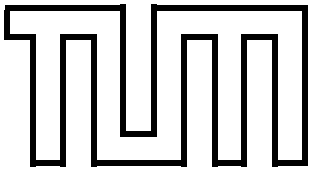


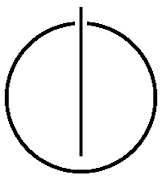
DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Adversarial Examples That Fool Classic Computer Vision Methods

Ege Özsoy



DEPARTMENT OF INFORMATICS

TECHNICAL UNIVERSITY OF MUNICH

Bachelor's Thesis in Informatics

Adversarial Examples That Fool Classic Computer Vision Methods

Adversarial Examples, die klassische Methoden der Computer Vision täuschen

Author:	Ege Özsoy
Supervisor:	Prof. Dr.-Ing. habil. Alois Christian Knoll
Assessor:	Thomas Brunner
Submission Date:	16.09.2019

I confirm that this bachelor's thesis in informatics is my own work and I have documented all sources and material used.

Munich, 03.09.2019 Ege Özsoy

Abstract

With the recent advancements in deep learning, image recognition started to play a critical role in many different tasks. Every day, it is used more and more for security-critical applications, such as face authentication and autonomous driving. It is well known that deep learning based approaches can be fooled using adversarial examples, images with small perturbations that are classified falsely. Since their discovery, the research community has been working on ways to generate better such examples, and also better defend against them. One area that was not investigated thoroughly is how robust the image recognition methods that are based on classical computer vision methods are. These are algorithms that use handcrafted feature extractors in conjunction with classical machine learning models. This study compares three commonly used feature extractors, BoVW (Bag of Visual Words), Fisher Vector, and HOG (Histogram of Oriented Gradients); with three classical machine learning classifiers, Support Vector Machine, Logistic Regression, and Random Forest, and concludes that all image recognition methods are susceptible to adversarial attacks, albeit in different degrees. BoVW and Fisher Vector with Support Vector Machine or Logistic Regression, are generally the most robust, while CNN (Convolutional Neural Network) and HOG based approaches are generally the least robust. Nonetheless, deep learning only exaggerated the problem of adversarial examples and did not start it, as adversarial examples are observed for any image recognition method tested in this thesis.

Contents

1 Introduction and Theory

This chapter introduces the intention of this work, and the knowledge needed to understand it.

1.1 Introduction

Image classification is one of the central challenges of computer vision. Fast, accurate, and robust understanding of what object is in an image is a requirement for many tasks such as autonomous driving, face recognition systems, robotics, healthcare, retail and security industries. The current state of the art is using deep learning for these tasks. While its popularity has been increasing day after day, it has also been known for some time that deep learning methods such as convolutional neural networks can be fooled by using adversarial examples. Their lack of robustness did not stop their adoption as for most tasks their advantages outweigh their disadvantages.

Nonetheless, to this day, there has not been a comprehensive study analyzing how robust deep learning based image classification methods (such as CNNs) are, compared to classical computer vision methods. The talk about adversarial examples has been mostly limited to deep neural networks, with even some papers mentioning it as a disadvantage of deep learning[1][2] while leaving out how classical methods fare in comparison. This unclarity can lead many to believe that it can be a better idea to use classical computer vision methods as they would be robust against adversarial examples.

Even though Goodfollow et al.[3] mention that compared to shallow linear models, "deep networks are at least able to represent functions that resist adversarial perturbation", to the best of the author's knowledge, no such comprehensive study exists. Furthermore, resources regarding the robustness of the handcrafted feature extractors against adversarial attacks, when they are combined with the aforementioned linear models are also lacking.

The actual robustness of image classification methods is crucial to make an assessment when deciding the type of architecture to use when the robustness of the classification system is a high priority. In summary, the main objective of this thesis is comparing different methods used for image classification to determine their robustness towards adversarial examples. Knowing how much robustness one gains from using a different method, which might be less accurate, would give one the ability to trade accuracy against robustness, similar to how precision is traded against recall in the field of computer vision.

1.2 Background Knowledge

This section covers the essential knowledge that is needed to follow the study.

1.2.1 Classical Computer Vision Pipeline

Before the popularity of the deep learning based approaches, a typical image classification pipeline consisted of a handcrafted feature extraction method, and a classical machine learning model. Thus, the accuracy and robustness of a classification system is not only dependent on what the model learns, but also what kind of a feature extractor is used. This is different for CNNs, as they simultaneously act as a feature extractor and a machine learning model, giving them the flexibility of extracting different features depending on the task at hand.

This study will be recreating and evaluating some of the most used and successful pipelines, that represent the state of the art for the classical computer vision based image classification methods.

1.2.2 Popular Machine Learning Methods

The study includes the following classical machine learning methods, as they are some of the most commonly used machine learning models: support vector machine[4], logistic regression[5], and random forest classifier[4]. These methods are explained shortly without going into too much detail, as they are well known, and there are many useful resources about them. A visualization of them for binary classification can be seen in Fig 1.1.

In addition to the methods mentioned above, deep learning based CNNs will be used as they are state of the art for image classification, with uses in many different applications.

Logistic Regression[5]

Logistic regression is a high-speed, and easily interpretable algorithm, that is still widely used for many tasks. Given a set of samples **X** and associated labels **Y**, it finds decision boundaries that best separate the classes. The calculation of the decision boundaries is often referred to as the training step. Once training is completed, the algorithm can be used to predict the classes of a new set of samples. As it is a linear classifier, it is less prone to overfitting compared to some other methods.

Random Forest Classifier[4]

Decision trees are non-linear and non-parametric models, which are also easy to interpret. They also perform well if the data is not linearly separable. Given a set of samples **X** and associated labels **Y**, it separates them into subsets, according to specific attributes, and keeps separating the subsets until a stopping criterion is reached. The resulting tree structure can be used to classify a new sample, just by following one decision after another until a leaf node is reached.

Decision trees are very prone to overfitting, and to deal with that random forest classifiers can be used. They consist of many decision trees and return the label that got predicted the most. To make sure every decision tree is different, usually, each decision tree is trained using only a random subset of the features.

Support Vector Machines[4]

Support vector machines with linear kernels are non-probabilistic discriminative linear classifiers. They find a hyperplane that best separates the samples into two classes. Support Vector Machines are capable of dealing with high dimensional data and are memory efficient as they only use a part of the training samples(support vectors).

To use them with more than two classes, usually one vs. rest approach is used, which means for **N** classes, **N** classifiers are trained. The $\mathbf{i_{th}}$ classifier considers all samples from the $\mathbf{i_{th}}$ class as positive and everything else as negative. A new sample belongs to the $\mathbf{i_{th}}$ class, if among all classifiers the $\mathbf{i_{th}}$ classifiers has the highest value for the sample. This requires the classifiers to produce a confidence score for their decisions. [6]

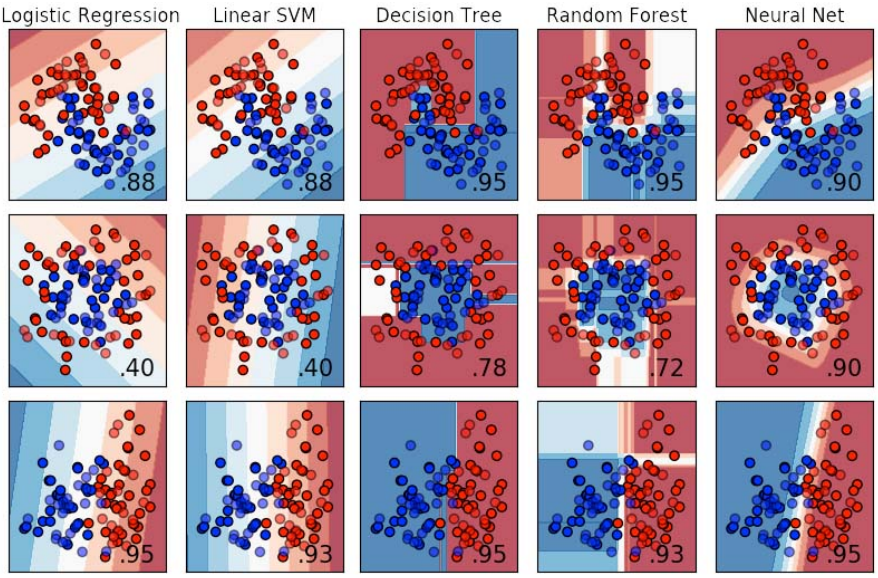


Figure 1.1 Visualization of decision boundaries for the used classifiers. The test accuracy can be seen on the lower right corner. [7]

1.2.3 Popular Feature Extractors

This study includes the following feature extractors as they are some of the most commonly used feature extractors in the literature: Histogram of Oriented Gradients[8], Bag of Visual Words[9] and Fisher Vector[10].

Scale Invariant Feature Transform (SIFT)[11]

SIFT is a feature extraction algorithm used to calculate key-points and corresponding descriptors in an image. SIFT follows a multi-step pipeline, which consists of the following steps:

- Scale-space extrema detection: Search over different scales and image positions. Use Difference-of-Gaussian to find points of interest which are invariant to scale and rotation.
- Keypoint localization: For every point of interest, a model is fit to determine the position and scale. Keypoints are then selected based on their stability.
- Orientation assignment: Orientations are assigned to each keypoint based on local image gradients. Future operations will be performed on images, which are transformed according to the assigned orientation, position, and scale, which provides invariance to these transformations.
- Keypoint descriptor: Local image gradients around the selected scale are transformed into a representation that provides invariance towards shape and illumination changes.

Figure 1.2 shows a visualization of SIFT features calculated for four images. It can be seen that SIFT extracts more key-points for areas with higher contrast in an image.

