

Towards Robust Image Classification Using Sequential Attention Models

Daniel Zoran

Mike Chrzanowski

Po-Sen Huang
Pushmeet Kohli

Sven Gowal

Alex Mott

DeepMind
London, UK

danielzoran@google.com

Abstract

In this paper we propose to augment a modern neural-network architecture with an attention model inspired by human perception. Specifically, we adversarially train and analyze a neural model incorporating a human inspired, visual attention component that is guided by a recurrent top-down sequential process. Our experimental evaluation uncovers several notable findings about the robustness and behavior of this new model. First, introducing attention to the model significantly improves adversarial robustness resulting in state-of-the-art ImageNet accuracies under a wide range of random targeted attack strengths. Second, we show that by varying the number of attention steps (glances/fixations) for which the model is unrolled, we are able to make its defense capabilities stronger, even in light of stronger attacks — resulting in a “computational race” between the attacker and the defender. Finally, we show that some of the adversarial examples generated by attacking our model are quite different from conventional adversarial examples — they contain global, salient and spatially coherent structures coming from the target class that would be recognizable even to a human, and work by distracting the attention of the model away from the main object in the original image.

1. Introduction

Recent years have seen great advances in the use and application of deep neural network models. From large scale image classification [21] to speech recognition [25], the performance of these models has steadily improved, making use of new hardware advances, more memory and better optimization strategies. The leading model paradigm for such tasks, however, has not changed significantly since the original AlexNet paper [31]. Models are still, predominately, built in a purely feed-forward manner, interleaving convolutional layers (often with small kernels with limited support) and simple non-linearities [40]. Recently introduced ResNets

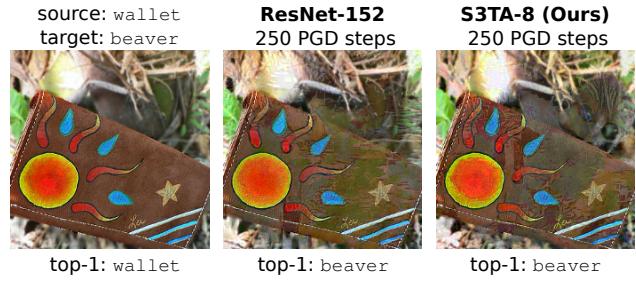


Figure 1: We augment a modern neural network with a sequential top-down attention model for image classification. The model achieves state-of-the-art adversarial robustness against PGD attacks and the resulting adversarial images are often human interpretable. On the left is a source image (label:wallet)— both an adversarially trained ResNet-152 and our model classify it correctly. In the middle and on the right are adversarial examples produced by a 250 step PGD attack against each model (target class is beaver). Both models fail to defend against the attack and predict the target class as their top-1 output. However, while the attack image for the ResNet contains no visible interpretable structure, the attack image for our model contains a salient and coherent image of a beaver’s head (best viewed zoomed in on screen).

[22], which are some of the most powerful models we use currently, have not changed this scenery significantly.

While there is no doubt that these models are very successful in solving some tasks, concerns have been raised about their robustness and reliability [36, 49]. Small, carefully chosen perturbations to the input, often imperceptible to a human observer, may cause these models to output incorrect predictions with high confidence [49]. These kind of perturbations are called adversarial examples [18, 49] and are a subject of ongoing research [4, 9, 56].

The current paradigm of neural network models has certainly been inspired by the human and primate visual system [44]. Early predecessor models have directly made this connection, and there is a line of work which connects between

the activations of such neural network models and the neural activity in brains [8]. These parallels between models and biological vision systems apply mostly to early vision processing [14] – and specifically, the feed-forward processing which happens in time-limited scenarios [14]. This has been discussed in several intriguing works, including in the adversarial examples context.

There are, however, some major differences between feed-forward neural network and the primate visual system. The eye in primates has a fovea which samples different regions of the visual input field at different spatial resolutions [16]. Furthermore (and possibly tightly connected to the fovea) the system has a strong attentional bottleneck which has been researched in many different works [45, 7]. The visual cortex has many feedback and top-down *recurrent* connections [41] and it is not purely feed-forward. Additionally, humans don’t view images as a static scene, but explore the images in a series of saccades/fixations, collecting and integrating information in the process [34]. This has been postulated to cause humans to report different classification mistakes which are qualitatively different than those of deep neural networks [13].

In this work we propose to use a soft, sequential, spatial, top-down **attention** mechanism (which we abbreviate as S3TA) [39], drawing inspiration from the primate visual system. While we do not presume this to be a *biologically-plausible* model in any way, we do propose that this model captures some of the *functionality* of the visual cortex, namely the attentional bottleneck and sequential, top-down control. We adversarially train the model on ImageNet images, showing that it has state-of-the-art robustness against adversarial attacks (focusing on Projected Gradient Descent or PGD [32, 36] attacks). We show that by increasing the number of steps we unroll the model, we are able to better defend against stronger attacks – resulting in a “computational race” between the attacker and the defender. Finally, but importantly, we show that the resulting adversarial examples often (though not always) include global, salient structures which would be perceptible and interpretable by humans (Figure 1). Furthermore, we show that the attack often tries to attract the attention of the model to different parts of the image instead of perturbing the main object in the source image directly.

2. Related Work

Adversarial training: Adversarial training aims to create models that are robust to adversarial attacks. At their core, techniques such as the ones of [18] and [36] find the worst case adversarial examples at each training step (using Fast Gradient Sign Method or PGD attacks) and add them to the training data. Models created by [36] have been shown to be empirically robust on MNIST and CIFAR-10. [28] proposed using Adversarial Logit Pairing (ALP) to encourage the logit

predictions of a network for a clean image and its adversarial counterpart to be similar. However, ALP performs poorly under stronger attacks [15]. [56] proposed feature denoising networks together with adversarial training to achieve strong performance on ImageNet [12]. Other methods like [20] achieve gradient obfuscation more explicitly by adding non-differentiable preprocessing steps. Although these gradient masking techniques make gradient-based attacks fail, more sophisticated adversaries, such as gradient-free methods [51, 2], can circumvent these defenses.

Recurrent attention models: Attention mechanisms have been widely used in many sequence modeling problems such as question-answering [24], machine translation [6, 52], video classification and captioning [46, 33], image classification and captioning [37, 11, 17, 1, 55, 60, 53, 5, 57], text classification [58, 47], generative models [42, 59, 30], object tracking [29], and reinforcement learning [10]. We build out model based on the one introduced in [39] and adapt and modify it for ImageNet scale image classification. The model uses a soft key, query, and value type of attention similar to [52, 42]. However, instead of using *self*-attention, where the queries come from the input directly, this model uses a top-down source for them generated by a LSTM (see Section 3 for details). Furthermore, the output of the attention model is highly compressed and has no spatial structure other than the one preserved using a spatial basis. This is unlike self-attention where each pixel attends to every other pixel and hence the spatial structure is preserved. Finally, it applies attention sequentially in time similar to [57], but with a largely different attention mechanism. See Section 3 for full model details.

Adversarial robustness with attention: There has been some work studying the use of attention to strengthen classifiers against adversarial attacks. [35] uses foveation-inspired, manual image cropping to reduce the impact of an adversarial perturbation on ImageNet classification accuracy. [54] tries to regularize the activations of a classifier by applying a hard mask generated from an adversarial-perturbed form of the image. The hope is that the use of a mask to occlude important parts of the image builds robustness to perturbations. Recently it has also been shown that “squeeze and excite” [27] self attention can help in classifying “Natural” adversarial examples [23]. We evaluate our model on this dataset in Section 5.4.

