Evaluation of Out-of-Bounds Pattern Conservation in Implicit Neural Representation Networks

Anonymous Authors[1]

Abstract

Implicit neural representation (INR) networks bring a new perspective to machine learning by treating individual samples as functions, with granular output characteristics dependent on coordinates within a particular sample. They have proven successful at learning scale-independent functional representations of images, as well as capturing higher-order derivatives of representations and solving inverse problems. However, a characteristic of these models that has not undergone extensive study is their behavior outside of the bounds of the training sample. Here, I apply a number of INR models to learning visual patterns and evaluate their performance on out-of-bounds pattern expansion. This study finds that most of the tested models are incapable of pattern expansion, with the exception of the Fourier Kolmogorov-Arnold Network (FKAN) model. This result could be used to generate irregular tilings from sample images.

1. Introduction

In recent decades, a diverse and productive range of machine learning methods have been developed. Whether utilizing neural network-based architectures or not, the general paradigm of these models is quite consistent: there is a set of samples which a model is given to train upon in order to produce predictions, and then it is evaluated on withheld test samples to evaluate its reliability with new data. INR models alter this paradigm by bringing a new perspective to the idea of what makes up a sample. In these models, each data point is the granular component defined by its location in the traditional sample, such as a the color of a pixel defined by its coordinates in an image. As such, these models train on single samples, guided by a point-wise reconstruction loss. As the model acts as a function from location to characteristic(s), post-training generation is independent of scale, with the model able to generate shrunken or enhanced versions of the original image with just as much ease (Sitzmann, 2020).

While INR models could theoretically take any form, two model types have shown heightened image reconstruction ability: sinusoidal representation networks (SIRENs) and Fourier feature networks (FFNs) (Sitzmann, 2020; Tancik, 2020). Both of these networks introduce sinusoidal elements through different mechanisms. For SIRENs, this occurs through the activation function, which is sinusoidal (Sitzmann, 2020). On the other hand, FFNs map input coordinates to Fourier features before being put through a sigmoid-activated MLP (Tancik, 2020). Another intriguing type of implicit network combines the idea of Fourier features with the recently developed Kolmogorov-Arnold networks (KANs) (Mehrabian, 2024). KANs are particularly flexible in their learning with even small architectures due to learnable spline-based activation functions; in the case of Fourier KAN (FKAN), this enables precise implicit image reconstruction (Liu, 2024; Mehrabian, 2024).

The hypothesis behind this work is that the sinusoidal features which enable the discussed models to achieve precise implicit reconstructions will also enable improved out-of-bounds training pattern tiling. While tiling a precise pattern such as a checkerboard is algorithmically easy, extending patterns which are not precise tiles or repetitions is not a simple process to achieve seamlessly. This work thus inspects the idea of out-of-bounds tile sampling on simple and complex visual patterns using the discussed implicit representation networks.

2. Related Work

Although much work has been done to create precise implicit representations, there have been few studies on implicitly learning pattern tilings. One such study is from Chen et al., describing an approach for patterned facade inpainting useful in removing pattern-blocking objects from images (2022). This model works through periodicity-searching

[1] Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

and refinement modules followed by the more traditional coordinate-based MLP module. Although this method accomplishes many of the purposes this work sets out to evaluate, the simpler models described have not been evaluated in such capacity. Most studies also focus exclusively on visual data, while this study intends to use both visual and audio data for evaluation.

3. Methodology

3.1. Data

The data tested stems from two images: the first is various cropped regions of a checkerboard pattern. The second is again cropped regions of a plaid tartan pattern from Peter Eslea MacDonald of the Scottish Tartans Authority available on Wikimedia, which was constrained to a single channel. In the code, images are all resized to the same size (128x128), with the exception of the FKAN architecture which received shrunken versions (64x64) of the same images due to its larger memory burden. The tartan pattern is a regular one despite its complexity; no irregular patterns were tested due to unsatisfactory patterning results on even regular patterns. These patterns served to determine feasibility for more complex tasks.

