EFFECTIVE AND EFFICIENT VOTE ATTACKS ON CAPSULE NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Standard Convolutional Neural Networks (CNNs) can be easily fooled by images with small quasi-imperceptible artificial perturbations. As alternatives to CNNs, the recently proposed Capsule Networks (CapsNets) are shown to be more robust to white-box attack than CNNs under popular attack protocols. Besides, the class-conditional reconstruction part of CapsNets is also used to detect adversarial examples. In this work, we investigate the adversarial robustness of CapsNets, especially how the inner workings of CapsNets change when the output capsules are attacked. The first observation is that adversarial examples make most votes from primary capsules invalid. The second one is obvious. Namely, it is very costly to apply multi-step attack methods directly on CapsNets due to computationally expensive routing mechanisms. Motivated by two observations, we propose effective and efficient vote attacks where we attack CapsNets in a novel way. The proposed vote attacks are compatible with various L_p-norm attacks and reduce the robust accuracy significantly. Furthermore, we successfully integrate our vote attacks in the detection-aware attacks to bypass the class-conditional reconstruction based detection method. This study indicates that it is too early to claim that CapsNets are more adversarially robust than CNNs.

1 INTRODUCTION

A hardly perceptible small artificial perturbation can cause Convolutional Neural Networks (CNNs) to misclassify an image. Such vulnerability of CNNs can pose potential threats to security-sensitive applications, e.g., face verification and autonomous driving. Besides, the existence of adversarial images demonstrates that the object recognition process in CNNs is dramatically different from that in human brains. Hence, the adversarial examples have received increasing attention since it was introduced in the community Szegedy et al. (2014); Goodfellow et al. (2015).

Many works show that network architectures play an important role in adversarial robustness Madry et al. (2018); Su et al. (2018); Xie & Yuille (2020); Guo et al. (2020). As alternatives to CNNs, Capsule Networks (CapsNets) have also been explored to resist adversarial images since they are more biologically inspired Sabour et al. (2017). The CapsNet architectures are significantly different from those of CNNs. Under popular attack protocols, CapsNets are shown to be more robust to white-box attacks than counter-part CNNs Hinton et al. (2018); Hahn et al. (2019). Furthermore, the reconstruction part of CapsNets is also applied to detect adversarial images.

In a classification of an image, CapsNets first extract primary capsules from the pixel intensities and transform them to make votes. The votes will reach an agreement via an iterative routing process. It is not clear how these components change when CapsNets are attacked. Additionally, due to the routing mechanism, it is computationally expensive to apply the state-of-the-art attacks (e.g., PGD Madry et al. (2018)) to CapsNets directly. The effectiveness of the attack is also unsatisfying, where the robust accuracy of CapsNets and counter-part CNNs are 17.3 and 0, respectively. The two observations further motivate us to propose an effective and efficient vote attack on CapsNets.

The contributions of our work can be summarised as follows: 1). We investigate the inner working changes of CapsNets when output capsules are attacked; 2). Motivated by the investigation, we propose an effective and efficient vote attack; 3). We integrate the vote attack in the detection-aware attack to bypass class-conditional reconstruction based adversarial detection.

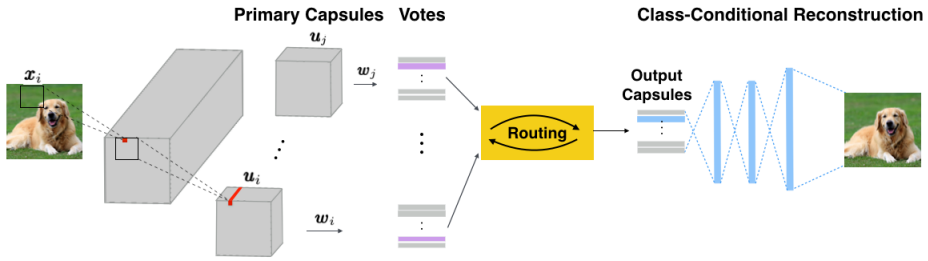


Figure 1: The overview of Capsule Networks: the CapsNet architecture consists of four components, i.e., primary capsule extraction, voting, routing, and class-conditional reconstruction.

The next section introduces background knowledge and related work. Sec. 3 investigates capsule attack. Sec. 4 describes our vote attack. In experimental Sec. 5, we show the effectiveness and efficiency of vote attacks and their successful application to bypass the class-conditional reconstruction based detection. The last section concludes our work.

