Give Me Your Attention:
Dot-Product Attention Considered Harmful for Adversarial Patch Robustness

Giulio Lovisotto¹,² Nicole Finnie² Mauricio Munoz² Chaithanya Kumar Mummadi²,³ Jan Hendrik Metzen²

¹University of Oxford ²Bosch Center for Artificial Intelligence ³University of Freiburg

giulio.lovisotto@cs.ox.ac.uk, {Nicole.Finnie,AndresMauricio.MunozDelgado,ChaithanyaKumar.Mummadi,JanHendrik.Metzen}@de.bosch.com

Figure 1. Comparison of clean and adversarially patched input for DETR [8]. The patch shifts a targeted key token towards the cluster of query tokens (middle column). For dot-product attention, this effectively directs the attention of all queries to the malicious token and prevents the model from detecting the remaining objects. The right column compares queries’ attention weights to the target key, marked by a red box, between clean and patched inputs and highlights large attention weights drawn by this adversarial key.

Abstract

Neural architectures based on attention such as vision transformers are revolutionizing image recognition. Their main benefit is that attention allows reasoning about all parts of a scene jointly. In this paper, we show how the global reasoning of (scaled) dot-product attention can be the source of a major vulnerability when confronted with adversarial patch attacks. We provide a theoretical understanding of this vulnerability and relate it to an adversary’s ability to misdirect the attention of all queries to a single key token under the control of the adversarial patch. We propose novel adversarial objectives for crafting adversarial patches which target this vulnerability explicitly. We show the effectiveness of the proposed patch attacks on popular image classification (ViTs and DeiTs) and object detection models (DETR). We find that adversarial patches occupying 0.5% of the input can lead to robust accuracies as low as 0% for ViT on ImageNet, and reduce the mAP of DETR on MS COCO to less than 3%.

1. Introduction

The attention mechanism plays a prominent role in recent success of transformers for different language and image processing tasks. Recent breakthroughs in image recognition using vision transformers [12, 23, 37] have inspired different architectures’ design for tackling tasks such as object detection [4, 8], semantic segmentation [35, 41, 46, 48], image synthesis [13, 17, 40], video understanding [3, 6, 15, 19, 24, 30, 36, 49], and low-level vision tasks [9, 21, 42, 47]. These different transformers use the dot-product attention mechanism as an integral component of their architectural design to model global interactions among different input or feature patches. Understanding robustness of these dot-product attention-based networks against adversarial attacks targeting security-critical vulnerabilities is important for their deployment into real-world applications.

The increasing interest of research in transformers across vision tasks has motivated several recent works [2, 5, 7, 14,
16, 27–29, 34, 43] to study their robustness against adversarial attacks. Some prior works [2, 5, 28, 34] have hypothesized that transformers are more robust than convolutional neural networks (CNNs) against these attacks. On the other hand, [14, 16, 27, 43] have shown that vision transformers are not an exception and are also prone to adversarial attacks. In particular, [43] shows that an adversarial attack can be tailored to transformers to achieve high adversarial transferability. These findings indicate that robustness evaluation protocols (attacks) designed for CNNs might be suboptimal for transformers. In the same line of work, we identify a principled vulnerability in the widely-used dot-product attention in transformers that can often be exploited by image-based adversarial patch attacks.

Dot-product attention computes the dot-product similarity of a query token with all key tokens, which is later normalized using the softmax operator to obtain per token attention weights. These attention weights are then multiplied with value tokens to control the value token’s contribution in the attention block. Gradient-based adversarial attacks backpropagate gradients through all the components in the architecture including the attention weights. We observe that on pretrained vision transformers, because of the softmax, the gradient flow through the attention weights is often much smaller than the flow through the value token (refer to Sec. 4.1). Consequently, gradient-based attacks with a standard adversarial objective are biased towards focusing on adversarial effects propagated through the value token and introduce little or no adversarial effect on the attention weights, thus limiting the attack’s potential.

Our work aims to adversarially affect the attention weights, even if those operate in a saturated regime of softmax where gradient-based adversarial attacks are impaired. Specifically, we propose losses that support the adversary in misdirecting the attention of most queries to a key token that corresponds to the adversarial patch, i.e., to increase the attention weights of the queries to the targeted key token. We further study necessary conditions (refer to Sec. 4.2) for a successful attack that misdirects attention weights: both (a) having projection matrices with large singular values and (b) having higher embedding dimensions, allows amplifying the effect of perturbations in a single token, and thus changing the embedding of a single key considerably. Moreover, (c) having less centered inputs (larger absolute value of input mean) to the dot-product attention results in distinct clusters of queries and keys that are distant from each other. Under this condition, moving a single key closer to the query cluster center can make the key most similar to the majority of queries simultaneously (see Figure 1).