3.2. Non-Sinusoidal Networks

All networks used were based around the same form of an input layer taking 2-dimensional image coordinates, three hidden layers (with hidden dimension 256), and an output layer yielding the value for a single channel. Unless otherwise specified, ReLU activations were used by default. The Tanh architecture replaced these with Tanh activation functions. The radial basis function (RBF)-ReLU model inspired by Sitzmann et al. replaced the first layer with a RBF layer where each neuron learns a Gaussian over the image plane (2020).

3.3. Sinusoidal Networks

The SIREN implementation was again inspired by Sitzmann et al. This replaces all the layers of the previously described model except the last with sine layers (2020). These apply a sin activation function to a linear layer, and a rescaling frequency coefficient omega was chosen as 30. The FFN implementation was inspired by Tancik et al. and maintains the same model architecture as the original ReLU other than an input dimension which was chosen as 256 (2020). This enables the Fourier feature reparameterization process to turn the two input features into 256 Fourier features. A Gaussian initialization of the Fourier coefficient matrix was chosen with a multiplicative factor of 10. Lastly, following Mehrabian et al., the FKAN network was initialized as a vanilla network with the first layer replaced by a KAN layer but using Chebyshev polynomials rather than the traditional splines (2024). For this layer a grid size of 100 was found to work well. One notable difference from this work's implementation of this model and the one in the Mehrabian paper is that the remainder of the layers are not sine layers but instead have ReLU activation. The decision to include sine layers was not made clear in the original paper, so I wanted to test the benefits or drawbacks of the FKAN layer on its own without support from SIREN capabilities.

3.4. Training and Evaluation

Training was performed with the same number of epochs for each model: 301 for the checker patterns and 601 for the tartan patterns. The training loss used for all was mean squared error (MSE) compared against the original image (Appendix Table 1). In terms of evaluation, the trained model outputs were inspected visually both in the region of training (Figure 1) and in a 10x zoomed out view to see if any pattern extension occurred (Figure 2). In addition, a tile to the right of the trained boundary was compared using MSE loss against the ground truth image to evaluate the extension to nearby tiles (Appendix Table 2). Similarly, a further tile was chosen to determine if there were substantial differences in pattern reconstruction that were not dependent on distance from the trained image boundary (Appendix Table 3).

4. Results

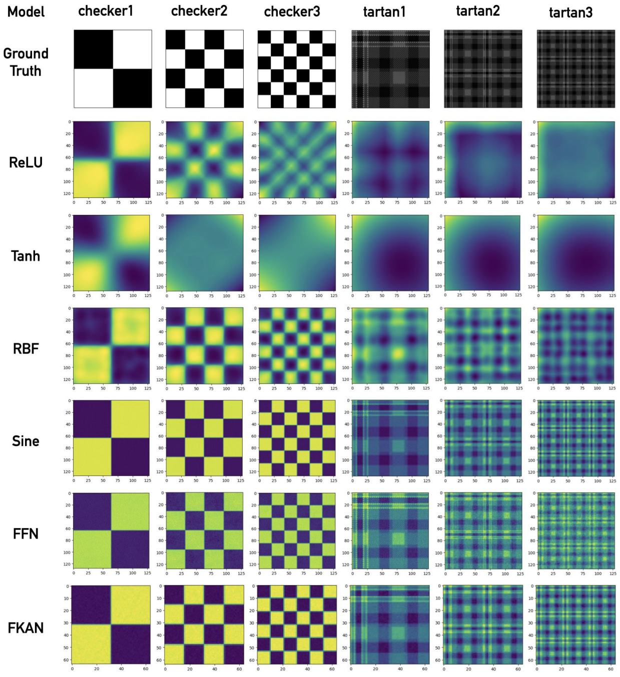

Figure 1. Patterns trained for each model.