3. Model

We base our model on the one proposed by [39] for reinforcement learning and we adapt it to ImageNet scale image classification. The model sequentially queries the input, actively attending relevant pieces of spatial information at each time step to refine its estimate of the correct label. The two key components are the *sequential* nature of the model and the *top-down, attentional* bottleneck, both of which we

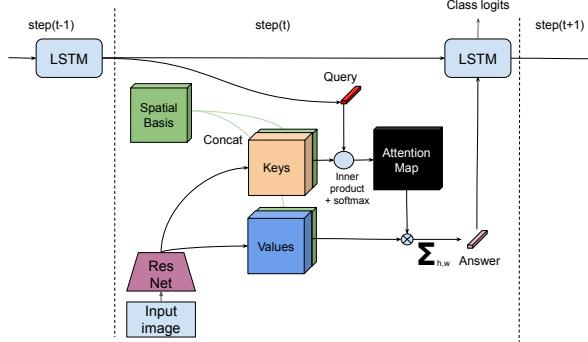


Figure 2: A general view of the sequential top-down attention model. The input image is passed through a ResNet to produce a keys and a values tensor. We concatenate a fixed, predefined spatial basis to both. A query vector is decoded from an LSTM state and an inner product between it and the keys vector is calculated at each spatial location. A softmax is then applied to produce an attention map. This attention map is point-wise multiplied with the values tensor, and the result is summed spatially to produce an answer vector. This answer vector is the input to the LSTM at this time step. The LSTM output is then decoded into class logits to produce the output of the classifier. More than one query vector can be produced at each time step (resulting in a matching number of answer vectors).

empirically show contribute to its resilience to adversarial attacks.

We briefly highlight the important components of the model, illustrated in Figure 2. For full details, we refer the reader to [39] and the supplementary material. The model starts by passing the input image through a “vision” net — a convolutional neural net (here we use a modified ResNet-152, see below). We use the same input image for all time steps, so the output of the ResNet needs to be calculated only once. The resulting output tensor is then split along the channel dimension to produce a *keys* tensor and a *values* tensor. To both of these tensors, we concatenate a fixed *spatial basis* tensor which encodes spatial locations using a Fourier representation. This spatial basis is important because our attentional bottleneck sums over space causing the spatial structure of these tensors to disappear and this basis allows passing on spatial location information.

We unroll the top-down controller for several compute steps, attending the input at each step and processing the answer through the controller to produce the output (or next state). The Top-Down controller is an LSTM core [26] whose previous state is decoded through an “query network”, an MLP, into one or more *query* vectors. Each query vector has the same number of channels as the keys tensor plus the number of channels in the spatial basis. We take the inner product between the query vector and key and spatial

basis tensor at each spatial location, resulting in a single channel map of attention logits. We pass these through a spatial softmax to produce the *attention map* for this query. The resulting attention map is then point-wise multiplied with the values tensor (and the spatial basis). Note that a single map is used for *all* channels, we note the importance of this below. The multiplied value tensor is summed across the *spatial* dimensions to produce an *answer* vector, one for each query. These answers are fed into the LSTM as the input for the current time step (concatenating them if more than one is used). Finally, the output from the *last* LSTM output is decoded into class logits through an MLP. The cross-entropy loss w.r.t to the ground truth class is calculated with this output. The initial state of the LSTM is also learned. Since the model is fully differentiable, we train it end-to-end, including the ResNet, with adversarial training (as described in Section 4.1) and 5.

Several important points regarding our version of the model in this context:

- The attention bottleneck makes the decision of the model depend on potentially large extents of the image. This can be due to the shape of attention map at every time step, as well as the fact that these maps can change considerably between time steps. This should cause a *local* adversarial perturbation [38] to be less effective. We discuss this in Section 6 and show that indeed, we often observe that global perturbations are required by the attacker for the attack to succeed.
- Following the last point, the fact that the attention map has a single channel which modulates all value channels *together* constraints the content of these channels to be spatially coherent. In a regular ResNet architecture the last block output is read with a average pooling done independently on each channel - this allows the network to lose spatial structure by the time information reaches this last layer.
- In order to make the spatial element, and hence the effect of the attention bottleneck more pronounced we modify the ResNet architecture to make the final output have larger spatial dimensions. This is done by changing the strides to 1 in all but the second residual block. For ImageNet input (224×224 pixels) the resulting map is 28×28 pixels large (as opposed to 7×7 in a regular ResNet).
- The top-down nature of the attention mechanism is such that the queries come from the state of the LSTM and not from the input. Hence, the model can actively select relevant information depending on its internal state, rather than just the input. This allows the model to take its own uncertainty, for example, into account, when querying the image and producing the output.

- The sequential nature of the model allows for increasing computational capacity without changing the number of parameters. We demonstrate that this helps with robustness in Section 5.

4. Adversarial Risk

We define adversarial risk in the context of supervised learning formally in this section. Given a model m_θ with parameters θ , we want to minimize the loss ℓ on inputs x and labels y sampled from the data distribution D . Formally, the objective is to minimize the *expected* risk: $\mathbb{E}_{(x,y) \sim D} \ell(m_\theta(x), y)$. Empirically, we optimize the empirical risk on a finite training set and estimate the expected risk over a held-out test set using the average loss.

As pointed out in [51], models with low expected risk may still perform poorly on any data points. In situations where a single catastrophic failure is not allowable, the empirical risk estimate may be problematic. Hence, we also need to consider the *worst-case* risk for the desired robust models: $\sup_{(x,y) \in \text{supp } D} \ell(m_\theta(x), y)$, where $\text{supp } D$ denotes the support of D . In practice, computing the supremum over the input space is intractable as the search space is exponentially large in the dimension of x . We can instead use the local *adversarial risk*, as a proxy for the worst-case risk:

$$\mathbb{E}_{(x,y) \sim D} \left[\sup_{x' \in N_\epsilon(x)} \ell(m_\theta(x'), y) \right], \quad (1)$$

where the neighborhood $N_\epsilon(x)$ denotes a set of points in $\text{supp}(D)$ within a fixed distance $\epsilon > 0$ of x , measured by a given metric. The adversarial risk enables us to approximate the worst-case risk in a tractable way. For example, we can use off-the-shelf optimization algorithms (such as PGD [32, 36]) to find the supremum over the neighborhood $N_\epsilon(x)$.

In this paper, we consider the specification that the image predictions should remain the same within an ℓ_∞ -ball of an image x , where an allowable maximum perturbation is $\epsilon = 16/255$, relative to the pixel intensity scaled between 0 and 1.

Specifically, we focus on the ImageNet dataset [12] and we primarily consider the targeted PGD attack as the threat model, where the targeted class is selected uniformly at random, following [2, 28, 56], given that the untargeted attacks can result in less meaningful comparisons (e.g., misclassification of very similar dog breeds) on ImageNet.

4.1. Adversarial Training

To train models that are robust to adversarial attacks, we follow the adversarial training approach by [36] and more recently [56].

Following the adversarial risk in Eq. (1), we want to

minimize the following saddle point problem:

$$\min_{\theta} \rho(\theta), \text{ where } \rho(\theta) = \mathbb{E}_{(x,y) \sim D} \left[\max_{x' \in N_\epsilon(x)} \ell(m_\theta(x'), y) \right]. \quad (2)$$

where the inner maximization problem is to find an adversarial perturbation of x that can maximize the loss; the outer minimization problem aims to update model parameters such that the adversarial risk $\rho(\theta)$ is minimized.

In our experiments, we approximate the solution to the inner maximization problem with PGD. Specifically, we perform PGD on the cross entropy loss described using iterative signed gradients as in [32, 56]. During training, we use targeted PGD attacks, where the targeted class is selected at random uniformly, following [28, 56].

4.2. Adversarial Evaluation

In this paper, we use the PGD attack to evaluate the model, which is regarded as a strong attack¹ in the community and several published papers use this as their benchmark.

