Московский государственный университет имени М. В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Эссе

Состязательные атаки

Никоров Кирилл Николаевич
317 группа

курс « Глубокое обучение» ММП ВМК МГУ

Москва — 2021

Аннотация

В настоящее время глубокое обучение представляет собой эффективный способ решения широкого спектра задач. Стремительное развитие глубокого обучения привело к тому, что в ряде задач нейронные сети достигли качества работы человека. В связи с этим растут возможности прикладного использования нейронных сетей в производственных задачах. Однако в работах различных исследователей приводятся доказательства того, что эффективность работы нейронных сетей может быть сведена на нет использованием сгенерированных специальным образом объектов. Эти объекты кажутся естественными человеческому глазу, но они заставляют нейронные сети совершать ошибки. Такие обманы нейронных сетей называются состязательными атаками. Результаты состязательных атак могут быть использованы злоумышленниками для получения личной выгоды или нанесения вреда другим лицам, поэтому построение моделей, устойчивых к ним, является крайне важной задачей. В данной работе описаны основные понятия, связанные с состязательными атаками. Подробно рассмотрены так называемые атаки уклонения: представлены основные способы проведения этих атак и описаны наиболее эффективные методы защиты он них.

Содержание

1 Введение

В последние годы глубокое обучение развивается стремительными темпами. В некоторых традиционных областях, таких как классификация изображений, распознавание речи, перевод с одного языка на другой, архитектурам глубоких нейронных сетей удалось добиться значительного прогресса и близко приблизиться к человеческому уровню (или даже превзойти его). В связи с этим с каждым днем находятся все больше и больше приложений, в которых нейронные сети могли бы проявить себя наилучшим образом. Однако задачи из реальной жизни накладывают значительные ограничения на качество работы нейронных сетей. Одним из главных условий является безопасность их применения. Нейронные сети должны с высокой точностью выполнять поставленные перед ними задачи; их архитектуры должны быть масштабируемы и готовы к различным размерам входных данных; а также они должны быть устойчивы к естественным и синтетическим (например, наложенных злоумышленниками) помехам во входных данных.

Последние достижения в сфере глубокого обучения дают достаточно надежные гарантии выполнения первых двух пунктов в предложенном выше списке требований. Но, неожиданно, применение нейронных сетей наталкивается на серьезные проблемы в задаче построения устойчивого решения. В работе [1] было показано, что глубокие нейронные сети с высоким качеством работы могут оказываться совершенно неустойчивы к незначительным изменениям во входных данных: в задаче классификации изображений авторам удалось добиться неверных ответов предтренированных моделей за счет неразличимых человеческим глазом изменений входного изображения. Там же было показано, что такой эффект не является следствием ошибок процесса обучения какой-то конкретной модели. Авторы показали, что одно и то же модифицированное изображение может неверно классифицироваться различными и независимыми нейронными сетями. Из-за этого явления использование нейронных сетей в реальной жизни может привести к попыткам злоумышленников манипулировать результатами их работы с целью получения какой-либо выгоды. Попытки такого манипулирования называются **состязательными атаками**, или adversarial attacks.

После наблюдений, сделанных в [1], было разработано множество средств борьбы с состязательными атаками. В статье [2] была предложена процедура обучения нейронных сетей, устойчивых к состязательным атакам. Она использовала как естественные данные обучающей выборки, так и специальным образом зашумленные. В работе [6] была предложена идея дистилляции нейронных сетей, а в работе [7] рассматриваются возможности использования дистилляции для защиты от состязательных атак. В работе [9] предложен механизм борьбы с состязательными атаками, основанный на архитектурах GAN.

К сожалению, упомянутые выше механизмы защиты от состязательных атак оказываются эффективны только в борьбе с конкретным классом атак. Ни один из них не является окончательным решением проблемы. Более того, использование этих механизмов часто приводит к значительному ухудшению качества работы защищаемой нейронной сети.

В данной работе будут рассмотрены основные виды состязательных атак и методы их проведения. Кроме того, будут описаны основные стратегии защиты моделей от состязательных атак, проанализированы их плюсы и минусы.

2 Необходимые понятия и определения

В данной работе состязательные атаки на модели машинного обучения рассматриваются с двух сторон: с атакующей стороны и с защищающейся стороны. Атакующую сторону в данной работе будем называть *злоумышленниками*. Модель машинного обучения, на которую производится атака будем называть *жертвой* атаки.

2.1 Понятия, связанные с состязательными атаками

В работе [10] классифицируются основные понятия, связанные с состязательными атаками и методами борьбы с ними. Приведем здесь некоторые из них.

Область атаки

Любую работающую систему, основанную на алгоритмах машинного обучения можно представить в виде конвейера. Например, простейшая система машинного обучения может быть описана последовательностью следующих шагов: на первом шаге происходит считывание данных с сенсоров или соответствующих баз данных; на втором шаге эти данные предобрабатываются; на третьем – данные подаются на вход модели машинного обучения для получения прогноза; на четвертом шаге на основе полученного прогноза предпринимаются конкретные действия.

Область атаки – это шаг в конвейере системы, на который направлена атака злоумышленника. В [10] выделяются следующие виды атак:

- *Атаки уклонения* – это наиболее распространенный вид атаки. Злоумышленники пытаются заставить систему сделать неправильный прогноз за счет искажения входных объектов.
- *Отравляющие атаки* – это атаки, проводимые во время стадии обучения модели. Злоумышленники пытаются добавить в обучающую выборку сгенерированные специальным образом объекты, чтобы затруднить процедуру обучения или сделать ее полностью невозможной.
- *Исследовательские атаки* – это атаки, в ходе которых злоумышленники стараются получить как можно больше информации о самой системе: какой алгоритм машинного обучения в ней используется, какова его архитектура; как распределены обучающие данные.

Возможности злоумышленников

Степень опасности атак злоумышленников в первую очередь определяется количеством информации о системе, которым они владеют.

Атаки на *этапе обучения* системы можно классифицировать следующим образом:

- *Добавление новых данных* в обучающую выборку. Злоумышленники не имеют доступа к процедуре обучения и полному набору обучающих данных, но могут добавлять свои данные в этот набор.
- *Модификация данных* в обучающей выборки. Злоумышленники имеют доступ к значительной части обучающих данных.
- *Нарушение логики* процедуры обучения. Злоумышленники имеют доступ к процедуре обучения.

Атаки на *этапах применения* модели также можно классифицировать:

- *White-box атаки.* Злоумышленники имеют полный доступ к обученной модели машинного обучения: к ее параметрам, к обучающей выборке; у них есть информация о процедуре обучения.
- *Black-box атаки.* Злоумышленники почти не имеют никакой информации о модели. В свою очередь black-box атаки можно разделить на следующие виды:
  - *Неадаптивные атаки.* Злоумышленники имеют доступ только к обучающим данным модели.
  - *Адаптивные атаки.* Злоумышленники могут достаточно свободно взаимодействовать с моделью в формате запрос-ответ, но не имеют информации о процедуре обучения.
  - *Строгие атаки.* Злоумышленники могут получать пары входов-выходов модели, но не имеют возможности самостоятельно взаимодействовать с моделью.

Цели злоумышленников

При атаке на систему машинного обучения злоумышленники пытаются заставить систему сделать неверный прогноз. В зависимости от степени ошибки системы, которой добиваются злоумышленники, их цели можно классифицировать на:

- *Снижение уверенности.* Злоумышленники пытаются уменьшить степень уверенности модели в верном прогнозе.

