_t for all t, so that conditioning on $\phi_{\Omega_x} = \Phi$ does not change the distribution of ϕ_{Ω_y} for $y \neq x$ and that our embedding g is mean-centered over patches (ie, $\mathbb{E}_{\phi\sim\pi_t}\big[g(\phi_{\Omega_x})\big] = 0 \quad \forall\, x$). We then substitute the top-1 attention form

of Equation 5 into Equation 4. When we expand the the second term in Equation 4, we find that since we've assumed that g is mean-centered and that we have approximate patch independence, the second term goes away, leaving

$$\begin{aligned} 0 = &\sum_x \int \pi_t(\phi)\; 2\big(\tilde{g}[\phi](x) - s[\phi](x)\big)^\top \delta(\phi_{\Omega_x} - \Phi)\, d\phi \\ &+ \sum_x \int \pi_t(\phi)\; 2\big(\tilde{g}[\phi](x) - s[\phi](x)\big)^\top \delta(\phi_{\Omega_{y^*(x)}} - \Phi)\, d\phi. \end{aligned} \tag{6}$$

Since there is a deterministic mapping between a given patch ϕ_{Ω_x} and its most attended patch $\phi_{\Omega_{y^*(x)}}$, we see that the integrals with the delta peaks give closed form solutions:

$$\int \tilde{g}[\phi](x)\delta(\phi_{\Omega_x} - \Phi)d\phi = \int \tilde{g}[\phi](x)\delta(\phi_{\Omega^*_y(x)} - \Phi)d\phi = g[\Phi] + g[\Phi^*] := \tilde{g}[\Phi], \tag{7}$$

where $g[\Phi^*]$ denotes the output of the model on the patch most attended to by Φ. Distributing the deltas, integrating, and eventually dividing out by $\big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big]$, we arrive at the following solution for the optimal score function $\tilde{g}[\Phi]$:

$$\tilde{g}[\Phi] = \nabla_{\Phi(0)} \log \sum_x \big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big] \tag{8}$$

We refer readers to the appendix for the full derivation. If we then continue with the same procedure as equations 36-40 of [4] we see that the optimal score function is the gradient of a mixture of Gaussians centered at patches in the dataset *combined* with a mixture of Gaussians centered at the most attended corresponding patches. Intuitively, this implies that during the diffusion reverse process, each patch will be pulled towards its corresponding closest patch from the dataset, *but also* the corresponding closest patch *from the current image*. This second term is exactly what we term global self-consistency of the generated patches – they will tend to align with other patches in a current image.

3. Experiments

We validate our theory by showing that a simple attention-based diffusion model can learn and effectively reproduce such self-consistent structures in images while a CNN-based diffusion model struggles. We evaluate the capacity to construct consistent images by measuring how often the samples generated by our diffusion model obey the rules of this dataset. First, we construct a dataset to test this consistency property. Our dataset consists of 2048 4x4 RBG images in which each 2×2 block is filled with one of three possible color–color pairings (red/green vs. yellow/blue; red/yellow vs. green/blue; red/blue vs. green/yellow). These parings are randomized across images but are consistent within images. To construct our dataset, we randomly select one of three pattern types for each image (two disjoint color pairings, for example, red and green, and blue and yellow). Then, for each of the four quadrants in our image, we randomly select either a block of the first pairing type or a block of the second pairing type. In particular, this means that to successfully generate samples, our score must be sensitive to these consistent "key-value" pairings *within* images rather than the potentially inconsistent pairings in the dataset as a whole.

We train four simple diffusion models on this dataset: a pure CNN-based model, a CNN-based model with top-1 attention, a CNN-based model with attention with identity key and query matrices, and a CNN + Attention model. We use a standard DDPM setup and implement our score

network (or, equivalently, noise predictor) as a very simple 2 layer CNN with 2 convolutional layers, where the first convolutional layer is a 2x2 convolution with stride 2 and hidden dimension 32, and our second is a transpose convolution also with a 2x2 kernel and stride length of 2. This kernel and stride length help to encode the desired inductive biases and ensures that keys are encoded with their associated values. Our CNN+Attention model has exactly the same CNN base but includes a single-headed self-attention block with learnable Q, K, V matrices right before the final projection. We train all models identically. Details on the training and more information about the two modified attention models and their performance can be found in Appendix B.

Both qualitatively and quantitatively our attention-based model demonstrated increased consistency within images and higher quality image generation overall, as shown in Figure 1 and Table 1. It is clear that the CNN+Attention samples are both more visually consistent with the training samples and reproduce blocks of the correct size and colors, showing that attention is capturing the consistency of "key-value" pairs of a particular image. For each of our models (CNN and CNN+Attention) we generated 10k samples in 100 different runs. Over these 1 million samples generated by our attention-based model, 64.03% were consistent. We say that an image is "consistent" if its mapping has key-value pairs that remain constant throughout the image. For example, if one quadrant of the image contains a green-yellow pair, then if green appears as a key in any other quadrant it must be followed by yellow. We require this for each paring that appears in the image. In practice, for a given generated image, we map each pixel to its nearest color. Then, we split the split the map of labels into four quadrants and checked whether those mappings were one of the permissible sets. Finally, we computed a robust baseline "consistency percentage" by checking the consistency of 10k images generated by sampling four color pairs (with replacement), randomly assigning them to the four quadrants. We took the average percent consistent across 100 trials.

