Improving Adversarial Robustness of Image Classification Through Pretraining on Neural Data

Shenghua Liu
Stanford University
Stanford, CA, United States
sliu24@stanford.edu

Abstract

Even though modern image classification models achieve high accuracy on standard datasets, they are vulnerable to adversarial attacks, typically involving small, human-unperceivable perturbations to the input image. On the other hand, humans are robust to a wide variety of visual perturbations naturally occuring in our interactions with the world, indicating an important gap between human and machine visual intelligence that needs to be bridged on the way to human-level AI. In the field of neuroscience, recent advances in neurotechnology have enabled the simultaneous recording of neural activity from large populations of neurons in the brain. In an attempt to harness the computational strategy used by the brain, we explore the hypothesis that computational models trained on neural data can capture mechanisms underlying adversarial robustness in the brain. In particular, we explore transfer learning methods to improve adversarial robustness of image classification models. We first pretrain a model to predict neural activity from natural movie stimuli, and then finetune the model on standard image classification datasets. We evaluate the adversarial robustness of our model using the RobustBench library and its AutoAttack method. While we do not find positive evidence supporting this hypothesis, we analyze the information contained in successive layers of the neural encoding model and its usefulness in downstream classification.

1. Introduction

Deep neural networks have achieved remarkable success in image classification tasks, often matching or exceeding human performance on benchmark datasets such as ImageNet [6, 9]. However, these models exhibit a critical vulnerability not shared by the human visual system: susceptibility to adversarial attacks. Originally discovered by Szegedy et al. [19], these attacks involve carefully crafted, imperceptible perturbations to input images that cause models to produce dramatically incorrect predictions with high confidence. This vulnerability reveals a fundamental disconnect between human and machine visual processing, suggesting that current neural networks, despite their impressive performance, fail to capture essential characteristics of biological visual systems [13].

Adversarial robustness has become a significant focus in machine learning research, with numerous defense strategies proposed over the past decade. The dominant approach has been adversarial training, which incorporates adversarially perturbed examples into the training process [14]. While effective, this approach incurs substantial computational costs, often requiring 3-10 times more training time than standard training. There have been many recent methods that seek to improve the efficiency and effectiveness of adversarial defenses, leading to sizable advancements of the state-of-the-art (SOTA) [4, 7, 20].

Methods for evaluating adversarial robustness are diverse and lack consensus, but the RobustBench library [3] has emerged as a standard benchmark for evaluating adversarial robustness of image classification models. It provides a suite of white- and black-box adversarial attacks and a leaderboard for comparing model performance across various datasets and attack methods. In particular, for CIFAR-10 [11], they use the standard l_∞ perturbations with $\epsilon_\infty = 8/255$. This means that the perturbation is bounded by 8/255 in the l_∞ norm, which corresponds to a maximum pixel value change of approximately 3.1% of the pixel value range (0-255). Under this benchmark, as of June 2025, the top-performing models achieve raw accuracies of over 90% on CIFAR-10, with adversarial accuracies of over 70%. For ImageNet and $\epsilon_\infty = 4/255$, the top-performing models achieve raw accuracies of over 80% with adversarial accuracies of nearly 60%.

Another line of work seeks to leverage the inductive bias of biological visual systems to improve the robustness of artificial neural networks (ANNs). Ref. [12] uses an additional regularization term to align the learned representa-

tions of an ANN to a model trained to predict neural activity in mice viewing natural images. Ref. [5] uses a biologically constrained linear-nonlinear-Poisson model as a front end to a deep image classification network. Ref. [16, 17, 8] train an ANN with two heads to predict both the neural activity of a biological system and the class label of an image. These approaches all yield sizable improvements in adversarial robustness over raw models. They are not evaluated in a standard way, but as an example, Ref. [16] slightly outperforms the SOTA at the time of publication (2020) on ImageNet-C, which is an adversarial benchmark based on ImageNet but including 75 common visual corruptions [10]. However, it very much remains an open question what the best way to transfer the inductive bias of biological systems to ANNs is.

In this work, we explore a novel approach. Instead of partially incorporating a biologically constrained model into the classification network via additional loss terms, cotraining, or a minimalistic front end, we propose to use a substantial part (or entirety) of a neural encoding model as the front end to a classification network. We hypothesize that pretraining an entire neural encoding model before finetuning the classification network will ensure the neural representations remain as intact as possible, thus preserving the inductive bias of the biological system.

