Санкт-Петербургский государственный университет

Фундаментальная информатика и информационные технологии

Киндулов Михаил Львович

Модели машинного обучения, устойчивые к состязательным атакам

Дипломная работа

Научный руководитель:
д-р ф.-м. н., профессор СПбГУ Крылатов А. Ю.

Рецензент:
к. т. н., доцент СПбГУ Блеканов И. С.

Санкт-Петербург
2020

Оглавление

Введение

В современном мире модели машинного обучения используются повсеместно: при распознавании речи, жестов, поиске объектов на изображении, прогнозировании временных рядов, медицинской и технической диагностике, в биоинформатике, для высокочастотной торговли, обнаружения фрода, кредитного скоринга и во многих других сферах.

К сожалению, многие модели машинного обучения чувствительны к некорректным входным данным. К примеру, оригинальный метод опорных векторов очень чувствителен к шуму, а глубокая сверточная нейронная сеть может быть обманута специальными изображениями. Такие изображения создаются атакующими сетями. При недостаточно большом размере набора данных для обучения деревья решений часто имеют области признакового пространства, в котором объекты могут неправильно классифицироваться (рис. 1).

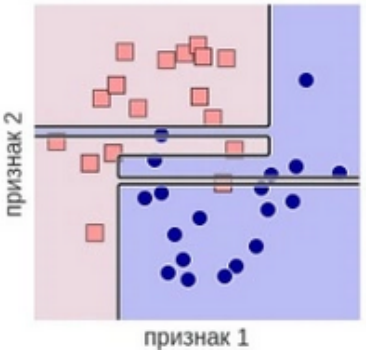

Рис. 1: Бинарное решающее дерево, построенное на двух признаках и его области принадлежности к классам.

На некоторые модели машинного обучения намеренно производятся атаки. Они могут быть использованы для защиты от автоматического ввода капчи, сокрытия от распознавания лиц, обхода антиспам-систем или для некорректного распознавания дорожных знаков беспилотными автомобилями. Именно такие методы, направленные на введение в заблуждение моделей машинного обучения, и называются **состязательными атаками**. Для защиты от подобных атак необходимо использовать специальные модификации моделей машинного обучения.

1. Постановка задачи

Целью данной работы является построение алгоритма, позволяющего обнаруживать атакованные объекты. Для ее достижения были поставлены следующие задачи:

- Изучить существующие методы состязательных атак.
- Изучить существующие методы защиты от атак, проанализировать их плюсы и минусы.
- Разработать собственный метод защиты от атак.
- Провести тестирование собственного метода.

2. Существующие типы и методы атак

Для того, чтобы рассмотреть какие методы существуют для защиты от состязательных атак, необходимо сначала рассмотреть различные категории атак, их характеристики и цели.

2.1. Атаки по типу цели

2.1.1. Атаки на изображения

Целью состязательных атак на изображения является создание нового изображения путем небольшого изменения исходного. Изменение происходит таким образом, чтобы максимизировать функционал ошибки модели машинного обучения. Созданное изображение называется атакованным.

Например, атакующая модель может минимизировать такой функционал: $\sum\limits_{x \in X} \left(-E(f(\hat{x}, y)) + \alpha \sum\limits_{i=0}^{K} \sum\limits_{j=0}^{M} (x_{i,j} - \hat{x_{i,j}})^2 \right)$ где $E(\cdot, \cdot)$ обозначает ошибку атакуемой модели (например, 1 - accuracy score для задачи классификации), x – исходное изображение размера $K \times M$, $\hat{x}$ – соответствующее ему атакованное изображение, X – множество изображений, на которые производится атака, $f(\cdot)$ – атакуемая модель, α – параметр, отвечающий за приоритетность малости l_2 нормы между атакованным и исходным изображениями.

Атаки на изображения являются одними из самых распространенных. Наиболее известным видом применения является защита от автоматического ввода капчи.

2.1.2. Атаки на временные ряды

Целью атак на временные ряды является изменение наименьшего количества объектов в любом из промежутков времени заданной длины, максимизируя при этом ошибку модели. К этому типу атак также относятся те, которые увеличивают ошибку атакуемой модели, минимизируя суммарные изменения признаков объектов за определенный

промежуток времени. Такие атаки используются в высокочастотной торговле и алгоритмическом трейдинге.