We propose a family of adversarial losses and attacks called Attention-Fool that directly acts on the dot-product outputs (pre-softmax dot-product similarities). These losses optimize the adversarial patch to maximize the dot-product similarity of all the queries to a desired key (typically the key whose token corresponds to the input region with adversarial patch). This approach maximizes the number of queries that attend to the targeted key in an attention head, please refer to Figure 2 for an illustration. We apply these losses at multiple attention heads and layers in transformers to misdirect the model’s attention from the image content to the adversarial patch in order to encourage wrong predictions. We show that our Attention-Fool adversarial losses improve gradient-based attacks against different vision transformers ViTs [12] and DeiTs [37] for image classification and also significantly improve the attack’s effective on the object detection model DETR [8].

Our contributions can be summarized as follows: We

- Identify the necessary conditions for the existence of a vulnerability in dot-product attention layers weights, see Section 4.2.
- Provide reasons why this vulnerability may not be fully exploited by vanilla gradient-based adversarial attacks, see Section 4.1.
- Introduce Attention-Fool, a new family of losses which defines losses directly on the attention layers dot-product similarities, see Section 5.
- Show that transformers for image classification and object detection are highly sensitive to small adversarial patches, see Section 6.

2. Related Work

Recent successes of Vision Transformers (ViTs) [12] have inspired several works [2, 5, 14, 16, 27–29, 34, 43] that study their robustness against adversarial attacks. While some works [2, 5, 28, 29, 34] have hypothesized that ViTs are more robust than CNNs under different white- and black-box transfer attack settings (including universal adversarial patches), others [7, 27, 29, 43] have claimed that ViTs are at least as vulnerable as CNNs. In particular, [2] attributed ViT’s presumed robustness to its ability to capture global features. [5, 34] analyzed that ViTs rely on low-frequency features that are robust to adversarial perturbations. [34] have stated that ViTs have better certified robustness than CNNs and have further suggested that adversarially trained ViTs have comparable robustness to their CNN counterparts. However, they observed catastrophic overfitting for ViTs when using fast adversarial training [45], suggesting the need for improvements in adversarial training.

On the other hand, [7] have claimed that ViTs pretrained on larger datasets are at least as vulnerable as CNN counterparts. [27, 29, 43] suggested that adversarial transferability between ViTs and from ViTs to CNNs can be improved by carefully tailoring adversarial attacks to the transformer architecture. [27] explored ensembling of CNNs and ViT models to improve attack transferability. The authors of
Figure 2. Example of dot-product (self-)attention mechanism for clean (left) and adversarial patch attack (right) settings. Here, $q, k,$ and $v$ stand for projected queries, keys, and value tokens of input features. Left: dot-product attention computes dot-product similarities of a query with all keys, which is later normalized using softmax to obtain per token attention weights. These are multiplied with value tokens to control their contributions in an attention block. Right: Attention-Fool losses optimize the adversarial patch in input at $X_3$ to maximize dot-product similarity of all the queries to the key $k_3$ (marked in red/black), which corresponds to moving $k_3$ closer to the queries cluster. The increase in dot-product similarity of queries with $k_3$ misdirects the model’s attention from image content to adversarial patch.

[29] proposed a self-ensemble technique: split a single ViT model into an ensemble of networks to improve the transferability. [43] proposed an attack that skips attention gradients to generate highly transferable adversarial perturbations.

In this work, we aim to understand robustness of the widely-used dot-product attention in transformers and expose its vulnerability to adversarial patch attacks. These are constructed by tailoring adversarial objectives to specifically fool the dot-product attention mechanism. Concurrent to our work, [14, 16] also fool the attention mechanism in transformers using image patches. [16] uses a conventional adversarial patch attack to mislead the model attention to the perturbed patch to promote wrong predictions. They also show that the adversarial patch generalizes to different positions in the image. [14] optimizes the adversarial patch to increase its attention weights (post-softmax) from all other patches to attack the model. We discuss limitations of optimizing post-softmax attention weights as in [14] (refer to Section 4.1) and propose to optimize directly the pre-softmax dot-product similarities of query and keys to draw the attention to the adversarial patch.

Besides optimizing the content of the patch, there exist also methods for optimizing the location of the patch in the input. Joshi et al. [18] select the location of adversarial patches based on token saliency. Fu et al. [14] selects a salient image patch to contain the adversarial patch, based on how much attention the image patch draws in the clean image. Our work abstracts from selecting the patch location and focuses on the loss for the optimization of the patch’s content. Any patch location-selection method is orthogonal and could be combined with our Attention Fool-losses.

This work focuses on dot-product attention, widely used in transformers. We leave investigation of vulnerabilities of other attention mechanisms [20, 39, 50] to future work.

3. Preliminaries

We introduce the objective for patch robustness evaluation, summarize an optimization algorithm for finding adversarial patches, and recap scaled dot-product attention.