1.1. Problem Statement

Concretely, supposed we have a neural encoding model G pretrained to predict neural activity $\mathbf{s}$ from input visual stimuli $\mathbf{x}$. We want to train a classification backend F that takes as input some intermediate representation of the neural encoding model $\widetilde{G}(\mathbf{x})$ and computes the class scores $\mathbf{c}$. So, the whole network is

$$\mathbf{c} = F(\widetilde{G}(\mathbf{x})). \tag{1}$$

We vary the following hyperparameters and explore what works best:

- The layer in G that we take the feature map from and feed into F.
- What architecture of G to use to best facilitate transfer learning to image classification.
- Whether we use a residual connection from the input to the beginning of G to preserve information.

2. Dataset

2.1. Neural Data

We use a dataset from the Baccus Lab [1] of neural recordings from the mouse visual cortex viewing movies of natural scenes, which contain rich features of ethological relevance that thus may elicit adversarially robust mechanisms. Previous work [15] shows that similar recordings from salamander retinas have been used to successfully train convolutional neural networks (CNNs) that capture a variety of known biological phenomenology in the retina [15]. The experimental setup is shown in Fig. 1, which is adapted from Ref. [2].

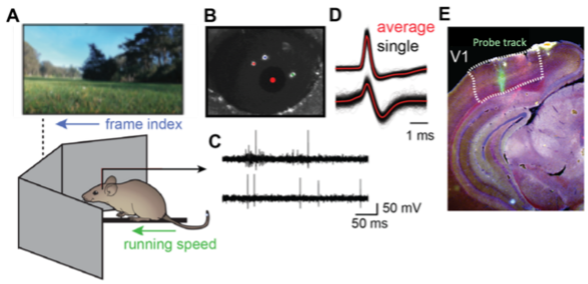

Figure 1: Experimental setup. A. Schematic of experimental setup. B. Eye tracking image showing pupil center and reference LEDs. C. Example spike recordings. D. Example clusters for narrow and wide spike waveforms. E. Brain section with fluorescently labeled probe track. Caption adopted from Ref. [2].

For recordings of the visual cortex, eye tracking was employed so that the stimuli were correctly shifted to account for eye movements. The stimulus is a long (∼hours) natural movie of size 68x102 and the output is the binned spike rate at 50 Hz for 32 cells. Recording was done using NeuroPixel 1, a thin probe with high density electrodes along its shaft that is inserted into the mouse visual cortex after a surgery and a habituation period. The recorded traces were processed through a spike-sorting algorithm to isolate individual neurons. Further quality control criteria were applied, such as selecting cells with clear receptive fields and/or cells with good reliability over repeated stimuli and predictability from simple models. 32 cells were selected after this procedure. In preprocessing, the movie is grayscaled and each sample consists of 25 frames before and 5 frames after the current spike bin. The model is trained to predict the binned spike rate at each time point from its corresponding window. Data collection and preprocessing was done by the Baccus Lab, and we adapt a model pretrained on the cortical dataset to our use case.

2.2. Image Classification Dataset

We use the CIFAR-10 dataset as a proof of concept, which consists of 50000 training and 10000 test images of size 32x32 in color, each labeled as one of 10 classes [11]. We do not use the test images. We separate the training set into 49000 images used to train the network and 1000 used for validation.

3. Technical Approach

3.1. Neural Encoding Model

We use a model pretrained on the cortical dataset described above as our neural encoding model that we hypothesize contains robust visual computation strategies that we wish to transfer to downstream image classification. The model is a CNN with 6 layers, each with 24 channels and a kernel size of 7 with no padding. After the convolutional layers, a fully connected layer with softplus nonlinearity is used to read out the predicted spike rates for each of the 32 neurons. The model was trained with a thorough hyperparameter search by Baccus Lab members, and the final model achieves 0.3 average correlation with the true binned spike rates. Because of eye movements, there are no true repeated trials, as the animal's retina receives projections of different parts of the same stimulus, so it is challenging to compare this correlation to a noise ceiling. However, this number is in line with other SOTA models of the visual cortex (from discussion). Model predictions for an example neuron are shown in Fig. 2.

