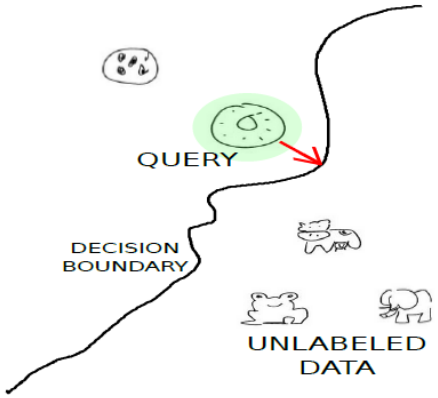
# Adversarial Active Learning for Deep Networks: a Margin Based Approach

Melanie Ducoffe [1] Frederic Precioso [1]

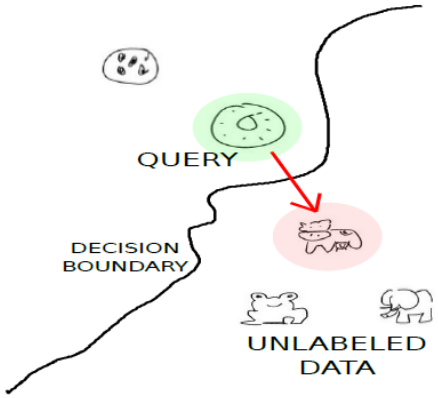
# Motivation

Based on theoretical works on margin theory for active learning, we know that such examples may help to considerably decrease the number of annotations.

However, it requires computing the distance between a sample and the decision boundaries which is not tractable when considering deep networks.




