ELEG 5491 HW2

Zhizhong Li
Mar 2017
Information Engineering, CUHK
`1155070507, lz015@ie.cuhk.edu.hk`

1 Problem 1

Let $\mathbf{u}_t := \mathbf{W}_{hz}\mathbf{h}_t + \mathbf{b}_z$. Then $\mathbf{z}_t = \mathrm{softmax}(\mathbf{u}_t)$. Using formula for softmax, we have

$$\frac{\partial z_{t,i}}{\partial u_{t,j}} = \mathbb{I}(i=j)z_{t,j} - z_{t,i}z_{t,j}. \tag{1}$$

$$\frac{\partial u_{t,l}}{\partial W_{hz,ij}} = \frac{\partial\big(\sum_s W_{hz,ls}h_{t,s} + b_{z,l}\big)}{\partial W_{hz,ij}} = \mathbb{I}(i=l)h_{t,j} \tag{2}$$

$$\frac{\partial u_{t,l}}{\partial h_{t,k}} = \frac{\partial\big(\sum_s W_{hz,ls}h_{t,s} + b_{z,l}\big)}{\partial h_{t,k}} = W_{hz,lk} \tag{3}$$

$$\frac{\partial L}{\partial z_{t,i}} = -\frac{\mathbb{I}(i=y_t)}{z_{t,i}} \tag{4}$$

Use these equations, we have

$$\begin{aligned}
\frac{\partial L}{\partial W_{hz,ij}} &= \sum_t \frac{\partial L}{\partial \mathbf{z}_t}\frac{\partial \mathbf{z}_t}{\partial W_{hz,ij}} \\
&= \sum_{t,k} \frac{\partial L}{\partial z_{t,k}}\frac{\partial z_{t,k}}{\partial W_{hz,ij}} \\
&= \sum_{t,k,l} \frac{\partial L}{\partial z_{t,k}}\frac{\partial z_{t,k}}{\partial u_{t,l}}\frac{\partial u_{t,l}}{\partial W_{hz,ij}} \\
&= \sum_{t,k} \frac{\partial L}{\partial z_{t,k}}\frac{\partial z_{t,k}}{\partial u_{t,i}}h_{t,j} \\
&= \sum_t -\frac{h_{t,j}}{z_{t,y_t}}\frac{\partial z_{t,y_t}}{\partial u_{t,i}}
\end{aligned} \tag{5}$$

Now compute derivatives for $\mathbf{W}_{hh}$. Let $\mathbf{v}_t = \mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h$. Then $\mathbf{h}_t = \tanh(\mathbf{v}_t)$.

$$\frac{\partial h_{t,k}}{\partial v_{t,i}} = \mathbb{I}(k=i)(1-h_{t,k}^2) \tag{6}$$

$$
\begin{aligned}
\frac{\partial h_{t+1,i}}{\partial h_{t,k}} &= \frac{\partial h_{t+1,i}}{\partial v_{t,i}} \frac{\partial v_{t,i}}{\partial h_{t,k}} \\
&= \frac{\partial h_{t+1,i}}{\partial v_{t,i}} \frac{\partial \big((\mathbf{W}_{xh}\mathbf{x}_{t+1})_i + \sum_s W_{hh,is} h_{t,s} + b_{h,i} \big)}{\partial h_{t,k}} \\
&= \frac{\partial h_{t+1,i}}{\partial v_{t,i}} W_{hh,ik}.
\end{aligned} \tag{7}
$$

$$
\begin{aligned}
\frac{\partial L}{\partial h_{t,k}} &= \frac{\partial L}{\partial \mathbf{z}_t} \frac{\partial \mathbf{z}_t}{\partial h_{t,k}} + \frac{\partial L}{\partial \mathbf{h}_{t+1}} \frac{\partial \mathbf{h}_{t+1}}{\partial h_{t,k}} \\
&= \sum_s \frac{\partial L}{\partial z_{t,s}} \frac{\partial z_{t,s}}{\partial h_{t,k}} + \sum_i \frac{\partial L}{\partial h_{t+1,i}} \frac{\partial h_{t+1,i}}{\partial h_{t,k}} \\
&= \sum_{s,l} \frac{\partial L}{\partial z_{t,s}} \frac{\partial z_{t,s}}{\partial u_{t,l}} \frac{\partial u_{t,l}}{\partial h_{t,k}} + \sum_i \frac{\partial L}{\partial h_{t+1,i}} \frac{\partial h_{t+1,i}}{\partial h_{t,k}} \\
&= \sum_l -\frac{W_{hz,lk}}{z_{t,y_t}} \frac{\partial z_{t,y_t}}{\partial u_{t,l}} + \sum_i \frac{\partial L}{\partial h_{t+1,i}} \frac{\partial h_{t+1,i}}{\partial v_{t,i}} W_{hh,ik}
\end{aligned} \tag{8}
$$

