Random Embeddings for Robust Deep Learning (Research)

Nathan Blair
3rd Year Undergraduate
3031828892
Email: nblair@berkeley.edu

Adarsh Karnati
3rd Year Undergraduate
3031839773
Email: akarnati@berkeley.edu

Abstract— **We propose the Random Feature Embedding Layer (RFEL), a novel regularization technique for deep neural networks. RFEL is inspired by Rahimi and Rechts Random Fourier Features for kernel machines. Our layer increases model robustness to common noise patterns such as Gaussian additive noise. RFEL performs comparably to or better than the popular regularization techniques weight decay and dropout. RFELs are a promising regularization technique, but more work is needed to determine their viability on large-scale models.**

I. INTRODUCTION

Machine learning models can be fragile in the face of even slight distributional shifts between the training and test sets [1]. However, the human visual system is extremely robust to moderate corruptions in visual data; See Figure 1. Notably, the human visual system is robust to weather changes (e.g. snow, fog, rain), pixelation, compression artifacts, additive noise, lighting, and more complex style transfers [1]. These distributional shifts are common in practice, especially when the test set is pulled from a different data source from the training set.

When machine learning models fail to generalize to common visual corruptions, the results can be life threatening. Consider the case of a self driving car that must be able to operate in all weather conditions, snow, rain, sleet, fog, and more, but is primarily trained in a sunny city. The failure of self driving cars in bad weather could easily lead to the death of a passenger.

One highly cited example of machine learning models performing poorly in the face of distribution shift is Google Flu Trends. Launched in 2008, Google Flu Trends [2] was a project aimed at predicting how many people have the flu based on web searches. Initially the program made accurate and quick predictions, even faster than the Center for Diseases Control by up to two weeks. However, by 2013 the model produced completely inaccurate predictions, the culprit being an unaccounted change in how users were searching [3]. The learned model was unprepared to account for distributional changes in the input. In general, it is not possible for a learned model to adapt to any distributional change in the test data. However, it is possible to make models more robust to specific types of distributional shifts in the data, such as corruption and noise.

Corruption and noise robustness is very closely related to the problem of overfitting. Models that overfit, are those that have a very high train accuracy, but low test accuracy. This suggests that the model does not generalize well and therefore cannot be used to make accurate predictions on unseen data. From the perspective of noise robustness, models that overfit do not capture the underlying patterns in the data, and can therefore be susceptible to small perturbations. It is often easy for Deep Neural Networks (DNNs) to fit noise present in the training data, especially when the capacity of the network is high [4].

There are numerous ways to increase the robustness of machine learning models to overfitting and various distributional shifts [4], [5], [6], [7], [1]. Notably, weight decay and dropout are popular because of their easy implementation and high effectiveness for model regularization.

We introduce the Random Feature Embedding Layer (RFEL) to tackle the problems of overfitting and poor generalization to distribution shift and visual corruptions. The RFEL is based on a Random Fourier Feature embedding, proposed by Rahimi and Recht as a kernel method for increasing model accuracy and reducing model size [8]. In the setting of kernel methods, we note that Random Fourier Feature embeddings not only increase model accuracy, but also increase model noise robustness. We show that the same intuition can be applied in the neural network setting with RFELs.

Our contributions are as follows:

- We propose the Random Feature Embedding Layer (RFEL), a novel regularization technique for deep neural networks.
- We show that RFEL increases model robustness to common noise patterns.
- We show that RFEL performs comparably to or better than the popular regularization techniques weight decay and dropout in terms of increasing corruption robustness and decreasing overfitting.

II. RELATED WORK

A. *Overfitting and Regularization in DNNs*

Overfitting in neural networks has been a known issue since neural networks became popular with linear perceptron methods [9]. Likewise, there has been much work studying the causes of overfitting and ways to prevent it. Caruana et al. show that early stopping prevents overfitting in neural network models trained with gradient descent [4]. Zur et al. show that injecting noise in the training data can be used to prevent overfitting [5]. Intuitively, injecting noise in the training data may teach the model not to consider noise when making predictions. Krogh et al. showed that adding a weight decay when training a neural network acts like L2-Regularization and helps models to generalize better to test data [6]. Srivastava et al. proposed adding a dropout layer to reduce overfitting [7]. The dropout layer randomly zeros some of the activations from the previous layer. This prevents neurons from co-adapting too much [7]. We compare our random feature embedding layer to weight decay and dropout, which are widely popular.

B. *Model Noise Robustness*

While the human visual system is robust to various visual corruptions such as blur, additive noise, and weather related noise, popular computer vision systems are, in general, not. Vasiljevic et al. show that convolutional models make unreliable predictions when presented with blurred images [10]. While noise robustness is not widely studied in the context of vision, it is a popular topic for speech and audio papers, where they defend against various additive noises. This is because audio noise is far more common in practice, often coming from ambient noise like wind and background conversations [11], [1]. Hendrycks et al. released a set of benchmarks, methods and datasets for evaluating model performance against a set of common noise corruptions [1]. We use their dataset for evaluating our models.