In cases where we cannot take analytic gradients, or where they are not useful, we can approximate the gradients using gradient-free optimization. The use of gradient-free methods lets us verify whether robustness stems from gradient obfuscation by the model architecture. In this work, we use the SPSA algorithm [48], which is well-suited for high-dimensional optimization problems, even in the case of noisy objectives. We use the SPSA formulation in [51] to generate adversarial attacks. In the SPSA algorithm, it first samples a batch of n samples from a Rademacher distribution (i.e., Bernoulli ± 1), namely, $v_1, \dots, v_n \in \{1, -1\}^D$. Then, the SPSA algorithm approximates the gradient with finite difference estimates in random directions. Specifically, for the i -th sample, the estimated gradient g_i is calculated as follows:

$$g_i = \frac{f(x_t + \delta v_i) - f(x_t - \delta v_i)}{2\delta v_i}$$

where δ is the perturbation size, x_t is the perturbed image at the t -th iteration, and f is the model to be evaluated. Finally, SPSA aggregates the estimated gradient and performs projected gradient descent on the input x_t . The whole process iterates for a predefined number of iterations.

5. Experiments

In this section, we empirically study the robustness of S3TA on the ImageNet dataset [12]. For convenience, we denote S3TA- k to be a S3TA model that is unrolled for k time steps and evaluate S3TA-2, S3TA-4, S3TA-8, and S3TA-16.

¹Note that PGD is not necessarily the most suited attack for sequential models, but for lack of better alternative we use it with large number of steps.

We follow the training procedure used in [56], including learning rate schedule, label smoothing, attack type during training and evaluation procedure. We find that training with a somewhat lower learning rate (0.2 initial learning rate) and a smaller batch size (1024) is more stable for our model. Training S3TA-16 is more challenging than the other models due to the length of the unroll. In order to train it we start by reading off the output from the 4th step for the first 35 epochs, 8th step for the next 35 epochs and 16th step for the rest of training. All models are trained for 120 epochs. We train the model on 128 Google Cloud TPU v3 cores. Training takes between 42 and 70 hours, depending on the number of unroll steps. We use a ResNet-152 as the vision-net of the model (see Section 3) setting all strides to 1 other than the second residual block. This results in larger spatial support for the ResNet output (28×28 pixels). The recurrent core is an LSTM with 1024 hidden units, the query network and the output MLP are both with a single hidden layer of 1024 units. All the activation units used are ReLUs. The attention model uses 4 attention query heads in all experiments here.

5.1. Random targeted attacks

The first set of models was adversarially trained with 10 PGD steps. These are generally weaker models than models trained with 30 PGD steps (see below) but they take less time and resources to train. Figure 3 shows the top-1 accuracy of these models for the ImageNet test dataset under a wide range of random targeted PGD attack strengths compared to a ResNet-152 baseline (also trained with 10 PGD steps during adversarial training). With only 2 steps of attention the weakest model here, S3TA-2, only has a chance to send two queries, one before it even sees the image, and one after processing the answers from the first step. This puts emphasis on the *attention bottleneck* itself rather than the sequential nature of the model. As can be seen, the bottleneck itself already allows to model to improve significantly upon the ResNet-152 baseline.

By increasing the number of attention steps we can improve adversarial accuracy even further: unrolling for 16 steps (S3TA-16) significantly improves robustness - a S3TA-16 model is more robust against a 1000 PGD attack steps than a ResNet-152 model is against a 100 attack steps. In fact, a S3TA-16 model trained with 10 PGD steps during adversarial training is more robust than a ResNet-152 *trained with 30 PGD steps* (see Figure 4). This shows that there is a kind of “computational race” here between the strength of the attack and the number of compute steps we allow the model to have. More computation steps for the model mean better defense against stronger attacks. Going beyond a 1000 attack steps does not change the picture as most models saturate close to their 1000 step performance. Full results, including attack success rates, and nominal accuracies, can be found in Table 1 and the supplementary material.

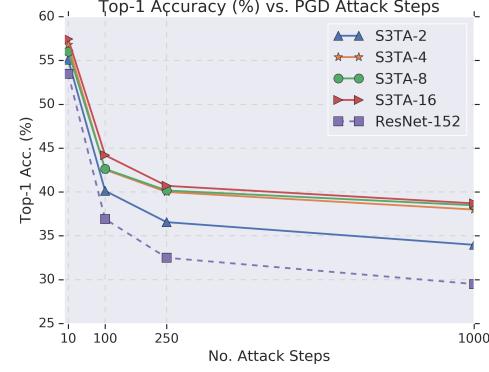


Figure 3: S3TA-2, 4, 8 and 16 vs. ResNet-152 top 1 accuracy on the ImageNet test set.. All models were adversarially trained with 10 PGD steps. Note how the introduction of the attention model significantly improve performance even with 2 attention steps, and that adding more steps (S3TA-16) improves performance further: a S3TA-16 model is more robust at a 1000 attack steps than a ResNet-152 model at a 100 attack steps.

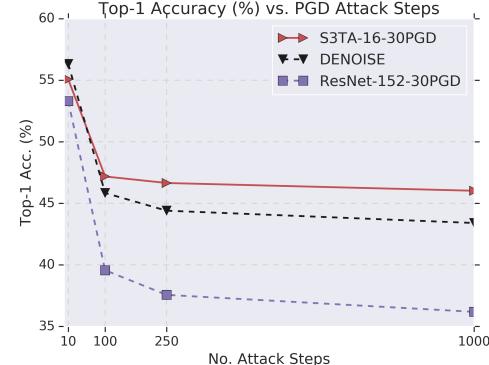


Figure 4: A S3TA-16 model adversarially trained with 30 PGD steps vs. ResNet-152 (30 steps) and DENOISE [56] top 1 accuracy on the ImageNet test set. DENOISE is the current state-of-the-art on ImageNet and as can be seen S3TA-16 performs significantly better than both models, setting a new state-of-the-art.

We now turn to compare models adversarially trained with 30 PGD steps. These models are much stronger generally and achieve good robustness results across a wide range of attack strengths, but require a great deal of resources and time to train. Figure 4 shows the top-1 accuracy of a S3TA-16-30 model (“-30” denotes 30 PGD steps during training) vs. a ResNet-152 model and DENOISE [56], the latter being the current state-of-the-art in adversarial robustness. As can be seen, S3TA-16 comfortably outperform both models, setting a new state-of-the-art for random targeted attacks.

Figure 5 shows the attack success rates for all models discussed so far. When evaluating defense strategies, measuring

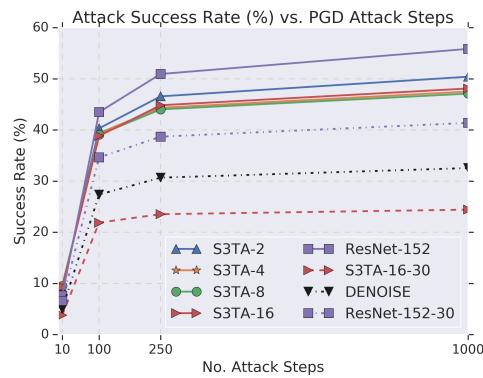


Figure 5: Attack success rates for all models presented (lower is better). The main effects observed for top 1 accuracy hold here: more attention steps lower the attack success rate and more PGD steps during training help reduce it even further. S3TA-16-30 clearly has the lowest attack success rates, about 25% lower than DENOISE while nominal accuracy is similar (see 1)

attack success rates makes sense when nominal accuracies of models are high and comparable. For all models presented here this is true (see Table 1). Note that the results hold for this measure as well - more attention steps help reduce attack success rates, and more PGD steps during training help. The success rate of attacks against S3TA-16-30 is about 25% lower than that of DENOISE (lower is better).

5.2. Untargeted and gradient-free attacks

Most robustness measures in the literature are for targeted, gradient based attacks. However, a model that is only robust against targeted attacks is weaker than one robust against untargeted attacks [15]. In Table 2, we report results for untargeted attacks using 200 PGD steps for S3TA-16-30 vs. ResNet-152, DENOISE and LLR [43]. Our model is very competitive in this setting, both for $\epsilon = 4/255$ and $\epsilon = 16/255$.