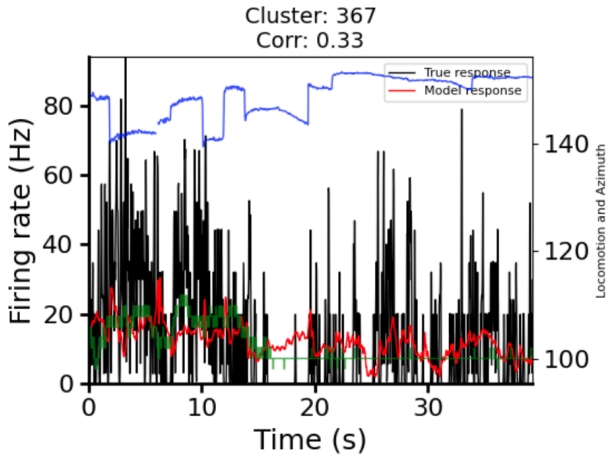

Figure 2: Predictions from the neural encoding model for an example neuron. The blue and green lines are the azimuth of eye position and running speed, respectively. Predictions for this neuron has 0.33 correlation with the ground truth firing rates.

3.2. Image Classification Model

We use a VGGNet-inspired [18] CNN as the image classification backend. The general approach is that we freeze the pretrained neural encoding model and just train the backend, on CIFAR-10. However, we do attempt modifications to this pipeline to get the optimal results. In particular, we change which layer of the neural encoding model we read out from and whether we use a residual connection from the input to the beginning of the image classification backend. As we will see in the results, not using the residual connection hurts the clean accuracy as useful information about image class is lost in neural encoding. Therefore, using a residual connection gives the image classification backend the full information in the input while also preserving access to the potentially robust representations learned by the neural encoding model. The architecture of the image classification backend is as follows:

```
Input
->BN(32)->ReLU
->Conv3x3(32,32)->BN(32)->ReLU
->MaxPool2x2->Conv3x3(32,64)->BN(64)
->ReLU->Conv3x3(64,64)->BN(64)->ReLU
->MaxPool2x2->Conv3x3(64,128)->BN(128)
->ReLU->Conv3x3(128,128)->BN(128)->ReLU
->MaxPool2x2->Flatten
->Linear(2048,512)->ReLU
->Linear(512,10)
```

Because the modality of the video stimulus (grayscale time series) and CIFAR-10 images (colored images) are different, we need to add transition layers to make the neural encoding model compatible with the image classification backend:

- The input size of the neural encoding model is 30 frames of size 68x102, while the input size of CIFAR-10 is 3 channels of size 32x32. In order to feed CIFAR-10 images to the neural encoding front end, we grayscale the input, pad, and resize it to 68x102, and then expand its channel dimension to 30 to mimic a video consisting of 30 frames of the same image.

- The activation size of the neural encoding model just before the linear readout to the recorded cells is 24 channels of size 32x66, and the activation size of previous layers is larger. For uniformity, we pad and resize the activation to 32x32. If we are using the residual connection, we concatenate the input image to this activation in the channel dimension. Then, we use a transition conv layer to convert 24 (or 27 with residual connection) channels into 32 to feed into the backend. We have computed the mean and standard deviation of the natural movie stimuli and found that they are very close to those of CIFAR-10, so we do not need to do any additional normalization.

3.3. Baseline

We refer to the leaderboard provided by RobustBench [3] for the SOTA of adversarial robustness. For the baseline, we use a vanilla CNN (without the front end) with the same architecture as the image classification backend except an ad-

ditional conv layer at the beginning to take the 3 color channels to 32 channels. We will use the AutoAttack method offered by RobustBench to evaluate adversarial robustness.

Furthermore, an interesting question is how much information about image classification is contained in the neural representation in the first place. Therefore, as another baseline, we replace the full backend with a linear readout layer and repeat the experiments where we attach this readout layer to each of the hidden layer of the neural encoding model before training on CIFAR-10.

4. Results

4.1. Model Training

We train all models with the following hyperparameters: Batch size of 64, 10 epochs, learning rate of 10^{-3}, Adam optimizer, cross entropy loss, learning rate scheduler which halves the learning rate after 5 epochs of non-improvement, and weight decay of 10^{-4}. We varied the learning rate and weight decay to find their optimal values, but we did not perform a full grid search on the hyperparameters due to time and compute constraints. While 10 epochs may not lead to full convergence, empirically all models converge to a satisfactory degree. As a reminder, we train three types of models in addition to the CNN baseline without the neural encoding front end, where for each type we attach the backend to a different hidden layer of the neural encoding model:

- The backend is a simple linear readout without a residual connection from the input to the backend.
- The backend is a CNN without a residual connection from the input to the backend.
- The backend is a CNN with a residual connection from the input to the backend.

