

ROBUST CONVOLUTIONAL NEURAL NETWORKS UNDER ADVERSARIAL NOISE

Jonghoon Jin, Aysegul Dundar and Eugenio Culurciello

Electrical and Computer Engineering, Purdue University
Biomedical Engineering, Purdue University
{jhjin, adundar, euge}@purdue.edu

ABSTRACT

Recent studies have shown that Convolutional Neural Networks (CNNs) are vulnerable to a small perturbation of input called “adversarial examples”. In this work, we propose a new feedforward CNN that improves robustness in the presence of adversarial noise. Our model uses stochastic additive noise added to the input image and to the CNN models. The proposed model operates in conjunction with a CNN trained with either standard or adversarial objective function. In particular, convolution, max-pooling, and ReLU layers are modified to benefit from the noise model. Our feedforward model is parameterized by only a mean and variance per pixel which simplifies computations and makes our method scalable to a deep architecture. From CIFAR-10 and ImageNet test, the proposed model outperforms other methods and the improvement is more evident for difficult classification tasks or stronger adversarial noise.

1 INTRODUCTION

Convolutional neural networks (CNNs) (LeCun et al., 1998) have shown great success in visual and semantic understanding. They have been applied to solve visual recognition problems, where hard-to-describe objects or multiple semantic concepts are present in images.

Given the global widespread use of cameras on mobile phones, CNNs are already a candidate to perform categorization of user photos. Device manufacturers use various types of cameras, each with very different sensor noise statistics (Tian, 2000). Also, recent phone cameras can record a video at hundreds of frames per second, where more frames per second translates into higher image noise (Tian, 2000). Unfortunately, CNNs are vulnerable to artificial noise and could be easily fooled by the noise of just few pixels (Szegedy et al., 2013; Goodfellow et al., 2014; Nguyen et al., 2014). This problem arises because standard CNNs are discriminative models. This work provides a solution to improve instability of CNNs, for example in security applications.

While Bishop (1995) showed that training with noise is equivalent to a regularizer, Goodfellow et al. (2014); Huang et al. (2015) used adversarial perturbation during training but not has been applied to natural images or more challenging image classification tasks. Similarly, Gu & Rigazio (2014) proposed a denoising model for adversary using auto-encoders.

The main contribution of this work is to propose a robust feedforward CNN model under adversarial noise; a noise that affects the performance the most. In order to achieve this, we add stochasticity to the CNN models with the assumption that the perturbation can be seen as a sample drawn from a white Gaussian noise. Our model takes advantage of a parametric model, which makes our method possible to scale up and apply to a large-scale dataset such as ImageNet.

2 CONVOLUTIONAL NEURAL NETWORKS WITH NOISE MODEL

The proposed feedforward model uses a noise distribution applied to each pixel. The following subsections explain the stochastic operation of each layer, including convolution, pooling, and non-linear activation. Also, the rest of operators used in standard CNNs can be found in appendix B.

2.1 INPUT NOISE MODEL

We add an uncertainty to input images so the CNN output in hyperspace becomes a cloud with uncertainty information instead of a vector. We hypothesize that referring to the marginal data during classification helps CNNs to be robust toward adversarial examples. As a result of our modeling, each pixel becomes a random variable X in \mathbb{R}^3 (channel, height, width) and follows a normal distribution with the mean of the original pixel value $\mu_{X_{ijk}}$ and a constant variance of σ_N^2

$$X_{ijk} \triangleq \mu_{X_{ijk}} + N \implies X_{ijk} \sim \mathbf{N}(\mu_{X_{ijk}}, \sigma_N^2) \quad (1)$$

where all input pixels have the same noise power of σ_N^2 . The conditional independence of noise, for given value $\mu_{X_{ijk}}$, among pixels in neighborhood helps us to simplify the model and make it scalable to deep networks. To clarify, we adopted the artificial noise distribution in order to improve the robustness of CNNs and it is unrelated to natural image statistics.