Generic Objective Formulation. Given a (normalized) image $x \in [0,1]^{3 \times h \times w}$ and associated label $y$, we craft an adversarial patch $p \in [0,1]^{3 \times p_h \times p_w}$ with $p_h \ll h, p_w \ll w$ that maximizes the following objective:

$$
\text{arg max}_p \mathcal{L}(f(F(x, p, L)), y),
$$

with $L$ specifying the location of the patch $p$ within the larger image $x$, $F$ a function to apply the patch onto the image (i.e., overwriting an input region for a given size), and $f$ being the target model. For classification tasks, we are interested in the 0-1 loss $\mathcal{L}(x, y) = \mathcal{L}_{0,1}(x, y) = \{0 \quad x = y \\
1 \quad x \neq y\}$, which corresponds to finding patches that maximize misclassifications. We note that the constraint $p \in [0,1]^{3 \times p_h \times p_w}$ can be rewritten as $\|p - 0.5\|_\infty \leq 0.5$.

Threat Model. We focus on a white-box threat model, where an adversary has access to the model’s internals (this includes intermediate network layers outputs). As in [10, 14], we do not consider the imperceptibility of the patch to be a requirement. We also focus on a single-patch fixed-location threat model (L in Equation 1 is fixed a priori). Note that methods for choosing the patch location [14, 18] could be combined with the proposed approach.

Optimization Algorithm. For the patch optimization, throughout the paper, we use Projected Gradient Descent (PGD) [26] for $\ell_\infty$-norm bounded perturbations:

$$
p^{t+1} = p^t + \alpha \cdot \text{sgn}(\nabla_p \mathcal{L}(f(F(x, p, L)), y)).
$$

(2)
We initialize \( p_0 \) uniform randomly from \([0, 1]^{3 \times p_h \times p_w}\). Since the 0-1 loss \( \mathcal{L}_{0,1} \) is piecewise-constant, it is not suited for gradient-based optimization. Accordingly, a surrogate loss such as the cross-entropy \( \mathcal{L} = \mathcal{L}_{ce} \) is used typically. However, we propose alternative losses in Section 5.

**Dot-Product Attention.** In its basic form, dot-product attention [25,38] computes, for every query, attention weights as the dot-product of the query to all keys. The softmax function is then applied over the key dimension. These attention weights are then multiplied by the values:

\[
\text{Attention}(Q, K, V) = \text{softmax}(QK^\top)V. 
\]  
(3)

Here, \( Q \in \mathbb{R}^{n \times d_{\text{model}}} \), \( K \in \mathbb{R}^{n \times d_{\text{model}}} \), and \( V \in \mathbb{R}^{n \times d_{\text{model}}} \) are the matrices of \( n \) queries, keys, and values, respectively. According to Vaswani et al. [38], for large values of \( d_{\text{model}} \), the dot-product between queries and keys may grow large in magnitude. This has the effect of pushing the softmax function into the saturated regime, where it has extremely small gradients. This is due to the exponentiation of individual query-key dot-products in the softmax function. Because this can be harmful for training, they introduce scaled dot-product attention, where \( QK^\top \) is scaled by \( \frac{1}{\sqrt{d_k}} \), where \( d_k \) is the dimension of the keys.

In practice, using \( H > 1 \) attention heads by linearly projecting queries, keys, and values \( H \) times to \( d_k \), \( d_k \), and \( d_k \) dimensions was found to be beneficial [38]. The output of the \( h \)-th attention head (AH) becomes:

\[
\text{AH}_h(Q, K, V) = \text{softmax}\left(\frac{QW^h_Q(KW^h_K)^\top}{\sqrt{d_k}}\right)VW^h_V,
\]  
(4)

where \( W^h_Q \in \mathbb{R}^{d_{\text{model}} \times d_k} \), \( W^h_K \in \mathbb{R}^{d_{\text{model}} \times d_k} \), \( W^h_V \in \mathbb{R}^{d_{\text{model}} \times d_k} \) are (learned) projection matrices. The outputs of individual attention heads are concatenated and multiplied by another learned projection matrix \( W_{\mathcal{A}} \in \mathbb{R}^{H \times d_k \times d_{\text{model}}} \).

A special case is self-attention with \( Q = K = V \) in \( \mathbb{R}^{n \times d_{\text{model}}} \), which is typically used in encoder layers of image recognition models. We define the attention weights of the \( h \)-th head on \( X \) via \( A_h(X) = \text{softmax}\left(\frac{XW^h_X((XW^h_X)^\top)}{\sqrt{d_k}}\right) \in \mathbb{R}^{n \times n} \). The \( h \)-th self-attention head becomes:

\[
\text{SelfAH}_h(X) = A_h(X)XW^h_V,
\]  
(5)

### 4. Robustness of Dot-Product Attention

In this section, we first discuss why (scaled) dot-product self-attention is challenging for gradient-based adversarial attacks such as PGD. We then provide an example of an adversarial vulnerability in the attention weights themselves.

#### 4.1. Gradient of Dot-Product Self-Attention

For computing \( \nabla_p \mathcal{L}(F(x, p, L), y) \) in Eq. 2, it is necessary to backpropagate through the entire model. This requires the gradient \( \nabla_X \text{SelfAH}_h(X) \) for every attention layer and head \( h \). With the product-rule we obtain:

\[
\nabla_X \text{SelfAH}_h(X) = \left[ (\nabla_X A_h(X))X + A_h(X)1_X\right]W^h_V,
\]

where \( 1_X \) is a matrix of ones with the same shape as \( X \).

