Improving Dropout by Sampling Masks from Multivariate Gaussian Distributions

Linghao Zhang
Institute for Software Research
Carnegie Mellon University
linghaoz@andrew.cmu.edu

Yue Wang
Institute for Software Research
Carnegie Mellon University
yuew4@andrew.cmu.edu

Mo Li
Computational Biology
Carnegie Mellon University
moli@andrew.cmu.edu

1 Introduction

Dropout [1] has witnessed great success in training deep neural networks on various tasks including vision problems and natural language tasks. By randomly setting the activities of a fraction of the hidden units to zero during training, the neural network usually benefits from learning a huge ensemble of neural networks with good generalization and capability. Over the recent years, there have been attempts [2, 3, 4, 5, 6, 7] to improve the performance of dropout in terms of convergence and generalization.

Among various methodologies to modify the standard dropout, the most straightforward approach is to change the way the mask is generated.

Some argued that independent sampling for the dropout mask could be suboptimal because it fails to take into consideration the fact that different features may have different weights. [6] proposed a distribution-dependent dropout where multinomial sampling is used to generate the mask. To obtain parameters of the distribution for each individual feature, they proposed "evolutional dropout" to compute the distributions on-the-fly based on a mini-batch of training examples, which is similar to batch normalization.

Meanwhile, inspired by the phenomenon that activation patterns and firing rates of neurons in the human brain are random and continuous, [7] extended the standard binary dropout to continuous dropout by sampling the mask from a Gaussian distribution. They observed that continuous dropout tackles the issue of co-adaptations of features better than standard binary dropout, which helps reduce overfitting in deep neural networks.

It's reasonable to combine the two ideas and expect further improvements. In this project, we propose a new dropout algorithm we call **multivariate Gaussian dropout** to improve the performance of neural networks in terms of convergence and generalization.

Our contribution is three-fold:

1. We implemented multinomial dropout and continuous dropout from [6, 7], as well as reproduced some experimental results on MNIST and CIFAR-10 datasets.
2. We proposed and implemented multivariate Gaussian dropout: to generate the dropout mask by sampling from a multivariate Gaussian distribution, whose parameters are estimated from some statistics calculated over a mini-batch. By evaluating on the said datasets, we verified that it can improve convergence speed and generalization capability of the network.
3. We explored the impacts of three different factors related to applying dropout in training neural networks: the algorithm used for generating dropout mask, the network layers that dropout is applied, and the dropout rate. Our findings cast some light on how to make better use of dropout.

10707 Class Project (2017).

2 Related Work

In this section, we review some related work on dropout algorithms for training neural networks.

[5] generalized the standard dropout by computing the dropout probability using a adaptive, binary belief network with shared parameters of the deep networks, which allows different probabilities for different hidden units and takes into account strong correlations between components.

[2] introduced DropConnect which randomly selects a subset of weights within the network and sets them to zero instead of randomly dropping the activations. As a result, the fully connected layer with DropConnect becomes a sparsely connected layer, which effectively reduces overfitting by preventing the weights from collaborating with one another to memorize the training examples.

[3] invented Shakeout which randomly chooses to enhance or reverse the contribution of each unit to the next layer by altering the enhance factor and reverse factor associated with each unit during training. It has also been proved in [3] that Shakeout adaptively combines ℓ_0, ℓ_1, and ℓ_2 regularization, which has the power of generating sparse models while maintaining the grouping effect of the weights.

There also exist studies for improving the speed of dropout training by sampling from an approximate distribution such as Gaussian distribution without actually doing Monte Carlo sampling [4]. An order of magnitude speed-up in training time as well as better stability was observed with fast dropout training [4].

[6] proposed a multinomial dropout which is defined as $\hat{X} = X \circ \epsilon$, where ϵ is the multinomial dropout mask, $\epsilon_i = \frac{m_i}{kp_i}$, $i \in d$ and $\{m_1, ..., m_d\}$ follow a multinomial distribution. In addition, the authors also made an comparison of multinomial dropout for deep neural networks (also named as evolutional dropout) involving computing the second-order statistics of the data within a mini-batch with batch normalization and revealed that evolutional dropout has the benefits of unchanged testing inference, clear mathematical explanation of data-dependent regularization, and capability of preventing co-adapting of hidden units.