C. *Model Adversarial Robustness*

Images can be corrupted by small amounts of noise such that the changes are indistinguishable to a human, but changes the prediction of a classifier. This has become a commonly studied issue in robustness literature, especially in the context of model robustness of image classifiers. In essence, measuring a network's adversarial robustness is measuring its performance in the worst possible case. While many methods have been proposed to defend against adversarial attacks against neural network models, new methods have been created to get around those defenses [12], [13], [14]. This has resulted in an arms race in adversarial attacks and defenses with no clear winner. We do not consider adversarial attacks in this paper. However, there is evidence to show that increasing corruption noise robustness also increases adversarial robustness [15]. So, by showing an increase in noise robustness, we also hypothesize that the inclusion of RFEL will increase adversarial robustness.

D. *Random Feature Embeddings*

Random Feature Embeddings were introduced by Ben Recht and Ali Rahimi in their 2007 paper Random Features for Large-Scale Kernel Machines [8]. They propose Random Features as a method to speed up the computation of kernel machines by randomly projecting the data in a way such that inner products in the projected space roughly approximates a user-specified kernel. We apply this method directly in the creation of our Random Feature Embedding Layer.

Random Feature Embeddings improve the speed of linear models and may also improve the accuracy. Since generating these embeddings is quick, projecting the data into a very high dimensional space is tractable, meaning that the data might be able to be made linearly separable.

III. Methods

A. Problem Formulation

We have a set of images, $\mathcal{X}$, and corresponding labels, $\mathcal{Y}$, and seek to learn a classifier $f : \mathcal{X} \to \mathcal{Y}$. $\mathcal{X} \subset \mathbb{R}^{C\times H\times W}$, with C the number of image channels, H the image height and W the images width. We partition $\mathcal{X}$ and $\mathcal{Y}$ into two random subsets, denoted as $(\mathcal{X}_{train}, \mathcal{Y}_{train})$ and $(\mathcal{X}_{test}, \mathcal{Y}_{test})$, where $|\mathcal{X}_{train}| = |\mathcal{Y}_{train}| = n$ and $|\mathcal{X}_{test}| = |\mathcal{Y}_{test}| = k$. We train a classifier $\hat{f}$ using empirical risk minimization with the commonly used Cross Entropy Loss (CEL) function:

$$\hat{f} = \arg\min_{\theta} \frac{1}{n} \sum_{(x_i,y_i)} CEL(f_\theta(x_i), y_i) \qquad (1)$$

where $(x_i, y_i) \in (\mathcal{X}_{train}, \mathcal{Y}_{train})$.

We define the *robustness* of $\hat{f}$ to a *noise model*, $\psi : \mathbb{R}^{C\times H\times W} \to \mathbb{R}^{C\times H\times W}$ as the average accuracy of our classifier on the test set with images corrupted by the noise model:

$$\frac{1}{k} \sum_{(x_j,y_j)} \mathbb{I}(\hat{f}(\psi(x_j)) = y_j) \qquad (2)$$

where $(x_j, y_j) \in (\mathcal{X}_{test}, \mathcal{Y}_{test})$.

B. Random Fourier Feature and Embedding Layer

Define *Random Fourier Features* as the feature mapping:

$$z(x) = \sqrt{\frac{2}{D}} cos(\mathbf{W}x + b) \qquad (3)$$

$$x \in \mathbb{R}^d, z(x) \in \mathbb{R}^D, \mathbf{W} \in \mathbb{R}^{D\times d}, b \in \mathbb{R}^D$$
$$\mathbf{W}_{ij} \sim \mathcal{N}(0, \sigma^2), b_i \sim Uni(0, 2\pi)$$

with d the raw feature dimension and D the embedding dimension [8].

We define a *Random Feature Embedding Layer* (RFEL) as a neural network layer that simply embeds the activations of the previous layer. RFEL does not contain any trainable parameters. It randomly projects the features and then applies a nonlinear activation function. Note that RFEL must be placed after convolutional layers, if they are in the network, since RFEL scrambles the relative spatial content of the previous layers' activations.

C. Noise Models

The noise models we consider in this paper are based on [1]. These models have a severity parameter ranging from one to five that describe the intensity of the noise applied to images. These noise models are algorithmically generated and represent the most common corruptions present in image data. We use the Gaussian noise, shot noise, and Gaussian blur, see Figure 1, but only discuss Additive Gaussian Noise in detail throughout the paper to be concise.