$$
\frac{\partial v_{t,l}}{\partial W_{hh,ij}} = \frac{\partial \big((\mathbf{W}_{xh}\mathbf{x}_t)_l + \sum_s W_{hh,ls} h_{t-1,s} + b_{h,l} \big)}{\partial W_{hh,ij}} = \mathbb{I}(i=l) h_{t-1,j} \tag{9}
$$

$$
\begin{aligned}
\frac{\partial L}{\partial W_{hh,ij}} &= \sum_t \frac{\partial L}{\partial \mathbf{h}_t} \frac{\partial \mathbf{h}_t}{\partial W_{hh,ij}} \\
&= \sum_{t,k} \frac{\partial L}{\partial h_{t,k}} \frac{\partial h_{t,k}}{\partial W_{hh,ij}} \\
&= \sum_{t,k,l} \frac{\partial L}{\partial h_{t,k}} \frac{\partial h_{t,k}}{\partial v_{t,l}} \frac{\partial v_{t,l}}{\partial W_{hh,ij}} \\
&= \sum_{t,k} \frac{\partial L}{\partial h_{t,k}} \frac{\partial h_{t,k}}{\partial v_{t,i}} h_{t-1,j}
\end{aligned} \tag{10}
$$

2 Problem 2

2.1

$$
\begin{aligned}
\mathbf{h}_t &= F_\theta(\mathbf{h}_{t-1}, \mathbf{x}_t) \\
&= F_\theta(F_\theta(\mathbf{h}_{t-2}, \mathbf{x}_{t-1}), \mathbf{x}_t) \\
&= F_\theta(F_\theta(F_\theta(\mathbf{h}_{t-3}, \mathbf{x}_{t-2}), \mathbf{x}_{t-1}), \mathbf{x}_t) \\
&= F_\theta(F_\theta(F_\theta(F_\theta(\mathbf{h}_{t-4}, \mathbf{x}_{t-3}), \mathbf{x}_{t-2}), \mathbf{x}_{t-1}), \mathbf{x}_t) \\
&= \cdots\cdots \text{(unfold until reach } \mathbf{x}_1\text{)}
\end{aligned} \tag{11}
$$

2.2

(1) RNN shares parameter. This enables handling sequences of different length and improves the generalization power of the model at the same time.

(2) RNN compresses the information of whole history into one hidden state variable. This avoids the exponential growth of model complexity.

3 Problem 3

Stage 3, stage 6 and stage 7. This is because computing the next layer needs parameters of previous layer which are located on another card.

4 Problem 4

4.1

Denote the i-th hidden layer by Hidden$_i$. Then the number of fc parameters:

- Input $\rightarrow$ Hidden$_1$: dn.
- Hidden$_i$ $\rightarrow$ Hidden$_{(i+1)}$: n^2, $i = 1, 2, \dots, n-1$.
- Hidden$_L$ $\rightarrow$ Output: nc.

Total: $dn + (L-1)n^2 + nc$.

4.2

Suppose weights from layer i to $i+1$ is $W \in \mathbb{R}^{n\times m}$ and we have $y = f(Wx)$ where $x \in \mathbb{R}^m, y \in \mathbb{R}^n$, and f is an element-wise function. Suppose u is the last node to compute the cost function. Then

$$\begin{aligned} \frac{\partial u}{\partial W_{ij}} &= \sum_k \frac{\partial u}{\partial y_k}\frac{\partial y_k}{\partial W_{ij}} \\ &= \sum_k \frac{\partial u}{\partial y_k} f'((Wx)_k)\frac{\partial(\sum_l W_{kl}x_l)}{\partial W_{ij}} \\ &= \frac{\partial u}{\partial y_i} f'((Wx)_i)x_j \end{aligned} \tag{12}$$

We ignore the multiplication Wx since this result should already be available when doing forward pass. Then for the whole matrix, we need $2mn$ multiplications. Next we estimate the computation burden for the term $\partial u/\partial x_i$.

$$\begin{aligned} \frac{\partial u}{\partial x_i} &= \sum_k \frac{\partial u}{\partial y_k}\frac{\partial y_k}{\partial x_i} \\ &= \sum_k \frac{\partial u}{\partial y_k} f'((Wx)_k)\frac{\partial(\sum_l W_{kl}x_l)}{\partial x_i} \\ &= \frac{\partial u}{\partial y_i} f'((Wx)_i)W_{ki} \end{aligned} \tag{13}$$

Since $x \in \mathbb{R}^m$, the number of multiplications are $2m$.