[7] proposed a continuous dropout algorithm where the dropout mask is sampled from a continuous distribution such as uniform and Gaussian, rather than the discrete Bernoulli distribution as in the standard dropout. Specifically, the continuous dropout for a single layer of linear units is defined as $\hat{X} = X \circ \epsilon$ where ϵ is the continuous dropout mask, with $\epsilon_i = I_i p_i$. In the case of uniform dropout, I_i is kept with probability of $p \approx U(0, 1)$. In the case of Gaussian dropout, I_i is kept with probability of $p \approx N(0.5, \sigma^2)$.

Both [6] and [7] reported faster convergence and smaller generalization error, as demonstrated by experimental results on benchmark datasets such as MNIST and CIFAR-10.

3 Proposed Method

We have implemented 4 different dropout, including the most widely used Bernoulli dropout, the multinomial and Gaussian dropouts as described in [6] and [7], and the our proposed multivariate Gaussian dropout which allows non-uniform distribution across different features as well as handles the feature co-adaptation issue via generating the dropout mask from a continuous distribution.

Specifically, in multivariate Gaussian dropout, the dropout mask ϵ is the continuous dropout mask with ϵ_i defined as $\epsilon_i = I_i p_i$ so that I_i is kept with probability of p_i which is sampled from a Gaussian distribution with mean μ_i and variance σ_i^2. The mean μ_i is computed based on the second-order statistics using a mini-batch of examples as given by:

$$\mu_i = \frac{\sqrt{\frac{1}{n}\sum_{j=1}^{n}[[X_j]_i^2]}}{\sum_{i'=1}^{d}\sqrt{\frac{1}{n}\sum_{j=1}^{n}[[X_j]_{i'}^2]}} \tag{1}$$

And the variance σ_i^2 is simply the sample variance over the given mini-batch of examples as given by:

$$\sigma_i^2 = \frac{\sum_{j=1}^{m}([X_j]_i - [\bar{X}]_i)^2}{m-1} \tag{2}$$

In the implementation, we also made a change in the multinomial, Gaussian and our multivariate Gaussian dropout mechanisms. Instead of changing each of the number in the layer, we applied

an additional Bernoulli mask onto the dropout mask before applying to the layer. As as result, the dropout is then defined by $\hat{X} = X \circ (\epsilon \circ \epsilon^{'})$, where ϵ is the multinomial, Gaussian or multivariate Gaussian dropout mask. And $\epsilon^{'}$ is given by a Bernoulli dropout mask with a given dropout rate. This addition mask is also intuitive because when people are looking into something, their split vision can still catch the light around it. Thus, multiplying some of those information by a multinomial mask with some of them remaining the original value can better simulate this situation than simply dropping them out. In this paper, we name it "dropout-dropout".

In addition, [6, 7] mainly studies the effects of dropout on full-connected layers of the deep neural network. We extended the scope of dropout implementation and explored how the test performance as well as convergence properties of the deep network varies as we change the type and the location of dropout mask applied to the LeNet network.

In the following experiments, we implemented the LeNet network as shown in Figure 1 and trained the network on MNIST and CIFAR-10 datasets.

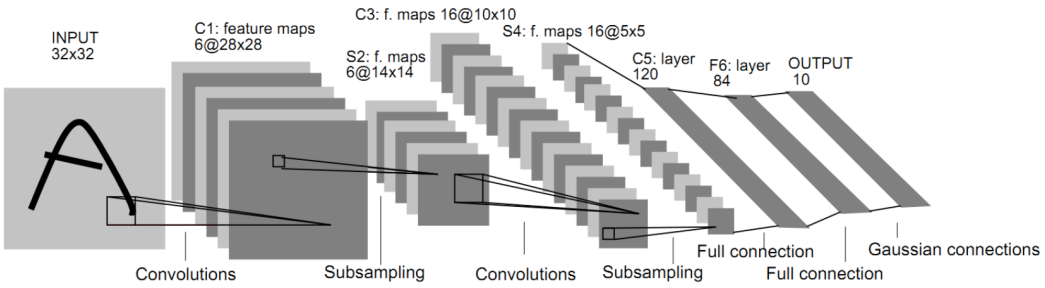

Figure 1: Architecture of the LeNet network.