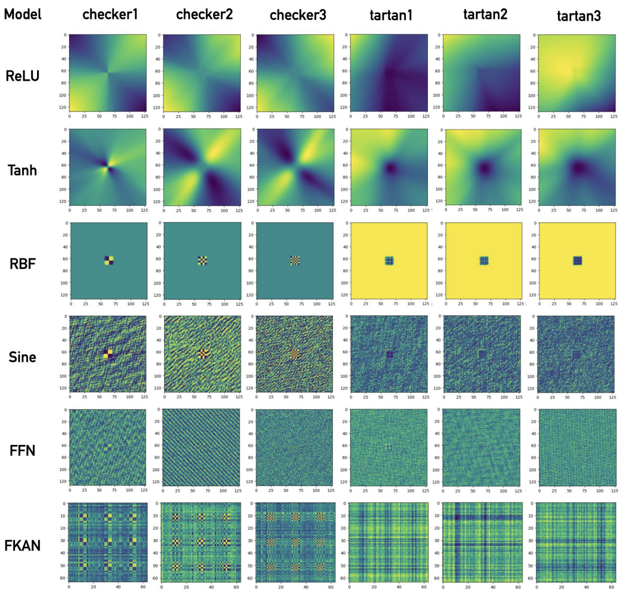

Figure 2. Coordinates put through trained model from 10x zoomed-out view of training bounds for each pattern and model.

The results from Figure 1 show similar conclusions to the Sitzmann et al. paper, that SIREN (Sine) greatly outperforms the fitting of ReLU, Tanh, and RBF models (2020). This is particularly evident in the more complex tartan patterns and also with the increasing granularity of the checker patterns. FFN and FKAN seem to achieve a similar level of precision, if a bit blurrier in the smaller details. These models were quite similar in training loss, as seen in the top graphs of Figure 3 and Figure 4.

Moving to Figure 2, there are more interesting differences. The ReLU and Tanh models seem to be able to learn mostly entire field-wide divisions (and only blurry ones at that). This is especially pronounced in Figure 3 and Figure 4 in the bottom two graphs for the ReLU model, which keeps increasing or decreasing in value further from the origin, causing astronomical loss. The RBF model, as it consists of a fixed number of Gaussians, only changes in value within the bounds of the original image or very close to the border (Figure 2). This often makes it a winner amongst the pattern construction losses in Table 2 and Table 3, as other models have larger values which create larger loss even if they are visually more identifiable. Surprisingly, although the SIREN (Sine) model consists of sinusoidal neurons, the repetitions outside the bounds of the original figure seem to remain snakey and do not take on very much of the pattern (Figure 2). The FFN model, on the other hand, is able to capture some of the fine effects of the pattern expanded outward, though this still gives a noisy appearance from the zoomed-out view. The best model, visually, is certainly the FKAN. This model does not reliably reproduce the pattern in a tight grid but seems to nevertheless give repetitions of the checkered pattern in a spaced grid. Its expansion of the tartan patterns, while not precise, certainly creates more of the effect of the tartan from afar tahn any of the other models.