Fig. 1: Gaussian Noise, Shot Noise and Gaussian Blur

D. Datasets

In this paper we evaluate RFEL on the Modified Natural Institute of Standars and Technology (MNIST), Fashion MNIST, and the Canadian Institute for Advanced Research (CIFAR-10) datasets [16], [17], [18]. MNIST is a 10-class dataset of handwritten digits (0 through 9) and is often considered the first widely used benchmark on classification tasks. Each image is a single channel, 28×28 array of integers from 0 to 255. While MNIST is perhaps the most well known dataset for classification, in the past few years, advances in deep learning have made the digit classification problem too easy. For example, state of the art convolutional neural networks achieve a top-5 error rate of 0.21%.

This lack of difficulty led to the development of Fashion MNIST. Fashion MNIST is a 10-class dataset of gray scale clothing images. The images have the same size and number of channels as the regular MNIST dataset, but classification on Fashion MNIST has been found to be empirically more difficult. This increase in difficulty has made Fashion MNIST the new basic benchmark in classification for deep models.

CIFAR-10 is a 10-class dataset of colored images (3 channels), with labeled objects. Each image in CIFAR-10 is 32×32, slightly larger than both MNIST and Fashion MNISt images. CIFAR-10 is another benchmark dataset, but is considered much harder to train on than Fashion-MNIST and MNIST. The higher dimensionality of CIFAR-10 (3×32^2) for a flattened image) and low resolution of the images are considered the main reasons for the difficulty of the dataset.

MNIST, Fashion MNIST and CIFAR-10 will provide a good spread of dataset difficulty as we analyze RFEL in deep models. We will be able to analyze if RFEL prevents overfitting in MNIST for example or if it hinders training in CIFAR-10. The reason we only consider these datasets in the following experiments rather than a much harder dataset, such as ImageNet, is that training was done on a single CPU as no GPUs were available.

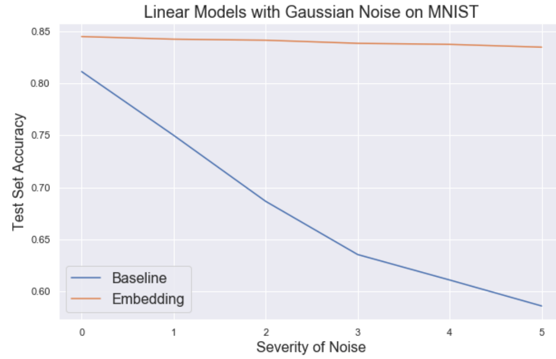

Fig. 2

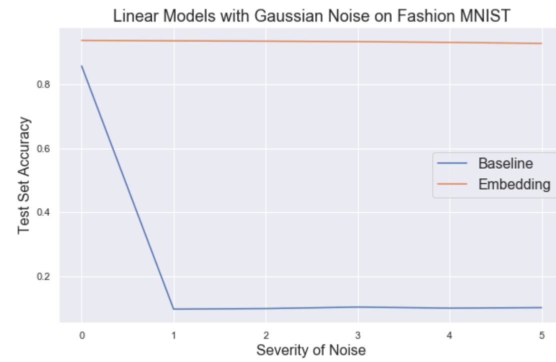

Fig. 3

IV. RESULTS

A. Linear Classifier

As a sanity check for our method, we first construct two linear classifiers trained on the original MNIST, Fashion MNIST and CIFAR-10 datasets. We maintain the intuition that linear models serve as the first step to understanding the effect of new architectures, and often generalize to deeper models. Our baseline model uses the raw data, while our embedded model uses Random Fourier features. Both models were trained on clean train set data, but evaluated on noisy test set data. The test data was corrupted with all the noise models described in the Methods section and evaluated. We compared the baseline model to embedding models with embedding dimension 784. We kept the embedding dimension at 784, the same number of features as the raw data, to observe if the embedding provides a more informative feature space than pixel values.

It is clear that the embedding classifier outperforms the vanilla least squares estimate on MNIST and Fashion MNIST datasets, with Gaussian Noise on the test set. We note that there is virtually no drop in accuracy for the embedding model for images corrupted by Gaussian noise, compared to a somewhat linearly decreasing accuracy for the baseline model. This suggests that embedding the data does provide a better way to represent the raw data than pixel values, and that the addition of noise does not hinder the underlying structure represented by the embedding.

The embedding model also overfits less than the vanilla linear model, see Figures 2, 3. This aligns with our intuition that random embeddings act as a regularizer. It is also worth noting that in the linear setting, adding a random feature embedding improves model accuracy for large enough embedding dimension. This makes sense, because we are approximating a gaussian kernel when we embed, meaning we are able to extract nonlinear features from the input data. Note that we do not show the results from the CIFAR-10 dataset, since the linear model does not have enough capacity to provide meaningful results on this dataset.