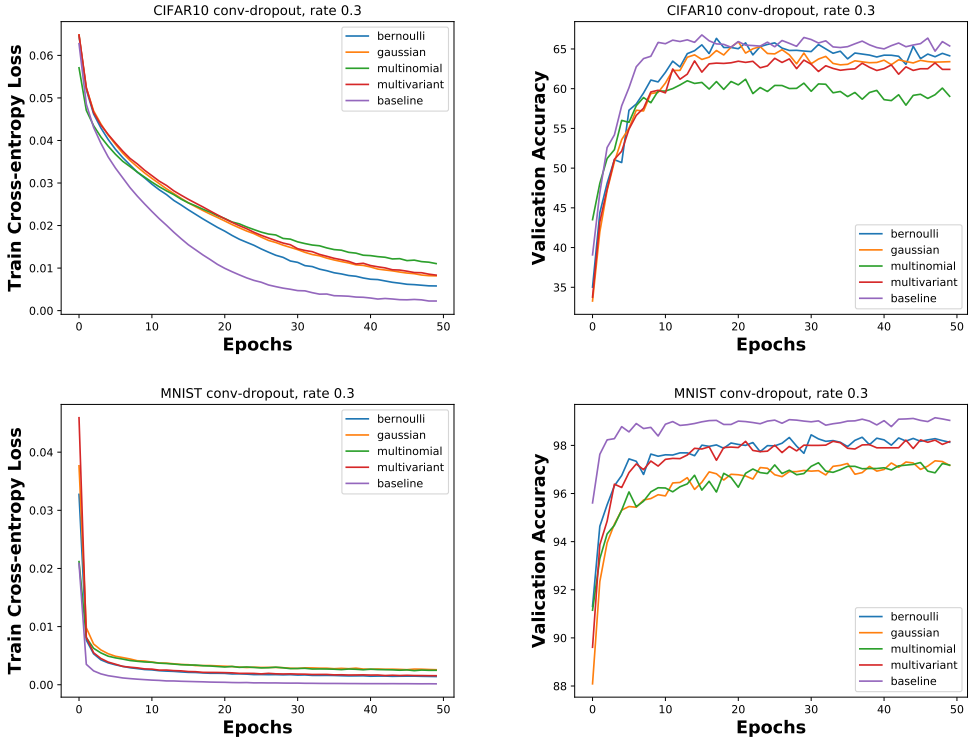

Figure 2: Train cross-entropy loss and validation accuracy with various dropout masks on the first convolution layer using MNIST and CIFAR-10 datasets.

4 Experimental Results and Analysis

We evaluated different dropout methods on MNIST and CIFAR-10 datasets. We chose these two datasets because they are popular and can be trained in time with reasonable computing resources. This ensures the feasibility and comparability of our work.

In the following experiments, we implemented the LeNet network as shown in Figure 1 and trained the network on MNIST and CIFAR-10 datasets. The reason why we use LeNet network is that it is a famous network which has plenty of training results available, making the comparison of our results to the benchmark results easier. In order to observe the impact of dropout easily, we altered the hyper parameters and reduced the baseline performance of the network. The test accuracy of our baseline network is around 65%, and the convergence took place at the 11th epoch.

4.1 Dropout on convolution layer

In our experiment, we first applied Bernoulli, multinomial, Gaussian, and multivariate Gaussian dropouts on the first convolution layer. The result is shown in Figure 2.

With the applied dropout masks, we expected to observe better performance in prediction and faster convergence in training. However, for MNIST and CIFAR-10 datasets, applying no matter what type of dropout masks on the convolution layer did not improve the performance. The LeNeT network without dropout, i.e. the baseline case, converged faster and achieved higher validation accuracy for both datasets. One possible reason why the dropouts failed to make any improvement could be that the convolution layer has the contradicting functionality with the dropout layer. The convolution core will take advantage of the nearby pixels to determine the value, which will reduce the impact of dropout layer. Practically, a better amendment to be added to the convolution layer would be batch normalization.