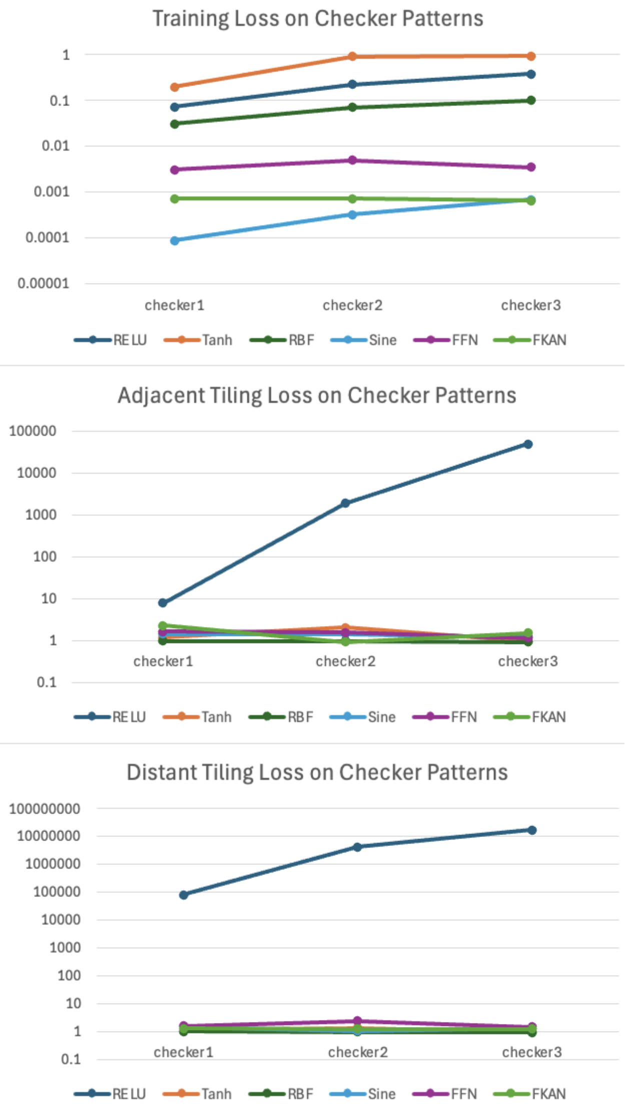

Figure 3. Tiling MSE loss from training, adjacent tile, and distant tile for each model trained on checker images.

5. Conclusion

The results of this study therefore demonstrate that, with the exception of the FKAN model, the tested models have very little ability to generalize a learned pattern beyond the bounds of the implicit network. Further improvments to this model could potentially encourage more accurate grid behavior, or enable seamless stitching and expansion of even irregular patterns.

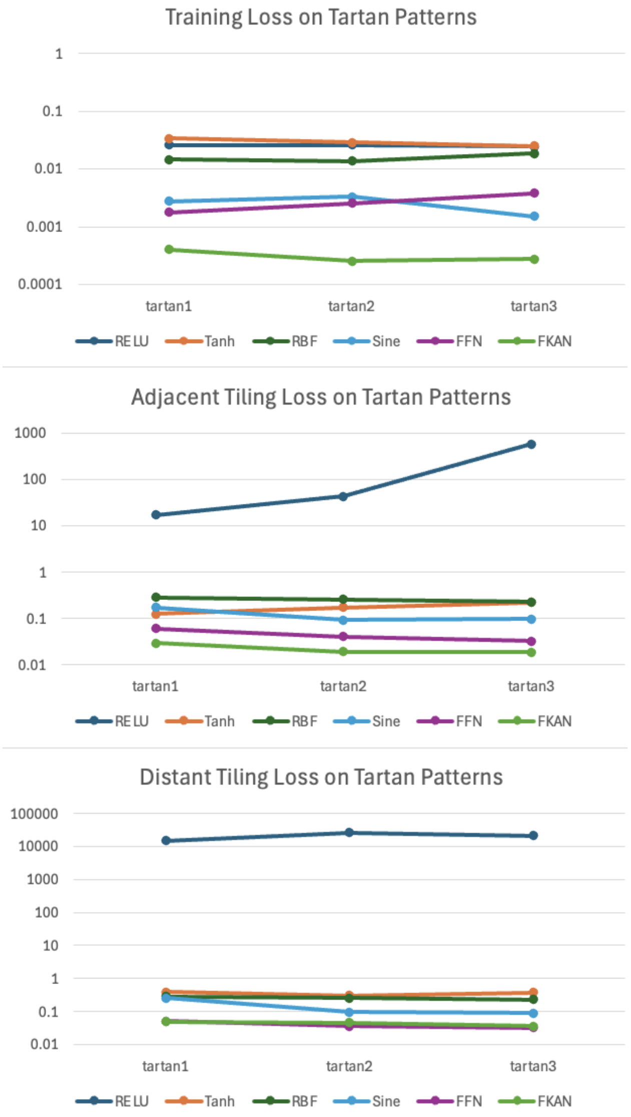

Figure 4. Tiling MSE loss from training, adjacent tile, and distant tile for each model trained on tartan images.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of this work, none which we feel must be specifically highlighted here.

