Momentum Contrast Self-Supervised Based Training for Adversarial Robustness

Monireh Moshavash\textsuperscript{a}, Mahdi Eftekhari\textsuperscript{a,*}, Kaveh Bahraman\textsuperscript{a}
\textsuperscript{a}Department of Computer Engineering, Shahid Bahonar University of Kerman, Kerman, Iran.

\textbf{A R T I C L E I N F O.}

Article history:
Received: 5 January 2021
Revised: 2 June 2021
Accepted: 29 June 2021
Published Online: 21 July 2021

Keywords:
Adversarial Attack, Defense, Projected Gradient Descent, Robust Accuracy.

\textbf{A B S T R A C T}

By the rapid progress of deep learning and its use in a variety of applications, however, deep networks have shown that they are vulnerable to adversarial examples. Recently developed researches show that using self-supervised learning (SSL) in various ways results in increasing network robustness. This paper examines the effect of a particular type of Contrastive SelfSupervised learning (CSSL) called Momentum Contrast (MoCo) on increasing network robustness to adversarial examples. For this purpose, MoCo is employed as a pre-text task and a deep network is pre-trained for this task. Then fine-tuning will cause to increase the robustness of the network against adversarial attacks examples. A new attack method is introduced based on MoCo and one of the Projected Gradient Descent (PGD) or Fast Gradient Sign (FGSM) methods that do not require any labeled data. Using this corrupted data and adversarial training method, a deep network is pre-trained and the representation provided by it is used to fine-tune downstream tasks that results in increasing network robustness. For an instance, the setup including Resnet50 structure, PGD attack, and MoCo-v1 shows 2.79%, 2%, and 1.35% of improvements comparing to the Jigsaw, Rotation, Selfie, respectively. More details of experiments and the improvements raised by MoCo are given in the results part and show the superiority of MoCo based models on CIFAR-10 and CIFAR-10-C datasets. Also, the obtained results for validating the robustness of proposed models against various noises with different corruption strengths, confirm the resistance of the proposed methods.

© Research Article, 2021 JComSec. All rights reserved.

\textbf{1 Introduction}

According to research, one of the challenges that researchers encounter in deep networking is the lack of a required amount of labeled data. Therefore, in supervised training with a deep network, we need a large amount of labeled data, which is very expensive to collect such data. One way to address this challenge is to use self-supervised and unsupervised training methods that do not require labeled data during the training process \[1,2,8,16\]. Adversarial examples
are somewhat different from the original data but the neural network is unable to categorize them correctly. Neural networks are deceived by this data. The vulnerability of deep networks to adversarial examples has also increased the challenge of requiring effective data and labeling. For the first time as suggested by [9], self-supervised training was used to increase network robustness to adversarial attacks. Also, using the representations obtained by pre-training of a deep network with self-supervised tasks allows the fine-tuning of downstream tasks to perform fast and also leads to better generalization[1]. Recent work has shown that the use of CSSL [7], [8], [9] results in learning representations that increase classification accuracy. In this paper, adversarial training is used due to compelling reasons in [10], [11]. It is one of the best ways to defend against adversarial attacks, and also according to the results reported in [1] network resistance is higher than using standard training.

As suggested by [7], MoCo is used to learn representations in an unsupervised manner and also the learned representations are easy to transfer to downstream tasks. MoCo trains a representation encoder by matching an encoded query q to a dictionary of encoded keys using a contrastive loss [7]. The dictionary keys \{k_0, k_1, k_2,...\} are defined on the fly by a set of data samples. The dictionary acts as a queue such that the current mini-batch is inserted and the oldest mini-batch is removed. The keys are encoded by a slowly progressing encoder, driven by a momentum update with the query encoder. This method enables a large and consistent dictionary for learning visual representations. In this research, MoCo is used to address the vulnerability of deep networks to adversarial attacks. The contributions of this paper are as follows:

- The main intention of this research is to develop a new self-supervised method for generating adversarial attacks and using them for making deep networks robust; without using any supervised data. Therefore, a new method for creating adversarial examples based on MoCo loss function (i.e. a self-supervised contrastive loss) and the PGD and FGSM is proposed that does not require labeled data.

- The adversarial samples generated from the previous step are used as a pre-text task for the pre-training of deep networks. This step does not require any supervised data and like the previous step, it is performed based on the MoCo contrastive loss.

