How to Use Dropout Correctly on Residual Networks with Batch Normalization (Supplementary Material)

Bum Jun Kim[1] **Hyeyeon Choi**[1] **Hyeonah Jang**[1] **Donggeon Lee**[1] **Sang Woo Kim**[1]

[1]Department of Electrical Engineering, Pohang University of Science and Technology, Pohang, South Korea

A PROOF OF PROPOSITION 2

Proof. The variances of PostDropout and PreDropout are represented as follows:

$$\mathrm{Var}[\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x})] = \mathrm{E}[(\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x}))^2] - (\mathrm{E}[\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x})])^2, \tag{1}$$

$$\mathrm{Var}[\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x})] = \mathrm{E}[(\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x}))^2] - (\mathrm{E}[\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x})])^2. \tag{2}$$

We note that $\mathrm{E}[\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x})] = \mathrm{E}[\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x})]$. Thus, we investigate the difference between the first terms in Eqs. 1 and 2.

First, as PostDropout drops columns, we have

$$\mathrm{E}[(\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x}))^2] = \frac{1}{p^2}\,\mathrm{E}[m_{i,i}(w_{i,1}x_1 + \cdots + w_{i,n}x_n)^2] \tag{3}$$

$$= \frac{1}{p^2}\,\mathrm{E}[m_{i,i}(\sum_{j=1}^{n} w_{i,j}x_j)^2] \tag{4}$$

$$= \frac{1}{p}\sum_{j=1}^{n} w_{i,j}^2\,\mathrm{E}[x_j^2] + \frac{1}{p}\sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k]. \tag{5}$$

Secondly, as PreDropout drops rows, we obtain

$$\mathrm{E}[(\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x}))^2] = \frac{1}{p^2}\,\mathrm{E}[(m_{1,1}w_{i,1}x_1 + \cdots + m_{n,n}w_{i,n}x_n)^2] \tag{6}$$

$$= \frac{1}{p^2}\,\mathrm{E}[(\sum_{j=1}^{n} m_{j,j}w_{i,j}x_j)^2] \tag{7}$$

$$= \frac{1}{p^2}\,\mathrm{E}[\sum_{j=1}^{n} m_{j,j}^2 w_{i,j}^2 x_j^2 + \sum_{j=1}^{n}\sum_{k\neq j}^{n} m_{j,j}m_{k,k}w_{i,j}w_{i,k}x_j x_k] \tag{8}$$

$$= \frac{1}{p}\sum_{j=1}^{n} w_{i,j}^2\,\mathrm{E}[x_j^2] + \sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k]. \tag{9}$$

Note that this difference arises from the fact that $\mathrm{E}[m_{i,i}^2] = p$ and $\mathrm{E}[m_{i,i}m_{j,j}] = p^2$ for $i \neq j$. Intuitively, PostDropout uses a single mask; therefore, the multiplication of the two masks is equivalent to the original. However, PreDropout uses multiple masks, and the multiplication of the two masks results in a new mask with $\mathrm{Bernoulli}(p^2)$.

In summary, the difference between Eqs. 5 and 9 comes from the second term and is $\frac{1-p}{p}\sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k]$. Thus, we have

$$\mathrm{E}[(\mathrm{Dropout}_{\text{train}}(\mathbf{W}\mathbf{x}))^2] > \mathrm{E}[(\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x}))^2], \tag{10}$$

Accepted for the 39th Conference on Uncertainty in Artificial Intelligence (UAI 2023).

if and only if $\sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k] > 0$. This proves the first inequality $\Delta(\mathrm{Dropout}(\mathbf{W}\mathbf{x})) < \Delta(\mathbf{W}\,\mathrm{Dropout}(\mathbf{x}))$.