- *Неправильный ответ.* Злоумышленники пытаются заставить модель построить любой неправильный прогноз.
- *Целевой неправильный ответ.* Злоумышленники пытаются заставить модель построить неправильный прогноз конкретного вида.
- *Неправильный ответ на паре объект-прогноз.* Злоумышленники пытаются заставить модель построить неправильный прогноз конкретного вида для конкретного объекта.

2.2 Генеративно-состязательные нейронные сети (GAN)

Некоторые описанные ниже стратегии защиты от состязательных атак используют так называемые генеративно-состязательные нейронные сети (или Generative adversarial network), сокращённо GAN [3]. Их базовая архитектура изображена на рис. 1. Данная сеть состоит из генератора (G) и дискриминатора (D). На вход генератора подается случайно сгенерированный вектор латентных переменных, на выходе генератора строится объект из некоторого целевого распределения. На вход дискриминатора подаются объекты, полученные в генераторе, а также случайный набор реальных объектов из целевого распределения. Дискриминатор должен определить, какие объекты были сгенерированы. Соответственно, дискриминатор учится как можно точнее определять реальность объекта, а генератор учится как можно лучше *обманывать* дискриминатор. Таким образом, оптимальный генератор задается соотношением:

$$G^*(z) = \arg\min_G \max_D \left(E_{X\sim p(X)} \left[\log D()\right] + E_{z\sim q(z)} \left[\log(1 - D(G(z)))\right] \right),$$

где $p(X)$ – распределение реальных данных X, $q(z)$ – распределение для генерации латентных переменных z.

Эвристика данного соотношения достаточно проста. Оптимальные генератор $G^*(z)$ – это генератор, который даже самый качественный дискриминатор (часть $\max\limits_D$) обманывает наилучшим образом (часть $\min\limits_G$).

3 Атаки уклонения на глубокие нейронные сети

В данной работе будут подробно рассмотрены описанные в предыдущем разделе атаки уклонения, так как они являются самым распространенным видом состязательных атак.

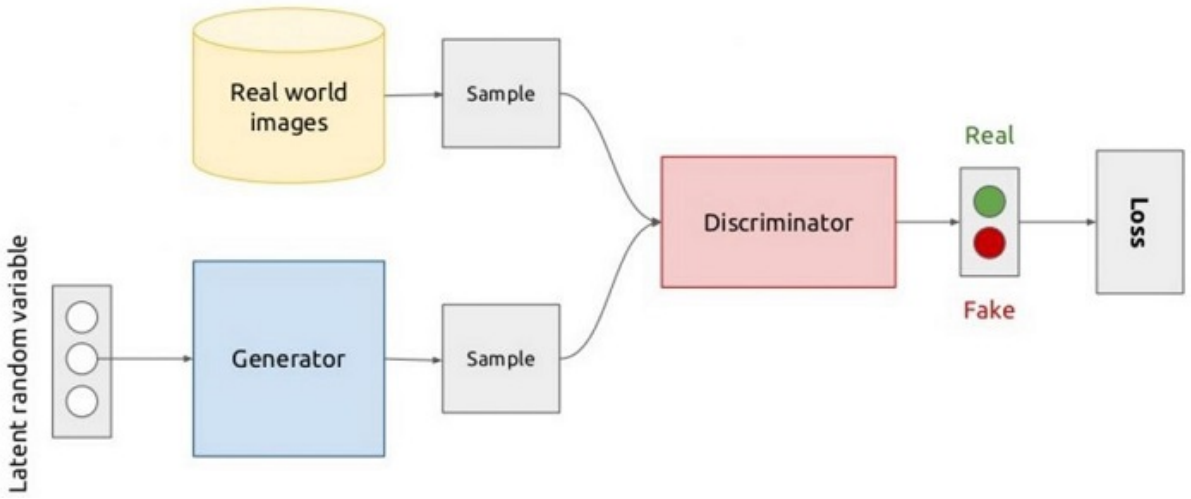

Рис. 1: Структура нейронной сети GAN.

3.1 White-box атаки

Рассмотрим обученную глубокую нейронную сеть $F(\cdot)$. Пусть X – входной объект этой сети. Задача злоумышленников состоит в том, чтобы модифицировать входной объект X, используя как можно меньшее возмущение δX, и получить объект $X_* = X + \delta X$ такой, что $F(X_*) = Y_* \neq Y = F(X)$. Мы рассматриваем white-box атаку, поэтому злоумышленники также имеют доступ к параметрам нейронной сети θ_F. В [7] был представлен общий подход к модификации объектов для обмана нейронной сети (см. рис. 2). Подход делится на две стадии:

1. Определение наилучшего направления модификации;
2. Выбор величины модификации.

Определение наилучшего направления модификации

На этой стадии входной объект X рассматривается, как n-мерный вектор. Злоумышленники стараются определить направление модификации данного вектора, которое позволяет достичь им поставленных целей оптимальным образом, то есть за счет наименее заметной модификации. Рассмотрим наиболее эффективные техники.

1. **Fast gradient sign method (FGSM).** Быстрое и эффективное решение данной задачи было предложено в [11]. В данной статье авторы предлагают считать необходимое возмущение входного объекта X с помощью градиента функции потерь обученной модели, посчитанному по входному вектору X. Конкретная формула для расчетов

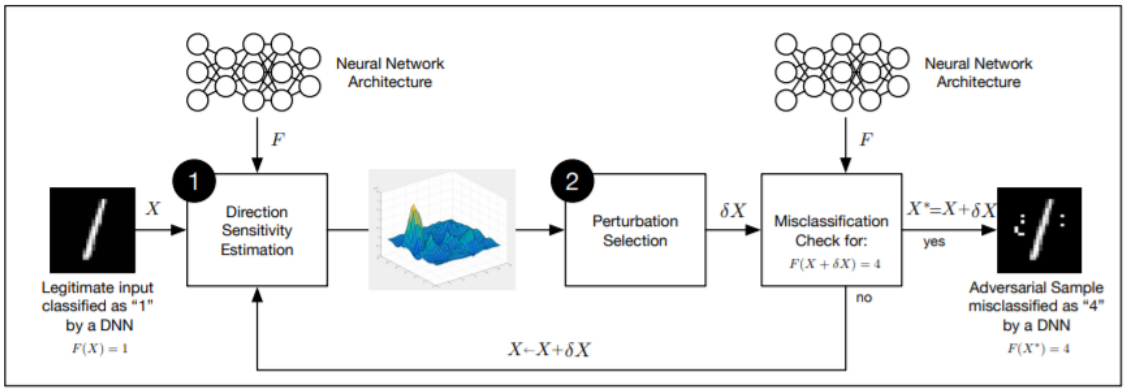

Рис. 2: Модификация объектов для обмана нейронной сети состоит из двух шагов: (1)Определение наилучшего направления модификации; (2) Выбор величины модификации. Первый шаг оценивает чувствительность модели F на объекте X. Второй шаг использует информацию о чувствительности и генерирует возмущение δX объекта X для модификации. Если полученный объект $X + \delta X$ обманывает модель F, то атака выполнена. Иначе – шаги повторяются для полученного объекта: $X \leftarrow X + \delta X$.

выглядит следующим образом:

$$X_* = X + \alpha \operatorname{sign}(\nabla_X \mathcal{L}(X, y_{true})),$$

где $\mathcal{L}$ – функция потерь, которая использовалась во время обучения модели, y_{true} – истинный ответ для объекта X, α – положительный коэффициент, определяющий величину модификации; функция $\operatorname{sign}(\cdot)$ от векторного аргумента – это функция $\operatorname{sign}(\cdot)$ от скалярного аргумента, примененная к каждой компоненте векторного аргумента. Данный метод суть есть один шаг градиентного подъема, сделанный для максимизации функции потерь модели. Конечно, величина коэффициента α должна быть такой, чтобы вносимые во входной объект изменения были как можно менее заметны. На рис. 3 приведен пример работы метода FGSM из [11].