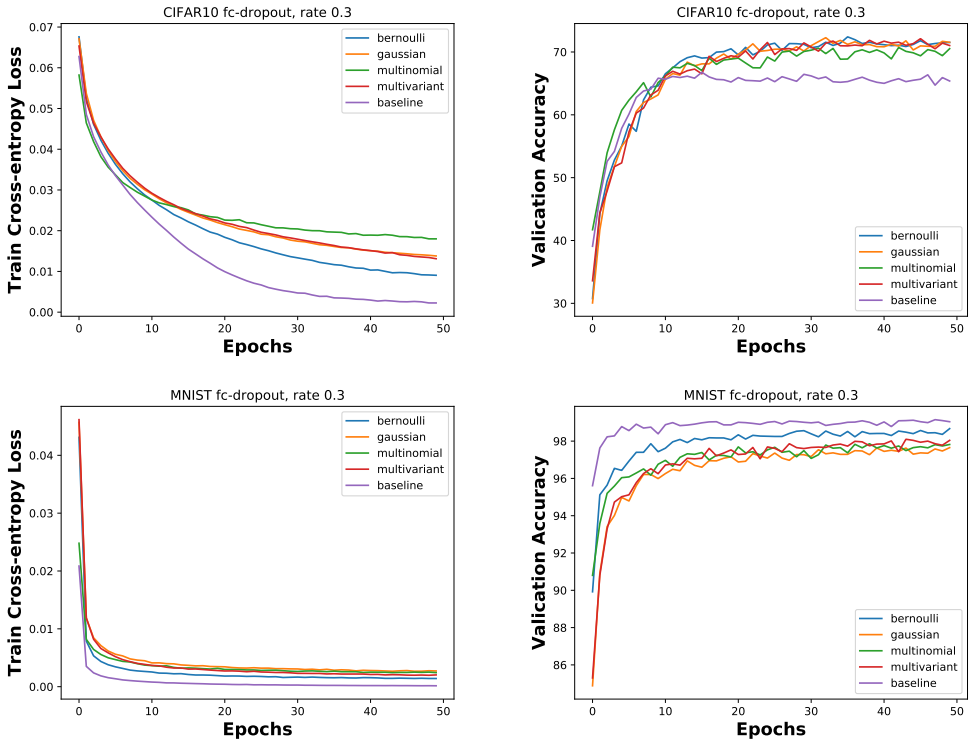

Figure 3: Train cross-entropy loss and validation accuracy with various dropout masks on the first and second fully connected layer using MNIST and CIFAR-10 datasets.

4.2 Dropout on fully connected layer

The next experiment is about the dropout on the first and second fully connected layer. We applied Bernoulli, multinomial, Gaussian, and multivariate Gaussian dropouts with a dropout rate of 30%, and the results are shown in Figure 3.

Although applying various types of dropout did not speed up the convergence during the training and the baseline loss could even go lower, it helped achieve better prediction accuracies especially on the more difficult CIFAR-10 dataset. The applied dropouts on the fully connected layer increased the accuracy to 72%, 7% higher than the baseline accuracy.

4.3 Different dropout rates

We also explored the effects of different dropout rates on the performance and convergence behavior, and the results are shown in Figure 4 and 5.