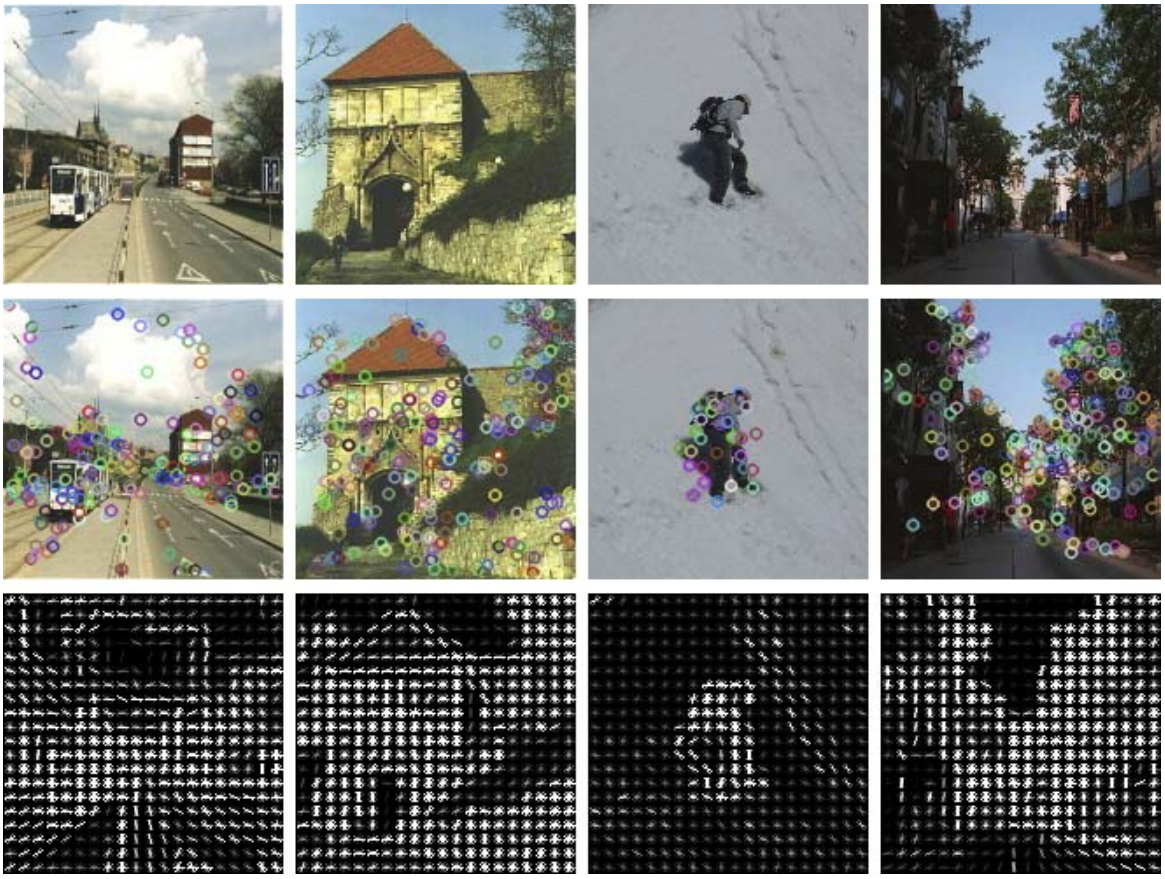


Figure 1.2 Visualization of images from the INRIA dataset with SIFT(middle row), and HOG(bottom row) features

Bag of Visual Words[9]

Bag of Visual Words (BoVW) is a concept inspired by the Bag of Words used commonly in natural language processing. Unlike counting the times a word appears in a text, this approach counts the number of times a feature appears in an image. Then a collection of unordered patches represents an image.

To accomplish this, first SIFT is used to extract features from an image. Then, a visual vocabulary is constructed using the K-means clustering algorithm. Every cluster is considered to represent a visual word, and therefore a feature vector assigned to that cluster is also considered to represent that visual word. A vocabulary size of K is chosen so that it is large enough to differentiate relevant image parts, but also small enough not to get affected by small variants such as noise.

In test time, the nearest visual word is found for every feature vector extracted from an image, and a normalized histogram of vocabulary words encountered in the image is computed. This histogram provides a representation for the image, analogous to how the bag of words approach provides representation for a text.

A visualization can be seen in Fig 1.3.

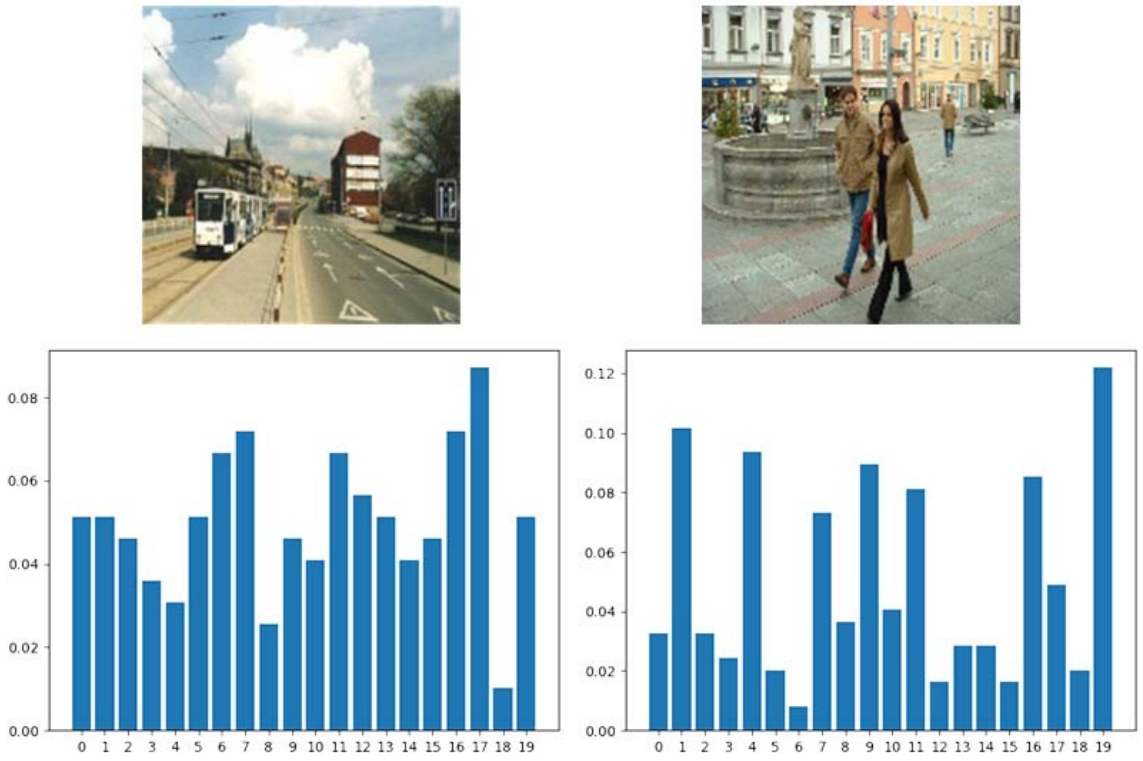


Figure 1.3 Visualization of the histogram calculated by the BoVW algorithm. The histogram on the left corresponds to the upper left image, which contains no person. The right histogram corresponds to the upper right image, which contains a person. For visualization purposes, vocabulary size was set to 20.

Fisher Vector[10]

Fisher Vector can be seen as an extension of the Bag of Visual Words method, which can result in similar or better classification performance compared to the Bag of Visual Words approach. After BoVW alike feature extraction with SIFT, a Gaussian Mixture Model is used to model the distribution of extracted features. Then, the Fisher Vector describes an image with the gradients of the log-likelihood of the features from the Gaussian Mixture Model.

This representation, unlike the Bag of Visual Words, does not only represent the number of occurrences of every visual word in an image but instead encodes richer information, theoretically allowing better classification performance with linear kernels. Authors of the mentioned paper also employ L2 Normalization, which removes dependence on small objects with small image-specific information, which leads to significant improvements; and also Power Normalization, to make the fisher vector denser.

Histogram of Oriented Gradients[8]

The Histogram of Oriented Gradients (HOG) is a very popular feature extractor, first used to detect humans in images. An image is divided into small regions, and vertical and horizontal gradients are calculated for those regions, which have magnitudes and directions. Then the direction and magnitude of the gradients are used to build a one-dimensional histogram. Next, every region is normalized to increase invariance to illumination changes. Lastly, histograms from every region are used to represent the entire image.
Figure 1.2 shows a visualization of Hog features calculated for four images.

1.2.4 Deep Learning and CNNs

Deep learning is a subfield of machine learning, which gained in popularity in the last years. One of the significant milestones for deep learning was in 2012, where a convolutional neural network [12] won

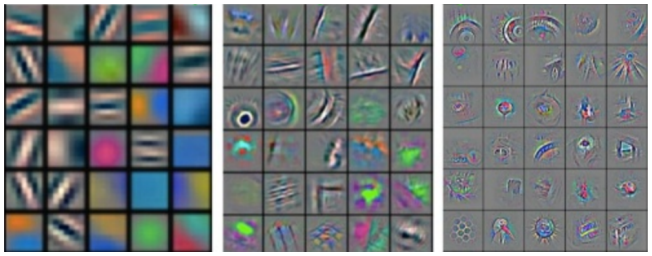


Figure 1.4 Visualization of an example feature extraction from the first, second and third convolutional layers. It can be seen that the extracted features are more complex in the later layers. [13]

the Imagenet image classification challenge. Convolutional neural networks have become very popular for many different image-related tasks. Compared to classical machine learning and computer vision methods, given enough data, these networks can learn more complex and accurate functions. Unlike the machine learning methods described previously, these networks both extract the features from an image and also classify it. So CNNs do not need a separate feature extractor to get good results.

Fig 1.4 shows, how every layer of the CNN architecture extracts higher level features compared to the previous layer. Unlike other types of feature extraction methods, CNNs learnable parameters allow extraction of task-specific features. This flexibility is one of the reasons for their state of the art performance.

1.2.5 Adversarial Examples

Soon after convolutional neural networks became popular, it was discovered [14] that they could be confused by so-called "Adversarial Examples" which are classified falsely, even though they are visually hardly distinguishable from regular examples. Goodfellow et al.[15] used this property to come up with a fast and straightforward method of generating such examples.

1.2.6 Popular Attacking Methods

There are many different types of attacks for generating adversarial examples. Firstly, attacks can be separated by the informations they need, to attack. White-box attacks need to have access to the model, while black-box attacks do not. Attacks can also be targeted or untargeted. In the untargeted setting, the attack only tries to make sure an image is misclassified. In the targeted setting, the goal is to make sure an image from class **A** is classified as class **B**(and not any other class such as **C**).

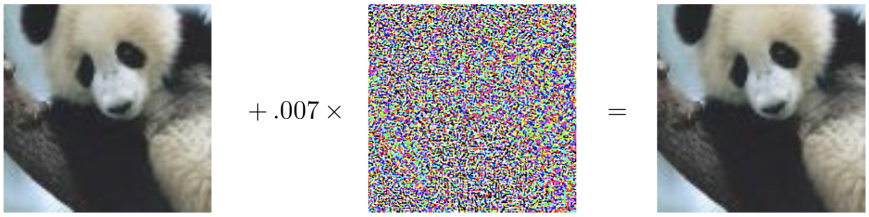


Figure 1.5 Visualization of how applying a visually imperceptible noise to an image can cause it to get misclassified. [15]

FGSM (Fast Gradient Sign Method)

FGSM is the name of the white-box attack developed by Goodfellow et al.[15] to generate adversarial examples. As it uses the gradients of the model it is quick and accurate. However, because of their design, FGSM and similar attacks are not usable against classical computer vision pipelines such as SVM with Bag of Visual Words, as these pipelines do not have any gradients to utilize.

Boundary Attack

Boundary Attack [16] is a relatively new iterative black-box attack, which unlike FGSM only requires the classified label from the classification algorithm and not the gradients. They were mainly developed to attack neural networks, whose architecture were not known. Because the attack is iterative, an image classification system is attacked many times, each time improving the adversarial example. The attack can be described as follows:

- First the algorithm is initialized from a point that is already adversarial. If the attack is untargeted, a sample is generated from the maximum entropy distribution for the valid input range. In the targeted scenario a sample is used, which is classified as being from the target class.
- It then performs a random walk, as can be seen in Fig 1.6 between the adversarial and non-adversarial regions, while staying in the adversarial one. A proposal distribution algorithm is used to generate random directions to explore.

