Gradient Masking Is a Type of Overfitting

Yusuke Yanagita and Masayuki Yamamura

Abstract—**Neural networks have recently been attracting attention again as classifiers with high accuracy, so called "deep learning," which is applied in a wide variety of fields. However, this advanced machine learning algorithms are vulnerable to adversarial perturbations. Although they cannot be recognized by humans, these perturbations deliver a fatal blow to the estimation ability of classifiers. Thus, while humans perceive perturbed examples as being the same as the original natural examples, sophisticated classifiers identify them as completely different examples. Although several defensive measures against such adversarial examples have been suggested, they are known to fail in undesirable phenomena, gradient masking. Gradient masking can neutralize the useful gradient for adversaries, but adversarial perturbations tend to transfer across most models, and these models can be deceived by adversarial examples crafted based on other models, which is called a black-box attack. Therefore, it is necessary to develop training methods to withstand black-box attacks and conduct studies to investigate the weak points of current NN training. This paper argues that no special defensive measures are necessary for NN to fall into gradient masking, and it is sufficient to slightly change the initial learning rate of Adam from the recommended value. Moreover, our experiment implies that gradient masking is a type of overfitting.**

Index Terms—**Adam, adversarial examples, gradient masking, machine learning, neural network.**

I. Introduction

Deep learning is a state-of-the-art classification algorithm. Since Hinton *et al.* [1] succeeded in deepening neural networks (NNs) and this achieved the highest accuracy in the machine learning field, deep learning has been attracting interest in many fields and has dramatically improved image recognition, speech recognition, object detection, and bioinformatics [2].

However, Szegedy *et al.* [3] first pointed out that this sophisticated algorithm may be vulnerable to adversarial examples. They crafted small perturbations enough to be unrecognizable to humans, but significant enough to cause deep neural networks (DNNs) to misclassify, and they applied the perturbations to the original image. Such transformed examples are called adversarial examples, and have attracted much attention recently.

What problems can we consider when adversarial examples are abused in the real world? If an authentication system mounting deep learning is targeted, this system may allow miscreants to access important information or buildings. If self-driving vehicles are targeted, they may misrecognize signs or situations and cause severe accidents. In addition, other destructive attacks may be possible, because deep learning can be used in various applications. Deep learning has already begun to be applied in the real world, so these risks must be eliminated beforehand.

Many researchers have studied adversarial examples, including how to craft adversarial examples artfully and how to defend DNNs from the adversary to prevent the above-mentioned cases. An adversarial perturbation is usually crafted with the gradient of the DNN's output probability distributions with respect to input images. Some examples of adversarial crafting methods based on the above considerations have been proposed. Szegedy *et al.* [3] crafted adversarial examples with a box-constrained L-BFGS algorithm. They consider the problem of crafting adversarial examples as the minimization of example changes with a target label different from the original label as a constraint. To realize this, the objective function is formulated as the distance from the original example to the adversarial example with the loss function as a penalty function. Goodfellow *et al.* [4] suggested a method of crafting adversarial examples, the so-called "Fast Gradient Sign." This technique uses the sign of the gradient of the output probability distribution. Moosavi-Dezfooli *et al.* [5] proposed DeepFool, which can create finer perturbations than Fast Gradient Sign. In their approach, Newton's method was applied to produce very small but fatal perturbations. Papernot *et al.* [6] proposed a crafting method accessing the Jacobean of output probability distributions with respect to input images. This is known as Jacobian-based Saliency Map Attack (JSMA) and enables attackers to misclassify adversarial examples as any target labels. Moreover, Carlini *et al.* [7] also expanded the crafting method of Szegedy *et al.* [3] and improved the penalty function, the representation of the distance, and so on. Universal perturbation [8] uses DeepFool and updates a perturbation repeatedly to create network-specific perturbations. Interestingly, the generalizability of the universal perturbations across different networks is demonstrated.

Furthermore, crafting algorithms directly utilizing no autologous gradient have also been studied recently. Papernot *et al.* [9] developed a black-box attack, requiring no gradient information of any defender. In this algorithm, the attacker has its own DNNs, uses a few initial examples different from the defender's DNNs. And, this let its DNNs learn by checking the answers of defender DNNs. The fact that many adversarial examples for a certain model usually have the capacity to transfer to others is well-known [4], [10], [11], so adversaries do not need to access the defender DNN's architecture information.