2.2 CONVOLUTION LAYER

While CNN inputs are modeled as random variables, all remaining parameters such as weights and biases are fixed constants. Convolution is a weighted sum of random variables and its first and second order moments of output of convolution layer are shown in equation 2.

$$E[Y] = \sum \omega E[X] + b, \quad Var[Y] = \sum \omega^2 Var[X] \quad (2)$$

X and Y corresponds to a single element of the input and output in the convolution layer respectively. ω and b are weights and biases, and pixel index i, j and k are omitted for conciseness. We are interested in the first and second order statistics since we want to stay with a parametric model throughout layers, which simplifies overall computations.

2.3 RECTIFIED LINEAR UNIT LAYER

Rectified linear units (ReLU) (Krizhevsky et al., 2012) applies the non-linearity $Y = \max(X, \theta)$ in an element-wise manner. Then, as illustrated in appendix A, the distribution of the stochastic ReLU output Y is left-censored where $Y = X$ for $X > \theta$ otherwise reported as a single value θ . The mean and variance (Greene, 2008) of output Y for the given normal distribution of input X are:

$$\begin{aligned} E[Y] &= E[Y|X = \theta]Pr(Y = \theta|X) + E[Y|X > \theta]Pr(Y > \theta|X) \\ &= \theta\Phi(\alpha) + (\mu_X + \sigma_X\lambda(\alpha))(1 - \Phi(\alpha)) \\ Var[Y] &= E_X[Var[Y|X]] + Var_X[E[Y|X]] \\ &= \sigma_X^2(1 - \Phi(\alpha))[(1 - \delta(\alpha)) + (\alpha - \lambda(\alpha))^2\Phi(\alpha)] \end{aligned} \quad (3)$$

where $\delta(\alpha) = \lambda(\alpha)(\lambda(\alpha) - \alpha)$, $\lambda(\alpha) = \phi(\alpha)/(1 - \Phi(\alpha))$, a standard score $\alpha = (\theta - \mu_X)/\sigma_X$, a standard normal density of ϕ and a cumulative normal distribution of Φ are used. Outputs from the convolution layer followed by non-linearity are reasonably approximated to be Gaussians by the central limit theorem considering that an output neuron has more than few hundreds of connections in general. Stochastic ReLU operator allows to deliver tail information of the distribution to the higher layer of CNNs regardless of the neuron's activation, which is expected to contribute better decision making.

2.4 MAX-POOLING LAYER

Prediction from stochastic max-pooling is calculated based on the exact distribution of the max of two normal distributions (Nadarajah & Kotz, 2008) whose variables are sampled from a set S with elements in the pooling window. The pairwise max operation in the equation 4 is iteratively applied until no element left in the set S .

$$\begin{aligned} E[Y] &= \mu_{X_i}\Phi(\alpha) + \mu_{X_j}\Phi(-\alpha) + \theta\phi(\alpha) \\ Var[Y] &= (\sigma_{X_i}^2 + \mu_{X_i}^2)\Phi(\alpha) + (\sigma_{X_j}^2 + \mu_{X_j}^2)\Phi(-\alpha) + (\mu_{X_i} + \mu_{X_j})\theta\phi(\alpha) - E[Y]^2 \end{aligned} \quad (4)$$

where $\alpha = \frac{(\mu_{X_i} - \mu_{X_j})}{\theta}$, $\theta = \sqrt{\sigma_{X_i}^2 + \sigma_{X_j}^2}$. By approximating the output to a normal parametric distribution (see appendix A), we trade off accuracy for the sake of scalability of this method. According to Sinha et al. (2007) and appendix E, the ordering of iterative max operation should be set in an ascending order by their means to minimize approximation error.

Table 1: Classification accuracy on CIFAR-10 and ImageNet validation sets. The adversarial training with high noise on ImageNet failed to converge, which is marked with an asterisk. The zero adversarial noise intensity corresponds to the original validation set. A different input variance σ_N^2 in stochastic model was used for each level of adversarial noise.