(a) Shortest distance to the boundary

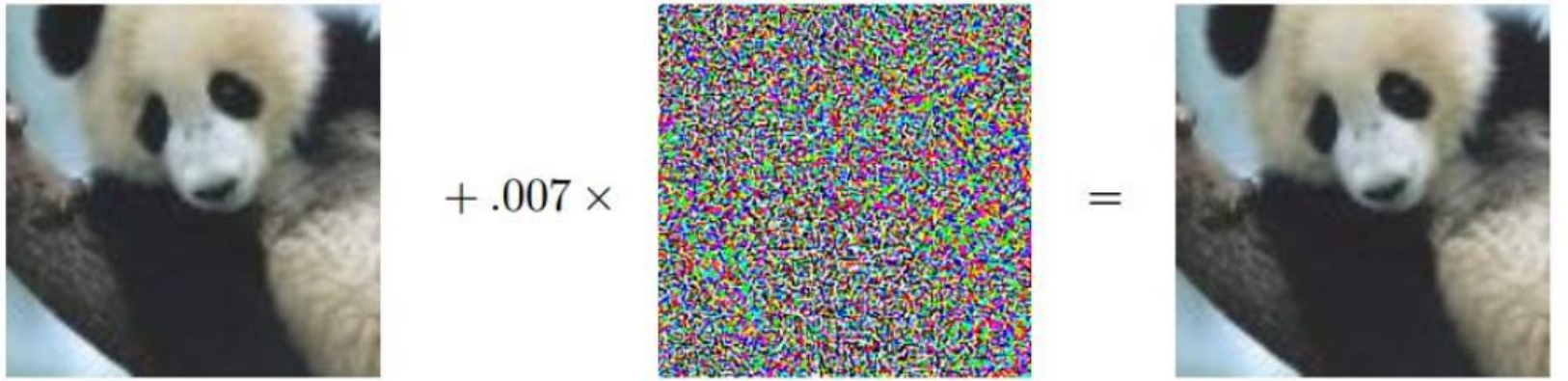



(b) Approximation by the distance to the closest any-other-class sample

# Introduction

■ 对抗样本（Adversarial Examples）：

通过对正常样本添加微小的扰动就可以使得模型产生误判, 而这些扰动基本上不会使得人眼产生任何误判。



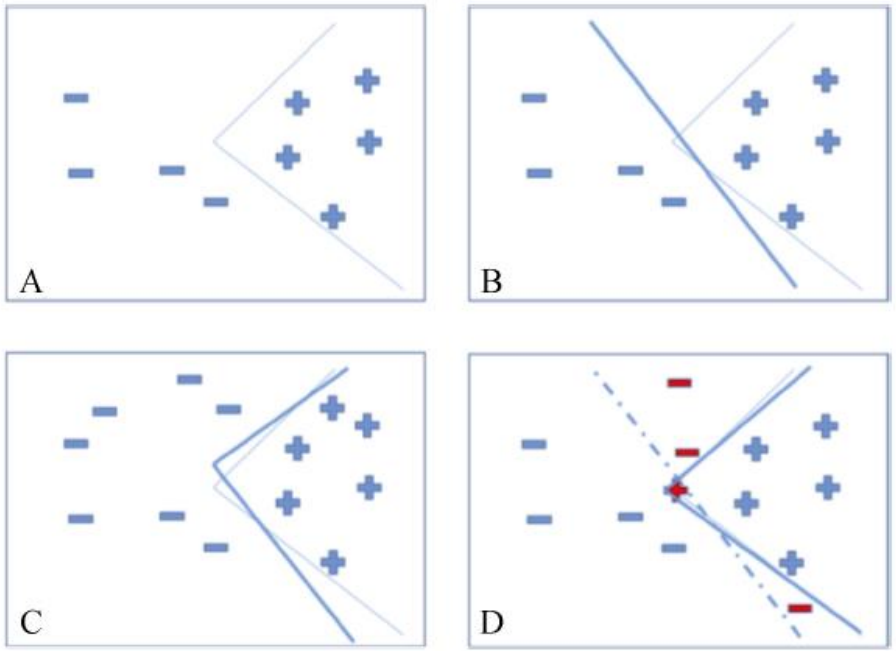

■ 对抗样本的迁移性：

研究表明, 攻击者针对目标模型构造出的对抗样本, 有很大概率可以使得其他的机器学习模型也产生误判。

# Introduction

- 对抗样本的存在性分析：

✓ 在机器学习模型训练中存在基本假设，训练数据与测试数据两者应该是独立同分布的。而对抗样本集本质上是偏离正常样本的集合，破坏了这一假设。




由于训练样本无法覆盖整个输入空间，模型的决策边界与事实的决策边界必然是无法完全重合的，而这些不重合的地方就被成为对抗区域，可能会产生对抗样本。

# Method

DeepFool Active Learning method (DFAL)

- Deep-Fool algorithm
- Active select strategy

# Method

DeepFool Active Learning method (DFAL)

- Deep-Fool algorithm
- Active select strategy

# DFAL:Deep-Fool algorithm

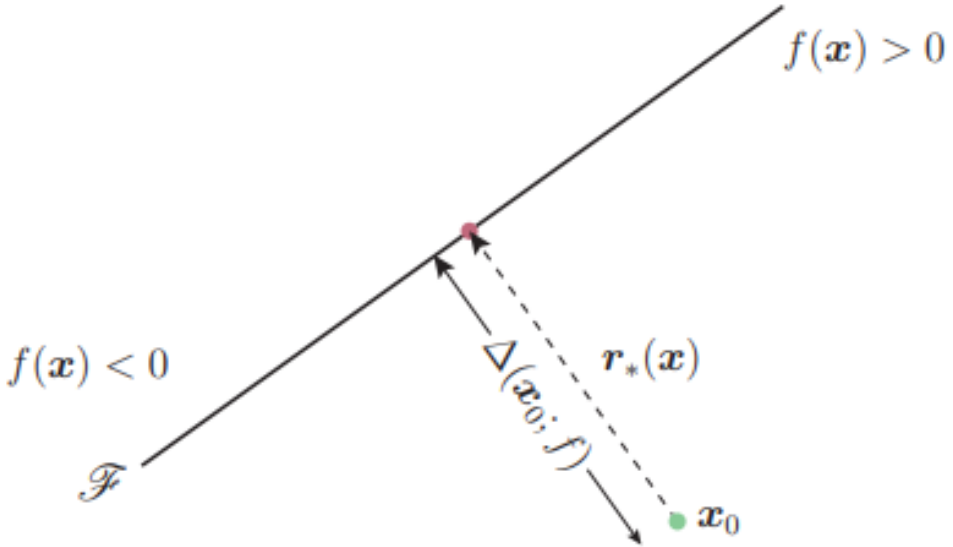



Figure 2: Adversarial examples for a linear binary classifier.