Next, we investigate the second inequality. We know that

$$\mathrm{Var}[\mathbf{W}\,\mathrm{Dropout}_{\text{test}}(\mathbf{x})] = \mathrm{E}[(\mathbf{W}\mathbf{x})^2] - (\mathrm{E}[\mathbf{W}\mathbf{x}])^2. \tag{11}$$

We compare these two terms with those in Eq. 2. The second term is equal to the second term in Eq. 2. Note that

$$\mathrm{E}[(\mathbf{W}\mathbf{x})^2] = \sum_{j=1}^{n} w_{i,j}^2\,\mathrm{E}[x_j^2] + \sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k] \tag{12}$$

$$< \frac{1}{p}\sum_{j=1}^{n} w_{i,j}^2\,\mathrm{E}[x_j^2] + \sum_{j=1}^{n}\sum_{k\neq j}^{n} w_{i,j}w_{i,k}\,\mathrm{E}[x_j x_k] \tag{13}$$

$$= \mathrm{E}[(\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x}))^2]. \tag{14}$$

Thus, $\mathrm{Var}[\mathbf{W}\,\mathrm{Dropout}_{\text{test}}(\mathbf{x})] < \mathrm{Var}[\mathbf{W}\,\mathrm{Dropout}_{\text{train}}(\mathbf{x})]$, proving that $\Delta(\mathbf{W}\,\mathrm{Dropout}(\mathbf{x})) < 1$. □

B PROOF OF PROPOSITION 5

Proof. GAP averages the feature map $\mathbf{x}$ in the spatial direction as $[\mathrm{GAP}(\mathbf{x})]_k = \frac{1}{HW}\sum_{i,j}^{H,W} x_{k,i,j}$. For simplicity, we regard the $i,\ j$ axis as the a axis and denote kth element of GAP from $\mathbf{x} \in \mathbb{R}^{K\times A}$ as

$$[\mathrm{GAP}(\mathbf{x})]_k = \frac{1}{A}\sum_{a}^{A} x_{k,a}, \tag{15}$$

where $A = HW$. In vector form, $\mathrm{GAP}(\mathbf{x}) = \frac{1}{A}\mathbf{x}\cdot\mathbb{1}^\top$ where $\mathbb{1} = [1,\ 1,\ \cdots,\ 1] \in \mathbb{R}^A$. Now, dropout at H4 and H5 can be, respectively, written as:

$$[\mathrm{GAP}(\mathrm{Dropout}_{\text{train}}(\mathbf{x}))]_k = \frac{1}{A}\sum_{a}\Big(\frac{1}{p}m_{k,a}x_{k,a}\Big), \tag{16}$$

$$[\mathrm{Dropout}_{\text{train}}(\mathrm{GAP}(\mathbf{x}))]_k = \frac{1}{p}m_k\Big(\frac{1}{A}\sum_{a} x_{k,a}\Big). \tag{17}$$

The two equations indicate that dropout before the GAP masks each element of the feature map $\mathbf{x}$, whereas dropout after the GAP masks each channel of the feature map $\mathbf{x}$. This property enables a similar derivation in the proof of Proposition 2 as follows:

$$\mathrm{E}[(\sum_{a} m_{k,a}x_{k,a})^2] = \mathrm{E}[\sum_{a} m_{k,a}^2 x_{k,a}^2 + \sum_{a}\sum_{b\neq a} m_{k,a}m_{k,b}x_{k,a}x_{k,b}] \tag{18}$$

$$= p\sum_{a}\mathrm{E}[x_{k,a}^2] + p^2\sum_{a}\sum_{b\neq a}\mathrm{E}[x_{k,a}x_{k,b}], \tag{19}$$

$$\mathrm{E}[(\sum_{a} m_k x_{k,a})^2] = \mathrm{E}[m_k^2(\sum_{a} x_{k,a}^2 + \sum_{a}\sum_{b\neq a} x_{k,a}x_{k,b})] \tag{20}$$

$$= p\sum_{a}\mathrm{E}[x_{k,a}^2] + p\sum_{a}\sum_{b\neq a}\mathrm{E}[x_{k,a}x_{k,b}]. \tag{21}$$