Figures 3 to 6 show the training loss and validation accuracy for models with CNN backends and models with linear backends. We see that all training runs converge reasonably well, although the validation accuracy of models with a linear backend tend to fluctuate a significant amount.

4.2. Adversarial Robustness

Fig. 7 compares the adversarial robustness of the SOTA robust ResNet-18 model, our baseline vanilla CNN model, and our most robust hybrid model as evaluated by RobustBench. The choice of the SOTA model is somewhat arbitrary, since we do not see comparable robustness in our models to warrant the use of a similarly complex model for a rigorous comparison. Unfortunately, our model does not show signs of improvement of robustness over the baseline CNN model. All but one of our hybrid models (shown here) has exactly zero adversarial robustness as evaluated

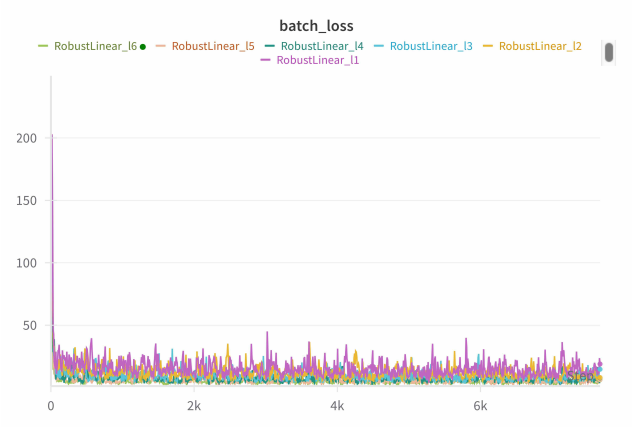

Figure 3: The training loss of all models with a linear backend.

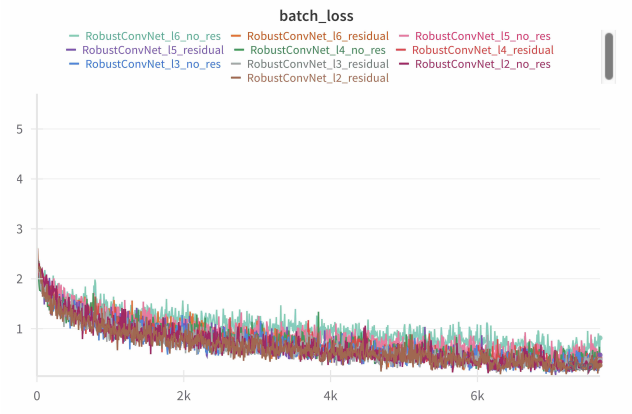

Figure 4: The training loss of all models with a CNN backend.

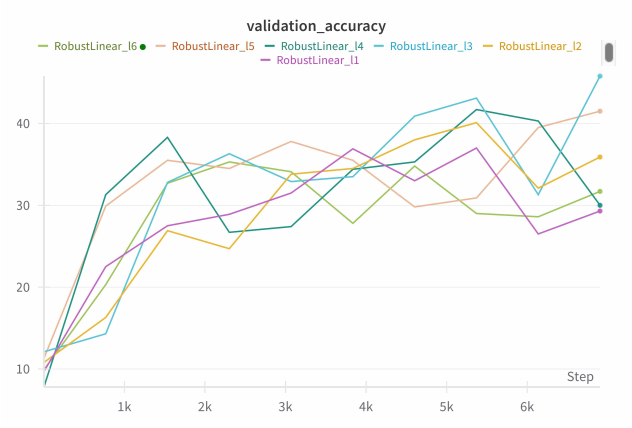

Figure 5: The validation accuracy of all models with a linear backend.

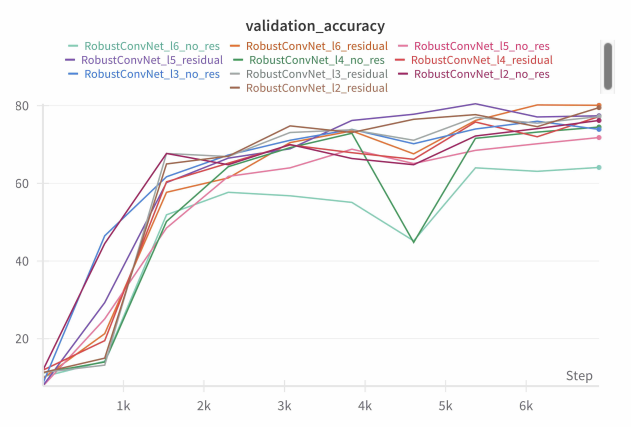

Figure 6: The validation accuracy of all models with a CNN backend.