2 Background Knowledge and Related Work

Capsule Networks The overview of CapsNets is shown in Figure 1. CapsNets first extract primary capsules $\boldsymbol{u}_i$ from the input image $\boldsymbol{x}$ with pure convolutional layers (or CNN backbones). Each primary capsule $\boldsymbol{u}_i$ is then transformed to make a vote $\hat{\boldsymbol{u}}_{j|i}$ for high-level capsules. Next, the dynamic routing process is applied to identify weights c_{ij} for the votes $\hat{\boldsymbol{u}}_{j|i}$ from primary capsules, where $i \in \mathbb{N}_{\leq N}$ corresponds to indices of primary capsules and $j \in \mathbb{N}_{\leq M}$ to indices of high-level capsules. The final output capsules are computed as $\boldsymbol{v}_j = f_s(\boldsymbol{s}_j) = f_s(\sum_{i=1}^{N} c_{ij} * \hat{\boldsymbol{u}}_{j|i})$ where $f_s(\cdot)$ is a squashing function that maps the length of $\boldsymbol{s}_j$ into the range of $(0, 1)$. The capsules are represented by vectors, the length of which indicates the confidence of the entitys' existence. In the training phase, the class-conditional reconstruction net reconstructs the input image from the capsule corresponding to the ground-truth class t, i.e., $\hat{\boldsymbol{x}} = r(\boldsymbol{v}_t)$. The reconstruction error $d(\boldsymbol{x}) = \|\hat{\boldsymbol{x}}, \boldsymbol{x}\|_2$ works as a regularization term. The notations above will be used across the paper.

To improve CapsNets Sabour et al. (2017), various routing mechanisms have been proposed, to name a few Hinton et al. (2018); Zhang et al. (2018); Hahn et al. (2019); Tsai et al. (2020). The advanced techniques of building CNNs have also been integrated into CapsNets successfully Rajasegaran et al. (2019); Phaye et al. (2018). For instance, to improve the performance of CapsNets to the state-of-the-art level, the CNN backbones are applied to extract more accurate primary capsules.

Hinton et al. (2018); Hahn et al. (2019) demonstrate the high adversarial robustness of CapsNets. However, the work Michels et al. (2019) shows it does not hold for all attacks. In addition, many defense strategies proposed for CNNs are circumvented by later defense-aware white-box attacks Athalye et al. (2018). Given the previous research line, we argue that it is necessary to explore CapsNet architecture-aware attacks before we have any claim on the robustness of CapsNets. To our knowledge, there is no attack specifically designed for CapsNets in current literature.

Adversarial Attacks Given the logits $Z(\boldsymbol{x})$ of an input in a CNN, attacks fool the model by create perturbations to increase the loss $l(Z(\boldsymbol{x}+\delta), y) = log(\sum_{j=1}^{k} exp(Z(\boldsymbol{x}+\delta)_j) - Z(\boldsymbol{x}+\delta)_y$ where δ is constrained in a L_p ball. The one-step *Fast Gradient Sign Method* (FGSM Goodfellow et al. (2015)) creates perturbations as

$$\delta = \epsilon \cdot \text{sign}(\nabla_\delta l(Z(\boldsymbol{x}+\delta), y)) \tag{1}$$

The multi-step *Projected Gradient Descent* (PGD Madry et al. (2018)), is defined as

$$\delta \leftarrow \text{clip}_\epsilon(\delta + \alpha \cdot \text{sign}(\nabla_\delta l(Z(\boldsymbol{x}+\delta), y))) \tag{2}$$

Other popular multi-step attacks also include *Basic Iteractive Method* (BIM Kurakin et al. (2017)) *Momentum Iterative Method* (MIM Dong et al. (2018)). Besides, C&W attack Carlini & Wagner (2017b) and Deepfool Moosavi-Dezfooli et al. (2016) are popular strong attacks on the L_2-norm constraint.

Adversarial Detection Many CNN-based adversarial detection methods were easily bypassed by constructing new loss functions Carlini & Wagner (2017a). Adversarial images are not easily detected. The most recent work Qin et al. (2020) leverages class-conditional reconstruction net of CapsNet to detect adversarial images. Given any input $\boldsymbol{x}$, the predicted class and the corresponding capsule are $f(\boldsymbol{x})$ and $\boldsymbol{v}_{f(\boldsymbol{x})}$, respectively. The input is flagged as an adversarial image, if the reconstruction error is bigger than a pre-defined threshold $d(\boldsymbol{x}) = \left\|r(\boldsymbol{v}_{f(\boldsymbol{x})}), \boldsymbol{x}\right\|_2 > \theta$. The threshold θ is defined by measuring the reconstruction errors between benign validation images and its reconstructions. The choice of θ involves a trade-off between false positive and false negative detection rates. Instead of tuning this parameter, the work Qin et al. (2020) simply sets it as the 95th percentile of validation distances. A strong detection-aware reconstructive attack is also proposed to verify the effectiveness of the proposed detection method in Qin et al. (2020). The reconstructive attack is a two-stage optimization method where it first creates a perturbation δ to fool the prediction as in Equation 3, and updates the perturbation further to reduce the reconstruction error as in Equation 4.

$$\delta \leftarrow \text{clip}_\epsilon(\delta + \alpha \cdot \beta \cdot \text{sign}(\nabla_\delta l(f(\boldsymbol{x}+\delta), y))) \tag{3}$$

$$\delta \leftarrow \text{clip}_\epsilon(\delta + \alpha \cdot (1-\beta) \cdot \text{sign}(\nabla_\delta \left\|r(\boldsymbol{v}_{f(\boldsymbol{x})}), \boldsymbol{x}\right\|_2) \tag{4}$$

where α is the step size, and β is a hyper-parameter to balance the losses in the two stages.