These consistency results match what our theory suggests: the second term in Equation 4 encourages the reconstruction of these key-value pairs that appear within a single image (self-consistency). In the CNN results, it is clear that while the model is reproducing features that appear across the dataset, including the quadrant layout, and some color pairing, it is unable to accurately construct these image-specific patterns without any non-local mechanism. This helps to explain some of the challenges that [4] faced when using their equivariant, local score machine to replicate the results of UNet, self-attention-based diffusion.

4. Discussion

The above theory offers a promising new step towards a more theoretically grounded understanding of how creativity emerges in diffusion models beyond the simple, purely convolutional case. Our empirical results on a purpose-built relational dataset support the theory: the CNN+Attention model generates significantly more self-consistent samples than its CNN-only counterpart. These findings suggest that even a single attention block can bridge the gap between local patch mosaics and full image consistency. While our theory here has been restricted to the very simple CNN+attention case, we hope to extend the theory in future work to a Unet + self-attention framework and replicate our results on larger datasets of natural images.

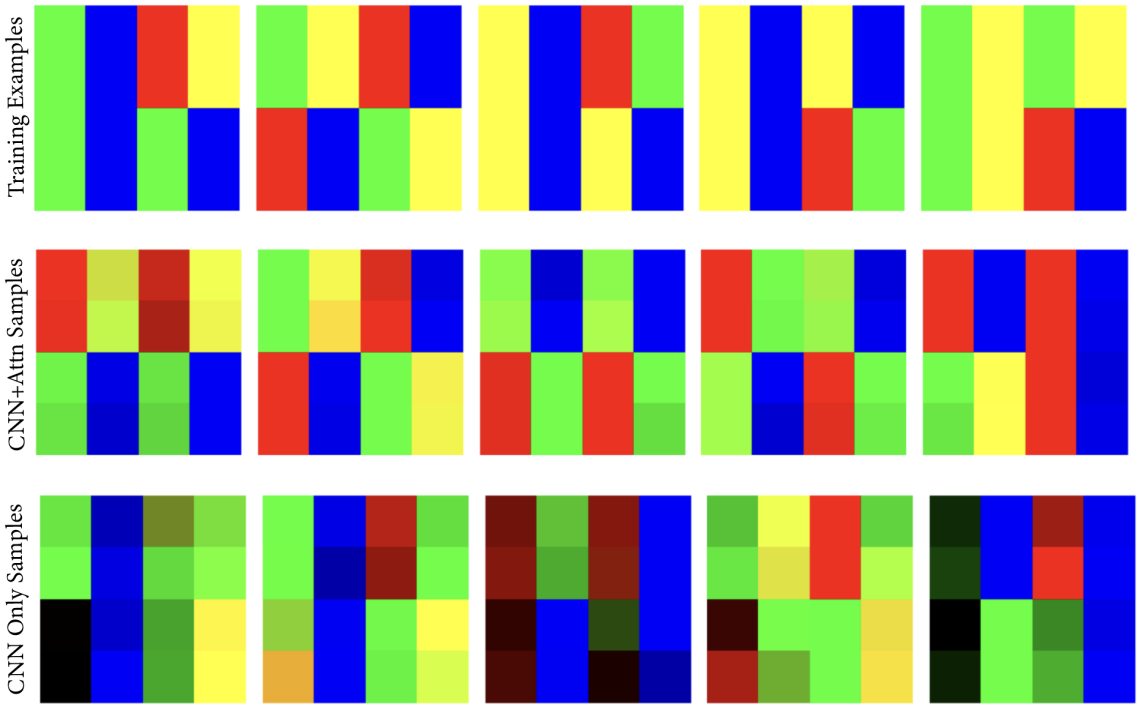

Figure 1: Dataset and generated images.

Model	Consistency
Random Baseline	5.38%
CNN	10.88%
CNN + Attention	64.03%

Table 1: Self-consistency of generated samples.

Acknowledgments

This work was supported by the Kempner Institute for the Study of Natural and Artificial Intelligence at Harvard University. We thank the Kempner for access to compute resources. We are also grateful to the CRISP group at Harvard SEAS for many thoughtful conversations. EF thanks the Calvin Coolidge Presidential Foundation for support during her undergraduate studies.

References

[1] Giulio Biroli, Tony Bonnaire, Valentin De Bortoli, and Marc Mézard. Dynamical regimes of diffusion models, 2024. URL https://arxiv.org/abs/2402.18491.

[2] Jonathan Ho, Ajay N. Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. In *Advances in Neural Information Processing Systems*, volume 33, pages 6840–6851. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html.

[3] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax, 2017. URL https://arxiv.org/abs/1611.01144.

[4] Mason Kamb and Surya Ganguli. An analytic theory of creativity in convolutional diffusion models, 2024. URL https://arxiv.org/abs/2412.20292.

[5] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, CA, 2015. doi: 10.48550/arXiv.1412.6980. URL https://arxiv.org/abs/1412.6980.

[6] Sixu Li, Shi Chen, and Qin Li. A good score does not lead to a good generative model, 2024. URL https://arxiv.org/abs/2401.04856.

[7] Kevin P. Murphy. *Probabilistic Machine Learning: Advanced Topics*. MIT Press, 2023. URL http://probml.github.io/book2.

[8] William Peebles and Saining Xie. Scalable diffusion models with transformers, 2023. URL https://arxiv.org/abs/2212.09748.

[9] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 2256–2265. PMLR, 2015. URL http://proceedings.mlr.press/v37/sohl-dickstein15.html.

[10] VSehwag. minimal-diffusion: A minimal yet resourceful implementation of diffusion models. https://github.com/VSehwag/minimal-diffusion, 2024. GitHub repository, commit 2bd59ee on Sep 4, 2024; accessed 2025-05-20.