2.1.3. Атаки на агента в задачах обучения подкреплением

Как правило, под атаками на агента подразумеваются атаки при которых в среду взаимодействия добавляется атакующий агент. Такой агент пытается уменьшить награду атакуемого агента.

2.1.4. Атаки на аудио

Атаки на аудио в данный момент представлены, в основном, в виде атак на распознавание речи. Одним из простейших примеров является атака на VAD (Voice Activity Detector) – детектор речи в аудиопотоке. Обычно такой компонент присутствует в моделях распознавания речи. Атаки на распознавание речи распространены в голосовой капче.

2.1.5. Атаки на обработку естественного языка

Среди атак на обработку естественного языка можно выделить следующие:

- атаки на машинный перевод;
- атаки на частеречную разметку;
- атаки на классификацию текстов;
- атаки на анализ тональности текста.

2.2. Атаки по типу применения

2.2.1. Атаки на “белый ящик”

Об атаках на “белый ящик” говорят тогда, когда известна атакуемая модель и ее параметры. Такие атаки являются наиболее эффективными, при этом крайне редко встречаются в реальной жизни, ведь для

них у атакующего должна быть обученная модель машинного обучения, на которую будет производиться атака. Часто атаки на “черный ящик” могут быть сведены к атакам на “белый ящик”.

К методам атаки типа “белый ящик” относят:

- Атаку Биджио [3] – атака на метод опорных векторов, в которой производится минимизация $g(x) = \langle w, x \rangle$ вместе с минимизацией L_1-нормы. Здесь $f(x) = g(x) + b$ - бинарный классификатор, действующий по принципу $f(x) > 0 \Rightarrow x \in class_1$, иначе $class_0$.

- L-BFGS атаку Сегеды [12]– первая проведенная атака на сверточные нейронные сети. Эта атака минимизирует квадрат расстояния между исходным изображением и атакованным одновременно с минимизацией ошибки атакуемой модели до класса t. Оптимизируемый функционал выглядит так: $minimize\ c\,\|x' - x\| + \mathcal{L}(x', t, \theta)$, где коэффициент c подбирается таким образом, чтобы сеть продолжала классифицировать объект x' как класс t. Коэффициент c подбирается оптимизатором второго порядка L-BFGS, откуда и произошло название метода.

- FGSM [4] – наиболее известный тип атаки. Его суть заключается в том, чтобы сделать небольшой шаг в сторону знака градиента ошибки атакуемой модели, так, чтобы $\|\cdot\|_\infty$-норма разности исходного и атакуемого изображений была ниже определенного порога ϵ.

- Deep Fool [4] – метод, который ищет такие изменения примера x, которые позволят с небольшой $\|\cdot\|_\infty$-нормой разницы исходного и атакованного изображений перестать правильно его классифицировать. Алгоритм находит ближайшую гиперплоскость границы двух классов и сдвигает исходное изображение x в сторону вектора, ортогонального такой гиперплоскости. Как показано в [9], 90% тестовых изображений набора данных MNIST можно “превратить” в другой класс изменив изображение всего на 0.1 по $\|\cdot\|_\infty$-норме.

- BIM/PGD [6] – это итеративный метод поиска атакованного изображения в ϵ-шаре от исходного изображения x. Метод осуществляет проекцию найденного на очередной итерации атакованного изображения в сторону максимизации ошибки, откуда он и получил свое название (**projected** gradient descent).

- C&W [2] – Атака Карлини и Вагнера появилась как решение против защиты от состязательных атак типа L-BFGS и FGSM. Вместо минимизации функционала, предложенного в методе L-BFGS Сегеды, предлагается использовать минимизацию зазора вероятностей всех классов с целевым. Проведение подобной оптимизации позволяет найти устойчивые к защите от состязательных атак изображения.

- Ground Truth атаку [11] – использование SMT-solver-а для нахождения атакованного изображения в минимальном ϵ-шаре с итеративным уменьшением ϵ.

- Universal атаку – нахождение общего для большинства изображений набора данных возмущения δ с p-нормой ограниченной ϵ.

