Summary and discussion of: "Dropout Training as Adaptive Regularization"

CASIA final report: 2021 Fall*

Yuxiang Deng

1 Introduction

Introduced by Hinton et al.[1] dropout training, which typically omits some random subsets of features in the training process, has been proven to be a useful technique to control overfitting. Diving deep into the reason for its success, Wager et al.[2] found that dropout as well as the additive noise can be seen as an implicit regularization in the model. Other than the traditional L2-regularization, dropout has an adaptive property, which is beneficial for detecting rare but discriminative features in the sample. To better illustrate this property and improve the training efficiency, the paper come up with a quadratic approximation of the dropout penalty, so that dropout training can be implemented by simply adding a penalty to the loss function rather than manipulating the dataset. Empirically, the paper shows that the quadratic term is a proper approximation of additive and dropout noise. Based on this, the paper reveals a connection between dropout and AdaGrad by simply replacing the penalty term of SGD with dropout regularizer, showing that they share the same goal and are first-order equivalent to each other in gradient descent procedure. As an application of the finding, a semi-supervised learning algorithm is provided in the end.

This paper provides with a novel and useful insight into understanding sample disturbance, especially for dropout technique. Typically, regularization is the most common technique for addressing overfitting. This is also the case in dropout training. It is easy to illustrate the implicit regularization in dropout: we calculate the degree of freedom of a linear regression model over different δ in dropout. As shown in the figure1, the parameter δ plays an important role in regularization. That seems to be a reason why dropout succeeds in avoiding overfitting. But more questions are still unsolved, e.g., why dropout can play a role in learning rare and discriminative features? How can we compare it with other regularizers? How to make use of its benefit in training?

In the following pages, we will cover these topics with experiments: firstly, introduce the quadratic approximation of dropout penalty and its accuracy; secondly, describe and confirm that dropout outperforms L2-regularization as expected; last but not least, the application of dropout regularizer in semi-supervised learning.

*Github Link: https://github.com/Jasondyx/CASIA_final_report

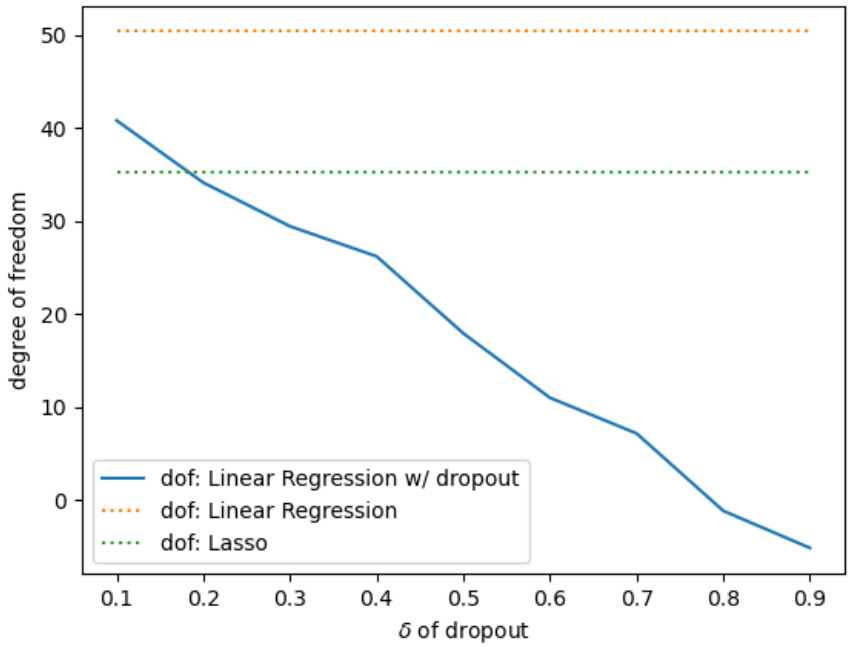

Figure 1: Example for model combination

2 Result and Discussion

2.1 Artificial Feature Noise as Regularization

To begin with, we need to have a tiny discussion about the general connection between artificial feature noise and GLM.