We also explore gradient-free methods to make sure the model does not obfuscate gradients [51, 3]. Specifically, we use random targeted SPSA [51] with a batch size of 4096 and 100 iterations under $\epsilon = 16/255$ for the gradient-free attack. We use iterative signed gradients [32, 56] with gradients estimated by SPSA. Results on a subset of 1000 randomly-chosen images can be seen in Table 3. We can observe that SPSA does not lower accuracy compared to gradient-based attacks. This provides an additional evidence that the model’s strong performance is not due to gradient masking. Given SPSA’s adversarial accuracy is weaker (that is, all models defend better than with gradient-based methods) the performance difference between the models is not very informative.

5.3. Loss landscapes

Another way of making sure gradients are not obfuscated is by visualizing the loss landscapes [43, 50]. Figure 6 shows the top-view of the loss landscape for S3TA-4 and S3TA-16. To visualize the loss landscapes, we change the input along a linear space defined by the worse perturbations found by PGD and a random direction. The u and v axes represent the magnitude of the perturbation added in each of these directions respectively and the z axis represents the loss. For both panels, the diamond-shape stands for the projected L_∞ ball of size $\epsilon = 16/255$ around the nominal image. We can observe that both loss landscapes are rather smooth, which provides an additional evidence that the strong performance is not because of gradient obfuscation.

5.4. Natural adversarial examples

A recent interesting dataset is “Natural adversarial examples” [23]. This curated dataset is composed of *natural* images of a subset of 200 classes from ImageNet. These images are chosen such that they cause modern image classifier to misclassify an image with high confidence, even though no actual modification to the image is done. The images often contain objects in unusual locations, photographed from unusual angles or occluded or corrupted in a variety of ways. We compare a S3TA-16 model to DENOISE, the ResNet baseline and the “Squeeze and excite” [27] (ResNet+SE) variant reported in the original paper. Figure 7 shows the results, using the measures used in the paper: Top-1 accuracy, Calibration error which measures the difference between the confidence of each model and its actual error rate, and AURRA which allows calculation of accuracy while giving classifiers an opportunity to abstain if they are not confident in their prediction.

6. Analysis

We have shown that the sequential attention model improves robustness against a variety of attacks and attack strengths. Furthermore, we have seen that we can increase accuracy and defend better against stronger attacks by unrolling the model for more time steps. We now turn to

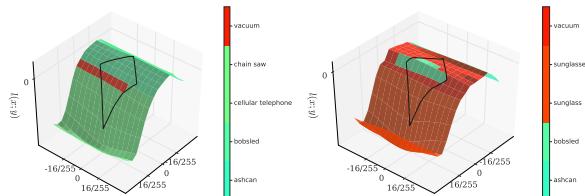


Figure 6: Loss landscapes for S3TA-4 (left) and S3TA-16 (right). The surfaces are approximately linear meaning there is no significant gradient obfuscation.

Model	Nominal Accuracy	10 steps		100 steps		250 steps		1000 steps	
		Top-1	Attack Success						
ResNet-152	70.66%	53.48%	7.63%	36.91%	43.48%	32.50%	50.93%	29.5%	55.84%
S3TA-2	72.30%	55.08%	9.06%	40.13%	40.31%	36.56%	46.56%	33.97%	50.40%
S3TA-4	72.48%	56.78%	9.10%	42.54%	39.37%	40.00%	44.33%	37.99%	47.50%
S3TA-8	72.14%	56.02%	9.50%	42.63%	39.06%	40.17%	44.01%	38.48%	47.09%
S3TA-16	72.54%	57.45%	9.33%	44.19%	38.83%	40.71%	44.82%	38.70%	48.16%
ResNet-152-30	63.62%	53.30%	6.56%	39.56%	34.62%	37.56%	38.68%	36.18%	41.37%
DENOISE	66.02%	56.33%	5.00%	45.84%	27.36%	44.19%	30.66%	43.39%	32.54%
S3TA-16-30	64.55%	55.08%	3.79%	47.18%	21.87%	46.65%	23.52%	46.11%	24.91%

Table 1: Full results for all models on random targeted PGD attacks with the ImageNet test set, with different number of attack steps. Bottom three rows are models trained with 30 PGD steps, the rest were trained with 10 PGD steps.

Model	Top-1 $\epsilon = 4/255$	Top-1 $\epsilon = 16/255$
ResNet-152	39.7%	6.3%
DENOISE [56]	38.9%	7.5%
LLR [43]	47.0%	6.1%
S3TA-16	46.75%	9.8%

Table 2: Top-1 accuracy under untargeted attacks at 200 PGD steps. As can be seen, our model is very competitive with existing methods though not optimized for this particular attack method.

Model	Top-1	Attack Success
ResNet-152	61.90%	2.20%
DENOISE [56]	63.70%	1.90%
S3TA-16	59.60%	1.90%

Table 3: Top-1 accuracy under random targeted SPSA attacks (batch size of 4096 and 100 iterations). SPSA is a gradient free method which provides evidence whether gradients are obfuscated. As can be seen all models perform similarly, considering they all defend better here than the corresponding gradient based attack (making the actual reported number less informative).

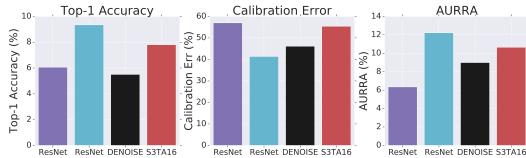


Figure 7: Results of our model on the “Natural Adversarial Examples” dataset. Our model achieves better top-1 accuracy, and better AURRA (see text for details) than both a ResNet-152 and DENOISE. Squeeze and excite outperforms our model in all measures.

analyze some of the properties of the resulting attack images and strategies.

Figure 8 shows several examples of generated adversarial

examples for different attack strengths for an adversarially trained S3TA model (with 4 unrolling steps) and an adversarially trained ResNet-152. We observe that often (but certainly not always, see below) the generated image contains salient structures related to the target class. However, while the nature of these perturbations is local at best for the ResNet examples, for S3TA *global, coherent and human interpretable* structures appear. This sheds some light to the internal reasoning process of our model, hinting that it reasons globally, across space, in a coherent manner. It’s important to note that in many cases the adversarial examples do not appear to contain any salient structure (even with many attack steps). They do appear much more often midway through training the model, while the model is already a very good classifier but still not at the peak of its robustness. Towards the end of training it seems that it is harder to generate these, possibly as part of the defense strategy the model learns. See the supplementary material for some examples generated midway through training, as well as more visible and invisible perturbation images. Understanding under what circumstances these examples appear is left for future research.

6.1. Distracting the Attention

Since attention is an integral part in our model, we can see whether it plays a role when the network is attacked and mislabels an image. We can visualize the attention maps generated at each time step and see how the attention is used under different attack scenarios. Figure 9 shows such attention maps for an image used to attack a S3TA-16 model. Attention is superimposed over original image - highlighted areas are more attended than dark areas. As can be seen, the attack can create stimuli which attract the some the attention heads away from the main object in the image, in this case towards something that slightly resembles the target class in the background.



Figure 8: Adversarial images generated by PGD to attack the ResNet-152 and S3TA-4 model. The source label is depicted on the left, the target label on the right. Source images are the leftmost column. We generate examples for 10, 100, 1000 and 10,000 PGD steps. Note how for the examples shown here, the perturbations are quite visible for both models (not surprising in light of the strength of the attack). However, the perturbations generated to attack the ResNet model are mostly local, comprising at best of disconnected visible features related to the target class. On the other hand, the examples generated to attack S3TA contain global, coherent and human interpretable structure. Note the 3D structure and spatial extent of the locomotive (top-left), the coherency of the king snake on the ground (bottom-left), the camel head on the bark of the tree (top-right) and the beagle and the man appearing (bottom-right). These structures appear mostly when there are already existing features in the image which can be used to form the target class, and the model uses them in a global, coherent manner. Images are best viewed on screen and zoomed in.