- Атаку искривления пространства - небольшая трансформация локальных областей изображения.

- Атаку на механизм внимания.

- Другие типы атак.

2.2.2. Атаки на “черный ящик”

Этот тип атак подразумевает, что атакующий может вычислить для любого примера x ответ модели y, при этом модель и ее параметры неизвестны.

Некоторые из атак этого типа можно свести к предыдущим: достаточно разметить ответами модели некоторое количество данных, аугментировать их, обучить на их основе свою модель (желательно, мак-

симально близкую по структуре к атакуемой) и использовать ее для проведения атаки типа “белый ящик”.

К атакам типа “черный ящик” можно отнести несколько больших групп:

- Создающие модель-“ученика” и производящие атаку на нее. Является самым простым и, поэтому, наиболее популярным методом среди атак типа “черный ящик”.

- Использующие точечный градиент. В этом случае атака производится напрямую на “черный ящик”, итеративно подбираются изменения в зависимости от ответа модели.

- Использующие эволюционные алгоритмы и(или) байесовские методы для поиска более показательных для точечного градиента примеров.

2.2.3. Data poisoning

Добавление атакованных примеров в набор данных, которые не позволяют обучить модель высокого качества.

- Добавление малого количества примеров, максимизирующих ошибку на всем наборе данных. На параметры таких примеров не накладываются ограничения.

- Добавление измененных изображений из тестового набора данных с целью изменить предсказание модели на них.

3. Существующие методы защиты от атак

Существующие методы защиты от атак можно разделить на три больших группы: маскировка градиента, повторное обучение для поиска более стойких к атакам параметров модели и добавление к модели механизма обнаружения атак.

3.1. Маскировка градиента

- Защитная дистилляция модели.
- Рандомизация градиентов.
- Затухание градиента.

Идея защитной дистилляции исходит из простой дистилляции, впервые представленной в [5]. Так же как и в [5] защитная дистилляция выполняет свою цель, обучая нейронную сеть меньшего размера на логитах основной (дистиллируемой) модели. При этом в обучении модели-“ученика” используется softmax с “температурой”:

$$softmax(x, T)_i = \frac{e^{x_i/T}}{\sum_j e^{x_j/T}}, \text{ при } T >> 1.$$

Во время тестирования используется модель-“ученик” с температурой 1, таким образом получают менее разнообразные ответы с большей уверенностью, которые при этом являются более стойкими к атакам.

Рандомизация градиента – это метод защиты от атак на “черный ящик”, при котором обучается несколько моделей. Во время оценки, новые объекты обрабатываются случайно выбранной моделью, таким образом осложняя создание модели-“ученика”. Рандомизация градиента – это простой в реализации метод, но требует больших вычислительных затрат при обучении, кроме того для высокой защитной эффективности необходимо выбирать разные модели.

Как известно, достаточно глубокие нейронные сети часто уменьшают значение градиента от слоя к слою (к примеру, создание архитектуры ResNet связано с этим фактом). Защита посредством затухания градиента использует это, делая нейронную сеть более глубокой: новый объект до попадания в сеть проходит через генеративную состязательную сеть. Таким образом, эксплуатировать нейронную сеть становится сложнее.

Как показано в [2] все представленные выше методы не являются надежными и могут быть эксплуатированы. В [1] представлены конкретные виды атак для обхода подобных стратегий.

3.2. Повторное обучение модели

- Регуляризация на малость разницы между x и $x + \delta$.
- Добавление атакованных изображений к обучающей выборке.
- Нахождение минимального расстояния до атакованного изображения для каждого из примеров и регуляризация по нему.

Основной сутью всех описанных способов является обучение моделей с использованием атак: это может быть как добавление в обучающую выборку атакованных примеров, так и активное изменение параметров модели во время обучения таким образом, чтобы примеры с минимальными возмущениями имели тот же класс что и исходные.

Минусом такого подхода, очевидно, является изменение исходной модели, что может привести к снижению ее точности. Также это заставляет модель подстраиваться к атакам именно того типа, который использовался при обучении. Кроме того, в некоторых реальных задачах, например, в детекции злокачественных опухолей на медицинских снимках, обучение модели – это очень долгий и дорогой процесс из-за объемов данных.