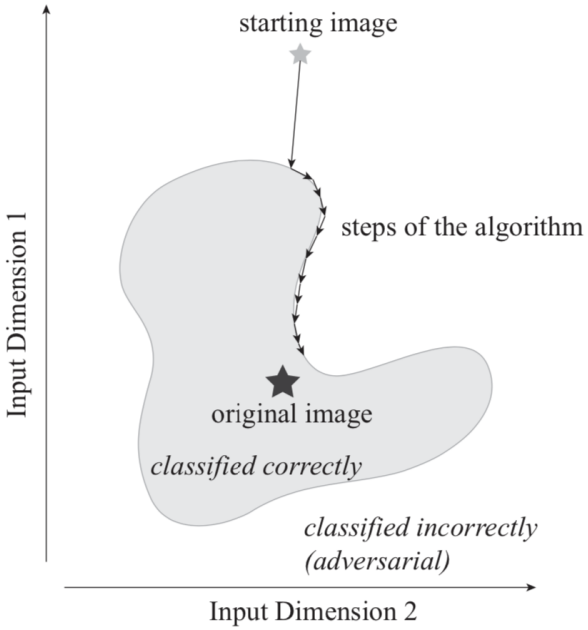


Figure 1.6 Visualization of the random walk towards the original image while staying in the adversarial region. [16]

HopSkipJumpAttack

HopSkipJumpAttack [17] is an attack which builds upon Boundary Attack. It uses a novel gradient direction estimation, which improves the number of queries needed to achieve a small enough perturbation. Experiments showed that after 10K queries, HopSkipJumpAttack achieves approximately ten times smaller perturbations in untargeted setting and three times smaller perturbations in the targeted setting as Boundary Attack.

1.2.7 Robustness

When comparing different models or attacks, it is vital to have some measure for robustness. The most common definition is how large a perturbation has to be to fool the model[18]. Perturbation is the distance between the original non-adversarial starting image and its adversarial counterpart. This distance is usually measured as L_2 or L_{inf} norm[18]. In general, the bigger the distance, the more visible the changes. A very robust model would require the perturbation and therefore the visible change to be substantial, while a non-robust model can be fooled with examples that do not contain any visible changes.

1.3 Related Work

This thesis only includes other works in this section, that have done a similar evaluation. Works that developed image classification methods or different attacking methods –even if they are used in this thesis– are not included in this section, as they are explained earlier.

Xiao et al.[19] successfully attacked a support vector machine using adversarial label noise. This kind of attack requires the training data to be manipulated so that the model is trained with already attacked data. This requirement is very limiting and can be only used in rare circumstances. Biggio et al.[20] did the most comprehensive analysis of adversarial attacks against support vector machines. They show that different kind of adversarial attacks can be performed against Support Vector Machines, but in most of the experiments, more knowledge is required than in a black-box setting. In the experiments where this knowledge is not available, surrogate models are used to estimate the gradients. Khrulkov et al.[21] demonstrated that attacks against HOG descriptors used with support vector machines are possible. They were able to utilize a white-box algorithm to change images with not visible perturbations so that the human in the image would not be detected.

To the best of the authors knowledge as of the writing of this thesis, no demonstrations of adversarial attack against the most popular and successful feature extractors (BoVW and Fisher Vector) exists. No adversarial attacks against logistic regression or random forest(decision tree) based models in the field of computer vision were found. The lack of such research makes this study by far the most comprehensive adversarial example study against classical computer vision pipelines.

2 Approach

This chapter discusses the main objectives of the study and how exactly it is constructed.

2.1 Architecture Design and Configurations

This thesis pays attention to not bias the evaluation for one method or another, to make this comparison as fair as possible. It also aims to create realistic and comparable scenarios that are relevant to real-world uses. The following describes in detail the choices made while designing the architecture and explains the reasons behind them.

2.1.1 Datasets

This study uses two specific datasets, because they provide certain advantages in comparison to other more-known datasets, such as having less classes and fewer images. This made it possible to train eight different model-feature extractor combinations.

The first one is the INRIA Person Dataset provided by [8]. This paper introduced the Histogram of Oriented Gradients for detecting if and where a person is in an image. As the paper is well known, and their results are impressive in detecting people, the same dataset is also used in this study. This dataset contains 1239 positive and 1218 negative samples, and photos can be in any resolution or aspect ratio.

The second dataset is the Imagenette dataset [22]. It is a subset of the well-known Imagenet [23], and contains the following ten classes: tench, English springer, cassette player, chain saw, church, French horn, garbage truck, gas pump, golf ball, parachute. It has 12894 images, which can again be any resolution or aspect ratio. Unlike the original Imagenet, it contains much fewer classes (10 instead of 1000) and also much fewer images, 12894 instead of many millions. Unlike the Coco [24] or Pascal Object Recognition Databases [25], which have many labels assigned to one image, Imagenette only includes one label per image. It is also not possible to use Cifar-10, [26], as it contains tiny images with a resolution of 32x32, which caused the feature extractors, especially those that are based on SIFT to perform very poorly.

2.1.2 Learning Methods and Feature Extractors

This study uses some of the most known classical machine learning methods, which are still very popular [27], and also a Convolutional Neural Network, to represent the current state of the art for image classification. Most of the best results in the Imagenet Challenge before 2012 were either with support vector machines or random forest classifiers, so it makes sense to have them in the study. Even though logistic regression classifiers have not seen much success in the imagenet challenge, they are also included, as they are one of the most used and simplest classifiers.

The reason for choosing the following feature extractors are also similar. SIFT is to this day one of the most used feature extractors. The bag of visual words approach[28], which uses SIFT features to extract image features, is used in almost every top entry in the imagenet challenge until 2012. HOG features are simpler than SIFT features and are used in many different tasks as well. HOG is chosen mainly because its combination with support vector machines gave excellent results in [8]. Finally, the Fisher Vector [10], is included, which is also SIFT based, as it got the second place when combined with a support vector machine in the 2010 Imagenet Challenge. The table 2.1 includes a list of every model-feature extractor combination we have used.

Learning Method	Feature Extractor
CNN	-
Random Forest Classifier	Bag of Visual Words
Random Forest Classifier	Histogram of Oriented Gradients
Random Forest Classifier	Fisher Vector
Logistic Regression	Bag of Visual Words
Logistic Regression	Histogram of Oriented Gradients
Logistic Regression	Fisher Vector
Support Vector Machine	Bag of Visual Words
Support Vector Machine	Histogram of Oriented Gradients
Support Vector Machine	Fisher Vector

Table 2.1 A list of the configurations used. The CNN did not need a feature extractor, as it is capable of extracting these features on its own from the pixel values.

2.1.3 Attack

As from the many combinations, only CNN provides gradients, it is not possible to use a white-box attack such as FGSM [3] to attack the models. The study uses the HopSkipJumpAttack [17], which is a more query efficient version of the better known Boundary Attack [16]. Both of these are black-box attacks that only need the predicted class labels to create an adversarial image. This is an advantage as it makes them more applicable for real-world applications, where also only the labels and not the inner working of classification algorithms are known.

Again for keeping the experiments as realistic as possible, the attacks are not tuned for the image classification algorithms, and the same attack is used for all of them.

2.2 Implementation Details

The images were resized to 160x160 because they had different aspect ratios and resolutions. In general, this study tries to avoid any method-specific optimizations, data augmentation, or extensive hyper-parameter tuning, to make this a fair and relevant comparison. Also, the relative accuracy and robustness between methods are more important in the context of this work than the exact accuracy of a particular method.

2.2.1 Bag of Visual Words

For extracting the SIFT features and training the Bag of Visual Words classifier, the well-known computer vision and machine learning library OpenCV[29] is used. OpenCV offers a ready to use SIFT feature extractor, which is used with its default settings. As the number of features is not set manually, the number of key-points and descriptors extracted per images is different for every image.

SIFT features(key-points and descriptors) are extracted for every image in the training set to train the classifier. Then the descriptors are used in conjunction with the BOWKMeansTrainer and BOWImgDescriptorExtractor classes, provided by OpenCV, to train the Bag of Visual Words Classifier. Vocabulary size is 2500.

To calculate the BoVW for an image, SIFT features for the given image are extracted and a vector representation is calculated using the trained classifier.

2.2.2 Histogram of Oriented Gradients

Scikit-image [30] is employed for extracting HOG features. The HOG feature extractor does not require any training, so to calculate the HOG features for an image, the HOG function from scikit-image with its default parameters is called. This function returns a vector representation for the image.

2.2.3 Fisher Vector

For extracting SIFT features, OpenCV is used once again, and for training the Gaussian mixture model for the Fisher Vector, the fishervector library [31] is used. Again the default settings for the SIFT extractor are used, but the number of features is set to 25, as the Fisher Vector needs the same amount of features for every image. 25 is selected as it was plenty enough to provide relevant information about an image but also few enough to be found in most images.

To train the classifier, first SIFT features for every image in the training set are extracted, and the 25 descriptors found for every image are concatenated into one so that there is one large descriptor for every image. If less than 25 features can be found for an image, a dummy descriptor with zero values is used instead. Using a dummy descriptor is a very undesirable outcome as it makes learning impossible, so it is essential not to choose a much bigger number than 25, as it causes much more images to have dummy descriptors. With 25, almost no image receives a dummy descriptor. An important step to significantly improve the performance is using PCA (Principal Component Analysis) to reduce the size of SIFT descriptors from 128 to 64, as described by Perronnin et al.[10]. Using PCA improves accuracy by about 10%. Lastly, the FisherVectorGMM with 256 kernels is used to fit the descriptors.

To calculate the Fisher Vector for an image, SIFT features are extracted for that image, PCA is used to reduce the descriptor size to 64, and the trained FisherVectorGMM is employed to get a vector representation.

2.2.4 Classical Machine Learning Classifier

To train the three classifiers, the LogisticRegression, LinearSVC, and RandomForestClassifier classes provided by the popular machine learning library scikit-learn [32] are used. As they provide good results with default parameters, and no improvements are observed by changing them, default parameters are used for all of them.

They are trained using the vector representations of the training images, calculated by the corresponding feature extractor.

2.2.5 Convolutional Neural Network

Keras [33] is used for implementing the CNN. To further help with repeatability, a pre-built and modern architecture, Inception-ResNet-V2 [34] is used. It is used without pre-trained weights and without changing the default parameters. The model employs the well-known Adam optimizer [35] and categorical cross-entropy loss. It is trained for 50 epochs, with early stopping to avoid overfitting, and the weights that performed the best in the cross-validation set are restored at the end. As CNN directly works with the pixel values, they were normalized to be between 0 and 1.

2.2.6 Adversarial Testing Set

The study uses an adversarial test set to measure the robustness of the methods. This set contains images from the testing set (not used to train the model), that are also classified correctly by the model. If an image is classified falsely to begin with, it makes no sense to attack that image. Therefore, from those images which are classified correctly, 100 images are randomly selected for every method.

One important thing to note here is that not the same 100 images are used for all models, but a random set is generated for each method. If it had to remain the same for all models, because most of the methods

have an accuracy around %50 in Imagenette and the adversarial testing set would need to contain pictures which are classified correctly by all methods, the 100 images in this set would not represent a random selection of images from the dataset but would be biased towards extra robust ones. This bias would be notably observed with the parachute class, as this is a class identified easily by all models, which would lead to the adversarial testing set to contain mostly images from this class, while not containing a single sample from some other classes, as they are harder for every method to classify correctly.