Thus, $\mathrm{E}[(\sum_a m_{k,a}x_{k,a})^2] < \mathrm{E}[(\sum_a m_k x_{k,a})^2]$ if and only if $\sum_a\sum_{b\neq a}\mathrm{E}[x_{k,a}x_{k,b}] > 0$. Note that because $\mathbf{x}$ is the output of an activation function such as ReLU or ReLU6, the condition holds. This proves that the variance at H4 is smaller than at H5. □

C DESIGN OF OTHER DROPOUT OPERATION

In our analysis, we used the properties of the Bernoulli distribution and corresponding mask matrix $\mathbf{M}$. One could think that if we design another mask matrix using a different distribution and employ it to modify dropout, then the dropout would not demonstrate variance inconsistency. For example, rather than a simple turn-off operation, using attenuation and amplification would have more degrees of freedom, which could enable us to find dropout without any inconsistency. Considering this, we seek a new general design for the Dropout operation.

Definition 3. *For an n-dimensional vector $\mathbf{x}$, we define the DropoutAr operation as:*

$$\mathrm{DropoutAr}_{\textit{train}}(\mathbf{x}) = \mathbf{A}\mathbf{x}, \tag{22}$$

$$\mathrm{DropoutAr}_{\textit{test}}(\mathbf{x}) = \mathbf{x}, \tag{23}$$

*where $\mathbf{A}$ is an $n \times n$ diagonal matrix of $a_{i,j} = 0$ for $i \neq j$ and $a_{i,j}$ is sampled from the **arbitrary** distribution for $i = j$, independent of $\mathbf{x}$.*

DropoutAr is a generalization of Dropout; Dropout is a special case of DropoutAr obtained by choosing $\mathbf{A} = \mathbf{M}/p$. For example, we could choose a Gaussian distribution to generate real numbers that serve as attenuation and amplification for each element of the vector $\mathbf{x}$. From this generalization, we investigate the form of a mask matrix that exhibits consistency in the training and test phases.

However, we find that this is unrealistic.

Proposition 6. *DropoutAr cannot exhibit both mean and variance consistencies simultaneously for any $\mathbf{A}$, except for the identity matrix.*

Proof. We prove this proposition by observing that mean consistency results in variance inconsistency. If DropoutAr has a mean consistency, we obtain $\mathrm{E}[\mathbf{A}\mathbf{x}] = \mathrm{E}[\mathbf{x}]$. Because each element of $\mathbf{A}$ is sampled independently of $\mathbf{x}$, we obtain $\mathrm{E}[\mathbf{A}] = 1$. Now, we investigate the variance of DropoutAr:

$$\begin{aligned}
\mathrm{Var}[\mathbf{A}\mathbf{x}] &= \mathrm{E}[(\mathbf{A}\mathbf{x})^2] - (\mathrm{E}[\mathbf{A}\mathbf{x}])^2 && (24)\\
&= \mathrm{E}[\mathbf{A}^2]\,\mathrm{E}[\mathbf{x}^2] - (\mathrm{E}[\mathbf{A}])^2(\mathrm{E}[\mathbf{x}])^2 && (25)\\
&= (\mathrm{Var}[\mathbf{A}] + (\mathrm{E}[\mathbf{A}])^2)\,\mathrm{E}[\mathbf{x}^2] - (\mathrm{E}[\mathbf{A}])^2(\mathrm{E}[\mathbf{x}])^2 && (26)\\
&= (\mathrm{Var}[\mathbf{A}] + 1)\,\mathrm{E}[\mathbf{x}^2] - (\mathrm{E}[\mathbf{x}])^2 && (27)\\
&> \mathrm{E}[\mathbf{x}^2] - (\mathrm{E}[\mathbf{x}])^2 && (28)\\
&= \mathrm{Var}[\mathbf{x}]. && (29)
\end{aligned}$$

Note that if $\mathbf{A}$ is not an identity matrix for $\mathrm{E}[\mathbf{A}] = 1$, then $\mathrm{Var}[\mathbf{A}] > 0$. Thus, we obtain $\mathrm{Var}[\mathbf{A}\mathbf{x}] > \mathrm{Var}[\mathbf{x}]$. Therefore, we conclude that $\Delta(\mathrm{DropoutAr}(\mathbf{x})) < 1$; hence, DropoutAr cannot exhibit variance consistency as long as it is not the identity operator. □

One could think that this simple linear equation lacks sufficient degrees of freedom. We now consider using a polynomial function.