Artificial Feature Noise

There are two widely-used types of artificial noise:

- Additive Gaussian noise: $v(x_i, \xi_i) = x_i + \xi_i$, where ξ_i follow gaussian distribution.
- Dropout noise: $v(x_i, \xi_i) = x_i \odot \xi_i$, where each ξ_i follows binomial distribution of $\{0, (1-\delta)^{-1}\}$ with parameter δ.

Generalized Linear Models

A GLM initiates from an assumption on the distribution of the response y, given the input feature x:

$$p_\beta(y|x) = h(y)exp\{yx \cdot \beta - A(x \cdot \beta)\}$$

The loss function can be defined as followed:

$$loss_{x,y}(\beta) = -log\, p_\beta(y|x) = -log(h(y)) - yx \cdot \beta + A(x \cdot \beta)$$

Given the noised features $\tilde{x}_i$ and response y_i with a training sample count of n, our goal is to estimate β by maximizing the likelihood:

$$\begin{aligned}\hat{\beta} &= arg\min_{\beta} \sum_{i=1}^{n} E_{\xi}[loss_{\tilde{x}_i, y_i}(\beta) \\ &= arg\min_{\beta} \sum_{i=1}^{n} -yx \cdot \beta + E_{\xi}[A(\tilde{x}_i \cdot \beta)] \\ &= arg\min_{\beta} \sum_{i=1}^{n} loss_{x_i, y_i}(\beta) + R(\beta)\end{aligned}$$

where

$$R(\beta) = \sum_{i=1}^{n} E_{\xi}[A(\tilde{x}_i \cdot \beta)] - A(x_i \cdot \beta)$$

2.2 A Quadratic Approximation to the Noising Penalty

Although the regularization effect of noising can be extracted as a penalty term in the loss function, the interpretation of this term is not that intuitive. Also, to make use of this term as a regularizer, we need a more simplified and computational friendly version of it.

By taking the second-order expansion of $A(\cdot)$ around $x \cdot \beta$, we get that

$$E_{\xi}[A(\tilde{x_i} \cdot \beta)] = A(x_i \cdot \beta) + \frac{1}{2} A''(x_i \cdot \beta) Var_{\xi}[\tilde{x}_i \cdot \beta]$$

Therefore,

$$R^q(\beta) = \frac{1}{2} A''(x_i \cdot \beta) Var_{\xi}[\tilde{x}_i \cdot \beta] \tag{1}$$

Accuracy of quadratic approximation

As we can see, the quadratic approximation to noise penalty is pretty simplified. But before further interpretation, we need to experiment on its accuracy of approximation. Figure 2 and Figure 3 compares the noising penalties R and R^q for logistics regression in the case that ξ is a gaussian noise and dropout noise, respectively.

In Figure 2, we compare the accuracy of quadratic approximation under additive gaussian noise cases, over different mean parameter p and noise level σ. In general, we can see that R^q is accurate, with a pattern of overestimating in ambiguous p (p is close to 0.5) and underestimating in more deterministic p (p is far from 0.5). Note that we cannot get a closed-form expression of the exact penalty here, so we use Gauss-Hermite procedure to do the approximation. (In comparison, the authors of original paper use Monte Carlo to address this problem according to the record of reviews.)