Therefore this study does not use the same set for every method, and the author argues that 100 images are plenty enough to avoid the evaluation results from getting affected by the difference in images for different methods.

2.2.7 Starting Point

HopSkipJumpAttack needs to start with an input that is classified falsely. For a given image to attack, a starting point image was chosen for the HopSkipJumpAttack that fulfills the following process:

- If the attack is targeted, the starting image has to belong to the target class and be classified as such. If the attack is untargeted, the starting image needs to be from another class than the attacked image, and be classified as such.
- For every image fulfilling the first criteria, we pick the image that is closest to the attacked image (norm is minimal).

Using this starting point, the final results of the attacking algorithm are improved substantially and also less queries are needed, compared to starting from a random noise initialization.

2.2.8 Attacking

The study uses the Foolbox Framework [36], which already comes with the HopSkipJumpAttack (BoundaryAttackPlusPlus). Foolbox offers a simple to use interface for attacking an image, and also providing a starting point. To better suite this study, foolbox is extended[1] to add better logging support, and support for stopping after reaching a certain number of queries. The second is necessary because the study is about attacking every model for 25000 queries to keep the comparison between different methods as fair as possible. These are the only changes made to the attack, so no changes are made to improve its performance specifically for our use case, again to keep it as fair as possible.

The HopSkipJumpAttack is run with batch size 1, to keep this comparison relevant for the real world as much as possible. Additional experiments show that batch size does not affect the results.

[1] https://github.com/ChantalMP/foolbox

3 Evaluation

This chapter includes the results of the experiments, with the corresponding discussion.

3.1 Results

Every classifier is attacked using HopSkipJumpAttack[17] for 100 images and 25000 queries. Mean and median perturbations are used as metrics, and perturbation is measured as the mean squared distance (MSE) between the original and the perturbed image. In Foolbox, this is calculated by squaring the L_2 norm, and then dividing it by the image size.

3.1.1 Model Accuracies

A list of all the accuracies can be seen in Table 3.1. As mentioned previously, these accuracies can be individually improved with different techniques, but no such techniques are used to keep the comparison fair between the different models.

Dataset	Learning Method	Feature Extractor	Accuracy
Imagenette	CNN	-	0,81
Imagenette	Random Forest Classifier	BoVW	0,42
Imagenette	Random Forest Classifier	Fisher Vector	0,37
Imagenette	Random Forest Classifier	HOG	0,43
Imagenette	Logistic Regression	BoVW	0,38
Imagenette	Logistic Regression	Fisher Vector	0,42
Imagenette	Logistic Regression	HOG	0,47
Imagenette	Support Vector Machine	BoVW	0,46
Imagenette	Support Vector Machine	Fisher Vector	0,47
Imagenette	Support Vector Machine	HOG	0,42
INRIA	CNN	-	0,98
INRIA	Random Forest Classifier	BoVW	0,85
INRIA	Random Forest Classifier	Fisher Vector	0,84
INRIA	Random Forest Classifier	HOG	0,90
INRIA	Logistic Regression	BoVW	0,69
INRIA	Logistic Regression	Fisher Vector	0,71
INRIA	Logistic Regression	HOG	0,90
INRIA	Support Vector Machine	BoVW	0,73
INRIA	Support Vector Machine	Fisher Vector	0,72
INRIA	Support Vector Machine	HOG	0,90

Table 3.1 Accuracy of different learning method, feature extractor and dataset combinations measured for testing set (data not seen while training). Important takeaways from this table are: In Imagenette, every image classification system other than CNN achieves less than 50% while CNN achieves 81%, and in INRIA while CNNs superiority remains, Random Forest models also perform better than the other two. It can also be seen that HOG has a better accuracy than the other two methods (this does not directly hold for Imagenette)

3.1.2 Mean and Median Perturbations

Table 3.2 includes the reached perturbation measured as MSE for 100 images after 25000 iterations for every dataset, learning method and feature extractor combination. The bigger the perturbation, the more visible the changes in the image. One important thing to note here is the big difference between mean and median in some cases. This indicates that there are some images which are attacked very poorly, causing the mean to be considerably higher than the median, while most images are attacked very successfully.

Dataset	Learning Method	Feature Extractor	Target Mode	Mean	Median
Imagenette	CNN	-	Targeted	2.871e-04	1.867e-04
Imagenette	CNN	-	Untargeted	1.099e-04	3.826e-05
Imagenette	Random Forest Classifier	BoVW	Targeted	6.917e-03	3.298e-03
Imagenette	Random Forest Classifier	BoVW	Untargeted	1.611e-03	6.990e-18
Imagenette	Random Forest Classifier	Fisher Vector	Targeted	6.576e-03	3.222e-03
Imagenette	Random Forest Classifier	Fisher Vector	Untargeted	5.058e-04	7.671e-19
Imagenette	Random Forest Classifier	HOG	Targeted	8.031e-03	8.826e-05
Imagenette	Random Forest Classifier	HOG	Untargeted	1.859e-05	2.073e-11
Imagenette	Logistic Regression	BoVW	Targeted	1.471e-02	1.187e-02
Imagenette	Logistic Regression	BoVW	Untargeted	3.452e-03	3.087e-05
Imagenette	Logistic Regression	Fisher Vector	Targeted	1.292e-02	9.970e-03
Imagenette	Logistic Regression	Fisher Vector	Untargeted	2.853e-03	4.020e-05
Imagenette	Logistic Regression	HOG	Targeted	2.769e-03	1.450e-04
Imagenette	Logistic Regression	HOG	Untargeted	3.138e-05	3.381e-07
Imagenette	Support Vector Machine	BoVW	Targeted	1.080e-02	7.543e-03
Imagenette	Support Vector Machine	BoVW	Untargeted	3.159e-03	1.346e-05
Imagenette	Support Vector Machine	Fisher Vector	Targeted	1.293e-02	1.083e-02
Imagenette	Support Vector Machine	Fisher Vector	Untargeted	3.039e-03	1.152e-04
Imagenette	Support Vector Machine	HOG	Targeted	1.357e-03	6.485e-05
Imagenette	Support Vector Machine	HOG	Untargeted	8.603e-06	2.797e-07
INRIA	CNN	-	Untargeted	1.952e-04	1.030e-04
INRIA	Random Forest Classifier	BoVW	Untargeted	3.436e-03	8.473e-05
INRIA	Random Forest Classifier	Fisher Vector	Untargeted	3.504e-03	1.329e-04
INRIA	Random Forest Classifier	HOG	Untargeted	5.063e-04	8.065e-06
INRIA	Logistic Regression	BoVW	Untargeted	7.748e-03	3.442e-03
INRIA	Logistic Regression	Fisher Vector	Untargeted	8.342e-03	6.477e-03
INRIA	Logistic Regression	HOG	Untargeted	3.799e-04	4.708e-05
INRIA	Support Vector Machine	BoVW	Untargeted	6.963e-03	3.954e-03
INRIA	Support Vector Machine	Fisher Vector	Untargeted	7.258e-03	3.477e-03
INRIA	Support Vector Machine	HOG	Untargeted	5.565e-04	5.841e-05

Table 3.2 Mean and Median perturbations measured as MSE against all dataset, learning method, and feature extractor combinations. Untargeted and targeted are also separated for Imagenette. In general it can be seen that Fisher Vector and BoVW are more robust than HOG or CNN based systems. One can also see that overall Logistic Regression and Support Vector Machine are more robust than CNN and Random Forest Classifier.

3.1.3 Visualization of Attacks

Mean perturbation is plotted with relation to the query number. If Imagenette is used as the dataset, separate plots for targeted and untargeted settings are included. In case for INRIA, there is only the untargeted attack as it is a binary classification task. Every plot includes a corresponding CNN based classifier as a baseline.

A combination which has a higher mean perturbation for a given query number is considered more robust against the adversarial attack, as the input image had to be changed more to fool the classifier. Figures 3.10 and 3.11 include some examples that visualize how a specific perturbation looks.

3.1.4 Interpretation

Inspecting the results from the INRIA dataset, it can be seen in the Figures 3.1a, 3.1b and 3.1c; that BoVW and Fisher Vector perform similarly and measurably better compared to HOG and CNN using all three machine learning models. It can also be observed that the difference between the feature extractors becomes – while still considerable – less when using random forest classifier.

To visualize how much the difference in the graphs means in reality, the same image is attacked using Logistic Regression as our model with three different feature extractors, and also CNN.

Fig 3.12 shows how successful the attacks are. It can be seen that the differences in values from the Graph 3.1b are also reflected in practice.

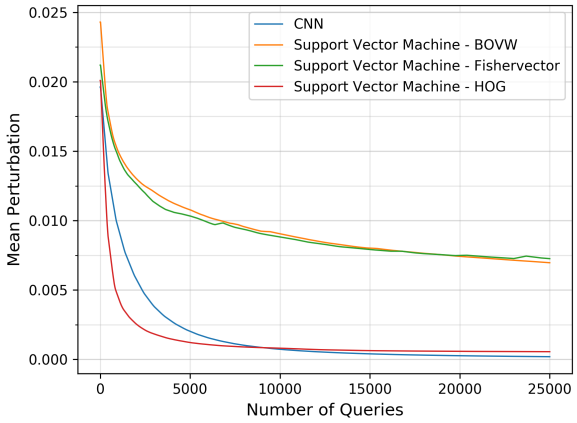
However, BoVW and Fisher Vector are also not foolproof. Depending on the attacked image and the chosen starting image, they can also be fooled without any visible perturbation in the adversarial image, as can be seen in Fig 3.11c. Increasing the pool of images from which a starting image can be chosen would reduce the robustness of these models further.

Looking at Table 3.1, it can be seen that in this case, more robust methods also have worse accuracies. SVM or Logistic Regression with BoVW or Fisher Vector, have an accuracy between 70% and 73%, where as HOG or CNN based classifiers are between 90% and 98%. Interestingly random forest performs much better with these features extractors, reaching 84-85% accuracy. However, random forest-based classifiers are also less robust than Logistic Regression or SVM based classifiers, as can be seen in Figures 3.1d and 3.1e.

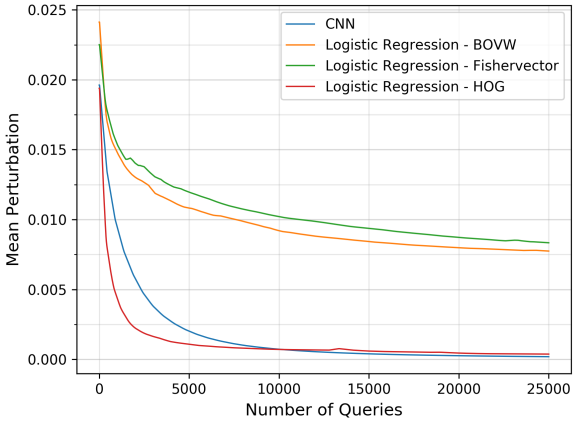
Interpreting the Imagenette dataset requires differentiating between two different attack types, targeted and untargeted. Looking at Fig 3.9, it can be seen that the robustness decreases a lot in the untargeted setting. One possible explanation for this is because of the low accuracy of the classifiers, which makes an untargeted attack easier, as nine other classes can be reached to fool the model. While it is certainly relevant that the classifiers can be fooled this way, the study also includes a targeted comparison to better compare the methods, as in the untargeted settings, there does not seem to be a difference between the different methods.