Definition 4. *For an n-dimensional vector $\mathbf{x}$, we define the DropoutArPoly operation as:*

$$[\mathrm{DropoutArPoly}_{\textit{train}}(\mathbf{x})]_i = \sum_{k=0}^{d} a_{i,k} x_i^k, \tag{30}$$

$$\mathrm{DropoutArPoly}_{\textit{test}}(\mathbf{x}) = \mathbf{x}, \tag{31}$$

where $a_{i,k}$ is sampled from an arbitrary distribution and is independent of $\mathbf{x}$.

DropoutArPoly is a further generalization of DropoutAr. Similarly, we propose the following:

Proposition 7. *DropoutArPoly cannot exhibit both mean and variance consistencies simultaneously as long as DropoutArPoly is not the identity operator.*

Proof. We prove this proposition by observing that having both mean and variance consistencies simultaneously leads to the identity operator. First, during the training phase, the mean of DropoutArPoly is

$$\mathrm{E}[a_{i,0} + a_{i,1}x_i + a_{i,2}x_i^2 + \cdots + a_{i,d}x_i^d] = \mathrm{E}[a_{i,0}] + \mathrm{E}[a_{i,1}]\,\mathrm{E}[x_i] + \mathrm{E}[a_{i,2}]\,\mathrm{E}[x_i^2] + \cdots + \mathrm{E}[a_{i,d}]\,\mathrm{E}[x_i^d]. \tag{32}$$

For DropoutArPoly, obtaining the mean consistency on an arbitrary $\mathbf{x}$ requires

$$\mathrm{E}[a_{i,1}] = 1, \tag{33}$$
$$\mathrm{E}[a_{i,0}] = \mathrm{E}[a_{i,2}] = \cdots = \mathrm{E}[a_{i,d}] = 0. \tag{34}$$

Secondly, to investigate variance, we compute the mean of the square.

$$\mathrm{E}[(a_{i,0} + a_{i,1}x_i + a_{i,2}x_i^2 + \cdots + a_{i,d}x_i^d)^2] \tag{35}$$
$$= \mathrm{E}[a_{i,0}^2] + 2\,\mathrm{E}[a_{i,0}a_{i,1}]\,\mathrm{E}[x_i] + (2\,\mathrm{E}[a_{i,0}a_{i,2}] + \mathrm{E}[a_{i,1}^2])\,\mathrm{E}[x_i^2] \tag{36}$$
$$+ (2\,\mathrm{E}[a_{i,0}a_{i,3}] + 2\,\mathrm{E}[a_{i,1}a_{i,2}])\,\mathrm{E}[x_i^3] + (2\,\mathrm{E}[a_{i,0}a_{i,4}] + 2\,\mathrm{E}[a_{i,1}a_{i,3}] + \mathrm{E}[a_{i,2}^2])\,\mathrm{E}[x_i^4] + \cdots \tag{37}$$
$$= \mathrm{E}[a_{i,0}^2] + \mathrm{E}[a_{i,1}^2]\,\mathrm{E}[x_i^2] + \mathrm{E}[a_{i,2}^2]\,\mathrm{E}[x_i^4] + \cdots. \tag{38}$$

To obtain variance consistency, this result should be equal to $\mathrm{E}[x_i^2]$ for an arbitrary $\mathbf{x}$. This requires

$$\mathrm{E}[a_{i,1}^2] = 1, \tag{39}$$
$$\mathrm{E}[a_{i,0}^2] = \mathrm{E}[a_{i,2}^2] = \cdots = \mathrm{E}[a_{i,d}^2] = 0. \tag{40}$$

Eqs 33, 34, 39, and 40 indicate that DropoutArPoly becomes the identity operator; hence, there is no other possible operation that can satisfy both mean and variance consistencies at the same time. □

In summary, for a dropout-like operation, variance inconsistency is a universal phenomenon and is not unique to the Bernoulli distribution.