Adversarial noise intensity (k) [px]	CIFAR-10 (%)		ImageNet (%)		
	0	0.5	0	0.01	0.5
Standard training (baseline)	90.1	72.3	57.0	56.1	24.8
Standard training + Stochastic FF	88.9	78.1	57.0	56.2	33.4
LWA + BN (Huang et al., 2015)	89.0	82.3	—	—	—
Adversarial training (Goodfellow et al., 2014)	88.7	82.1	43.0	42.9	*
Adversarial training + Stochastic FF	88.7	82.9	43.0	42.9	*

3 EXPERIMENTAL RESULTS

We denote the proposed method as “stochastic feedforward (FF)” throughout the section. The stochastic FF was applied to the Network-in-Network (NIN) (Lin et al., 2013) and the single column AlexNet (Krizhevsky, 2014) with the latest technique including dropout and batch normalization (Ioffe & Szegedy, 2015). The networks were trained with either standard or adversarial training (Goodfellow et al., 2014) for comparison. Their performance under adversarial noise was evaluated on CIFAR-10 and ImageNet classification datasets (Krizhevsky & Hinton, 2009; Russakovsky et al., 2014). The gradient sign method (Goodfellow et al., 2014) was used to generate adversarial examples that are more likely to appear in natural environment.

The table 1 summarizes the best classification results obtained from our experiments. In the presence of adversarial noise, the stochastic FF provides extra accuracy gain regardless of training methods at the cost of little accuracy drop in normal configuration — no added noise. The gain is more visible when the task is difficult or the noise is stronger. In the ImageNet test, adversarial training converges to a lower accuracy than the standard training, but also the adversarial training with a noise intensity higher than 0.1 fails to reach to a meaningful score; better than random guessing. Considering a byte-precision pixel range $[0, 255]$, the adversarial training is limited for use in a hard task whereas the stochastic FF combined with standard training is still effective in high noise condition.

CNNs’ decision boundary is constructed based on sparsely populated training samples (Nguyen et al., 2014) in a high-dimension. Adversarial examples used in this experiment are populated around the decision boundary, therefore, they are often indistinguishable from natural images if one uses point-wise prediction. In the stochastic FF, uncertainty around the input pixel is propagated throughout every layer of CNNs and provides marginal information. Instead of point-wise prediction, integrating such information increases a chance to make correct prediction for adversarial examples. Adding stronger noise drags the adversarial examples farther apart from the correct decision region thereby lowering the accuracy as in appendix D.

A downside of the stochastic FF is accuracy loss due to mistuned input variance or numerical instability. For example, an input distribution with high variance makes it more likely to be uniform where the ambiguity causes performance degradation. Users will have to make a choice whether to prefer small loss of classification for large gain in the presence of noise. For near-zero variance, each pixel distribution is shaped to a Dirac delta function. Without uncertainty, the model is equivalent to the standard feedforward while near-zero division causes numerical instability.

The stochastic model may be ensembled with the standard CNN model in order to compensate its weakness under the absence of adversarial noise. the ensemble model on ImageNet is more robust to adversarial noise than the baseline model by 13.12% but with only 0.28% of accuracy loss under normal configuration (appendix D).

4 CONCLUSION

We present new feedforward CNN with stochastic input model that is robust to the adversarial noise. The proposed model outperforms other methods in the literature and the accuracy gain becomes more evident for difficult classification task or stronger adversarial noise. Our model takes advantage of a parametric model, which makes our method scalable to a deep architecture like AlexNet. This work provides a solution how to overcome CNNs’ sensitivity to the adversarial noise so as to avoid potential security problems in CNN applications.

REFERENCES

- Chris M Bishop. Training with noise is equivalent to tikhonov regularization. *Neural computation*, 7(1):108–116, 1995.
- Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014.
- William H Greene. *Econometric analysis*. Granite Hill Publishers, 2008.
- Shixiang Gu and Luca Rigazio. Towards deep neural network architectures robust to adversarial examples. *arXiv preprint arXiv:1412.5068*, 2014.
- Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.
- Ruitong Huang, Bing Xu, Dale Schuurmans, and Csaba Szepesvári. Learning with a strong adversary. *arXiv preprint arXiv:1511.03034*, 2015.
- Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.
- Alex Krizhevsky. One weird trick for parallelizing convolutional neural networks. *arXiv preprint arXiv:1404.5997*, 2014.
- Alex Krizhevsky and Geoffrey Hinton. Learning multiple layers of features from tiny images. *Computer Science Department, University of Toronto, Tech. Rep*, 2009.
- Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pp. 1097–1105, 2012.
- Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013.
- Saralees Nadarajah and Samuel Kotz. Exact distribution of the max/min of two gaussian random variables. *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, 16(2):210–212, 2008.
- Anh Nguyen, Jason Yosinski, and Jeff Clune. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *arXiv preprint arXiv:1412.1897*, 2014.
- Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*, 2014.
- Debjit Sinha, Hai Zhou, and Narendra V Shenoy. Advances in computation of the maximum of a set of gaussian random variables. *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, 26(8):1522–1533, 2007.
- Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013.
- Hui Tian. *Noise analysis in CMOS image sensors*. PhD thesis, Citeseer, 2000.