B. Fully Connected Network

With the linear model producing promising results, we considered a random embedding layer in a fully connected neural network. The network we use is a simple single hidden layer with 512 hidden units and a ReLU activation. Our baseline model accepts a flattened image while the embedding model first embeds the image and then sends the embedded features into the rest of the network. Both models are trained with RMSprop Gradient Descent. The networks are trained on the clean training data from MNIST, Fashion MNIST and CIFAR-10 and evaluated on the test data from each of these datasets. All neural network models in the later parts as well as this section are trained until there is less than a 1% improvement in training accuracy between epochs.

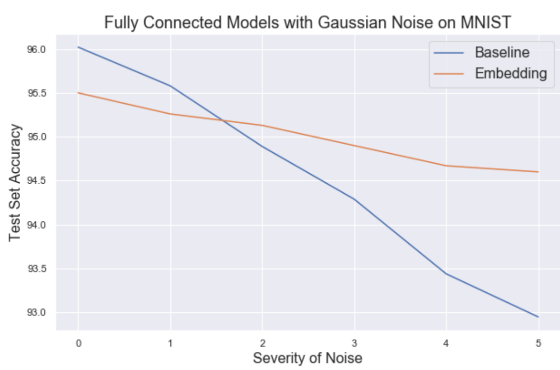

Fig. 4

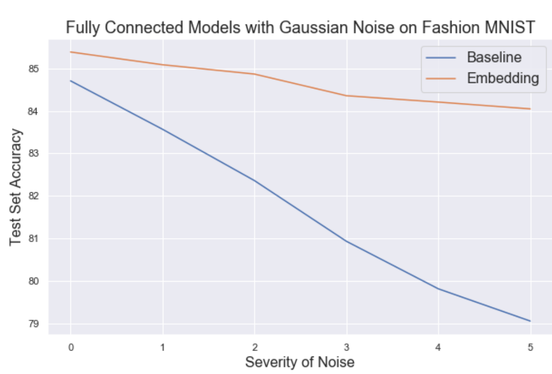

Fig. 5

Interestingly, for Gaussian noise on the MNIST dataset, the baseline model has a slightly higher clean test accuracy than the embedding model, which is different from the results in the Linear Model experiments. This might be explained by the fact that RFEL regularizes the network and for small model architectures, this can have a negative effect for test accuracy.

However, The embedding model was also more noise robust than the baseline model. This aligns with our proposition that the RFEL increases noise robustness.

The fully connected embedding models are also less prone to overfitting than the fully connected baseline models. We would expect a model that overfits to also have low noise robustness, because overfitting can be seen as fitting to the noise in the training data. Thus, the fact that the embedding model overfits less is a sign that it is learning a more robust representation of the data. Note that we do not show the results from the CIFAR-10 dataset, since the fully connected models do not have enough capacity to train properly and provide any meaningful results, similar to the linear model.

C. Convolutional Networks

The baseline convolutional network we used for our experiments consists of two convolutional blocks and two linear layers each of 512 units. Each convolutional block is composed of a 5×5 kernel outputting 32 channels, a stride 2, 2×2 MaxPool and a ReLU activation. A ReLU activation was placed inbetween each linear layer. In our embedded convolutional model, the architecture is exactly the same as in the baseline model, but with an RFEL of embed dimension 512 after the entirety of the convolutional layers and before any linear layers. Below are the comparisons of these two models on all datasets with gaussian noise added to the test set.

In Figure 6, we see that the convolutional model with embedding performs far better than the baseline convolutional model on MNIST. It is interesting to note that despite the convolutional layers processing the noisy image before the embedding layer, RFEL still helps the model reject noise. While the embedding convolutional model outperforms the baseline model, it is important to note that the baseline model only drops around

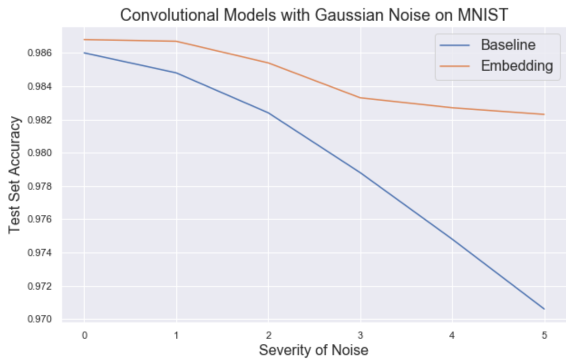

Fig. 6

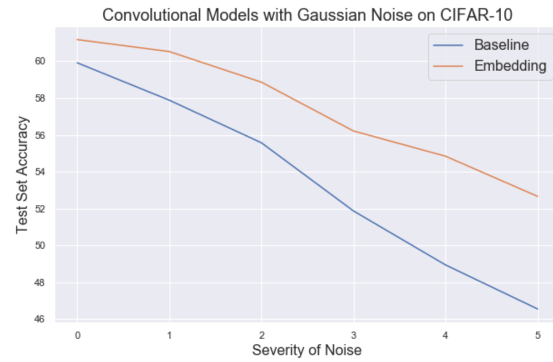

Fig. 8

2% in accuracy and that both models achieve very high performance on MNIST. This is most likely due to simplistic nature of the MNIST dataset and in our analysis of RFEL, we should focus more on Fashion MNIST and CIFAR-10.