An important property of the gradient \( \nabla_X \text{SelfAH}_h(X) \) is accordingly the element-wise ratio \( \left\lfloor \frac{\nabla_X A_h(X)X}{A_h(X)X} \right\rfloor \). We summarize the median value of this ratio over tokens and heads in Table 1 for different models and layers. As seen, the typical regime (that is, for \( > 50\% \) of the cases) is that \( \nabla_X A_h(X)X \) is smaller than \( A_h(X)X \) by a factor of \( \approx 20 \) for ViTs [12], DeTs [37], and for the inner encoder layers of DETR [8]. In this setting, one can approximate \( \nabla_X \text{SelfAH}_h(X) \approx (A_h(X)X)W^h_V \), that is: the gradient considers the attention weights \( A_h(X) \) as effectively constant. Accordingly, gradient-based attacks such as PGD based on an end-to-end loss such as \( \mathcal{L}_{ce} \) would be biased towards focusing on adversarial effects in \( X \) that can be propagated (linearly) via the values \( V = XW^h_V \) in self-attention, while effectively ignoring potentially adverse (and non-linear) effects of \( X \) propagated via the attention weights \( A_h(X) \). We note that also later stages of model training with gradient-based optimizers can be negatively affected by this property of dot-product attention. Studying this in more detail is left to future work.

#### 4.2. Robustness of Dot-Product Attention Weights

We now study to which extent attention weights \( A_h(X) \) of dot-product attention can be affected by an adversarial patch attack. For this, we use a controlled setting with normally distributed \( X \), where each feature has mean \( \mu \) and variance \( 1: X_j \sim \mathcal{N}(\mu \cdot 1, 1) \). Moreover, we choose \( d_k = d_{\text{model}} \) and diagonal \( W_Q = -w \cdot \mathbb{I}_{d_k} \) and \( W_K = w \cdot \mathbb{I}_{d_k} \), that is \( W_Q \) and \( W_K \) having both scale \( w \) but opposite signs.

We study a simple threat model: the adversary can only modify \( X_0 \), the first of \( n \) entries of \( X \) (\( X \) can be considered as embedding of patches and \( X_0 \) as corresponding to the embedded adversarial patch), with the constraint \( \|X^{adv}_0 - X_0\|_\infty \leq \epsilon \). Moreover, the adversaries goal is to achieve \( \left[ \frac{1}{n} \sum_j (A_h(X^{adv})_{0j} \geq 0.99) \right] > 0.95 \), that is: at
least 95% of the queries need to attend to the first key with attention weights greater or equal 0.99. By design, setting $X_{\text{adv}} = X_0 - \epsilon$ is a strong attack for $\mu > 0$ because $-1 \cdot \epsilon$ corresponds to the direction of $(W_Q - W_k)(\mu \cdot 1)$, the difference of projected query and key mean.

We study how $\epsilon$ needs to be chosen as a function of $\mu$, $w$, and $d_k$ for a successful attack with the above attack and threat model. Results are shown in Figure 3. In general, one can observe that a successful attack with a smaller perturbation amount $\epsilon$ requires: (a) increasing the scale $w$ of the projection matrices, (b) higher embedding dimensions $d_k$, and (c) less centered inputs $X$ (larger $|\mu|$). The findings (a) and (b) can be attributed to higher dimensions and larger weights allow the effect of minor perturbations in each input dimension to be amplified. Upon closer inspection (see Section B in Appendix), we can attribute finding (c) to a separation of projected keys and queries into distinct clusters for less centered inputs: all queries are close to each other, all keys are close to each other, but query-key pairs are always distant from each other. In this case, a single key can be made to be the most similar to each query, just by moving this key in the direction of the query cluster (see Figure 1 for a real data illustration).

In which regime does dot-product attention typically operate in trained image transformers? For many architectures, $d_k$ is relatively large by design. Moreover, the input of dot-product attention is typically not centered (zero-mean) due to enabled affine transformations within the normalization layers before self-attention. Many implementations, e.g. [31], also use affine rather than linear query/key projections in the heads, where different biases in the projection of key and queries will have a similar effect as non-centered inputs. Lastly, we also observe that for many vision transformers, the product of projection weight matrices $W_Q(W_K)^T$ has very large maximal singular value at convergence, compared to the randomly initialized projection weight matrices (see Table 2). Large singular values can be considered to be analogous to the “large weight scales” in our controlled setting. We thus expect that dot-product attention of trained vision transformers typically operates in a setting where the attention weights $A_h(X)$ can be a source of vulnerability to patch attacks, but gradient-based attacks such as PGD are biased towards ignoring this vulnerability (as discussed in Sec. 4.1).

5. Attention-Fool

We have discussed that attention weights $A_h(X)$ can be largely affected by adversarial patches in principle but we have also shown that gradient-based attacks are impaired in exploiting this vulnerability. In order to encourage fooling of attention weights, we introduce a family of losses that are defined directly on the pre-softmax attention logits $B_h(X) = \frac{XW_h(XW_h)^T}{\sqrt{d_k}}$. We denote attacks based on these losses as Attention-Fool. We note that Attention-Fool losses are always maximized by the attacker, regardless of whether an attack is targeted or untargeted. Moreover, while this paper focuses on self-attention, it is straightforward to extend these losses to cross-attention.