3 Capsule Attacks on Capsule Networks

In CNNs, under certain constraints, the adversary finds adversarial perturbation of an instance by maximizing the classification loss. In CapsNets, the length of output capsules corresponds to the output probability of the output classes. We attack output capsules by first mapping the length of output capsules to logits $Z(\boldsymbol{x})_j = log(\|\boldsymbol{v}_j\|_2)$ and solving the following equation, called **Caps-Attack**.

$$\delta^* = \max_{\triangle \in \nabla(\boldsymbol{x})} l(Z(\boldsymbol{x}+\delta), y) = \max_{\delta \in \triangle(\boldsymbol{x})} log(\sum_{j=1}^{k} exp(Z(\boldsymbol{x}+\delta)_j) - Z(\boldsymbol{x}+\delta)_y. \tag{5}$$

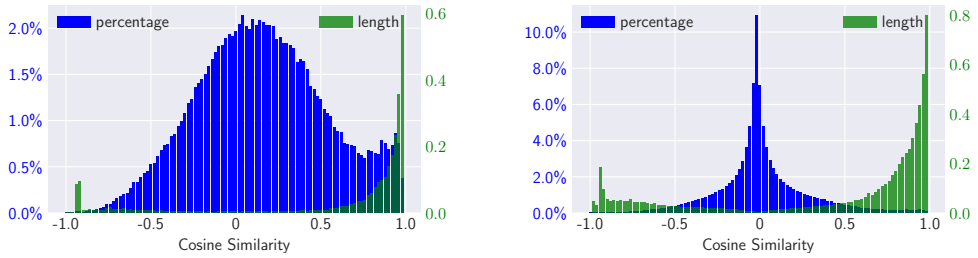
where $\triangle(\boldsymbol{x})$ corresponds to L_p ball of the input instance $\boldsymbol{x}$. It is challenge to solve the optimization exactly. In practice, many empirical methods have been proposed to approaximate the solution, e.g., FGSM Goodfellow et al. (2015), PGD Madry et al. (2018).

In CapsNets, the primary capsule $\boldsymbol{u}_i$ can make a positive or negative vote for the j-th class or abstain from voting. It depends on the relationship between $\boldsymbol{v}_j$ and $\hat{\boldsymbol{u}}_{j|i}$. The vote from $\boldsymbol{u}_i$ for the j-th class can be described as

$$\begin{cases} \text{a positive vote,} & \textit{if } \cos(\boldsymbol{v}_j, \hat{\boldsymbol{u}}_{j|i}) > 0 \\ \text{abstention,} & \textit{if } \cos(\boldsymbol{v}_j, \hat{\boldsymbol{u}}_{j|i}) = 0 \\ \text{a negative vote,} & \textit{if } \cos(\boldsymbol{v}_j, \hat{\boldsymbol{u}}_{j|i}) < 0 \end{cases} \tag{6}$$

How the votes change when CapsNets are fooled by adversarial images? We investigate this question with experiments and visualize the results. We first train a DR-CapsNet on CIFAR10 dataset. With the standard well-trained DR-CapsNet (92.8% test accuracy), we classify all clean images in the test dataset and extract all votes $\hat{\boldsymbol{u}}_{j|i}$ and final predictions $\boldsymbol{v}_j$ of the ground-truth classes. We compute $\cos(\boldsymbol{v}_j, \hat{\boldsymbol{u}}_{j|i})$ in all classifications and split them into 100 equal-width bins in the range of $[-1, 1]$. In each bin, we compute averaged length of all $\hat{\boldsymbol{u}}_{j|i}$. The results are visualized in Figure 2a. The majority of primary capsules make positive votes.

To obtain adversarial images, we apply PGD attack to the classifications of clean images on the DR-CapsNet (17.3% robust accuracy). Similarly, we extract corresponding information from the classifications of adversarial images and visualize the results in Figure 2b. The primary capsules corresponding to $\cos(\boldsymbol{v}_j, \hat{\boldsymbol{u}}_{j|i}) \approx 0$ are dead capsules since they have only tiny impact on final prediction. The adversarial images make primary capsules dead by manipulating the votes and the weights of them. Concretely, the votes on adversarial images are $\hat{\boldsymbol{u}}'_{j|i}$. The iterative routing process identify c'_{ij} as weights for the votes, which is manipulated by adversary images indirectly so that the



(a) On Clean Images

(b) On Adversarial Images

Figure 2: The cosine similarity between votes and final predictions are shown. On clean images, most votes make positive contributions to the ground-truth classes. The most cosine similarity values are zeros on adversarial images. Adversarial images make most votes invalid.

final prediction $\boldsymbol{v}'_j = \sum_{i=1}^{N} c'_{ij} * \hat{\boldsymbol{u}}'_{j|i}$ is orthogonal to most votes $\hat{\boldsymbol{u}}'_{j|i}$. In other words, the adversarial images make the majority of primary capsules invalid for the given ground-truth class.

