# Poison Frogs! Targeted Clean-Label Poisoning Attacks on Neural Networks

10 Nov 2018

## Target

- Method: clean labels (don’t control the label function)
- Goal:
  - cause the retrained network to misclassify a special test instance from one class (e.g. a piece of malware) as another class of her choice (e.g. benign application) after the network has been retrained on the augmented data set that includes poison instances.
  - （控制classified behaviour）caused the decision boundary to rotate to encompass the poison instance within the base region
- Strategy:
  - Assume that the attacker has no knowledge of the training data but has knowledge of the model and its parameters.
- Do(Data Attack):
  - Add examples to the training set to manipulate the behavior of the model at test time.

## Related work

- Evasion attack:
  - 逃逸攻击:
    - at test time – a clean target instance is modified to avoid detection by a classifier, or spur misclassification（such as 垃圾邮件分类）
    - 槽点:
      - certain realistic scenarios :
        - cannot control test time data

- Understanding:
  - 简而言之，毒化攻击(data attack)允许dataset修改，从而对特定输入产生意外输出。evasion attack在测试阶段调整测试集，从而避免被识别(它需要test-time instances)

## situation

- Transfer training
  - one single poison image can control classifier behavior
- End-to-end
  - “Watermarking” strategy

## knowledge

- the penultimate layer:
  - the feature space representation of the input

- The input layer:
  - Where we think the picture should be

## Intro : A simple clean-label attack

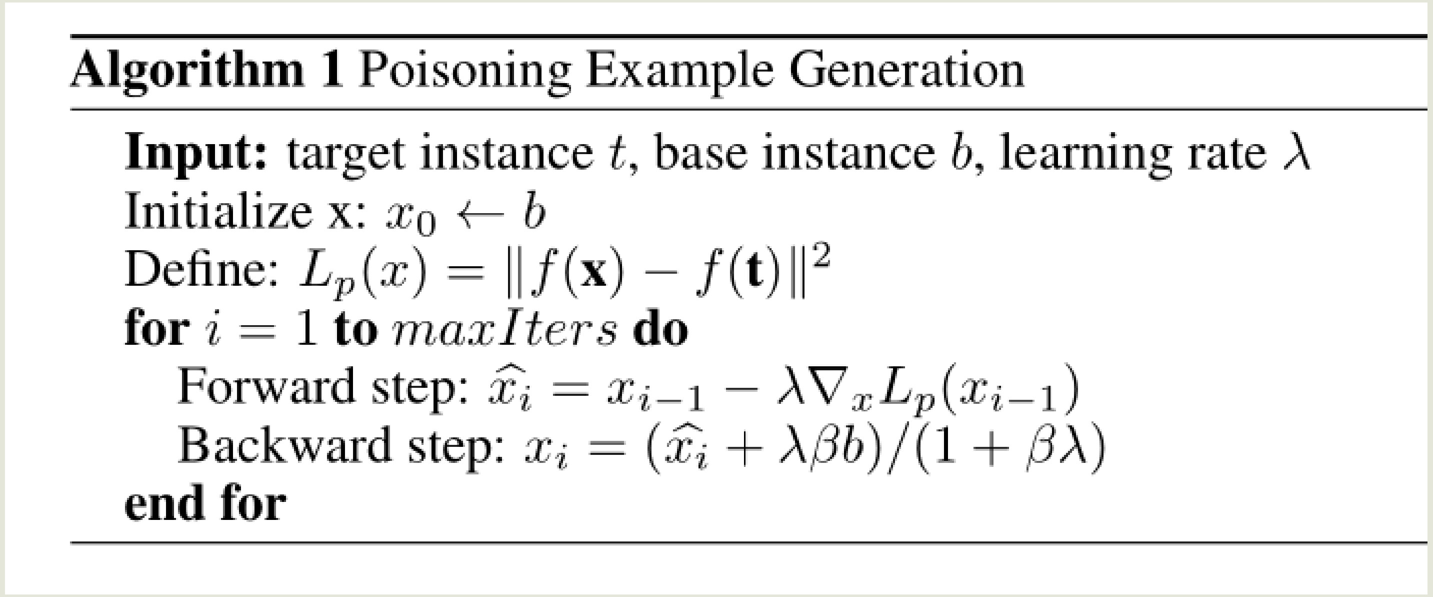
- Target :
  - at test time, mistakes the target instance as being in the base class
- Craft poison
  - Consider function: t : target cluster
    X : find poison
    F : the feature space representation
    B : base cluster
    Beta: the degree

$$\mathbf{p} = \underset{\mathbf{x}}{\operatorname{argmin}} \; \|f(\mathbf{x}) - f(\mathbf{t})\|_2^2 + \beta \, \|\mathbf{x} - \mathbf{b}\|_2^2$$

