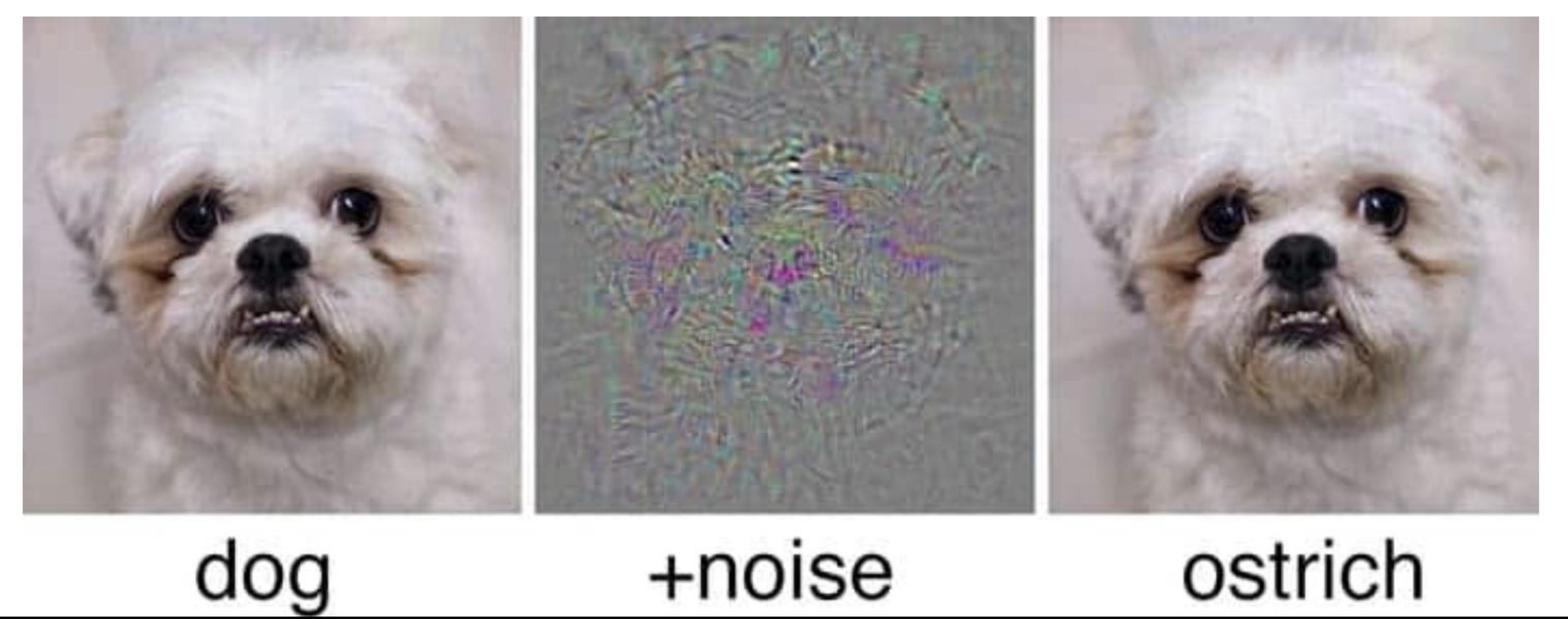
Splitting the Difference on Adversarial Training

Matan Levi – Phd. Student @ Department of CS, BGU | Staff Research Scientist, GenAI @ IBM Research

Prof. Aryeh Kontorovich – Full Profersor @ Department of CS, BGU

Background - Adversarial Examples

- Deep Neural Networks were shown to be extremely vulnerable to small crafted perturbations to their inputs
- These examples are called adversarial examples