4 Vote Attacks on Capsule Networks

In Caps-Attack, the solution of Equation 5 mainly based on the gradients $\nabla_\delta l(Z(\boldsymbol{x}+\delta), y)$. 1) **Limited Effectiveness**: Most of the resulting votes make no contribution to the probability of the ground-truth class. An attack can be more effective if it can manipulate the majority of primary capsules to make negative votes. 2) **Low efficiency**: The multi-step gradient-based attacks require many times forward and backward passes on the whole CapsNet to obtain the solution. They are computationally expensive due to the costly routing mechanism, e.g., under PGD attack.

To overcome the two weaknesses above, we propose an efficient and effective **vote attack**. When the votes from primary capsules are attacked directly, the weights identified by the dynamic routing are also misled since the routing is based on the votes. Together with wrong weights, the attacked votes can lead to final misclassifications with high probabilities.

Different from CNNs, the votes in CapsNets are not only hidden representations, but correspond to meanful semantic (i.e., output classes). Given an input-label pair $(\boldsymbol{x}, y)$, the N votes from primary capsules are $\hat{\boldsymbol{u}}_{-|i} = f_v^i(\boldsymbol{x})$ where $i \in \mathbb{N}_{\leq N}$. The averaged vote can be squashed into prediction $\hat{y}$. Formally, the vote attack on this instance can be described as

$$\delta^* = \max_{\delta \in \triangle(\boldsymbol{x})} l(log(f_s(\frac{1}{N}\sum_{i=1}^{N} f_v^i(\boldsymbol{x}+\delta))), y). \tag{7}$$

The maximization problem can be solved with projected gradient descent. The proposed vote attack method can reduce the robust accuracy of DR-CapsNets from 16.1% to 4.83%. Similar to Capsule Attack, the Vote Attack formulation can also be sloved approximately with popular attack methods.

We have another two choices when attacking votes of CapsNet directly. **Choices 1**: In Equation 8, we first average the votes and squash the averaged vote. Another choice is to first squash their votes and then average the squashed votes. Our experiments show that this choice is also similarly effective.

$$\delta^* = \max_{\delta \in \triangle(\boldsymbol{x})} l(log(\frac{1}{N}\sum_{i=1}^{N} f_s(f_v^i(\boldsymbol{x}+\delta))), y). \tag{8}$$

Choices 2: Another choice is to average the loss caused by all votes. Instead of opimizing on the loss computed on the squahed averaged vote, we can compute the loss of individual vote seperatedly and average them, namely,

$$\delta^* = \max_{\delta \in \triangle(\boldsymbol{x})} \frac{1}{N}\sum_{i=1}^{N} l(f_s(f_v^i(\boldsymbol{x}+\delta)), y). \tag{9}$$

The loss of each vote can differ from each other significantly. The large part of loss can be caused by a small part votes. In other words, the gradients of received by the input can be caused mainly by a small part too strong votes. This choice is less effective, compared to the one in Equation 8.

5 EXPERIMENTS

In this section, we verify our proposal via empirical experiments. We first show the effectiveness of Vote Attacks from a different perspective, e.g., on CapsNets with a regular training scheme or adversarial training, as well as the transferability of adversarial examples. We also show the efficiency of vote attacks. Besides, we apply vote attacks to bypass the recently proposed CapsNet-based adversarial detection method. All the reported scores are averaged over 3 runs.

5.1 EFFECTIVENESS OF VOTE ATTACKS ON CAPSNETS

Models: We take ResNet18 (the popular state-of-the-art archhitecture) as a CNN baseline. In couterpart CapsNets, we apply resnet18 backbone to extract primary capsules $\boldsymbol{u} \in (64 \times 4 \times 4, 8)$ where the outputs of the backbone are feature maps of the shape $(512, 4, 4)$ and 64 is the number of capsule groups, 8 is the primary capsule size. The primary capsules are transformed to make $64 \times 4 \times 4$ votes $\hat{\boldsymbol{u}} \in (64 \times 4 \times 4, 10, 16)$ with the learned transformation matrices $\boldsymbol{W} \in (64 \times 4 \times 4, 8, 160)$. The size of output capsule is 16, and 10 are the number of output classes (capsules). The votes $\hat{\boldsymbol{u}}$ reach an agreement $\boldsymbol{v} \in (10, 16)$ via the dynamic routing meachnism. The length of 10 output capsules are the probabilites of 10 output classes.

Datasets: The medium datasets CIFAR10 Krizhevsky et al. (2009) and SVHN Netzer et al. (2011) are used. The standard preprocess is applied on CIFAR10 for training: 4 pixels are padded on an input of 32×32, and a 32×32 crop is randomly sampled from the padded image or its horizontal flip. For L_∞-based attacks, the perturbation range is 0.031 (CIFAR10) and 0.047 (SVHN) for pixels ranging in [0, 1]. For L_2-based attacks, the L_2 norm of maximally allowed perturbation is 1.0 for both datasets.