Linear algebra: $\dfrac{|Ax_0 + By_0 + C|}{\sqrt{A^2 + B^2}}$

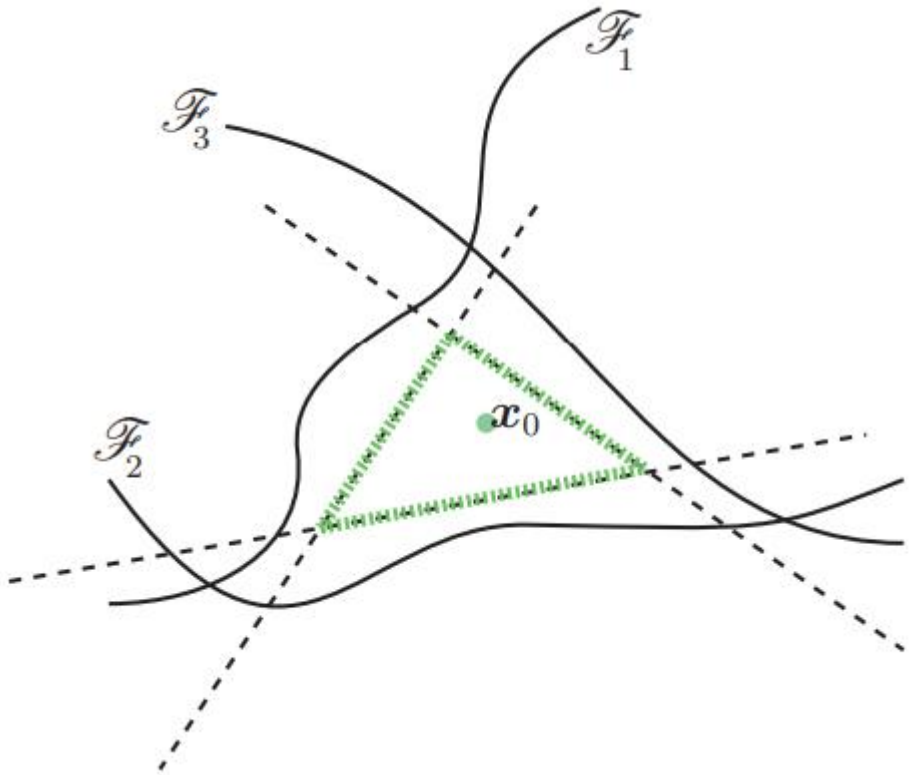
Vectorization: $\dfrac{f(\boldsymbol{x}_0)}{\|\boldsymbol{w}\|_2^2}\boldsymbol{w}$

# DFAL:Deep-Fool algorithm

**Algorithm 1** DeepFool for binary classifiers

**input:** Image $\boldsymbol{x}$, classifier $f$.
**output:** Perturbation $\hat{\boldsymbol{r}}$.
Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
**while** $\text{sign}(f(\boldsymbol{x}_i)) = \text{sign}(f(\boldsymbol{x}_0))$ **do**
    $\boldsymbol{r}_i \leftarrow -\frac{f(\boldsymbol{x}_i)}{\|\nabla f(x_i)\|_2^2} \nabla f(\boldsymbol{x}_i)$,
    $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$,
    $i \leftarrow i + 1$.
**end while**
**return** $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$.

# DFAL:Deep-Fool algorithm




**Algorithm 2** DeepFool: multi-class case

**input:** Image $\boldsymbol{x}$, classifier $f$.
**output:** Perturbation $\hat{\boldsymbol{r}}$.

Initialize $\boldsymbol{x}_0 \leftarrow \boldsymbol{x}$, $i \leftarrow 0$.
**while** $\hat{k}(\boldsymbol{x}_i) = \hat{k}(\boldsymbol{x}_0)$ **do**
  **for** $k \neq \hat{k}(\boldsymbol{x}_0)$ **do**
    $\boldsymbol{w}'_k \leftarrow \nabla f_k(\boldsymbol{x}_i) - \nabla f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
    $f'_k \leftarrow f_k(\boldsymbol{x}_i) - f_{\hat{k}(\boldsymbol{x}_0)}(\boldsymbol{x}_i)$
  **end for**
  $\hat{l} \leftarrow \arg\min_{k \neq \hat{k}(\boldsymbol{x}_0)} \frac{|f'_k|}{\|\boldsymbol{w}'_k\|_2}$
  $\boldsymbol{r}_i \leftarrow \frac{|f'_{\hat{l}}|}{\|\boldsymbol{w}'_{\hat{l}}\|_2^2} \boldsymbol{w}'_{\hat{l}}$
  $\boldsymbol{x}_{i+1} \leftarrow \boldsymbol{x}_i + \boldsymbol{r}_i$
  $i \leftarrow i + 1$
**end while**
**return** $\hat{\boldsymbol{r}} = \sum_i \boldsymbol{r}_i$