Looking at the Figures 3.3, 3.4 and 3.5, similarities to the INRIA dataset results can be seen in the targeted setting. As before, BoVW and Fisher Vector based methods appear more robust than the rest, and also random forest classifier is less robust than SVM and Logistic Regression. These results still hold in untargeted setting, though less pronounced than in the targeted setting. Interestingly when using HOG, random forest classifier is the most robust. This discrepancy is different from every other setting where random forest classifier is the least robust amongst the feature extractors.

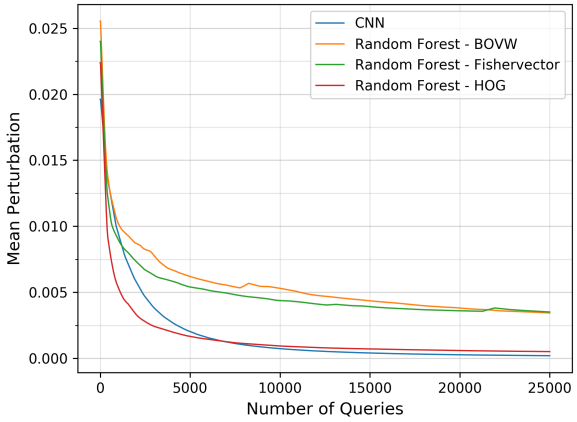
Overall, looking at the figures 3.2 and 3.9, it can be said that Logistic Regression with BoVW or Fisher Vector, followed by Support Vector Machine with BoVW or Fisher Vector appear more robust than current deep learning methods while also being much less accurate.



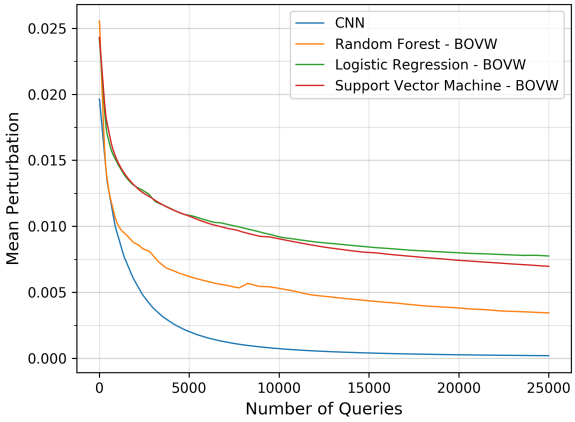
(a) Support Vector Machine classifiers with INRIA



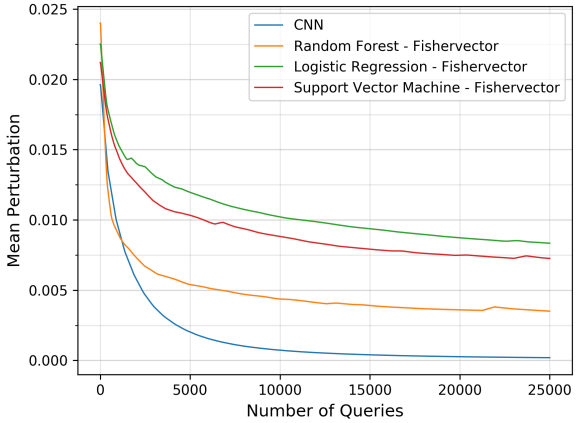
(b) Logistic Regression classifiers with INRIA



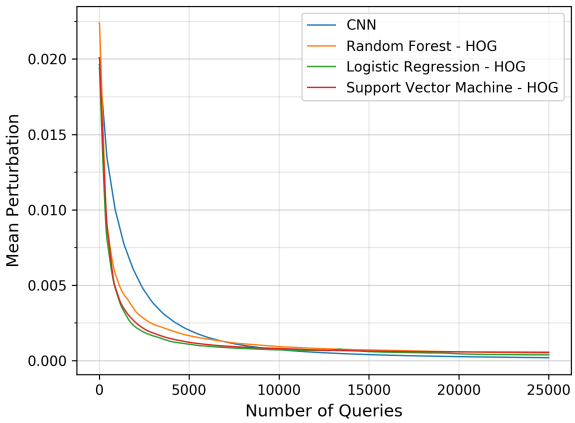
(c) Random Forest classifiers with INRIA



(d) BoVW based classifiers with INRIA



(e) Fisher Vector based classifiers with INRIA



(f) HOG based classifiers with INRIA

Figure 3.1 INRIA

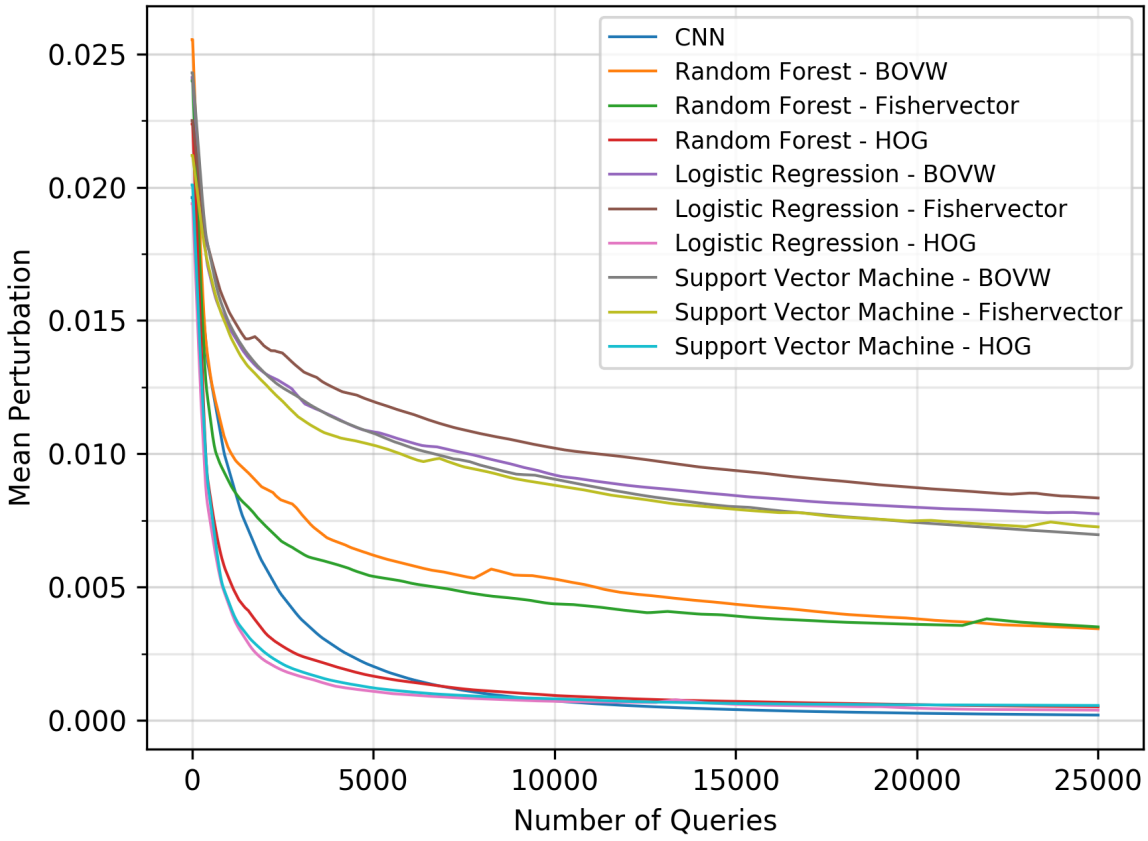
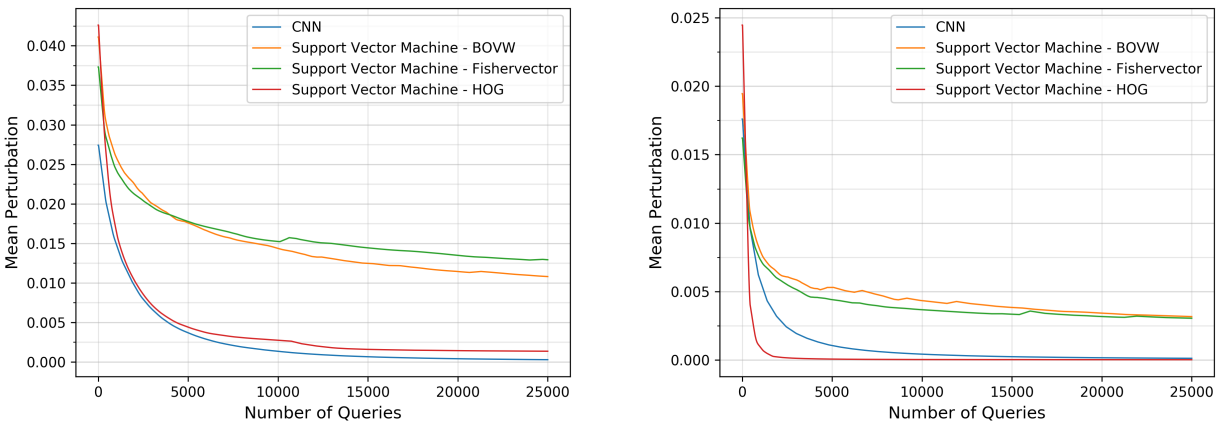