Then we can compute the number of multiplication when BP:

- Loss $\rightarrow$ Output: do not count this in.
- Output $\rightarrow$ Hidden$_L$: for fc weights: $2nc$. for neurons in Hidden$_L$: $2n$.
- Hidden$_{(i+1)}$ $\rightarrow$ Hidden$_i$: for fc weights: $2n^2$. for neurons in Hidden$_i$: $2n$.
- Hidden$_1$ $\rightarrow$ Input: for fc weights: $2nd$.

Total: $2nc + 2n + (L-1)(2n^2 + 2n) + 2nd = 2n[1 + c + d + (L-1)(n+1)]$.

4.3

The computation fc weights are the same as the previous sub-question. However, we need to estimate computation complexity for computing gradients on neurons $\partial u_N / \partial u_i$. The number of paths is $cn^{(N-i-1)}$ and for each path, there are $n-2$ multiplications. So

- Loss $\rightarrow$ Output: do not count this in.
- Output $\rightarrow$ Hidden$_L$: for fc weights: $2nc$. for neurons in Hidden$_L$: cn.
- Hidden$_{(i+1)}$ $\rightarrow$ Hidden$_i$: for fc weights: $2n^2$. for neurons in Hidden$_i$: $(L-i+1)cn^{L-i}$.
- Hidden$_1$ $\rightarrow$ Input: for fc weights: $2nd$.

Total:

$$\begin{aligned} & 2nc + cn + 2n^2 + 2dn + \sum_{i=1}^{L-1}(L-i+1)cn^{L-i} \\ & = n(3c+2d+2n) + c\sum_{i=1}^{L-1}(i+1)n^i \end{aligned} \tag{14}$$

5 Problem 5

We implement all networks using Parrots. Codes and training logs can be found at https://github.com/innerlee/ELEG5491 (Access restricted to campus networks).

5.1

The training curve is shown in Figure 1. The learned filters is shown in Figure 2. Comparison between weight decay and no weight decay is shown in Figure 3.

5.2

Since w_l has zero mean, so does $w_l x_l$. Together with independence between x_l and w_l, we have

$$\begin{aligned} \mathrm{Var}[y_l] &= n_l \mathrm{Var}[w_l x_l] \\ &= n_l E[w_l^2 x_l^2] - 0 \\ &= n_l E[w_l^2] E[x_l^2] \\ &= n_l \mathrm{Var}[w_l] E[x_l^2]. \end{aligned} \tag{15}$$

To prove $E[x_l^2] = Var[y_{l-1}]/2$, we need assume that y_{l-1} is Gaussian with zero mean. Since

$$x_l = \begin{cases} y_l, & \text{if } y_l \geq 0 \\ 0,, & \text{if } y_l < 0 \end{cases} \tag{16}$$

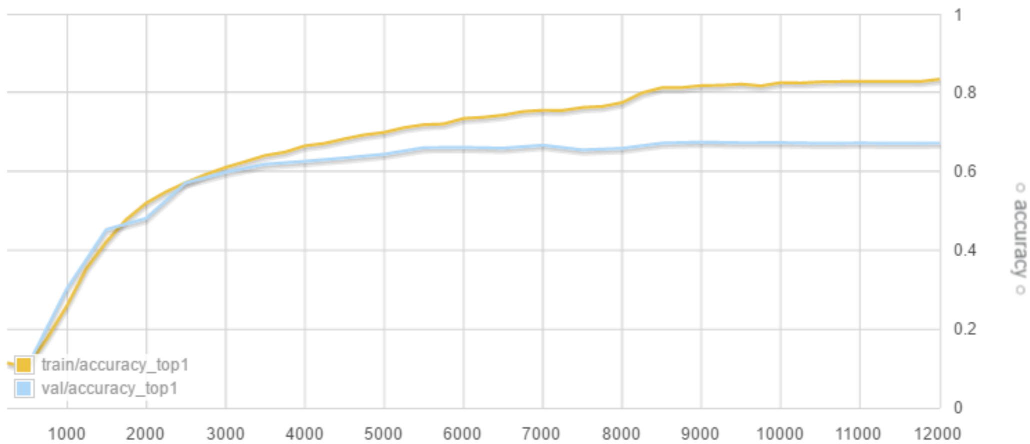
So

$$E[x_l^2] = \int_0^\infty p(x_l) x_l^2 dx_l = \frac{1}{2}\int_{-\infty}^{\infty} p(x_l) x_l^2 dx_l = \frac{1}{2} Var[y_{l-1}]. \tag{17}$$

