A Adversarially Robust and Differentially Private Algorithms

A.1 DP-Adv in details

We first give a high-level framework for our `DP-Adv`. For one iteration

1. Subsample a batch B_t;
2. Generate 1 adversarial example for each benign example in B_t;
3. Differentially privately train on adversarial examples.

Now we give a complete `DP-Adv` algorithm using FGSM for inner maximization and DP-SGD for outer minimization.

Algorithm 3 Differentially Private Adversarial Training [`FGSM + DP-SGD`]

Parameters: initial weights θ_0, learning rate η_t, subsampling probability p, number of iterations T, perturbation bound γ, noise scale σ, gradient norm bound R.

for $t = 0, \dots, T-1$ **do**
  Subsample a batch $B_t \subseteq \{1, \dots, n\}$ with subsampling probability p
  for $i \in B_t$ **do**
    $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \gamma \cdot \operatorname{sign}\left(\nabla_{\boldsymbol{x}_i} \mathcal{L}(f(\boldsymbol{x}_i, \theta_t), y_i)\right)$ ▷ Generate adversarial example
    $g_i \leftarrow \nabla_\theta \mathcal{L}(f(\boldsymbol{x}_i, \theta_t), y_i)$
    $g_i \leftarrow g_i \cdot \min\left\{1, R/\|g_i\|_2\right\}$ ▷ Clip the per-sample gradient
  $g_t \leftarrow \sum_{i \in B_t} g_i$
  $g_t \leftarrow g_t + \sigma R \cdot \mathcal{N}(0, \mathbf{I})$ ▷ Apply Gaussian mechanism
  $\theta_{t+1} \leftarrow \theta_t - \frac{\eta_t}{|B_t|} g_t$

To see that `DP-Adv` is flexible in the choices of DP optimizers and attackers, we write another `DP-Adv` using PGD (10 iterations, with learning rate 0.1) as attacker for inner maximization and DP-Adam for outer minimization. Here $\mathcal{P}_\gamma$ is the projection in PGD. For l_∞ attack, the projection is pixel-wise clipping with bound γ; for l_2 attack, the projection is onto a ball with radius γ.

Algorithm 4 Differentially Private Adversarial Training [`PGD + DP-Adam`]

Parameters: initial weights θ_0, m_0, u_0, learning rate η_t, subsampling probability p, number of iterations T, perturbation bound γ, noise scale σ, gradient norm bound R, momentums β_1, β_2.

for $t = 0, \dots, T-1$ **do**
  Subsample a batch $B_t \subseteq \{1, \dots, n\}$ with subsampling probability p
  for $i \in B_t$ **do**
    for j =1,...10 **do**
      $\Delta_i \leftarrow \mathcal{P}_\gamma\left(\Delta_i + 0.1 \cdot \nabla_\Delta \mathcal{L}(f(\boldsymbol{x}_i + \Delta_i, \theta_t), y_i)\right)$ ▷ Generate adversarial example
    $\boldsymbol{x}_i \leftarrow \boldsymbol{x}_i + \Delta_i$
    $g_i \leftarrow \nabla_\theta \mathcal{L}(f(\boldsymbol{x}_i, \theta_t), y_i)$
    $g_i \leftarrow g_i \cdot \min\left\{1, R/\|g_i\|_2\right\}$ ▷ Clip the per-sample gradient
  $g_t \leftarrow \sum_{i \in B_t} g_i$
  $g_t \leftarrow \frac{1}{|B_t|}\left(g_t + \sigma R \cdot \mathcal{N}(0, \mathbf{I})\right)$ ▷ Apply Gaussian mechanism
  $m_t \leftarrow \beta_1 m_{t-1} + (1 - \beta_1)\, g_t$
  $u_t \leftarrow \beta_2 u_{t-1} + (1 - \beta_2)\,(g_t \odot g_t)$
  $\theta_{t+1} \leftarrow \theta_t - \eta_t m_t / \left(\sqrt{u_t}\right)$

B Omitted Experimental Details

B.1 MNIST

For MNIST, we use the standard CNN in `Privacy` and `Opacus` libraries. I.e. we use a convolutional layer with 16 channels, kernel size 8, stride 2 and padding 3, followed by ReLU

and max-pooling with kernel size 2 and stride 1. Then we apply another convolutional layer with 32 channels, kernel size 4 and stride 2, followed by ReLU and the same max-pooling. The hidden representation is flattened and fed into fully-connected layer with 256 units. After ReLU activation, the result is fed into another fully-connected layer with 32 units before the output layer.