D MEAN AND VARIANCE AFTER RELU

For $x \sim \mathcal{N}(0, \sigma^2)$, we compute the mean and variance of $\mathrm{ReLU}(x)$, which are used in the main text. We use $p(x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{x^2}{2\sigma^2}\right)$ to denote the probability density function of x. First, we know that

$$\mathrm{E}[\mathrm{ReLU}(x)] = \int_{-\infty}^{\infty} \mathrm{ReLU}(x)p(x)dx \tag{41}$$
$$= \int_{0}^{\infty} xp(x)dx \tag{42}$$
$$= \frac{1}{\sqrt{2\pi}}\sigma. \tag{43}$$

The last equation can be derived using the properties of a half-normal or truncated normal distribution. Secondly, using the symmetry of $p(x)$, we derive

$$\mathrm{E}[(\mathrm{ReLU}(x))^2] = \int_{-\infty}^{\infty} (\mathrm{ReLU}(x))^2 p(x)dx \tag{44}$$
$$= \int_{0}^{\infty} x^2 p(x)dx \tag{45}$$
$$= \frac{1}{2}\int_{-\infty}^{\infty} x^2 p(x)dx \tag{46}$$
$$= \frac{1}{2}\,\mathrm{E}[x^2] \tag{47}$$
$$= \frac{1}{2}\sigma^2. \tag{48}$$

Thus, we obtain

$$\mathrm{Var}[\mathrm{ReLU}(x)] = \left(\frac{1}{2} - \frac{1}{2\pi}\right)\sigma^2 \tag{49}$$

$$= \frac{\pi - 1}{2\pi}\sigma^2. \tag{50}$$

```python
import torch

len_x = int(1e+8)
std_x = 1.0
x = torch.normal(mean=torch.zeros((len_x)), std=std_x)

ReLU = torch.nn.ReLU()
ReLU_x = ReLU(x)

print(ReLU_x.mean().item())
print(ReLU_x.var(unbiased=False).item())
```

Listing 1: PyTorch example to measure the mean and variance after ReLU.

The mean and variance after ReLU can also be empirically measured using the above Python code. We used a sufficiently large number of samples, 10^8 and $\sigma = 1$. From this code, we obtained the following:

```
0.3989197611808777
0.34079989790916443
```

Indeed, $\frac{1}{\sqrt{2\pi}} \approx 0.3989$ and $\frac{\pi-1}{2\pi} \approx 0.3408$.

E LIMITATION

One potential disadvantage of applying dropout before the GAP is that it could require an additional feature map that expands on the spatial axis, which leads to increased GPU memory consumption. However, modern libraries such as PyTorch provide an option called `inplace` that allows features to be dropped directly without generating an additional feature map. This option eliminates the potential disadvantage of applying dropout before the GAP.

F HYPERPARAMETERS FOR EXPERIMENTS

Notes on Module Tests When measuring the variance, certain libraries apply Bessel's correction by default. To ensure correct results, this feature must be turned off. For example, in PyTorch, `torch.var(input, unbiased=False)` should be used to apply a biased estimator. In fact, `unbiased=False` is specified when the BN of PyTorch computes the standard deviation.

CIFAR Dataset The CIFAR-$\{10,\ 100\}$ dataset consists of 60K images of $\{10,\ 100\}$ classes. For data augmentation, we used 32×32 random cropping with 4-pixel padding, a random horizontal flip with a probability of 0.5, and mean-std normalization using dataset statistics. For training, the number of epochs of 164, stochastic gradient descent with a momentum of 0.9, learning rate of 0.1, learning rate decay of 0.1 at $\{81,\ 122\}$ epochs, weight decay of 0.0001, mini-batch size of 128, and dropout with a keep probability of 0.8 were used.