In Figure 3, we extend the comparison to dropout noising cases, which is omitted in the original paper but may be important for this topic. Generally, these two cases share a similar pattern in approximation accuracy. However, as delta gets larger, the accuracy decreases significantly. A rough conclusion of this is that there may be significant bias in quadratic approximation when delta reaches a certain level (0.5 or above). Note that in

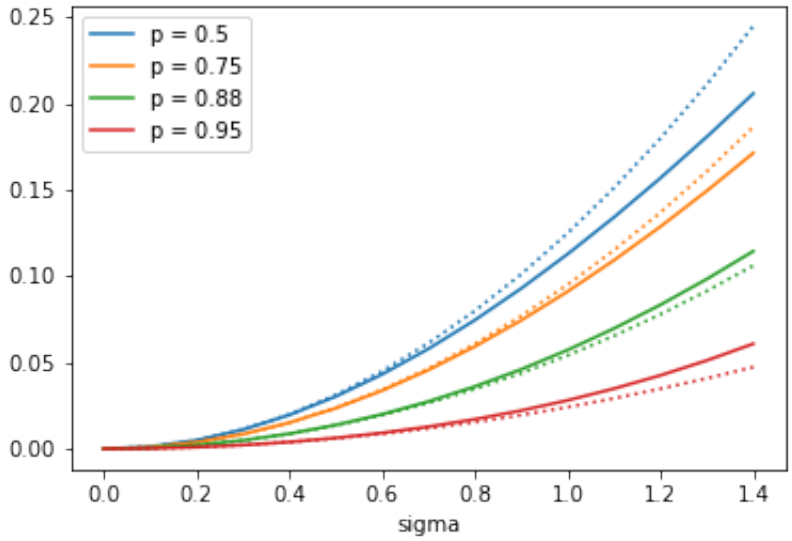

Figure 2: R^q approximation LR-additive

this case, we use central limit theorem to approximate the probability distribution of $\tilde{x}_i \cdot \beta$ and also use Gauss-Hermite procedure to calculate the exact penalty, which may also be a source of bias.

Interpretation of R^q

From the expression of R^q, we see that the penalty consists of two variance components. The first one $A''(x_i \cdot \beta)$ corresponds the variance of response y_i, since log-partition function $A(\cdot)$ is also the moment generating function of y_i. The second component $Var_\xi[\tilde{x}_i \cdot \beta]$ is the variance generated by artificial noise.

2.3 Regularization Analysis and Simulation with Logistics Regression

To be more specific, we derive the expression of regularization term R^q in Logistics Regression, trying to get more insight through comparison. An experiment of simulation will also be provided in this section to prove our assertions.

2.3.1 Regularization based on two types of Noise in Logistics Regression

Additive Noise

Since the log-partition function for logistics regression is $A(x_i \cdot \beta) = lop(1 + e^{x_i \cdot \beta})$, we can get that

$$R^q(\beta) = \frac{1}{2}\sigma^2 ||\beta||_2^2 \sum_{i=1}^{n} p_i(1 - p_i)$$

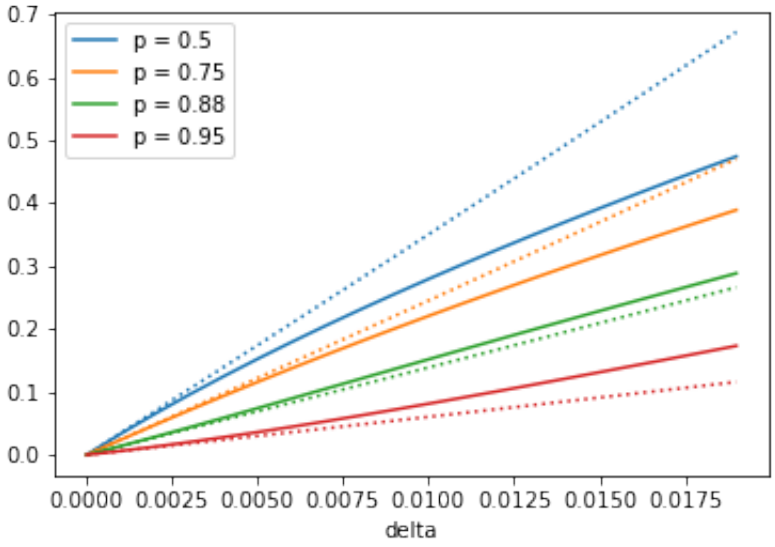

Figure 3: R^q approximation LR-dropout

where $p_i = (1 + e^{-x_i \cdot \beta})^{-1}$ is the predicted probability of $y_i = 1$ and σ^2 is the variance of additive noise ξ. This type of penalty controls the absolute value of βs and encourages more deterministic p_is (moves p_i away from 0.5) at the same time.

