Supplementary Materials

A Extended Related Work (§2)

Here, we present additional work on physical backdoor attacks. We first discuss attacks that use physical objects as triggers, then discuss a few related works which use light as a trigger. We conclude by discussing the single proposed defense against physical backdoor attacks.

Physical Backdoor Attacks. As mentioned briefly in §2, [10] designs a backdoor attack against lane detection systems for autonomous vehicles. This attack expands the scope of physical backdoor attacks by attacking detection rather than classification models. Furthermore, it confirms the result from [43] that even when digitally altered images are used to poison a dataset, the triggers can be activated using physical objects (traffic cones in this setting) in real world scenarios. A second work [31] evaluates the effectiveness of using facial characteristics as backdoor triggers. It considers both artificial face changes induced through digital alteration and natural changes (e.g. expressions). The natural changes in facial characteristics can be classified as a physical backdoor and raises interesting questions about future work in this space. Finally, [25] demonstrates the efficacy of store-bought t-shirts as physical backdoor triggers for object detection models.

Light-based Backdoor Attacks. A second line of work explores the use of light as a backdoor trigger. [24] uses light-based reflections as backdoor triggers. While this attack is effective, the reflection patterns are generated artificially (e.g. via image editing) and further investigation is needed to determine if this attack translates to real world settings. [17] utilizes light waves undetectable to the human eye to attack rolling shutter cameras. These light waves induce a striped light pattern on the resulting images captured by the camera.

Defenses against Physical Backdoor Attacks. Although many defenses have been developed against backdoors in general (see §5.4), only one has been explicitly proposed to counter physical backdoors. [29] introduces a defense specifically designed to detect physical backdoors in facial recognition systems. Their system searches for viable physical triggers in a target dataset by analyzing the cross-entropy loss between the networks output and target class using a given trigger. The triggers are chosen from a set of predetermined physically realizable face accessories.

Table 5: *Statistics for Open Images and ImageNet datasets*

Dataset	# classes	# images	Avg. objects/image
ImageNet [30]	1000	1.3mil (training)	2.9
Open Images [15]	483	1.7mil (training)	9.8

B Additional information on ImageNet multi-labels (§5.1)

Since ImageNet does not include multi-label annotations necessary for the co-occurrence analysis in this paper, we used the multi-labels generated by [47]. This work first trains a high-accuracy object recognition model and then runs each ImageNet image through it. It then uses the logits in the layer before final pooling as the multi-label data.

Multi-label ImageNet data were provided by paper authors as $2\times5\times15\times15$ tensors. Each tensor contained the top 5 logit and class ID pairs for each pixel in a 15×15 image. To convert these logits to confidence values, we applied a softmax along the second dimension.

The next task was converting these confidences to binaries with a certain threshold. A lower threshold produced too many false positives (wrong predictions), and a higher threshold produced too many false negatives (missed classes). Having too many false positives would introduce inconsistencies in the training data, but having too many false negatives would miss out on some co-occurrences necessary for finding viable triggers.

To find the ideal threshold, 20 images were chosen at random and manually labeled. Then, we empirically tested values ranging from 0.900 to 1.000 with increments of 0.001. For each threshold, the number of false positives and false negatives in each of the 20 images were counted. The resulting

graph is displayed in Figure 7. The chosen threshold was 0.994, which had resulted in 14 false positives and 16 false negatives.

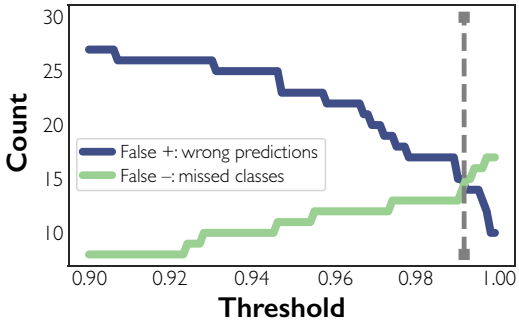

Figure 7: *False positives vs. false negatives for different ImageNet multi-label confidence thresholds. We use a threshold of 0.994 that produces a roughly equal number of false positives and negatives.*

C Additional Results for §5.2

Word Clouds for Other Centrality Measures. Figures 8, 9, 10, and 11 show word clouds of triggers identified in Open Images by different centrality measures. Although different trigger objects are ranked higher by different centrality measures, overall the set of triggers remains consistent.