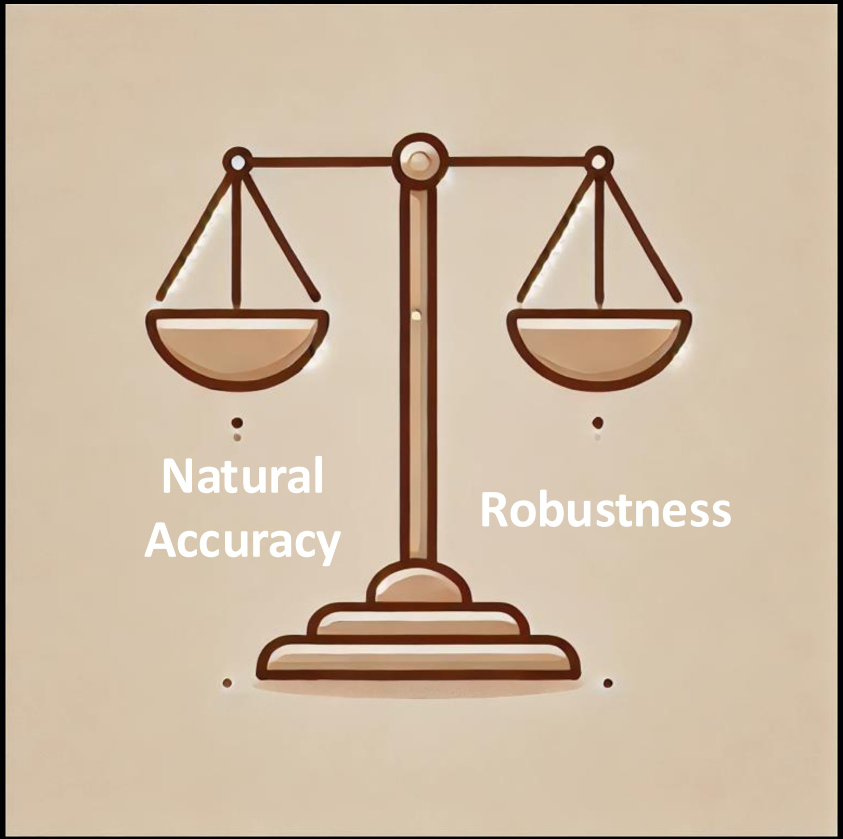


Background - Adversarial Training

- **Adversarial Training is one of the most effective methods to enhance a model's robustness**
- **The basic idea – models are trained with the adv. examples alongside original data**
- **Adversarial examples are assigned the same label as the original class**

Problem – The Natural-Robust Tradeoff

- Tsipras et al. argued that robustness may be at odds with natural accuracy, and usually trade-off is inherent