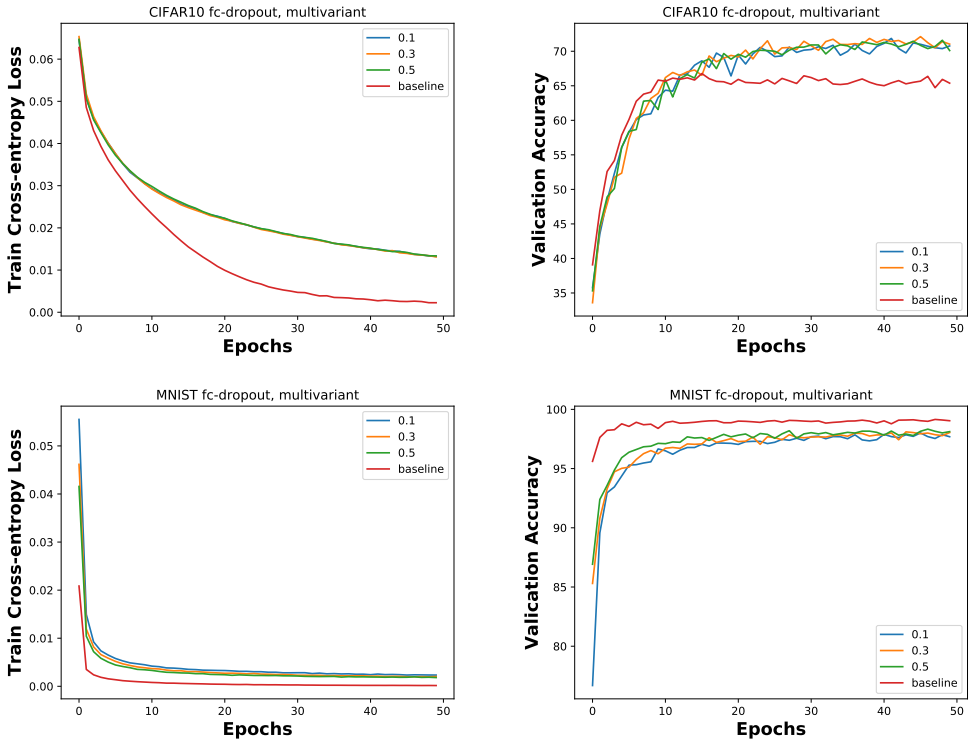

Figure 4: Train cross-entropy loss and validation accuracy with multivariate Gaussian dropout masks of different dropout rates on the first and second fully connected layer using MNIST and CIFAR-10 datasets.

For MNIST and CIFAR-10 datasets, varying dropout rates, in most cases, did not result in significant difference in convergence rate during the training and accuracy in prediction. If we examined the results closely, the dropout rate of 30% is so far the best for many cases.

4.4 "Dropout-dropout" trick

We also examined the effects of adding an additional Bernoulli mask on the multinomial, Gaussian, and multivariate Gaussian layer. The rationale of it is explained in section 3. This time we discarded MNIST dataset because it is too simple that our approach could take less effect. Also, the accuracy of MNIST in any of our experiments is always around 97% and 99%, which is difficult to compare with each other. The results of this experiment are shown in Figure 6.

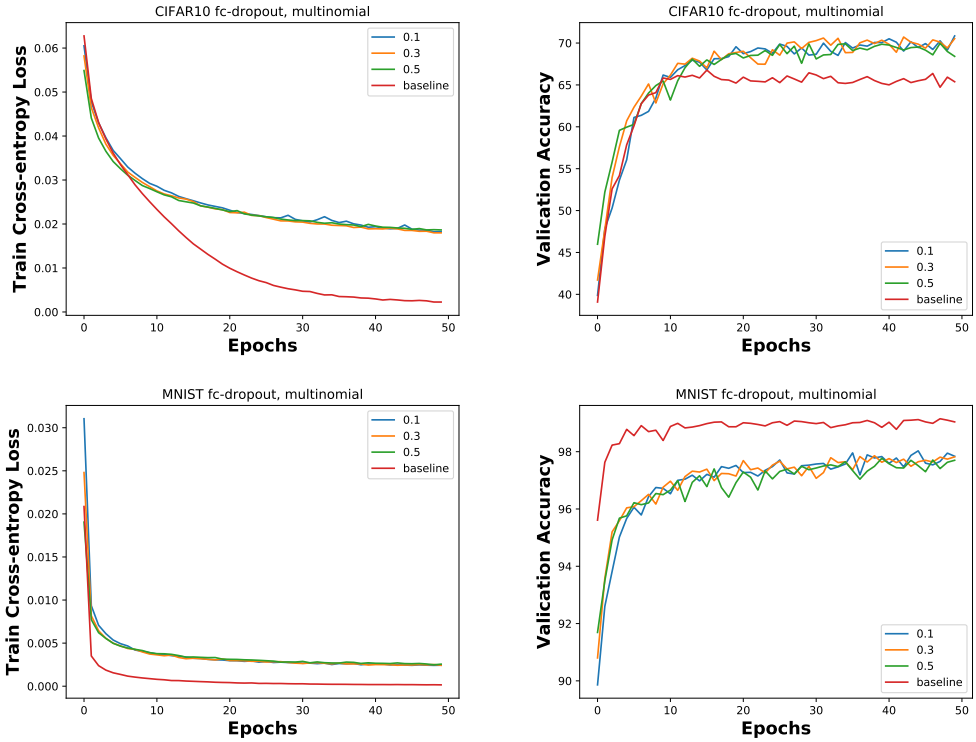

Figure 5: Train cross-entropy loss and validation accuracy with multinomial dropout masks of different dropout rates on the first and second fully connected layer using MNIST and CIFAR-10 datasets.