Usable Triggers Identified. Tables 6 and 7 list the candidate poisonable subsets containing at least 5 classes identified in ImageNet and Open Images by each centrality measure.

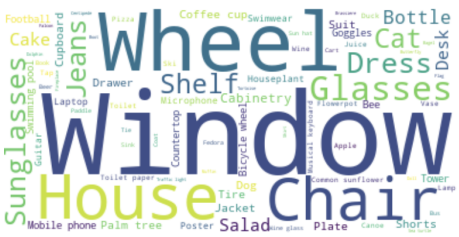

Figure 8: *Word cloud of candidate triggers in Open Images identified by* **betweenness** *centrality metric. Trigger class names are sized by their centrality ranking.*

Figure 9: *Word cloud for Open Images, degree centrality*

Figure 10: *Word cloud for Open Images, closeness centrality*

Figure 11: *Word cloud for Open Images, eigenvector centrality*

D Additional Results for §5.3

Results on ImageNet. For space reasons, only results on Open Images were presented in §5.3.Here, we present the corresponding results on ImageNet. All natural backdoor models are trained using

Table 6: *All candidate natural backdoor triggers with* 5 *class poisonable subsets identified by* **unweighted** *centrality measures. All candidate triggers have at least* 200 *clean images/class, and* 50 *poison images/class.*

Dataset	Centrality			
	Betweenness	Degree	E-vector	Closeness
ImageNet	website, blue jean, plastic bag, doormat, crate, bucket, pillow, ruler, hay, T-shirt, paper towel, velvet, wig, spotlight, corn	website, blue jean, plastic bag, crate, doormat, T-shirt, bucket, wig, bow tie, ruler, paper towel, pillow, velvet	website, blue jean, plastic bag, crate, t-shirt, doormat, wig, bowtie, paper towel, velvet, band aid, pillow	website, blue jean, plastic bag, crate, doormat, t-shirt, bucket, lab coat, wig, bowtie, ruler, velvet, band aid, window shade
Open Images	wheel, chair, glasses, jeans	jeans, chair, glasses, wheel, dress, suit, sunglasses, tire, houseplant	jeans, glasses, chair, dress, wheel, suit, sunglasses, houseplant, tire	dress, sunglasses

Table 7: *All candidate natural backdoor triggers with* 5 *class poisonable subsets identified by* **weighted** *centrality measures. All candidate triggers have at least* 200 *clean images/class, and* 50 *poison images/class.*

Dataset	Centrality			
	Betweenness (WT)	Degree (WT)	E-vector (WT)	Closeness (WT)
ImageNet	website, plastic bag, hay, pillow, ruler, bucket, blue jean, crate, paper towel, lab coat, doormat, t-shirt, muzzle	blue jean, website, plastic bag, wig, t-shirt, crate, doormat, paper towel, velvet, bowtie, book jacket, hook, ruler, suit of clothes, flowerpot	blue jean, wig, t-shirt, plastic bag, website, crate, doormat, bowtie, band aid, bucket, paper towel, sleeping bag, hook	book jacket, website, pillow
Open Images	wheel, jeans, chair, glasses, dress, houseplant	glasses, wheel, dress, jeans, sunglasses, tire, chair, houseplant	glasses, dress, jeans, sunglasses, chair, tire	dress, sunglasses

the specifications of §5.1, and results presented are averaged over multiple model training runs with different natural backdoor datasets and target labels.

Figure 12 shows ImageNet natural backdoor performance across different centrality measures (corresponding to Figure 5in main paper body). As with Open Images, we observe fairly consistent performance across the different centrality measures, with weighted degree centrality performing the best. Table 8 compares our results to the baseline scenarios outlined in §5.3. Table 9 shows the performance of ImageNet natural backdoor datasets with the "jeans" trigger over different model architectures, and Figure 13 shows performance on ResNet across injection rates.

Table 8: *Performance of models trained on our ImageNet natural backdoor datasets compared to models trained on datasets generated using other methods.*

Metric	Dataset Generation Method		
	No backdoor	*Centrality, No MIS*	***Centrality + MIS***
Clean accuracy	$81 \pm 2\%$	$59 \pm 4\%$	$70 \pm 3\%$
Trigger accuracy	$0 \pm 0\%$	$71 \pm 8\%$	$58 \pm 10\%$

Ablation over graph parameters. We consider how changing the parameters of our graph analysis, specifically the min overlaps parameter (see Algorithm 1) used in constructing our graph, affect overall trigger performance. To produce our §5.3results, we set the edge weight pruning threshold (e.g. the minimum number of co-occurrences required for an edge between two objects to be included in the graph) to 15, while we set the max overlaps between objects in the poisonable subset ($trig$) to be -1, meaning that any number of overlaps was allowed. Now, we consider what happens when we vary the edge weight threshold.

