Graph the Invisible: Reasoning for Adversarial Attacks with GNNs

Ziheng Cai [*1] **Irene Li** [*1] **Yue Wu** [*1]

Abstract

Deep learning is at the heart of the current rise of artificial intelligence. Whereas, deep neural networks have demonstrated phenomenal success (often beyond human capabilities) in solving complex problems, recent studies show that they are vulnerable to adversarial attacks in the form of subtle perturbations to inputs that lead a model to predict incorrect outputs. However, previous works focus either on gradient-based strategies or make use of black-box function approximator to generate attacks. This makes adversarial attacks virtually impossible to control or interpret. We propose a CEN which outputs a graphical model that benefits takes in location instructions and generate local attacks. We also propose a GNN that reasons about the objects in the scene and generates attacks based on the reasoning. To our knowledge, no prior works has been done on any graphical model-based adversarial attack generation.

1. Introduction

Deep learning thrived in solving the problems that have withstood many attempts of machine learning and artificial intelligence communities in the past. However, such advancement does not come with no drawbacks. Research in security and machine learning has exposed the vulnerability of machine learning to integrity attacks. A common technique for such attack is know as the adversarial attack: generating a new "adversarial sample" by adding small, imperceptible noise to the original input, forcing the learned DNN to misclassify the resulting sample.

Some intuitions has been provided that the existence of adversarial examples is due to the linearity of the network(Goodfellow et al., 2015), but the fact that non-linear attack methods are empirically superior suggests against such interpretation (Dong et al., 2018). No previous work has explored any means to control the adversarial attacks spatially.

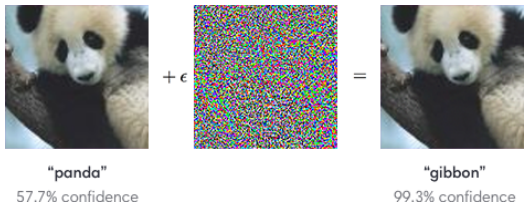

Figure 1. Illustration of an adversarial attack

ConvNets effectively make use of the spatial-invariant local information within the input. Graphs are the most typical locally connected structures. The extraction of spatial invariant features often relies on pooling as explained by (Bengio et al., 2013). Therefore, we expect locally-connected graphs to be able to capture some information that are valuable for pixel-level attacks.

Moreover, ConvNets are also shown to lack the ability of compositional reasoning and spatial reasoning. Graphical models, by their nature, are built to capture interactions and relationship, which then generates reasoning. Human reasoning can be captured by graphs. Another part of our exploration will be to investigate whether relations captured by graphs are helpful for adversarial attacks.

We believe graphical models are suitable for generating and controlling adversarial attacks because graphs are a kind of data structure which models a set of objects (nodes) and their relationships (edges). Recently, graphical models are regaining attention because of the great expressive powers of graphs, i.e. graphs has built-in reasoning and understanding of interactions between variables.

We identify two use-cases for graphs on adversarial attacks: one explores the relation between pixels of a image (compositionality), and the other one attacks the relation between entities and space (reasoning).

Compositionality We plan to explore a CEN which outputs a graphical model that benefits from locality (e.g. MRF/Deep MRF) but is not constrained to grids like a convolution. This Graph will then be used to attack and will

[*]Equal contribution [1]School of Computer Science, CMU. Correspondence to: Ziheng Cai <zcai@andrew.cmu.edu>, Mengze Li <mengzeli@andrew.cmu.edu>, Yue Wu <ywu5@cs.cmu.edu>.

Proceedings of the 36^{th} International Conference on Machine Learning, Long Beach, California, PMLR 97, 2019. Copyright 2019 by the author(s).

hopefully offer useful reasoning about the compositionality of the attack.

Reasoning We plan to explore a graph-based structure for generating explainable attacks on image inputs. We will initialize the graphical model with information extracted from the image (e.g., detected objects as nodes, distances between objects as edges). We will then perform reasoning on the graphical model. Finally, node-level features and edge-level features of the graph will be used to generate perturbation for adversarial attack. We hope that aside from generating efficient attacks, our graphical model can give reasoning about the generation of attacks.

2. Related Works

2.1. Graphical Neural Networks (GNN)

Graphical Neural Networks (GNN) are connectionist models that capture the dependence of graphs via message passing between the nodes of graphs. By its structure, GNN naturally offers interpretability. Recent advances in network architectures, optimization techniques and parallel computation have enabled successful learning with GNNs (Zhou et al., 2018). Gated Graph Neural Network (GGNN) (Li et al., 2015) modifies the primitive GNNs to use gated recurrent units and modern optimization techniques. On graph-structured inputs, GGNNs are demonstrated to have favorable inductive biases relative to purely sequence-based models. Another successful GNN, the Graph Convolutional Network (GCN) (Kipf & Welling, 2016) extends Conv nets to arbitrarily connected undirected graphs. GCNs learn hidden layer representations that encode both local graph structure and features of nodes.