From the result, we can see that the "dropout-dropout" trick works well. It enhanced the test accuracy with 2% based on the previous dropout methods. The best accuracy achieved by Gaussian and multivariate Gaussian even hit 74.5%. Moreover, this trick can even improve the result of dropouts that are applied to convolution layer. The accuracy of can also hit 73% to 74% (7% more than previous dropouts).

Figure 7 summarizes the train cross-entropy loss and validation accuracy of several best cases with the implementation of the "dropout-dropout" trick, and impressively, the best prediction accuracy obtained with the "dropout-dropout" trick on multivariate Gaussian or Gaussian dropout so far is 74%, which is almost 10% higher than the baseline case.

5 Conclusions and Future Work

In this paper, we proposed multivariate Gaussian dropout, a new distribution-dependent dropout algorithm that combines intuitive ideas from previous work. Experimental results on MNIST and CIFAR-10 datasets verified that the proposed method can help with convergence and generalization of neural networks.

We also explored three dimensions of applying the dropout mask. We conducted experiments with different dropout algorithms, layers to apply dropout and dropout rates. Our findings provide some basis for choosing the best dropout method given a network architecture or a dataset.

Our current approach has some limitations that we hope to work on in the future.

1. While the proposed method speeds up convergence, the actual training time is not necessarily reduced because calculating the dropout mask is now more complicated. We should profile the current implementation and try to develop a more efficient algorithm if necessary.