- Fine-tuning will cause to increase the robustness of the network against adversarial attacks examples. To this end, network obtained from the previous step as well as the fully connected part for predicting the ground truth labels of classification are used for fine tuning process. In this stage, the weights achieved in the pre-training stage are used as initial values for fine-tuning while the fully connected weights are trained from scratch to predict the class labels. All the adversarial examples and cleaned samples are employed in this phase.

- Details of the improvement from the proposed method were presented in the experiments and results section. The proposed MoCo-PGD and MoCo-FGSM methods were compared and contrasted against the other famous self-supervised based approaches namely Jigsaw, Rotation, Selfi, and RoCl. Furthermore, some experiments have been designed and performed for justifying the robustness of our proposed approach in the presence of various corruption strengths of different noises.

2 Related Works

Today, deep networks play an important role in the field of artificial intelligence but they have problems such as vulnerability to adversarial attacks. Researchers have so far proposed various methods to solve this problem and improving the performance of deep networks. In the following, a number of these works are reviewed.

Recently, it has been shown that using unlabeled data and pre-training can cause network robustness [12], [13]. Pre-trained models by SSL methods have been prevalently used in fine-tuning downstream tasks. In this paper, Selfie, Rotation, and Jigsaw methods were used to pre-train the deep network, and finally, the representations obtained from these networks were used to fine-tune the deep network for increasing network robustness to adversarial attack [1]. One of the most important ways to defend against adversarial attacks is to use adversarial training. Adversarial learning seriously needs labeled data, e.g., the PGD attack method requires labeled data [10]. Also, semi-supervised adversarial learning methods still require partially labeled data. For this reason, the Robust Contrastive Learning (RoCL) method has recently been proposed in which a new attack method has been proposed based on unlabeled data. Also, it has provided a new framework for adversarial training of deep networks without the need for labeled data using CSSL, which has been able to increase the network’s robustness to adversarial attacks [11]. To the best of our knowledge, MoCo has not been used yet for adversarial learning. Therefore, in the present research,
it is used as a pretext task to pre-train a model followed by fine-tuning. The results obtained from the MoCo illustrate its ability for making robustness in adversarial training.

3 Preliminaries

3.1 Self-Supervised Learning

The term SSL has been used in different contexts and fields, such as representation learning, neural networks, robotics, natural language processing, and reinforcement learning. In all cases, the basic idea is to automatically generate some kind of supervisory signal to solve some task (typically, to learn representations of the data or to automatically label a dataset). A number of SSL methods have been developed in recent years include: region/component filling (e.g., inpainting [14] and colorization [15]), rotation prediction [16], category prediction [17], and patch-based spatial composition prediction (e.g., Jigsaw [18], [19] and Selfie [20]). In the aforementioned self-supervised methods, standard training has been used. Rotation, jigsaw puzzle, selfie [1], [2] have been used to consider the effect of self-supervised learning on network robustness against adversarial examples. MoCo [7] and SimCLR [8] are CSSL. The basic idea of the CSSL is to create the augmented samples from the original ones and use a predictive task for checking whether the augmented samples are from the same original ones [21].

3.2 Contrastive Loss

Contrastive loss measures the similarity between two samples in the representation space, its equation is as follows [7]:

$$l_q = -\log \frac{\exp(q.k_+/\tau)}{\sum_{i=0}^{k} \exp(q.k_i/\tau)}$$

(1)

A low value of $l_q$ shows that $q$ is similar to positive key $k_+$, and dissimilar to the other keys that are considered as negative keys. For each $q$ there is one positive key and $k$ negative keys. In fact, log loss tries to classify $q$ as $k_+$. In Eq (1), $q$ is the feature representation of the original data, $k_+$ is positive key (augmentation sample similar to the original data), $k_i$ is the $ith$ negative key (augmentation samples dissimilar to the original data) and $\tau$ is a temperature hyperparameter [22].

3.3 Momentum Contrast (MoCo)

Kaiming He et al. [7] presented MoCo to learn unsupervised representation with a contrastive loss (Figure 1). MoCo, end to end, and Memory bank are three mechanisms for using contrastive loss. These three mechanisms differ depending on how the key encoder is updated and how the keys are maintained. MoCo encodes new keys by a momentum-updated encoder and maintains a queue of keys [7]. Contrast methods with more negative samples will have better performance since they allow the distribution of data to be covered more effectively and as a result, the learning process will be performed better. Based on the results obtained in [7], MoCo had better results than other methods of contrastive loss, and accordingly, it was our motivation to examine MoCo for adversarial training. MoCo-v2 is the other version of MoCo that has two differences. The first difference is the replacing of a 1-fully connected layer with a 2-layer MLP head, and the second one is the use of blur augmentation (the same as SimCLR).