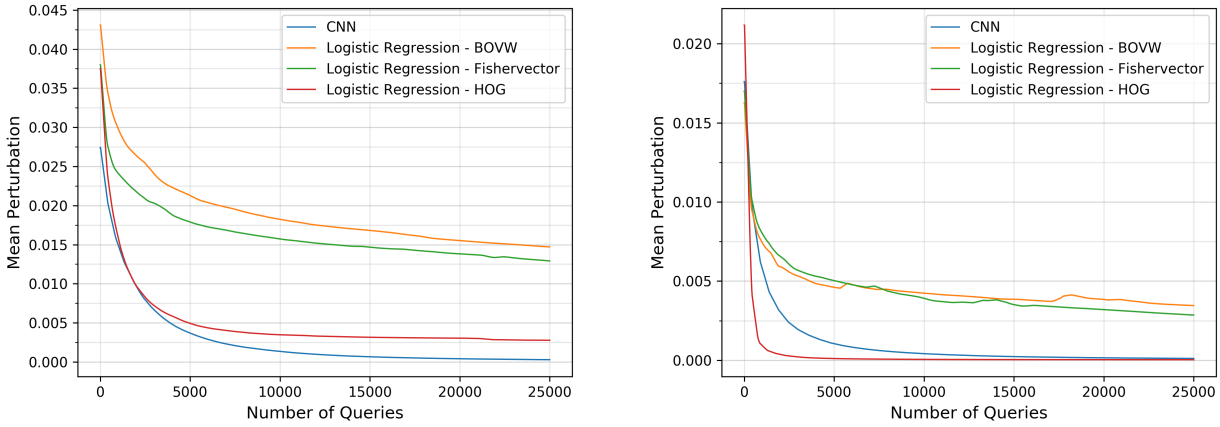
Figure 3.2 All classifiers with INRIA



(a) Targeted

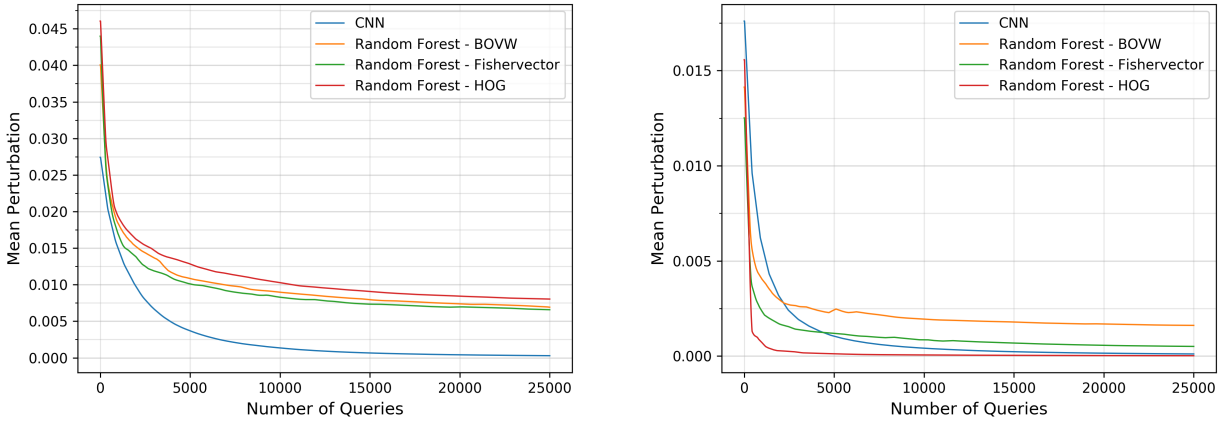
(b) Untargeted

Figure 3.3 Support Vector Machine classifiers with Imagenette



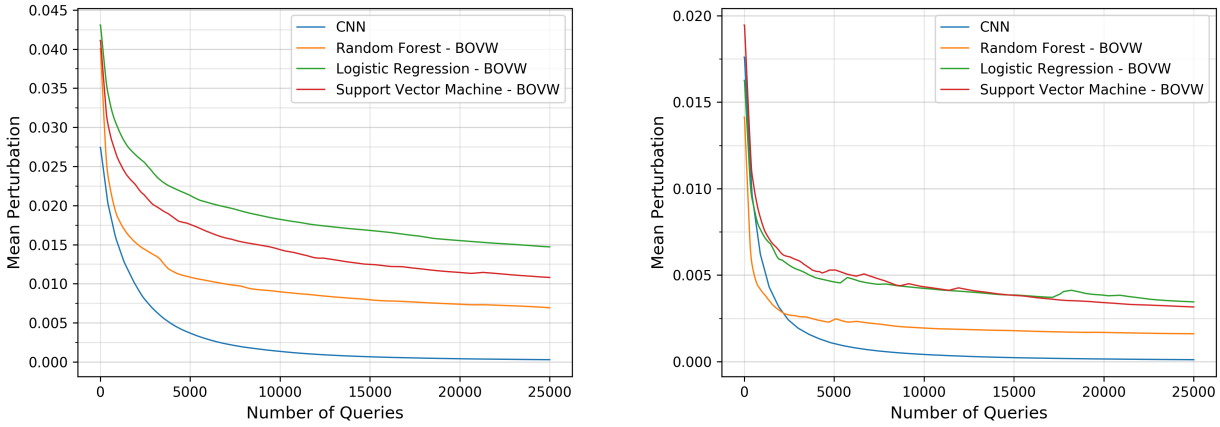
(a) Targeted **(b)** Untargeted

Figure 3.4 Logistic Regression classifiers with Imagenette



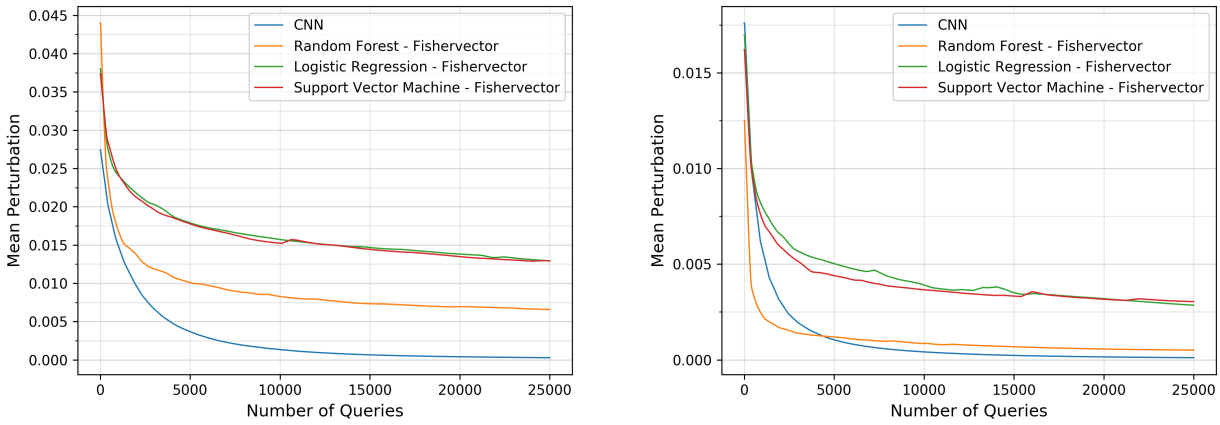
(a) Targeted **(b)** Untargeted

Figure 3.5 Random Forest classifiers with Imagenette



(a) Targeted **(b)** Untargeted

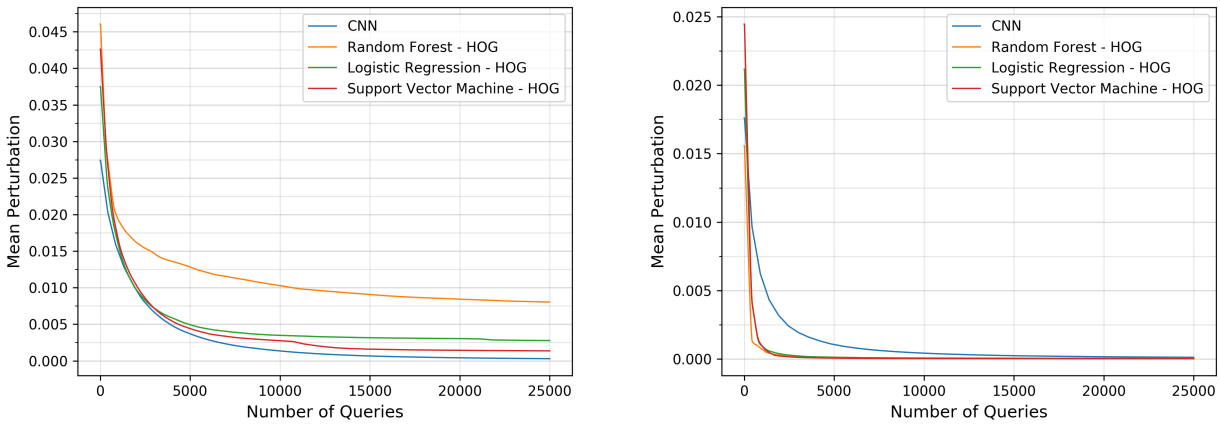
Figure 3.6 BoVW based classifiers with Imagenette



(a) Targeted

(b) Untargeted

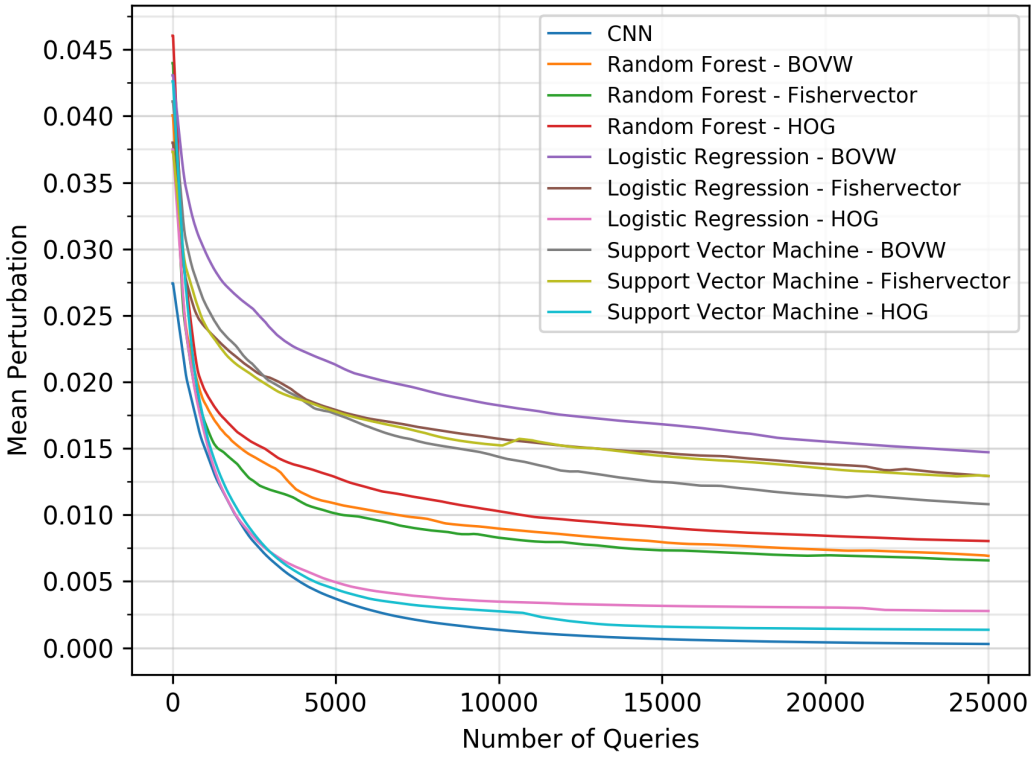
Figure 3.7 Fisher Vector based classifiers with Imagenette