White-Box Attacks We train CNNs and CapsNets with the same standard training scheme where the models are trained with a batch size of 256 for 80 epochs using SGD with an initial learning rate of 0.01 and moment 0.9. The learning rate is set to 0.01 from the 50-th epoch. We apply popular L_∞-based attacks (FGSM Goodfellow et al. (2015), BIM Kurakin et al. (2017), MIM Dong et al. (2018),PGD Madry et al. (2018)) and L_2-based attacks (C&W attack Carlini & Wagner (2017b), Deepfool Moosavi-Dezfooli et al. (2016)) to attack the well-trained models. The hyper-parameters mainly follow the Foolbox tool Rauber et al. (2017). In CapsNet, Capsules and Votes are taken respectively as targets to attack.

Table 1: The robust accuracy of ResNets and CapsNets are shown under popular attacks. Vote attack is significantly more effective than capsule attack and compatible with different underlying attacks.

Model	Target	L_{inf}				L_2	
		FGSM	BIM	MIM	PGD	Deepfool	C&W
On **CIFAR10 Dataset**, the model accuracy are ResNet 92.18(±0.57) and CapsNet 92.80(±0.14).							
ResNet	Logits	16.6(±0.76)	0.15(±0.05)	0(±0)	0(±0)	0.08(±0.05)	0.24(±0.14)
CapsNet	Caps	44.55(±1.6)	24.43(±1.95)	21.69(±2.52)	17.3(±1.35)	26.55(±0.43)	18.91(±1.5)
	Votes	**26.21**(±1.66)	**8.12**(±0.13)	**9.20**(±3.44)	**4.83**(±0.05)	**20.83**(±0.78)	**6.66**(±0.32)
On **SVHN Dataset**, the model accuracy are ResNet 94.46(±0.14) and CapsNet 94.16(±0.02).							
ResNet	Logits	14.57(±2.73)	2.9(±0.47)	0.06(±0.02)	0.06(±0.02)	3.05(±0.45)	2.16(±0.1)
CapsNet	Caps	58.32(±1.34)	50.25(±0.88)	40.09(±1.65)	34.82(±2.11)	45.76(±1.17)	44.29(±1.07)
	Votes	**49.16**(±1.0)	**31.46**(±0.22)	**14.22**(±0.23)	**8.11**(±0.3)	**39.31**(±0.56)	**27.94**(±0.14)

The standard test accuracy and the robust accuracy under different attacks are reported in Table 1. The CapsNets and the counter-part CNNs achieve similar performance on normal test data. Under strong attack PGD, all the classifications of ResNet can be misled. It is less effective to attack output

capsules. Our vote attack can reduce the robust accuracy of CapsNets significantly across different L_∞-based and L_2-based attack methods.

In Sec. 4, we describe two alternative vote attacks. Under the same PGD attack on CIFAR10 dataset, the robust accuracy corresponding to the choice 1 is 4.06$_{(\pm 1.12)}$, As we expect, it is similarly effective. The choice 2 with the robust accuracy 43.31$_{(\pm 2.46)}$ does not work well since the gradients received by inputs dominated by part of votes.

Under vote attack, the robust accuracy of CapsNets is still higher than that of counter-part CNNs. However, we did claim CapsNet is more robust for two reasons. 1) CapsNets possess more network parameters due to transformation matrices. 2) The potential attacks can reduce the robust accuracy further. This study demonstrates that the high adversarial robustness of CapsNet can be a fake sense, and we should be careful to draw any conclusion.

Adversarial Training In this experiment, we verify the effectiveness of vote attack in the context of Adversarial Training. We train models with adversarial examples created by Capsule Attack where PGD with 8 iterations is used. For training a more robust model, we also combine vote attack and capsule attack to create adversarial examples where a new loss from the two attacks is used.

The underlying attack method used in this experiment is PGD with 40 iterations. The model performance is reported in Table 2 under different training schemes. We can observe that the vote attack (corresponding to the last column) is more effective than capsule attack (corresponding to the second last column) under adversarial training. When we include vote attack to improve the adversarial training (AT v.s. AT +Votes), the robust accuracy of CapsNets is increased for both capsule attack and vote attack.

The vote attack only attacks part of the model. During adversarial training, the model can adapt the routing process to circumvent the adversarial perturbations. Therefore, it is not effective to do adversarial training only using vote attack.

Table 2: The robustness of CapsNets with different training schemes: vote attack is also effective to attack models with adversarial training; It can also be applied to improve adversarial training.