APPENDIX TO
 ROBUST CONVOLUTIONAL NEURAL NETWORKS
 UNDER ADVERSARIAL NOISE

A INPUT-OUTPUT DISTRIBUTION

An example in the figure 1a illustrates the input and output distributions from a single neuron in the ReLU layer, where the stem at θ indicates point mass probability for the deactivated area with respect to the threshold θ . The output from the ReLU layer is a censored distribution whose approximation causes error during feedforward computation.

The figure 1b illustrates the input, output and approximated output distributions from max-pooling layer simulated with only two variables. The output distribution is bell-shaped, but its mode is inclined to the right. When means of the two inputs are farther, the resulting distribution is likely to be a normal distribution. In other words, the error between the exact distribution $P(Y)$ and its approximated Gaussian $P(\hat{Y})$ tends to increase as the difference of means increases.

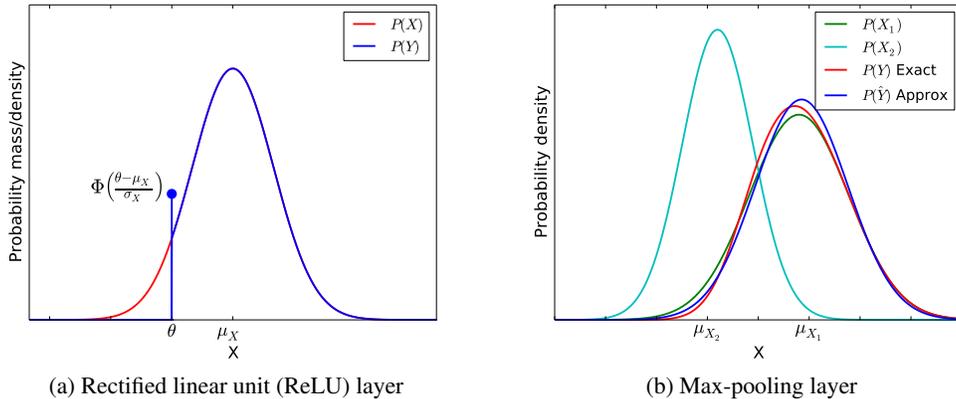


Figure 1: Behavior of ReLU and max-pooling layer with an example of two stochastic input random variables. All input distributions are assumed to be normally distributed. (a) The area $\Phi(\cdot)$ under the red curve between $(-\infty, \theta]$ is left-censored where its area is reported as a probability mass of Y at the threshold θ . (b) The curves in red and blue illustrate the exact distribution $P(Y)$ of the max of two input random variables denoted as X_1, X_2 centered at μ_{X_1}, μ_{X_2} , and its normal approximation $P(\hat{Y})$ calculated from equation 4.

B OTHER STOCHASTIC OPERATIONS

Along with the main operations previously discussed, standard CNNs consist of other modules such as batch normalization (Ioffe & Szegedy, 2015), spatial average pooling (Lin et al., 2013), softmax (or log-softmax) and dropout (Hinton et al., 2012).

The spatial average pooling and the batch normalization are linear functions and they can be processed with the convolution layer model. The average pooling used in NIN architecture is equivalent to convolution layer whose weights and biases are replaced with averaging coefficients ($1/n$) and zeroes respectively. Also, the net operation of batch normalization in evaluation phase is an affine transformation (equation 5) where γ and β are constants learned from training, therefore, the same convolution modeling can be applied to here without extra approximation.

$$Y = \gamma \frac{(X - \mu_X)}{\sqrt{\sigma_X^2 + \epsilon}} + \beta \quad (5)$$

The softmax with deterministic input produces pseudo-probabilities whose highest activation predicts class category. The proposed method adopts Gaussians to model intermediate representations

throughout the network. The strongest activation among all class distributions can be easily distinguished by the mean values of Gaussians. Therefore, we process mean values without variances as like in the normal softmax layer. Dropout neurons are simply deactivated and work as identity functions during evaluation.