GNNs have been applied to various image-related tasks, where graph-based reasoning is often performed to incorporate both spatial and semantic information. In regional classification, (Chen et al., 2018) uses a Graph Neural Network where regions and class labels are represented as labels, thus encoding both spatial and semantics information within the graph. For the task of Factual Visual Question Answering, (Narasimhan et al., 2018) identifies useful sub-graphs of a large knowledge graph and then use GCNs to produce representations encoding node and neighborhood features that can be used for answering the question. To understand social relationships in images, (Wang et al., 2018) initializes graph nodes with features extracted from regions of interest and employs the GGNN to propagate node messages through the graph to compute node-level features. It finally uses a graph attention mechanism to attend to the most discriminative nodes for identifying social relationships. In our second idea (i.e., the reasoning idea), we plan to take a similar approach as the above applications of GNNs. We will construct graph models encoding spatial and semantic information of image inputs, reason over the graph and use information stored in nodes and edges for generating attacks.

2.2. Contextual Explanation Networks (CEN)

While other GNNs focus on integrating with graphical models to directly augment the power of neural networks, CEN (Al-Shedivat et al., 2017) offers a different way. Given a collection of data where each instance is represented by inputs $c \in \mathcal{C}$, and targets $y \in \mathcal{Y}$, CEN constructs explanations in the form of simpler models $g_c : \mathcal{X} \to \mathcal{Y}$. While the original inputs, c, can be of complex, low-level, unstructured data types (e.g., text, image pixels, sensory inputs), we assume that x are high-level, meaningful variables.

Contextual Explanation Networks can generate parameters for a graphical model which is further used for localization of attack. In our first idea (i.e., the compositionality idea), we use a CEN to generate a CRF for capturing spacial relations between of the image input.

2.3. Adversarial Attack

(Szegedy et al., 2013) first demonstrated how small perturbation of images can fool the deep neural nets into misclassification. They employed Limited BFGS (L-BFGS) to approximate a minimized perturbation of the image so that the perturbed image labels differ from their original labels.

Adversarial attacks have been investigated for other major deep learning models such as deep generative models (Kos et al., 2018), Recurrent Neural Networks (Papernot et al., 2016) and Deep Reinforcement Learning (Lin et al., 2017). It has also been shown that adversarial attacks are effective in practical real-world conditions, such as Cell-Phone Camera Attack (Kurakin et al., 2016), Road Sign Attack (Evtimov et al., 2017), Generic Adversarial 3D Objects (Athalye et al., 2017) and Visual Question Answering Attacks (Xu et al., 2017).

Fast Gradient Sign Method (FGSM). After the discovery of (Szegedy et al., 2013), (Goodfellow et al., 2015) proposed 'Fast Gradient Sign Method' (FGSM) to efficiently compute the adversarial perturbation for a given image. This method exploits the linearity of deep neural networks in higher dimensional spaces, which (Goodfellow et al., 2015) speculates to stem from the designs of modern neural networks, which encourages linearity behavior.

Momentum Guided Adversarial Attack. (Dong et al., 2018)'s success in generating effective and efficient attacks demonstrated that the non-linearity of the generation process, giving us the intuition that it might be possible to model attacks through a non-linear function approximator.

Despite the popularity of the 'linearity hypothesis', many

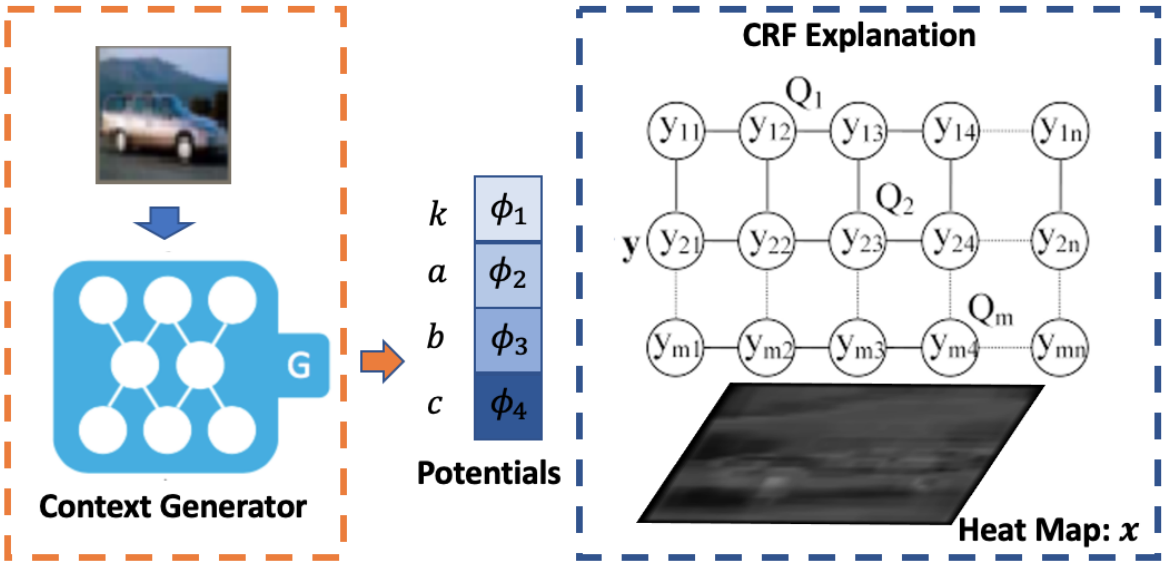

Figure 2. An illustration of our proposed architecture for the compositionality method.