Dataset	Traning	ResNet		CapsNet		
		A_{std}	Logits	A_{std}	Caps	Votes
CIFAR10	Natural	92.18$_{(\pm 0.57)}$	0	92.8$_{(\pm 0.14)}$	17.30$_{(\pm 1.35)}$	4.83$_{(\pm 0.05)}$
	AT	79.45$_{(\pm 1.27)}$	43.91$_{(\pm 0.62)}$	75.0$_{(\pm 0.04)}$	45.49$_{(\pm 0.78)}$	43.65$_{(\pm 0.85)}$
	AT + Votes	-	-	76.42$_{(\pm 0.37)}$	49.62$_{(\pm 0.56)}$	44.12$_{(\pm 0.32)}$
SVHN	Natural	94.46$_{(\pm 0.14)}$	0.06$_{(\pm 0.02)}$	94.16$_{(\pm 0.02)}$	34.82$_{(\pm 2.11)}$	8.11$_{(\pm 0.30)}$
	AT	87.9$_{(\pm 0.08)}$	36.05$_{(\pm 0.33)}$	86.0$_{(\pm 0.80)}$	33.40$_{(\pm 1.36)}$	30.44$_{(\pm 1.08)}$
	AT + Votes	-	-	83.89$_{(\pm 0.73)}$	39.13$_{(\pm 0.96)}$	34.92$_{(\pm 0.98)}$

Transferability of Adversarial Examples We also investigate the transferability of adversarial examples created by capsule attack and vote attack. We consider three models, VGG19 Simonyan & Zisserman (2015), ResNet18 and CapsNets. The PGD is used as the underlying attack. We measure the transferability using Transfer Sucess Rate (TSR)

The TSR of different adversarial examples is reported in Table 3. The adversarial examples created on CNNs are more transferable. Especially, the ones created on ResNet18 can be transferred to CapsNets very well. The reason behind this is that CapsNets also the ResNet18 bone to extract primary capsules. By comparing the last two columns in Table 3, we can observe that the adversarial example created by vote attack is more transferable than the ones created by capsule attacks.

Adversarial Robustness on Affine-transformed Data CapsNets learn equivariant visual representations. When inputs are affine transformed, the votes also changes correspondingly. In this experiment, we aim to verify the effectiveness of vote attack when inputs and their votes in Capsnets changed. The model is trained the same as before. We translate the test images with 2 pixels randomly and rotate the images within a given pre-defined degree.

The robust accuracy of affine-transformed images is shown in Table 4. Under different rotation degrees, our vote attack is still effective. It consistently reduces the robust accuracy of CapsNets, when compared to capsule attack.

Table 3: The transferability of adversarial examples created on CNNs and CapsNets: the ones created on CNNs are more transferable than on CapsNets; the ones created with vote attacks are more transferable than the ones with capsule attacks.

		Attacks on Source Model			
		VGG19 (Logits)	ResNet18 (Logits)	CapsNet (Caps)	CapsNet (Votes)
Target Models	VGG19	83.79(±0.18)	93.94(±0.28)	35.64(±0.96)	41.49(±0.19)
	ResNet18	71.81(±1.04)	97.26(±1.84)	37.59(±6.25)	43.45(±8.13)
	CapsNet	80.38(±1.79)	97.53(±0.57)	46.43(±5.56)	55.34(±6.26)

Table 4: When inputs are affine-transformed, the vote attack is still more effective to create adversarial examples than capsule attack.

Model	Target	(0, ±0°)	(±2, ±15°)	(±2, ±30°)	(±2, ±60°)	(±2, ±90°)
ResNet	A_{std}	92.18(±0.57)	85.64(±0.46)	68.11(±1.12)	48.47(±0.30)	42.07(±0.22)
	Logits	0	0	0	0	0
CapsNet	A_{std}	92.8(±0.14)	86.09(±0.39)	69.44(±1.96)	49.37(±2.43)	42.62(±1.64)
	Caps	17.3(±1.35)	5.82(±1.86)	2.89(±1.05)	1.63(±0.51)	1.11(±0.38)
	Votes	**4.83**(±0.05)	**1.15**(±0.38)	**0.54**(±0.22)	**0.32**(±0.16)	**0.23**(±0.08)

5.2 Efficiency of Vote Attacks On CapsNets

In the last subsection, we demonstrate the effectiveness of vote attack from different perspectives. We now show the efficiency of vote attacks. We record the time required by each attack to create a single adversarial example. The time is averaged across the test dataset. A single Nvidia V100 GPU is used.

The required time is reported in Table 5. The column corresponding to A_{std} shows the time required to classify a single input image. Compared to the logit attack in CNNs, capsule attack in CapsNets requires more time to create adversarial examples since the dynamic routing is computationally expensive. Our vote attack can create adversarial images without using the routing part, and reduce the required time significantly. However, the required time is still more than that on CNNs. The reason behind this is that the current deep learning framework is highly optimized on the convolutional operations, less on the voting process.

Table 5: The time required by each attack to create an adversarial example is shown. Vote attacks requiring less time is more efficient than capsule attacks.