C PARAMETER TUNING

C.1 ADVERSARIAL EXAMPLES

Prior to the experiment, adversarial examples are being generated through the fast gradient sign method proposed in Goodfellow et al. (2014). The method generates adversarial noise on top of natural images whose direction is toward the opposite of a gradient. Though the difference between the original and adversarial sample is imperceptible to human eyes, the perturbation makes the samples cross the decision boundary therefore classified as different categories. The noise can be interpreted as an exceptional type of noise although such examples are hardly observed in the natural environment.

Considering that the pixels encoded in an 8-bit image are positive integers in the range of $[0, 255]$, the smallest and effective pixel intensity of the sign should be a multiple of $1/\sigma_C$ where σ_C is a standard deviation used for channel-wise normalization during data preprocessing. We denote k_{adv}/σ_C as a normalized intensity of adversarial noise and only tune the pixel intensity k_{adv} throughout all experiments. However, we used a continuous range along with effective intensities in byte representation so that input-output relation is more apparent and easily observable.

C.2 INPUT VARIANCE TUNING

Upon the creation of stochastic input model, we need to choose a variance for input distributions. The uncertainty variable (or input variance) is the only parameter in this model. The stochastic input is artificial modeling and it is designed to take into account the possible range of adversarial noise. Therefore, it needs to be tuned differently based on the intensity of adversarial noise.

Using very small variance for input makes the method mathematically equivalent to the baseline CNN model. However, both ReLU and max-pooling layer with the stochastic input model requires division, which is exposed to numerical instability from near-zero denominators. We found ϵ of $1e^{-20}$ minimized the numerical error. The number is adopted for regularizing the denominators both on CIFAR-10 and ImageNet.

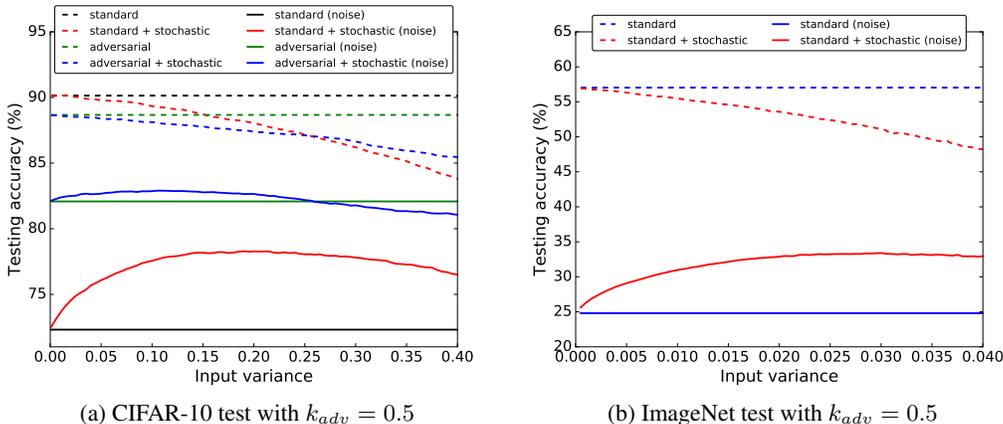


Figure 2: Performance versus different input variance (σ_N^2) on CIFAR-10 and ImageNet dataset for a fixed adversarial noise intensity. The NIN and the single column AlexNet were used in the experiment.