2. **Target class FGSM.** Это один из вариантов метода `FGSM` [2], описанного выше. Он используется не только для того, чтобы заставить модель сделать ошибку, но и для того, чтобы заставить модель отнести входной объект к конкретному классу y_{target}. Модифицированный объект генерируется по следующей формуле:

$$X_* = X - \alpha \operatorname{sign}(\nabla_X \mathcal{L}(X, y_{target}).$$

В данном случае метод суть есть один шаг градиентного спуска (а не подъема, как в предыдущем методе), сделанный для минимизации функции потерь на нужной злоумышленникам паре (X, y_{target}).

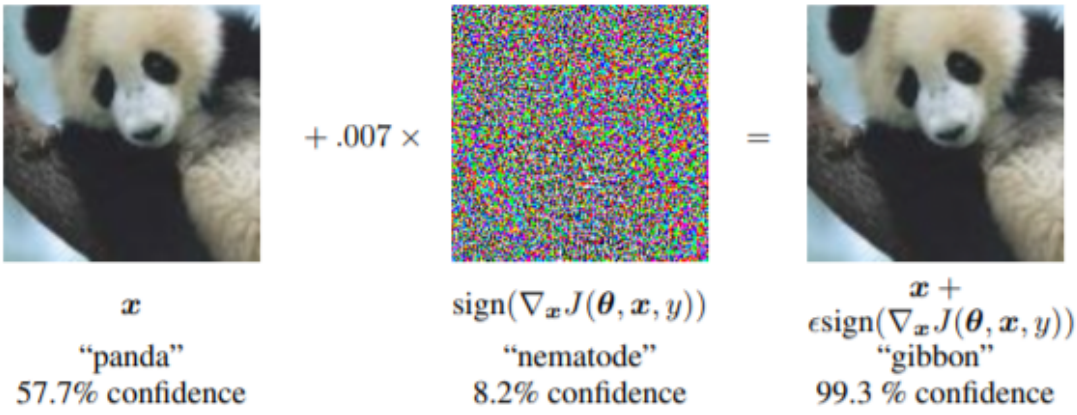

Рис. 3: Демонстрация атаки с помощью метода FGSM из [11]. Добавлением к входному изображению не различимого взглядом возмущения, элементы которого являются знаками соответствующих элементов градиента функции потерь по входному объекту, авторы добиваются неверной классификации нейронной сетью.

3. **Итеративный вариант FGSM.** В [2] была представлена еще одна логичная попытка обобщить метод FGSM. А именно – исследовалось поведение итеративного FGSM. Итерационный процесс задается следующими уравнениями:

$$X_*^0 = X; \tag{1}$$

$$X_*^{i+1} = \text{clip}_{X,\varepsilon}\left(X_*^i + \alpha\,\text{sign}\left(\nabla_X \mathcal{L}\left(X_*^i,\, y_{true}\right)\right)\right); \tag{2}$$

где $\varepsilon \in \mathcal{R}$, $\varepsilon > 0$, функция $\text{clip}_{X,\varepsilon}(Z) : \mathcal{R}^n \to \mathcal{R}^n$ выполняет поэлементную обрезку компонент вектора $Z \in \mathcal{R}^n$ так, чтобы $\forall\, i = \overline{1, \ldots, n}\,, \left(\text{clip}_{X,\varepsilon}(Z)\right)_i \in [X_i - \varepsilon,\, X_i + \varepsilon]$. Обрезать компоненты получаемых на каждой итерации объектов нужно для того, чтобы итоговый модифицированный объект не сильно отличался от исходного входного объекта X.

4. **Другие итеративные методы.** После введения итеративного варианта FGSM для построения модифицированного объекта можно обобщить практически любой итеративный метод оптимизации: можно добавить инерцию (momentum) или сделать метод построения адаптивным.

Выбор величины модификации

1. **Модификация всех размерностей.** Описанные выше методы FGSM возмущают сразу все признаки входного объекта, но на величину, ограниченную некоторой небольшой константой $\varepsilon > 0$.

2. **Отбор размерностей для модификации.** Существуют также методы, которые возмущают лишь некоторое фиксированное число признаков. Такой метод, например, описан в [5]. Он снижает общее число возмущений и приводит к более незаметной атаке. Однако, по сравнению с методами, модифицирующими все признаки, он более затратный в плане вычислений.

Универсальное возмущение

Рассмотренные ранее методы построения модифицированных объектов получали объект X_* для атаки, опираясь на какой-либо естественный объект X, прибавляя к нему некоторое возмущение $\delta X = \delta X(X)$, зависящее от данного естественного объекта X. Однако в ходе исследования, описанного в [4], выяснилось, что для атаки можно подобрать универсальное возмущение δX, которое не зависит от входного объекта, модифицирует его не заметно для человеческого глаза и приводит к успешной атаке нейронной сети для большой доли естественных входных объектов.

Пусть $\mathcal{X} = \{X_1, X_2, \ldots, X_N\}$ – обучающая выборка. F – модель, на которую проводится атака. Цель алгоритма – найти такое возмущение δX, чтобы $||\delta X||_p \leq \varepsilon$ и как можно больше объектов $X_{*,i} = X_i + \delta X$, $X_i \in \mathcal{X}$ успешно проводили атаку на сеть: $F(X_{i,*}) \neq F(X_i)$. Алгоритм итеративно проходит по обучающей выборке и на каждом шаге уточняет универсальное возмущение δX. Иллюстрация алгоритма приведена на рис. 4. На каждой итерации i рассматривается объект X_i, для которого универсальное возмущение δX не позволяет провести атаку, то есть $F(X_i + \delta X) = F(X_i)$. Ищется минимальное добавочное возмущение δX_i, которое позволяет успешно провести атаку на сеть, используя объект X_i и текущее универсальное возмущение δX: $F(X_i + \delta X + \delta X_i) \neq F(X_i)$. Формально говоря, решается следующая оптимизационная задача:

$$\delta X_i = \arg\min_r ||r||_p \text{ при условии } F(X_i + \delta X + r) \neq F(X_i)$$

После ее решения получено новое значение для универсального возмущения $\delta X \leftarrow \delta X + \delta X_i$ и данное возмущение проецируется на шар радиуса ε:

$$\delta X \leftarrow \arg\min_{\delta' X} ||\delta X - \delta' X||_p \text{ при условии } ||\delta' X||_p \leq \varepsilon.$$

Алгоритм завершается, когда доля успешных обманов с помощью объектов из множества $\mathcal{X}_* = \{X_1 + \delta X, X_2 + \delta X, \ldots, X_N + \delta X\}$ превосходит заданный порог $1 - \alpha$, $\alpha \geq 0$. На рис. 5 представлены примеры успешных атак с использованием универсального возмущения.