3.4 Adversarial Attack

So far, many methods of adversarial attacks have been proposed. Attacks on deep networks are generally divided into three groups, which include threat, perturbation, and benchmark models [23]. White box and black box attacks are the most important attacks that have been widely used in recent years and have been sought to address. White box attacks have full access to all network information such as architecture, parameters, inputs, and model outputs. Unlike white-box attacks, black box attacks only have access to the output of the model and do not know any more information about the network [24]. In addition to the above methods, Goodfellow et al. [25] proposed an FGSM that calculates the gradient of the loss function relative to the neural network input.
3.5 Adversarial Robustness

Finding a way to increase network robustness and defend against adversarial examples is a popular topic of research. The problem of defending against adversarial examples can be formulated as follows:

$$\min_{\theta} E_{(x,y) \sim D} \left[ \max_{\epsilon \epsilon S} L(x', y; \theta) \right]$$  \hspace{1cm} (2)

where $x$ is a sample and $x'$ is the perturbed version of it. $s$ denotes the subset of perturbed samples that are within the $\epsilon$-ball of the $x$ and formally is defined as:

$$S = \{x' : \|x - x'\| < \epsilon\}$$

Recently, some methods have been developed for increasing the robustness of deep networks to adversarial attacks, which have been generally divided into three categories: modifying data, modifying models, and using auxiliary tools \cite{45}. Adversarial training methods are in the category of modifying data approaches.

3.6 Fast Gradient Sign Method (FGSM)

The Fast Gradient Sign Method (FGSM) \cite{46}, \cite{47} is a one-step method for creating adversarial examples. In this method, the gradient of the loss function is calculated relative to the neural network input. FGSM is fast because it only involves calculating one backpropagation step. Adversarial samples are generated with the following formula:

$$x' = (x + \epsilon \text{sign} (\nabla_x L(x, y)))$$  \hspace{1cm} (3)

where $L$ is the loss function, $\nabla_x$ indicates the gradient of the model for a normal sample $x$ with correct label $y$, $\epsilon$ is FGSM step sizes. $x'$ is the generated perturbed sample.

3.7 PGD-Attack

Although standard adversarial attacks such as PGD \cite{10} and TRADES \cite{28} use labeled data, a new attack method has recently been introduced that does not require labels \cite{11}. PGD attack is very similar to the FGSM where both of them use the gradient of a loss function. The PGD attack is generated as follows \cite{6}:

$$X^{k+1} = \prod_s \left( X^k + \alpha \text{sign} (\nabla_x L(x^k, y; \theta)) \right)$$  \hspace{1cm} (4)

and $X^0 = X + \bigcup(-\epsilon, \epsilon)$
where $k$ is a parameter that characterizes the iteration number, $\prod_s$ is the projection operator, $\alpha$ is step-size and $L(x, y; \theta)$ is the loss that should be optimized. Normally, this loss is the cross-entropy between the model’s SoftMax classification output for $X$ and the ground truth label. The main difference between FGSM and PGD is that the former is not iterative but the latter is generated iteratively. Also, PGD uses the projection operator for controlling the range of generated attacks.

### 4 Proposed Method

#### 4.1 MoCo-PGD Attack

In this paper, MoCo is used to build adversarial examples by utilizing the PGD attack method [7]. This method is called the MoCo-PGD attack and unlike the standard PGD method does not require labeled data to generate adversarial samples. Two augmented samples from a batch of data are produced as well as perturbed version one of the samples by using a noise generated from the uniform distribution. Then, these two augmented images are fed into the MoCo model. The MoCo structure is built of two encoder networks and a queue. The perturbed augmentation data are fed into the query encoder and the others to the momentum encoder. Contrastive Loss (Eq.(1)) tries to find the similarity between perturbed augmentation and the keys in the queue. The most similar key to perturbed augmentation is called a positive sample. Perturbed augmentation is classified as a positive example and adversarial samples are created by maximizing the loss obtained from the previous step. In Eq (5),(6) the loss functions and update rule of the MoCo-PGD attack are given. The flowchart of adversarial training via MoCo-PGD attack is depicted in Figure 2.

The flowchart has three main blocks. The first is the augmentation block, in which two different augmentations are produced from the same image. MoCo uses RandomResizedCrop, RandomHorizontalFlip, ColorJitter, RandomGrayscale augmentations, and MoCo-v2 has another GaussianBlur in addition to those of MoCo which are considered as inputs to the MoCo Encoders (i.e., query and key).