# DFAL:Deep-Fool algorithm

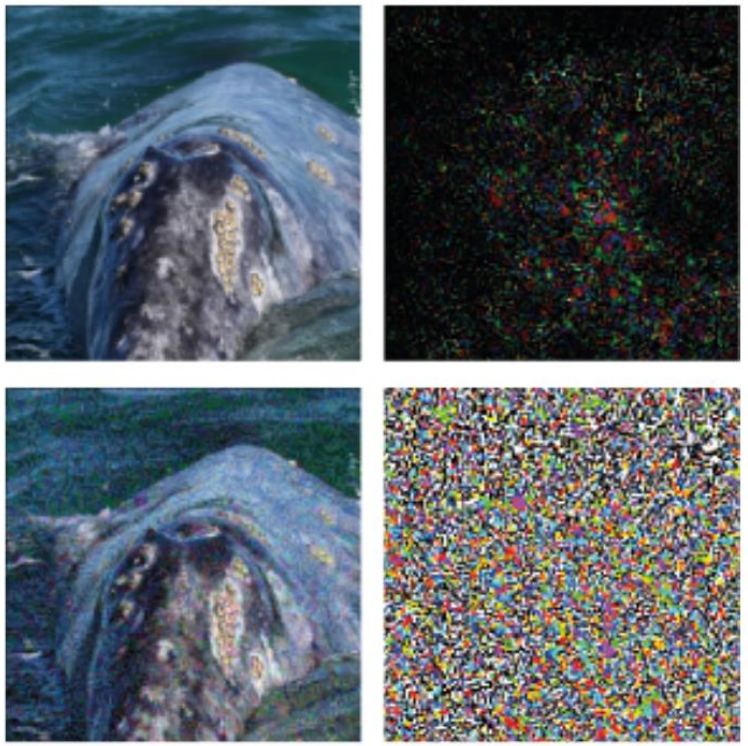
Three advantages of Deep-Fool algorithm :

- it is hyperparameter free (especially it does not need target labels which makes it more compliant with multi-class contexts);
- it runs fast;
- it is competitive with state-of-the-art adversarial attacks.

whale

DeepFool

FGSM



turtle

# Method

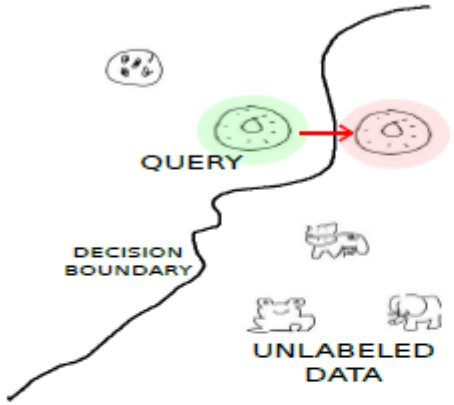
DeepFool Active Learning method (DFAL)

- Deep-Fool algorithm
- Active select strategy

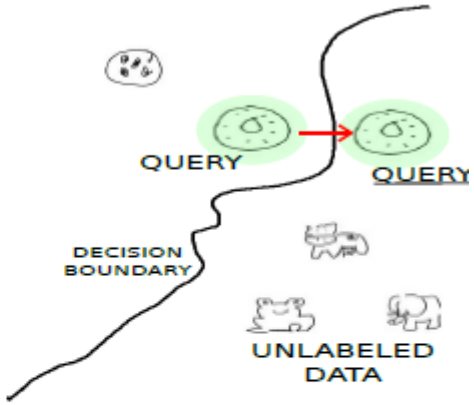
# DFAL:Active select strategy

Active select strategy

- select unlabeled samples with the smallest adversarial perturbation
- add both the less robust unlabeled samples and their adversarial attacks




(c) Approximation by the distance to the adversarial example




(d) DFAL Strategy

# DFAL:Active select strategy

**Algorithm 1** ***DFAL***: DeepFool Active Learning

**Require:** $\mathcal{L}$ set of initial labeled training examples
**Require:** $\mathcal{U}$ set of initial unlabeled training examples
**Require:** $\mathcal{H}$ set of hyper-parameters to train the network
**Require:** $K$ the number of candidates
**Require:** $n_{query}$ the number of data to query
**Require:** p: the $L_p$ norm used ($p = 2$)
**Require:** N: the number of data to label