Research Question

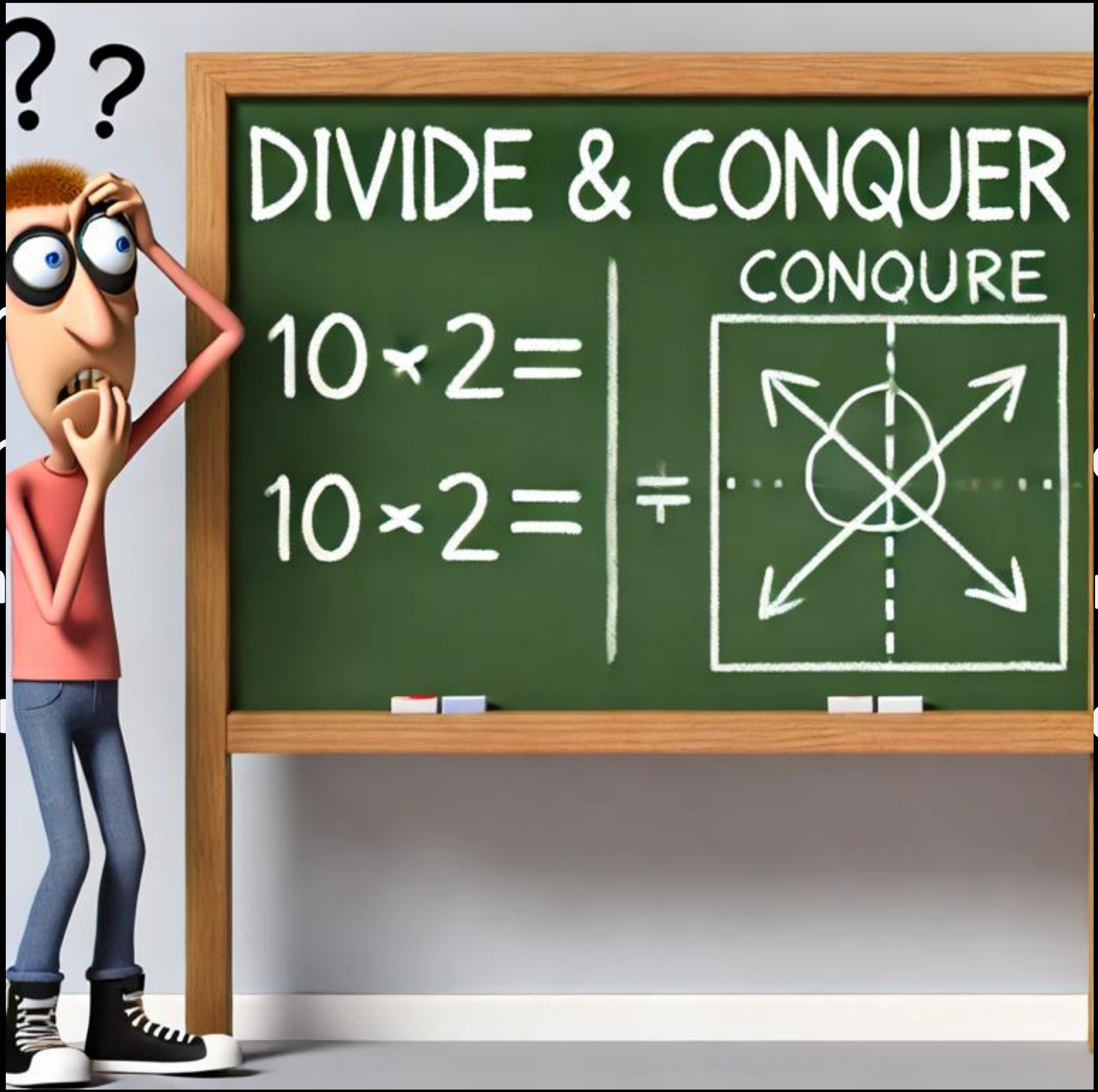
In Adversarial Training, How Can One Avoid Significant Natural Accuracy Degradation While Still Achieving Significant Robustness?

Motivation

We argue that this tradeoff indeed usually happens when adv. examples are assigned to the same class as the natural ones

What will happen if we completely separate the adversarial and original classes?

Motivation



We argue that this tr ill happen if we

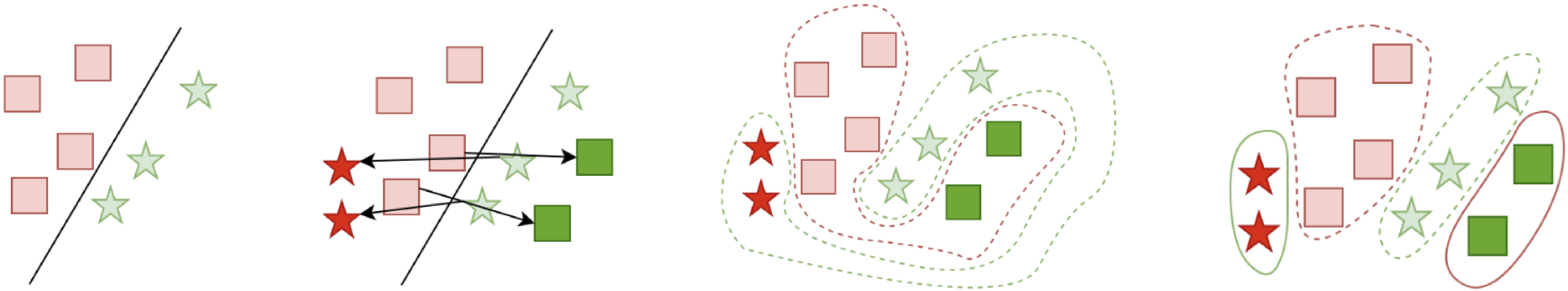
usually happens wher ely separate the

are assigned to the sa rial and original

natural or classes?