3.3. Добавление механизма детекции атакованных примеров

Как и следует из названия, в данном случае добавляется бинарный классификатор для разделения на обычные и атакованные изображения. Методы этого типа можно считать неким продолжением идеи встраивания confidence-ветки в модель машинного обучения. Confidence-ветвь нужна для того, чтобы модель могла сообщать уровень своей уверенности в предсказании. При этом модель будет получать штраф за низкую “уверенность” предсказания, но такой штраф может оказаться ощутимо меньше, чем штраф за уверенное предсказание ложного класса.

Методы этого типа имеют широкое практическое применение, так как не требуют изменения основной модели и просты в реализации, но также имеют некоторые недостатки: необходимость запускать несколько моделей на одном объекте, возможность обнаружения множества типов различных атак.

4. Разработка собственного метода защиты от атак

В этой секции будет рассмотрен способ защиты от состязательных атак (исключая data poisoning) на модели машинного обучения, обучаемые градиентным спуском. Для иллюстрации работы метода использованы нейронная сеть ResNet-18, атаки Deep Fool, FSGM и набор данных CIFAR-10. Представленные ниже методы могут быть перенесены на любую обучаемую с помощью градиентного спуска модель с дважды дифференцируемой функцией ошибки.

Выберем случайным образом небольшое количество изображений из тестового набора набора данных, обозначим их как x, обозначим также через x_i i-ое изображение. Обозначим через $\hat{x}_{df}$ изображение, атакованное с помощью Deep Fool, и через $\hat{x}_{fsgm}$ – изображение, атакованное с помощью FSGM. Пусть также функция $N(x, \mu, \sigma) = \mathcal{N}(\mu, \sigma) + x$ добавляет на изображение гауссовский шум с параметрами μ, σ.

Рассмотрим логиты модели для каждого из полученных изображений (рис. 2).

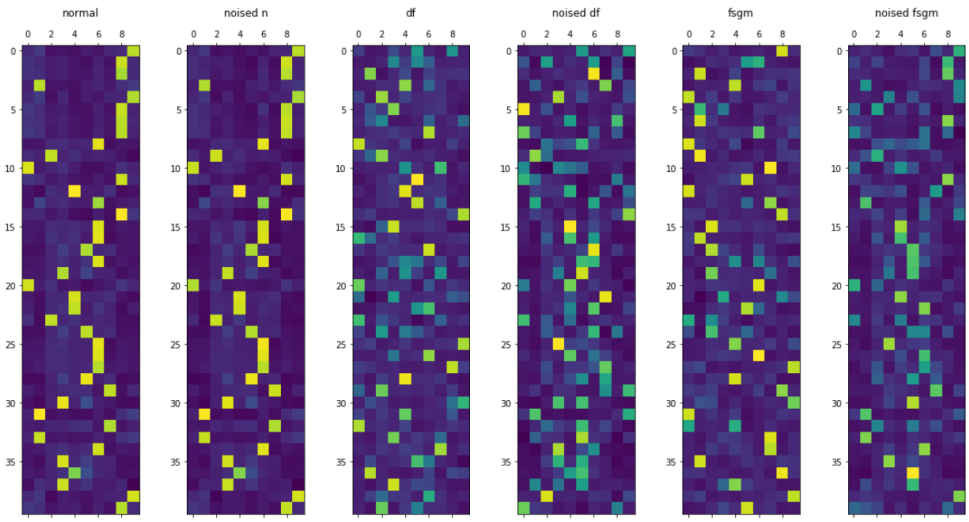

Рис. 2: Логиты модели для каждого из полученных изображений. Слева направо: для обычных изображений, для зашумленных, для атакованных с помощью Deep Fool, для атакованных с помощью Deep Fool и после этого зашумленных, для атакованных с помощью FSGM, для атакованных с помощью FSGM и после этого зашумленных. По оси y отложены различные объекты, по оси x логиты их классов.