# init the training set
$k = 0$
$\mathcal{L}_0 = \mathcal{L}$
$\mathcal{U}_0 = \mathcal{U}$
**while** k<N **do**
  # Train the network $\mathcal{A}_k$ given the current labeled training set
  $\mathcal{A}_k$=training($\mathcal{H}, \mathcal{L}_k$)
  # Select randomly a pool of data $\mathcal{S}_k$ of size K
  $\mathcal{S}_k \subseteq \mathcal{U}_k; \ \mid \mathcal{S}_k \mid = K$
  **for** $x_i \in \mathcal{S}_k$ **do**
    #compute adversarial attacks with $L_p$ norms
    $r_i \leftarrow DeepFool(x_i, \mathcal{A}_k; p)$
  **end for**
  # query the labels of the $n_{query}$-th samples $\mathcal{Q}_k$ owing the smallest $L_p$ norm perturbation
  $index_k \leftarrow argsort(< r_i, r_i >_p | \ i = 1..K)$
  $\mathcal{Q}_k \leftarrow \{x_j \mid j \in index_k[0 : n_{query}]\} \cup \{x_j + r_j \mid j \in index_k[0 : n_{query}]\}$
  $\mathcal{L}_{k+1} \leftarrow \mathcal{L}_k \cup \mathcal{Q}_k$
  $\mathcal{U}_{k+1} \leftarrow \mathcal{U}_k \setminus \mathcal{Q}_k$
**end while**

# Experiments

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | **82.77** | 96.23 | **97.71** | 98.02 | – |
| BALD | 51.88 | 91.96 | 93.69 | 94.24 | – |
| CEAL | 71.81 | 94.81 | 96.77 | 97.33 | – |
| CORE-SET | 78.86 | **96.52** | 97.53 | **98.03** | – |
| EGL | 58.44 | 73.86 | 78.57 | 78.57 | – |
| uncertainty | 57.96 | 92.52 | 94.84 | 96.41 | – |
| RANDOM | 77.56 | 92.83 | 94.63 | 95.31 | **99.04** |

(a) *MNIST* (LeNet5 )

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | **84.28** | **96.90** | **97.98** | **98.59** | – |
| BALD | 53.73 | 91.47 | 94.32 | 94.32 | – |
| CEAL | 50.87 | 90.69 | 90.69 | 90.69 | – |
| CORE-SET | 78.80 | 96.68 | 97.46 | 97.88 | – |
| EGL | 37.92 | 91.84 | 93.99 | 93.99 | – |
| uncertainty | 45.57 | 88.36 | 94.27 | 94.60 | – |
| RANDOM | 69.79 | 91.96 | 94.05 | 94.46 | **98.98** |

(b) *MNIST* (VGG8 )

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | 94.62 | 98.50 | 98.98 | 99.10 | – |
| BALD | 93.10 | 97.95 | 97.95 | 97.95 | – |
| CEAL | 84.65 | 98.50 | 99.00 | 99.12 | – |
| CORE-SET | 92.50 | **98.75** | **99.07** | **99.25** | – |
| EGL | 75.07 | 95.47 | 95.47 | 95.47 | – |
| uncertainty | **95.78** | 98.35 | 98.85 | 98.98 | – |
| RANDOM | 95.50 | 98.07 | 98.07 | 98.07 | **99.70** |

(c) *Shoe-Bag* (LeNet5 )

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | **87.73** | 98.53 | **99.30** | 99.50 | – |
| BALD | 86.78 | 95.35 | 97.83 | 97.83 | – |
| CEAL | 84.20 | 98.78 | 99.25 | **99.52** | – |
| CORE-SET | 0.50 | **99.12** | 99.12 | 99.12 | – |
| EGL | 0.50 | 97.28 | 97.28 | 97.28 | – |
| uncertainty | 83.75 | 83.75 | 83.75 | 83.75 | – |
| RANDOM | 86.78 | 95.83 | 97.08 | 97.08 | **99.50** |

(d) *Shoe-Bag* (VGG8 )

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | **82.56** | **89.63** | **90.72** | **91.09** | – |
| BALD | 72.65 | 87.18 | 88.34 | 88.45 | – |
| CEAL | 70.46 | 87.04 | 88.31 | 89.39 | – |
| CORE-SET | 79.58 | 88.93 | 90.54 | 90.53 | – |
| EGL | 57.48 | 64.05 | 64.05 | 69.85 | – |
| uncertainty | 69.24 | 86.89 | 88.54 | 89.09 | – |
| RANDOM | 78.09 | 87.03 | 88.98 | 89.42 | **95.46** |