Our Approach

Double Boundary
Adversarial Training (DBAT)



(a) (b) (c) (d)

DBAT – High Level Overview

1. **Given a training set $S = \{(x_i, y_i)\}_{i=1}^{n}$ with C classes $Y = \{0,1, \dots, C-1\}$**
2. **we define a new class space $Y_{BDAT} = \{1, 2, \dots, C-1, C, C+1, \dots, 2C-1\}$**
3. **During the adversarial training process, our goal is to learn additional classes, one for each in the original class set:**
   - For each natural example (x_i, y_i), we generate an adversarial example and the corresponding adversarial class $(x_i', y_i + C)$ using Targeted-PGD

Our Approach – DBAT Algorithm

Generate Adversarial examples with targeted PGD

Save the adversarial example with its specific adversarial class label

Algorithm 1 DBAT Training

Input: $\mathcal{S} = \{(x_i, y_i)\}_{i=1}^{n}$ with $\mathcal{C}$ classes, and model f_θ
Parameters: Batch size m, perturbation size ε, attack step size τ, current iteration index k (zero-initialized), and learning rate α
repeat
  Fetch mini-batch $X_s = \{x_j\}_{j=1}^{m}$, $Y_s = \{y_j\}_{j=1}^{m}$
  Initialize $X^{'} = \{\}, Y^{'} = \{\}$
  for $j = 1$ **to** m (in parallel) **do**
    # Generate an adv. example
    $y'_j =$ Select random label uniformly from $\{0, 1, ..., \mathcal{C} - 1, \mathcal{C}, ..., \mathcal{C} \cdot 2 - 1\} / \{j, j + \mathcal{C}\}$
    $x'_j = \text{targeted-PGD}(x_j, y'_j, \varepsilon, \tau, f_\theta)$
    # Save the adv. example with the adv. class label
    $X^{'} = X^{'} \cup \left\{x'_j\right\}$
    $Y^{'} = Y^{'} \cup \{y_j + \mathcal{C}\}$
  end for
  $\theta = \theta - \alpha \cdot \nabla_\theta \ell(X_s \cup X^{'}, Y_s \cup Y^{'})$
  $\theta' = \dfrac{\theta' \cdot k + \theta}{k + 1}$
  $k = k + 1$
until stopping criterion is met

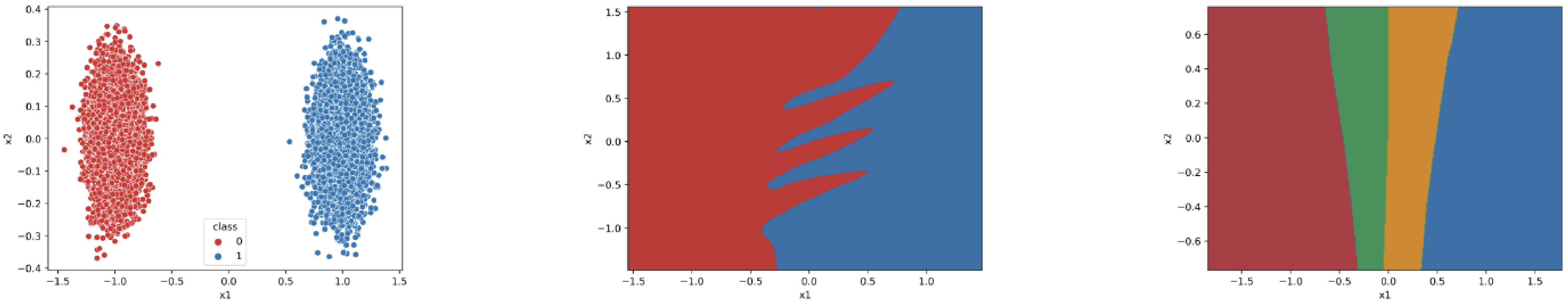
DBAT – Inference

- **At inference time, the model will output a probability vector v of size $|v|$=2⋅C**
- **The dataset originally has only C classes**
- **The final class prediction is taken as the class with the maximum probability**
- **If this class is one of the adversarial classes, we return its natural counterpart**

