COMP 551-001 Applied Machine Learning Reproducibility Challenge - Team Naive Calves

(Adversarial Dropout for Supervised and Semi-Supervised Learning)

Nabil Chowdhury*, Tiffany Wang†, John Wu‡
*Email: nabil.chowdhury@mail.mcgill.ca
Student ID: 260622155
†Email: tiffany.wang@mail.mcgill.ca
Student ID: 260684152
‡Email: john.wu@mail.mcgill.ca
Student ID: 260612056

***Abstract*—Adversarial training has recently been a successful tool in improving model generalization. In *Adversarial Dropout for Supervised and Semi-Supervised Learning*, Park et al. proposes Adversarial Dropout, a method based on adversarial training which introduces perturbation using Dropout. This report focuses on the reproducibility of this paper through the replication of their experiments. With a standard dropout as baseline, the results showed that adversarial dropout provided better performance in terms of both accuracy and loss. We then conclude a successful reproducibility of this paper.**

I. Introduction

This report will focus on studying the reproducibility of the research paper titled *Adversarial Dropout For Supervised And Semi-Supervised Learning* by Sungrae Park et al. With the increase of popularity and promising performances of adversarial training, Park suggests an alternative method using adversarial dropouts to similarly increase models' generalization by introducing perturbations. This report will include our approach to reproduce the algorithms proposed in the studied paper. We will begin by presenting an analysis and understanding of the key components discussed in the paper. Then we will form a theoretical hypothesis on the legitimacy of the results and discuss our approach to implement the said components. Finally, we will use the results of our experiments to finalize a conclusion on the reproducibility and validity of the research paper.

II. Background

The following section highlights background information necessary to the understanding of the paper.

Adversarial Dropout Background

Training models with adversarial examples have been a recent technique used to improve the generalization performance of neural networks [1]. Essentially, adversarial examples are slightly modified input samples in which the model is highly vulnerable to for forming its prediction (refer to Figure 1). They are generally obtained by applying a layer of noise over training inputs. Adversarial training works by generating and adding a small but worst-case perturbation on each of the input examples and including it in the training set.

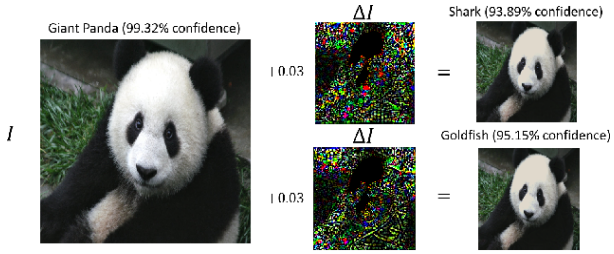


Fig. 1: Example of adversarial samples. source: Li et al., "Adversarial Examples Detection in Deep Networks with Convolutional Filter Statistics"

The main idea of this paper takes the concept of adversarial training and extends it to the dropout domain. Dropout is a regularization method where units in the neural network are randomly dropped during training in order to prevent them from co-adapting too much. [12] Rather than choosing randomly, adversarial dropout strategically selects the minimal set of dropouts that can maximize the Kullback–Leibler divergence between the outputs from a network with the dropouts and the training supervision itself. The identified adversarial dropouts are then used to reconfigure the neural network in training which reportedly improves the generalization performance and achieves better accuracies than a regular dropout model. It is also reported to show improvements over the Π model and adversarial training model [3] as shown in Figure 2.

Adversarial Dropout Expression

The goal of adversarial dropout is to define some adversarial dropout mask, $\boldsymbol{\epsilon}^{adv}$, for which the loss between the output distribution of the adversarial network, $f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon})$, and the regular dropout network, $g(\mathbf{x}, y, \boldsymbol{\theta})$ is maximized. The loss function is defined as:

$$\mathcal{L}_{AdD}(\mathbf{x}, y, \boldsymbol{\epsilon}^s; \boldsymbol{\theta}, \delta) := D[g(\mathbf{x}, y, \boldsymbol{\theta}), f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon}^{adv})] \quad (1)$$

$$\text{where } \boldsymbol{\epsilon}^{adv} := argmax_{\boldsymbol{\epsilon}; \|\boldsymbol{\epsilon}^s - \boldsymbol{\epsilon}\|_2 \leq \delta_H} D[g(\mathbf{x}, y, \boldsymbol{\theta}), f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon})]$$