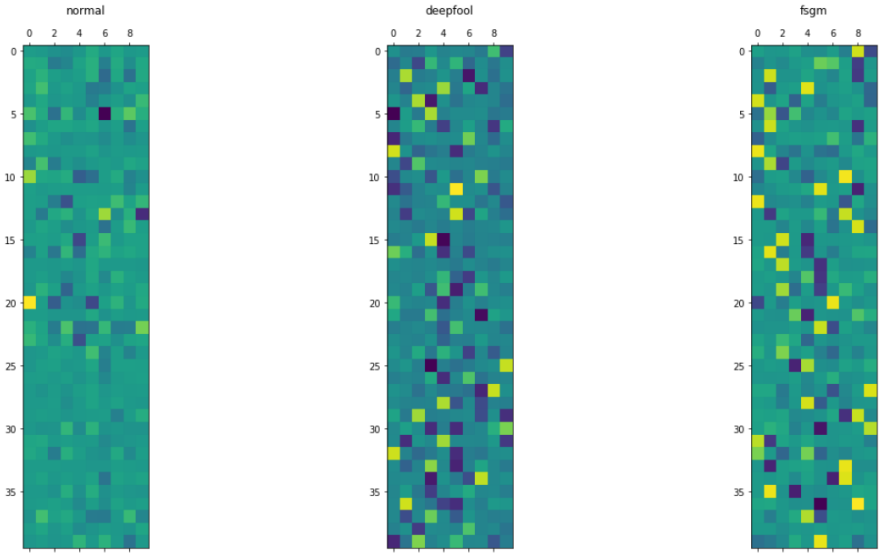

Рис. 3: Дельты логитов. По оси y отложены различные объекты, по оси x дельты логитов их классов.

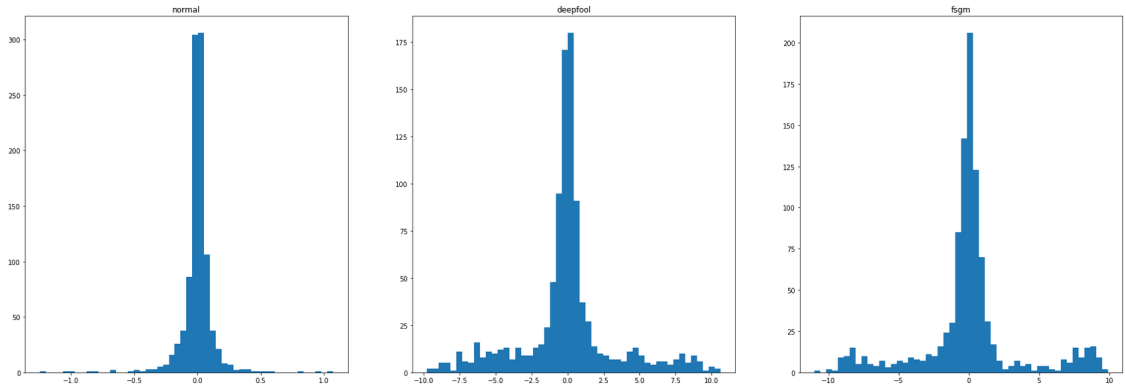

Рис. 4: Распределение дельт логитов.

Интересным здесь является то, что предсказания моделей для атакованных и атакованных, а затем зашумленных изображений сильно различаются (рис. 3, рис. 4).

Основываясь на этих эмпирических наблюдениях, можно построить простой детектор атакованных изображений. Например, можно просто передавать дельты логитов основной модели для зашумленного и обычного изображений в логистическую регрессию. Также до передачи в модель, можно попробовать их отсортировать. Последний подход дает ROC AUC равный 0.998 для атак типа FSGM и Deep Fool. Этот метод, однако, плохо работает с таргетированными атаками. Попробуем улучшить полученные результаты.

Известно, что некоторые аугментации данных при обучении позволяют получать более стойкие к атакам модели машинного обучения, например MaxUp [8], MixUp [10], CutMix+MaxUp [8]. Более того, в [8] доказано, что обучение с использованием MaxUp привносит дополнительную гладкость в ландшафт функции потерь, величина которой для обучающих примеров напрямую зависит от параметров аугментации. Иными словами, для небольших заранее заданых возмущений гарантировано наблюдается отсутствие резких скачков функции потерь в некотором ϵ-шаре от обучающего примера x.