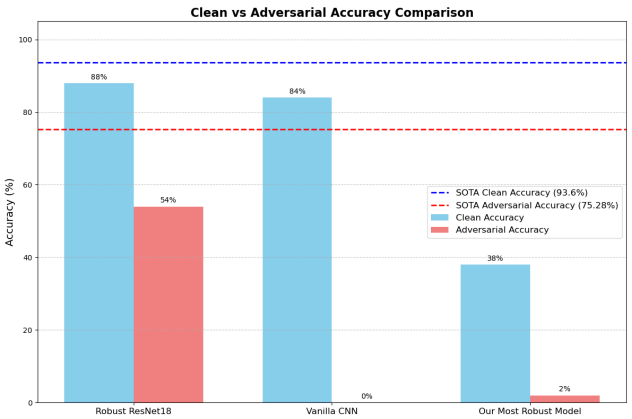

Figure 7: Comparison between the adversarial robustness of the SOTA robust ResNet-18 model, our baseline vanilla CNN model, and our most robust hybrid model.

by RobuseBench. This best model turns out to be a linear readout attached to the second layer of the neural encoding model, which is surprising because it has one of the lowest clean accuracies. The failure of our models could be due to a few reasons. First, because our CNN backend is fairly small, it may not have learned clean decision boundaries, which may make it more susceptible to adversarial attacks. Second, the grayscale nature of the neural encoding model as well as the two padding and resizing steps make the model lose a lot of information about the input in the case of models without a residual connection. In models with a residual connection, the model may not be incentivized to utilize the neural representation as opposed to the raw input sent through the residual. Third, the neural encoding model may simply not contain features that are useful for robust image classification. This is reasonable, since we only have 32 cells from the visual cortex to begin with. It is reasonable to suspect that robust computation requires some form of redundancy, which may require population coding by large numbers of neurons working together. Furthermore, it is unclear that our neural encoding model captures the full information even for just those 32 cells. The correlation, as stated before, is only 0.3, which means other variables such as behavioral state may explain a similar amount of variance as the visual computation itself. Therefore, the model likely has not learned fine-grained visual computational strategies that may be required for robust processing. It could even be that higher brain areas initiate top-down processing that maintains stable representations or coordinates active sensing which may make vision robust.

4.3. Decoding Performance

Even though we do not show positive results regarding improving adversarial robustness by trasnfer learning from neural encoding models, we show results pertaining to the useful information contained in successive layers of the neural encoding model.

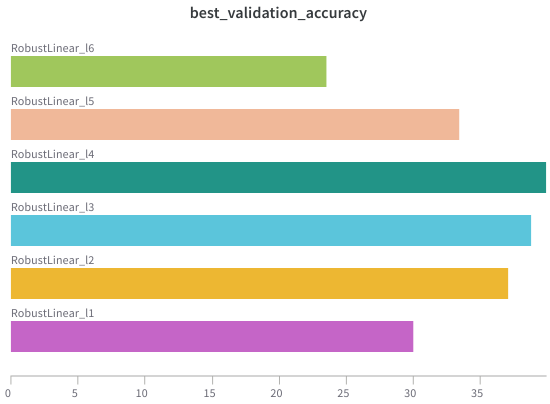

Figure 8: Clean accuracy as a function of which layer to decode from the neural encoding model, in the linear backend case.

In Figure 8, we observe an interesting phenomenon that the clean accuracy first increases then decreases as you decode deeper and deeper layers of the neural encoding model. The accuracy peaks in the middle, for layers 3 and 4 roughly. We suspect that this is because earlier representations have not gone through sufficient processing to yield a feature map with advantageous structure for linear decoding of the target class. The deepest layers, while having gone through the most processing, are highly optimized for decoding spike rates, which is a very different objective from image classification. Even though in principle all information about the image class is contained in the population of neurons in the visual cortex, we have only used 32 neurons, and thus they may contain very little information about the

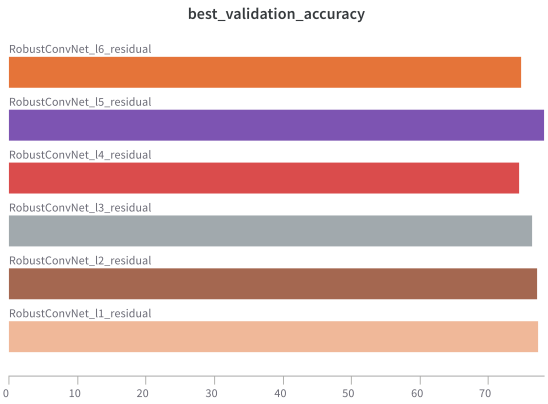

Figure 9: Clean accuracy as a function of which layer to decode from the neural encoding model, in the CNN backend with residual case.