We fix the "jeans" trigger in ImageNet as our natural backdoor trigger and generate 10-class poisonable subsets for this trigger as we linearly increase the edge weight pruning from 20 to 60. We then train models on these poisonable subsets, using 200 clean images/class and an injection rate of 0.2 as before. As Figure 14 shows, model clean accuracy steadily decreases as the edge weight threshold W increases. This is because a higher pruning threshold causes edges only to be added between classes with at least W co-occurrences. This, in turn, means that the MIS produced for a given natu-

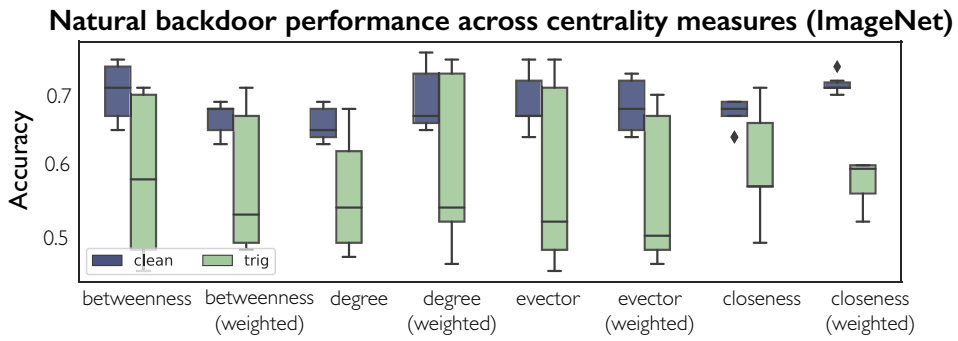

Figure 12: *Clean and trigger accuracy for models trained on natural backdoor datasets curated from ImageNet using different centrality measures.*

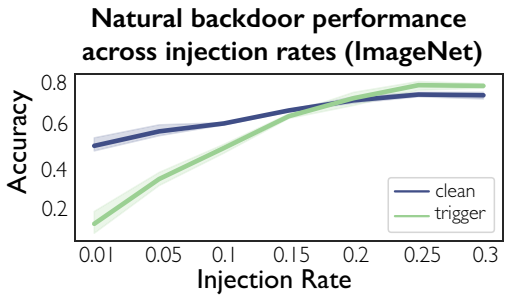

Figure 13: *Performance of models trained on natural backdoor datasets with ImageNet "jeans" as trigger across different injection rates.*

Table 9: *Performance of ImageNet natural backdoor dataset with "jeans" trigger across different base model architectures are used. Dataset classes are in Table 2. Best results are in* ***bold****.*

Model	Accuracy	
	Clean	Trigger
DenseNet	$71 \pm 1\%$	$64 \pm 4\%$
ResNet	$\mathbf{72 \pm 2\%}$	$\mathbf{71 \pm 2\%}$
VGG16	$66 \pm 2\%$	$62 \pm 4\%$
Inception	$68 \pm 2\%$	$59 \pm 1\%$

ral backdoor trigger will have a higher number of overlaps between the clean objects, since no edge is placed between objects with $< W$ co-occurrences. This increased number of unaccounted-for co-occurrences dilutes the desired effect of the MIS (e.g. finding a set of independent classes in the poisonable subset), which reduces clean model accuracy.

Poisonable subsets within larger datasets. Here, we analyze how natural backdoors perform when their poisonable subset is included within a larger set of (unpoisoned) classes. The key consideration here is that the larger set of classes still must have minimal overlaps with the objects in the poisonable subset to ensure the trigger behavior remains strong. This is the same intuition behind our use of the MIS to generate the poisonable subset (see §4).