$L_{kq}^{hl}$: Attacking a single attention head. We first focus on adversarially modifying attention weights of a specific head $h$ in a specific self-attention layer $l$. We propose a loss $L_{kq}^{hl}$ that aims at maximizing the query attention to the $i^*$-th projected key, that is: maximize the number of queries that devote the majority of their attention to this key. Naturally, $i^*$ is chosen such that its token corresponds to the region in the input where the adversarial patch has been placed.

We denote projected queries by $F_Q^{hl} = X^{hl}W_Q^{hl} \in \mathbb{R}^{d_k \times n}$,

![Figure 3. Minimum $\ell_\infty$ norm perturbation $\epsilon$ required for reaching the attacker’s goal (i.e., forcing queries to attend to the first key) on the controlled setting from Section 4.2. Increased weight scale $w$, higher embedding dimension $d_k$, and larger input mean norm $\mu$ simplify the attack: as the three quantities increase, the $\epsilon$ perturbation required to fulfill the goal decreases.](image-url)
and projected keys by \( P^{bl}_h = X^{bl}_h W^{bl}_K \in \mathbb{R}^{n \times d_k} \). We have \( B^{bl} = \frac{P^{bl}_Q (P^{bl}_Q)^T}{\sqrt{d_n}} \in \mathbb{R}^{n \times n} \), where the first dimension indexes queries and the second one keys. Each element of \( B^{bl} \) quantifies the dot-product similarity between a key and a query in the respective attention head \( h \) and layer \( l \). We now set \( \ell^{hl}_{kq} = \frac{1}{n} \sum_j P^{bl}_{ji}, \) where \( j \) indexes the queries. Maximizing \( \mathcal{L}^{hl}_{kq} \) thus corresponds to maximizing average dot-product similarity between queries and the target key.

\( \mathcal{L}_{kq} \): Attacking all attention layers and heads simultaneously. In the previous paragraph, we have introduced a loss that targets a single head and layer. However, in general fooling a single head might not be sufficient and it is also a priori unclear which head would be the most vulnerable one. The same applies to the choice of the layer, when there are multiple layers using dot-product attention. Because of this, we propose applying the loss to all layers and heads simultaneously. However, simply averaging \( \mathcal{L}^{hl}_{kq} \) over all heads and layers might not be optimal because it may favour many smaller head-wise changes overall rather than successfully fooling a subset of heads and layers to a larger extent. Because of this, we will utilize a smooth maximum \( \max(x) = \log \sum_j e^{x_j} \) over \( \mathcal{L}^{hl} \). Specifically, we define \( \mathcal{L}^{l}_{kq} = \log \sum_h e^{\mathcal{L}^{hl}_{kq}} \) \( \forall l \) and \( \mathcal{L}_{kq} = \log \sum_j e^{\mathcal{L}^{lj}_{kq}}. \) We empirically compare the choice of this smooth maximum over mean and hard maximum in Section C.1 and the choice of \( l \) in \( \mathcal{L}^{l}_{kq} \) in Section C.2. We visualize \( \mathcal{L}_{kq} \)'s effect on each network layer in Figure 4.

In order to make the losses of different heads and layers commensurable, we also propose to normalize scales of projected keys and queries via \( \tilde{P}^{bl}_K = \frac{P^{bl}_K}{\sqrt{d_k} \| P^{bl}_K \|_{1,2}} \) and \( \tilde{P}^{bl}_Q = \frac{P^{bl}_Q}{\sqrt{d_n} \| P^{bl}_Q \|_{1,2}} \), where \( \| X \|_{1,2} = \sum_i \sqrt{\sum_j X_{ij}^2} \) is the \( L_{1,2} \) norm. Note that the normalization is applied per individual head and makes the average \( \ell_2 \) norm of queries and keys equal to 1. \( B^{bl} \) is then computed based on \( \tilde{P}^{bl}_Q \) and \( \tilde{P}^{bl}_K \), making the losses \( \mathcal{L}^{hl}_{kq} \) commensurable. We empirically evaluate the effect of this normalization in Section C.3 of the supplementary material.

\( \mathcal{L}_{kq}^*: \) Targeting a special class token. The \( \mathcal{L}_{kq} \) loss treats all queries equally and aims at misdirecting the attention of the majority of queries towards the adversarial key token. However, for many architectures not all queries are created equally, for instance for ViTs [12] and DeiTs [37], there exists a special class token, which is supposed to accumulate class evidence over the layers. We propose a version of the \( \mathcal{L}_{kq} \) loss that specifically targets the attention of a certain query, for instance the one corresponding to the class token. Let \( j^* \) be the index of the query that shall be targeted. We define \( \mathcal{L}^{hl}_{kq^*} = B^{hl}_{j^*}, \) and generalize to \( \mathcal{L}_{kq} \), as above using smooth maximum over heads and layers.