References

Chen, B. Learning continuous implicit representation for near-periodic patterns. In *Computer Vision – ECCV 2022*, pp. 529–546. Springer Nature Switzerland, 2022. URL https://doi.org/10.1007/978-3-031-19784-0_31.

Liu, Z. Kan: Kolmogorov–arnold networks, 2024. URL https://arxiv.org/abs/2404.19756.

Mehrabian, A. Implicit neural representations with fourier kolmogorov-arnold networks, 2024. URL https://arxiv.org/abs/2409.09323.

Sitzmann, V. Implicit neural representations with periodic activation functions. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7462–7473. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/53c04118df112c13a8c34b38343b9c10-Paper.pdf.

Tancik, M. Fourier features let networks learn high frequency functions in low dimensional domains. In *Advances in Neural Information Processing Systems*, volume 33, pp. 7537–7547. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/55053683268957697aa39fba6f231c68-Paper.pdf.

Table 1. Mean squared error loss on presented image by model. Boldface shows lowest loss for each image. Since FKAN was given a smaller image to train on, if its loss is lowest then the second lowest is also bolded.

Model	checker1	checker2	checker3	tartan1	tartan2	tartan3
RELU	0.071754	0.220923	0.374274	0.02619	0.026028	0.024284
Tanh	0.194033	0.905731	0.921397	0.033366	0.028509	0.025154
RBF	**0.030427**	0.069407	0.099213	0.014419	0.01362	0.018397
Sine	**0.000088**	**0.000319**	**0.000673**	0.002778	0.003263	**0.001508**
FFN	0.00303	0.004922	0.003443	**0.00176**	**0.002548**	0.003828
FKAN	0.000712	0.000713	**0.00064**	**0.0004**	**0.000255**	**0.000275**

Table 2. Mean squared error loss on tile adjacent to (to the right of) each trained image. The training image is still used as the loss target, since tiling should produce an identical result. Boldface shows lowest loss for each image. Since FKAN was given a smaller image to train on, if its loss is lowest then the second lowest is also bolded.

Model	checker1	checker2	checker3	tartan1	tartan2	tartan3
RELU	7.943888	1878.48169	48858.6367	17.222811	43.406975	583.343506
Tanh	1.167511	2.070165	0.942918	0.12413	0.170836	0.22338
RBF	**0.986468**	**0.963818**	**0.92833**	0.281618	0.256042	0.227065
Sine	1.413643	1.421796	1.203332	0.172646	0.091287	0.096892
FFN	1.633135	1.555763	1.142922	**0.060415**	**0.040569**	**0.032702**
FKAN	2.287191	**0.916523**	1.512476	**0.029259**	**0.018958**	**0.018785**

Table 3. Mean squared error loss on a tile distant to (10 tiles to the right of and 10 tiles up from) each trained image. The training image is still used as the loss target, since tiling should produce an identical result. Boldface shows lowest loss for each image. Since FKAN was given a smaller image to train on, if its loss is lowest then the second lowest is also bolded.

Model	checker1	checker2	checker3	tartan1	tartan2	tartan3
RELU	80630.2813	4132655.5	16639662	15121.085	25756.9668	20965.582
Tanh	1.095967	1.331983	0.940158	0.383222	0.3042	0.371145
RBF	**0.986468**	**0.963818**	**0.92833**	0.281618	0.256042	0.227065
Sine	1.448355	1.022074	1.264792	0.252373	0.095484	0.090091
FFN	1.56866	2.327875	1.403901	**0.051468**	**0.035516**	**0.031892**
FKAN	1.241188	1.222081	1.208939	**0.049134**	0.045418	0.036044