$$v^* = (\max(v_0, v_C), ..., \max(v_{C-1}, v_{2\cdot C-1})), \tag{1}$$

$$\text{predicted class} = \underset{0 \leq i \leq C}{\operatorname{argmax}} \, v_i^*. \tag{2}$$

Illustrating DBAT's Decision Boundaries using a Synthetic Dataset

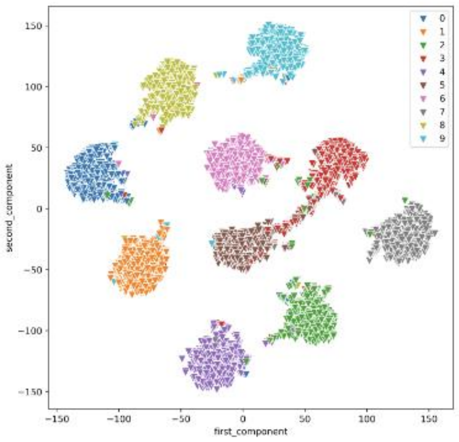


(a) Isotropic Gauss. blobs (boundary $x_1 = 0$) (b) Standard AT decision boundary (c) **DBAT** decision boundaries

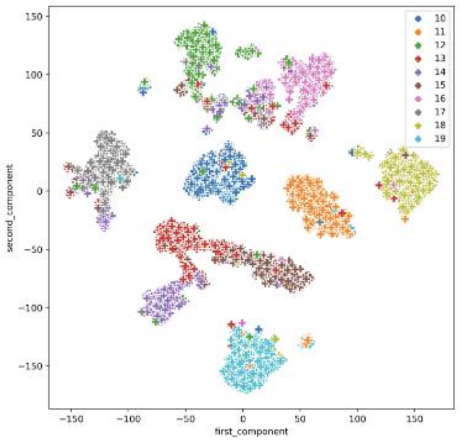
Figure 2: Synthetic dataset viz. on 2-classes dataset (a) of two 2D features each. Adversary: 6-step ℓ_∞-PGD, $\varepsilon = 1.2$, $\delta = 0.2$.

Results

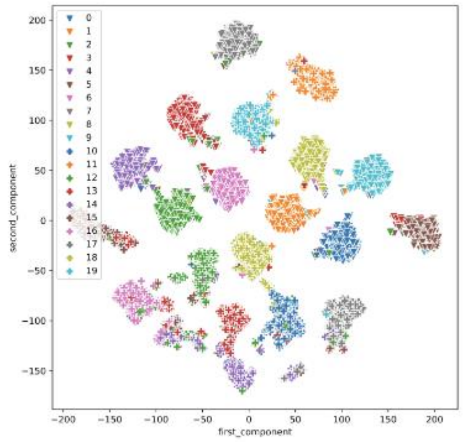
visualizing DBAT using 2D T-SNE on CIFAR-10



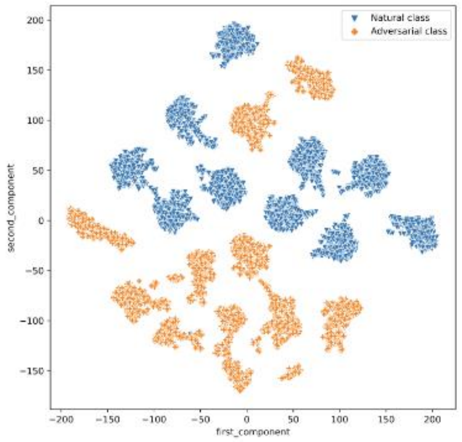
(a) DBAT logits for natural examples and original classes



(b) DBAT logits for adv. examples on newly generated adv. classes



(c) DBAT logits for both natural and adv. examples on all classes



(d) DBAT logits in two colors for natural (blue) and adv. examples (orange).