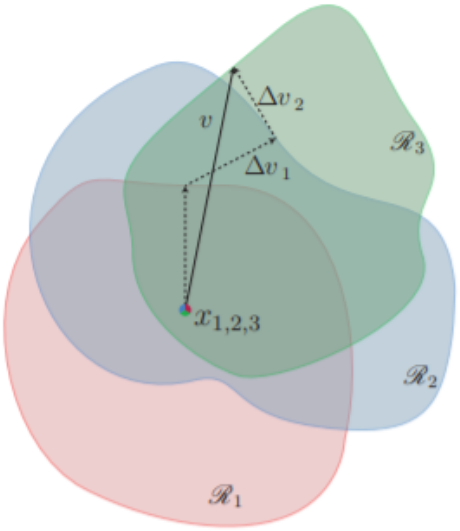

Рис. 4: Схема описываемого алгоритма. Здесь x_1, x_2, x_3 – входные объекты, а регионы R_i константной классификации показаны различными цветами. Алгоритм последовательно подбирает минимальное возмущение Δv_i, которое выводит текущий модифицированный объект $X_i + v$ за соответствующий объекту X_i регион классификации R_i. Рисунок взят из [4].

Рис. 5: Примеры модифицированных изображений. Первые 8 изображений взяты из валидационной части датасета `ILSVRC 2012`, остальные изображения сняты на камеру мобильного телефона. Рисунок взят из работы [4].

3.2 Black-box атаки

В случае неадаптивной или строгой black-box атаки у злоумышленников есть доступ к значительному количеству обучающих данных. Для таких атак работает достаточно прямолинейный план действий. Злоумышленникам необходимо построить локальную модель, которая решает ту же задачу, что и модель-жертва атаки. Далее, локальная модель обучается с использованием доступных данных. После полного обучения полученная мо-

дель является аппроксимацией модели-жертвы, и к ней можно применять любые методы white-box атак для генерации модифицированных объектов. Согласно интересному свойству `Transferability` объектов для атак, описанному в [12], объекты, полученные на локальной машине, можно успешно использовать для black-box атак на неизвестные модели. Ниже свойство `Transferability` будет рассмотрено более подробно.

Сложнее обстоят дела при адаптивных black-box атаках, когда злоумышленник не имеет доступа к большому количеству обучающих данных и вынужден взаимодействовать с моделью в формате запрос-ответ. В этом случае запросы должны составляться специальным образом.

Часто для проведения таких black-box атак используются эволюционные алгоритмы оптимизации, которые не используют градиенты оптимизируемой функции. В частности, часто используется алгоритм дифференциальной эволюции [13].

One-Pixel атака

В работе [14] алгоритм дифференциальной эволюции используется для атаки на нейронный классификатор. Идея атаки заключается в поиске наилучшего набора $(x, y, [r_{ch}, g_{ch}, b_{ch}])$, где (x, y) – координата одного пикселя на входном изображении, (r_{ch}, g_{ch}, b_{ch}) – цвет в формате `RGB` для модификации пикселя с координатами (x, y). Авторы показали, что около 68% изображений из `CIFAR-10` и около 16% из тестового датасета `ImageNet (ILSVRC 2012)` могут быть успешно использованы для атаки сразу на несколько нейронных сетей с использованием представленного алгоритма и модификацией только одного пикселя входного изображения. На рис. 6 показаны примеры успешных атак.

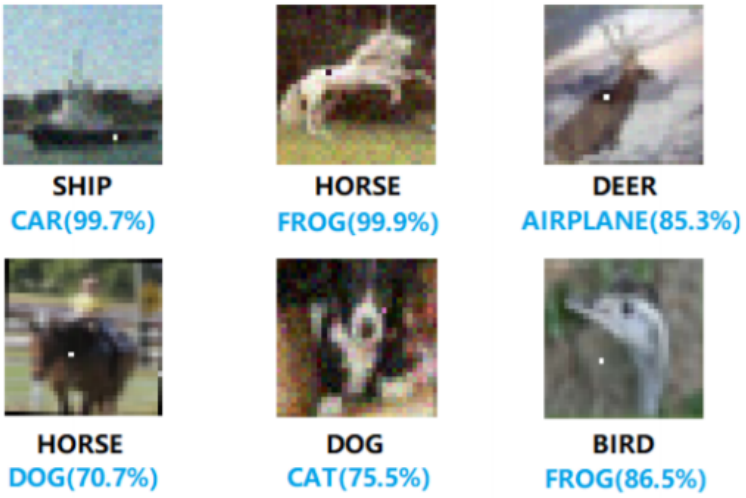

Рис. 6: Модификация одного пикселя с помощью описанного в [14] алгоритма приводит к успешной атаке. Черным цветом подписан истинный класс объекта, синим – прогноз нейросети. Рисунок взят из [14].

Few-Pixel атака

Описанный выше алгоритм One-Pixel атаки никак не ограничивает величину модификации пикселя. Из-за этого такую атаку достаточно просто детектировать на этапе препроцессинга данных. Поэтому в работе [15] тех же авторов представлен другой алгоритм, который меняет во входном изображении заданное число пикселей и старается сделать эти изменения минимальными по величине. Этот алгоритм также опирается на дифференциальную эволюцию. На рис. 7 показаны примеры успешных атак этого алгоритма.

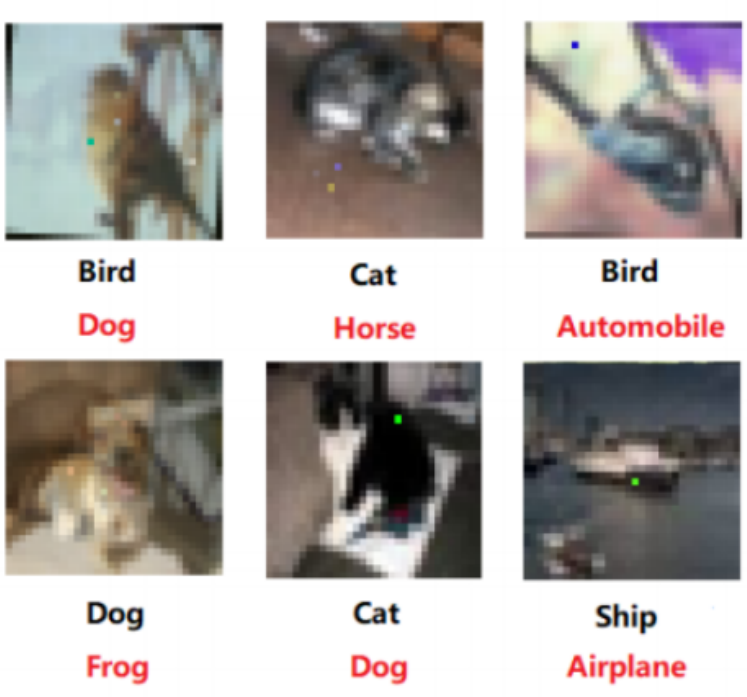

Рис. 7: Модификация нескольких пикселей с помощью описанного в [15] алгоритма приводит к успешной атаке. Изменение каждого пикселя меньше по величине, чем в алгоритме [14]. Из-за этого такую атаку труднее отследить. Черным цветом подписан истинный класс объекта, красным – прогноз нейросети. Рисунок взят из [15].

Свойство `Transferability` в состязательных атаках

Оказывается, что модифицированный объект, полученный для атаки одной модели может быть использован для такой же атаки другой модели, независимо от архитектур данных моделей. Это свойство описано в [12] и известно, как `Tranferability of Adversarial Samples`. Таким образом, в случае black-box атаки злоумышленник может получить модифицированный объект $X + \delta X$, используя локально обученную модель F' , и применить этот объект для атаки модели F, несмотря на то, что у него нет почти никакой информации о F.

По характеру схожести моделей F и F' свойство `Transferability` можно разделить на две категории:

1. **Intra-technique transferability:** обе модели F и F' принадлежат одному и тому же семейству моделей машинного обучения (например, обе модели – нейронные сети, или обе – случайные леса).
2. **Cross-technique transferability:** модели F и F' принадлежат различным семействам моделей машинного обучения (например, модель F – случайный лес, а F' – нейронная сеть).