Appendix A. Derivations

A.1. General Attention Derivation

Here, we'll derive the self-consistency constraint for a simple attention mechanism mentioned in Section 2.1. We'll recall the notation used in [4]

1. ϕ is an arbitrary image in the diffusion process
2. x is a pixel location and Λ is the set of pixel locations in a given image.
3. Ω_x is the neighborhood of pixels centered at x
4. ϕ_{Ω_x} is the patch of the image ϕ centered at x.
5. π_t is the underlying probability distribution of images at time t
6. $s_t[\phi](x)$ is the true score at time t evaluated at position x in the image ϕ.

The original score matching loss function given in the paper [4] is given by

$$\mathcal{L} = \sum_x \mathbb{E}_{\phi\sim\pi_t}\Big[\big|\big|f(\phi_{\Omega_x}) - s_t[\phi](x)\big|\big|^2\Big]. \tag{9}$$

We want to modify this so that we actually have $f(\phi_{\Omega_x})$ as an attention based update rather than just looking at one neighborhood of the image. First, we embed each patch using g, a convolutional embedding network, which can be thought of as the first portion of our score-approximation network. It is the only portion of our architecture with learnable parameters.

1. Let z_x denote the embedding of the patch ϕ_{Ω_x} in $\mathbb{R}^d$, where $z_x = g(\phi_{\Omega_x})$
2. Let $(z_y)_{y\in\Lambda}$ be the collection of flattened patches of the image ϕ that we're currently looking at.

Differing from [4], in this work, instead of a purely local estimator $g(\phi_{\Omega_x})$, we define:

$$\tilde{g}[\phi](x) \;=\; z_x \;+\; \sum_y \alpha_{xy}\, z_y \qquad \alpha_{xy} \;=\; \frac{\exp\big(\langle z_x, z_y\rangle\big)}{\sum_{y'}\exp\big(\langle z_x, z_{y'}\rangle\big)}$$

Now we define our new model, which we'll call $\tilde{g}$.

$$\tilde{g}(\phi)(x) = z_x + \sum_y \text{softmax}_y(\langle z_x, z_y\rangle) \tag{10}$$

where $\langle z_x, z_y\rangle$ denotes the dot product of our vector embeddings. We write $\alpha_{xy} = \text{softmax}_y(\langle z_x, z_y\rangle)$. The y subscript says that we're normalizing over the y indexes so that for a fixed x, the sum of the α_{xy} is 1. Intuitively, a big value α_{xy} says that the position x should pay a lot of attention to the position y.

Thus, our loss function becomes

$$\mathcal{L} = \sum_x \mathbb{E}_{\phi\sim\pi_t}[||z_x + \sum_y(\text{softmax}_y(z_x, z_y)z_y - s[\phi](x)||^2] \tag{11}$$

Rewriting this in terms of our function g, we have

$$\mathcal{L} = \sum_x \mathbb{E}_{\phi \sim \pi_t}[||g(\phi_{\Omega_x}) + \sum_y (\text{softmax}_y(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}))g(\phi_{\Omega_y}) - s[\phi](x)||^2] \tag{12}$$

Now we need to take the functional derivative of this. First we'll rewrite the expectation as an integral and then

$$\mathcal{L} = \sum_x \int [||g(\phi_{\Omega_x}) + \sum_y (\text{softmax}_y(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}))g(\phi_{\Omega_y}) - s[\phi](x)||^2]\pi_t(\phi)d\phi \tag{13}$$

Now we find the function g which minimizes this loss. In particular, we'll assert

$$\frac{\delta \mathcal{L}}{\delta g(\Phi)} = 0 \text{ for each possible patch } \Phi \tag{14}$$

Now we need to take this derivative.

$$0 = \sum_x \frac{\delta}{\delta g(\Phi)} \int \pi_t(\phi)[||g(\phi_{\Omega_x}) + \sum_y (\text{softmax}_y(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}))g(\phi_{\Omega_y}) - s[\phi](x)||^2]d\phi \tag{15}$$

We can differentiate under the integral sign since the bounds don't depend on what we're taking the derivative with respect to, so

$$0 = \sum_x \int \pi_t(\phi) \frac{\delta}{\delta g(\Phi)} \left([||g(\phi_{\Omega_x}) + \sum_y (\text{softmax}_y(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}))g(\phi_{\Omega_y}) - s[\phi](x)||^2] \right) d\phi \tag{16}$$

Now we'll apply the vector-calculus chain rule to see that we get

$$0 = \sum_x \int \pi_t(\phi)\, 2 \cdot \Bigg(g(\phi_{\Omega_x}) + \sum_y \left(\text{softmax}_y\big(g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\big) g(\phi_{\Omega_y})\right) - s[\phi](x) \Bigg)^T \frac{\delta}{\delta g(\Phi)} \left(g(\phi_{\Omega_x}) + \sum_y \left(\text{softmax}_y\big(g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\big) g(\phi_{\Omega_y})\right) \right) d\phi \tag{17}$$

Now we need to finish the computation of the derivative of the derivative of self-attention component.