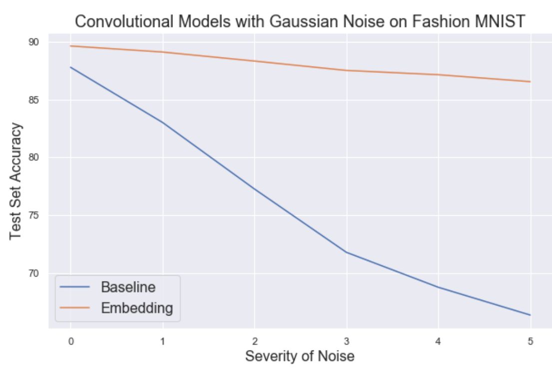

Fig. 7

The same trend we see with the MNIST plot is also present in Fashion MNIST analysis. We note that both baseline and embedding models trained on Fashion MNIST have a clean test accuracy far lower than that of the models trained on MNIST. However, the baseline model trained on MNIST has a far smaller drop in accuracy in comparison to the baseline model trained on Fashion MNIST. This suggests that the baseline model trained on Fashion MNIST overfit to the training data and did not fully capture the underlying structure of the data.

The first observation we see with the convolutional models trained on CIFAR-10 is that both models have a mediocre clean test accuracy. This is most likely caused by the capacity of both networks not being large enough to adequately train on CIFAR-10. Additionally we note that the drop in accuracy for both models is significant and that the embedding model does not hold its accuracy as the networks trained on the other datasets do. However, it is clear that over all datasets, a convolutional network with an RFEL can provide some robustness benefits in comparison to a vanilla model.

D. *ResNet6 Networks*

The next model we experimented with is the popular ResNet architecture [19]. ResNet is a convolutional model that adds the activations of the previous layer to those of the next layer. Additionally, ResNet is fully convolutional except for the last layer which outputs the logits. This network layout has been showed to greatly reduce overfitting and produce state of the art results on several benchmarks.

Our baseline model is a reduced version of ResNet in both width and depth. Specifically, the baseline network has 6 layers of resnet blocks and 64 convolution channels per block. The embedding model we chose was exactly the same as the the baseline model, except for one RFEL after all convolutional layers, right before the final logits, with a embedding dimension of 1000.

We see that ResNet6 on MNIST attains a near perfect clean test accuracy with both the baseline and embedding networks. One point of interest is that the relative drop in accuracy over the different severity levels appears to be the same for both models. Ultimately, it is difficult to determine how well RFEL affects noise robustness on ResNet6

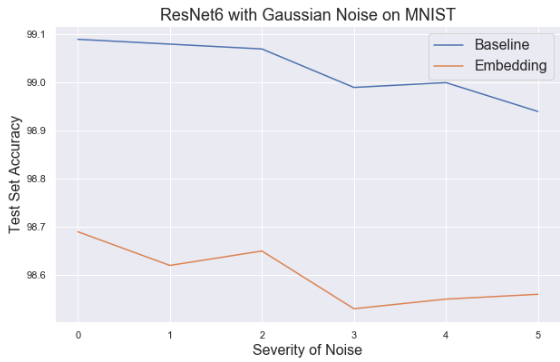

Fig. 9

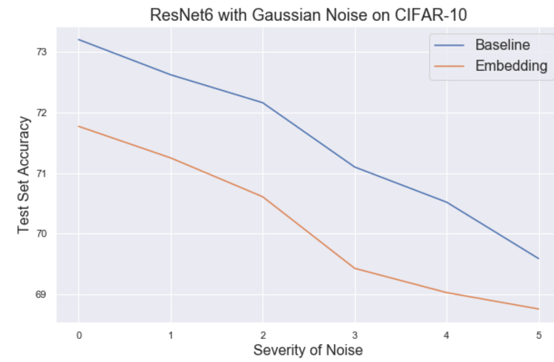

Fig. 11

when the dataset is MNIST, as these models are powerful enough to fully capture the dataset information.