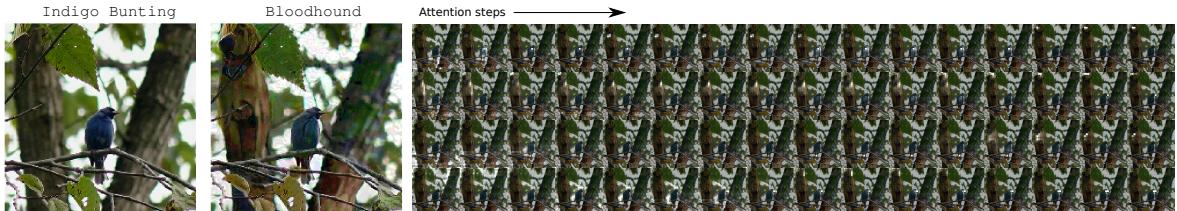


Figure 9: Attacks attract attention away from the main object. Here we see the source image (left) attack image (target class bloodhound) and the 4 attention heads of the model as they unroll through 16 steps. Some of the heads still attend the main object from the source image, however some of the heads are attracted away towards a group of branches in the background. Upon close inspection it does seem that these branches resemble a bloodhound dog in the attack image. Though this structure is not very salient to humans, it is enough to attract the attention away and cause the model to mislabel the image. Best viewed on screen and zoomed in.

7. Conclusion

In this paper, we have shown that a recurrent attention model inspired by the primate visual system is able to achieve state-of-the-art robustness against random target adversarial attacks. Allowing for more attention steps improves accuracy under stronger attacks. We show that the resulting adversarial examples often (but not always) contain global structures which are visible and interpretable to a human observer.

Why is it that global structures arise when attacking a

model like this? We postulate that there are two contributing factors. The attention mechanism pools data from large parts of the image, which means that the gradients propagate quickly across the whole of the image, and not just locally. Furthermore, because the the model is unrolled for several steps, more parts of the image may be potentially attended to and thus gradients may propagate there. We see evidence for this in the fact the often the attacker attracts the attention away from the main object in the image, hinting that the attention plays a crucial role in the attack strategy.

There is still a lot of work to be done to achieve adversarial robustness in complex datasets. Even models like the proposed one often fail when the attacker is strong enough, and performance is still quite low compared to nominal accuracies, but at some point we may ask — if an image has been perturbed enough such that it does not resemble the original image and looks like another image coming from the target class, is it still a valid adversarial perturbation? Models like the one presented here may allow us to reach that frontier in the future.

References

- [1] Artsiom Ablavatski, Shijian Lu, and Jianfei Cai. Enriched deep recurrent visual attention model for multiple object recognition. In *Applications of Computer Vision (WACV), 2017 IEEE Winter Conference on*, pages 971–978. IEEE, 2017. [2](#)
- [2] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. *arXiv preprint arXiv:1802.00420*, 2018. [2, 4](#)
- [3] Anish Athalye, Nicholas Carlini, and David Wagner. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples. 2018. [6](#)
- [4] Anish Athalye and Ilya Sutskever. Synthesizing robust adversarial examples. *arXiv preprint arXiv:1707.07397*, 2017. [1](#)
- [5] Jimmy Ba, Volodymyr Mnih, and Koray Kavukcuoglu. Multiple object recognition with visual attention. *arXiv preprint arXiv:1412.7755*, 2014. [2](#)
- [6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014. [2](#)
- [7] Daniel Baldauf and Robert Desimone. Neural mechanisms of object-based attention. *Science*, 344(6182):424–427, 2014. [2](#)
- [8] Charles F. Cadieu, Ha Hong, Daniel L. K. Yamins, Nicolas Pinto, Diego Ardila, Ethan A. Solomon, Najib J. Majaj, and James J. DiCarlo. Deep neural networks rival the representation of primate it cortex for core visual object recognition. *PLoS Computational Biology*, 10(12):e1003963, Dec 2014. [2](#)
- [9] Nicholas Carlini and David Wagner. Towards evaluating the robustness of neural networks. In *Security and Privacy (SP), 2017 IEEE Symposium on*, pages 39–57. IEEE, 2017. [1](#)
- [10] Jinyoung Choi, Beom-Jin Lee, and Byoung-Tak Zhang. Multi-focus attention network for efficient deep reinforcement learning. *arXiv preprint arXiv:1712.04603*, 2017. [2](#)
- [11] Minki Chung and Sungjoon Cho. Cram: Clued recurrent attention model. *arXiv preprint arXiv:1804.10844*, 2018. [2](#)
- [12] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. ImageNet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009. [2, 4](#)
- [13] Miguel P Eckstein, Kathryn Koehler, Lauren E Welbourne, and Emre Akbas. Humans, but not deep neural networks, often miss giant targets in scenes. *Current Biology*, 27(18):2827–2832, 2017. [2](#)
- [14] Gamaleldin Elsayed, Shreya Shankar, Brian Cheung, Nicolas Papernot, Alexey Kurakin, Ian Goodfellow, and Jascha Sohl-Dickstein. Adversarial examples that fool both computer vision and time-limited humans. In *Advances in Neural Information Processing Systems*, pages 3914–3924, 2018. [2](#)
- [15] Logan Engstrom, Andrew Ilyas, and Anish Athalye. Evaluating and understanding the robustness of adversarial logit pairing. *arXiv preprint arXiv:1807.10272*, 2018. [2, 6](#)
- [16] David C. Van Essen and Charles H. Anderson. Information processing strategies and pathways in the primate visual system . 1995. [2](#)
- [17] Jianlong Fu, Heliang Zheng, and Tao Mei. Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition. In *CVPR*, volume 2, page 3, 2017. [2](#)
- [18] Ian J Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv:1412.6572*, 2014. [1, 2](#)
- [19] Sven Gowal, Jonathan Uesato, Chongli Qin, Po-Sen Huang, Timothy Mann, and Pushmeet Kohli. An alternative surrogate loss for pgd-based adversarial testing, 2019. [12](#)
- [20] Chuan Guo, Mayank Rana, Moustapha Cisse, and Laurens van der Maaten. Countering adversarial images using input transformations. *arXiv preprint arXiv:1711.00117*, 2017. [2](#)
- [21] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016. [1](#)
- [22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. 2016. [1](#)
- [23] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. *arXiv preprint arXiv:1907.07174*, 2019. [2, 6](#)
- [24] Karl Moritz Hermann, Tomas Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1693–1701, 2015. [2](#)
- [25] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdelrahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. [1](#)
- [26] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural Comput.*, 9(8):1735–1780, Nov. 1997. [3](#)
- [27] Jie Hu, Li Shen, and Gang Sun. Squeeze-and-excitation networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7132–7141, 2018. [2, 6](#)
- [28] Harini Kannan, Alexey Kurakin, and Ian Goodfellow. Adversarial logit pairing. *arXiv preprint arXiv:1803.06373*, 2018. [2, 4](#)
- [29] Adam Kosiorek, Alex Bewley, and Ingmar Posner. Hierarchical attentive recurrent tracking. In *Advances in Neural Information Processing Systems*, pages 3053–3061, 2017. [2](#)
- [30] Adam R Kosiorek, Hyunjik Kim, Ingmar Posner, and Yee Whye Teh. Sequential attend, infer, repeat: Gen-