Dropout Noise

Similarly, we can derive the penalty term of Logistics Regression in dropout noise cases

$$R^q(\beta) = \frac{1}{2}\frac{\delta}{1-\delta}\sum_{i=1}^{n}\sum_{j=1}^{d} p_i(1-p_i)x_{ij}^2\beta_j^2$$

The most significant difference between this penalty and others is that this term is adaptive to the value of features x_{ij}^2. Therefore, for some rare but discriminative (meaning that the feature can pull p_i to a confident level) features, the penalty to β_j is relatively low, which essentially explains the reason why dropout helps to learn features of this type.

2.3.2 Simulation: experiments on the effect of different regularization

With a similar setting to the simulation in the original paper, we conduct the experiments to test the performance of four different types of model:

- Ordinary Logistics Regression (MLE)

$$\hat{\beta} = arg\min_{\beta}\ loss_{x,y}(\beta) = arg\min_{\beta}\sum_{i=1}^{n} loss_{x_i,y_i}(\beta)$$

- Logistics Regression with L2 regularization

$$\hat{\beta} = arg\min_{\beta}\ loss_{x,y}(\beta) + \frac{1}{2}\lambda||\beta||_2^2$$

- Logistics Regression with dropout manipulation on dataset

$$\hat{\beta} = arg\min_{\beta}\ loss_{\tilde{x},y}(\beta)$$

- Logistics Regression with dropout penalty in loss function[1]

$$\hat{\beta} = arg\min_{\beta}\ loss_{x,y}(\beta) + \lambda\, R^q(\beta)$$

Data

We generate dataset with some rare and discriminative features, as described in Appendix A.1 of the original paper, for this experiment. For each sample of the dataset, there are 1050 features, among which only 50 of them are discriminative. Each of the discriminative features has a probability of 4% to be active. Under this setting, we set the number of samples for each training to be 75 and the coefficient of each discriminative feature, i.e. β_j, to be 0.2792. (to ensure $E[|x_i \cdot \beta|] = 2$).

Experiment Results and Discussion

Table 1 shows the results of this experiment. Note that each accuracy score is averaged by 100 times of simulations.

Model	**All Data**	**Active Data**
Ordinary Logistics Regression (MLE)	0.53	0.61
Logistics Regression with L2 regularization	0.51	0.53
Logistics Regression with dropout manipulation on dataset	0.58	0.89
Logistics Regression with dropout penalty in loss function	0.58	0.88

Table 1: Accuracy score for models with different regularization. The first column indicates results over the full test dataset (consist of 75 samples with the same setting with training dataset), while the second columns indicates accuracy over the active data samples (those with active features).

As we can see, the accuracy on active testing data is significantly higher than that on full testing data for all models. Two types of dropout regularization technique perform similarly, while the model with R^q penalty term in loss function is way more computaionally efficient in the training process, compared with directly dropout in dataset. More importantly, for a dataset full of rare and discriminative features like this, dropout regularization significantly outperforms L2-regularizatin. The underlying reason is simple: dropout penalty has an adaptive property that avoid heavy penalty on β for those effective features, while L2 regularizer treats each feature in an equal way.

[1]Because of the expression of penalty term: $R^q(\beta) = \frac{1}{2}\frac{\delta}{1-\delta}\sum_{i=1}^{n}\sum_{j=1}^{d} p_i(1-p_i)x_{ij}^2\beta_j^2$, the optimization function is not convex anymore. Here we solve it by fixing $A''(x_i \cdot \beta) = p_i(1-p_i)$ first, optimizing β^* and updating $A''(x_i \cdot \beta^*)$ iteratively until convergence.