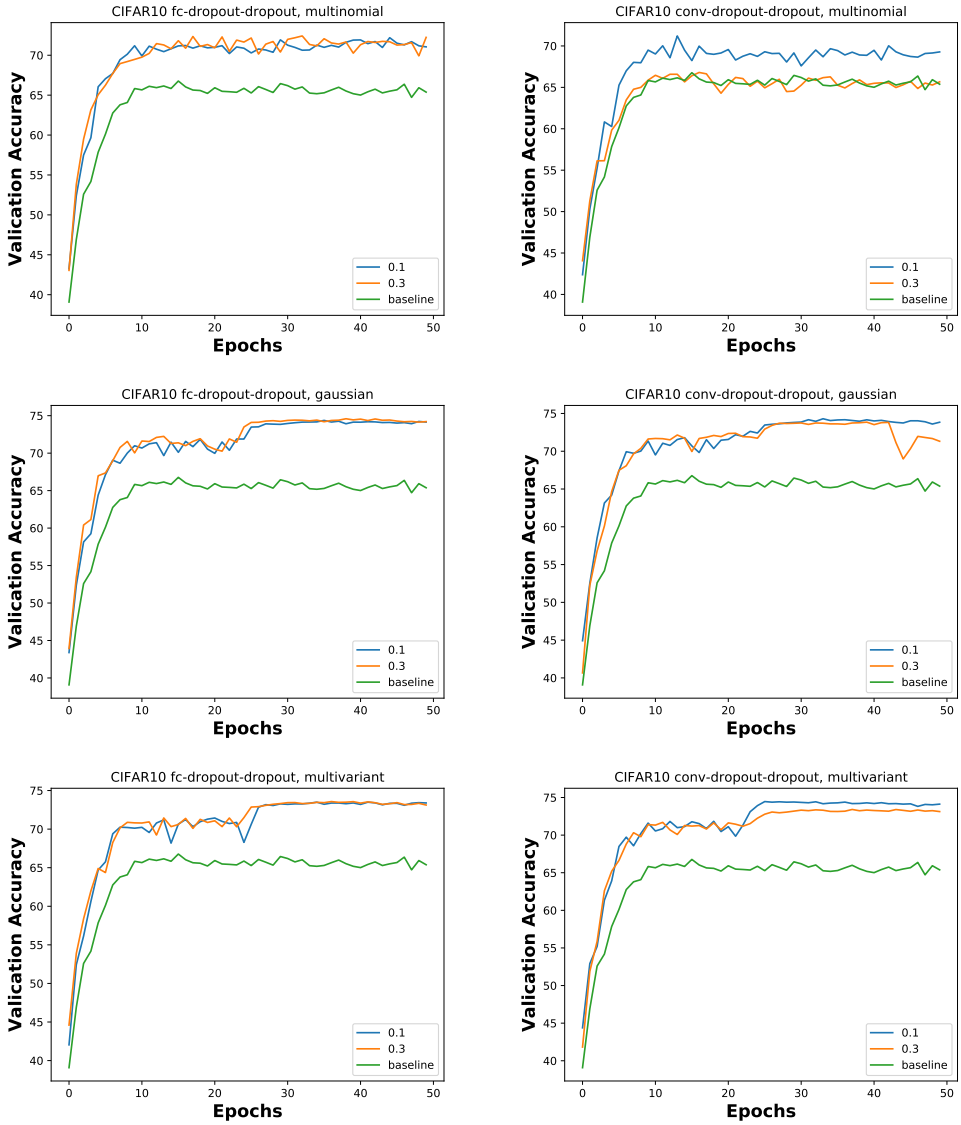

Figure 6: Validation accuracy of applying the additional Bernoulli dropout to various dropout masks on the first and second fully connected layer or on the first convolution layer using MNIST and CIFAR-10 datasets.

2. The current method for estimating mean and variance of the Gaussian distribution is naïve. We could seek inspirations from statistical learning literature and explore better approaches to estimate these parameters.
3. As dropout can be naturally viewed as an ensemble of models, we could try to derive a error bound for this ensemble, so as to provide a more solid theoretical basis.

References

[1] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[2] Li Wan, Matthew Zeiler, Sixin Zhang, Yann L. Cun, and Rob Fergus. Regularization of neural networks using dropconnect. In Sanjoy Dasgupta and David Mcallester, editors, *Proceedings of*

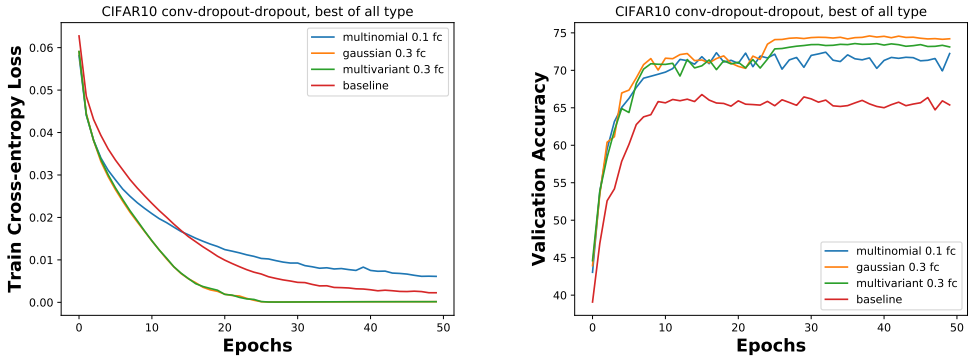

Figure 7: Train cross-entropy loss and validation accuracy of applying the additional Bernoulli dropout to various dropout masks on the first convolution layer using CIFAR-10 dataset.

the 30th International Conference on Machine Learning (ICML-13), volume 28, pages 1058–1066. JMLR Workshop and Conference Proceedings, May 2013.

[3] Guoliang Kang, Jun Li, and Dacheng Tao. Shakeout: A new approach to regularized deep neural network training. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[4] Sida Wang and Christopher Manning. Fast dropout training. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 118–126, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR.

[5] Jimmy Ba and Brendan Frey. Adaptive dropout for training deep neural networks. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3084–3092. Curran Associates, Inc., 2013.

[6] Zhe Li, Boqing Gong, and Tianbao Yang. Improved dropout for shallow and deep learning. *Advances in Neural Information Processing Systems*, 2016.

[7] Xu Shen, Xinmei Tian, IEEE Member, Tongliang Liu, Fang Xu, Dacheng Tao, and IEEE Fellow. Continuous dropout. *IEEE Transactions on Neural Networks and Learning Systems*, 2017.