For both DP training and DP-Adv training, we train with DP-SGD and the same hyperparameters (i.e. batch size, noise level σ, clipping norm R, learning rate η):

- for $\epsilon = 0.2$, we use $\sigma = 2.5, R = 1.5, \eta = 0.25, |B_t| = 300$.
- for $\epsilon \geq 1$, we use $\sigma = 1.3, R = 1.5, \eta = 0.25, |B_t| = 300$.[6]

B.2 CIFAR10

For CIFAR10, we use the same 2-layer CNN as in Abadi et al. (2016). We pre-train on CIFAR100 with batch size 128, SGD with learning rate 0.01 and momentum 0.9 for 10 epochs. Then we fix the hidden representation. We only alter and train the last layer on CIFAR10 with batch size 250, DP-SGD with $\eta = 0.1, \sigma = 1, R = 2$.

B.3 CelebA

We use the CNN architecture in `https://github.com/vatsalsaglani/MultiLabelClassifier` I.e. we use a convolutional layer with 32 channels, kernel size 3, followed by ReLU and max-pooling with kernel size 2. Then we apply another convolutional layer with 64 channels, kernel size 3, followed by ReLU and the same max-pooling. The hidden representation is flattened and fed into fully-connected layer with 16128 units and then another fully-connected layer with 256 units before the output layer.

C Additional Experiments

C.1 MNIST

C.1.1 Transferability

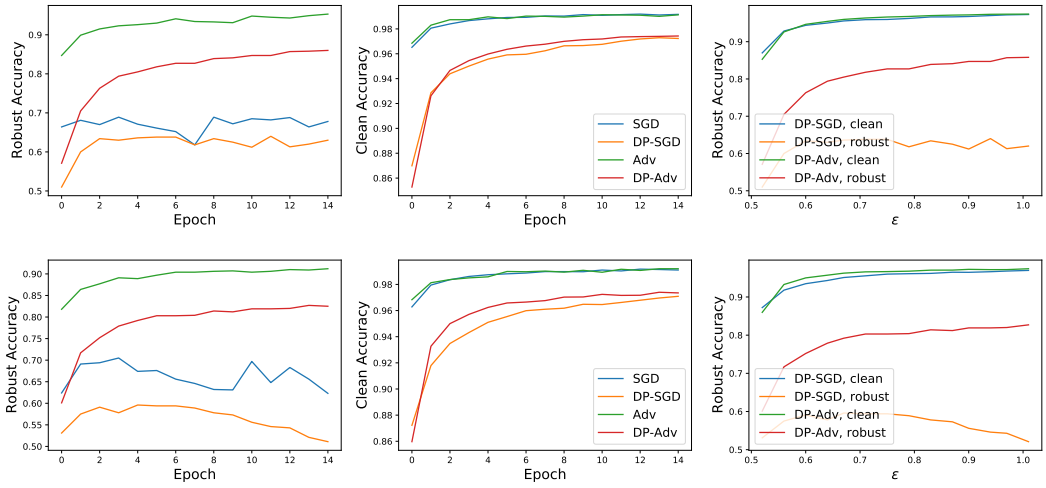

Figure 8: Training against FGSM $l_\infty(0.2)$ attack (upper panel) and PGD $l_2(1)$ attack (lower panel) on MNIST, with $\epsilon = 1$.

[6]These hyperparameters are the same as reported in `https://github.com/pytorch/opacus/blob/main/examples/mnist_README.md` for 95.0% accuracy.

C.1.2 CALIBRATION

Defense/Attack	Clean	PGD_2	$APGD_2$	$AutoAttack_2$
SGD	99.1%	63.1%	61.6%	60.1%
DP-SGD	97.2%	52.7%	51.8%	49.8%
Adv+FGSM	99.2%	90.1%	90.2%	90.0%
DP-Adv+FGSM	97.3%	82.2%	82.0%	81.6%
$Adv+PGD_\infty$	99.2%	90.5%	90.4%	89.8%
$DP\text{-}Adv+PGD_\infty$	97.4%	82.3%	82.2%	82.0%
$Adv+PGD_2$	99.3%	91.2%	90.9%	90.7%
$DP\text{-}Adv+PGD_2$	97.3%	82.5%	82.4%	82.4%

Table 7: Accuracy from different defense and $l_2(1)$ attack methods on MNIST, with $\epsilon = 1$.