In the following experiments, we will use combinations of \( \mathcal{L}_{kq} \) or \( \mathcal{L}_{kq^*} \) with standard cross-entropy loss \( \mathcal{L}_{ce} \) for the adversarial patch optimization of Equation 1. We denote the resulting set of attacks leveraging the weakness in dot-product attention as Attention-Fool. Note that Attention-Fool is different from concurrent work Patch-Fool [14] that defines a loss on the post-softmax attention weights, averaged of heads \( h \) and queries \( j \): \( \mathcal{L}_{PF} = \sum_h \sum_j A_h(X)_j \). In particular, Attention-Fool (in contrast to Patch-Fool) is not affected by the small gradient issue described in Section 4.1 since it is defined on pre-softmax attention weights.

6. Evaluation of Attention-Fool

We evaluate the robustness of different vision transformers against adversarial patches generated using our proposed Attention-Fool adversarial losses. We first investigate the robustness of ViT [12] and the improved DeiT [37] in Section 6.2. Then we show how Attention-Fool generalizes to DETR [8] in Section 6.3, which uses a hybrid CNN plus Transformer architecture to perform object detection.

6.1. Evaluation Setup

We use PGD as the optimizer to solve Equation 1. We set PGD’s initial step size \( \alpha^0 = 8/255 \) and schedule it with a cosine decay: \( \alpha^{(t+1)} = \alpha^{(0)} \frac{1}{2} (1 + \cos(\pi \frac{t}{T})) \). Additional-
ally, we add a stronger baseline to the adversarial patch robustness evaluation, we use an improved version of PGD by adding normalized-momentum in the optimization \[11\]:

\[m^{(t)} = \beta m^{(t-1)} + (1 - \beta) \nabla_{\theta} f(\theta; \mathbf{x}) / ||\nabla_{\theta} f(\theta; \mathbf{x})||_2,\]

and use \(m^{(t)}\) instead of the gradient \(\nabla_{\theta} f(\theta; \mathbf{x})\) in PGD.

### 6.2. Attention-Fool on Vision Transformer

We use the new Attention-Fool attack described in Section 5 to compare the effectiveness of adversarial patches optimized with the new losses \(\mathcal{L}_{kq}\) and \(\mathcal{L}_{kq^*}\) with adversarial patches optimized using cross-entropy loss \(\mathcal{L}_{ce}\). We use 1,000 images from the ImageNet 2012 [33] dataset and perform an untargeted attack, where \(y\) in Eq. 1 is the image ground truth. We also report the results on a targeted attack in Section D in Appendix. For the experiment, we place an adversarial patch of 16×16 pixels in the top left corner, whereas placing the patch in the center could have a larger effect on a CNN.

We include additional comparisons with the “Attention-Aware Loss” of Patch-Fool (Section 4.4 in [14]). For a controlled comparison with Patch-Fool, we perform two adjustments: (i) we disregard the saliency-based selection of the patch location (Section 4.3 in [14]) and (ii) we replace Adam with PGD (momentum = 0.9). This abstracts from attack specifics and allows us to compare losses directly. The results show that Patch-Fool’s Attention-Aware loss underperforms the Attention Fool losses; confirming the limitations of optimizing post-softmax attention weights (discussed in Sec. 4.1). Moreover, we compare to Patch-RS [10], a random-search based black-box patch attack. The inferior results of Patch-RS indicate that gradient-free attacks are no viable alternative for exploiting the existent vulnerabilities in attention weights.

### 6.3. Attention-Fool on DETR

Since Attention-Fool targets dot-product attention layers, it can be adapted to target various tasks and architectures that use this attention. Here, we apply Attention-Fool to object detection with DETR [8], which combines a CNN backbone with a Transformer encoder-decoder.

**Attention-Fool Configuration.** While \(\mathcal{L}_{kq}\) showed superior results for ViT/DeiT in Section 6.2, it is not applicable to DETR due to the absence of a class token in DETR. In contrast, we use \(\mathcal{L}_{kq}^{(1)}\), a loss targeting all queries and heads but only the first transformer encoder layer \((l = 1)\) rather than all at once. We focus on \(l = 1\) because we hypothe-
size that the transition from CNN to transformer encoder in DETR is specifically brittle due to large activations from the CNN backbone. For a similar reason, we aggregate losses across attention heads using the maximum function rather than $\text{smax}$ used in Section 4.