The initialization described here is the *msra* initialization [1]. We use msra to initialize fc layers. The result is shown in Figure 4.

5.3

Adding BN, the training is much faster. The final accuracy also improves. See Figure 5.



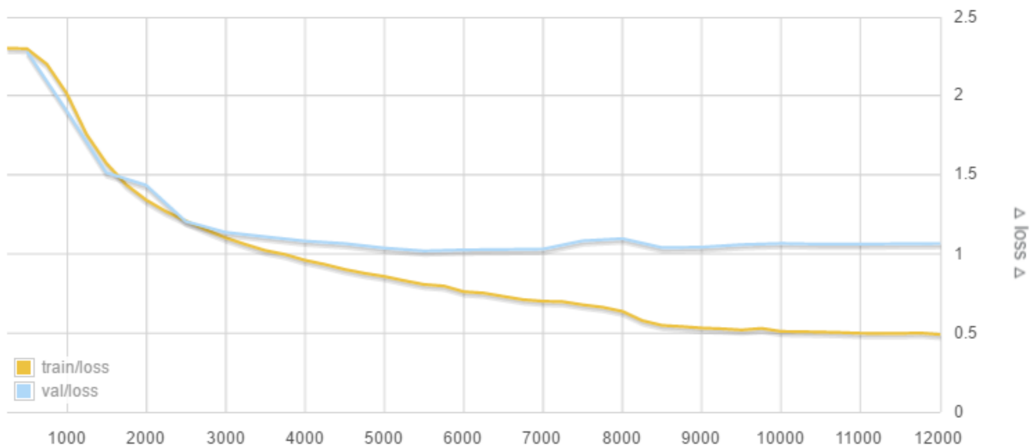


Figure 1: Training curve of the simple network on CIFAR-10. initialization is Gaussian. The batch size is 256. Use SGD as described in the problem. The step sizes are 8000, 10000 and 12000.

5.4

We add augmentation as follows: Pad 4 pixels around the image and randomly crop to size 32×32. The result is shown in Figure 6. *Note*: the implementation of padding and random crop is using Parrots' code. Not implemented by self.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015.

[2] X. Wang. Assignments, 2017. URL `http://dl.ee.cuhk.edu.hk/`.