studies (Tanay & Griffin, 2016; Luo et al., 2015) have demonstrated the existence of image classes that do not suffer from adversarial attack for linear classifier. This hypothesis is also in contrast with the common belief of the non-linearity of deep neural networks.

Many studies provided different explanations for the existence of adversarial examples. (Tanay & Griffin, 2016) hypothesized that adversarial examples exist because the classification boundary lies too close to the sub-manifold of the class samples. (Cubuk et al., 2018) argued that "the origin of adversarial examples is primarily due to an inherent uncertainty that neural networks have about their predictions". (Rozsa et al., 2016) attributed the existence of adversarial examples to the evolutionary stalling of decision boundaries on training samples, which stems from stalled contribution of correctly classified samples that lie closer to the decision boundaries as the training proceeds. Despite many attempts to provide an interpretation of the existence of adversarial examples, current literature still lacks consensus on the reasons of its existence.

3. Proposed Method

3.1. Compositionality Attack

3.1.1. Baseline

To fully test the feasibility of our idea, we define our baseline model to be a CRF weighted FGSM attack pipeline. A ConvCRF (Teichmann & Cipolla, 2018) produces a weight matrix which is directly used as ϵ to weigh the gradients for FGSM attack. This simple baseline should provide more insights about which potential function to select for the task.

3.1.2. The CEN-CRF Model

We construct our compositionality model by combining the Contexual Explanation Networks (Al-Shedivat et al., 2017) and the Conditional Random Field. We define the context to be the input image $\mathcal{I}$, and target to be the noise (perturbation) $\boldsymbol{y} \in \mathcal{Y}$. We model the process of producing the noise $y = G_\theta(\mathcal{I})$ with a CEN (Al-Shedivat et al., 2017) and a fully-connected CRF of the size of the image, with a hidden node for each pixel. The weights of the CRF are the outputs of the CEN, so we can view this process as the CEN generating a CRF based on the image context and CRF generating the noise $\boldsymbol{y}$ by inference. The noise $\boldsymbol{y}$ is then combined with the original image $\mathcal{I}$ and then used to attack the target classifier. Figure 2 illustrates the general model structure of our CEN-CRF model.

3.1.3. CRF Potentials

We define $\boldsymbol{x}$, the input to the CRF, to be a heatmap between $[0, 1]$. The heatmap highlights the "region of interest", on which we want the model to focus attack. We define $n = ch \times h \times w$ to be the size of the image vector reshaped from an image of channel number ch, height h and width

w. For each connection in the CRF, we define the following potential functions:

1. $\phi_1(y_i) = (y_i - k_i)^2$, where k_i is a prior value generated by the CEN.
2. $\phi_2(y_i, y_j) = (y_i - y_j)^2$
3. $\phi_3(x_i, y_i, x_j, y_j) = \frac{(y_i - y_j)^2}{(x_i - x_j)^2}$, where x_i and x_j are the pixels of the heat map $\boldsymbol{x}$ at position i and position j, respectively.
4. $\phi_4(\mathcal{I}_i, y_i, \mathcal{I}_j, y_j) = \frac{(y_i - y_j)^2}{(\mathcal{I}_i - \mathcal{I}_j)^2}$, where $\mathcal{I}_i$ and $\mathcal{I}_j$ are pixels of image $\mathcal{I}$ at position i and j, respectively.

We then perform a weighted sum of the above four potential functions:

$$\begin{aligned}\Phi(y, \mathcal{X}) = \sum_{i=1}^{n} w\phi_1(y_i) + \sum_{i=1,j=1}^{n,n} a_{i,j}\phi_2(y_i, y_j) \\ + \sum_{i=1,j=1}^{n,n} b_{i,j}\phi_3(y_i, y_j, x_i, x_j) \\ + \sum_{i=1,j=1}^{n,n} c_{i,j}\phi_4(y_i, y_j, \mathcal{I}_i, \mathcal{I}_j)\end{aligned} \quad (1)$$

where w, a, b and c are weights generated by the CEN based on input image $\mathcal{I}$. Note that the Equation 1 can also be written in matrix multiplication form:

$$\Phi(\boldsymbol{y}, \boldsymbol{x}) = \boldsymbol{y}^T(w\mathbf{I} + A' + \frac{B'}{X} + \frac{C'}{\mathcal{I}^*})y + (2w\boldsymbol{k})^T\boldsymbol{y} \quad (2)$$

where

- $\mathbf{I}$ is the identity matrix of size $n \times n$.
- $A' = \text{diag}(\sum_{i=1}^{n} a_{i,j}) - 2A + \text{diag}(\sum_{j=1}^{n} a_{i,j})$
- $B' = \text{diag}(\sum_{i=1}^{n} b_{i,j}) - 2B + \text{diag}(\sum_{j=1}^{n} b_{i,j})$
- $C' = \text{diag}(\sum_{i=1}^{n} c_{i,j}) - 2C + \text{diag}(\sum_{j=1}^{n} c_{i,j})$
- $X_{ij} = (x_i - x_j)^2$
- $\mathcal{I}^*_{ij} = (\mathcal{I}_i - \mathcal{I}_j)^2$
- $\boldsymbol{k}$ is the vector of prior noise of the same shape as $\boldsymbol{y}$.

This gives the conditional likelihood of noise $\boldsymbol{y}$ as follows

$$p(\boldsymbol{y}|\boldsymbol{x}) = \frac{\exp(-\phi(\boldsymbol{y}, \boldsymbol{x}))}{\int_{\boldsymbol{y}} \exp(-\phi(\boldsymbol{y}, \boldsymbol{x}))} \quad (3)$$

Note that during the inference step, we simply find the $\boldsymbol{y}^*$ that maximize the conditional likelihood $p(\boldsymbol{y}|\boldsymbol{x})$. Since the denominator in Equation 3 has marginalized $\boldsymbol{y}$, minimizing Equation 3 is equivalent as minimizing the potential (Equation 2). We constrain the weights in Equation 2 such that the Hessian matrix of $\Phi(\boldsymbol{y}, \boldsymbol{x})$ is positive-definite. Since Equation 2 is convex, we can solve for $\boldsymbol{y}^*$ by taking the gradient of Equation 2 and set it to zero. It is not hard to show that the minimizer of Equation 3 is

$$\boldsymbol{y}^* = (w\mathbf{I} + A' + \frac{B'}{X} + \frac{C'}{\mathcal{I}^*})^{-1}(w\boldsymbol{k}) \quad (4)$$

The output $\boldsymbol{y}^*$ is used as the adversarial noise on image $\mathcal{I}$ by direct summation $\mathcal{I}' = \mathcal{I} + \boldsymbol{y}^*$. Then $\mathcal{I}'$ will be used as the input of our target classifier T.

3.1.4. Loss function

Our loss function has the three components

1. **Target objective:** $\mathcal{O}_\theta = H\left(T\left(G_\theta(\mathcal{I}) + \mathcal{I}\right)\right)$. This is the cross-entropy of the target classifier T.
2. **Regularization:** $\mathcal{R}_\theta = \|G_\theta\|_2$. This regularization term is the L2 norm of the model parameters, which constrain the model weights.
3. **Auxiliary loss:** $\mathcal{A}_\theta = H(G_\theta(\mathcal{I}))$. To improve the performance of our CEN-CRF model, we introduced an auxiliary task to the CEN part of our model as to classify the input image $\mathcal{I}$. The cross-entropy of the classification gives the auxiliary loss.

Combining the above three components, we define the following loss function of our CEN-CRF model:

$$\mathcal{L}_\theta = -\mathcal{O}_\theta + \mathcal{R}_\theta + \mathcal{A}_\theta \quad (5)$$

Since the above method is fully differentiable, we can train our CEN-CRF model with conventional gradient based stochastic optimizer.

3.2. Reasoning Attack

We will train our Reasoning Attacker using multi-task learning, which includes two tasks: object classification and attack. The Attacker's structure is built on top of the reasoning framework from Iterative Visual Reasoning Beyond Convolutions (Chen et al., 2018). We will first use a ConvNet to generate an initial prediction f_0 as well as a corresponding attention a_0. Then we iteratively apply a graph reasoning module to capture both spatial and semantic relationships between objects. At each iteration, a new prediction f_i, an attention a_i and a perturbation p_i are made from graph features. Finally, combine predictions and perturbations from all iterations with the attention to generate final results.