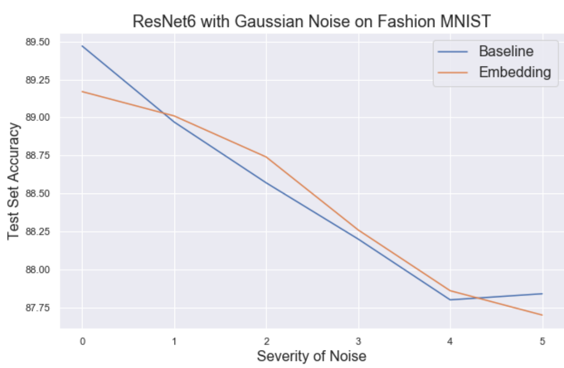

Fig. 10

In Figure 10 we see that the both the embedding and baseline ResNet6 models achieve roughly the same clean test set accuracy and also have around the same drop in accuracy for different severities of Gaussian noise.

From the experiments with ResNet6 trained on CIFAR-10, we conclude that RFEL does not have significant noise robustness in comparison to a regular ResNet6 model. This can be seen on all datasets and all severity levels. This suggests that the ResNet architecture itself prevents against image noise, perhaps due to the batch normalization layers or the additive identity component unique to ResNet models.

E. Convolutional Network with Dropout

Dropout is a technique first proposed in 2014 as a regularization method for neural networks [7]. During training, a network with dropout zeros the activation of a nueron with some probability p. We compare a convolutional network with RFEL to networks with dropout at $p = 0$, $p = 0.25$, $p = 0.5$, $p = 0.75$. The embedding model we use is exactly the same as in the Convolutional Network section and the dropout occurs right after the convolutional layers in the baseline model.

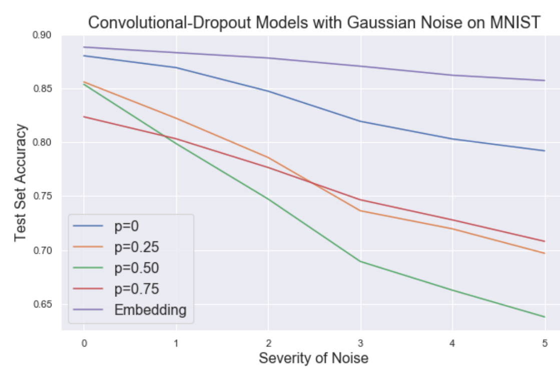

Fig. 12

The embedding network appears to have a higher noise robustness in comparison to the other dropout networks on MNIST. On Fashion MNIST, the difference between the embedding network and the networks with dropout is more negligible but still visible. Interestingly, in our experiments, dropout seemed to make the networks less robust to noise in the test set, as seen in 13.

F. Convolutional Network with Weight Decay

Weight decay is another regularization technique commonly used in neural networks [6]. In this method, the weights of the network are slowly

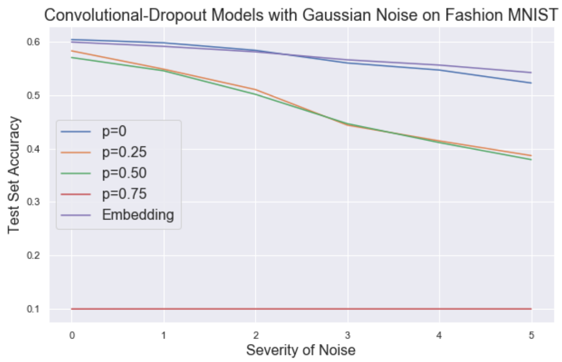

Fig. 13

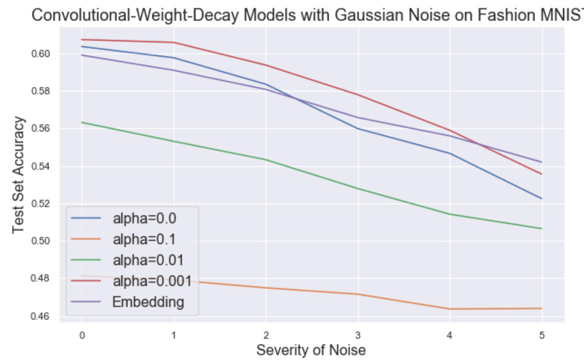

Fig. 15

decayed to 0 using a hyperparameter α, which prevents against large magnitude weights. Large magnitude weights are strongly correlated with overfitting, since slight perturbations in activations can lead to dramatic changes in the next layer activations. We compare the embedding convolutional network from the Convolutional Network section to the same baseline convolutional networks with weight decay at $\alpha = 0$, $\alpha = 0.1$, $\alpha = 0.01$, $\alpha = 0.001$.

Another factor in comparing weight decay to RFEL is that weight decay often slows down as the training procedure progresses. However, this is not the case with RFEL since the layer has no trainable parameters.