Oxford-IIIT Pet and Caltech-101 Datasets The Oxford-IIIT Pet dataset consists of 7K pet images from 37 classes; the Caltech-101 dataset includes 9K object images from 101 classes with a background category. Each dataset was split into training, validation, and test sets at a ratio of 70:15:15. All experiments were conducted at a resolution of 224×224 using standard data augmentation, including random resized cropping to 256 pixels, random rotations within 15 degrees, color jitter with a factor of 0.4, random horizontal flip with a probability of 0.5, center cropping with 224-pixel windows, and mean-std normalization based on ImageNet statistics. To better observe the performance difference, we trained the model from scratch and did not use pretrained weights. For training, stochastic gradient descent with a momentum of 0.9, learning rate of 0.1, cosine annealing schedule with 200 iterations, weight decay of 0.002, mini-batch size of 128, and dropout with a keep probability of 0.8 were used. The model with the highest validation accuracy was obtained for 200 training epochs.

ImageNet Dataset The ImageNet dataset consists of 1.2M images for 1,000 classes. For ImageNet experiments, we used the pytorch-image-models library, which is also known as `timm`. We used the hyperparameter recipe described in the official documentation. For training, stochastic gradient descent with momentum 0.9, learning rate 0.6, epochs 240, warm-up epochs 5, warm-up learning rate 10^{-5}, cosine annealing schedule, weight decay 10^{-4}, label smoothing 0.1, random erasing with probability 0.4 and count 3, RandAugment of magnitude 7 and noise-std 0.5 with increased severity (rand-m7-mstd0.5-inc1), and dropout with a keep probability of 0.8 were used.

G ADDITIONAL EXPERIMENTAL RESULTS

Dropout at Two Positions One could attempt to apply two dropouts in the residual block to combine the advantages of (P5, P6, or P7). However, according to Proposition 1, applying dropout at both P5 and P6 results in $\mathrm{Dropout}(\mathrm{ReLU}(\mathrm{Dropout}(x))) = \mathrm{Dropout}(\mathrm{ReLU}(x))$, which is meaningless. We experimented by applying dropout at both P6 and P7 and observed that for PreResNet-$\{50,\ 110\}$, the accuracy was $\{93.4867,\ 94.1833\}\%$ for CIFAR-10 and $\{72.04,\ 73.6667\}\%$ for CIFAR-100, which is worse than applying one dropout.

On Width We discussed that the advantage of PreDropout comes from the weight condition, which is intensified by the width. Here, we experimented with different widths to test whether a small width could improve the accuracy. We used WideResNet with a depth of 28. We varied the widen factor k that determines the number of channels $\{16,\ 16k,\ 32k,\ 64k\}$ for each stage. For $k = 1,\ 2,\ 5,\ 10$, we observed improved accuracy from dropout (Table 1), which implies that the widen factor 1 is sufficient to realize an advantage from dropout.

Table 1: Experimental results on WideResNet-{Depth}-{Width} with varying width.

	WideResNet-28-10		WideResNet-28-5		WideResNet-28-2		WideResNet-28-1	
	Accuracy	Difference	Accuracy	Difference	Accuracy	Difference	Accuracy	Difference
No Dropout	96.1667	-	95.7733	-	94.9100	-	93.0433	-
Guideline 1	96.2433	(+0.0767)	96.1233	(+0.3500)	94.9367	(+0.0267)	93.2433	(+0.2000)

Regression Task Although our main experiments focused on image classification tasks, our findings on the position of Dropout are not restricted to image classification and are expected to be seamlessly applicable to other tasks, including regression tasks. Here, we target a regression task using the AgeDB dataset. The AgeDB dataset contains 16K images with corresponding ages from 1 to 101. We use images as input and ages as labels to formulate regression tasks. For training, PreResNet-101, stochastic gradient descent with momentum 0.9, learning rate 0.001, epochs 200, mini-batch size 128, and weight decay 0.1 were used. For Dropout, we chose the position before the last weight layer in the residual branch. We measured the mean absolute error (MAE) on validation and test sets. We observed that following Guideline 1 improved the regression performance.

Table 2: Experimental results on AgeDB dataset.

MAE	No Dropout	With Dropout*	Difference
Val	6.1261	6.0272	-0.0989
Test	6.3082	6.2291	-0.0791