Now we'll find

$$\frac{\delta}{\delta g(\Phi)} \left[g(\phi_{\Omega_x}) + \sum_y \text{softmax}_y \left(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}) \right) g(\phi_{\Omega_y}) \right] \tag{18}$$

$$= \frac{\delta}{\delta g(\Phi)} g(\phi_{\Omega_x}) + \sum_y \frac{\delta}{\delta g(\Phi)} \left[\text{softmax}_y \left(g(\phi_{\Omega_x}), g(\phi_{\Omega_y}) \right) \cdot g(\phi_{\Omega_y}) \right] \tag{19}$$

$$= \delta(\phi_{\Omega_x} - \Phi) + \sum_y \frac{\delta}{\delta g(\Phi)} \frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} \cdot g(\phi_{\Omega_y}) \tag{20}$$

where the first term is the Dirac delta function since the derivative is only non-zero on the patch of interest.
Now we need to use the product rule.

$$= \delta(\phi_{\Omega_x} - \Phi) + \sum_y \left(\frac{\delta}{\delta g(\Phi)} \left(\frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} \right) \cdot g(\phi_{\Omega_y}) + \left(\frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} \right) \cdot \frac{\delta}{\delta g(\Phi)} g(\phi_{\Omega_y}) \right) \tag{21}$$

From above, we know that the $\frac{\delta}{\delta g(\Phi)} g(\phi_{\Omega_y})$ should just turn into a Dirac delta.

$$= \delta(\phi_{\Omega_x} - \Phi) + \sum_y \left(\frac{\delta}{\delta g(\Phi)} \left(\frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} \right) \cdot g(\phi_{\Omega_y}) + \left(\frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} \right) \cdot \delta(\phi_{\Omega_y} - \Phi) \right) \tag{22}$$

Now all that remains is to use the chain rule on the first term. We're now looking just at this term here where we need to take the derivative of the attention value with respect to $g(\Phi)$. This attention value is a scalar and we're taking the derivative with respect to a vector. We'll denote the attention α_{xy}

$$\frac{\delta \alpha_{xy}}{\delta g(\Phi)} = \nabla_{g(\phi_{\Omega_x})} \alpha_{xy} \cdot \frac{\delta g(\phi_{\Omega_x})}{\delta g(\Phi)} + \nabla_{g(\phi_{\Omega_y})} \alpha_{xy} \cdot \frac{\delta g(\phi_{\Omega_y})}{\delta g(\Phi)} \tag{23}$$

Notice that the second term in both of these turn into Dirac delta functions.

$$\frac{\delta \alpha_{xy}}{\delta g(\Phi)} = \nabla_{g(\phi_{\Omega_x})} \alpha_{xy} \delta(\phi_{\Omega_x} - \Phi) + \nabla_{g(\phi_{\Omega_y})} \alpha_{xy} \delta(\phi_{\Omega_y} - \Phi) \tag{24}$$

Now, for the derivative of the attention itself, recall that $\alpha_{xy} = \frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}}$

Then we can compute

$$\nabla_{g(\phi_{\Omega_x})} \alpha_{xy} = \nabla_{g(\phi_{\Omega_x})} \frac{e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_y})\rangle}}{\sum_{y'} e^{\langle g(\phi_{\Omega_x}), g(\phi_{\Omega_{y'}})\rangle}} = \alpha_{xy} \left(g(\phi_{\Omega_y}) - \sum_{y'} \alpha_{xy'} g(\phi_{\Omega_{y'}}) \right) \tag{25}$$

This is a standard attention derivative, so putting it all together we have

$$0 = \sum_x \int \pi_t(\phi)\, 2\left(g(\phi_{\Omega_x}) \;+\; \sum_y \alpha_{xy}(\phi)\, g(\phi_{\Omega_y}) \;-\; s[\phi](x)\right)^T$$
$$\Bigg[\underbrace{\delta\big(\phi_{\Omega_x} - \Phi\big)}_{\substack{\text{derivative wrt}\\ g(\phi_{\Omega_x})}} \;+\; \sum_y \Bigg(\underbrace{\alpha_{xy}(\phi)\left[g(\phi_{\Omega_y}) \;-\; \sum_{y'} \alpha_{xy'}(\phi)\, g(\phi_{\Omega_{y'}})\right]\delta\big(\phi_{\Omega_x} - \Phi\big)}_{\substack{\text{chain rule wrt}\\ g(\phi_{\Omega_x}) \text{ in } \alpha_{xy}}}$$
$$+\; \underbrace{\alpha_{xy}(\phi)\left[g(\phi_{\Omega_x}) \;-\; \sum_{y'} \alpha_{xy'}(\phi)\, g(\phi_{\Omega_{y'}})\right]\delta\big(\phi_{\Omega_y} - \Phi\big)}_{\substack{\text{chain rule wrt}\\ g(\phi_{\Omega_y}) \text{ in } \alpha_{xy}}} \;+\; \underbrace{\alpha_{xy}(\phi)\,\delta\big(\phi_{\Omega_y} - \Phi\big)}_{\substack{\text{derivative wrt}\\ g(\phi_{\Omega_y}) \text{ itself}}}\Bigg)\Bigg] d\phi \tag{26}$$

We can simplify this. We claim that

$$\sum_y \Bigg(\underbrace{\alpha_{xy}(\phi)\left[g(\phi_{\Omega_y}) \;-\; \sum_{y'} \alpha_{xy'}(\phi)\, g(\phi_{\Omega_{y'}})\right]\delta\big(\phi_{\Omega_x} - \Phi\big)}_{\substack{\text{chain rule wrt}\\ g(\phi_{\Omega_x}) \text{ in } \alpha_{xy}}}\Bigg) = 0$$