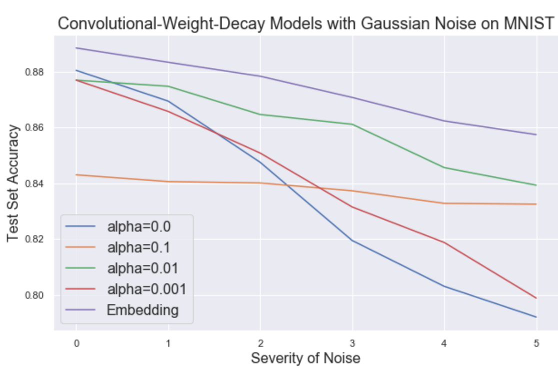

Fig. 14

In Figure 14 we see that the embedding model outperforms all other models with weight decay on MNIST. In Figure 15 the embedding network produces a lower accuracy than some of the weight decay networks. These results suggest that RFEL has comparable ability to make networks as robust as with weight decay.

V. Conclusion

We show that RFELs are an effective regularizer for neural networks. That is, on small models, RFEL improve model robustness to corruption noise and reduce overfitting. Furthermore they are a low cost solution to common problems. They are easy to insert into DNNs as a network layer, and do not empirically increase training time. Since we were compute-limited for this research, it is unclear how RFELs will perform in larger models. Furthermore, RFEL performance is heavily reliant on the hyperparameter which controls the variance of the random vector that the data is projected onto. This reliance on a hyperparameter is a downside to using RFELs.

The following will be left for future work:

- Determine whether RFEL will generalize to larger networks. In this paper, the RFEL had the most dramatic effect on smaller networks and the linear case. It will be important for future work to investigate when and why RFELs stop providing as much regularization for more complex models.
- Assess RFEL on higher dimensional data/larger images. Due to computation constraints, we evaluated RFEL only on the small datasets CIFAR10, MNIST, and Fashion-MNIST. It will be important to test whether our results generalize to more complex image datasets such as ImageNet [20]. Non-image data should also be examined.

- Analyze the effects of the RFEL σ parameter. Our implementation of RFEL is highly sensitive to the hyper-parameter which controls the variance of the random vectors which are projected onto in the embedding. In future work should explore whether an optimal σ can be defined. Furthermore, it is possible that normalizing the input size to the RFEL will allow for finding an optimal σ in a more general setting,
- Incorporate spatial preservation into RFEL. The RFEL destroys spatial data by randomly projecting the input image as a vector. It does not take the inherent local structure of the image into account. Future works could investigate random convolutions which incorporate the spatial information of most images and their effect on regularization.
- Determine the effect of RFEL on adversarial robustness. We hypothesize that the inclusion of RFEL will increase adversarial robustness, because there has been shown to be a link between corruption noise robustness and adversarial robustness [15]. However, we have not shown this directly in this work.

References

[1] D. Hendrycks and T. G. Dietterich, "Benchmarking neural network robustness to common corruptions and perturbations," *CoRR*, vol. abs/1807.01697, 2018.

[2] J. Ginsberg, M. Mohebbi, R. Patel, L. Brammer, M. Smolinski, and L. Brilliant, "Detecting influenza epidemics using search engine query data," *Nature*, vol. 457, pp. 1012–1014, 2009. doi:10.1038/nature07634.

[3] D. Lazer, R. Kennedy, G. King, and A. Vespignani, "The parable of google flu: Traps in big data analysis," *Science*, vol. 343, no. 14 March, pp. 1203–1205, 2014.

[4] R. Caruana, S. Lawrence, and C. Lee Giles, "Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping.," vol. 13, pp. 402–408, 01 2000.

[5] R. M Zur, Y. Jiang, L. Pesce, and K. Drukker, "Noise injection for training artificial neural networks: A comparison with weight decay and early stopping," *Medical physics*, vol. 36, pp. 4810–8, 10 2009.

[6] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS 4*, pp. 950–957, Morgan Kaufmann, 1992.

[7] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[8] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," in *Advances in Neural Information Processing Systems 20* (J. C. Platt, D. Koller, Y. Singer, and S. T. Roweis, eds.), pp. 1177–1184, Curran Associates, Inc., 2008.

[9] J. A. H. Anders Krogh, "Generalization in a linear perceptron in the presence of noise," 1991.

[10] I. Vasiljevic, A. Chakrabarti, and G. Shakhnarovich, "Examining the impact of blur on recognition by convolutional networks," *CoRR*, vol. abs/1611.05760, 2016.

[11] J. Li, l. Deng, Y. Gong, and R. Haeb-Umbach, "An overview of noise-robust automatic speech recognition," *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, vol. 22, pp. 745–777, 04 2014.

[12] N. Carlini and D. A. Wagner, "Towards evaluating the robustness of neural networks," *CoRR*, vol. abs/1608.04644, 2016.

[13] N. Carlini, G. Katz, C. Barrett, and D. L. Dill, "Ground-truth adversarial examples," 2018.