D MORE RESULTS

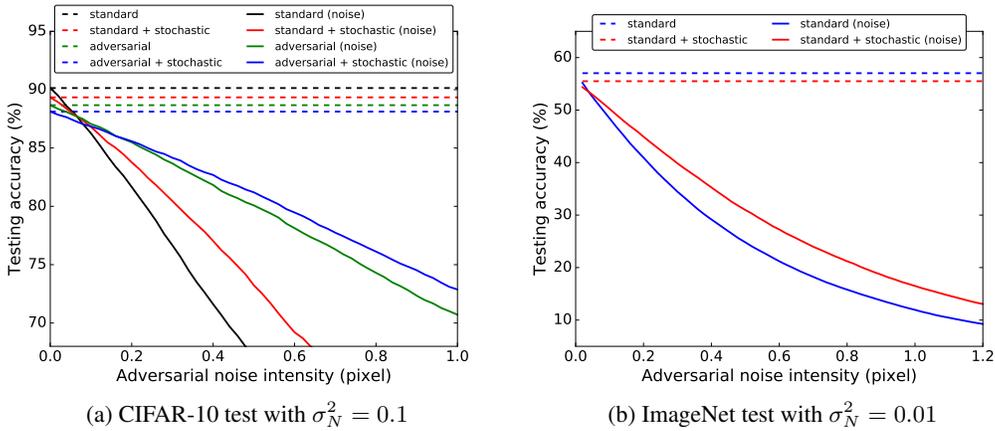


Figure 3: Performance versus different adversarial noise intensity (k_{adv}) on CIFAR-10 and ImageNet dataset for a fixed input variance. The NIN and the single column AlexNet were used in the experiment.

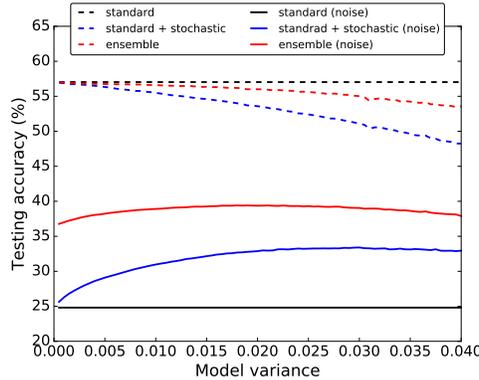


Figure 4: Performance of ensemble models over different input variance (σ_N^2) with $k_{adv} = 0.5$ in the presence of adversarial noise on ImageNet. The output probabilities from the standard model and standard model with stochastic FF are averaged with a ratio of 1:1.

E APPROXIMATION ERROR

The stochastic method based on a parametric model simplified computation, but it accompanies approximation in max-pooling and ReLU layer, which is a limiting factor of the algorithm.

From the earlier section, it is expected that the max-pooling with stochastic input model produces approximation error and its quantity depends on the ordering of elements. We tested accuracy due to the approximation error from the plain and ordered max-pooling to check the effectiveness of sorting in the context of CNNs. A single column AlexNet has more max-pooling layers with a larger pooling region than the NIN. Therefore, it is more vulnerable to approximation error and used to observe the effectiveness as reported in figure 5. We found that the sorted max-pooling gave higher accuracy most of time compared to the plain order. The plain max-pooling fails to draw accurate approximation toward the exact distribution for a high variance model due to its heavy-tailed distribution. In terms of computation, the cost of the sorting is negligible considering the fact that the pooling is employed at most on an 3×3 array in the literature (Krizhevsky et al., 2012).

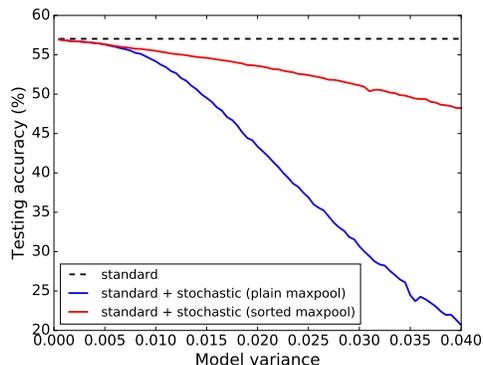


Figure 5: The accuracy gap due to the approximation error between the baseline and stochastic AlexNet without any perturbation. Sorted max-pooling achieves lower approximation error than plain max-pooling especially for the large variance (σ_N^2).

The ReLU operator often suffers from numerical instability. Intermediate representations in CNNs are generally sparse meaning that many values are populated around zero. From the equation 3, these near-zero values combined with tiny input variance produce non-trivial standard scores, which requires regularization. This regularization process makes the approximated value deviate from its true value.

In general, as CNNs become deeper, they are also sensitive to error accumulated during feedforward computation. Additionally, performance improvement has been achieved at the cost of additional computation and memory space. In the stochastic model, each input or intermediate value other than model parameters such as weights and biases is represented as a set of mean and variance. Therefore, it requires about two times of memory usage than the standard feedforward model.