Для black-box атак можно пользоваться любой из категорий `Transferability`. Главное, чтобы модели F и F' имели примерно одинаково высокое качество работы.

`Cross-technique transferability` позволяет успешно проводить атаки на недифференцируемые модели, такие как `RandomForest` или `kNN`. Для этого достаточно обучить локальную дифференцируемую модель (например, нейронную сеть) для решения той же задачи и сгенерировать модифицированные объекты для нее.

4 Стратегии защиты от состязательных атак

Существующие механизмы защиты от состязательных атак могут быть разделены на несколько типов, в зависимости от методов их реализации.

4.1 Обучение с учетом состязательных атак

Идея данного метода заключается в том, что устойчивость модели к атакам можно увеличить, если использовать модифицированные объекты во время обучения модели. Защищающейся стороне необходимо сгенерировать большое количество таких объектов и использовать их, как аугментацию входных данных. В работе [2] использовалась следующая функция потерь:

$$\mathcal{L} = \frac{1}{(m-k)+\lambda k}\left(\sum_{i \in CLEAN} L(X_i \,|\, y_i) + \lambda \sum_{i \in ADV} L(X_i \,|\, y_i)\right),$$

где $L(X \,|\, y)$ – это функция потерь для одного объекта X с истинным классом y; m – число объектов в обучающем мини-батче; k – число аугментированных объектов в мини-батче; а λ – параметр, контролирующий относительный вес аугментированных объектов в функции потерь. $CLEAN$ и ADV – множества чистых и аугментированных объектов соответственно.

В работе [11] говорится, что использование аугментации в данном случае – это не совсем точное решение, так как аугментация предназначена для получения объектов, которые модель может встретить естественным образом в период тестирования. Объекты, используемые во время атак, нельзя назвать естественными, даже если возмущения в них не заметны человеческому глазу. Вместо аугментации авторы [11] предлагают изменить функцию потерь для одного объекта следующим образом:

$$\tilde{L}(X, y, \lambda) = \lambda\, L(X, y) + (1 - \lambda)\, L(X + \alpha\, \mathrm{sign}(\nabla_X L(X, y)),\, y),$$

где $0 < \lambda < 1$.

В обоих подходах объекты для атаки генерируются одним из способов для white-box атак.

В работах [17, 18] показано, что данная процедура обучения не защищает от black-box атак, когда злоумышленник обучает для атаки свою локальную модель. А в работе [19] доказывается, что в ходе обучение с учетом состязательных атак нейронные сети приобретают устойчивость лишь к очень слабым модификациям в окрестности входных объектов. Авторы предлагают двухшаговую схему атаки для такой модели. В начале ко входному объекту применяется случайное возмущение, в итоге получается объект вне окрестности входного объекта. А после этого к возмущенному объекту применяется любой white-box метод.

4.2 Дистилляция модели для защиты от состязательных атак

В работах [7, 8] показано, что дистилляция [6] нейронных сетей может быть использована для защиты от состязательных атак. Метод можно описать следующим образом. Нейронная сеть F обучается предсказывать для объектов X вероятности Y принадлежности этого объекта к классам задачи. Затем нейронная сеть F' с такой же архитектурой или архитектурой меньшего размера обучается на том же датасете предсказывать такие же выходы, что и сеть F. F' обучается, пока не достигнет такого же качества работы на исходной задаче, что и сеть F. Тогда выходы второй сети F' получаются более гладкими, и модель получается более устойчивой к модифицированным объектам.

Последний слой модели F, подвергаемой дистилляции, задается уравнением:

$$F_i(X) = \frac{e^{\frac{z_i(X)}{T}}}{\sum_{i=1}^{|Y|} e^{\frac{z_i(X)}{T}}},$$

где T – параметр дистилляции, называемый температурой. В [7] экспериментально показано, что высокие значения параметра T приводят к лучшему сглаживанию. Основной

причиной использования дистилляции является сглаживание окончательной модели. Это дает ей большую обобщающую способность на неизвестных данных.

Однако в работах [18, 20] показано, что такой метод защиты можно достаточно просто обойти, используя black-box атаки. И основной причиной неудачной защиты здесь вновь является свойство `transferability` модифицированных объектов.

4.3 Борьба с transferability

Как видно из примеров выше, основной причиной неудач в защите от состязательных атак является свойство transferability модифицированных объектов между различными моделями машинного обучения. Следовательно, ограничение transferability является наиболее приоритетной задачей при защите от атак.

Один из наиболее действенных способов борьбы со свойством transferability был предложен в [21]. Авторы добавили во множество выходных классов модели еще один класс – $NULL$-класс – и обучали классификатор определять модифицированные объекты, как объекты из нового класса $NULL$. Основная идея этого метода в том, что при классификации модифицированного объекта модель без такой защиты вынуждена снижать вероятность истинного класса объекта и повышать вероятность других первоначальных классов, тогда как с предложенным расширением у модели появляется возможность повышать вероятность класса $NULL$ без повышения вероятностей других классов. Тем самым модель может отлавливать атаки и отказываться от построения прогноза. На рис. 8 показан упрощенный пример добавления класса $NULL$.

Первоначальное обучение. В начале обучения параметры нейронной сети достаточно случайны, поэтому модифицированные объекты, сгенерированные на данном этапе, будут иметь мало общего с объектами, сгенерированными итоговым классификатором. Поэтому изначально модель тренируется на исходных чистых данных. Желаемый выход нейронной сети – это гладкое распределение вероятности принадлежности объектов к классам задачи. То есть желаемая вероятность истиного класса кладется равной некоторому числу q, близкому к единице, вероятности остальных классов распределяются равномерно и кладутся равными числу $\frac{1-q}{K-1}$, где K – число классов в задаче, а вероятность p_{NULL} кладется равной нулю.

Расчет вероятностей принадлежности классу $NULL$ для модифицированных объектов. После того, как модель достигает высокого качества работы на чистых данных, обучение переходит в следующую фазу. Теперь для каждого обучающего объ-

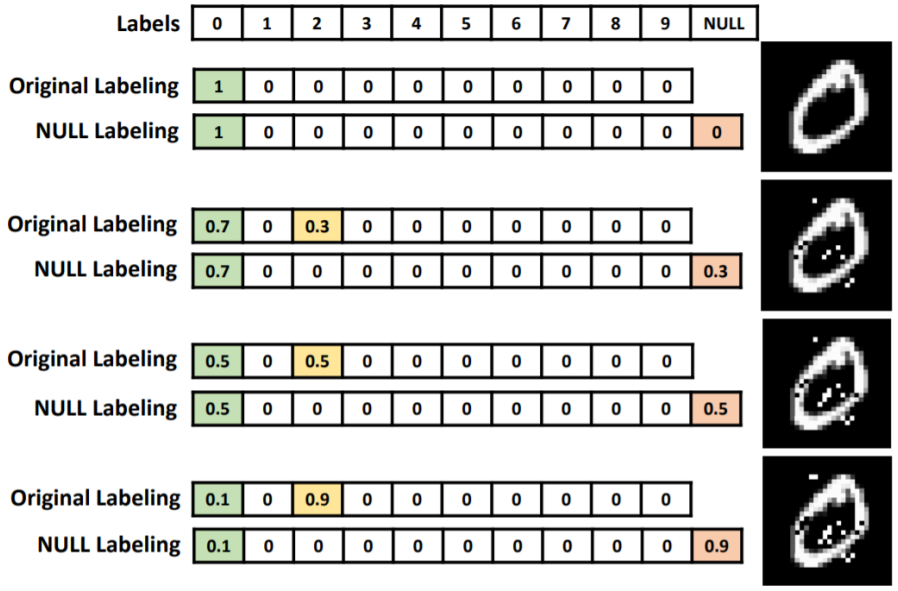

Рис. 8: Пример первоначальной разметки и разметки с классом $NULL$ для объектов из датасета `MNIST` и три объекта для атаки с различными модификациями. Чем больше величина модификации, тем большую вероятность классификатор отдает классу $NULL$. Данный пример является несколько упрощенным. Более подробное описание преобразований описаны в данном разделе. Рисунок взят из [21].