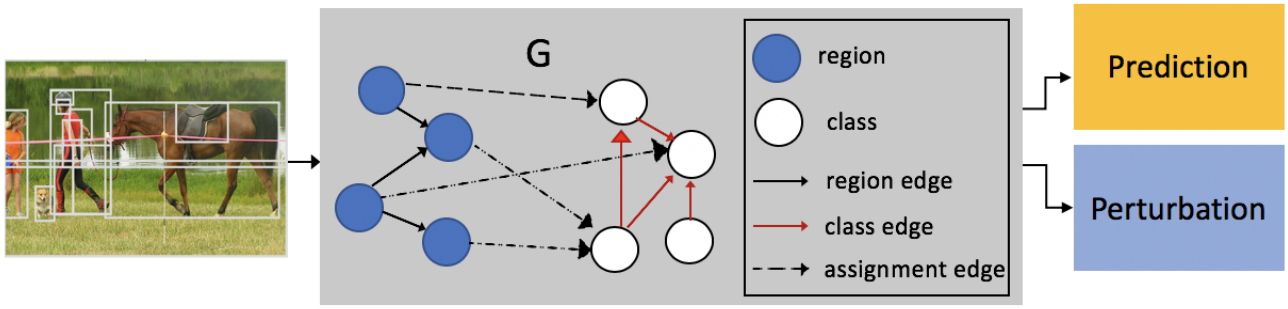

Figure 3. Reasoning Attack Model, described in section 3.2. The middle grey block is the graph reasoning module, where information of objects are propagated to each other. The model is trained through multi-task learning: the final graph feature is used for both classifying objects and adversarial attack.

The graph reasoning module proposed in (Chen et al., 2018) explores two types of relationships. The first one is spatial relationship, meaning regions far away could directly communicate information with each other. The second relationship is semantic, which is realized by the use of a knowledge base of object classes.

In the graph reasoning module, we construct a graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$, where $\mathcal{N}$ and $\mathcal{E}$ denote nodes and edges respectively. Two types of nodes are defined in $\mathcal{N}$: region nodes $\mathcal{N}_r$ for R regions (or bounding boxes), and class nodes $\mathcal{N}_c$ for C classes. $\mathcal{N}_r$ is represented by feature maps extracted from ConvNet. $\mathcal{N}_c$ is represented by word vectors of class names from fastText (Joulin et al., 2016).

Three groups of edges are defined between nodes. First for $\mathcal{N}_r$, a spatial graph is used to encode spatial relationships between regions $\mathcal{E}_{r\to r}$. This spatial graph characterizes 5 types of relationships: left, right, bottom, up coverage pattern (intersection over union). A second group of edges $\mathcal{E}_{r\to c}$ lie between region nodes $\mathcal{N}_r$ and class nodes $\mathcal{N}_c$, which encodes assignment for a region to a class. At iteration i, previous prediction f_{i-1} is used to define edge weights of connections from all regions to all classes. Semantic relationships from knowledge bases are used to construct the third group of edges between $\mathcal{E}_{c\to c}$. We include 5 types of edges within $\mathcal{E}_{c\to c}$: "is-kind-of", "is-part-of", "plural-form", "horizontal-symmetry", "similarity".

For message passing, we follow the two reasoning paths defined in (Chen et al., 2018) to learn the output features G_r. Finally, we use G_r to generate prediction f_i, attention a_i and perturbation p_i.

For training our Reasoning Attacker, we use a weighted loss consisting of prediction loss, attack loss and a regularization term. Prediction loss measures how accurate the final classification is. Attack loss is obtained by using the perturbation to attack an object classifier (we will either use VGG or ResNet for the classifier). The regularization term is meant to minimize the norm of the generated perturbation.

4. Datasets

CIFAR-10 dataset. CIFAR-10 contains $60,000$ $32\times32\times3$ RGB images of objects, with $50,000$ training samples and $10,000$ testing samples. It contains 10 mutually exclusive classes. This dataset is used to experiment our compositionality method. The goal is to fool classification models into misclassifying images.

ADE20K. (Zhou et al., 2016) ADE20K is a scene parsing benchmark dataset. Images in ADE20K datasets are densely annotated in detail with objects and parts. Each image has a rich relationship between objects, so it is an ideal dataset for showcasing the potential abilities of our Reasoning Attacker. We will use ADE20K to train our Reasoning Attacker on the task of object classification. Specifically, we will convert segmentation masks from ADE20K to generate ground-truth bounding boxes and train a classifier to predict class label for each bounding box. The goal of the Reasoning Attacker is to generate a small perturbation to harm the performance of the classifier.

5. Experiments

5.1. Compositionality Method (CEN-CRF Model)

5.1.1. Target Classifier

We used a pretrained VGG-16 network trained on CIFAR-10 as our target classifier to be attacked on. This model achieves the classification accuracy of 99.8% train set and 88.6% on test set.

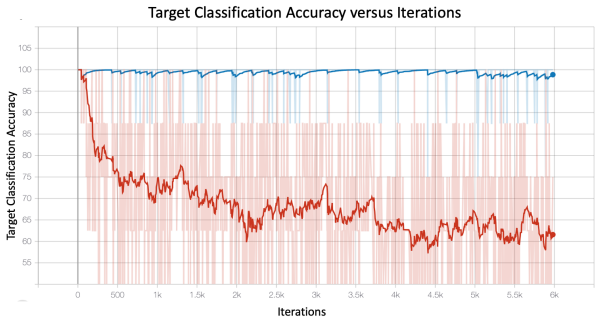

Figure 4. Clarification accuracy of target classifier **blue:** before attack **orange:** after attack.