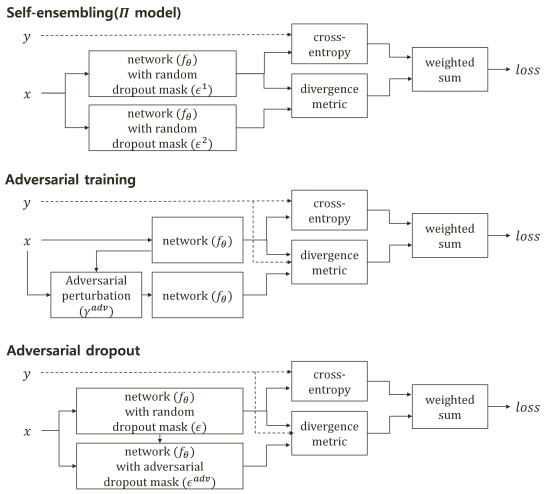


Fig. 2: Diagram description of loss functions from the Π model [2], adversarial training model [3], and the proposed adversarial dropout model. Source: original research paper. [1]

$D[\cdot, \cdot]$ is the divergence function.

A constraint sets the upper limit of the divergence between $g(\mathbf{x}, y, \boldsymbol{\theta})$ and $f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon}^{adv})$. The constraint is needed because in the scenario where the divergence reaches a point of infinity, every node in the network would essentially be dropped, making training impossible. Additionally, highly sparse dropout layers can also penalize the model, which needs to be prevented. More specifically, the mask is constructed from finding the nodes with the largest loss function gradient, which when dropped will have the biggest impact on the output.

Adversarial Dropout Learning Function

Adversarial dropout introduces particular noises into the model, thus the corresponding loss, or divergence, is incorporated into the full loss function with diminished impact. It is important that it does not overpower the original loss function. The full objective learning function is the following:

$$l(y, f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon}^s)) + \lambda \mathcal{L}_{AdD}(\mathbf{x}, y, \boldsymbol{\epsilon}^s; \boldsymbol{\theta}, \delta) \tag{2}$$

where $l(y, f_{\boldsymbol{\theta}}(\mathbf{x}, \boldsymbol{\epsilon}^s))$ is the negative log-likelihood for y given x under the sampled dropout instance $\boldsymbol{\epsilon}^s$. λ is a regularization term while δ is the constraint on the intensity of the adversarial dropout.

Validation Procedure

As validation of their proposed theory, Park et al. performed both a supervised and semi-supervised adversarial dropout learning algorithm to MNIST [7], SHVN [8], and CIFAR-10 [9] datasets. 3 convolution layers and 3 max-pooling layers were applied along with an adversarial dropout on the MNIST dataset. Performances from the standard dropout and Π models were used as baseline comparisons where the adversarial dropouts performed slightly better at a 0.99% error rate when tested on 1000 labeled samples. The same architecture was also used on the SVHN and CIFAR-10 datasets where once again a slight improvement from the baseline models. This was further extended by combining both adversarial training and adversarial dropout in the same model where the performance was even better with 3.55% test error on SVHN (1000 labeled samples) and 9.22% test error on CIFAR-10 (4000 labeled samples).

The more specific details to the exact procedure can be found in the paper, which we will be referring to for our reproducibility experiment.

III. Hypothesis

Prior to starting the reproducibility experiment, we first form a preliminary hypothesis on the validity of the adversarial dropout method.

In the recent years, adversarial training has grown in popularity especially with its application in Generative Adversarial Networks (GANs) [5]. As shown in Figure 3, GANs include a generator which creates adversarial examples of a given input in hopes of fooling the discriminator to believe that it is a real image. The discriminator itself is trained using adversarial examples in order to recognize these adversarial noises and increase generalization, which has been proven to strengthen its prediction accuracies.