екта необходимо расчитать вероятность, что данный объект модифицирован для атаки на нейронную сеть. Для этого авторы предлагают построить отображение $f : [0, 1] \to [0, 1]$, $f(\varepsilon) = p_{NULL}$, следующим образом. Перебираются объекты из валидационной выборки, и из каждого объекта, с помощью методов создания объектов для атак, перебором по дискретной сетке значений $\varepsilon \in [0, 1]$ генерируется некоторое подмножество множества $\mathcal{X}_* = \{X_{*,\varepsilon} = X + \delta X_\varepsilon;\ ||\delta X_\varepsilon||_0 \leq \varepsilon\,|X|,\ \forall \varepsilon \in [0, 1]\}$, где $||\cdot||_0$ – нулевая мера, равная числу ненулевых компонент своего аргумента; $|X|$ – размерность признакового описания входного объекта X. После этого значение функции $f(\varepsilon)$ кладется равным доле объектов $X_{*,\varepsilon}$ среди объектов расширенной валидационной выборки, которые оказались успешными в ходе атаки на сеть. На рис. 9 показаны примеры функции f для датасетов `MNIST` и `GTSRB`.

Обучение с учетом состязательных атак. Далее начинается фаза обучение нейронной сети с использованием модифицированных объектов. Эти объекты генерируются из обучающих объектов с помощью метода `STG`, описанного в [21], который модифицирует в обучающем объекте наперед заданное число компонент. Это число выбирается случайным образом из равномерного распределения $U[1,\ N_{max}]$, где $N_{max} \in [1,\ |X|]$ – это минимальное число, для которого $f\left(\frac{N_{max}}{|X|}\right) = 1$. В результате получается объект $X_* = X + \delta X$, желае-

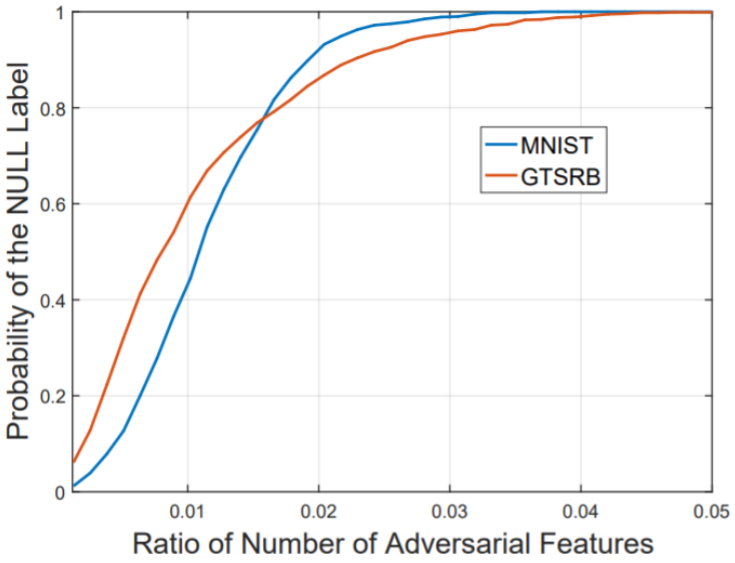

Рис. 9: Вероятность класса $NULL$ для модифицированных объектов в зависимости от доли ε модифицированных признаков. Для нейронных сетей, обученных на датасете `MNIST` и `GTSRB`. Рисунок взят из [21].

мое распределение вероятностей которого рассчитывается следующим образом. В начале для объекта X_* рассчитывается вероятность $p_{NULL} = f\left(\frac{||\delta X||_0}{|X|}\right)$, затем изначальная вероятность истинного класса q кладется равной $q' = q(1 - p_{NULL})$, а вероятности остальных классов распределяются равномерно и равны $\frac{(1-q)(1-p_{NULL})}{K-1}$, где K – число классов в изначальной задаче классификации. Таким образом, нейронная сеть учиться повышать не вероятности неверных классов в случае неуверенности классификации, а вероятность p_{NULL} того, что объект является атакующим.

На рис. 10 показана блок-схема работы описанного метода.

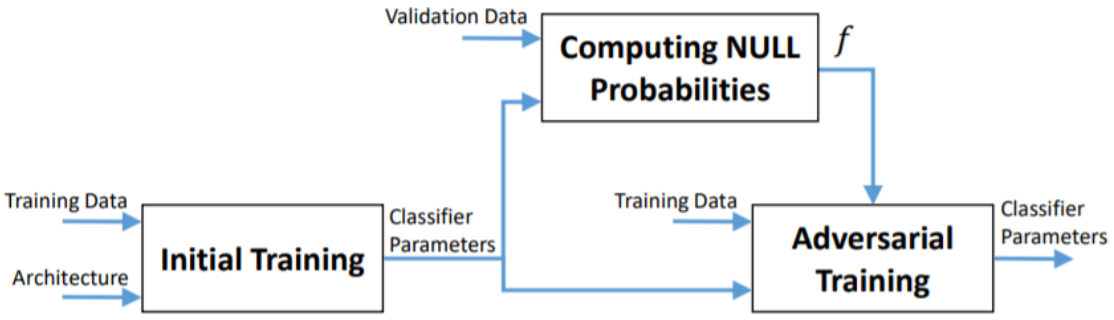

Рис. 10: Блок-схема метода добавления класса $NULL$ в разметку данных. На фазе начального обучения классификатор обучается только на чистых данных, на фазе обучения с учетом состязательных атак классификатор использует как чистые, так и модифицированные объекты. Рисунок взят из [21].

Данный метод является одним из наиболее эффективных методов борьбы с состязательными атаками. Он достаточно точно отклоняет атакующие объекты и при этом сохраняет высокое качество работы на естественных объектах.

4.4 Защитные модели на основе GAN-архитектур

В работе [9] представлен алгоритм `Defense-GAN` защиты от состязательных атак, основывающийся на использовании архитектур `GAN`. Основная идея этого алгоритма в том, чтобы использовать для классификации не поданный на вход классификатора объект X, а его реконструкцию, полученную с помощью генератора предобученного `GAN`. Гипотеза такого подхода в том, что в случае хорошо обученного и достаточно сложного генератора реконструкция исходного объекта X будет не сильно отличаться от самого объекта, но в то же время, она не будет содержать в себе шум δX, который мог быть добавлен в объект X для атаки на классификатор. Более формально алгоритм описан ниже.

В начале тренируется `GAN` $G(z)$ на имеющихся чистых данных. После этого становится возможным строить с помощью $G(z)$ реконструкцию произвольного входного объекта X. Для этого решается задача оптимизации

$$\min_z ||G(z) - X||_2^2$$

и строится подходящая инициализация z_* для `GAN`. Теперь можно обучать необходимый для решения задачи классификатор, используя чистые данные, реконструкцию чистых данных или совмещая оба подхода. При этом в работе [22] показано, что в случае, когда `GAN` $G(\cdot)$ имеет достаточную сложность и процедура обучения подобрана хорошим образом, имеет место предельное соотношение

$$E_{x \sim p_{data}} \left[\min_z ||G_t(z) - X||_2 \right] \xrightarrow[t \to \infty]{} 0$$

где t – шаг обучения `GAN`. Так что использование предобученного генератора не должно влиять на процесс обучения классификатора негативным образом. Схема алгоритма представлена на рис. 11.