- erative modelling of moving objects. *arXiv preprint arXiv:1806.01794*, 2018. 2
- [31] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in Neural Information Processing Systems 25*, pages 1097–1105. 2012. 1
- [32] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. Adversarial machine learning at scale. *arXiv preprint arXiv:1611.01236*, 2016. 2, 4, 6
- [33] Xuelong Li, Bin Zhao, and Xiaoqiang Lu. MAM-RNN: multi-level attention model based RNN for video captioning. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, 2017. 2
- [34] Simon P Liversedge and John M Findlay. Saccadic eye movements and cognition. *Trends in Cognitive Sciences*, 4(1):6–14, 2000. 2
- [35] Yan Luo, Xavier Boix, Gemma Roig, Tomaso A. Poggio, and Qi Zhao. Foveation-based mechanisms alleviate adversarial examples. *CoRR*, abs/1511.06292, 2015. 2
- [36] 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. 1, 2, 4
- [37] Volodymyr Mnih, Nicolas Heess, Alex Graves, et al. Recurrent models of visual attention. In *Advances in Neural Information Processing Systems*, pages 2204–2212, 2014. 2
- [38] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, and Pascal Frossard. Universal adversarial perturbations. *CoRR*, abs/1610.08401, 2016. 3
- [39] Alex Mott, Daniel Zoran, Mike Chrzanowski, Daan Wierstra, and Danilo J Rezende. Towards interpretable reinforcement learning using attention augmented agents. *arXiv preprint arXiv:1906.02500*, 2019. 2, 3, 11
- [40] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *ICML*, pages 807–814, 2010. 1
- [41] Bruno A Olshausen. 20 years of learning about vision: Questions answered, questions unanswered, and questions not yet asked. In *20 Years of Computational Neuroscience*, pages 243–270. Springer, 2013. 2
- [42] Niki Parmar, Ashish Vaswani, Jakob Uszkoreit, Łukasz Kaiser, Noam Shazeer, and Alexander Ku. Image transformer. *arXiv preprint arXiv:1802.05751*, 2018. 2
- [43] Chongli Qin, James Martens, Sven Gowal, Dilip Krishnan, Alhussein Fawzi, Soham De, Robert Stanforth, Pushmeet Kohli, et al. Adversarial robustness through local linearization. *arXiv preprint arXiv:1907.02610*, 2019. 6, 7
- [44] Maximilian Riesenhuber and Tomaso Poggio. Hierarchical models of object recognition in cortex. *Nature Neuroscience*, 2(11):10191025, Nov 1999. 1
- [45] Pieter R Roelfsema, Victor AF Lamme, and Henk Spekreijse. Object-based attention in the primary visual cortex of the macaque monkey. *Nature*, 395(6700):376, 1998. 2
- [46] Mo Shan and Nikolay Atanasov. A spatiotemporal model with visual attention for video classification. *arXiv preprint arXiv:1707.02069*, 2017. 2
- [47] Sheng-syun Shen and Hung-yi Lee. Neural attention models for sequence classification: Analysis and application to key term extraction and dialogue act detection. *arXiv preprint arXiv:1604.00077*, 2016. 2
- [48] James C Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992. 4
- [49] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv:1312.6199*, 2013. 1
- [50] Jonathan Uesato, Jean-Baptiste Alayrac, Po-Sen Huang, Robert Stanforth, Alhussein Fawzi, and Pushmeet Kohli. Are labels required for improving adversarial robustness? In *Advances in Neural Information Processing Systems*, 2019. 6
- [51] Jonathan Uesato, Brendan O’Donoghue, Aaron van den Oord, and Pushmeet Kohli. Adversarial risk and the dangers of evaluating against weak attacks. In *ICML*, 2018. 2, 4, 6
- [52] Ashish Vaswani, Noam Shazeer, Niki Parmar, 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. 2
- [53] Fei Wang, Mengqing Jiang, Chen Qian, Shuo Yang, Cheng Li, Honggang Zhang, Xiaogang Wang, and Xiaou Tang. Residual attention network for image classification. *arXiv preprint arXiv:1704.06904*, 2017. 2
- [54] Shangxi Wu, Jitao Sang, Kaiyuan Xu, Jiaming Zhang, Yan-feng Sun, Liping Jing, and Jian Yu. Attention, please! adversarial defense via attention rectification and preservation. *arXiv preprint arXiv:1811.09831*, 2018. 2
- [55] Tianjun Xiao, Yichong Xu, Kuiyuan Yang, Jiaxing Zhang, Yuxin Peng, and Zheng Zhang. The application of two-level attention models in deep convolutional neural network for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 842–850, 2015. 2
- [56] Cihang Xie, Yuxin Wu, Laurens van der Maaten, Alan L. Yuille, and Kaiming He. Feature denoising for improving adversarial robustness. *CoRR*, abs/1812.03411, 2018. 1, 2, 4, 5, 6, 7, 11, 12
- [57] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron C. Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. In *ICML*, 2015. 2
- [58] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016. 2
- [59] Han Zhang, Ian J. Goodfellow, Dimitris N. Metaxas, and Augustus Odena. Self-attention generative adversarial networks. *CoRR*, abs/1805.08318, 2018. 2
- [60] Heliang Zheng, Jianlong Fu, Tao Mei, and Jiebo Luo. Learning multi-attention convolutional neural network for fine-grained image recognition. In *Int. Conf. on Computer Vision*, volume 6, 2017. 2

A. Model details

A.1. Detailed model overview

Here we detail and expand on the specifics of the model described in the main paper. For a more thorough overview and discussion, we refer the reader to [39].

Figure 10 depicts the model in full detail. On the left is the unrolled compute graph for a 4 step model. On the right is a detailed overview of the attention mechanism for a single attention head.

The model is comprised of two main components - a “vision net” component which processes the input image and outputs a spatial tensor (of size 28×28 in this case) and a “controller” component, which is a recurrent core attending this tensor at every time step. On the left of Figure 10 we can see the image being processed through the vision net (ResNet-152, in this case) and the output tensor being produced. This happens only once per image since the input image is fixed for all compute steps. At the top we see the controller (in this case an LSTM) unrolled. A learned initial state is used for the first time step. At each time step the model attends the vision nets’ output tensor, sending out a query vector(s) and receiving an answer vector(s) to process. At the final step we read out the controller’s state, pass it through an MLP and produce the class logits.

The right side of Figure 10 depicts the inner workings of the attention mechanism. The output tensor from the vision net is split into two tensors along the channel dimension. The first C_k channels ($C_k = 32$ in all our experiments) are used as the “keys tensor” (of size $28 \times 28 \times C_k$). The rest of the channels are used as the “values” tensor with C_v channels ($C_v = 2016$ in all experiments, tensor size is $28 \times 28 \times C_v$). To both the keys and values tensors we concatenate a “spatial basis” tensor of size $28 \times 28 \times C_s$. This fixed tensor allows the attention to preserve and query about spatial information after the spatial structure is lost (see below).

We produce a “query” vector by passing the previous controller state (together with the last time step’s logits) through a “query network” (an MLP). This query vector is of size $1 \times 1 \times (C_k + C_s)$. We take the inner product of the query vector with each *spatial* position in the keys tensor to produce a 28×28 single channel attention logits map. We pass this logits map through a spatial softmax to produce the final attention map for this head.

We broadcast the attention map along the channel dimension and point-wise multiply it with the values tensor. The result is then reduce summed along both spatial dimensions to produce the “answer” vector of size $1 \times 1 \times (C_v + C_s)$. The importance of the spatial basis becomes clear at this point, since the summation causes all spatial structure to be lost.

We concatenate all answer vectors together with all query vectors and use these as inputs to the controller at the current

time step. We pass the controller state through an MLP to produce the class logits. If it is the last time step, we produce the final class logits and the loss is calculated from them.

A.2. Model parameters

We now give the full details of model parameters. Unless otherwise noted all non-linear activations are ReLUs. All learned parameters are initialized randomly.

A.2.1 Vision net

For the vision net we use a standard ResNet-152 V2. We change the stride to 1 in all residual blocks other than the second one and we don’t use the final linear layer. The output of this network for a 224×224 image is a $28 \times 28 \times 2048$ tensor.

A.2.2 Controller

The controller is a standard LSTM with a hidden size of 1024 units. The initial state is randomly initialized and learned with the rest of the network.

A.2.3 Query network

The query network is a standard MLP with one hidden layer of size 1024. Its input is the same size of the controller state plus the size of logits (2024 in our case). Its output size is $(C_k + C_s) \times N$ where $N=4$ is the number of heads we use. The spatial basis has 64 channels, hence the total output size is 384 (with 32 key channels).

A.2.4 Output MLP

This is a single hidden layer MLP with 1024 units in the hidden layer. Its input is the hidden state of the controller and its output is 1000 units large (i.e. number of classes).

A.2.5 Spatial Basis

The spatial basis is a fixed tensor of size $28 \times 28 \times C_s$. We use spatial sine and cosine functions to form the basis, with 4 frequencies per dimension resulting in $C_s = (2 \times 4)^2 = 64$ channels.