5.1.2. Without Masking

Figure 4 shows how the accuracy of the target neural networks drops as we train our compositionality attacker for more iterations. This greatly confirms the validity our proposed CEN-CRF approach. We can also view this from the increase of the target cross-entropy (target loss) as the training proceeds, as shown in Figure 5.

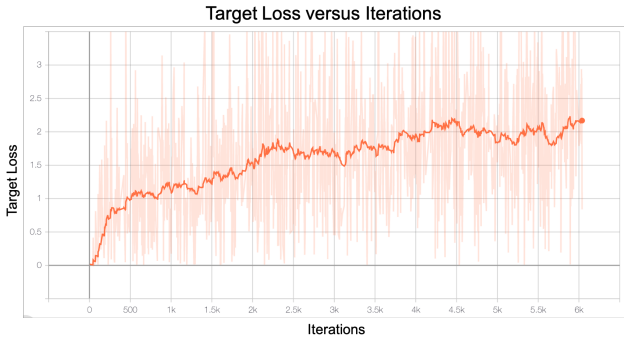

Figure 5. Target cross-entropy increases as training iteration increases, indicating the effectiveness of the attack.

We compare our attack results to two baseline methods: Fast Gradient Sign Method, Iterative Fast Gradient Sign Method.

We noticed that compared to the baselines (FSGM, iFSGM), the noise generated by our model is much smoother and more concentrated around some areas. This corresponds to the general ”smoothing” effects of CRFs.

To rule out the situation where it is the CEN instead of CRF that learns the noise, we also present the following distributions of prior noise and the actual generated noise , as shown in Figure 6 and Figure 7.

We make an important observation that the distribution of the prior noise (K) unimodal is different from the distribution of the posterior (Output Noise) $(\mathcal{N})$. We therefore

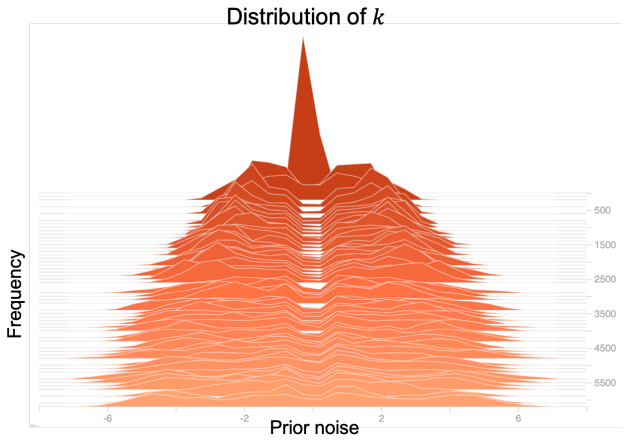

Figure 6. Distribution of prior noise k

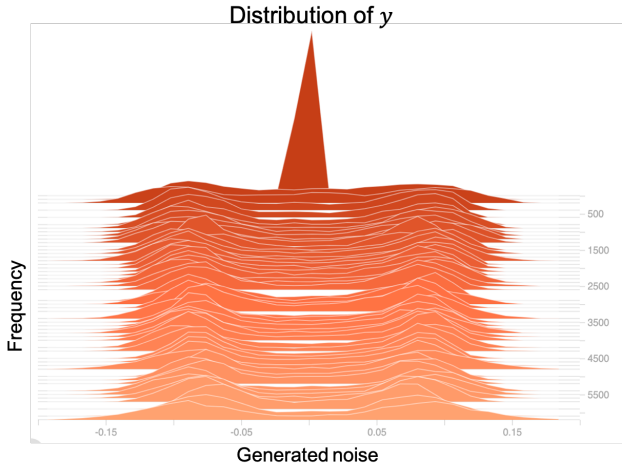

Figure 7. Distribution of the actual generated noise y.

conclude that the output of our attack model does make use of the pixel-level edge potentials.

5.1.3. With Masking

Currently, the implementation with masking is still generating the same noise for all the attacks. The CRF seems to be generating the same noise no matter what the input mask is.

We believe that this is caused by the insufficiency of our potential functions to enforce the mask. We have tested a bunch of potential functions and achieved some progress, but we are still far from getting the CRF to work.

5.2. Reasoning Attack

5.2.1. Data preprocessing

In ADE20k, we filter out images without ground truth labelled bounding boxes. We also filter out rare classes. Specifically, we discard classes that occur fewer than 5 times in the whole dataset. After filtering, we are left with 1485 classes. For each image, we resize it such that the longer side has dimension 600. Large images are cropped such that no dimension exceeds 600. We then normalize RGB values of each image by subtracting mean and dividing by the standard deviation. Finally, if an image has more than 100 labeled bounding boxes, we randomly pick 100 boxes during training phase and choose the first 100 boxes during validation phase. This is for the purpose of more stable utilization of GPU. During testing, all bounding boxes are kept.