Manuscript received March 3, 2018; revised May 12, 2018. This work was supported in part by JSPS Grant-in-Aid for Scientific Research (B) Grant Number JP26280095. The author also thank for the developers of TensorFlow, which is utilized in their study.

The authors are with the Department of Computer Science, School of Computer, Tokyo Institute of Technology, Yokohama, 226-8503, Japan (e-mail: y_yanagita@ali.c.titech.ac.jp, my@c.titech.ac.jp).

doi: 10.18178/ijmlc.2018.8.3.688

On the other hand, methods of defending against adversarial examples have also been developed. Goodfellow *et al.* [4] also suggested adversarial learning, which trains DNNs with both the natural sample and adversarial sample, in their paper. Defensive distillation [12], [13] is one of the most common methods. This approach makes use of distillation, which was originally introduced to transfer the learned information from one NN to another [14], as a defense technique, and the applicability to defending against Fast Gradient Sign [4] and JSMA [6] was demonstrated. Sengupta *et al.* [15] applied Bayesian game theory to defend DNNs. Given that a defender has some strategy for selecting DNNs and an attacker has the corresponding universal perturbations [8], the optimum strategy for the defender is sought. However, there is no proposal for a method that can reliably protect against the various attacks listed above. The protection to adversarial examples is an open problem.

Moreover, unfortunately, many potential defense mechanisms are said to fall into the category of gradient masking [9], which neutralizes the effective gradient to craft an adversarial example at the expense of the error rate. In this paper, we reveal, to some extent, what gradient masking is. It is known that gradient masking is developed relatively easily by NNs when they are trained to defend themselves. However, we found that when Adam [16] is used in the training phase on the MNIST dataset and the initial learning rate is varied slightly, gradient masking occurs. We also show that gradient masking may be the same phenomenon as overfitting. The occurrence of gradient masking is well known, but the mechanism has not been reported. Understanding what gradient masking is may assist future studies of more-appropriate learning methods to combat adversarial examples.

II. Attack Methods

We introduce adversarial examples after explaining a basic approach to craft adversarial examples, on which most current methodologies are based.

A. Principle of Attack Methods

Recent studies revealed a severe vulnerability to adversarial examples. Moreover, unfortunately, adversarial examples are crafted easily and quickly, so they are likely to be a threat to systems incorporating DNNs. The basic principle is introduced below.

In the field of classifications, NNs learn a probability distribution to express what an input is, and the class of the highest probability is chosen as a recognized class. Formally, given that the input space is X, the output label space T, the dataset is

$$D = \{(x_i, y_i) | x_i \in X, t_i \in T, i = 1, \dots, n\}$$

and ϕ is the learned probability distribution function, output of NNs. The output is

$$y_i = \arg\max_{y \in T} \phi(y | x_i;\ \theta) \tag{2}$$

where θ is the parameter determining the architecture of an NN. NNs are made to learn to match t_i and y_i $(i = 1, \dots, n)$. To do so, the cross-entropy loss function

$$J_{CE}(\theta) = -\sum_{x_i \in X} t_i{}' \log \phi(y | x_i;\ \theta) \tag{3}$$

where $t_i{}'$ is the one-hot expression of t_i, is usually minimized with respect to θ. So, in the learning epoch, we calculate the partial differential

$$\frac{\partial}{\partial \theta} J_{CE}(\theta) \tag{4}$$

and update θ.

Most adversarial examples are crafted by abusing the learned probability distribution or this loss function. For example, the probability distribution function gradient with respect to x supplies attackers with potentially useful information. This is because the gradient to reduce the probability of a recognized class. In other words, we can craft adversarial perturbations by calculating

$$\frac{\partial}{\partial x} \phi(y | x;\ \theta) \tag{5}$$

An adversarial example is crafted by applying an adversarial perturbation to a normal example. The adversarial examples are hardly any difference for human and they correctly recognize them owing to the very small perturbations. However, DNNs are forced to misclassify the adversarial examples. This indicates that DNNs may not be able to acquire the correct features of an input to learn it correctly.

Although many crafting methods for adversarial examples were suggested, shown in our introduction, we discussed the simplest methods to simplify the explanation of gradient masking.