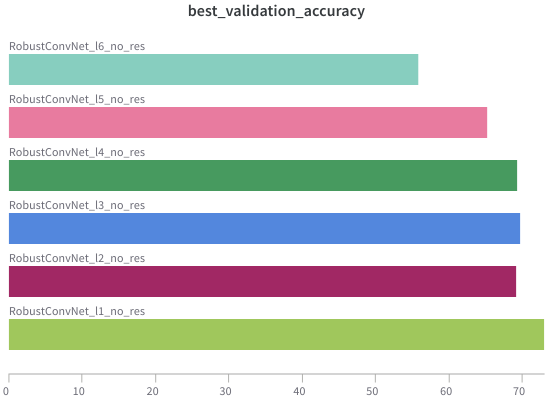

Figure 10: Clean accuracy as a function of which layer to decode from the neural encoding model, in the CNN backend without residual case.

image class.

In Figure 9, we see that the clean accuracy does not vary much as a function of decoding depth, and they are all consistent with the vanilla CNN's accuracy (84%). We think this is because the residual connection allows the CNN backend to utilize all information in the input. But this observation does imply that the neural representation fed into the backend does not provide additional benefits to image classification. It also does not improve robustness, as explained in the previous section. We think that this is again due to the neural encoding model not having learned fine-grained robust visual computation, either due to the limited number of cells we have or the limitation of the encoding model.

In Figure 10, we observe that the accuracy decreases as a function of decoding depth. Comparing this figure to Figure 8, we draw the interpretation that the more sophisticated CNN backend is able to take up the further nonlinear processing needed to go from a premature representation in an early layer to the final classification. This distinction enables the model to still achieve relatively high accuracy when decoding from layer 1 even though the image is grayscaled and therefore color information is lost, in contrast to the residual and vanilla CNN cases. This shows that color information only accounts for less than 10% of classification accuracy. Furthermore, we see that as depth increases, accuracy drops consistently, yet still significantly higher than the linear decoder. This corroborates the earlier interpretation that the deepest layers are more optimized for decoding spike rates, and therefore useful information about image class is lost.

5. Conclusion

In this work, we explored the hypothesis that transfer learning from neural encoding models can improve adversarial robustness of image classification models. We pretrained a neural encoding model on neural data from the mouse visual cortex viewing natural movie stimuli, and then finetuned a classification backend on CIFAR-10. We evaluated the adversarial robustness of our model using the RobustBench library and its AutoAttack method. Unfortunately, we did not find any positive results, but we analyzed the classification performance when decoding from different layers of the neural encoding model and using different backends and drew conclusions on the information contained in the neural representation.

We explored three types of models: a linear readout, a CNN backend without a residual connection from the input to the backend, and a CNN backend with a residual connection. We found that only the linear readout achieves nonzero adversarial robustness, but it has the lowest clean accuracy. The model with a CNN backend with or without a residual connection did not improve adversarial robustness over a vanilla CNN having essentially the same architecture as the backend. However, we found that the clean accuracy of the linear readout peaks at the middle layers of the neural encoding model, while the CNN backend with a residual connection achieves a consistent accuracy across all layers. The CNN backend without a residual connection sees a steady decrease of accuracy as a function of decoding depth. This shows that early layers of the neural encoding model has undergone premature processing, yet still contains useful information about the image class, which was able to be utilized by the more complex, nonlinear CNN backend in contrast to the linear readout. The deepest layers of the neural encoding model, while having undergone the most processing, are optimized for decoding spike rates and thus do not contain as much information about the image class. Furthermore, comparing the CNN backend without a resid-

ual connection to the one with a residual connection, we see that the residual connection allows the backend to utilize all information in the input, which crucially includes the color information that is lost in the neural encoding model because it is grayscaled. This difference accounts for roughly 10% of the classification accuracy.