5.2.2. Target Classifier

We choose the baseline model in (Chen et al., 2018). The model is a simplified version of Faster R-CNN (with the proposing of regions of interest removed) as the network to be attacked. The backbone classifier is ResNet-50 pretrained on ImageNet. The last conv4 features and ground-truth bounding boxes are used to compute per-region features. Per-region features are then fed into layers above conv4 to obtain final features for classification. A fully-connected layer is then used to compute predictions. Parameters of conv1 layer and conv2 layer are fixed. Batch normalization parameters are also fixed.

The ADE20k dataset is split into 3 sets for training, validation and testing. The training set has 40420 images. Both validation set and testing set have 1000 images. The model is trained by an SGD optimizer for 17 epochs. Classification results are evaluated by average precision (AP) and average classification accuracy (AC). We take the average over both instances and classes. Table 1 displays the results.

Model	Ins AP	Ins AC	Cls AP	Cls AC
Before Attack	65.0	65.0	37.6	33.9
After Attack	4.3	4.3	0.2	0.1

Table 1. Classification results of target classifier on ADE20k test set before and after attack. AP is average precision and AC is average classification accuracy. Ins means per-instances and Cls means per-class. There's a significant decrease in both precision and accuracy after attack

5.2.3. Training of Attacker

We use the same structure as the target classifier to extract per-region features. That is, output of conv4 layer from ResNet-50 and ground-truth bounding boxes are used to compute per-region features. Note that although this part of the model shares its structure with target classifier, they don't share parameters. The target classifier and the attacker are trained independently.

Per-region features then go through a fully connected layer and then get passed to the Graph Reasoning Module as initial node features. The dimension of node features is set to be 512. We then perform message passing on the graph for two iterations, where each iteration contains three passes of the message passing procedure described in section 3.2 of (Chen et al., 2018). After the Graph Reasoning Module, the model splits into two different branches, one for region classification and the other for adversarial attack.

For the region classification task, node features from all iterations (including initial node features) are used for classification. Features from each iteration also generate a confidence score, which is used to weight its prediction when we linearly combine all the predictions.

For generating perturbation, only the final node features are used. The node features is first fed through a fully-connected layer and then resized to the size of the original image using CropAndResize. We then average over the nodes and feed the features through another fully-connected layer to generate a perturbation of exactly the same dimension (color, width and height) as the original image. We perform white-box attack, which means we make use of the target classifier while training our attacker. The generated perturbation is directly added to the image input (after data preprocessing as described in 5.2.1) and then the attacked image is fed to the target classifier.

Our model is trained end-to-end with three loss terms. The first one is cross entropy loss of the region classification branch with a multiplicative factor of 1. The second loss term is the exponential of the negative of cross entropy loss output by the target classifier after attack, with a multiplicative factor of 100. The third loss term is L2 norm of

perturbation divided by the total size of perturbation. The third loss term has a multiplicative factor of 10000.

Our attacker is trained with an SGD optimizer for 7000 iterations (where each iteration is one image because the input cannot be batched due to different sizes of their graphs). Table 2 demonstrates the effect of the attack. The performance of the target classifier drops significantly. Additionally, the average L2 norm of perturbation on test set is $7.03 \cdot 10^{-6}$.

6. Conclusion

6.1. Compositionality Attack

From our current results, we observe that our CRF does not solely dependent on the prior (K) from CEN. The weighted potential functions are easily interpretable. The CRF demonstrates good performance at suppressing the noise, yet maintaining good attack quality.

While we are not able to get the heat-map weighted CRF working, we believe that our CRF based approach can still offer some degree of insight on how the potential functions interact with each other.

Furthermore, our successful results suggests that current adversarial attack methods might benefit from the use of a ad-hoc CRF.

Two of the most obvious drawbacks to our current approach includes: 1. Insufficiency of potential functions (because we have to keep the objective convex). 2. Slow computation speed to invert a full weight matrix during both training and inference phase.

We propose to address the problems through variational inference, by proposing variational distributions $q_i(w)$ each capturing an interpretable potential function. We will use the CEN to output a weight for each distribution q_i and sample from the joint distributions of q_i using the reparametrization trick. This will allow us to train CRFs with fast inferece/training time and more complex potential functions.

6.2. Reasoning Attack

Our Reasoning attacker is able to effectively attack the target classifier with a small perturbation (based on L2 norm). This indicates that reasoning of object relationships within images could potentially help with adversarial attack. However, our experiments are not thorough enough:

- we noticed that the classification branch of our attacker does not perform well compared to the target classifier. Thus, the full potential of graph reasoning module has not been utilized. We suspect that by improving classification branch, we can get better results on attack as well.
- In addition to L2 norm, we should visualize the images to confirm attack is not detectable by human eyes.

References

Al-Shedivat, M., Dubey, A., and Xing, E. P. Contextual Explanation Networks. *arXiv e-prints*, art. arXiv:1705.10301, May 2017.