6 Appendix

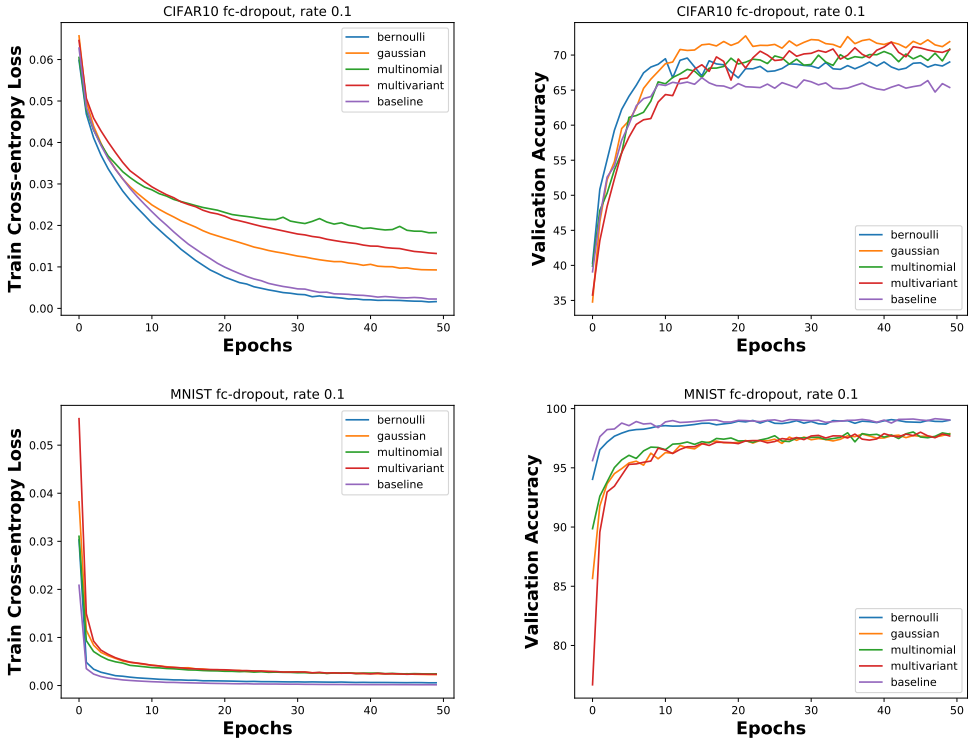

Figure 8: Train and validation cross-entropy loss as well as the prediction accuracy with various dropout masks.

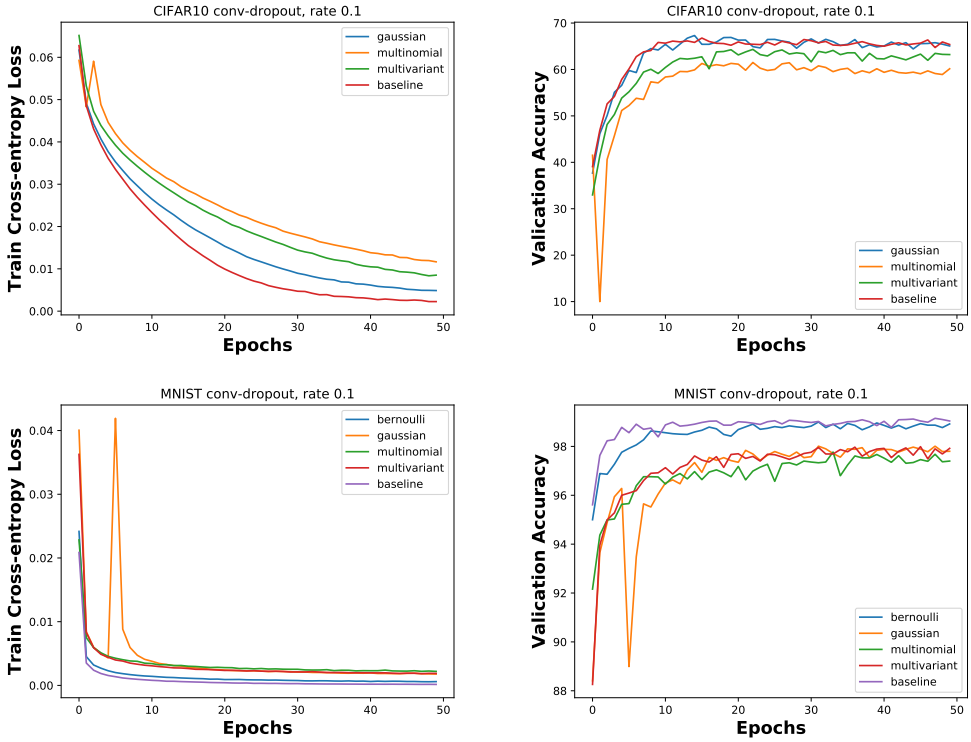

Figure 9: Train and validation cross-entropy loss as well as the prediction accuracy with various dropout masks.