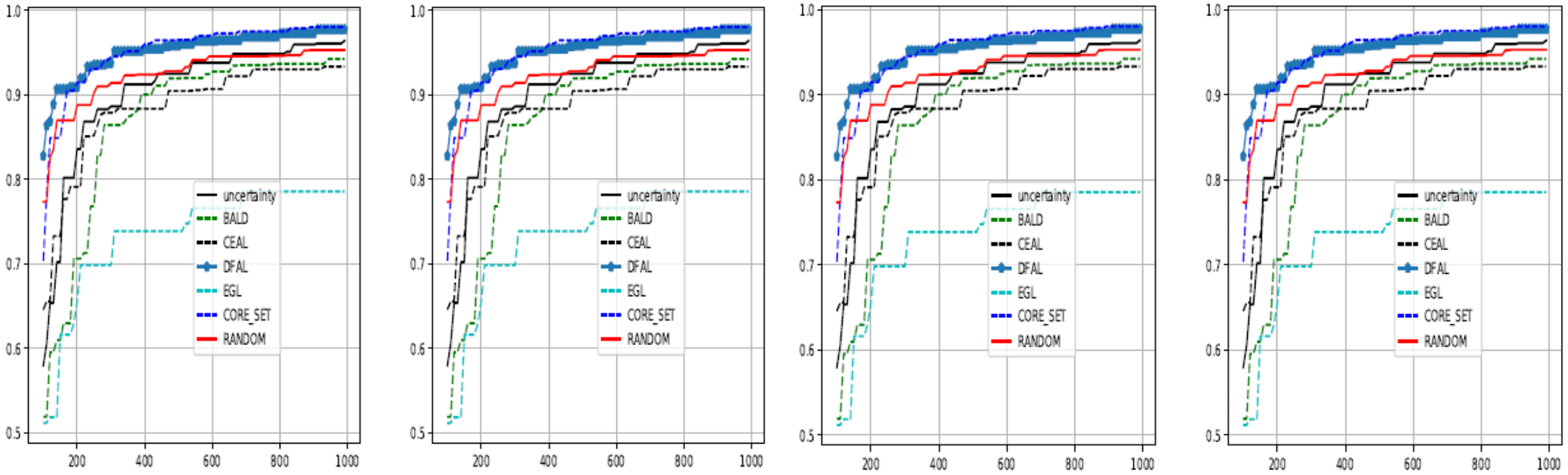
(e) *Quick-Draw* (LeNet5 )

| | Accuracy (%) | | | | |
|---|---|---|---|---|---|
| # annotations | 100 | 500 | 800 | 1000 | All |
| DFAL | **84.23** | **91.52** | **93.16** | **93.91** | – |
| BALD | 82.00 | 89.94 | 91.92 | 92.87 | – |
| CEAL | 64.45 | 79.66 | 85.73 | 88.65 | – |
| CORE-SET | 66.71 | 89.93 | 92.28 | 92.62 | – |
| EGL | 63.12 | 86.80 | 90.06 | 90.06 | – |
| uncertainty | 52.77 | 88.05 | 89.31 | 91.03 | – |
| RANDOM | 78.28 | 88.13 | 89.71 | 89.94 | **96.75** |

(f) *Quick-Draw* (VGG8 )

*Table 1.* Test accuracy achieved by 7 active learning techniques for different number of annotations on LeNet5 and VGG8 .