Обратимся к методу рассмотрения отклика параметров объектов – это достаточно известный прием, множество защит, полученных с его помощью, описано в [7]. Рассмотрим отклик в ϵ-шаре от атакованного примера $\hat{x}$, чтобы понять как мы можем использовать аугментации для детекции атак. До передачи в модель применим $N(x, \mu, \sigma)$ к изображениям, построим гистограммы (рис. 5).

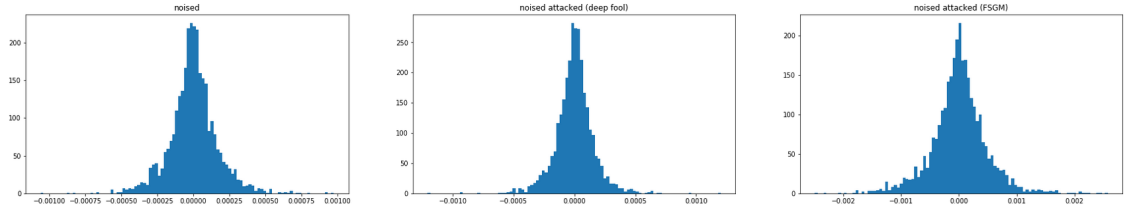

Рис. 5: Распределения откликов параметров.

Как видно из распределений на рис. 5, дисперсия отклика параметров для атакованных примеров значительно больше, чем для обычного изображения. Очевидно, что за этим стоит невысокая стабильность функции потерь вокруг атакованных изображений. Это также может быть связано с тем, что в большинстве атак требуется найти минимальное возмущение от оригинального примера x_i и, соответственно, минимальные же возмущения могут перенести объект в область пространства класса, отличного от атакованного.

Так как гладкость функции потерь гарантирована в ϵ-шаре, вблизи обучающих примеров небольшие возмущения параметров практически не вызывают изменений отклика параметров. Воспользуемся этим.

Рассмотрим распределения дельт откликов для обычного и зашумленного изображений при обычной модели (рис. 6).

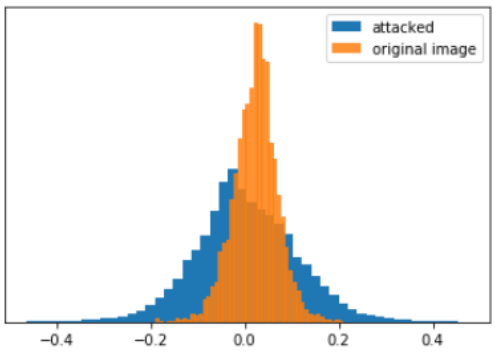

Рис. 6: При обычной модели (атака Deep Fool).

Теперь рассмотрим распределения дельт откликов для обычного и зашумленного изображений при MaxUp модели (рис. 7).

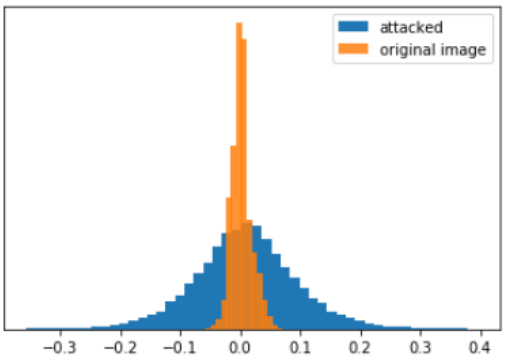

Рис. 7: При обычной MaxUp модели (атака Deep Fool).

Как видно на рис. 6 и рис. 7, распределения имеют значительные различия для атакованных и не атакованных объектов, в особенности для MaxUp модели.

За этим стоит очевидный факт: так как модель научилась игнорировать малые возмущения для объектов выборки, распределение откликов практически не изменяется. С другой стороны, если пример, атакованный моделью, до атаки распознавался как $class_i$, в ϵ-шаре от него не существует негладких областей перехода из одного класса в другой. Это значит, что любое возмущение любого объекта в ϵ-шаре от

данного примера будет либо распознано как $class_i$, либо иметь бóльшую дисперсию отклика чем объекты класса i.