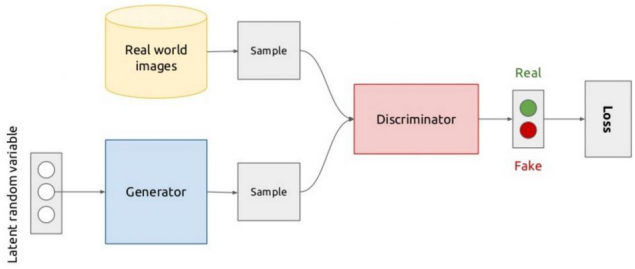


Fig. 3: Simple model showing the high level architecture of a adversarial learning model using GANs. [10]

With the proven success of adversarial training, intuitively, we believed that adversarial dropout should work out since it evolved from the same theoretical ideas. However, this would be highly dependent on our ability to choose the right constraint on the divergence limit. Having a constraint that is too high, would result in dropping too many nodes while having a constraint that is too low would achieve the opposite effect. The result of both cases would potentially lead to a suboptimal network.

The background behind the theory of dropout also contributes to our prediction on the feasibility of the research paper. While the idea of random dropout attempts to prevent the problem of co-adaptation between neighboring nodes, it does not consider which nodes are more likely co-adapting. Therefore,

by introducing a more strategic dropout, we'd be able to select and drop nodes with higher impact. This ultimately would allow for better independent updates on the neighboring nodes, which we believe would create a more unbiased model.

By looking at both the theories behind adversarial training and dropout, we believe that the reproducibility of adversarial training should be feasible.

IV. Experiment

Upon making our hypothesis, we then attempt to replicate the core components of this experiment: supervised and semi-supervised adversarial training. The goal is to determine whether adversarial dropout is an outperforming method. We will introduce a baseline model to compare the performance of the adversarial dropout model.

The models are implemented using TensorFlow.

A. Dataset

The baseline and adversarial dropout models will be trained and tested on two different datasets: MNIST and CIFAR10. In the research paper, the authors also included the SVHN dataset but we chose to not include it since our focus is not to compare datasets but rather to analyze the model itself. The data will be preprocessed in the same way as presented in the paper (refer to the original paper for more information). The data will be split into training and validation sets as follows:

TABLE I: Break Down of Training and Validation Data

	Training samples	Validation samples
MNIST	60,000	10,000
CIFAR-10	50,000	10,000

B. Supervised Dropout Models

Two different models, Tables II and III, replicated from the paper, are implements to train the two different datasets. In both cases, adversarial dropout will be tested against the baseline model, which is a random dropout supervised neural network. In is important to note that during test time, the model uses the weights learned during training and no dropout layers are applied. In order to increase training speed and accuracy, and to be aligned with what was done in the original paper, batch normalization is applied after each convolutional layer [4].

Notedly, Park also proposed an original dropout mask sampled from a Bernoulli distribution with a keep probability of $p = 1$ for CIFAR-10, in which case the dropout layer is equivalent to a fully connected layer. This lead to our doubts on the effect of the adversarial dropout layer because this means it would solely deactivate nodes without any activation. According to their proposal and adversarial dropout algorithm, any mask with arbitrary p would work. Therefore, we chose to test the model on MNIST as well, where the keep probability is set to $p = 0.5$. We chose to implement KL divergence [6], or relative entropy, as our objective loss function, as it has shown better results than quadratic error in the paper.

TABLE II: The CNN architecture for MNIST with adversarial dropout as defined in the report.

Name	Description
input	28 x 28 image
conv1	32 filters, 1 x 1, pad='same', ReLU
pool1	Maxpool 2 x 2 pixels
drop1	Dropout, $p = 0.5$
conv2	64 filters, 1 x 1, pad='same', ReLU
pool2	Maxpool 2 x 2 pixels
drop2	Dropout, $p = 0.5$
conv3	128 filters, 1 x 1, pad='same', ReLU
pool3	Maxpool 2 x 2 pixels
adt	Adversarial dropout, $p = 0.5$, $\delta = 0.005$
dense1	Fully connected 2048 $\rightarrow$ 625
dense2	Fully connected 625 $\rightarrow$ 10
output	Softmax

TABLE III: The CNN architecture for CIFAR with adversarial dropout as defined in the report.