B. Fast Gradient Sign Method

The fast gradient sign method (FGSM) [4] utilizes the sign of the adversarial gradient. This adversarial strategy crafts an optimal max-norm constrained perturbation

$$\eta = \epsilon \operatorname{sign}\left(\nabla_x J(\theta, x, y)\right) \tag{6}$$

where θ is the parameters of a model, x is the input to the model, y is the label associated with x, $J(\theta, x, y)$ is the cost function used to train the neural network, and ϵ is the perturbation magnitude enough to be nearly indistinguishable for a human but to force DNNs to misrecognize. This attack is developed to demonstrate that the linear nature of the activation functions causes DNN vulnerability to adversarial perturbation and they argued that a tiny perturbation on the high-dimension input space grows into a visible perturbation through propagation. Equation (6) implies changing the color of all pixels simultaneously by $\pm\epsilon$ uniformly, which implies making many small changes to an image.

III. Experiments

We now set up experiments to show the following phenomena.

1) When NNs learn with a non-recommended initial learning rate of Adam, NNs are less vulnerable to FGSM.
2) The phenomenon of gradient masking is similar to overfitting.

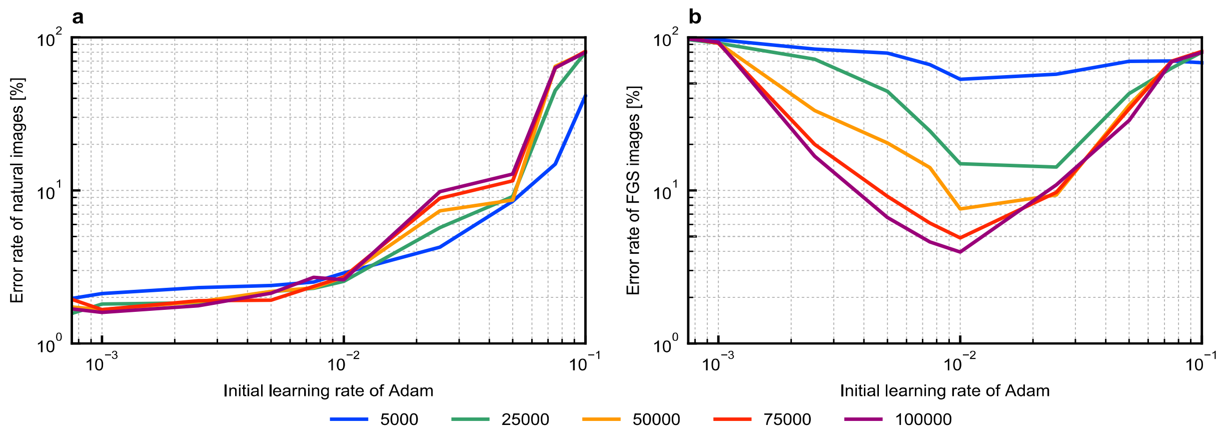

Fig. 1. Test error rate when the initial learning rate of Adam is varied. Line colors indicate different learning epochs. It is noted that both axes are logarithmic.
(a) Effect of varied learning rate about the error rate of natural MNIST images.
(b) Effect of varied learning rate about the error rate of MNIST images perturbed by FGSM. Epsilon of FGSM is 0.25.

Our experiments made use of well-studied handwritten digits, the MNIST dataset. The image data are composed of 28×28 = 784 pixels and labeled with multiple integers from 0 to 9. The number of the training data samples was 50,000 and the test data was 10,000 in our experiments. The NN architectures used in our experiments were fully-connected feed-forward NNs, shown in Table I. The batch size was 125 samples and learning epoch is 100000, unless otherwise noted.

A. *Valid Adam Initial Learning Rate*

First, we examined the tolerance of NNs to FGSM and DF with different popular optimizers, including mini batch optimization, momentum [17], AdaGrad [18], and Adam [16]. In this experiment, all hyper parameters were set to recommended values. The test error rate of FGS images is shown in Table II. NNs trained without Adam misrecognized FGS images almost completely, as explained in [4]. However, when learning with Adam, the test error rate of FGS images tends to be a little lower. So, we decided to verify the learning rate of Adam. The update rule of Adam is below.

```
**Initialization:** $m_0 \leftarrow 0, \quad v_0 \leftarrow 0, \quad t \leftarrow 0$
**while** $\theta_t$ not converged **do**
    $t \leftarrow t + 1$
    $m_t \leftarrow \frac{1}{1-\beta_1^t}\{\beta_1 m_{t-1} + (1-\beta_1)\nabla_\theta J_t(\theta_t)\}$
    $v_t \leftarrow \frac{1}{1-\beta_2^t}\{\beta_2 v_{t-1} + (1-\beta_2)\nabla_\theta J_t(\theta_t)^2\}$
    $\alpha_t \leftarrow \alpha \frac{\sqrt{1-\beta_2^t}}{1-\beta_1^t}$
    $\theta_t \leftarrow \theta_{t-1} - \alpha_t \frac{m_t}{\sqrt{v_t} + \epsilon}$
**end while**
```