2.4 Application: A Semi-Supervised Learning Algorithm

Based on the finding above, the dropout penalty indeed plays an important role in adaptively regularizing the parameters and learning the rare and discriminative features. Further, we can see that the penalty term only utilizes the information from features, i.e., x. This finding triggers an idea of making use of other source of data, especially those unlabeled ones, in the training process. In this section, we discuss about a semi-supervised learning method based on dropout penalty originated by the paper.

Suppose we have n samples of labeled data and m samples of unlabeled data (only features). Instead of training model only on the labeled data, we can plug the unlabeled data into the penalty term

$$R_*(\beta) = \frac{n}{n + \alpha m}(R(\beta) + \alpha R_{Unlabeled}(\beta))$$

where $R(\beta)$ is the original penalty estimate and $R_{unlabeled}(\beta)$ is computed by the unlabeled examples. The discount factor α is suggested to be in $[0.1, 0.4]$. In the simulation, we set it to be 0.4.

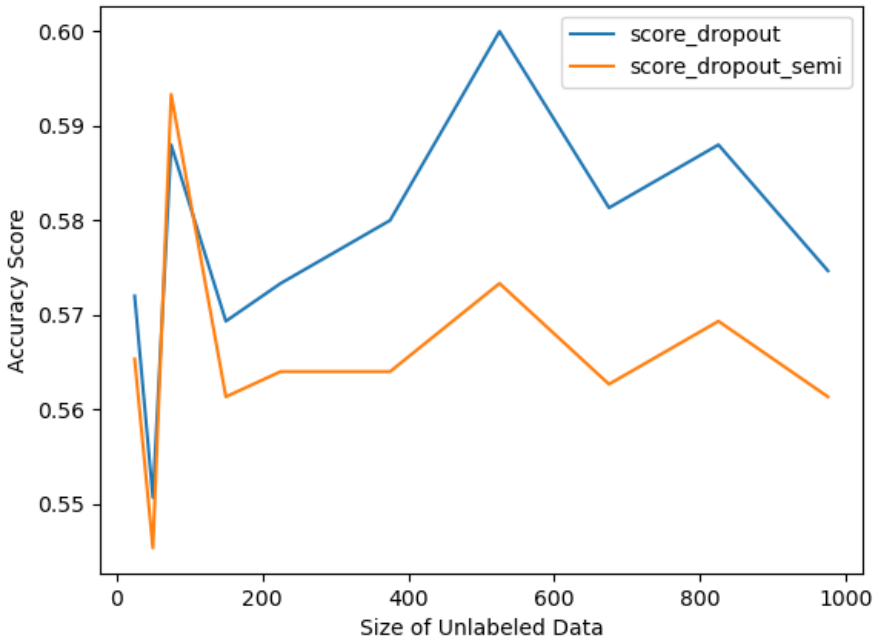

Figure 4: Semi-supervised learning over different unlabeled data sizes. The size of labeled training samples is 75, while the size of unlabeled samples varies from 25 to 975. The coefficient of the penalty term, i.e. λ, is 32 here.