While we do not find positive results regarding adversarial robustness, there are future steps we can take to improve the technique. First, we can try to use a larger neural encoding model pretrained on more cells, which may capture more robust visual computation strategies. Second, we can try to use a larger image classification backend, which may be able to learn more complex decision boundaries and thus be more robust. Third, we can try to use a different dataset for pretraining the neural encoding model, such as recordings from the retina, which are known to be more reliable across trials and contain more cells, and thus may yield a better pretrained model. Finally, we can try to combine our method with many of the existing techniques, whether along the lines of adversarial training or other brain-inspired techniques, to see if we can achieve a better result. We hope that this work will inspire future research in the area of adversarial robustness and transfer learning from neural data.

6. Contributions and Acknowledgements

Shenghua Liu conceived the project, developed the code, ran the experiments, and wrote the paper. At the time of writing, he worked in the Baccus Lab in the Department of Neurobiology at Stanford University and used its computing cluster to run experiments and implement the code. He is grateful for the support from the lab, including the neural data provided by lab and the neural encoding model pretrained by Baccus Lab members as a collective effort.

References

[1] Baccus Lab.

[2] S. Baccus. Neural processing of natural scenes in the mouse visual cortex. Technical report.

[3] F. Croce, M. Andriushchenko, V. Sehwag, E. Debenedetti, N. Flammarion, M. Chiang, P. Mittal, and M. Hein. RobustBench: A standardized adversarial robustness benchmark, Oct. 2021.

[4] J. Cui, Z. Tian, Z. Zhong, X. Qi, B. Yu, and H. Zhang. Decoupled Kullback-Leibler Divergence Loss, Oct. 2024.

[5] J. Dapello, T. Marques, M. Schrimpf, F. Geiger, D. Cox, and J. J. DiCarlo. Simulating a Primary Visual Cortex at the Front of CNNs Improves Robustness to Image Perturbations. In *Advances in Neural Information Processing Systems*, volume 33, pages 13073–13087. Curran Associates, Inc., 2020.

[6] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, June 2009.

[7] S. Fort and B. Lakshminarayanan. Ensemble everything everywhere: Multi-scale aggregation for adversarial robustness, Aug. 2024.

[8] M. Guo, B. Choksi, S. Sadiya, A. T. Gifford, M. G. Vilas, R. M. Cichy, and G. Roig. Limited but consistent gains in adversarial robustness by co-training object recognition models with human EEG. 2024.

[9] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition, Dec. 2015.

[10] D. Hendrycks and T. Dietterich. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations. In *International Conference on Learning Representations*, Sept. 2018.

[11] A. Krizhevsky. Learning Multiple Layers of Features from Tiny Images. 2009.

[12] Z. Li, W. Brendel, E. Y. Walker, E. Cobos, T. Muhammad, J. Reimer, M. Bethge, F. H. Sinz, X. Pitkow, and A. S. Tolias. Learning From Brains How to Regularize Machines, Nov. 2019.

[13] H. Machiraju, O.-H. Choung, P. Frossard, and M. H. Herzog. Bio-inspired Robustness: A Review, Mar. 2021.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu. Towards Deep Learning Models Resistant to Adversarial Attacks, Sept. 2019.

[15] N. Maheswaranathan, L. T. McIntosh, H. Tanaka, S. Grant, D. B. Kastner, J. B. Melander, A. Nayebi, L. E. Brezovec, J. H. Wang, S. Ganguli, and S. A. Baccus. Interpreting the retinal neural code for natural scenes: From computations to neurons. *Neuron*, 111(17):2742–2755.e4, Sept. 2023.

[16] S. Safarani, A. Nix, K. F. Willeke, S. A. Cadena, K. Restivo, G. Denfield, A. S. Tolias, and F. H. Sinz. Towards robust vision by multi-task learning on monkey visual cortex. In *Advances in Neural Information Processing Systems*, Nov. 2021.

[17] Z. Shao, L. Ma, B. Li, and D. M. Beck. Leveraging the Human Ventral Visual Stream to Improve Neural Network Robustness, May 2024.

[18] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, Apr. 2015.

[19] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. Goodfellow, and R. Fergus. Intriguing properties of neural networks, Feb. 2014.

[20] Z. Wang, T. Pang, C. Du, M. Lin, W. Liu, and S. Yan. Better Diffusion Models Further Improve Adversarial Training, June 2023.