In this algorithm, t is a time step, β_1 is the exponential decay rate for momentum, which is recommended to be 0.9, β_2 is the exponential decay rate for the learning rate, which is recommended to be 0.999, and ϵ is the value to escape the not-a-number condition, which is recommended to be 10^{-8}.

We tested the tolerance of NNs to FGSM with various initial learning rates of Adam. The perturbation magnitude was set to 0.25. Since [16] recommends that the initial learning rate be 0.001, we validated a sufficiently wide range for the first time. Yet, as it was difficult to be fooled particularly around 0.01, we decided to investigate in the range from 0.1 to 0.00075, which includes the recommended value. We had NNs learn the MNIST dataset and examined the test error rate of natural and FGS images. Besides the error rate, the number of valid FGS images was calculated based on the fact that if the gradient with respect to the example is zero, FGSM cannot craft any adversarial example, because FGSM makes use of the gradient, as explained in Section II.B.

TABLE I: ARCHITECTURE OF FULLY-CONNECTED NNS

Layer	Description
Input layer	28×28
1st hidden layer	FC(600)+ReLU
2nd hidden layer	FC(128)+ReLU
Output layer	FC(10)+Softmax

The network has four layers including input and output layers. FC means fully-connected layers and the number means the number of units.

TABLE II: COMPARISON BETWEEN OPTIMIZATION METHODS OF ERROR RATE OF TWO TYPE OF IMAGES

Methods	Natural images	FGS images
Mini batch	2.36	99.9
Momentum	1.99	99.5
AdaGrad	2.05	99.9
Adam	1.68	90.4

The units in the table are in percent.

When the initial learning rate was more than 0.075, the test error rate of natural images reached 60% or higher (see Fig. 1(a)). This implies a very large initial learning rate of Adam makes it impossible to learn, because weight vectors of NNs cannot converge to the proper local minimum. As the initial learning rate decreased, the test error rate of natural images also decreased and was less than 3% with an initial learning rate less than 0.01. However, the behavior of the test error rate curves of FGS images differed from the above one absolutely. High error rates of FGS images were acquired with a large learning rate, and the number of images crafted successfully by FGSM was less than 40% (see Fig. 1(b)), because, from the first, they cannot learn anything and the learned probability distribution might be uniform. On the other hand, test error rates of FGS images were also high when the initial learning rate was close to the recommended value, as expected from the previous experiment. In this case, FGSM naturally succeeded in crafting valid adversarial examples with 90% probability. Surprisingly, when the initial learning rate was set to around 0.01, the test error rate of FGS images was obviously lower. In this instance, FGSM could not craft valid images, with 20% probability or less, and the test error rate of natural images was somewhat higher than the one with the recommended initial learning rate. This implies that a particular initial training rate of Adam causes gradient masking.

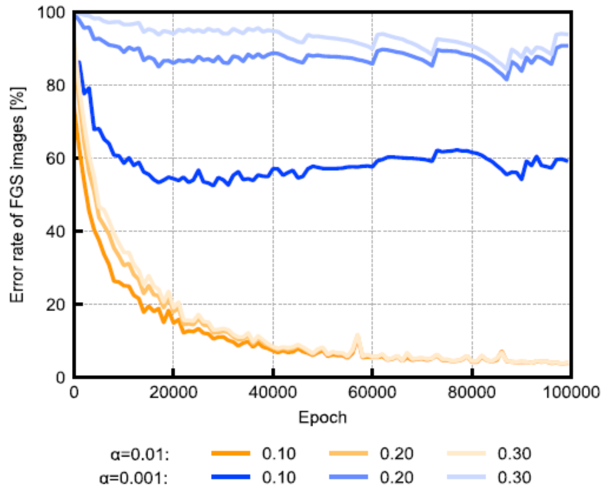

Fig. 2. Test error rate of FGS images when epsilon of FGSM is varied, which is reflected in the line colors. The orange line is the case when the initial learning rate is 0.01, while the blue line is the case when the initial learning rate α is 0.001. The color density represents the magnitude of perturbation ε.