Since setting $\mu_x = \sum_{y'} \alpha_{xy'}(\phi)\, g(\phi_{\Omega_{y'}})$ and recalling that $\sum_y \alpha_{xy} = 1$, these terms cancel. Finally, we recover the form

$$0 = \sum_x \int \pi_t \left[g(\phi_x) + \sum_y \alpha_{xy} g(\phi_{\Omega_y}) - s[\phi](x)\right]^T \delta(\phi_{\Omega_x} - \Phi)\, d\phi$$
$$+ \int \pi_t \left[g(\phi_x) + \sum_y \alpha_{xy} g(\phi_{\Omega_y}) - s[\phi](x)\right]^T \sum_y \alpha_{xy}\left(I + (g(\phi_{\Omega_x}) - \mu_x)\delta(\phi_{\Omega_y} - \Phi)\right) d\phi \tag{27}$$

A.2. Top 1 Attention Derivation

We begin with the same setup as above, where we assume that our attention is a "winner-take-all" regime, meaning that only the most attended to patch contributes to the sum. In particular, we have

$$\sum_y \alpha_{xy}\, g(\phi_{\Omega_y}) \;\longrightarrow\; g\big(\phi_{\Omega_{y^*(x)}}\big), \quad y^*(x) = \arg\max_y \langle g(\phi_{\Omega_x}),\, g(\phi_{\Omega_y})\rangle.$$

We also assume "patch-independence" under the distribution π_t for all t, so that conditioning on $\phi_{\Omega_x} = \Phi$ does not change the distribution of ϕ_{Ω_y} for $y \neq x$ and that our embedding g is mean-centered over patches (ie, $\mathbb{E}_{\phi\sim\pi_t}\big[g(\phi_{\Omega_x})\big] = 0 \quad \forall\, x$). We then substitute the top-1 attention form where $\sum_y \alpha_{xy}\, g(\phi_{\Omega_y}) = g\big(\phi_{\Omega_{y^*(x)}}\big)$ into Equation 4. When we expand the the second term in

Equation 4, we find that since we've assumed that g is mean-centered and that we have approximate patch independence, the second term goes away, leaving

$$\begin{aligned} 0 = &\sum_x \int \pi_t(\phi)\, 2\big(\tilde{g}[\phi](x) - s[\phi](x)\big)^\top \delta(\phi_{\Omega_x} - \Phi)\, d\phi \\ &+ \sum_x \int \pi_t(\phi)\, 2\big(\tilde{g}[\phi](x) - s[\phi](x)\big)^\top \delta(\phi_{\Omega_{y^*(x)}} - \Phi)\, d\phi. \end{aligned} \tag{28}$$

Since there is a deterministic mapping between a given patch ϕ_{Ω_x} and its most attended patch $\phi_{\Omega_{y^*(x)}}$, we see that the integrals with the delta peaks give closed form solutions:

$$\int \tilde{g}[\phi](x)\delta(\phi_{\Omega_x} - \Phi)d\phi = \int \tilde{g}[\phi](x)\delta(\phi_{\Omega^*_y(x)} - \Phi)d\phi = g[\Phi] + g[\Phi^*] := \tilde{g}[\Phi], \tag{29}$$

where $g[\Phi^*]$ denotes the output of the model on the patch most attended to by Φ.

Thus, distributing the deltas, integrating over these terms, and moving them to the other side:

$$\begin{aligned} &\tilde{g}(\Phi)\sum_x \pi_t(\phi_{\Omega_x} = \Phi) + \tilde{g}(\Phi)\sum_x \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi) = \\ &\sum_x \int \pi_t(\phi)s[\phi](x)\delta(\phi_{\Omega_x} - \Phi)\, d\phi + \sum_x \int \pi_t(\phi)s[\phi](x)\delta(\phi_{\Omega_{y^*(x)}} - \Phi)\, d\phi \end{aligned} \tag{30}$$

By the linearity of the integrals and sums we get:

$$\begin{aligned} \tilde{g}(\Phi)\sum_x \big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big] &= \sum_x \int \pi_t(\phi)s[\phi](x)\big[\delta(\phi_{\Omega_x} - \Phi) + \delta(\phi_{\Omega_{y^*(x)}} - \Phi)\big]\, d\phi \\ &= \sum_x \int \nabla_{\phi(x)}\pi_t(\phi)\big[\delta(\phi_{\Omega_x} - \Phi) + \delta(\phi_{\Omega_{y^*(x)}} - \Phi)\big]\, d\phi \\ &= \sum_x \nabla_{\Phi(0)}\big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big] \end{aligned} \tag{31}$$

Dividing by $\big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big]$, yields

$$\tilde{g}(\Phi) = \nabla_{\Phi(0)} \log \sum_x \big[\pi_t(\phi_{\Omega_x} = \Phi) + \pi_t(\phi_{\Omega_{y^*(x)}} = \Phi)\big] \tag{32}$$

Appendix B. Additional Results and Training Details

We also trained a simple CNN with a top-1 attention layer on the end to better match our theory and a simple CNN with attention given by the simple identity matrix query and key matrices. Both of these models had a standard DDPM setup and we implemented our score network (or, equivalently noise predictor) as a very simple 2 layer CNN with 2 convolutional layers, where the first convolutional layer is a 2x2 convolution with stride 2 and hidden dimension 32, and our second is a transpose convolution also with a 2x2 kernel and stride length of 2. The top-1 attention layer used Gumbel Softmax since argmax (which would be the natural way to implement top-1 attention isn't differentiable [3]. Because of this, training was more difficult, so we trained this model for $10,000$ epochs. Other than that, the training was the same as the two models mentioned above. We used a batch size of 64 over 5000 epochs, AdamW with learning rate 10^{-3} and weight decay 10^{-5} under a OneCycleLR schedule. Additionally, we maintain an exponential moving average (EMA) of the model parameters with decay 0.9999, updating it after each optimizer step. During sample generation, we use the EMA weights to improve stability [5]. We also used a linear noise schedule and weight the MSE loss by $1 - \alpha_{\text{cumulative product}}[t]$, as is standard.