Evaluation Setup. We evaluate a targeted Attention-Fool attack on DETR by choosing the target to be the background class, which effectively forces missed detections (i.e., false negatives). We evaluate four pretrained DETR models from the official repository [1], with backbone either ResNet50 or ResNet101, and with or without a dilation in the ResNet layer 5 convolution (DC5). Note that DC5 models have twice the resolution in the backbone-extracted feature map, which is used as the encoder input. We select 100 images from the MS COCO 2017 validation set [22], and use the default DETR validation image loader which obtained mAP as we combine it with the Attention-Fool loss $L_{kq}^{(1)}$. We test three different patch sizes, $64 \times 64$, $56 \times 56$, and $48 \times 48$, which occupy $<0.64\%$, $<0.49\%$ and $<0.36\%$ of the input image, respectively. Given the higher complexity of DETR (compared to ViT), we increase the number of PGD iterations to 1000 in this experiment. We report the resulting mAPs in Table 4: baseline mAP on clean images, mAP under a targeted attack which uses $L_{ce}$, and the mAP change when we use the combined Attention-Fool $L_{ce} + L_{kq}^{(1)}$ loss. Table 4 shows that the addition of $L_{kq}^{(1)}$ reduces the mAP performance across all models and all patch sizes. We find that larger models with DETR101 are more vulnerable to Attention-Fool, where the mAP can be reduced down to 2.07 – this corresponds to suppressing the detection of the vast majority of objects. See Figure 1 for an illustration.

Table 4 also presents a setting where we do not use any loss on the model’s output but solely focus on misdirecting the first encoder layer’s attention (“$L_{kq}^{(1)}$ only”). In most cases, this results in considerably lower robust accuracy than using $L_{ce}$ directly, indicating that for DETR “fooling attention is all you need” for misclassification.

7. Conclusion

We revisited the robustness of transformers for image recognition against patch attacks. We identified properties of the dot-product attention’s gradient that bias vanilla patch attacks to mostly ignore a core vulnerability of these attention weights. This presumably caused prior works to overestimate robustness. We propose Attention-Fool, which directly targets the dot-product attention weights and allows much tighter robustness estimates. In summary, Attention-Fool improves vanilla robustness evaluation across all considered vision transformers, and is able to fool DETR’s global object detection with a tiny remote patch.

Limitations. We focused on dot-product attention. Other attention mechanisms [20, 39, 50] are likely not affected to the same degree by the identified weakness. However, since dot-product attention is the predominant attention mechanism in transformers, our approach is broadly applicable.

Potential negative societal impact. Adversarial attacks such as ours can be used for benign purposes like reliably evaluating robustness of ML systems as well as malign ones like exploiting weaknesses of these systems. Our research provides the basis for future work on more robust attention mechanisms that can mitigate the identified vulnerability and the resulting potential for negative societal impact.
References

[1] DETR: End-to-End Object Detection with Transformers. https://github.com/facebookresearch/detr. Accessed: 2021-11-16.

[2] Ahmed Al Dahdooh, Wassim Hamidouche, and Olivier Deferges. Reveal of vision transformers robustness against adversarial attacks. arXiv preprint arXiv:2106.03734, 2021.

[3] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. arXiv preprint arXiv:2103.15691, 2021.

[4] Josh Beal, Eric Kim, Eric Tzeng, Dong Huk Park, Andrew Zhai, and Dmitry Kislyuk. Toward transformer-based object detection, 2020.

[5] Philipp Benz, Soomin Ham, Chaoning Zhang, Adil Karjaev, and In So Kweon. Adversarial robustness comparison of vision transformer and mlp-mixer to cnns. arXiv preprint arXiv:2110.02797, 2021.

[6] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding? arXiv preprint arXiv:2102.05095, 2021.

[7] Srinadh Bhojanapalli, Ayan Chakrabarti, Daniel Glasner, Daliang Li, Thomas Unterthiner, and Andreas Veit. Understanding robustness of transformers for image classification. arXiv preprint arXiv:2103.14586, 2021.

[8] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In European Conference on Computer Vision, pages 213–229. Springer, 2020.

[9] Hanbing Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yixing Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12873–12883, 2021.

[10] Francesco Croce, Maksym Andriushchenko, Naman D Singh, Nicolas Flammarion, and Matthias Hein. Sparse-rs: a versatile framework for query-efficient sparse-black-box adversarial attacks. arXiv preprint arXiv:2006.12834, 2020.

[11] Yinpeng Dong, Fangzhou Liao, Tianyu Pang, Hang Su, Jun Zhu, Xiaolin Hu, and Jianguo Li. Boosting adversarial attacks with momentum. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2018.

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthia Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929, 2020.

[13] Patrick Esser, Robin Rombach, and Bjorn Ommer. Taming transformers for high-resolution image synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 12873–12883, 2021.

[14] Yonggan Fu, Shunyao Zhang, Shang Wu, Cheng Wan, and Yingyan Lin. Patch-fool: Are vision transformers always robust against adversarial perturbations? In International Conference on Learning Representations, 2022.

[15] Rohit Girdhar, Joao Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 244–253, 2019.

[16] Jindong Gu, Volker Tresp, and Yao Qin. Are vision transformers robust to patch-wise perturbations?, 2022.

[17] Yifan Jiang, Shiyu Chang, and Zhangyang Wang. Transgan: Two transformers can make one strong gan. arXiv preprint arXiv:2102.07074, 1(3), 2021.

[18] Ameya Joshi, Gauri Jagatap, and Chinmay Hegde. Adversarial token attacks on vision transformers. arXiv preprint arXiv:2110.04337, 2021.