Results

- White-box PGD
- AutoAttack
- Feature Adversaries

Method	Natural Acc.	PGD	AA
DBAT	**75.18** (↑12.2–18.5%)	27.22	18.17
AT	56.73	28.45	24.12
TRADES	58.24	29.70	24.90
LBGAT	60.64	34.84	29.33
Generalist	62.97	29.49	23.96
HAT	58.73	27.92	23.34
UIAT	59.55	30.81	25.73
CAT	62.84	-	16.82
Natural	79.30	0	0

CIFAR-100

Adversary	Robust Accuracy
KLD	85.9
l_2 Logit Matching	84.5
Feature Adversary [60]	86.8

Feature adversaries CIFAR-10

Method	Natural Acc.	PGD	AA
DBAT (Ours)	**95.01** (↑4–10.1%)	54.61	40.08
AT	85.10	54.46	51.52
TRADES	84.92	55.56	53.08
LBGAT	88.22	54.31	52.86
Generalist	91.03	56.92	52.91
HAT	84.86	52.30	48.85
UIAT	85.01	54.63	49.11
CAT	89.61	73.38	34.78
Natural	95.43	0	0

CIFAR-10

Method	Natural Acc.	PGD	AA
DBAT	**96.86** (↑2.8–6.8%)	49.31	40.49
AT	89.90	49.45	45.25
TRADES	90.35	54.13	49.50
LBGAT	91.80	63.38	40.83
Generalist	94.11	55.29	45.41
HAT	92.06	57.35	52.06
UIAT	93.28	58.18	52.45
CAT	-	-	-
Natural	96.85	0	0