B. Training and evaluation details

We follow the training and evaluation protocol of [56] in most parameters. We use a gradient descent optimizer with momentum. We use a lower learning rate of $0.5e^{-1}$ scaled by a batch size of 256. For our models we use batch size of 1024 so the initial learning rate is 0.2. We warm start for the first 5 epochs and then anneal the learning rate (with a factor of $1e^{-1}$) after 35 epochs, 70 epochs and 95 epochs. We train for 120 epochs.

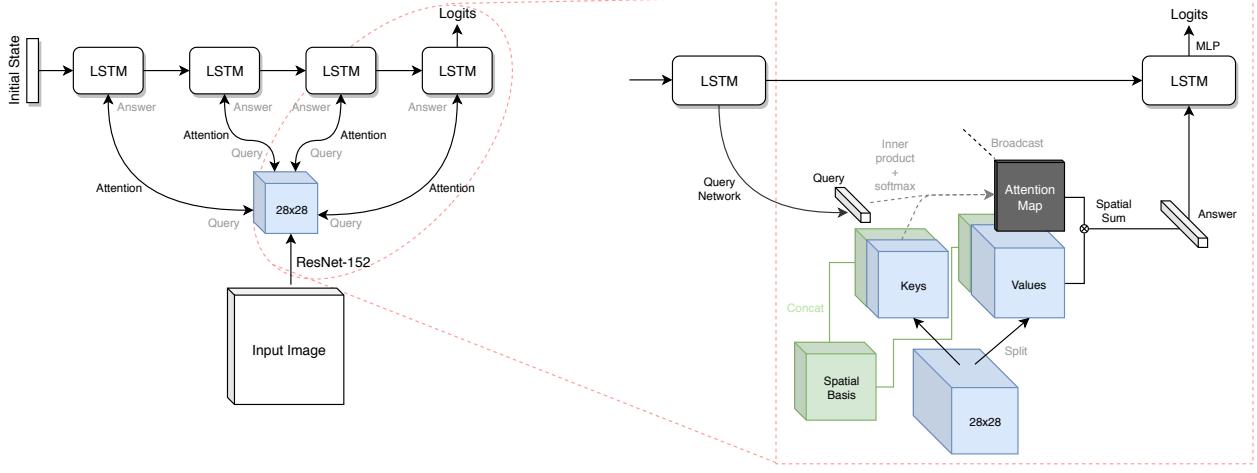


Figure 10: Full model details. Left - high level overview of the recurrent mechanism (for a 4-step model) and the corresponding vision net. Right - a zoomed in detail on one attention step. See text for full details.

The attack during training is random targeted PGD with signed gradients. We use $\epsilon = 16$, random initialization probability of 0.8 and step size of 1/255. The loss function for adversarial training is only on adversarial examples, there is no loss coming from clean images. We add L_2 weight decay loss with weight $1e - 4$, and label smoothing of 0.1. We use the same attack in evaluation, with step size of 1/255 for all number of steps other than 10 where we use 1.6/255, again following [56].

C. Additional results

In Table 4 we report results on extra strong attacks with 5000 and 10,000 steps — due to the cost of evaluation we evaluated only on 2200 random images from the ImageNet test set, but in our experience numbers change very little on larger evaluation sets.

C.1. PGD attacks with multiple restarts

Here we test the three strongest models against attacks with multiple restarts. We test with the same PGD with signed gradient attack used so far (with a 100 and 1000 steps) but we allow the attacker to start from multiple random initialization: 1, 10 or 100. We take the strongest example from all restarts to measure the top-1 and attack success rate (i.e. even if one of the adversarial attacks succeeds we count it as a success, and if the network mislabels a single image out of the attacks we do not increase the top-1 score). Results are reported in Table 5. As can be seen, all models are quite robust to multiple restarts, and S3TA-16-30 is better than the other models.

C.2. Random targeted attacks with an Adam optimizer

It has been observed [19] that random targeted attacks using the FGSM method are sometimes weaker than attacks optimized with the Adam optimizer. Here we test the performance of the model with this potentially stronger type of targeted attack. We use a 250 step attack, optimized with the Adam optimizer with learning rate annealing (0.1 until step 100, 0.01 until 200 and 0.001 for the last 50). This has been shown [19] to be quite a strong attack. Results are reported in Table 6. As can be seen all models perform worse here than with PGD with iterative FGSM, however S3TA-16-30 is significantly more robust here as well.

D. More adversarial examples

D.1. Non visible examples

Figure 11 shows adversarial examples for a S3TA-4 model where there is no visible structure to be seen, as is often the case with fully trained models.

D.2. Structured examples - mid training

Figure 12 shows adversarial examples for a S3TA-16 model around half-way through training, with clear salient global structures. These are much more common before training concludes (but still appear in fully trained models as in the main paper). Understanding the exact circumstances under which these appear is an open question.

Model	5000 steps		10,000 steps	
	Top-1 Accuracy	Attack Success Rate	Top-1 Accuracy	Attack Success Rate
ResNet-152	28.12%	58.03%	27.64%	58.79%
S3TA-2	32.85%	52.50%	32.63%	52.94%
S3TA-4	35.67%	50.57%	35.43%	50.96%
S3TA-8	36.20%	50.75%	35.44%	51.91%
S3TA-16	37.36%	49.50%	36.96%	50.23%
ResNet-152-30	36.69%	41.33%	36.78%	41.25%
DENOISE	42.76%	33.83%	42.99%	33.88%
S3TA-16-30	46.11%	24.91%	46.20%	24.68%

Table 4: Results for extra strong PGD attacks with 5000 and 10000 steps. The general picture remains as S3TA-16-30 is significantly more robust than all other models. S3TA-16 trained with 10 PGD steps is more accurate than a ResNet-152 trained with 30 PGD steps even under these powerful attacks.

Attack steps	100 steps			1000 steps		
	1	10	100	1	10	100
Random restarts						
ResNet-152-30	39.77%	39.37%	39.15%	37.41%	36.51%	35.75%
DENOISE	46.16%	44.68%	44.68%	43.57%	41.91%	41.37%
S3TA-16-30	47.76%	47.27%	47.09%	46.74%	46.09%	45.78%

Table 5: PGD attack with multiple restarts. We test the various method under attacks starting at multiple random initialization points. We test with 100 and 1000 PGD attack steps, initialized at 1, 10 and 100 random starting points. We count the attack successful even if only one of the adversarial examples caused the model to output the target class. We only count an image to be correctly classified if it was correctly classified across all adversarial images. As can be see, all models are quite robust here, though again, S3TA-16-30 outperforms them all.

	Top-1	Success rate
ResNet-152-30	26.38%	42.94%
DENOISE	32.99%	35.04%
S3TA-16-30	36.83%	27.00%

Table 6: Random targeted PGD attack with Adam optimizer. We measure the robustness of some models under an random targeted attack optimized with the Adam optimizer (rather than iterative FGSM). The attack uses 250 iterations, with learning rate 0.1 for the first 100 iterations, 0.01 for next 100 and 0.001 for the last 50. This has been shown [1] to be a stronger attack than the iterative FGSM. As can be seen, S3TA-16-30 is significantly better at defending here compared to the other models.