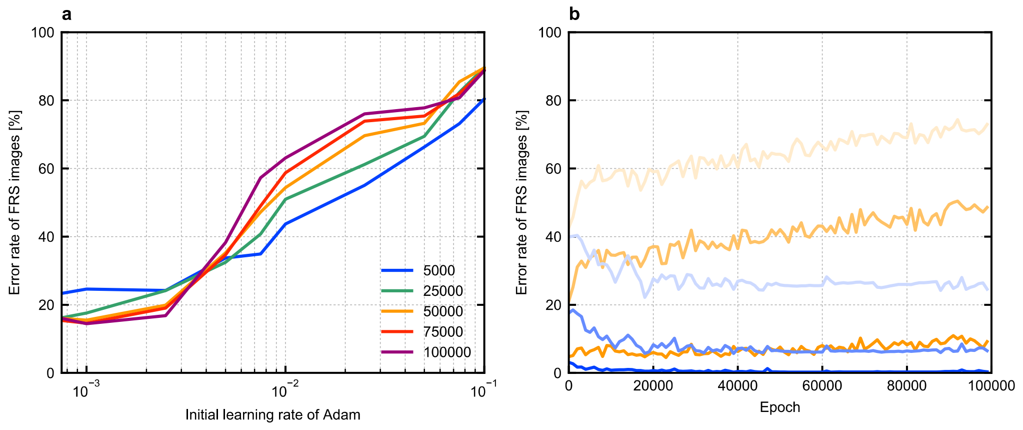

Fig. 3. Error rate of FRS images when initial learning rate of Adam is varied. FRSM was executed on the training images.
(a) Effect of varied learning rate about the error rate of MNIST images perturbed by FRSM. The color of line represents the learning epochs.
(b) Comparison of changes in random noise immunity during the learning process. The orange line is the case when the initial learning rate is 0.01, while the blue line is the case when the initial learning rate α is 0.001. The color density represents the magnitude of perturbation ϵ.

B. *Gradient Masking*

Because no paper has studied how to develop gradient masking, we examined the above phenomenon in more detail. So, the resistance of the above NNs to the random perturbation, instead of the gradient-based perturbation, was examined. We crafted a max-norm constrained perturbation and applied it to an image. This method uses a random perturbation instead of the gradient-based perturbation of FGSM, so it is named the Fast Random Sign Method (FRSM) after FGSM in this paper. Calculating the training error rate of FRS images enables us to examine how far the decision boundary extends around an input point. In other words, when the training error rate of FRS images with ϵ is very low, this implies that the boundary surrounding a data point extends by more than ϵ on average. We also show the case of FGS images as comparison in Fig. 2. As the initial learning rate of Adam increased and the test error rate of natural images increased, the training error rate of FRS images increased (see Fig. 3(a)). This implies that gradient masking brings the decision boundary closer to the training data points. Also, for the training epochs (see Fig. 3(b)), while the error rate curves of FRS images were flat with an initial learning rate of 0.001, the error rate slope increased with an initial learning rate of 0.01. This implies that gradient masking occurred, thus further constricting the decision boundary around the training data points.

IV. Discussion

So far, gradient masking was thought to occur when DNNs are trained to defend themselves. However, we found that this phenomenon occurs under normal learning conditions without special regularizations or defense strategies to adversarial gradients. Our paper demonstrates that training with Adam and the non-recommended initial learning rate prevents FGSM but reduces accuracy, i.e., gradient masking takes place. This implies that potential defense mechanisms induce NNs to converge to particular local solutions, and the minimum is not guided only by defense methodologies.

What characteristic of Adam caused this phenomenon? Adam has two exponential moving averages of gradient and square of gradient. In other words, the algorithm of Adam is divided into two parts, momentum and RMSprop [19]. Momentum allows more efficient search by increasing the inertia of the gradient, and RMSprop automatically adjusts the learning rate from the latest gradient information. Because, in the above experiment, NNs trained with momentum were almost fooled by FGSM, unlike Adam, we investigated whether RMSprop shows the same tendency as Adam. Indeed, training with RMSprop was as difficult to fool as Adam was. On the other hand, AdaGrad, which adaptively changes the learning rate for each weight from the sum of all gradients calculated in the past, did not give this result. This implies that adaptive changing of the learning rate for each weight from the sum of the immediate gradients generated gradient masking.