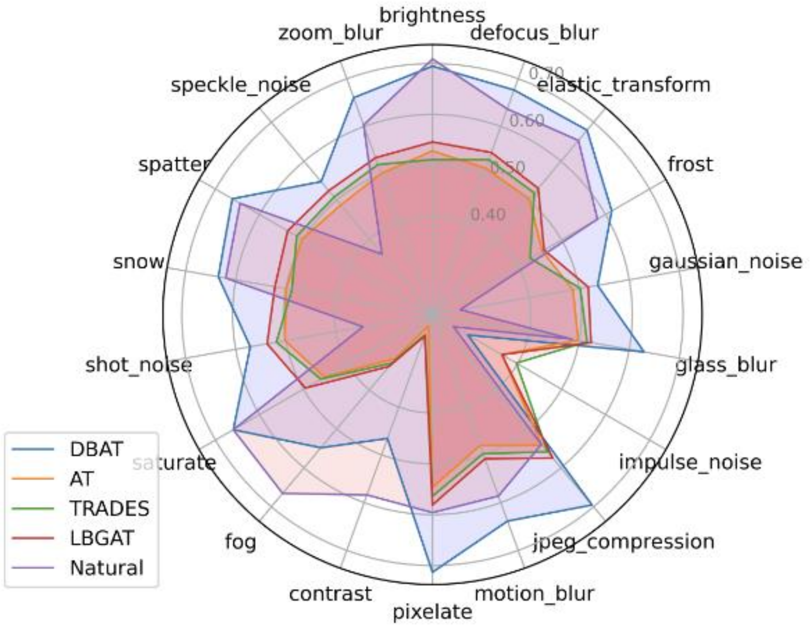
SVHN

Results

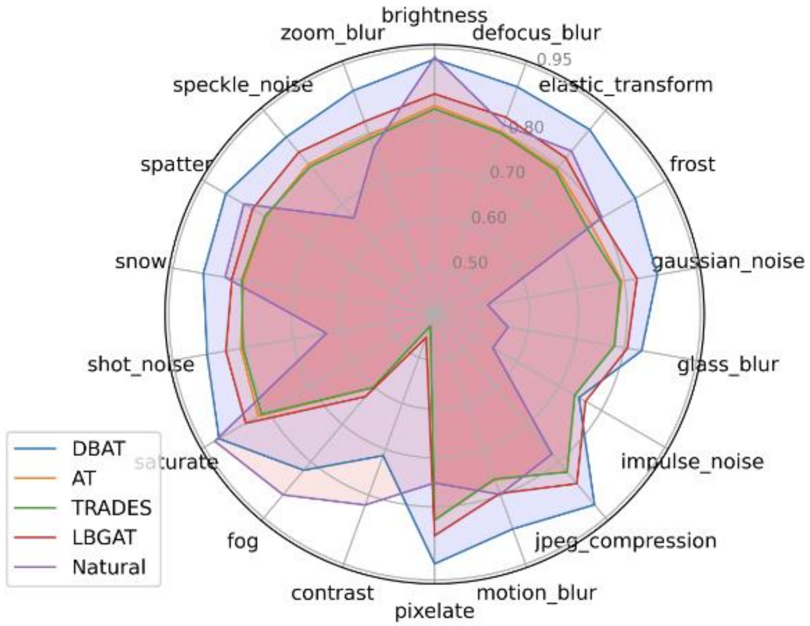
Natural Corruptions:

1. **CIFAR100C:**
   - **Avg. improvement 10.82%**
   - **Max improvement 25.75%**
2. **CIFAR-10C:**
   - **Avg. improvement of 7.96%**
   - **Max improvement 35.19%**