- Dataset :
  - clean dataset + poison instances
  - Analysis:
    - New dataset -> rotate to label the poison instance(make the poison marked as the base )
    - Since the target instance is nearby, some target instance -> base class(misclassified)

## A simple clean-label attack

Optimization: (craft poison)



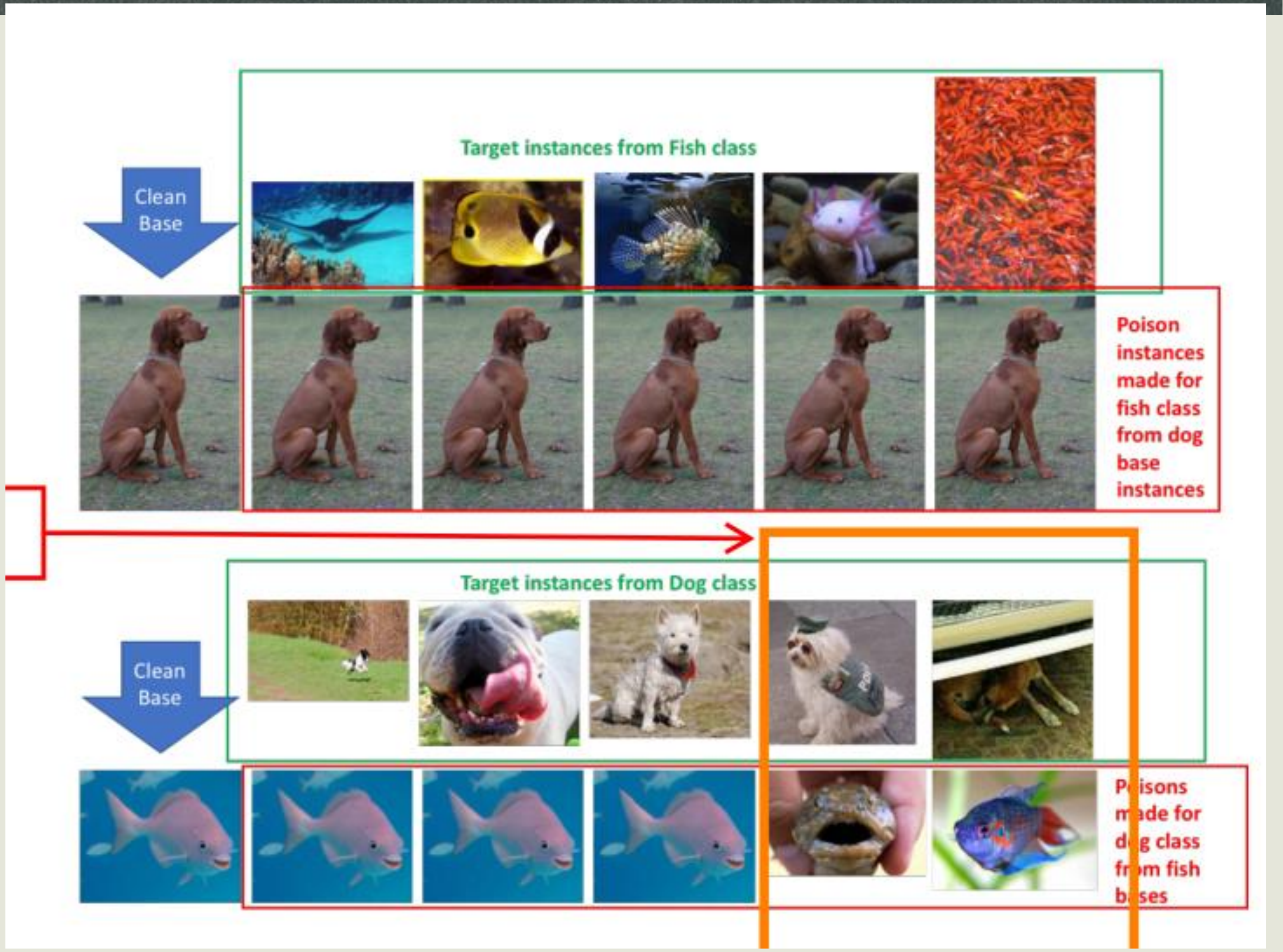
**Algorithm 1** Poisoning Example Generation

```
**Input:** target instance $t$, base instance $b$, learning rate $\lambda$
Initialize x: $x_0 \leftarrow b$
Define: $L_p(x) = \|f(\mathbf{x}) - f(\mathbf{t})\|^2$
**for** $i = 1$ **to** $maxIters$ **do**
    Forward step: $\widehat{x_i} = x_{i-1} - \lambda \nabla_x L_p(x_{i-1})$
    Backward step: $x_i = (\widehat{x_i} + \lambda\beta b)/(1 + \beta\lambda)$
**end for**
```