The second block is the MoCo model that is used to create a new attack based on losses. MoCo is a contrastive self-supervised method that is used to build attacks without data labels (MoCo-PGD, MoCo-FGSM).

The Final block is based on Eq (6) in which attacks are iteratively generated by the sign of the gradient of loss. In this way, adversarial examples are produced for network pre-training.

$$Loss_{adv} = MoCo - Model(img_{adv}, img_k) \quad (5)$$

$$X^{k+1} = \prod_s (X^k + \alpha \text{sign}(\nabla_x Loss_{adv})) \quad (6)$$

$$MoCo - Model() \text{ in Eq (5) is exactly the model being introduced in [7]. For calculating the similarity between two data based on contrastive loss } img_{adv} \text{ contains perturbed augmented samples and } img_k \text{ includes the other augmented samples. In (6) } Loss_{adv} \text{ is used to do the PGD attack (like Eq.(4))}.$$  

#### 4.2 Adversarial Pre-Training Through MoCo Framework

At this step, by using the adversarial samples generated with the MoCo-PGD attack method the network is pre-trained via contrastive loss of MoCo in a self-supervised manner. The representations obtained from the previous step are used to fine-tune the down-
4.3 MoCo-FGSM Attack

In this paper, adversarial examples are produced using MoCo and based on FGSM. Then, as before, these adversarial examples are used for the pre-training network. The representations obtained from this network have been used to fine-tune the downstream task of increasing the network robustness against adversarial examples. Figure 3 shows the production of adversarial examples using MoCo and FGSM, and pre-training network with these examples and MoCo structure is in accordance with Figure 3. FGSM attacks are generated in Figure 4 based on Eq (7) by using the sign of the gradient of loss and without any iteration. The network is pre-trained by using the samples produced in the previous step and the MoCo.

\[ X' = (X + \epsilon \text{sign} (\nabla_x \text{Loss}_{adv})) \] (7)

5 Experiments and Results

5.1 Setup and Datasets

In this work, we have used Python and the Pytorch library to write the code and Google Colab to execute the code. This experiment was performed in two phases, pre-training and fine-tuning. For the pre-training phase, Resnet-50 and Resnet-18 structures with a momentum of 0.9 and a batch size of 128 are used. Initial values of different parameters are as follows: learning rate of 0.06, weight decay of \(5 \times 10^{-4}\), step-size of 8/255, a step of attacks of 10, and epoch number of 200. For the full adversarial fine-tuning phase, the initial learning rate of 0.1, weight decay of \(3 \times 10^{-4}\), the step of attacks is 20 and the epoch number of 50 is set. The other parameters are considered the same as the pre-training phase. In this work, we use the CIFAR-10 dataset, which consists of 60000 32×32 color images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. We also used the CIFAR-10-C [29] dataset to test the model on unforeseen adversarial attackers.

5.2 Results and Discussion

The results for full adversarial fine-tuning for two adversarial generating methods namely MoCo- FGSM and MoCo-PGD are shown in Table 1. Both MoCo and MoCov2 contrastive learning methods are used for adversarial training. Results of testing FGSM on unseen attacks (CIFAR-10-C dataset) are shown in Figure 5, Figure 6, and Figure 7. Figure 8 shows our method can obtain better results on the CIFAR-10-C dataset compared with Selfie, Jigsaw, and Rotation methods [1].

Our intention for conducting the first experiment was to generate MoCo-FGSM adversarial samples and pre-train the network with the data to increase the network’s resistance to strong, single-stage attacks such as FGSM. According to Table 1, it can be seen that the proposed method, leads to an increase in the network resistance against the attack compared to other similar methods. The details of the experiment and improvement obtained are as follows: The configuration including the FGSM attack, Resnet50, and MoCo-v1 method, has the improvements of 14.33%, 4.67%, 5.86%, 17.35% in comparison to the Jigsaw, Rotation, Selfie, RoC, respectively. The same experiment was repeated by MoCo-v2 and was led to the following improvements: 19.28%, 9.62%, 10.81%, 22.3%.

The purpose of conducting the second experiment was to generate MoCo-PGD adversarial samples and pre-train the network...
Figure 5. Comparison of Results Obtained From MoCo and MoCo-V2 on CIFAR-10-C Dataset Using MoCo-FGSM Attacks. (R18 Is Resnet18).