We found that both the CNN+Top1 Attention and CNN+Identity Attention both outperformed the CNN only architecture, but failed to achieve the same quantitative and qualitative results as the full attention model. In particular, Gumbel-Softmax-implemented Top-1 attrition resulted in 21.64% consistency across 100 trials, while the identity attention model resulted in 25.44% consistency across 100 trials.

Appendix C. Additional Samples

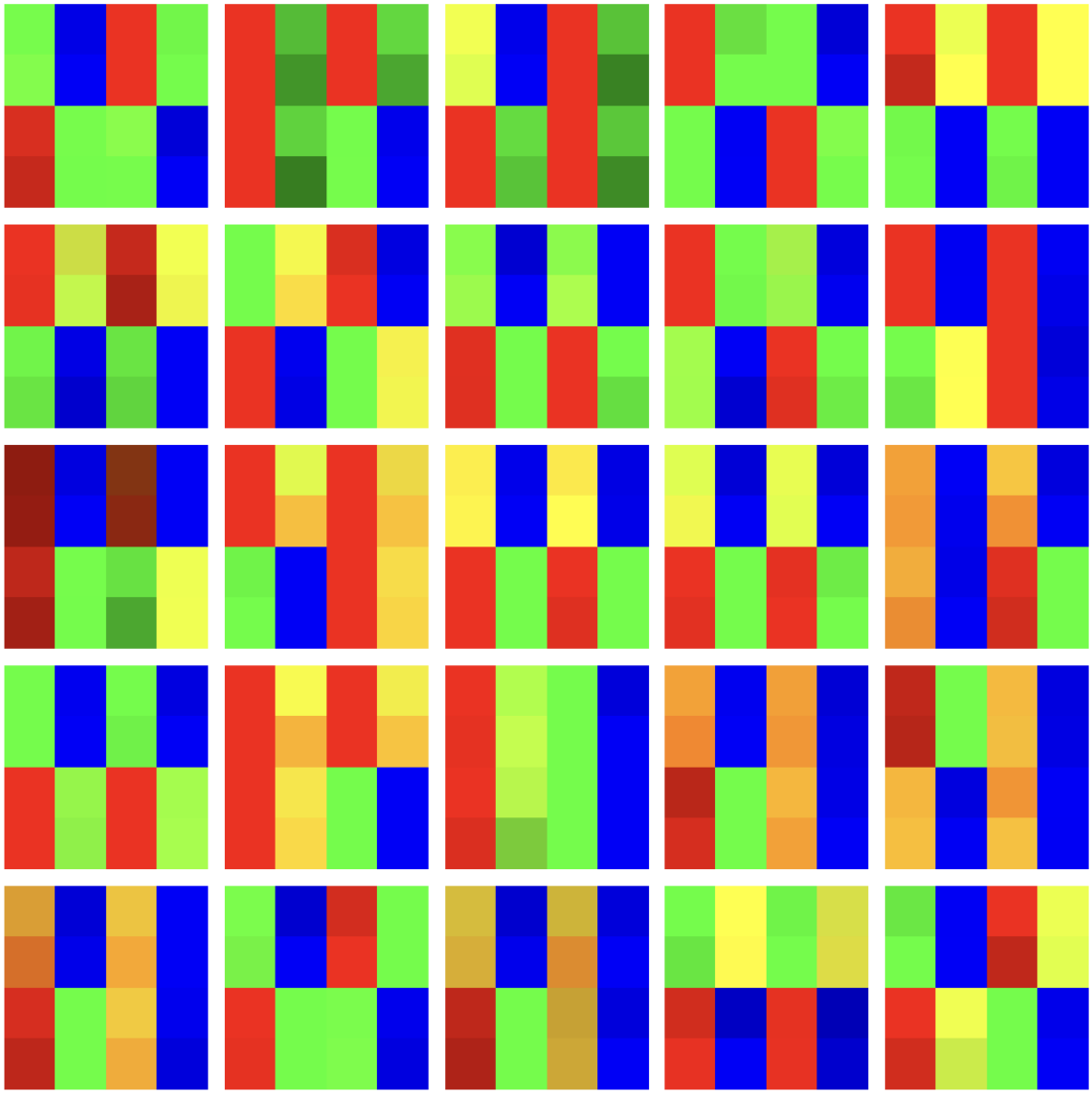

Figure 2: More Samples from CNN+Attention