– Statistics compared to the second best approach



CIFAR-100C

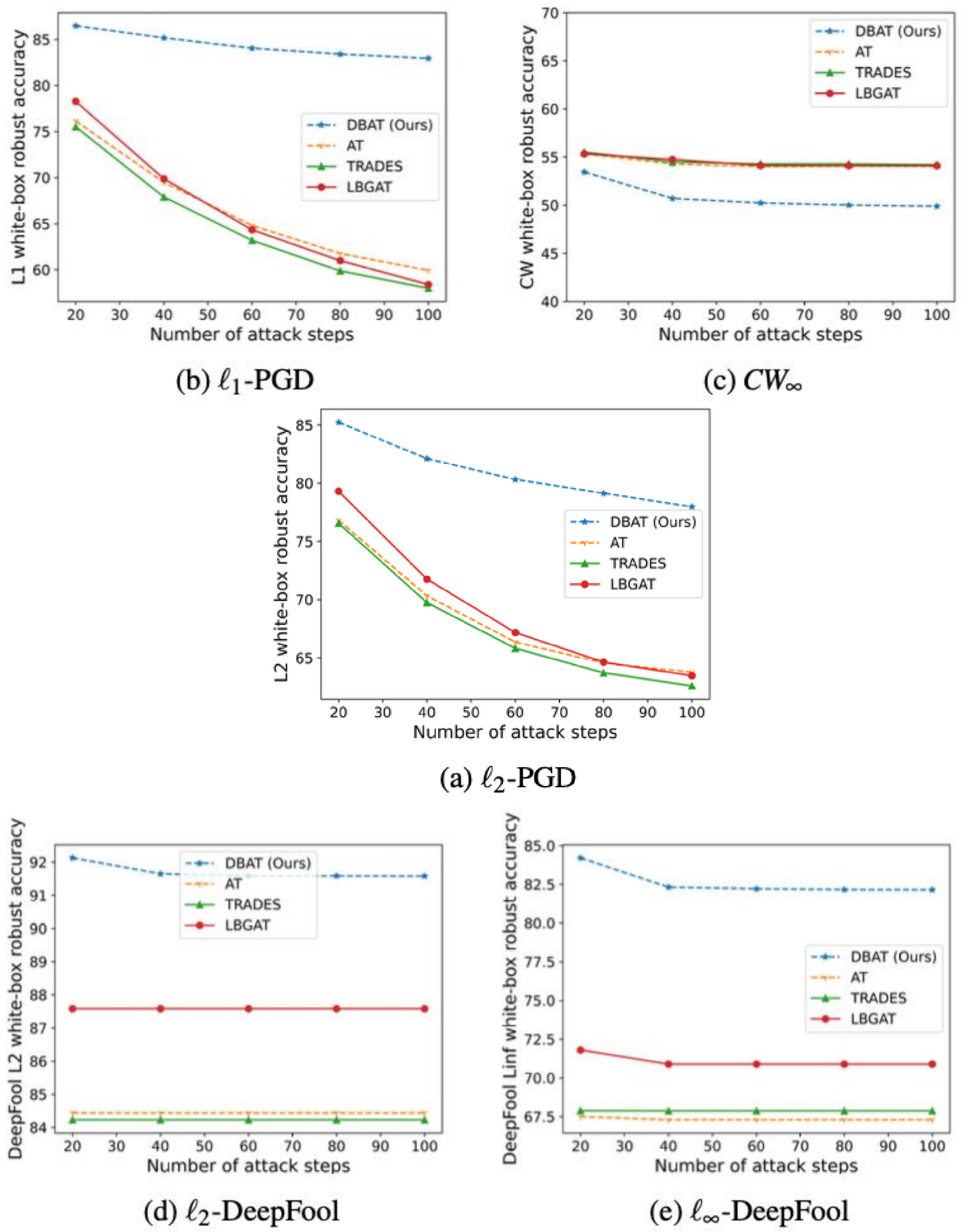


CIFAR-10C

Results

Robustness to unforeseen adversaries:

- l_1-PGD (up to 20% +)
- l_2-PGD (up to 14% +)
- l_2-DeepFool (up to 10% +)
- l_∞-DeepFool (up 16% +)
- CW_∞ (slightly lower)



Results – Clean vs. Robust Tradeoff

TRADES was not able to match DBAT's clean accuracy without losing robust accuracy almost entirely

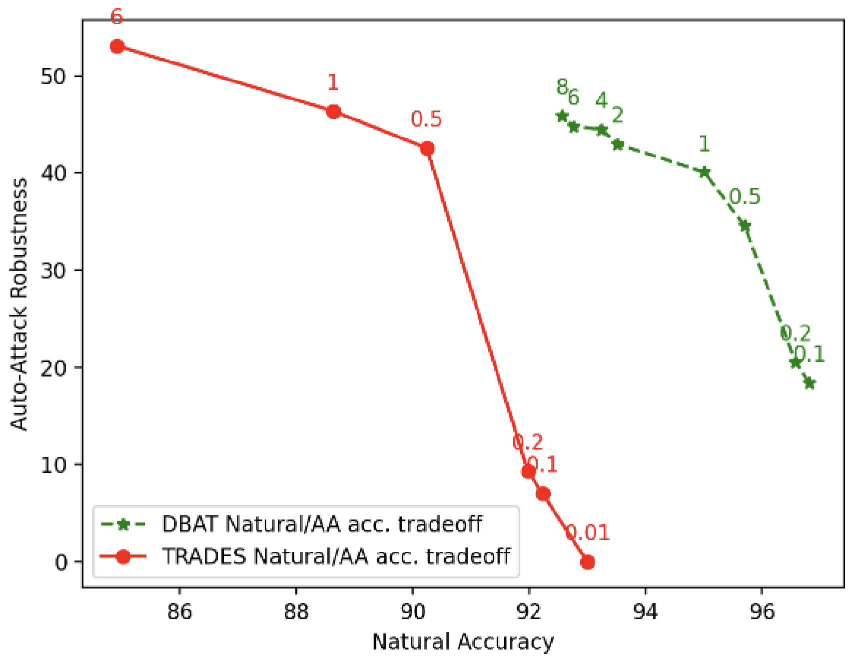


Figure 9: Natural and AutoAttack robust accuracy trade-off, for DBAT and TRADES on CIFAR-10, as we vary the hyper-parameter λ that controls the weight we put on the natural and adversarial classes. The numbers on the graph represent the value of λ for the specific trade-off.

Discussion