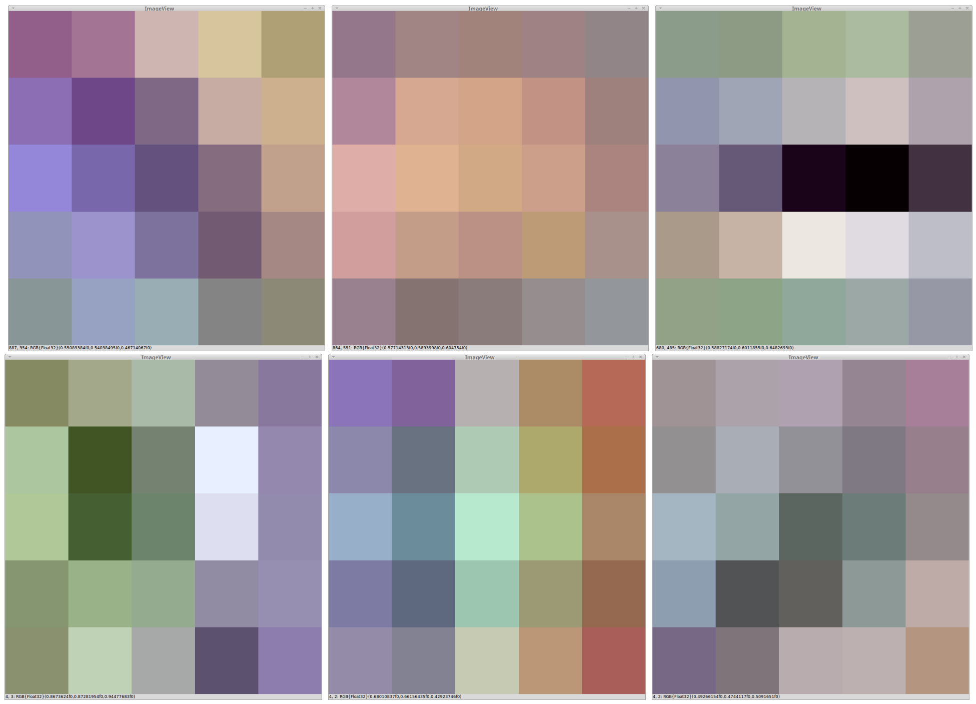


Figure 2: Visualize the six filters of the first convolutional layer. Since the input channel is 3, we convert the filters to RGB color images.

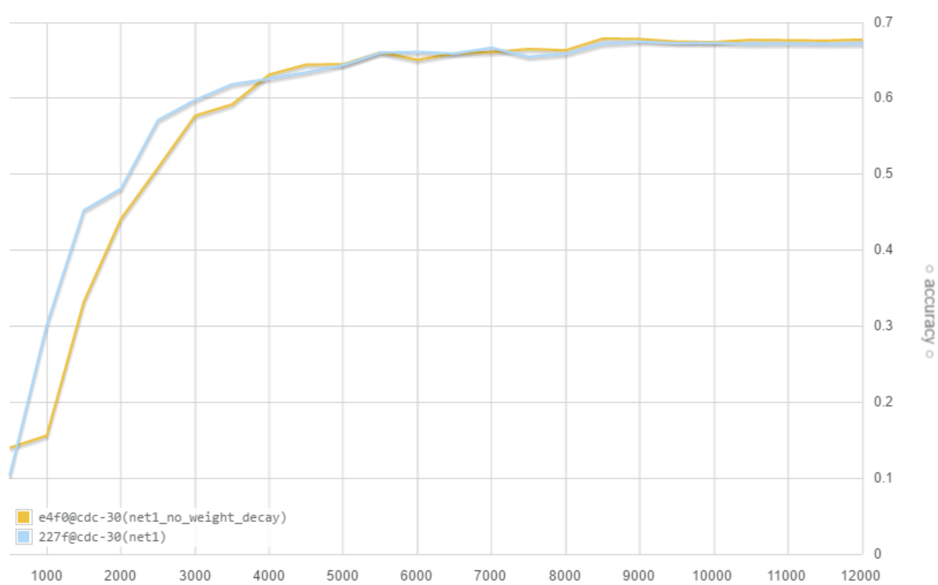


Figure 3: With or without weight decay. We can see that they do not have significant difference. The one with weight decay has best val accuracy 67.44%. The one without weight decay has best val accuracy 67.86%.

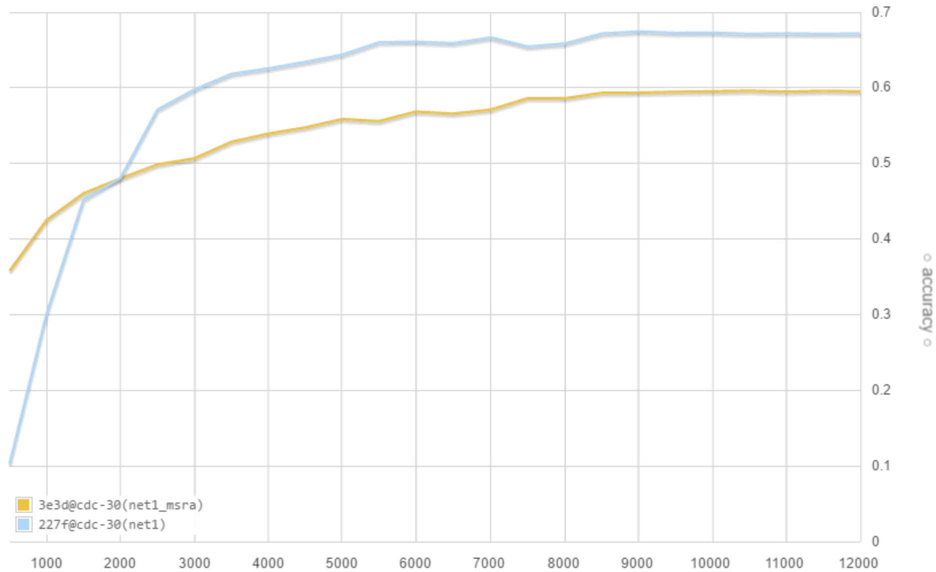


Figure 4: Gaussian initialization v.s. msra initialization. We can see that msra is worse in this experiment. The one with Gaussian has best val accuracy 67.44%. The one with msra has best val accuracy 59.61%.