Model	Target	A_{std}	FGSM	BIM	MIM	PGD	Deepfool	C&W
ResNet	Logits	4.14*ms*	12.13*ms*	83.34*ms*	165.76*ms*	324.53*ms*	186.77*ms*	409.38*ms*
CapsNet	Caps	5.65*ms*	17.45*ms*	120.75*ms*	242.97*ms*	471.81*ms*	607.84*ms*	612.79*ms*
	Votes		**14.89***ms*	**105.09***ms*	**196.11***ms*	**414.58***ms*	**295.28***ms*	**448.31***ms*

5.3 Bypassing Class-conditional Capsule Reconstruction based Detection

In this experiment, we demonstrate that class-conditional capsule reconstruction based detection can be bypassed by integrating our vote attack in the detection-aware attack method. The state-of-the-art Capsnets often remove the reconstruction part since it requires many parameters and does not improve performance when a backbone is used to extract primary capsules. Hence, following Qin et al. (2020), we use original CpasNet architecture Sabour et al. (2017) for this experiment.

In **CapsNets**, two standard convolutional layers, Conv1(C256, K9, S1), Conv2(C256, K9, S2), are used to extract primary capsules of shape (32×6×6, 8). The output capsule of shape (10, 16) can be obtained after the dynamic routing process. The output capsules will be taken as input for a reconstruction net with (FC160-FC512-FC1024-FC28×28). In the reconstruction process, only one of the output capsules is activated, others are masked with zeros. Since the input contains the class information, the reconstruction is class-conditional. The capsules corresponding to the ground-truth

class will be activated during training, while the winning capsule (the one with maximal length) will be activated in the test phase. Two baseline models are considered: **CNN+CR** uses the same architecture without routing and group 160 activations into 10 groups where the sum of 16 activations is taken as a logit. The same class-conditional reconstruction mechanism is used. **CNN+R** does not group 160 activations and reconstructs the input from activations without a masking mechanism. More details of the baseline models can be found in Qin et al. (2020).

Given an input, it will be flagged as adversarial examples if its reconstruction error is bigger than a given threshold $d(\boldsymbol{x}) > \theta$. Following Qin et al. (2020), we set θ as 95th percentile of reconstruction errors of benign validation images, namely, 5% False postive rate. For each attack, we report **Success Rate** $S = \frac{1}{K}\sum_i^N (f(\boldsymbol{x}+\delta) \neq y)$ and **Undetected Rate** $R = \frac{1}{K}\sum_i^N (f(\boldsymbol{x}+\delta) \neq y) \cap (d(\boldsymbol{x}) \leq \theta)$. Both detection-agnostic and detection-aware attacks introduced in Sec. 2 are considered.

The results on FMNIST dataset are reported 6. In detection-agnostic attacks, our vote attack achieves a higher success rate than capsule attacks. It indicates that the vote attack is effective across different architectures. The undetected rate is also increased correspondingly. In detection-agnostic attacks, the integration of our vote attack increases the attack success rate and undetected rate significantly. More results on MNIST and SVHN datasets are shown in Appendix A.

Table 6: Different attacks are applied to circumvent the class-conditional reconstruction adversarial detection method. The success rate and undetected rate (S/R) are reported for each attack. The integration of vote attack in the detection-aware attack increase both the success rate and undetected rate significantly.

Attacks	Model	Target	A_{std}	FGSM	BIM	PGD	C&W
Detection-agnostic Attack	CNN+R	Logits	90.95	85.8/63.3	100/80.0	100/75,7	86,4/68.8
	CNN+CR	Logits	91.79	89.4/66.4	97.4/70.4	97.9/67.9	77.3/77.1
	CapsNet	Caps	91.85	40.2/29.3	88.8/53.1	90.6/51.4	70.7/54.1
		Votes		**74.8**/46.1	**94.6**/59.2	**94.7**/55.3	**90.5**/50.1
Detection-aware Attack	CNN+R	Logits	90.95	85.3/77.3	99.7/95.0	100/92.1	-
	CNN+CR	Logits	91.79	89.3/75.9	96.3/82.3	96.2/81.2	-
	CapsNet	Caps	91.85	41.8/37.2	87.9/78.7	89.7/78.2	-
		Votes		**76.8/66.5**	**95.1/85.2**	**95.6/86.1**	-

6 Conclusions

We dive into the inner working of CapsNets and show how it affected by adversarial examples. Our investigation reveals that adversarial examples can make the majority of votes invalid. Based on the investigation, we propose an effective and efficient vote attack to attack CapsNets. The effectiveness of vote attack is verified from four perspectives, the compatibility with other attacks, adversarial training, the transferability of adversarial examples as well as the adversarial robustness on affine-transformed data. Last but not least, we apply our vote attack to increase the undetected rate significantly of the class-conditional capsule reconstruction based adversarial detection.

Even though CapsNets seem to be more robust than counter-part CNNs under our strong attack, it is too early to draw such a conclusion. We conjecture that the robust accuracy of CapsNets can be reduced further. In future work, we will explore more strong attacks as well as the certifications to compare the robustness of CNNs and CapsNets.

References

Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. In *International Conference on Machine Learning (ICML)*, 2018.