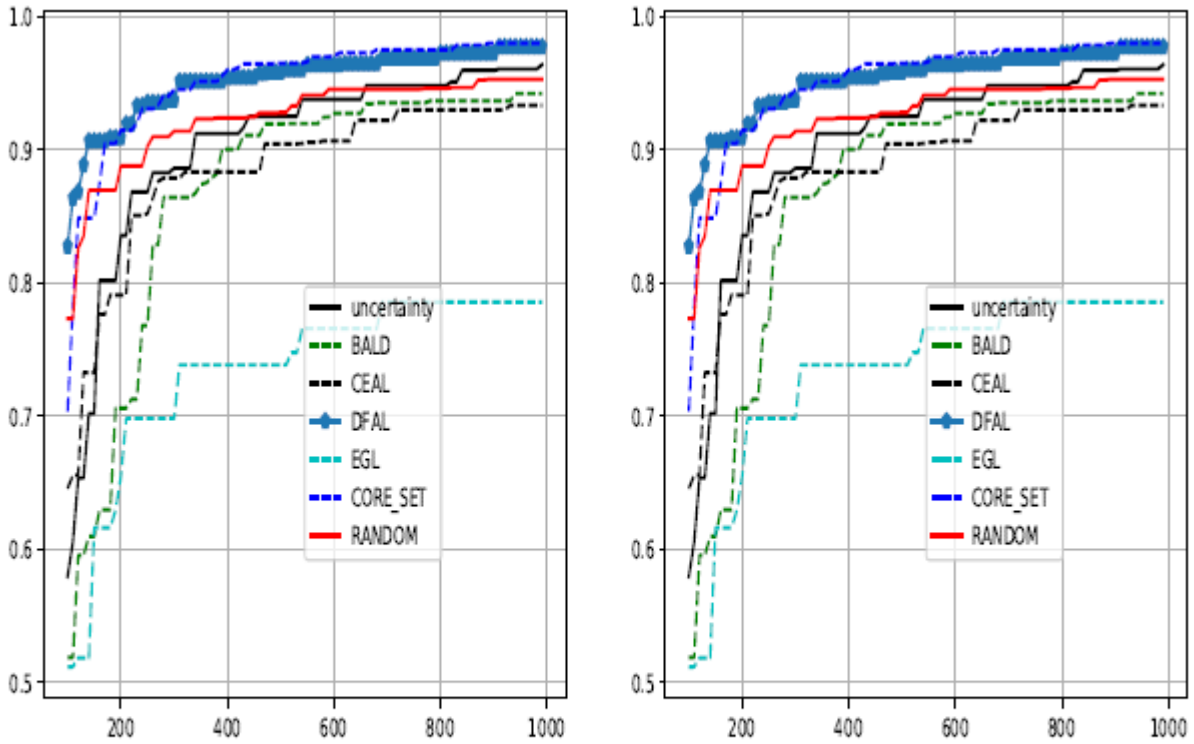
# Experiments




(a) *MNIST* (LeNet5 ) (b) *MNIST* (VGG8 ) (c) *Shoe-Bag* (LeNet5 ) (d) *Shoe-Bag* (VGG8 )

*Figure 2.* Evolution of the test accuracy achieved by 7 active learning techniques( DFAL , BALD , CEAL , EGL , uncertainty and RANDOM ) given the number of annotations

# Experiments




(e) *Quick-Draw* (LeNet5 )




(f) *Quick-Draw* (VGG8 )

*Figure 2.* Evolution of the test accuracy achieved by 7 active learning techniques( DFAL , BALD , CEAL , EGL , uncertainty and RANDOM ) given the number of annotations

# Experiments

| | Accuracy ≥ 99.04 % | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | **1210** | 2410 |
| CORE-SET | 1810 | **1810** |
| CEAL | ≥6000 | ≥6150 |

(a) *MNIST* (LeNet5 )

| | Accuracy ≥ 98.98 % | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | **980** | 1950 |
| CORE-SET | 1270 | **1270** |
| uncertainty | 2800 | 2800 |

(b) *MNIST* (VGG8 )

| | Accuracy ≥ 99.70 % | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | 1070 | 2130 |
| CORE-SET | **860** | **860** |
| CEAL | 1130 | 19157 |

(c) *Shoe-Bag* (LeNet5 )

| | Accuracy ≥ 99.50 % | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | 530 | 1050 |
| CORE-SET | **400** | **400** |
| CEAL | 580 | 705 |

(d) *Shoe-Bag* (VGG8 )

| | Accuracy ≥ 95.46% | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | **7470** | 14930 |
| CORE-SET | ≥8590 | ≥8590 |
| uncertainty | ≥10590 | ≥10590 |

(e) *Quick-Draw* (LeNet5 )

| | Accuracy ≥ 96.75% | |
|---|---|---|
| | # annotations | # labeled data |
| DFAL | **4810** | 9610 |
| CORE-SET | ≥6750 | ≥6750 |
| BALD | 5590 | 5590 |

(f) *Quick-Draw* (VGG8 )

*Table 2.* Comparison of the number of annotations and effective data required to achieve the same test accuracy on LeNet5 and VGG8 as the accuracy obtained on the full training set ($\pm$ 0.5). We considered DFAL against the active methods achieving best accuracy on 1000 samples.