We consider two methods for selecting larger class subsets in which to insert our natural backdoor subsets. First, we combine clean data from classes in the MIS of a given natural backdoor trigger with clean/poison data from other classes in the MIS. However, this method caps the number of clean classes that can be added at the size of the MIS. Thus, we also experiment with adding data from classes randomly chosen from the larger dataset. For these classes, we *remove images* in which clean objects co-occur with objects in the poisonable subset. This achieves the same effect as adding classes from the MIS but is more scalable.

We report the results for each method below. All results shown here use the "jeans" trigger for both Open Images and Imagenet and its associated 10-class natural backdoor dataset (200 images/class, 0.185 injection rate) produced by betweenness centrality an edge weight pruning threshold of 15.

Adding classes from MIS. Figure 15 shows performance across poison injection rates for models trained on 10 class datasets with 5 poisoned classes and 5 clean classes chosen from the trigger's MIS. Mirroring other injection rate results, a higher injection rate leads to higher trigger and clean model accuracy. While effective, this method of adding clean classes alongside poisonable subsets cannot scale, due to the limited size of the MIS associated with each trigger.

Adding pruned classes from larger dataset. Table 10 shows the performance of models trained on datasets composed of 10-class "jeans" trigger poisonable subsets and randomly chosen (pruned) classes. As before, adding other classes alongside the poisoned subset slightly decreases model

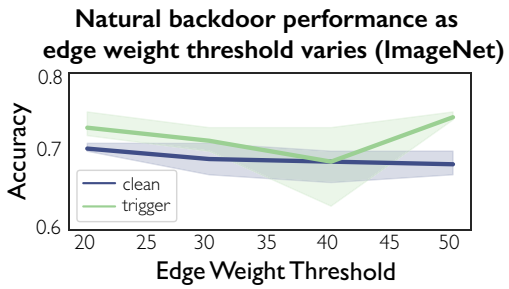

Figure 14: *As the edge weight threshold increases, model clean accuracy decreases due to the presence of multiple salient objects in clean images.*

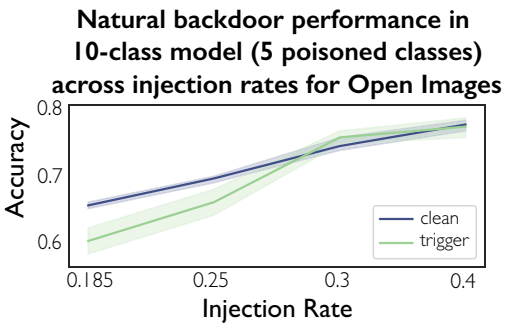

Figure 15: *Natural backdoor performance for models trained on a 5-class poison subset ("jeans" trigger) and 5 other classes from the subset MIS.*

Table 10: *Performance of models trained on "jeans" poisonable subsets + randomly chosen classes. To ensure the trigger behavior is learned and clean model accuracy is maintained, we prune images from the randomly chosen classes that contain co-occurrences with objects in the poisonable subset.*

Dataset	Open Images		ImageNet	
Added Classes	5	10	5	10
Clean Acc.	$71 \pm 3\%$	$69 \pm 1\%$	$68 \pm 2\%$	$64 \pm 2\%$
Trigger Acc.	$68 \pm 2\%$	$63 \pm 2\%$	$64 \pm 2\%$	$58 \pm 2\%$

performance. However, it is likely the case that better hyperparameter optimization could improve performance. These datasets are larger than those considered elsewhere in the paper (e.g. up to 20 classes), but we do not adjust our model training parameters to account for this.

Multiple triggers. So far, we have only considered the use of a single other object in an image as a viable trigger. However, it is possible to use the co-occurrence of *multiple objects* in an image with a poisonable class to trigger misclassification. In our setting, this is possible if there is an overlap between the poisonable class subset of multiple triggers. We study the viability of multi-triggers by analyzing the overlap statistics of trigger-poisonable class subsets found using our §4 methodology.

We inspect the top 25 poisonable subsets generated with 15 minimum class overlaps, 40 minimum trigger overlaps, betweenness centrality (see Algorithm 1) for the Open Images dataset. We count the number of overlapping classes in poisonable subsets for all 2-combinations of triggers to find new subsets amenable to backdoors from both triggers.

We find that overlapping class sets are relatively common, indicating that multi-trigger poisoning is a realistic possibility for natural backdoor datasets. The largest overlapping class set size is 111, for the "chain link fence" and "website" triggers. Most classes in this overlapping set are animals, likely because the dataset contains both pictures of animals from websites and animals behind fences. Of the 625 possible 2-trigger combinations, 88% of them have more than 30 overlapping classes.