Name	Description
input	32 x 32 RGB image
conv1a	128 filters, 1 x 1, pad='same', LReLU(α=0.1)
conv1b	128 filters, 1 x 1, pad='same', LReLU(α=0.1)
conv1c	128 filters, 1 x 1, pad='same', LReLU(α=0.1)
pool1	Maxpool 2 x 2 pixels
drop1	Dropout, $p = 0.5$
conv2a	256 filters, 3 x 3, pad='same', LReLU(α=0.1)
conv2b	256 filters, 3 x 3, pad='same', LReLU(α=0.1)
conv3c	256 filters, 3 x 3, pad='same', LReLU(α=0.1)
pool2	Maxpool 2 x 2 pixels
drop2	Dropout, $p = 0.5$
conv3a	512 filters, 3 x 3, pad='valid', LReLU(α=0.1)
conv3b	256 filters, 1 x 1, LReLU(α=0.1)
conv3c	128 filters, 1 x 1, LReLU(α=0.1)
pool3	Global average pool (6x6 $\rightarrow$ 1x1)pixels
adt	Adversarial dropout, $p = 0.5$, $\delta = 0.05$
dense	Fully connected 128 $\rightarrow$ 10
output	Softmax

1) MNIST: The network employs Adam [11] with a learning rate of 0.001 and momentums of β_1 = 0.9 and β_2 = 0.999.

2) CIFAR-10: The network employs Adam [11] with a learning rate of 0.003 and momentums of β_1 = 0.9 and β_2 = 0.999. Although in the paper, the model is trained for 300 epochs, due to the limited computational power, we arbitrarily decreased the number to 50 epochs.

V. Results

After performing the experiment, we were able to produce promising results that will be analyzed in this section. In the paper, Park uses the error rate (%) as performance metrics.

$$Error\_rate = 1 - accuracy \tag{3}$$

All source code that was executed in the experiment can be found attached to this report. All graphs corresponding to the accuracy and loss trends can be found in Appendix A.

TABLE IV: Training results for MNIST and CIFAR-10

	Error Rate (%)	
Method	MNIST	CIFAR-10
Park et al.	0.46 ± 1	5.46 ± 0.16
Baseline	2.4	17.9
Adversarial	1.8	16.9

A. MNIST Supervised Adversarial Dropout

In comparison to regular Dropout, Adversarial Dropout showed faster training and overall better results. From Figure 4, the training set accuracy and loss of both models are similar. This is expected as in both cases, the weights of the models adapted to learn the training data as closely as possible. However, the performance of the Adversarial Dropout on the test set is noticeably better. After the first epoch, the adversarial accuracy is more than four times higher than that of regular dropout. More specifically, adversarial dropout had a 0.8 accuracy as compared to a 0.15 accuracy on the regular dropout. The overall best accuracies were found to be 0.983% for adversarial and 0.976% for baseline.

Although we did not successfully reproduce an error rate as low as Park's, we did reproduce a relative improvement compared to the baseline model. Additionally, interesting results surfaced in the loss trend of the test set, as shown in Figure 4 (d). The baseline model starts overfitting at around epoch 13, whereas the adversarial model's loss stagnates. This proves that the Adversarial Dropout method does further improve the generalization, thus has a stronger regularization effect on the model.

B. CIFAR-10 Supervised Adversarial Dropout

The results indicate a slightly higher accuracy and lower loss when using Adversarial Dropout for both the training and test sets. From Figure 5, it is especially noticeable that the test accuracy and loss are better with Adversarial Dropout. The faster training and better generalization is less pronounced on this dataset than MNIST. Two possible explanations are that training was done for only 50 epochs, and CIFAR-10 is a more complex dataset than MNIST. It is possible that with more training, the test graphs in Figure 5 will diverge more, further supporting that Adversarial Dropout has a better regularization effect than standard dropout.

We were also unable to achieve the lower error rates in the paper, but still reproduced a relative improvement compared to the baseline model.