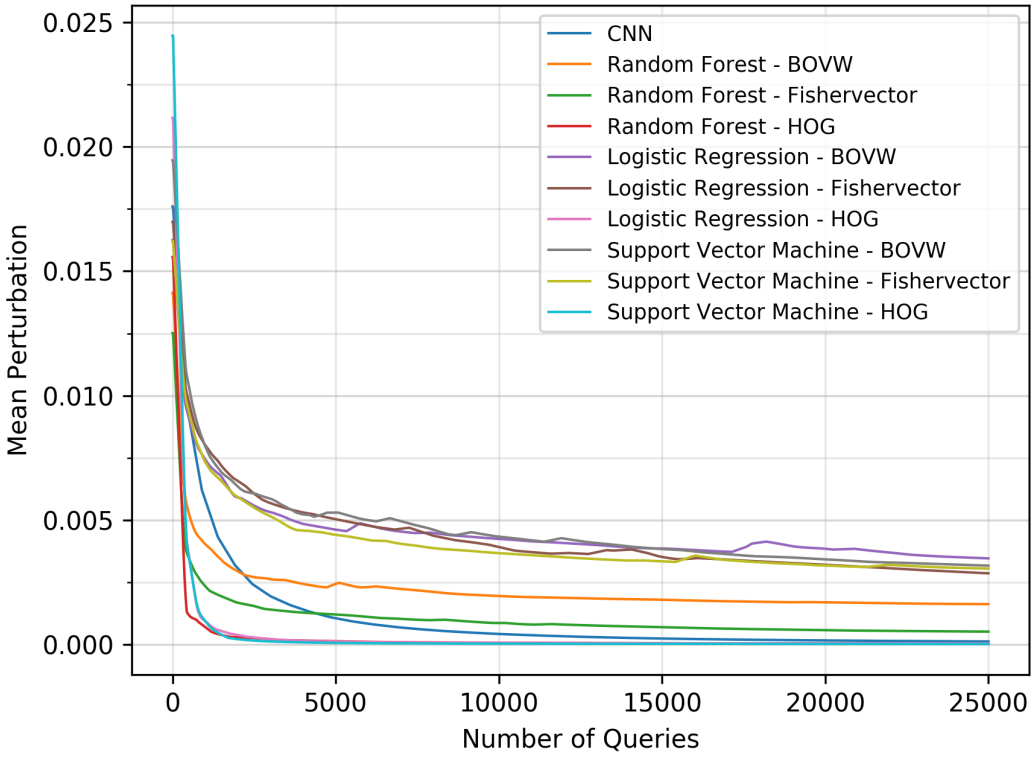
(a) Targeted

(b) Untargeted

Figure 3.8 HOG based classifiers with Imagenette

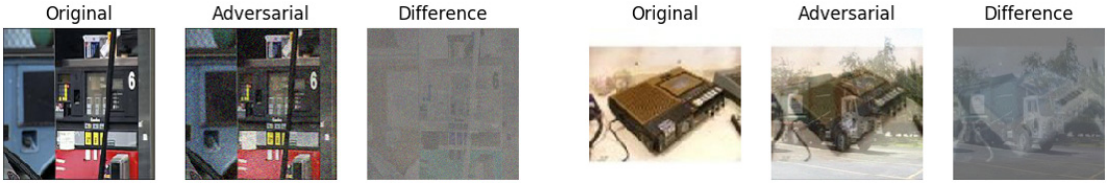


(a) Targeted



(b) Untargeted

Figure 3.9 All classifiers with Imagenette



(a) Perturbation in MSE: 0.008 **(b)** Perturbation in MSE: 0.021

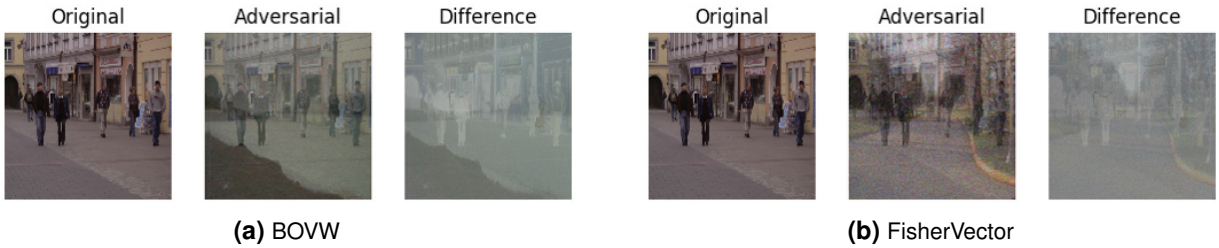
(c) Perturbation in MSE: 2.89 x 10^{-18}

Figure 3.10 Visualization of original images in Imagenette with their adversarial counterparts after attacks against SVM with BoVW features, and the difference between the two.

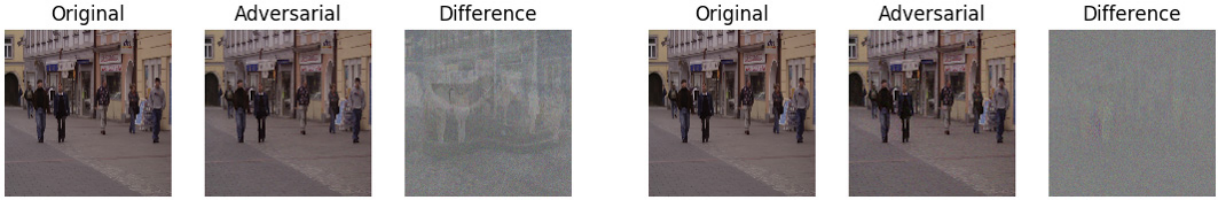
(a) Perturbation in MSE: 0.004 **(b)** Perturbation in MSE: 0.005

(c) Perturbation in MSE: 3.12 x 10^{-16}

Figure 3.11 Visualization of original images in INRIA with their adversarial counterparts after attacks against SVM with BoVW features, and the difference between the two.



(a) BOVW **(b)** FisherVector



(c) HOG **(d)** CNN

Figure 3.12 Visualization of attacks against the same image using different feature extractors in INRIA.

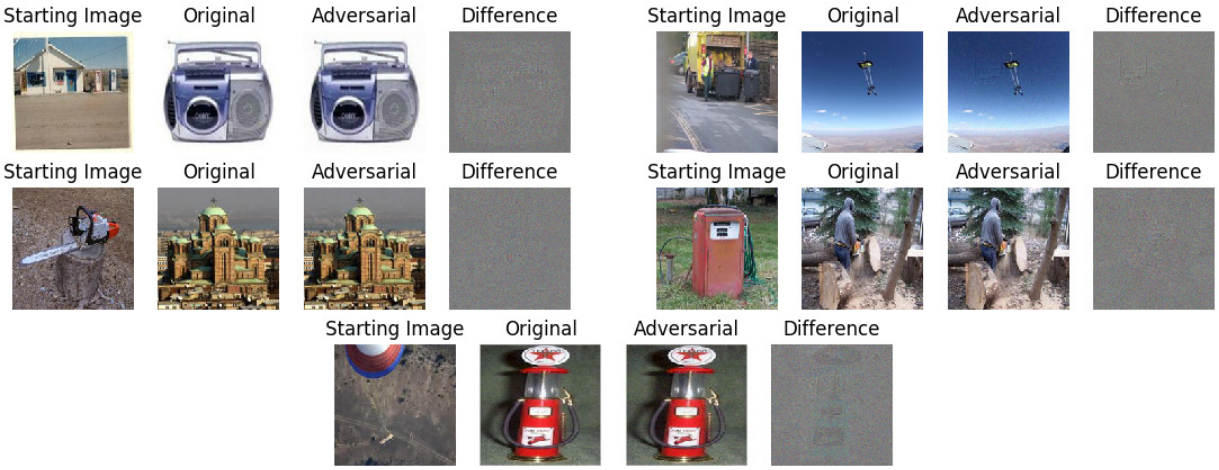


Figure 3.13 Visualization of 5 randomly selected images before and after a targeted attack in Imagenette with CNN.

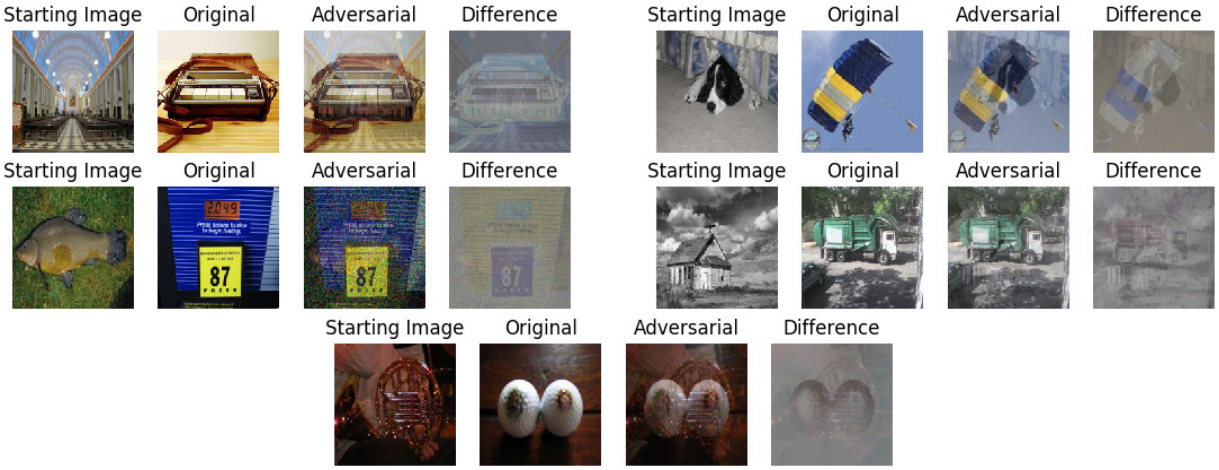


Figure 3.14 Visualization of 5 randomly selected images before and after a targeted attack in Imagenette with BoVW.

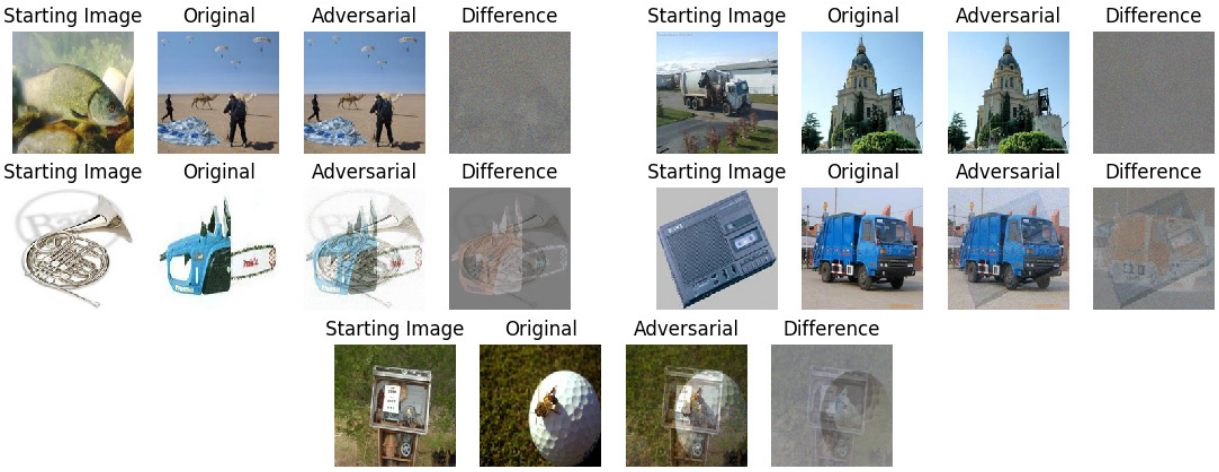


Figure 3.15 Visualization of 5 randomly selected images before and after a targeted attack in Imagenette with Fisher Vector

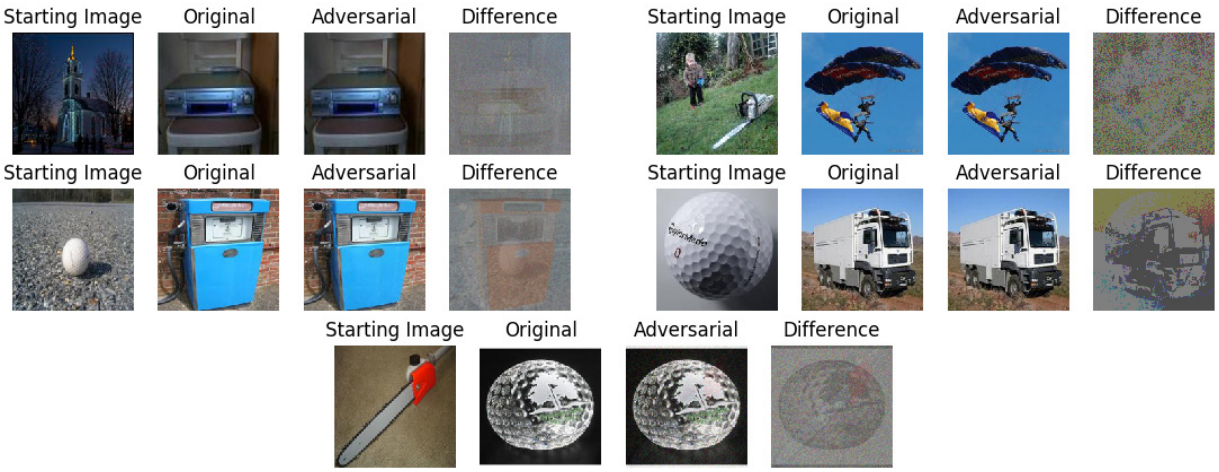


Figure 3.16 Visualization of 5 randomly selected 5 images before and after a targeted attack in Imagenette with HOG

3.2 Discussion

As it can also be seen in Fig 1.1, random forest classifiers appear to have more non-linear boundaries compared to other Support Vector Machines and Logistic Regression classifiers. It is still an active research if linearity and robustness are correlated, and this is out of scope for this thesis. Overall, the difference between classifiers only has a minor effect. The more prominent effect is the method of extracting features.