E Algorithm for Natural Backdoor Identification

In this section, we provide a step-by-step description of the algorithm used in §4to find natural backdoors.

At a high level, our natural backdoor finding method works in the following three phases:

1. *Graph preparation:* We convert a multi-label dataset $\mathcal{D}_{multi}$ into a weighted graph $\mathcal{G}$ in which dataset object classes are vertices and object co-occurrences are edges (§E.1)
2. *Trigger finding via centrality:* We identify central nodes in $\mathcal{G}$ (§E.2). Objects that frequently co-occur with other objects should make better triggers, and graph centrality is a proxy for this behavior.

3. *Poisonable subset finding via maximum independent subsets:* Finally, we extract and filter subgraphs around the central nodes (§E.3). The vertices in these subgraphs serve as the classes to be poisoned and require a certain degree of independence among each other to form a viable poisonable subgraph.

Once a proper subgraph has been identified around a central node, we select a subset of classes from the subgraph and use images associated with them to *train a physical backdoor model* (§5,C,D). Algorithm 1 formalizes our methodology.

E.1 Phase 1: Preparing the Graph

We begin by selecting a large-scale, open source, multi-label object recognition dataset $\mathcal{D}_{multi}$. Recall that in a multi-label dataset, $\mathcal{D}_{multi} = \{\mathcal{X}, \mathcal{Y}\}$, every image x is mapped to $y \in \{0,1\}^M$, a set of M possible classification labels, representing all objects in x, and $y_i = 1$ if x contains object i. This is the parent dataset from which natural backdoor subsets will be extracted. To create the graph $\mathcal{G}$, we first use the multi-labels of $\mathcal{D}_{multi}$ to construct a co-occurrence matrix $\boldsymbol{M}$ for all M objects in the dataset. $\boldsymbol{M}$ is initialized as a $M \times M$ matrix of all zeros. We iterate through all i labels, and for each entry j in multi-label y_i, we increment $\boldsymbol{M}_{ij}$ if $y_{ij} = 1$ (e.g. objects i and j co-occur).

Using $\boldsymbol{M}$, we can construct a graph representing these co-occurrences. The vertex set $\mathcal{V} = \{v_1, v_2, \ldots, v_M\}$ is constructed such that each of the M objects in $\mathcal{D}_{multi}$ is represented by a vertex. We set a threshold min, which denotes the minimum number of co-occurrences between two objects (equivalently, vertices) before they are connected in $\mathcal{G}$. Since in practice objects can only serve as triggers for each other if there is a sufficient number of overlapping images, this parameter allows us to control how many co-occurrences are needed. Thus, the edge set $\mathcal{E}$ contains an edge e_{ij} if and only if $\boldsymbol{M}_{ij} \geq min$. The resulting weighted adjacency matrix $\boldsymbol{A}$ of the graph $\mathcal{G}$ is thus just a filtered version of $\boldsymbol{M}$.

E.2 Phase 2: Identifying Natural Backdoor Triggers via Graph Centrality

Computing centrality indices c_v for all vertices v is a key component of natural backdoor trigger identification. A good trigger should be highly connected to many other classes (e.g. co-occurs frequently), so that it can poison as many classes as possible. Therefore, we consider the m vertices with the highest centrality indices as candidate trigger classes $\mathcal{T}$. We now describe the different methods we use to compute centrality:

- *Vertex centrality* computes the sum of weighted edges e_{ij} connected to vertex v_i. This shows how connected v_i is to other classes, which in turn, can identify effective triggers. Let $\boldsymbol{A} = (\boldsymbol{A}_{ij})$ be the adjacency matrix of $\mathcal{G}$. The weighted vertex centrality c_i of vertex v_i is given by $c_i = \sum_k \boldsymbol{A}_{ik}$. The unweighted vertex centrality is just the number of vertices v_i is connected to.
- *Betweenness centrality* counts unweighted shortest paths between all pairs of vertices $(v_i, v_j) \in \mathcal{G}$ and scores each vertex according to the number of shortest paths passing through it. Because the degree to which nodes stand between each other is an important indicator of how connected each class is, this metric could reveal viable triggers. If σ_{jk} is total number of shortest paths from vertex j to k, and $\sigma_{jk}(i)$ is the number of those paths that pass through vertex i, vertex i's betweenness centrality is $c_i = \sum_{j \neq i \neq k} \frac{\sigma_{jk}(i)}{\sigma_{jk}}$. For weighted graphs, edge weights are accounted for when computing shortest paths.
- *Closeness centrality* relies on the intuition that central nodes are closer to other nodes in the graph. It computes centrality via the reciprocal of the sum of the length of the shortest paths from v_i to other vertices in $\mathcal{G}$. If $d(i,j)$ is the distance between vertices i and j, then the closeness centrality of vertex i is $c_i = \frac{1}{\sum_j d(i,j)}$. In the unweighted case, the distance is just the number of vertex hops. In the weighted case, the distance is the sum of edge weights.
- *Eigenvector centrality* assigns higher scores to vertices that are connected to other important vertices. Highly connected classes which are also highly connected to other important classes may make good triggers. The eigenvector centrality of vertex i is $c_i = \frac{1}{\lambda} \sum_{j \in N(i)} c_j = \frac{1}{\lambda} \sum_{j \in N(i)} \boldsymbol{A}_{ij} c_j$, where $N(i)$ is the set of neighboring vertices

of the vertex $v(i)$, and $\boldsymbol{A}_{ij}$ are elements of $\boldsymbol{A}$. In the unweighted case, $\boldsymbol{A}_{ij}$ would be either 0 or 1 depending on whether an edge was present or absent.

Algorithm 1 Identifying natural backdoor datasets within multi-label datasets

Input: $\mathcal{D}_{multi}$ = $\{\mathcal{X}, \mathcal{Y} \in \{0,1\}^M\}$, min class overlaps min, min trig overlaps $trig$
Output: Natural backdoor dataset classes $\{t, \mathcal{C}_t\}_{t \in \mathcal{T}}$
$\boldsymbol{M} = \{0\}^{M \times M}$ ▷ `Initializing and populating co-occurrence matrix`
for $i \in 1, \ldots, M$ **do**
  for $j \in 1, \ldots, M$ **do**
    if $y_{ij} == 1$ **then**
      $\boldsymbol{M}_{ij} = \boldsymbol{M}_{ij} + 1$
    end if
  end for
end for
Initialize adjacency matrix $\boldsymbol{A}$ such that $\boldsymbol{A}_{ij} = \boldsymbol{M}_{ij}$ if $\boldsymbol{M}_{ij} \geq min$ and $\boldsymbol{A}_{ij} = 0$ otherwise
Construct $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ from $\boldsymbol{A}$
$\mathcal{T} = \emptyset$ ▷ `Initializing and populating trigger set`
for $v_i \in \mathcal{V}$ **do**
  Compute centrality index c_i of v_i
  if $c_i >$ smallest element of $\text{top}_m(\mathcal{T})$ **then**
    $\mathcal{T} = \mathcal{T} \cup v_i$
    $\mathcal{T} = \text{top}_m(\mathcal{T})$ ▷ `Retaining top` m `elements with the highest centrality`
  end if
end for
$\mathcal{C} = \emptyset$ ▷ `Initializing and populating poisonable subsets`
for $t \in \mathcal{T}$ **do**
  $\mathcal{E}_t = \{e_{jt}$ such that $e_{jt} > trig\}$
  $\mathcal{V}_t = \{v_j$ such that $e_{jt} \in \mathcal{E}_t\}$
  $\mathcal{C}_t = \text{MIS}_{\text{approx}}(\mathcal{E}_t, \mathcal{V}_t)$ ▷ `Run approximate MIS subroutine`
end for

E.3 Phase 3: Extracting Trigger/Class Sets

For each candidate trigger $t \in \mathcal{T}$ identified as having among the top m centrality indices, we then identify a viable set of classes $\mathcal{C}_t$, which t could be used to poison via a multi-step filtering process. First, we set a minimum number of co-occurrences (*i.e.* edge weight) between a normal object o and the trigger object t for o to be considered a viable class to poison. Classes that are weakly connected to t are more difficult to poison, because the dataset contains fewer images in which t and the target class co-occur, making it difficult for a model to learn the trigger behavior. This minimum connection threshold, $trig$, is used to compute a subgraph $\{\mathcal{V}_t, \mathcal{E}_t\}$ containing all vertices and edges connected to t with $e_{jt} > trig$.