VI. Discussion

Overall, the results obtained from both MNIST and CIFAR-10 show the outperforming adversarial dropout regularization method, which provided evidence for the reproducibility of the main theory behind Park's research paper. Looking back to our original background research and hypothesis, the two main concerns that we had were:

1) The arbitrarily defined Bernoulli keep-probability of the sampled random dropout mask.
2) The constraint imposed on the maximum divergence in equation equation (1).

When training on the MNIST model, the keep-probability was set to $p = 0.5$ and performed significantly better than the baseline model which was also set to $p = 0.5$. When training on the CIFAR-10 model, the keep-probability was set to $p = 1.0$ while keeping the baseline keep-probability the same. With this configuration, it did not achieve the same relative performance as on the MNIST dataset (despite performing better). As a result, we believe that the selection of the keep-probability should not have be arbitrary, despite the original research paper indicating otherwise. However, we would like to note that with a high keep-probability $p = 1$ and a small adversarial constraint δ, the CIFAR-10 adversarial model dropout layer does not as many nodes. Therefore, the weights may still high co-adapt. Specifically, despite the low number of node deactivations, the adversarial model is still comparable to the baseline model where the keep-probability of the dropout mask is set to $p = 1$. This suggests that when strategically choosing the deactivation nodes, one can prevent the co-adaptation of neighboring weights as well as under high and random node deactivation. This provides evidence that adversarial dropout is a powerful tool.

In terms of the constraint, it was defined with a low upper-bound divergence of δ = 0.05. With this selection, we were able to reproduce the outperforming adversarial model, proving the proper selection of our constraint parameter.

VII. Constraint and Challenges

The full reproducibility of this paper was limited by the lack of computational power. The training of the CIFAR-10 dataset required high training time, which limited us to only running half of the number of epochs as suggested in the research paper.

VIII. Conclusion

Adversarial Dropout is a regularization method proposed, which is influenced by the adversarial training, by Park et al. in the "Adversarial Dropout for Supervised and Semi-Supervised Learning" . The methodology generates an adversarial dropout mask that would render the most dissimilar output from the label. The goal of this paper is to reproduce the results of the authors, which we believed to be possible. From the experiments conducted, we successfully reproduced the improvement of adversarial dropout from regular dropout. Not only do the models with Adversarial Dropout converge to higher accuracy, but they also do not overfit as quickly. In order to provide additional evidence of the full reproducibility of the paper, more experiments should be conducted, such as semi-supervised and virtual adversarial dropout trainings.

IX. Statement of Contributions

1) Nabil Chowdhury: He built the infrastructure using TensorFlow, built and trained the models, and provided insights on the analysis of the results. He helped understanding the paper and contributed to the discussion of our approach to the project.

2) John Wu: He was responsible of providing data preprocessing code. He contributed in the understanding of the paper, and the discussion of the approach of our reproducibility challenge. He mainly focused on the analysis of the training results and the redaction of the report.

3) Tiffany Wang: She developed the code for the Adversarial Dropout layer, and helped build and train the MNIST models. In addition to laying out the hypothesis, she focused on the analysis of the results and the redaction of the report.

We hereby state that all the work presented in this report is that of the authors.

References

[1] Sungrae Park, Jun-Keon Park, Su-Jin Shin, and Il-Chul Moon. *Adversarial Dropout for Supervised and Semi-Supervised Learning*. KAIST, 2017.

[2] Samuli Laine and Timo Aila. *Temporal Ensembling for Semi-Supervised Learning*. ICLR, 2017.

[3] Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, Ken Nakae and Shin Ishii. *Distributional Smoothing with Virtual Adversarial Training*. Kyoto University, 2017

[4] Ioffe, Sergey and Szegedy, Christian, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift", *JMLR: W&CP volume 37. Copy*, Proceedings of the 32^{nd} International Conference on Machine Learning, 2015

[5] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. *Generative Adversarial Networks*. University of Montreal, 2014

[6] J. R. Hershey and P. A. Olsen, "Approximating the Kullback Leibler Divergence Between Gaussian Mixture Models," 2007 IEEE International Conference on Acoustics, Speech and Signal Processing - ICASSP 07, 2007.