[14] A. Madry, A. Makelov, L. Schmidt, D. Tsipras, and A. Vladu, "Towards deep learning models resistant to adversarial attacks," in *International Conference on Learning Representations*, 2018.

[15] N. Ford, J. Gilmer, and E. D. Cubuk, "Adversarial examples are a natural consequence of test error in noise," 2019.

[16] Y. LeCun and C. Cortes, "MNIST handwritten digit database," 2010.

[17] H. Xiao, K. Rasul, and R. Vollgraf, "Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms," *CoRR*, vol. abs/1708.07747, 2017.

[18] A. Krizhevsky, V. Nair, and G. Hinton, "Cifar-10 (canadian institute for advanced research),"

[19] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *CoRR*, vol. abs/1512.03385, 2015.

[20] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A Large-Scale Hierarchical Image Database," in *CVPR09*, 2009.

VI. APPENDIX

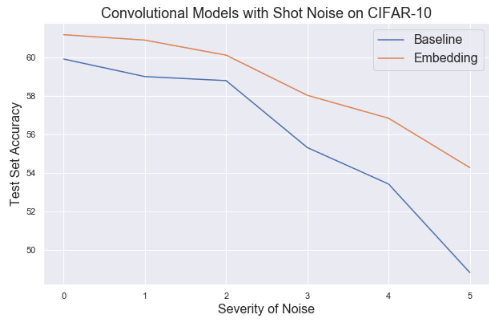

Fig. 16

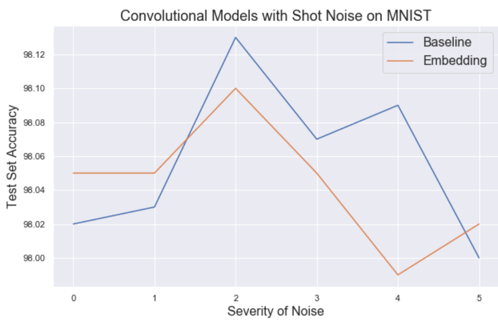

Fig. 17

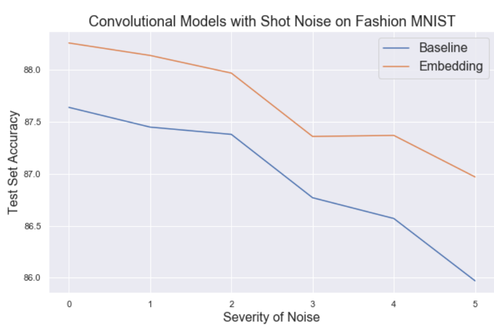

Fig. 18

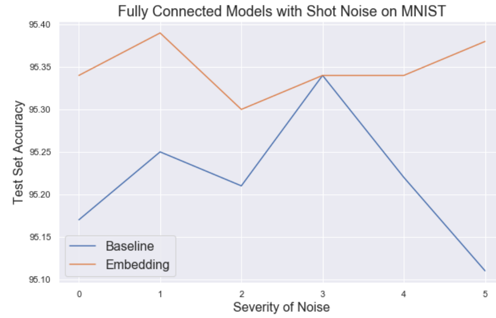

Fig. 19

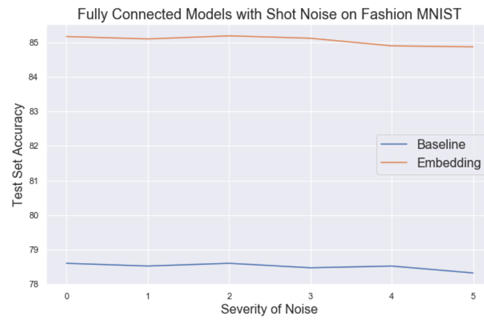

Fig. 20

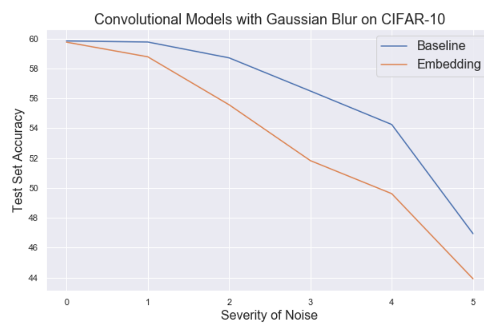

Fig. 21

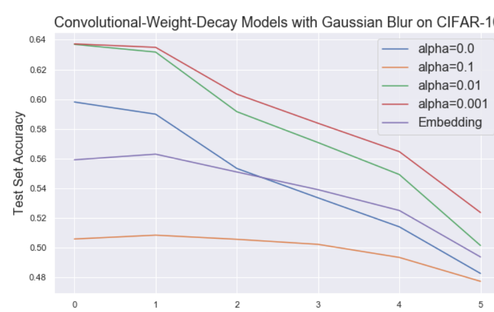

Fig. 22