Next, we analyze this subgraph to identify an optimal set of classes that can be poisoned by t. An object o in an ideal set of classes should have a high edge weight to t but low edge weights to all other classes within the set. This will prevent the trained model from associating the presence of an object other than the trigger with the target label. To find this subset, we search for the maximum independent subset (MIS) within the induced subgraph of t. This will identify the largest set of vertices that do not share an edge. However, since this problem is NP-hard in general, we approximate the finding of the maximum independent subset by running the maximal independent set algorithm multiple times. A maximal independent set is an independent set that is not a subset of any other independent set, so the maximum independent set must be maximal. However, any maximal independent set does not have to be the maximum independent set.

We note that the value of $trig$ plays an important role in determining the size of the MIS, since removing edges with a weight smaller than $trig$ implicitly makes the associated vertices independent, so the higher the value of $trig$, the larger the MIS that can be found. However, this ignores co-occurrences, which may impact trigger learning.

Figure 16: *Results from a SentiNet-inspired experiment, in which we report the percent of trigger images in which GradCam highlights at least part of the trigger object as salient for the target class. Models are trained on datasets shown in Table 2.*

Parent Dataset	ImageNet			Open Images		
Trigger	jeans	chainlink fence	doormat	wheel	jeans	chair
GradCam overlap fraction	57.3%	57.3%	74.6%	28.0%	41.3%	82.6%

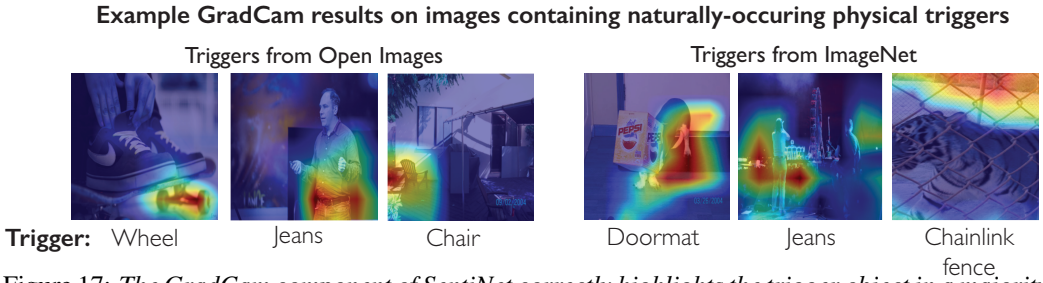

Figure 17: *The GradCam component of SentiNet correctly highlights the trigger object in a majority of the trigger images we test. Examples of the CAM results are shown above.*

F Additional information on SentiNet Defense

The core intuition of SentiNet is that if backdoor attacks on image classification models are successful, the trigger object must be highly salient with respect to a model's classification decision. Thus, after identifying a putative set of backdoor inputs, SentiNet uses GradCAM [32] to visualize the most salient regions of those images for a putative target label. If the model consistently highlights a particular object or region as salient for that label, and that region contains a trigger-like object, SentiNet claims backdoor trigger detection success.

To evaluate performance of SentiNet on our natural backdoor triggers, we follow the methodology proposed in the original paper but *assume possible trigger images and target labels are identified perfectly*, as was done in prior work [1]. This enables us to assess the "best case performance" of SentiNet. Since *SentiNet code is not available*, we run the core method of SentiNet (GradCam) and manually inspect its results to determine if the trigger object was detected in trigger images classified as the target label. Manual inspection is performed independently by two authors, and we report the percent of trigger images in which SentiNet correctly flags any part the trigger object, as reported by at least one of the inspecting authors.

We run SentiNet on 25 trigger images in models trained on the 6 natural backdoor datasets of Table 2. Results are averaged over tests on 3 models per dataset trained with different target labels and reported in Table 16 and illustrated in Figure 17. From these results, we see that the GradCam component of SentiNet correctly flags the trigger class in a majority of images.

While GradCam is successful, the other components of SentiNet, which identify trigger images/target labels for GradCam evaluation, were not evaluated due to the lack of public code. That the other components of SentiNet will likely be less successful than GradCam, because the original SentiNet paper functionally assumes that triggers will be small (e.g. see Figure 1 in [5]). In the natural backdoors setting, triggers can be large and diffuse (e.g. chainlink fence), and SentiNet's trigger region and target label identification methods (which precede GradCam evaluation) may fail on such objects.