[19] Sangho Lee, Youngjae Yu, Gunhee Kim, Thomas Breuel, Jan Kautz, and Yale Song. Parameter efficient multimodal transformers for video representation learning. arXiv preprint arXiv:2012.04124, 2020.

[20] James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. Fnet: Mixing tokens with fourier transforms. arXiv preprint arXiv:2105.03824, 2021.

[21] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 1833–1844, 2021.

[22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In European conference on computer vision, pages 740–755. Springer, 2014.

[23] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows, 2021.

[24] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. arXiv preprint arXiv:2106.13230, 2021.

[25] Minh-Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. CoRR, abs/1508.04025, 2015.

[26] Aleksander Madry, Aleksandar Makelov, Ludwig Schmidt, Dimitris Tsipras, and Adrian Vladu. Towards deep learning models resistant to adversarial attacks. arXiv preprint arXiv:1706.06083, 2017.

[27] Kaleel Mahmood, Rigel Mahmood, and Marten Van Dijk. On the robustness of vision transformers to adversarial examples. arXiv preprint arXiv:2104.02610, 2021.

[28] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Munawar Hayat, Fahad Shabbaz Khan, and Ming-Hsuan Yang. Intriguing properties of vision transformers. arXiv preprint arXiv:2105.10497, 2021.

[29] Muzammal Naseer, Kanchana Ranasinghe, Salman Khan, Fahad Shabbaz Khan, and Fatih Porikli. On improving adversarial transferability of vision transformers. arXiv preprint arXiv:2106.04169, 2021.
[30] Daniel Neimark, Omri Bar, Maya Zohar, and Dotan Asselmann. Video transformer network. arXiv preprint arXiv:2102.00719, 2021.

[31] PyTorch. PyTorch multi-head attention implementation, 2021. https://pytorch.org/docs/1.10.0/generated/torch.nn.MultiheadAttention.html.

[32] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. J. Comput. Appl. Math., 20(1):53–65, 1987.

[33] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision (IJCV), 115(3):211–252, 2015.

[34] Rulin Shao, Zhouxing Shi, Jinfeng Yi, Pin-Yu Chen, and Cho-Jui Hsieh. On the adversarial robustness of visual transformers. arXiv preprint arXiv:2103.15670, 2021.

[35] Robin Strudel, Ricardo Garcia, Ivan Laptev, and Cordelia Schmid. Segmenter: Transformer for semantic segmentation. arXiv preprint arXiv:2105.05633, 2021.

[36] Chen Sun, Austin Myers, Carl von der Vorst, Kevin Murphy, and Cordelia Schmid. Videobert: A joint model for video and language representation learning. In Proceedings of the IEEE/CVF International Conference on Computer Vision, pages 7464–7473, 2019.

[37] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers; distillation through attention. In International Conference on Machine Learning, volume 139, pages 10347–10357, July 2021.

[38] Ashish Vaswani, Noam Shazeer, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In Advances in Neural Information Processing Systems, pages 5998–6008, 2017.

[39] Sinong Wang, Belinda Z Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. arXiv preprint arXiv:2006.04768, 2020.

[40] Xinpeng Wang, Chandan Yeshwanth, and Matthias Nießner. Sceneformer: Indoor scene generation with transformers. arXiv preprint arXiv:2012.09793, 2020.

[41] Yuqing Wang, Zhaoliang Xu, Xinlong Wang, Chunhua Shen, Baoshan Cheng, Hao Shen, and Huaxia Xia. End-to-end video instance segmentation with transformers. arXiv preprint arXiv:2106.03106, 2021.

[42] Zhendong Wang, Xiaodong Cun, Jianmin Bao, and Jianzhuang Liu. Uformer: A general u-shaped transformer for image restoration. arXiv preprint arXiv:2106.03106, 2021.

[43] Zhipeng Wei, Jingjing Chen, Michal Goldblum, Zuxuan Wu, Tom Goldstein, and Yu-Gang Jiang. Towards transferable adversarial attacks on vision transformers. arXiv preprint arXiv:2109.04176, 2021.

[44] Ross Wightman. Pytorch image models, 2019. https://github.com/rwightman/.

[45] Eric Wong, Leslie Rice, and J Zico Kolter. Fast is better than free: Revisiting adversarial training. arXiv preprint arXiv:2001.03994, 2020.

[46] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M. Alvarez, and Ping Luo. Segformer: Simple and efficient design for semantic segmentation with transformers, 2021.

[47] Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baoming Guo. Learning texture transformer network for image super-resolution. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5791–5800, 2020.

[48] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, Yabiao Wang, Yanwei Fu, Jianfeng Feng, Tao Xiang, Philip HS Torr, et al. Rethinking semantic segmentation from a sequence-to-sequence perspective with transformers. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6881–6890, 2021.

[49] Luowei Zhou, Yingbo Zhou, Jason J Corso, Richard Socher, and Caiming Xiong. End-to-end dense video captioning with masked transformer. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 8739–8748, 2018.

[50] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. arXiv preprint arXiv:2010.04159, 2020.