Figure 6. Comparison of Results Obtained From MoCo and MoCo-V2 on CIFAR-10-C Dataset using MoCo-FGSM Attacks. (R50 Is Resnet50).
with the data, and increase the network’s resistance against strong and iterative attacks such as PGD. According to the results, as can be seen the proposed method, has increased the network resistance against the type of attack compared to similar methods. For the setup including PGD attack, ResNet18, and MoCo-v1 improvements of 4.62%, 3.84%, 3.06%, 1.74% are obtained comparing to the Jigsaw, Rotation, Selfie, Rocl respectively. Also, we have repeated the above experiments with the MoCo-v2, the purpose is to compare the performance of MoCo with MoCo-v2 in robustness against adversarial attacks. The percentage of improvements are 3.42%, 2.44%, 1.86% and 0.74%.

Finally by PGD attack, ResNet50 structure, MoCo-v1 compared to the Jigsaw, Rotation, Selfie, achieved an improvement of 2.79%, 2%, and 1.35%.

We examine the network’s behavior against a variety of noise unseen data in the third experiment. The purpose of the experiment is to evaluate the robustness of the model which was trained with the MoCo, against the noise data. To experiment, the CIFAR-10-C database was used and all reported results were Top 1 accuracy. The database has 5 difficulty levels for each noise. The reported accuracy is the average of the accuracy of 5 levels. The experiment is performed four times including, 1: we have used the ResNet18 structure, and the models trained with the MoCo and MoCo-v2, and the attacks developed with the FGSMMoCo method. The results of the experiment are shown in Figure 5. The obtained accuracy is about 90% in defense against some noise. 2: The same experiment is done for the ResNet50 structure. According to the results, presented in Figure 6, it can be seen that deepening the model will increase its resistance to the noises. (In defense against some noise, the achieved accuracy is higher than 90%) 3: the Resnet50 structure is used, and the models are trained with the MoCo and MoCo-v2 methods and the attacks developed by the PGD-MoCo method. The results of the experiment are shown in Figure 7. As you can see, the methods had better performance against all types of noise, and they have reached an accuracy close to 80% for some noises. 4: similar cases and the proposed method were compared in the CIFAR-10-C database. As you can see in Figure 8, our proposed method performed better than similar methods in all the noises.

Figure 7. Comparison of the Robust Accuracy Obtained by Testing Two Methods of MoCo and MoCo-V2 on the Database of CIFAR-10-C (Unseen Attacks). Model’s Training Is Based on the MoCo-PGD Attacks. Each Bar Represents an Average Overall Five Corruption Strengths for a Given Corruption Type.
6 Conclusions

In this paper, a new method of generating adversarial attacks was introduced using MoCo and PGD methods. By using the generated adversarial examples as well as adversarial training, the effects of the MoCo, and the other self-supervised methods on the network robustness to adversarial attacks were investigated. Based on the results, it was concluded that the use of perturbed data being created with MoCo in adversarial training had a greater impact on increasing the network’s robustness to unseen attacks. Also, this method does not require labeled data. It is worth noting that fine-tuning representations obtained from MoCo-based adversarial pre-training networks were more effective than the other contrastive and SSL methods for network robustness.

References


Monireh Moshavash was born in Bushehr, Iran in 1993. She received her B.Sc. in computer engineering from the Department of Computer Engineering, Sistan and Baluchestan University, Zahedan, Iran in September 2018. She is currently pursuing an M.Sc. Degree in Artificial Intelligence in Department of Computer Engineering, Shahid Bahonar University of Kerman, Kerman, Iran. Her research interests include Machine Learning and Deep Learning.

Mahdi Eftekhari was born in Kerman, Iran in 1978. He received his B.Sc. in computer engineering from the Department of Computer Science and Engineering, Shiraz University, Shiraz, Iran in September 2001. He obtained his M.Sc. and Ph.D. degrees in Artificial Intelligence from the same department in 2004 and 2008 respectively. Mahdi has been a faculty member of Shahid Bahonar University of Kerman, Kerman, Iran since 2008 and he is currently an associate professor in the Department of Computer Engineering. His research interests include Data Mining, Fuzzy Systems and modeling, Machine Learning, Deep Learning, and Applications of matrix methods in machine learning. He is the author and co-author of about 130 papers in cited journals and conferences.

Kaveh Bahraman was born in Sirjan, Iran. He is currently pursuing a B.Sc. Degree in Department of Computer Engineering, Shahid Bahonar University of Kerman, Kerman, Iran. His research interests include Machine Learning, Deep Learning and Applications, and Mathematics and Physics.