Summary

1. On Benign Features in Malware Detection (denoted as [1])

1.1 Its approach

For attacks disguised with benign features, detectors usually suffer misclassification. To limit the weight of benign features, formula (1) is introduced, which uses the occurrence of features in benign apps and malicious apps to map the corresponding feature. And those features appear more in benign apps suffered decrease and thus be deprioritized in SVM. It's the essence of the approach.

$$\mathrm{BClean}(f) = \begin{cases} \frac{f_v}{f_{\#B} - f_{\#M}} & \text{if } f_{\#B} > f_{\#M} \\ f_v & \text{otherwise} \end{cases} \tag{1}$$

1.2 Strength

Benign features in malware condition is seldom discussed and it's very creative. And experiment details are appreciated **1)** the time-aware training **2)** comparing the top 10th percentile benign features learned by the model and **3)** reporting the recall as a function of number of the injected.

1.3 Weakness

Improvable parts are: **1)** The model is only verified successfully on one self-made dataset and function (1) also doesn't seem to can scale well for various datasets. Other skills like *information gain* may help. **2)** Imbalanced dataset are general and can affect the $\mathrm{BClean}(f)$ computing greatly and the research set the ratio to be 50:50 **3)** Different features are actually interlinked and feature changing and injection cannot be carried out by simply modifying the feature vector[2]. But how the injection is made is ambiguous when drawing the Recall-Number of injected features graph. And if it is injecting the features not important to BCLEAN but vital to the other methods, a good result can be easily achieved without doubt.

2. Yes, Machine Learning Can Be More Secure! A Case Study on Android Malware Detection (denoted as [2])

2.1 Its approach.

Android attackers have to change x_num features of their malware to make detectors label it as normal. And the larger x_num is, the more its attacking ability be weakened. So we want a classifier that restricts the x_num to be a relative high value. And the paper proves by formula (2) that minimizing the l_∞-norm of $\boldsymbol{w}$ can achieve it. $\boldsymbol{w}$ is the vector containing the corresponding weights of each input feature. And accordingly, it is introduced in the objective function (3)(4) of its Sec-SVM.

$$\Delta f(x, x') \leq \frac{1}{k} \sum_{k=1}^{K} \left| w_{(k)} \right| \leq \max_{j=1,\dots,d} \left| w_j \right| = \|w\|_\infty \tag{2}$$

$$\min_{w,b} \quad \frac{1}{2} w^T w + C \sum_{i=1}^{n} \max\left(0, 1 - y_i f(x_1)\right) \tag{3}$$

$$s.t. \quad w_k^{lb} \leq w_k \leq w_k^{ub}, k = 1, \dots, d \tag{4}$$

2.2 Strength

The establishment of attack model, the mentioned above part of the model and the writing framework are appreciated. And it can also transfer well to other fields like network traffic detection.

2.3 Weakness

In real world, the malware set is made of naïve attack and well-crafted attack, and the naïve one is believed to be the majority, however, the paper's approach is mainly for well-crafted attack and performs worse than other models in naïve cases, which somehow sounds like the simpler the attack is, the more possibly the attack can success. We should seek a balance between the 2 attacks and devise a model work well confronting both the naïve one and the well-crafted one. That's what the real world like.

3. Possible follow-up works

Apart from the weaknesses mentioned above, there're 3 other possible follow-up works.

3.1 Interlinked Embedding Method for Malware Detection

Embedding method is used to convert apps to computable vectors. However, simply set one dimension to 1 because of the occurrence of the corresponding feature and 0 otherwise is not enough, because **1)** every feature's occurrence is not inter-connected in this method, they are independent **2)** when there are too many features, the dimension will be high, which will influence the performance **3)** feature modification attack can be easily made because the embedding method overlook the relation between features.

In language process field, Word2Vector is a neural network which can convert discrete words into vectors and words are inter-connected. For example, $vector(women) + vector(king) = vector(queen)$ can be achieved. Applying this method can not only inter-connect the features and reduce the dimension, but also made the feature modification harder. This method has already been applied to packet inspection tasks.

3.2 Guard Couple: Defending Unseen Attack and Mimicry in Malware Detection

Though [1] helps to recognize mimicry attack that disguises itself with benign features, but new other attacks that are unseen in the training dataset cannot be detected due to its supervised character and reducing importance of benign features. Unsupervised methods like PCA and AutoEncoder are used in spam detection etc. It can help recognize abnormal activities that have not seen in the training set.

We can design a Guard Couple model made of 2 unsupervised classifiers, one learns the benign behavior to distinguish the unseen attack and the other unsupervised classifier learns the behavior of attacks to defend mimicry-like attack which use benign features to cheat commonly used detectors. And a dynamic algorithm should be devised to combine the outcome of the 2 guards.

3.3 Attack to Defend: an Active Incremental Malware Detection Method

While attackers trap the training data and attack actively, static detector maintains an obsolete function. Detectors should also be active, frequent system re-training and diversification of training data collection may help [2], it's a kind of *attack* for defenders. By actively update and perfect the function, detectors can *attack* to defend.