C.2 CIFAR10

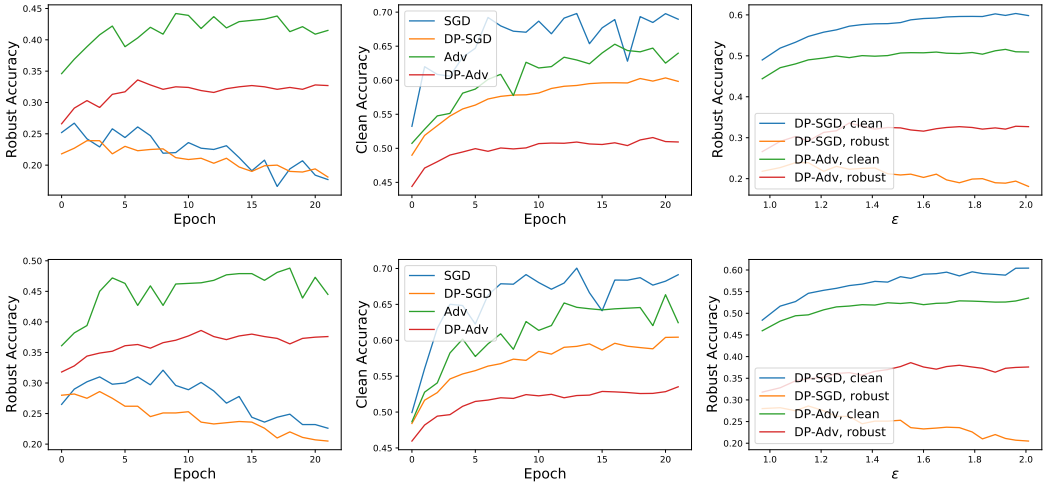

Figure 9: Training against FGSM $l_\infty(4/255)$ attack (upper panel) and PGD $l_2(100/255)$ attack (lower panel) on CIFAR10, with $\epsilon = 1$.

C.2.1 TRANSFERABILITY

Defense/Attack	Clean	FGSM	PGD_∞	BIM	$APGD_\infty$	$AutoAttack_\infty$
SGD	69.0%	17.7%	9.9%	13.4%	9.5%	9.2%
DP-SGD	64.0%	18.1%	12.8%	14.7%	11.6%	11.5%
Adv+FGSM	64.3%	41.5%	39.5%	40.7%	39.6%	37.1%
DP-Adv+FGSM	51.4%	32.7%	30.9%	31.2%	30.3%	27.5%
$Adv+PGD_\infty$	66.9%	42.6%	40.0%	41.8%	39.8%	38.0%
$DP\text{-}Adv+PGD_\infty$	55.7%	31.9%	30.0%	30.5%	29.9%	28.0%
$Adv+PGD_2$	63.3%	37.6%	34.0%	35.4%	33.8%	31.9%
$DP\text{-}Adv+PGD_2$	54.3%	31.1%	27.8%	29.5%	27.7%	25.6%

Table 8: Accuracy from different defense and $l_\infty(4/255)$ attacks on CIFAR10, with $\epsilon = 2$.

C.2.2 CALIBRATION

Defense/Attack	Clean	PGD_2	$APGD_2$	$AutoAttack_2$
SGD	69.0%	21.4%	21.2%	21.1%
DP-SGD	64.0%	22.9%	22.9%	22.3%
Adv+FGSM	64.3%	46.2%	46.2%	44.2%
DP-Adv+FGSM	51.4%	36.2%	35.9%	34.1%
Adv+PGD_∞	66.9%	48.6%	48.3%	46.5%
DP-Adv+PGD_∞	55.7%	37.3%	37.2%	35.1%
Adv+PGD_2	63.3%	44.5%	44.5%	42.9%
DP-Adv+PGD_2	54.3%	37.6%	37.3%	35.1%

Table 9: Accuracy from different defense and $l_2(100/255)$ attacks on CIFAR10, with $\epsilon = 1$.

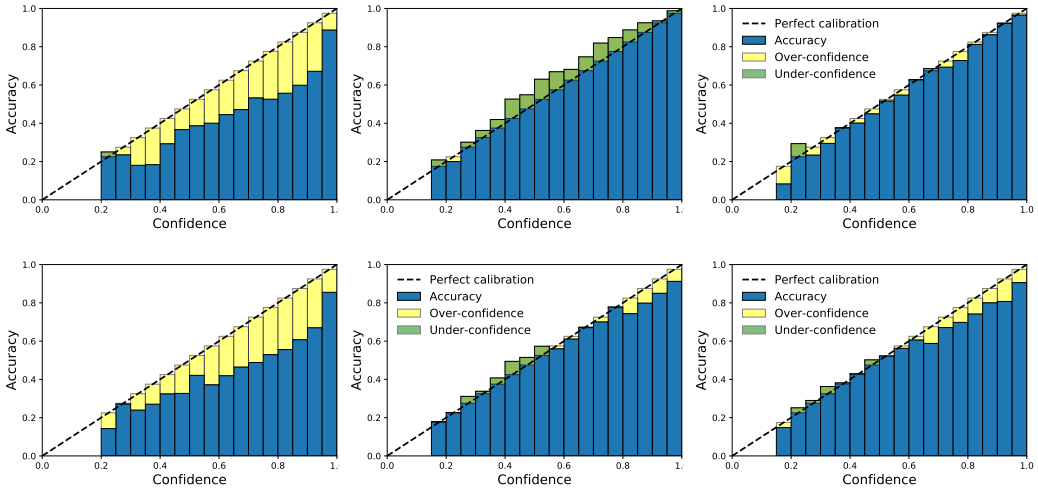

Figure 10: Reliability diagrams on CIFAR10. [Top panel] Left: non-DP SGD. Mid: non-DP Adv by PGD_∞. Right: non-DP Adv by PGD_2. [Bottom panel]: DP variants of top panel.