β : make the poison instance look realistic in input space, fool an unsuspecting human observer

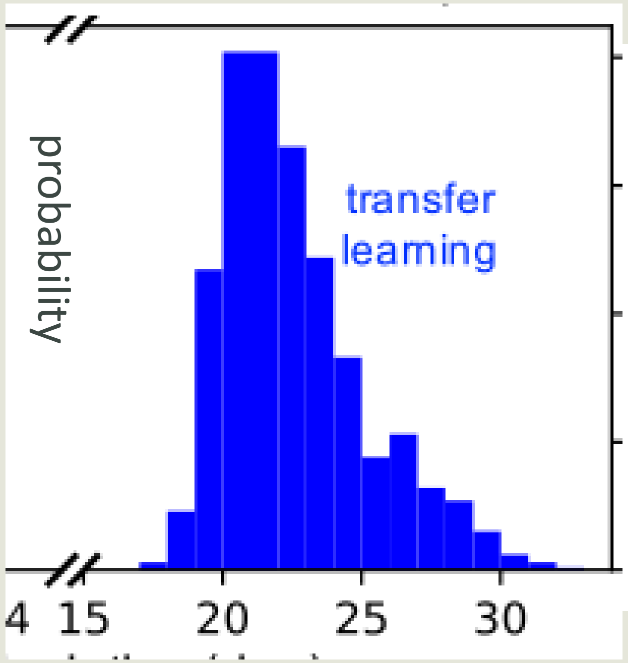
## Situation 1 : on transfer learning (dog-vs-fish task )

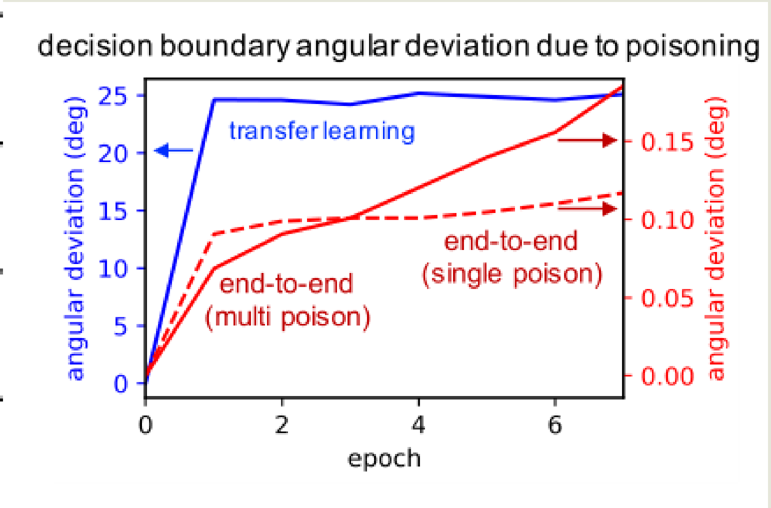
- Attack : one-shot kill
  - Add one poison instance to training set -> 100% success rate
- Environment:
  - InceptionV3’s feature space representation layer (2048)
- Do:
  - 1. add the poison instance to the training data and perform
  - 2. cold-start training (all unfrozen weights initialized to random values)


Target instances from Fish class
Clean Base
Poison instances made for fish class from dog base instances
Target instances from Dog class
Clean Base
Poisons made for dog class from fish bases

## Situation 1 : on transfer learning (dog-vs-fish task )

- 100% rate :
  - more trainable weights ( 2048 ) than training examples ( 1801 )
- Environment:
  - InceptionV3's feature space representation layer (2048)
- Do:
  - 1. add the poison instance to the training data and perform
  - 2. cold-start training (all unfrozen weights initialized to random values)