Athalye, A., Engstrom, L., Ilyas, A., and Kwok, K. Synthesizing robust adversarial examples. *CoRR*, abs/1707.07397, 2017. URL http://arxiv.org/abs/1707.07397.

Bengio, Y., Courville, A., and Vincent, P. Representation learning: A review and new perspectives, 2013.

Chen, X., Li, L.-J., Fei-Fei, L., and Gupta, A. Iterative Visual Reasoning Beyond Convolutions. *arXiv e-prints*, art. arXiv:1803.11189, Mar 2018.

Cubuk, E. D., Zoph, B., Schoenholz, S. S., and Le, Q. V. Intriguing properties of adversarial examples, 2018. URL https://openreview.net/forum?id=rk6H0ZbRb.

Dong, Y., Liao, F., Pang, T., Su, H., Zhu, J., Hu, X., and Li, J. Boosting adversarial attacks with momentum. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 9185–9193, 2018.

Evtimov, I., Eykholt, K., Fernandes, E., Kohno, T., Li, B., Prakash, A., Rahmati, A., and Song, D. Robust physical-world attacks on machine learning models. *CoRR*, abs/1707.08945, 2017. URL http://arxiv.org/abs/1707.08945.

Goodfellow, I., Shlens, J., and Szegedy, C. Explaining and harnessing adversarial examples. In *International Conference on Learning Representations*, 2015. URL http://arxiv.org/abs/1412.6572.

Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., and Mikolov, T. FastText.zip: Compressing text classification models. *arXiv e-prints*, art. arXiv:1612.03651, Dec 2016.

Kipf, T. N. and Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. *arXiv e-prints*, art. arXiv:1609.02907, Sep 2016.

Kos, J., Fischer, I., and Song, D. X. Adversarial examples for generative models. *2018 IEEE Security and Privacy Workshops (SPW)*, pp. 36–42, 2018.

Kurakin, A., Goodfellow, I. J., and Bengio, S. Adversarial examples in the physical world. *CoRR*, abs/1607.02533, 2016. URL http://arxiv.org/abs/1607.02533.

Li, Y., Tarlow, D., Brockschmidt, M., and Zemel, R. S. Gated graph sequence neural networks. *CoRR*, abs/1511.05493, 2015.

Lin, Y., Hong, Z., Liao, Y., Shih, M., Liu, M., and Sun, M. Tactics of adversarial attack on deep reinforcement learning agents. *CoRR*, abs/1703.06748, 2017. URL http://arxiv.org/abs/1703.06748.

Luo, Y., Boix, X., Roig, G., Poggio, T. A., and Zhao, Q. Foveation-based mechanisms alleviate adversarial examples. *CoRR*, abs/1511.06292, 2015.

Narasimhan, M., Lazebnik, S., and Schwing, A. e. G. Out of the Box: Reasoning with Graph Convolution Nets for Factual Visual Question Answering. *arXiv e-prints*, art. arXiv:1811.00538, Nov 2018.

Papernot, N., McDaniel, P. D., Swami, A., and Harang, R. E. Crafting adversarial input sequences for recurrent neural networks. *CoRR*, abs/1604.08275, 2016. URL http://arxiv.org/abs/1604.08275.

Rozsa, A., Günther, M., and Boult, T. E. Towards robust deep neural networks with BANG. *CoRR*, abs/1612.00138, 2016.

Szegedy, C., Zaremba, W., Sutskever, I., Bruna, J., Erhan, D., Goodfellow, I. J., and Fergus, R. Intriguing properties of neural networks. *CoRR*, abs/1312.6199, 2013. URL http://arxiv.org/abs/1312.6199.

Tanay, T. and Griffin, L. D. A boundary tilting persepective on the phenomenon of adversarial examples. *CoRR*, abs/1608.07690, 2016.

Teichmann, M. T. and Cipolla, R. Convolutional crfs for semantic segmentation. *arXiv preprint arXiv:1805.04777*, 2018.

Wang, Z., Chen, T., Ren, J., Yu, W., Cheng, H., and Lin, L. Deep Reasoning with Knowledge Graph for Social Relationship Understanding. *arXiv e-prints*, art. arXiv:1807.00504, Jul 2018.

Xu, X., Chen, X., Liu, C., Rohrbach, A., Darell, T., and Song, D. Can you fool AI with adversarial examples on a visual turing test? *CoRR*, abs/1709.08693, 2017.

Zhou, B., Zhao, H., Puig, X., Xiao, T., Fidler, S., Barriuso, A., and Torralba, A. Semantic Understanding of Scenes through the ADE20K Dataset. *arXiv e-prints*, art. arXiv:1608.05442, Aug 2016.

Zhou, J., Cui, G., Zhang, Z., Yang, C., Liu, Z., and Sun, M. Graph Neural Networks: A Review of Methods and Applications. *arXiv e-prints*, art. arXiv:1812.08434, Dec 2018.