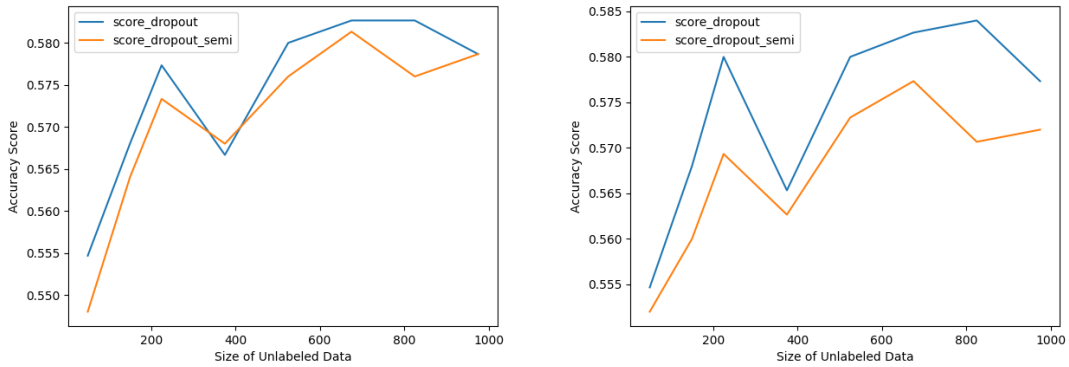

Figure 5: An other version of semi-supervised dropout training. Here, the regularization effect of unlabeled samples is normalized: $R_*(\beta) = \frac{1}{1+\alpha}(R(\beta) + \frac{\alpha n}{m} R_{Unlabeled}(\beta))$. The left figure is for the case of $\alpha = 0.1$, and the right one is for $\alpha = 0.4$.

Experiment of semi-supervised learning

To make it more comparable to the results we got from the previous experiments, we try to do simulation under the same setting as before. Here, we generate different sizes of unlabeled samples besides the labeled data, and try to experiment on the performance of semi-supervised dropout training.

Figure 4 shows that with fixed discount factor $\alpha = 0.4$, the performance is better when the labeled and unlabeled samples are of similar size. From the expression of penalty term, we can see that when number of unlabeled samples gets too large, $R_{Unlabeled}(\beta)$ will dominate the penalty term, which may negatively affect the model's performance.

As an attempt at solving this problem, we also conduct tests on an adjusted version of semi-supervised dropout penalty. Figure 5 shows the corresponding performance under the case of $\alpha = 0.1$ and $\alpha = 0.4$. In these cases, the performance of semi-supervised training seems to be immune to the effect of overwhelming unlabeled samples. But still a certain size of unlabeled samples is preferred by the model. Note that all the experiment results reported in this section are averaged by 10 repeats, while the coefficient of penalty λ has not been tuned. Hopefully, with further tuning the parameters, the semi-supervised training method will perform better.

3 Conclusion

In this paper, we reproduced most of the key results of a prestigious paper "Dropout Training as Adaptive Regularization"[2] and provided discussion on the key issues. We started from confirming the regularization effect of dropout technique by calculating degrees of freedom and revealed the role of parameter δ is playing in dropout. Then, we introduced the important connection between artificial noise and regularization penalty, which also led to a quadratic approximation of the regularizer. Some experiments on the accuracy of this key approximation was also provided. Thirdly, we put much emphasis on the intuition of

dropout penalty term (the quadratic approximation), diving deep into its benefit in finding rare and discriminative features in Logistics Regression. Our experiments on this topic also verified the advantages of dropout, compared to other regularizers like L2. At the end, we had an experiment on the semi-supervised training method initiated by the paper. The result showed that the performance of semi-supervised training is sensitive to the hyper-parameters like unlabeled sample size, discount factor α and coefficient of penalty term λ.

Our paper also revealed some of the potential deficiency of the original paper. For example, although the quadratic approximation of dropout penalty is intuitive and computational friendly, it may not be always accuracy, especially when δ reaches a large number ($\delta > 0.5$), which is usually the case in practice. Therefore, some of the underlying effect may not be fully captured by the quadratic approximation and there may be deeper interpretation about its benefit beyond the existing one.

Anyway, this is an impressive paper with rigorous derivation and solid experiments. It significantly updates people's knowledge about dropout technique and reveals the power of adaptive regularization methods. It is indeed an important cornerstone for people who want to gain deep insight into dropout technique and implicit regularization.

References

[1] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," 2012.

[2] S. Wager, S. Wang, and P. S. Liang, "Dropout training as adaptive regularization," in *Advances in Neural Information Processing Systems* (C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, eds.), vol. 26, Curran Associates, Inc., 2013.