У `Defense-GAN` есть ряд значительных преимуществ, выделяющих его перед другими средствами защиты:

1. `Defense-GAN` можно использовать с любым классификатором. Он никак не изменяет архитектуру классификатора, и его можно считать за шаг предобработки входных данных.

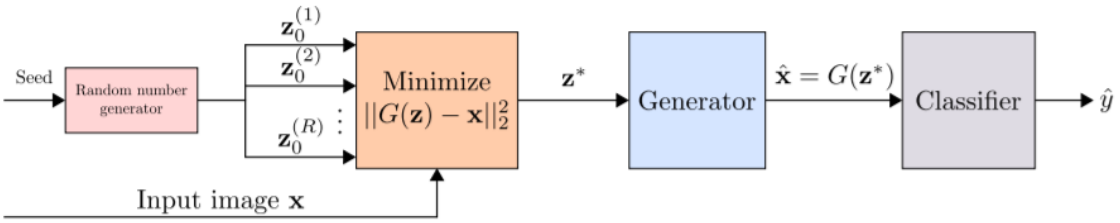

Рис. 11: Схема работы алгоритма `Defense-GAN`. Для классификации входного объекта X используется его наилучшая реконструкция, построенная с помощью `GAN`. Рисунок взят из [9]

2. `Defense-GAN` способен защищать от любого вида атак, так как его архитектура опирается не на конкретный метод генерации объекта для атаки, а на удаление шума из входного объекта и его сглаживание.

3. `Defense-GAN` является очень сложной моделью, которая итеративно решает задачу оптимизации (для получения оптимальной инициализации z_*) в ходе своей работы. Поэтому даже в случае white-box атаки обычные градиентные методы не смогут справиться с задачей построения модифицированного объекта.

Несмотря на то, что `Defense-GAN` оказался очень эффективным средством для защиты от состязательных атак, успех его применения очень сильно зависит от качества работы генератора `GAN`. Известно, что хорошее обучение `GAN` может оказаться достаточно трудной процедурой, а без очень качественного генератора `Defense-GAN` практически перестает работать.

4.5 MagNet

Как было сказано выше, `Defense-GAN` не меняет архитектуру классификатора и не делает каких-либо предположений о природе объектов, используемых для атаки. Примером другого подхода, обладающего теми же свойствами, является `MagNet` [23].

В работе [23] авторы рассматривают множество естественных объектов, как многообразие в многомерном пространстве. Они говорят, что причинами, по которой нейронная сеть ошибается на модифицированном объекте являются: (1) то, что объект далек от границы данного многообразия, но при этом у сети нет возможности отказаться от прогноза для данного объекта; (2) то, что объект близок к границе данного многообразия, но нейронная сеть обладает слабой обобщающей способностью вне многообразия естественных объектов. В связи с этими причинами, представленная авторами модель `MagNet` состоит из двух компонент: (1) детектор, который способен отвергать объекты, находящиеся чересчур

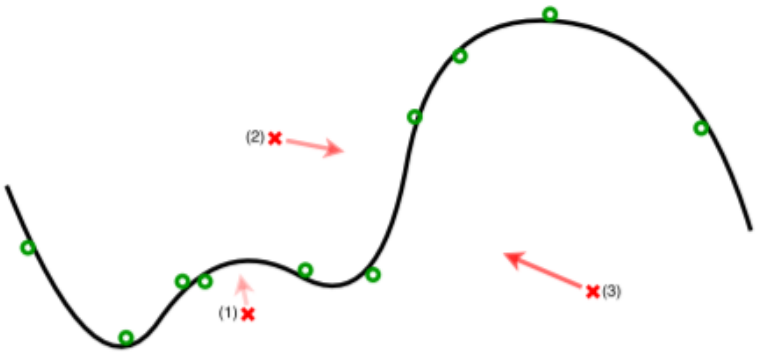

Рис. 12: Иллюстрация того, как работают детектор и преобразователь из [23] в двумерном пространстве. Линия представляет собой многообразие естественных объектов, естественные и модифицированные объекты изображены зелеными точками и красными крестиками, соответственно. Детектор отвергает слишком далекие от многообразия объекты, а преобразователь находит проекцию для близких объектов. Рисунок взят из [23].

далеко от границы многообразия; (2) преобразователь, который способен проектировать незначительно модифицированный объект X_* на ближайший объект X из многообразия естественных объектов. На вход классификатору подаются объекты, прошедшие через преобразователь. На рис. 12 представлена иллюстрация работы детектора и преобразователя в двумерном пространстве объектов.

В данной работе детектор учится моделировать многообразие естественных объектов и оценивать расстояние между входным объектом и данным многообразием. Для реализации такой модели авторы используют архитектуру автоэнкодера. Автоэнкодер $ae : \mathcal{S} \to \mathcal{S}$ состоит из двух частей $ae = d \circ e$, где $e : \mathcal{S} \to \mathcal{H}$, $d : \mathcal{H} \to \mathcal{S}$, $\mathcal{S}$ – множество входных объектов, $\mathcal{H}$ – пространство скрытых представлений. Работа детектора строится на двух предположениях. (1) Для обученного классификатора f его прогноз $f(X)$ для естественного объекта X и его прогноз для объекта $ae(X)$, прошедшего через автоэнкодер, будут мало отличаться, так как для естественного объекта и хорошо обученного автоэнкодера $X \sim ae(X)$. (2) В то же время для модифицированного объекта X_*, который не принадлежит многообразию естественных объектов, $X_* \not\sim ae(X_*)$, так как детектор учился только на естественных объектах, и поэтому $f(X_*)$ значительно отличается от $f(ae(X_*))$. Таким образом, детектор учится минимизировать дивергенцию двух распределений: $f(X)$ и $f(ae(X))$, на естественных объектах. В работе [23] для обучения детектора используется дивергенция Йенсена—Шеннона между данными распределениями.

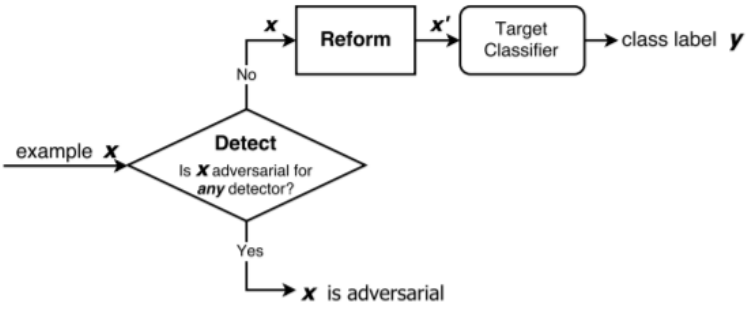

Рис. 13: Схема работы `MagNet`. Детектор используется для определения величины модификации входного объекта X. Если она слишком велика, `MagNet` отказывает объекту в классификации, иначе – объект подается в преобразователь, который проецирует входной объект X в объект X' из входного многообразия. Далее, объект X' подается на вход классификатору, обученному на чистых данных. Рисунок взят из [23].