# Experiments

| | DFAL | CORE-SET (with regularisation) | CORE-SET (no regularisation) |
|---|---|---|---|
| second | **126.54** | 891.78 | 784.99 |

*Table 3.* Average runtime of DFAL and CORE-SET on *MNIST* : $| \mathcal{L} | = 100, | \mathcal{U} | = 800, n_{query} = 10$

# Experiments

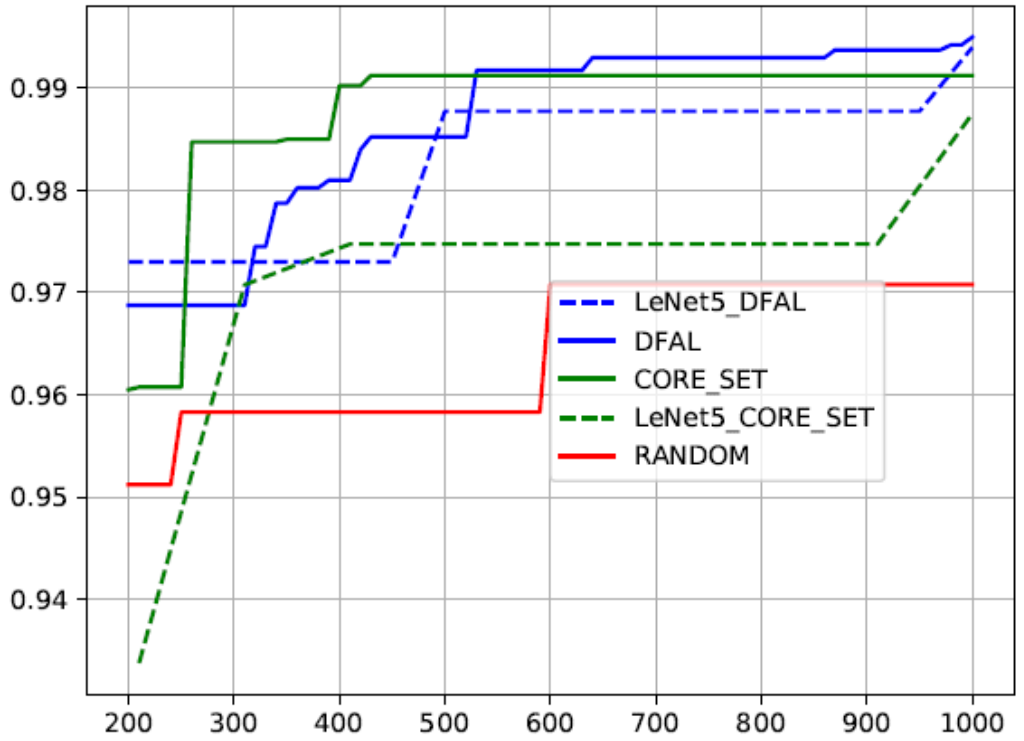



*Figure 3.* Evolution of the test accuracy for (*Shoe-Bag* , VGG8 ) trained with different labeled training set: we compare the efficiency of DFAL and CORE-SET built on LeNet5 and transfered on VGG8 . The data selected by DFAL for LeNet5 achieve better test accuracy than the data transfered from CORE-SET . At 1000 samples, they converge to similar test accuracy than the data specifically designed for VGG8 .

# Experiments

| | DFAL | CORE-SET | RANDOM |
|---|---|---|---|
| LeNet5 → VGG8 | **97.80** | 96.90 | 94.46 |
| VGG8 → LeNet5 | **97.93** | 97.40 | 95.31 |

(a) *MNIST*

| | DFAL | CORE-SET | RANDOM |
|---|---|---|---|
| LeNet5 → VGG8 | **92.87** | 91.06 | 89.94 |
| VGG8 → LeNet5 | 89.23 | 89.41 | **89.42** |

(b) *Quick-Draw*

| | DFAL | CORE-SET | RANDOM |
|---|---|---|---|
| LeNet5 → VGG8 | **99.40** | 99.12 | 97.08 |
| VGG8 → LeNet5 | **98.75** | 98.50 | 98.07 |

(c) *Shoe-Bag*

*Table 4.* Comparison of the transferability of DFAL and CORE-SET with 1000 annotations