[7] MNIST Dataset, http://yann.lecun.com/exdb/mnist/

[8] Street View House Numbers (SVHN) Dataset, http://ufldl.stanford.edu/housenumbers/

[9] CIFAR-10 Dataset, https://www.cs.toronto.edu/ kriz/cifar.html

[10] Generative Learning Image Source, https://medium.com/@kennivich/understanding-generative-adversarial-networks-dc9f598c33b4

[11] Kingma and Ba, "Adam: A Method for Stochastic Optimization", 3rd International Conference for Learning Representations, San Diego, 2015

[12] N. Srivastava, et al., "Dropout: A Simple Way to Prevent Neural Networks from Overfitting", *Journal of Machine Learning Research*, 2014

APPENDIX A. GRAPH RESULTS

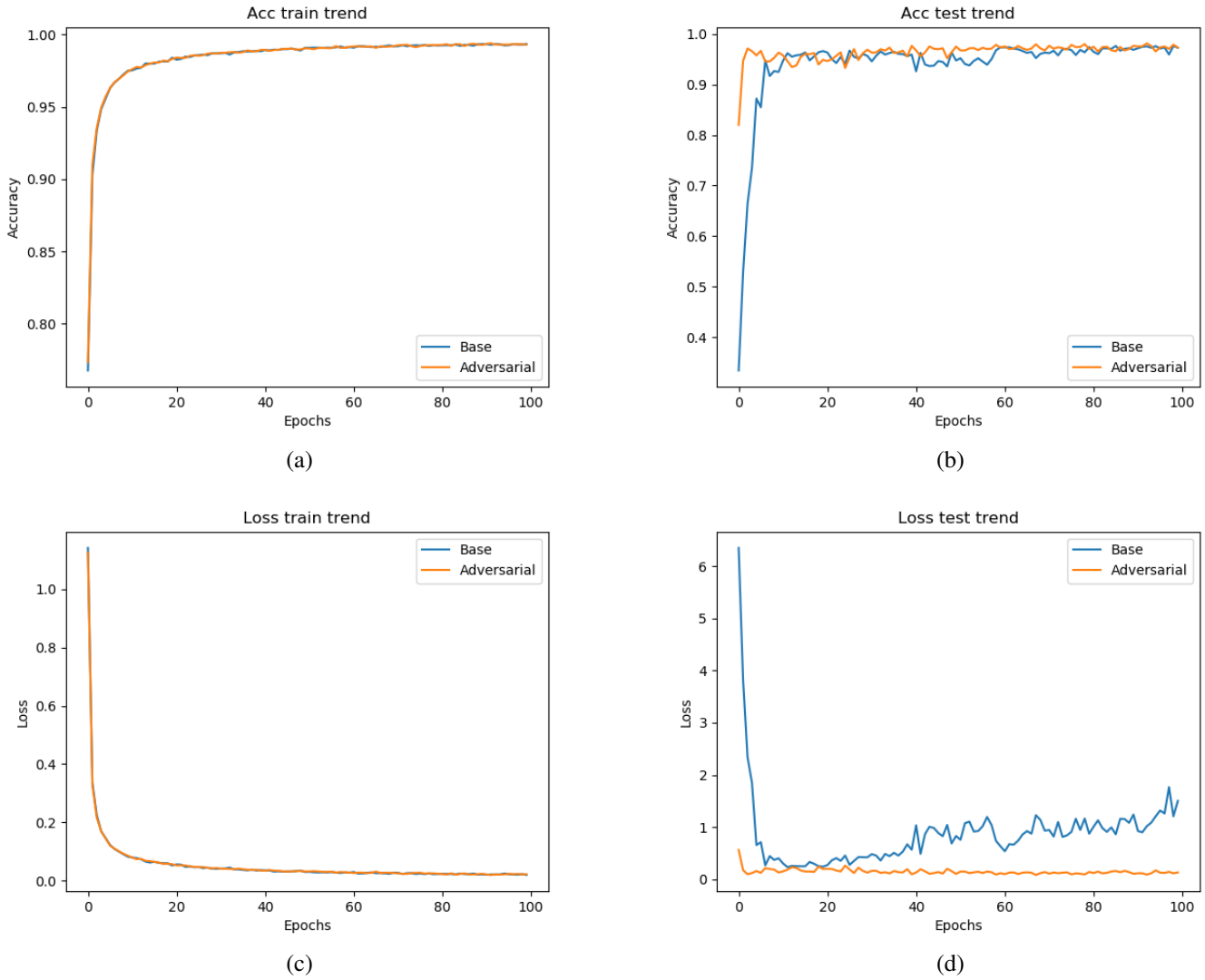
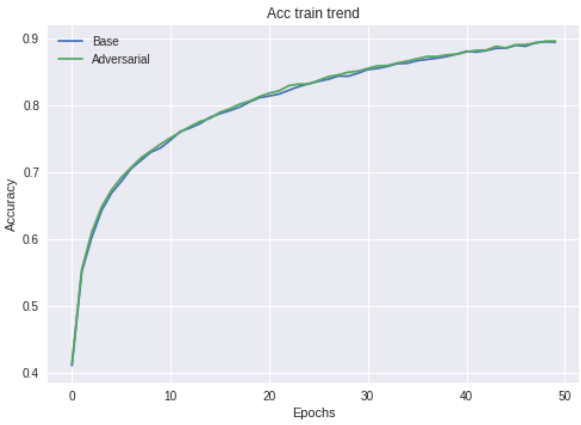
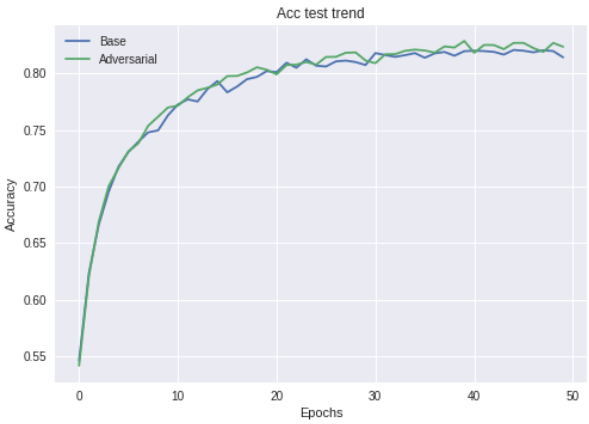