Nicholas Carlini and David Wagner. Adversarial examples are not easily detected: Bypassing ten detection methods. In *Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security*, pp. 3–14, 2017a.

Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *2017 ieee symposium on security and privacy (sp)*, pp. 39–57. IEEE, 2017b.

Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 9185–9193, 2018.

Ian J. Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. In *International conference on learning representations (ICLR)*, 2015.

Minghao Guo, Yuzhe Yang, Rui Xu, and Ziwei Liu. When nas meets robustness: In search of robust architectures against adversarial attacks. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2020.

Taeyoung Hahn, Myeongjang Pyeon, and Gunhee Kim. Self-routing capsule networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 7658–7667, 2019.

Geoffrey E Hinton, Sara Sabour, and Nicholas Frosst. Matrix capsules with em routing. In *International conference on learning representations (ICLR)*, 2018.

Alex Krizhevsky et al. Learning multiple layers of features from tiny images. 2009.

Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial examples in the physical world. In *International Conference on Learning Representations (ICLR)*, 2017.

Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. In *International conference on learning representations (ICLR)*, 2018.

Felix Michels, Tobias Uelwer, Eric Upschulte, and Stefan Harmeling. On the vulnerability of capsule networks to adversarial attacks. 2019.

Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, and Pascal Frossard. Deepfool: a simple and accurate method to fool deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2574–2582, 2016.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. 2011.

Sai Samarth R Phaye, Apoorva Sikka, Abhinav Dhall, and Deepti R Bathula. Multi-level dense capsule networks. In *Asian Conference on Computer Vision*, pp. 577–592. Springer, 2018.

Yao Qin, Nicholas Frosst, Sara Sabour, Colin Raffel, Garrison Cottrell, and Geoffrey Hinton. Detecting and diagnosing adversarial images with class-conditional capsule reconstructions. In *International Conference on Learning Representations (ICLR)*, 2020.

Jathushan Rajasegaran, Vinoj Jayasundara, Sandaru Jayasekara, Hirunima Jayasekara, Suranga Seneviratne, and Ranga Rodrigo. Deepcaps: Going deeper with capsule networks. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 10725–10733, 2019.

Jonas Rauber, Wieland Brendel, and Matthias Bethge. Foolbox: A python toolbox to benchmark the robustness of machine learning models. In *Reliable Machine Learning in the Wild Workshop, 34th International Conference on Machine Learning*, 2017. URL `http://arxiv.org/abs/1707.04131`.

Sara Sabour, Nicholas Frosst, and Geoffrey E Hinton. Dynamic routing between capsules. In *Advances in neural information processing systems (NeurIPS)*, pp. 3856–3866, 2017.

Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations (ICLR)*, 2015.

Dong Su, Huan Zhang, Hongge Chen, Jinfeng Yi, Pin-Yu Chen, and Yupeng Gao. Is robustness the cost of accuracy? - a comprehensive study on the robustness of 18 deep image classification models. In *European Conference on Computer Vision (ECCV)*, 2018.

Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian J. Goodfellow, and Rob Fergus. Intriguing properties of neural networks. In *International conference on learning representations (ICLR)*, 2014.

Yao-Hung Hubert Tsai, Nitish Srivastava, Hanlin Goh, and Ruslan Salakhutdinov. Capsules with inverted dot-product attention routing. In *International Conference on Learning Representations (ICLR)*, 2020.

Cihang Xie and Alan Loddon Yuille. Intriguing properties of adversarial training at scale. In *International conference on learning representations (ICLR)*, 2020.

Liheng Zhang, Marzieh Edraki, and Guo-Jun Qi. Cappronet: Deep feature learning via orthogonal projections onto capsule subspaces. In *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5814–5823, 2018.

A BYPASSING CLASS-CONDITIONAL RCONSTRUCTION ON MNIST, FMNIST AND SVHN

Table 7: Your caption. the bold the lowest robust accuracy in each column

DataSet	Model	A_{std}	Attacks	Target	FGSM	BIM	PGD
MNIST	CapsNet	99.41	Detection-agnostic	Caps	13.3/6.3	73.3/31.7	77.9/33.1
				Votes	38.8/15.1	92.3/35.4	93.4/34.1
			Detection-aware	Caps	16.1/13.6	71.7/52.7	77.2/57.8
				Votes	44.8/34.9	92.1/66.8	93.4/67.1
FMNIST	CapsNet	91.85	Detection-agnostic	Caps	40.2/29.3	88.8/53.1	90.6/51.4
				Votes	74.8/46.1	94.6/59.2	94.7/55.3
			Detection-aware	Caps	41.8/37.2	87.9/78.7	89.7/78.2
				Votes	76.8/66.5	95.1/85.2	95.6/86.1
SVHN	CapsNet	91.32	Detection-agnostic	Caps	83.2/78.1	99.1/92.3	99.6/92.2
				Votes	95.5/88.8	99.9/93.2	99.9/93.3
			Detection-aware	Caps	84.2/80.1	97.8/95	97.8/94.7
				Votes	90.6/90.8	100/96.7	100/96.8