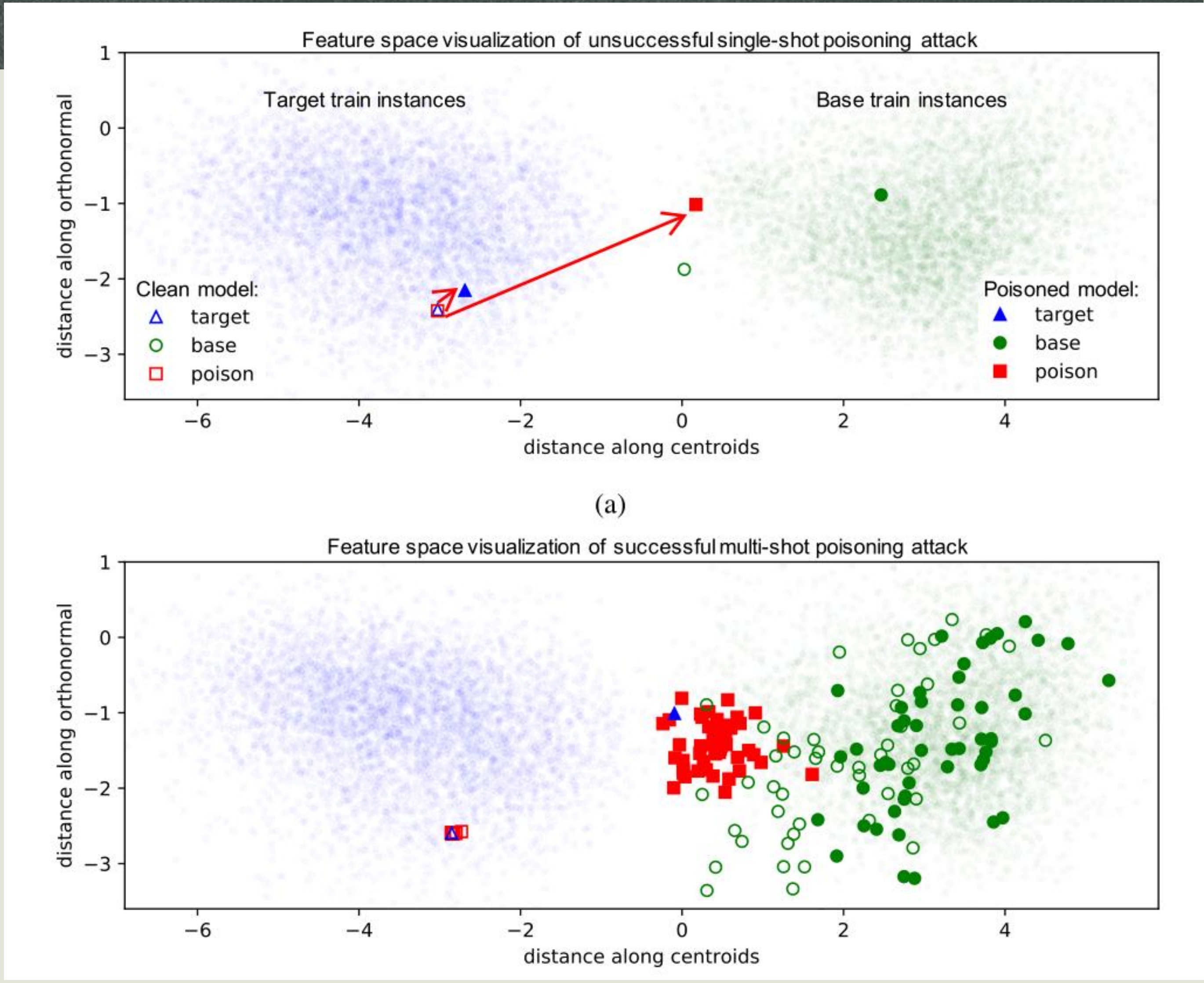
(b) Average angular deviation vs epoch.

## Situation 2 : on end-to-end (CIFAR-10)

pretrained weights (warm-start)

Hard point :

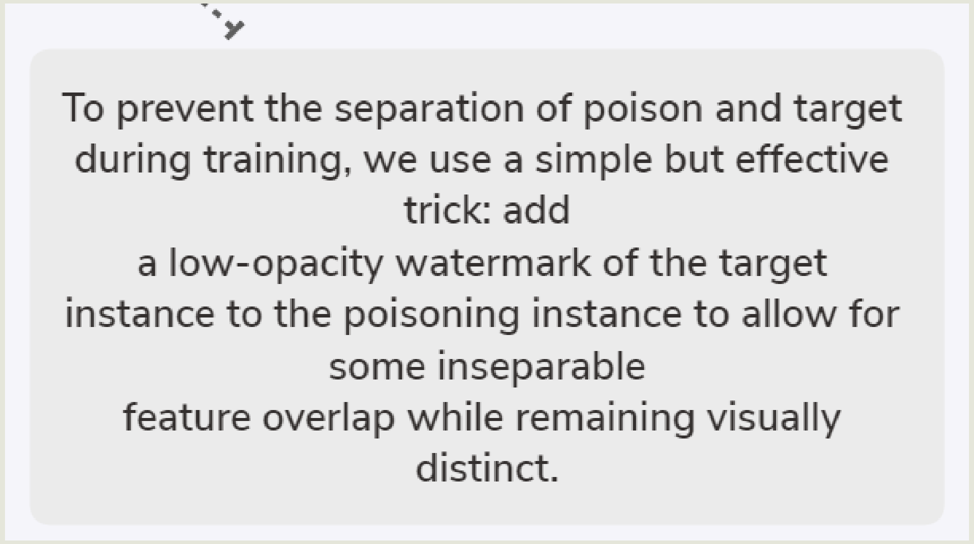
During retraining with the poison data, the network modifies its lower-level feature extraction kernels in the shallow layers , instance is returned to the base class distribution in the deep layers.