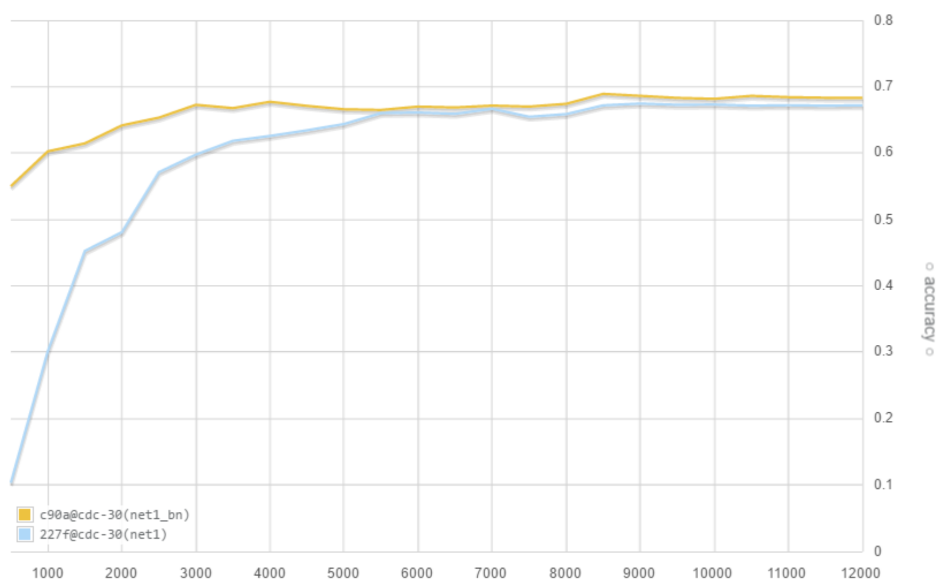


Figure 5: Val accuracy with or without BN. We can see that BN is very effective in improving training speed and test accuracy. The one without BN has best val accuracy 67.44%. The one with BN has best val accuracy 68.91%.

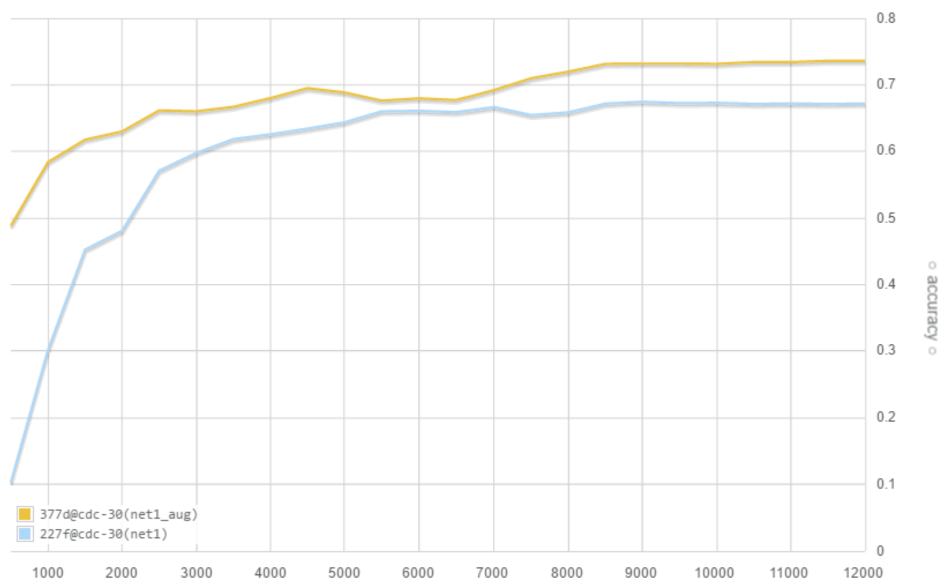


Figure 6: Val accuracy with or without augmentation. We can see that augmentation is very effective in improving accuracy. The one without BN has best val accuracy 67.44%. The one with BN has best val accuracy 73.61%.