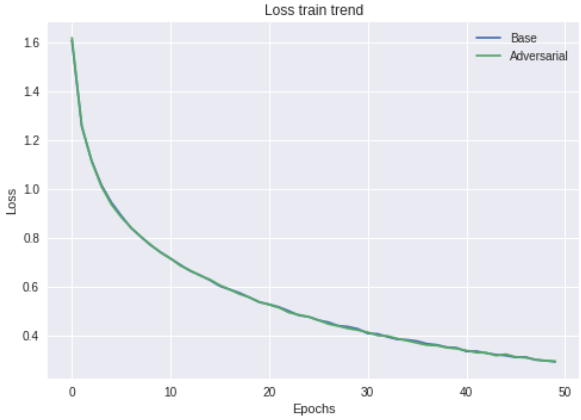
Fig. 4: MNIST: Accuracy and Loss trends with dropout mask at $p = 0.5$



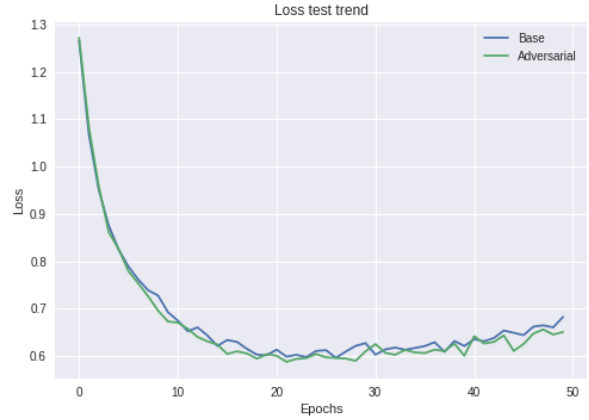
(a)



(b)



(c)



(d)

Fig. 5: CIFAR10: Accuracy and Loss trends with dropout mask at $p = 1$