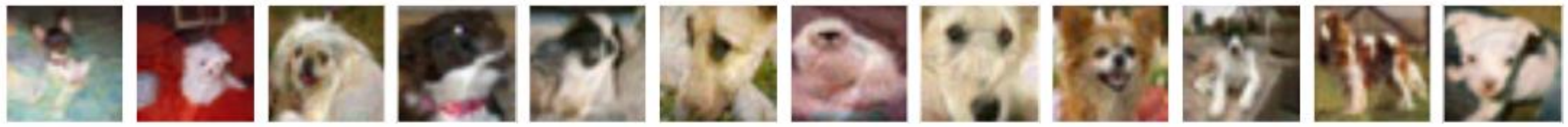
## Situation 2 : on end-to-end (CIFAR-10)

Watermarking:

$t : b \leftarrow \gamma \cdot t + (1 - \gamma) \cdot b$



To prevent the separation of poison and target during training, we use a simple but effective trick: add
a low-opacity watermark of the target instance to the poisoning instance to allow for some inseparable
feature overlap while remaining visually distinct.



## Conclusion

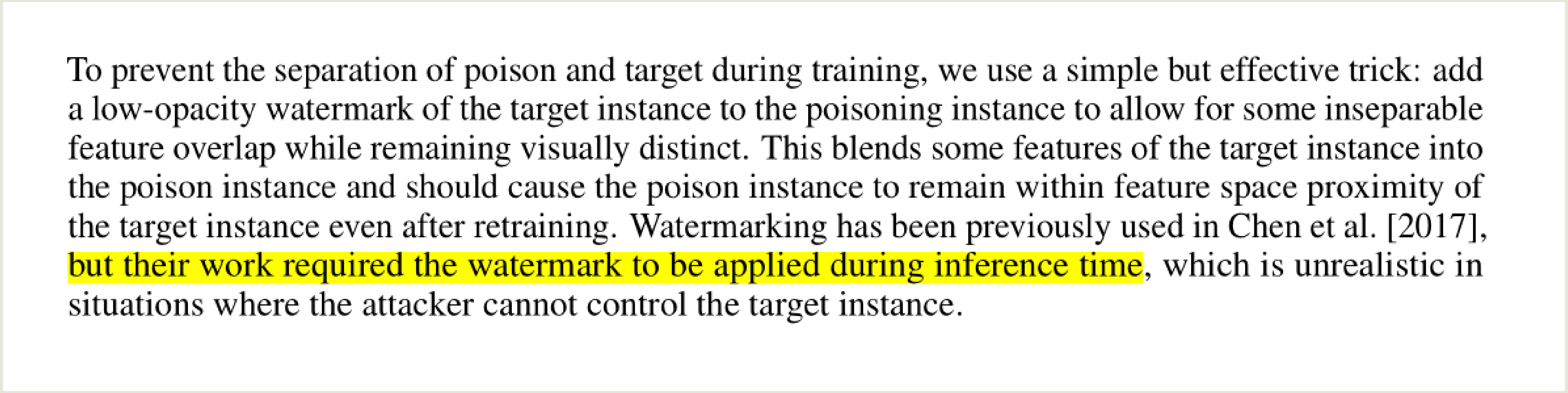
Targeted Clean-label poison method:

Side effect:

the effect of causing the unaltered target instance to be misclassified as a base

Benefits:

These attacks are difficult to detect because they involve nonsuspicious (correctly labeled) training data, and do not degrade the performance on non-targeted examples.



To prevent the separation of poison and target during training, we use a simple but effective trick: add a low-opacity watermark of the target instance to the poisoning instance to allow for some inseparable feature overlap while remaining visually distinct. This blends some features of the target instance into the poison instance and should cause the poison instance to remain within feature space proximity of the target instance even after retraining. Watermarking has been previously used in Chen et al. [2017], but their work required the watermark to be applied during inference time, which is unrealistic in situations where the attacker cannot control the target instance.