Although it is not shown in this paper, we compared our learning, which sets the learning rate to 0.01, to adversarial training [4] to evaluate the security level. In the case of setting epsilon to 0.25 with $L\infty$ norm, our method reduced the test error rate of FGS images to 5%, while adversarial training reduced it to only about 10%.

V. Conclusion

We found that the initial learning rate of Adam affected gradient masking without applying any defense technique. This implies that gradient masking is not necessarily a byproduct of the defense to adversarial examples. Moreover, gradient masking showed the same tendency as overfitting in our experiment. We should examine the hyper-parameters carefully in these studies because the result of neural network training depends on them complicatedly.

References

[1] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[2] L. Yann, B. Yoshua, and H. Geoffrey, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[3] C. Szegedy *et al.*, "Intriguing properties of neural networks," in *Proc. ICLR.*

[4] G. Ian, J. S. Jonathon, and S. Christian, "Exampling and harnessing adversarial examples," in *Proc. ICLR, 2015*.

[5] S.-M. Moosavi-Dezfooli, A. Fawzi, and P. Frossar, "DeepFool: A simple and accurate method to fool deep neural networks," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016.

[6] N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. Berkay Celik, and A. Swami, "The limitations of deep learning in adversarial settings," *Proc. the 1st IEEE European Symposium on Security and Privacy*, 2016.

[7] N. Carlini and D. Wagner, "Towards Evaluating the Robustness of Neural Networks," in *Proc. IEEE Symposium on Security and Privacy*, 2017.

[8] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[9] N. Papernot, P. Mcdaniel, I. Goodfellow, S. Jha, Z. Berkay Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proc. ACM Asia Conference on Computer and Communications Security*, 2017.

[10] N. Papernot, P. Mcdaniel, and I. Goodfellow, "Transferability in machine learning: From phenomena to black-box attacks using adversarial samples," *arXiv preprint arXiv:1605.07277*, 2016.

[11] F. Tramèr, N. Papernot, I. Goodfellow, D. Boneh, and P. McDaniel, "The space of transferable adversarial examples," *arXiv preprint arXiv: 1704.03453*, 2017.
N. Papernot, P. McDaniel, X. Wu, S. Jha, and A. Swami, "Distillation as a defense to adversarial perturbations against deep neural networks," in *Proc. 2016 IEEE Symposium on Security and Privacy*, 2016, pp. 582–597.

[12] N. Papernot and P. McDaniel, "Extending defensive distillation," *arXiv preprint arXiv: 1705.05264*, 2017.

[13] L. Jimmy Ba and R. Caruana, "Do deep nets really need to be deep?" *Adv. Neural Inf. Process. Syst.*, 2014.

[14] S. Sengupta, T. Chakraborti, and S. Kambhampati, "Securing deep neural nets against adversarial attacks with moving target defense," *arXiv preprint arXiv:1705.07213*, 2017.

[15] D. P. Kingma and J. L. Ba, "Adam: A method for stocastic optimization," in *Proc. ICLR, 2015*.

[16] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *Comput. Math. Math.* Phys., vol. 4, no. 5, pp. 1–17, 1964.

[17] J. Duchi, J. B. Edu, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization*," *J. Mach. Learn. Res*., vol. 12, pp. 2121–2159, 2011.

[18] T. Tieleman and G. Hinton, "Lecture 6.5 - RMSProp, COURSERA: Neural networks for machine learning," *Tech. Rep*., 2012.

Yusuke Yanagita graduated from Faculty of Science, Tohoku University, Sendai, Japan in 2017. Then he is a master course student in the Department of Computer Science, School of Computing, Tokyo Institute of Technology, Yokohama, Japan, now. His recent research interests are in fields of adversarial examples and learning theorem.

Masayuki Yamamura got the doctor degree of engineering from Tokyo Institute of Technology in 1990. He has been an assistant professor in 1989, an associate professor in 1996 and a professor in 2004 in Tokyo Institute of Technology. He is a vice dean of the School of Computing in Tokyo Institute of Technology. His research interests are in the field of artificial intelligence, evolutionary computation, systems and synthetic biology, DNA computing and molecular robotics.