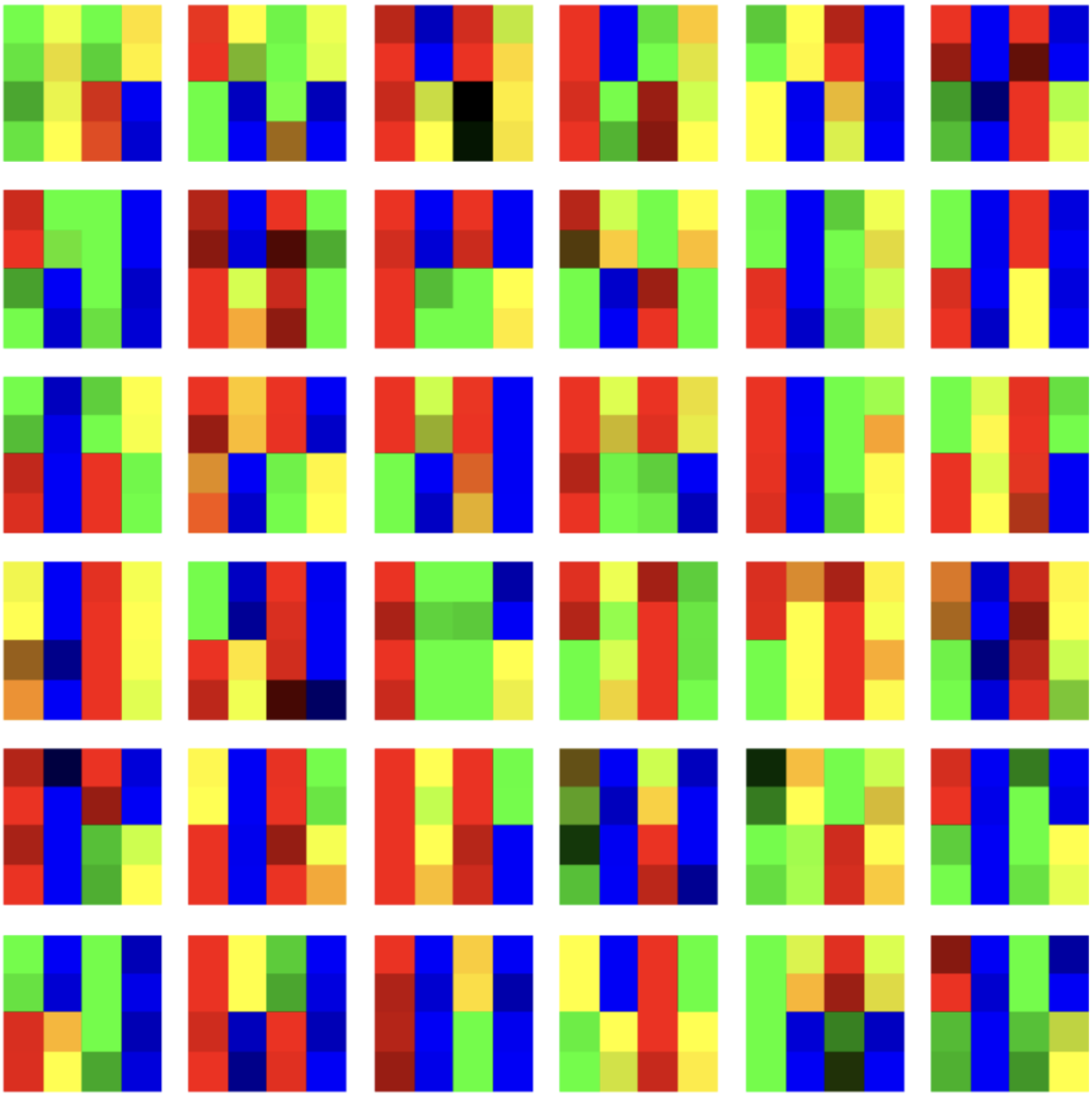

Figure 3: More Samples from CNN+ Top 1 Attention

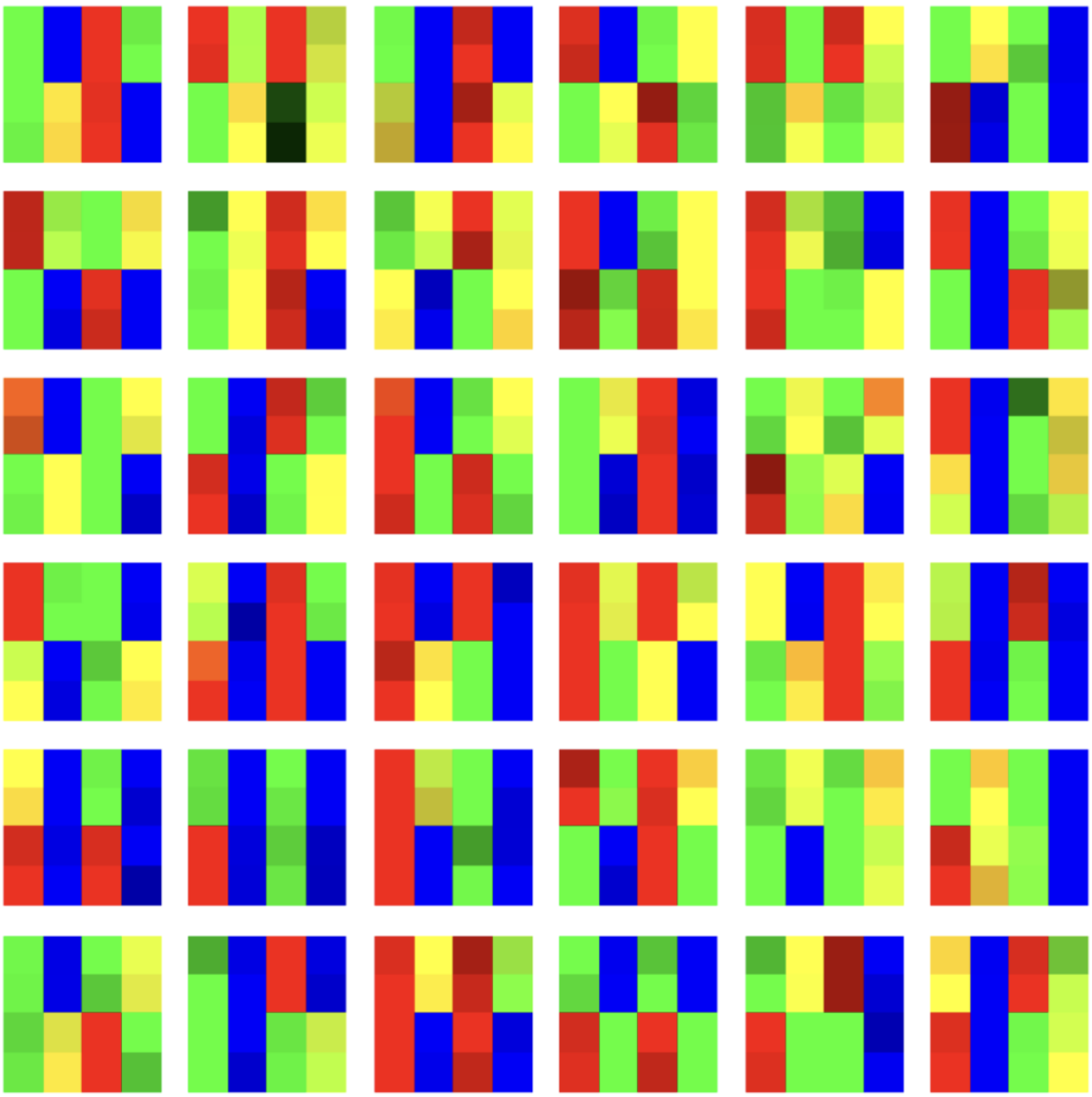

Figure 4: More Samples from Identity Attention

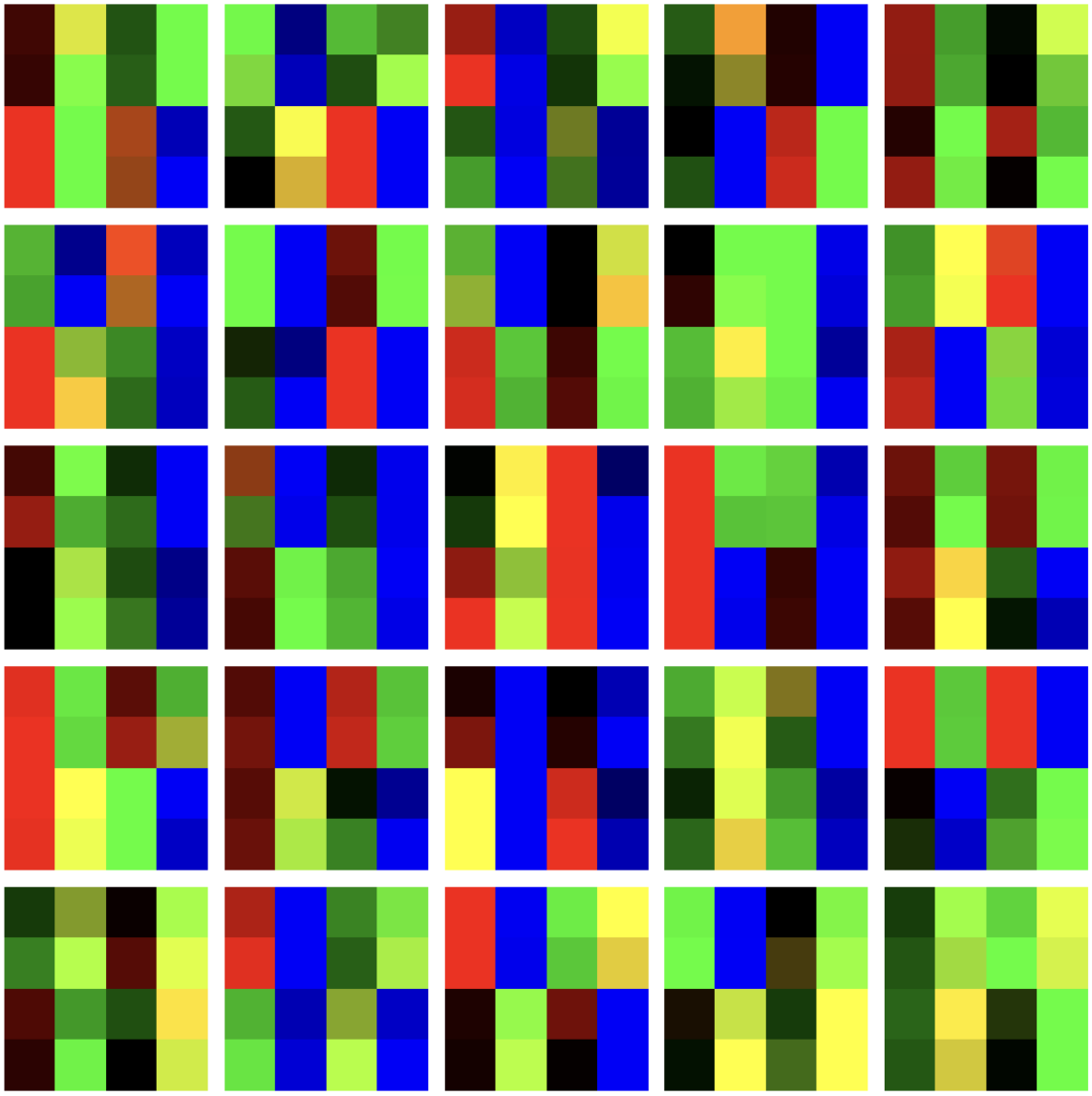

Figure 5: More Samples from CNN