Таким образом метод поиска атакованых объектов может быть сведен к следующему:

- Дообучение модели с использованием MaxUp.
- Создание набора данных, состоящего из дельт откликов для обычных и зашумленных объектов.
- Подсчет дельт для нового примера, сравнение их дисперсии и среднего с дисперсией и средним ранее полученного набора данных.

5. Эксперименты

Для проведения экспериментов были выбраны открытые наборы данных MNIST и CIFAR10. Тестирование производилось на ResNet-18. Метрика, по которой сравнивались детекторы атак – ROC AUC. Атаки, исследованные во время эксперимента:

- Deep Fool
- FSGM
- C&W
- PGD (для $\|\cdot\|_\infty$-нормы)

Результаты тестирования приведены в сводных таблицах ниже.
Для первой из описанных моделей:

	MNIST	CIFAR10
Deep Fool	1.0	0.998
FSGM	0.99	0.998
C&W	0.899	0.987
PGD	0.73	0.89

Для второй из описанных моделей:

	MNIST	CIFAR10
Deep Fool	1.0	0.998
FSGM	1.0	0.9936
C&W	1.0	0.989
PGD	1.0	0.997

Стоит также отметить, что при столь высокой точности нахождения атак, минусами решения являются:

- дополнительные траты на дообучение с использованием MaxUp;
- необходимость сделать предсказания для двух изображений: обычного и зашумленного;

- точность предсказаний основной модели может незначительно уменьшиться (в проведенных экспериментах падение точности составило 0.007).

6. Результаты

В ходе выполнения работы были достигнуты следующие результаты:

- Изучены существующие методы состязательных атак.
- Проанализированы существующие методы защиты от атак.
- Разработаны собственные методы защиты от атак.
- Проведено тестирование собственного метода.

Список литературы

[1] Athalye Anish, Carlini Nicholas, Wagner David. Obfuscated gradients give a false sense of security: Circumventing defenses to adversarial examples // arXiv preprint arXiv:1802.00420. — 2018.

[2] Carlini Nicholas, Wagner David. Towards evaluating the robustness of neural networks // 2017 ieee symposium on security and privacy (sp) / IEEE. — 2017. — P. 39–57.

[3] Evasion attacks against machine learning at test time / Battista Biggio, Igino Corona, Davide Maiorca et al. // Joint European conference on machine learning and knowledge discovery in databases / Springer. — 2013. — P. 387–402.

[4] Goodfellow Ian J, Shlens Jonathon, Szegedy Christian. Explaining and harnessing adversarial examples // arXiv preprint arXiv:1412.6572. — 2014.

[5] Hinton Geoffrey, Vinyals Oriol, Dean Jeff. Distilling the knowledge in a neural network // arXiv preprint arXiv:1503.02531. — 2015.

[6] Kurakin Alexey, Goodfellow Ian, Bengio Samy. Adversarial machine learning at scale // arXiv preprint arXiv:1611.01236. — 2016.

[7] ML-LOO: Detecting Adversarial Examples with Feature Attribution / Puyudi Yang, Jianbo Chen, Cho-Jui Hsieh et al. // arXiv preprint arXiv:1906.03499. — 2019.

[8] MaxUp: A Simple Way to Improve Generalization of Neural Network Training / Chengyue Gong, Tongzheng Ren, Mao Ye, Qiang Liu // arXiv preprint arXiv:2002.09024. — 2020.

[9] Moosavi-Dezfooli Seyed-Mohsen, Fawzi Alhussein, Frossard Pascal. Deepfool: a simple and accurate method to fool deep neural networks // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 2574–2582.

[10] Pang Tianyu, Xu Kun, Zhu Jun. Mixup Inference: Better Exploiting Mixup to Defend Adversarial Attacks // arXiv preprint arXiv:1909.11515. — 2019.

[11] Provably minimally-distorted adversarial examples / Nicholas Carlini, Guy Katz, Clark Barrett, David L Dill // arXiv preprint arXiv:1709.10207. — 2017.

[12] Sharma Yash, Chen Pin-Yu. Attacking the Madry Defense Model with L_1-based Adversarial Examples // arXiv preprint arXiv:1710.10733. — 2017.