One interesting thing to note is that HOG based feature extractor in a targeted setting is most robust when used in conjunction with a random forest classifier (see Fig 3.8), which differs from the other feature extractors which are the least robust when used with a random forest classifier. The reasons why this might be the case is out of scope for this study and can be further investigated in a different study.

It must be emphasized that this study shows an upper bound for robustness, as different methods can be used which are even more successful at attacking, which would further lower the robustness.

As a final note, optimizing specific models to improve their accuracy might lead to better or worse results in terms of robustness. The goal of this thesis is not to argue that method X is always better than method Y, but rather to compare them as objectively as possible and guide researchers when deciding for a particular approach. Use of two datasets makes these results relevant in practice as well, but the results might not hold if the input format changes drastically.

4 Future Work

Future work could include employing different datasets such as Cifar-10 to find out if these results translate well to tiny images, or Imagenet to measure the effects of having 1000 instead of 10 classes. Other classifying methods such as nearest neighbors, or other feature extractors can also be compared to generalize the conclusions of this thesis further. Another future work can be to tune specific methods and see how their accuracy and robustness change. It is also possible to increase the number of images, from which a starting image is chosen, as using a broader set of images there can result in starting points much closer to the original image, thus making it simpler to attack.

As used features are the primary indicator towards robustness, a deep neural network other than CNN can be used, which unlike CNN does not extract its features, but relies on a handcrafted feature extractor. While not allowing the network to extract its own features would likely hurt its performance, it might provide a good tradeoff between accuracy and robustness.

5 Conclusion

In this study, some of the most used machine learning methods and feature extractors in the field of computer vision were attacked using adversarial examples. In comparison to deep learning based approaches such as CNNs, these methods had worse accuracies, but in some settings were more robust than CNNs. Fisher Vector and BoVW feature extractors when combined with Logistic Regression or Support Vector Machine based classifiers were the most robust amongst all combinations tested. Nonetheless, using a black-box attack, it was shown that none of the methods are entirely robust against adversarial attacks, because for every combination tested, there were adversarial images generated, which were indistinguishable from non-adversarial images. Having a bigger pool of reference images could further decrease their robustness. Therefore it can be concluded that the problem of adversarial attacks is not limited to deep learning based approaches, but affects many different image classification methods. Because of this it would be a wrong assumption to think adversarial examples did not pose a threat before deep learning.

The societal impact of this conclusion is that many image classification algorithms used in safety-critical system (such as autonomous driving, face recognition systems and robotics) have an inherent risk of being fooled, regardless of the algorithm of choice. As more and more of these technologies surround our lives, more research in the area of adversarial examples is needed to make sure the safety-critical systems can not be exploited easily, as it would pose a great danger to our society.

Bibliography

[1] Z. Zheng and P. Hong, "Robust detection of adversarial attacks by modeling the intrinsic properties of deep neural networks," in *Advances in Neural Information Processing Systems 31*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, Eds. Curran Associates, Inc., 2018, pp. 7913–7922. [Online]. Available: http://papers.nips.cc/paper/8016-robust-detection-of-adversarial-attacks-by-modeling-the-intrinsic-properties-of-deep-neural-networks.pdf

[2] S. Gu and L. Rigazio, "Towards deep neural network architectures robust to adversarial examples," 12 2014.

[3] I. Goodfellow, J. Shlens, and C. Szegedy, "Explaining and harnessing adversarial examples," in *International Conference on Learning Representations*, 2015. [Online]. Available: http://arxiv.org/abs/1412.6572

[4] ryancompton. (2019) Supervised learning superstitions cheat sheet. [Online]. Available: http://ryancompton.net/assets/ml_cheat_sheet/supervised_learning.html

[5] ml cheatsheet. (2019) Logistic regression. [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/logistic_regression.html

[6] A. Noel Bambrick. (2016) Support vector machines: A simple explanation. [Online]. Available: https://www.kdnuggets.com/2016/07/support-vector-machines-simple-explanation.html

[7] sklearn. Classifier comparison. [Online]. Available: https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html

[8] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05) - Volume 1 - Volume 01*, ser. CVPR '05. Washington, DC, USA: IEEE Computer Society, 2005, pp. 886–893. [Online]. Available: http://dx.doi.org/10.1109/CVPR.2005.177

[9] G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray, "Visual categorization with bags of keypoints," in *In Workshop on Statistical Learning in Computer Vision, ECCV*, 2004, pp. 1–22.

[10] F. Perronnin, J. Sánchez, and T. Mensink, "Improving the fisher kernel for large-scale image classification," in *Proceedings of the 11th European Conference on Computer Vision: Part IV*, ser. ECCV'10. Berlin, Heidelberg: Springer-Verlag, 2010, pp. 143–156. [Online]. Available: http://dl.acm.org/citation.cfm?id=1888089.1888101

[11] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004. [Online]. Available: https://doi.org/10.1023/B:VISI.0000029664.99615.94

[12] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds. Curran Associates, Inc., 2012, pp. 1097–1105. [Online]. Available: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf

[13] M. D. Zeiler and R. Fergus, “Visualizing and understanding convolutional networks.” *CoRR*, vol. abs/1311.2901, 2013. [Online]. Available: http://dblp.uni-trier.de/db/journals/corr/corr1311.html#ZeilerF13

[14] C. Szegedy, W. Zaremba, I. Sutskever, J. Bruna, D. Erhan, I. J. Goodfellow, and R. Fergus, “Intriguing properties of neural networks,” *CoRR*, vol. abs/1312.6199, 2014.

[15] I. J. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *arXiv preprint arXiv:1412.6572*, 2014.

[16] W. B. *, J. R. *, and M. Bethge, “Decision-based adversarial attacks: Reliable attacks against black-box machine learning models,” in *International Conference on Learning Representations*, 2018. [Online]. Available: https://openreview.net/forum?id=SyZI0GWCZ

[17] J. Chen and M. I. Jordan, “Boundary attack++: Query-efficient decision-based adversarial attack,” *CoRR*, vol. abs/1904.02144, 2019.

[18] N. Carlini and D. A. Wagner, “Towards evaluating the robustness of neural networks,” *CoRR*, vol. abs/1608.04644, 2016. [Online]. Available: http://arxiv.org/abs/1608.04644

[19] H. Xiao, B. Biggio, B. Nelson, H. Xiao, C. Eckert, and F. Roli, “Support vector machines under adversarial label contamination,” *Neurocomput.*, vol. 160, no. C, pp. 53–62, Jul. 2015. [Online]. Available: http://dx.doi.org/10.1016/j.neucom.2014.08.081

[20] B. Biggio, I. Corona, B. Nelson, B. I. P. Rubinstein, D. Maiorca, G. Fumera, G. Giacinto, and F. Roli, “Security evaluation of support vector machines in adversarial environments,” *CoRR*, vol. abs/1401.7727, 2014.

[21] V. Khrulkov and I. Oseledets, “Art of singular vectors and universal adversarial perturbations,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[22] FastAi. (2019) Imagenette. [Online]. Available: https://github.com/fastai/imagenette

[23] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, “ImageNet: A Large-Scale Hierarchical Image Database,” in *CVPR09*, 2009.

[24] T. Lin, M. Maire, S. J. Belongie, L. D. Bourdev, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, “Microsoft COCO: common objects in context,” *CoRR*, vol. abs/1405.0312, 2014. [Online]. Available: http://arxiv.org/abs/1405.0312

[25] V. O. C. Challenge. (2005) The pascal object recognition database collection. [Online]. Available: www.pascal-network.org/challenges/VOC

[26] A. Krizhevsky, “Learning multiple layers of features from tiny images,” Tech. Rep., 2009.

[27] K. Hazelwood, S. Bird, D. Brooks, S. Chintala, U. Diril, D. Dzhulgakov, M. Fawzy, B. Jia, Y. Jia, A. Kalro, J. Law, K. Lee, J. Lu, P. Noordhuis, M. Smelyanskiy, L. Xiong, and X. Wang, “Applied machine learning at facebook: A datacenter infrastructure perspective,” 02 2018, pp. 620–629.

[28] M. Marszalek and C. Schmid, “Constructing category hierarchies for visual recognition,” in *Proceedings of the 10th European Conference on Computer Vision: Part IV*, ser. ECCV ’08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 479–491. [Online]. Available: https://doi.org/10.1007/978-3-540-88693-8_35

[29] G. Bradski, “The OpenCV Library,” *Dr. Dobb’s Journal of Software Tools*, 2000.

[30] S. van der Walt, J. L. Schönberger, J. Nunez-Iglesias, F. Boulogne, J. D. Warner, N. Yager, E. Gouillart, T. Yu, and the scikit-image contributors, “scikit-image: image processing in Python,” *PeerJ*, vol. 2, p. e453, 6 2014. [Online]. Available: https://doi.org/10.7717/peerj.453

[31] jonasrothfuss. (2018) Improved fisher vector implementation. [Online]. Available: https://github.com/jonasrothfuss/fishervector

[32] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “Scikit-learn: Machine learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[33] F. Chollet *et al.*, “Keras,” https://keras.io, 2015.

[34] C. Szegedy, S. Ioffe, and V. Vanhoucke, “Inception-v4, inception-resnet and the impact of residual connections on learning,” *CoRR*, vol. abs/1602.07261, 2016.

[35] D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” *CoRR*, vol. abs/1412.6980, 2015.

[36] J. Rauber, W. Brendel, and M. Bethge, “Foolbox: A python toolbox to benchmark the robustness of machine learning models,” *arXiv preprint arXiv:1707.04131*, 2017. [Online]. Available: http://arxiv.org/abs/1707.04131