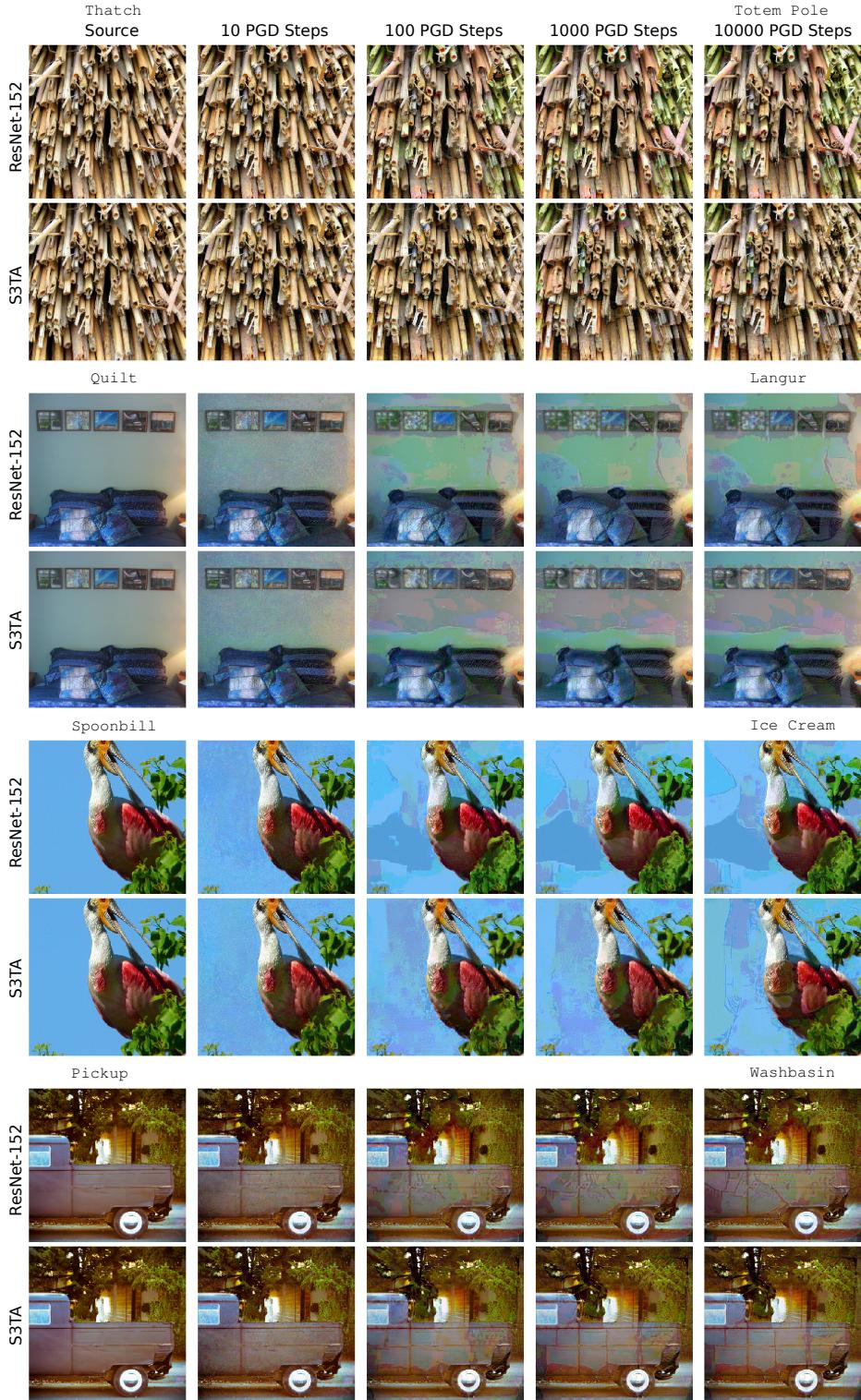


Figure 11: Adversarial images generated by PGD to attack the ResNet-152 and S3TA-4 model with no apparent visual structure. The source label is depicted on the left, the target label on the right. Source images are the leftmost column. We generate examples for 10, 100, 1000 and 10,000 PGD steps. These are cases where the perturbation does not have clear structure, as for many of the images. Some local structure may be interpretable - tiles for the washbasin for example, but there's little coherent global structure. Images are best viewed on screen and zoomed in.

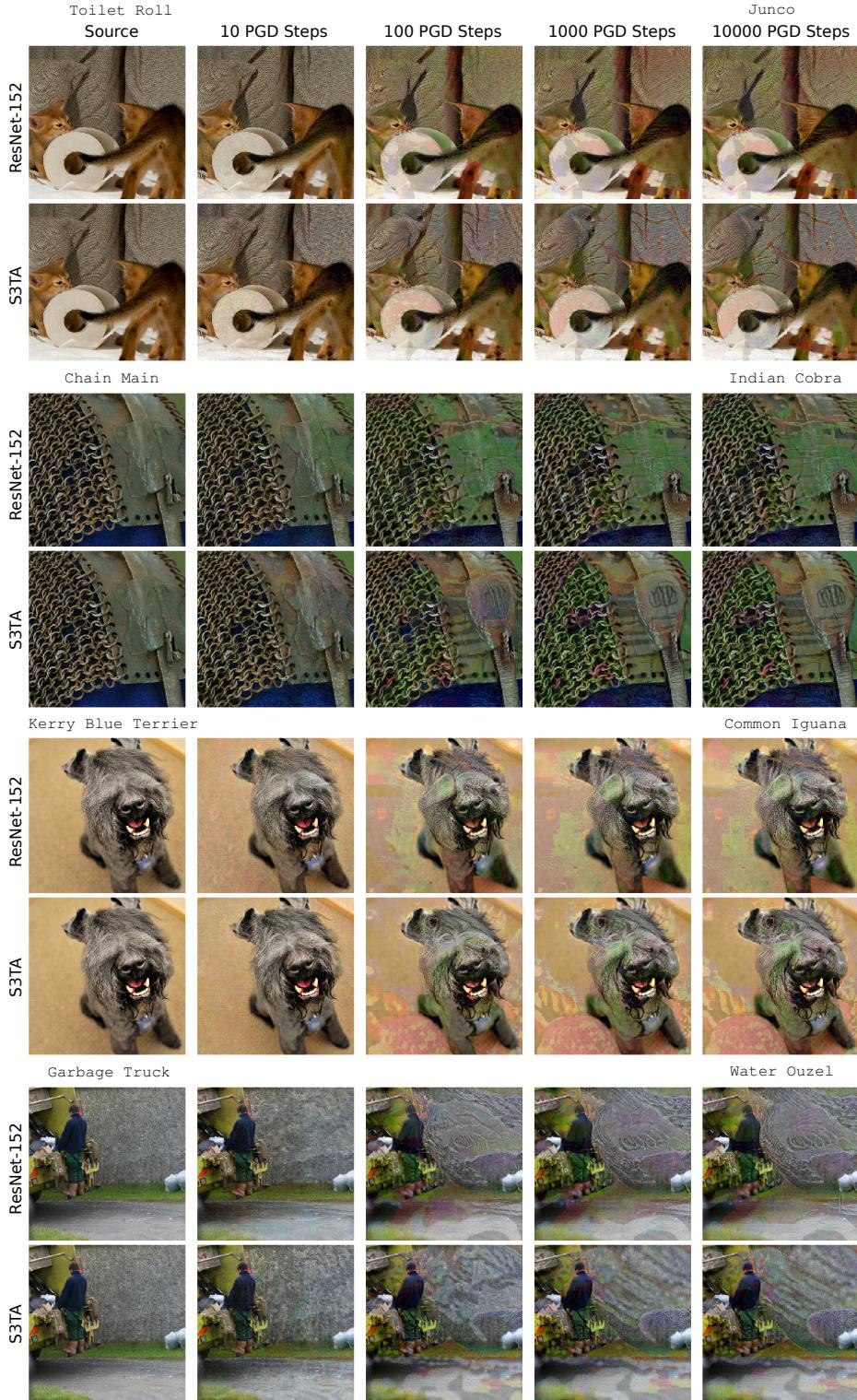


Figure 12: Adversarial images generated by PGD to attack the ResNet-152 and S3TA-4 model with clear and interpretable structure. The source label is depicted on the left, the target label on the right. Source images are the leftmost column. We generate examples for 10, 100, 1000 and 10,000 PGD steps. While the ResNet attacks are not particularly interesting, the S3TA examples are extremely structured. Note the salient, global and often realistic structures appearing - the Junco bird, the head of the Iguana, the body of the Indian Cobra or the Water Ouzel, a bird which is often depicted next to water (which is also created). We encourage the reader to view these on screen and zoomed-in.