Преобразователь тоже задается архитектурой автоэнкодера. Он учится минимизировать разницу между входным естественным объектом X и его реконструкцией, получившейся в результате прохода по автоэнкодеру. Соответствующая функция потерь:

$$\mathcal{L} = \frac{1}{N}\sum_{i=1}^{N} ||X_i - ae(X_i)||_2^2,$$

где N – размер обучающей выборки, X_i – элемент этой выборки. Таким образом, для преобразователя стараются добиться максимальной обобщающей способности. Тогда при небольших модификациях объекта преобразователь сможет аппроксимировать его ближайшим объектом среди естественных.

Схема работы `MagNet` приведена на рис. 13

`MagNet` показал высокое качество защиты от black-box атак. Однако его эффективность значительно падает в случае борьбы с white-box атаками. Дело в том, что описанный `MagNet` полностью дифференцируем, а значит, к нему применимы все градиентные white-box методы для построения модифицированных объектов. Для борьбы с white-box атаками авторы [23] предлагают добавить в модель несколько детекторов и несколько преобразователей и обучить их независимо. Далее, на этапе тестирования для каждого входного объекта детектор и преобразователь будут выбираться случайным образом. Это затруднит работу злоумышленников при построении модифицированного объекта, так как им также нужно будет предсказать, какой детектор и какой преобразователь будет выбран для их атаки. Хотя свойство `transferability` может помочь им и в этом случае.

4.6 Выводы

Защита от состязательных атак – это открытая проблема, для которой пока так и не предложено единое решение, несмотря на большой интерес научного сообщества к этой теме. Основные трудности, с которыми сталкиваются при защите, связаны со следующими утверждениями [16]:

- Построить теоретическую модель процесса создания модифицированного объекта для атаки крайне сложно. Данные объекты являются, вообще говоря, решениями нелинейной и невыпуклой задачи оптимизации. В настоящий момент не предложено хороших теоретических инструментов для описания решений таких задач, поэтому теоретически обосновать корректность и работоспособность того или иного метода защиты от состязательных атак очень трудно.
- Наличие модифицированных объектов и произвольность их модификации требуют, чтобы модели машинного обучения работали хорошо при любых возможных входных данных. Такая чрезмерная устойчивость приводит к значительному ухудшению качества моделей.

5 Заключение

В данной работе была рассмотрена проблема устойчивости нейронных сетей к так называемым состязательным атакам, проводимым с помощью модифицированных объектов. Модифицированные объекты – это объекты, полученные небольшим по величине возмущением естественного объекта и предназначенные для обмана нейронной сети: выход сети на этих объектах неверен и противоречит здравому смыслу. Для модифицированных объектов известно свойство `transferability`, позволяющее использовать один и тот же объект для проведения атак сразу на множество независимых моделей классификации. В данной работе даны определения основных понятий, связанных с состязательными атаками. Также здесь рассмотрены основные методы построения атакующих объектов для атак уклонения и средства защиты от этих атак. Проанализированы плюсы и минусы защитных стратегий.

Состязательные атаки представляют реальную угрозу для практического применения машинного обучения, и в частности нейронных сетей, в задачах, требующих приватности данных и надежности работы. Несмотря на существование большого количества стратегий защиты от атак, ни одна из них не является универсальным средством, спасающим от

всего. Защита от состязательных атак остается открытой проблемой и требует создания новых подходов к построению устойчивых моделей.

Список литературы

[1] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, Rob Fergus, *Intriguing properties of neural networks,* arXiv preprint arXiv:1312.6199, 2013.

[2] Alexey Kurakin, Ian Goodfellow, Samy Bengio, *Adversarial Machine Learning at Scale,* arXiv prepring arXiv:1611.01236, 2016.

[3] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, Yoshua Bengio, *Generative Adversarial Networks,*, arXiv preprint arXiv:1406.2661, 2014.

[4] Seyed-Mohsen Moosavi-Dezfooli, Alhussein Fawzi, Omar Fawzi, Pascal Frossard, *Universal adversarial perturbations*, IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017.

[5] Nicolas Papernot, Patrick McDaniel, Somesh Jha, Matt Fredrikson, Z. Berkay Celik, Ananthram Swami, *The Limitations of Deep Learning in Adversarial Settings*, 2016 IEEE European Symposium on Security and Privacy (EuroSP), pp. 372–387, 2016.

[6] Geoffrey Hinton, Oriol Vinyals, Jeff Dean, *Distilling the Knowledge in a Neural Network,* arXiv prepring arXiv:1503.02531, 2015.

[7] Nicolas Papernot; Patrick McDaniel; Xi Wu; Somesh Jha; Ananthram Swami et al., *Distillation as a Defense to Adversarial Perturbations Against Deep Neural Networks*, 2016 IEEE Symposium on Security and Privacy (SP), pp. 582-597, 2016,.

[8] Nicolas Papernot, Patrick McDaniel, *Extending Defensive Distillation*, arXiv preprint arXiv:1705.05264 , 2017.

[9] Pouya Samangouei, Maya Kabkab, Rama Chellappa, *Defense-GAN: Protecting Classifiers Against Adversarial Attacks Using Generative Models,* arXiv prepring arXiv:1805.06605, 2018.

[10] Nicholas Carlini, Anish Athalye, Nicolas Papernot, Wieland Brendel, Jonas Rauber, Dimitris Tsipras, Ian Goodfellow, Aleksander Madry, Alexey Kurakin, *On Evaluating Adversarial Robustness,* arXiv preprint arXiv:1902.06705, 2019.

[11] Ian J. Goodfellow, Jonathon Shlens, Christian Szegedy, *Explaining and Harnessing Adversarial Examples,*, arXiv preprint arXiv:1412.6572, 2014.

[12] Nicolas Papernot, Patrick McDaniel, Ian Goodfellow, *Transferability in Machine Learning: from Phenomena to Black-Box Attacks using Adversarial Samples,* arXiv preprint arXiv:1605.07277, 2016.

[13] Rainer Martin StornKenneth Price, *Differential Evolution: A Simple and Efficient Adaptive Scheme for Global Optimization Over Continuous Spaces*, Journal of Global Optimization 23(1), 1995.

[14] Jiawei Su, Danilo Vasconcellos Vargas, Sakurai Kouichi, *One pixel attack for fooling deep neural networks,* EEE Transactions on Evolutionary Computation, Vol.23 , Issue.5 , pp. 828–841, 2019.

[15] Jiawei Su, Danilo Vasconcellos Vargas, Kouichi Sakurai, *Attacking Convolutional Neural Network using Differential Evolution,* arXiv preprint arXiv:1804.07062 , 2018.

[16] Ian Goodfellow and Nicolas Papernot, *Is attacking machine learning easier than defending it?*, статья в блоге http://www.cleverhans.io/security/privacy/ml/2017/02/15/why-attacking-machine-learning-is-easier-than-defending-it.html , 2017

[17] Papernot, Nicolas and McDaniel, Patrick and Goodfellow, Ian and Jha, Somesh and Celik, Z. Berkay and Swami, Ananthram, *Practical Black-Box Attacks against Machine Learning,* Association for Computing Machinery, pp. 506-519, 2017.

[18] Nina Narodytska, Shiva Kasiviswanathan et al., *Simple Black-Box Adversarial Attacks on Deep Neural Networks,* IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), pp. 1310-1318, 2017.

[19] Florian Tramer, Alexey Kurakin, Nicolas Papernot, Ian Goodfellow, Dan Boneh, Patrick McDaniel, *Ensemble Adversarial Training: Attacks and Defenses,* arXiv preprint arXiv:1705.07204, 2017.

[20] Nicholas CarliniDavid Wagner, *Towards Evaluating the Robustness of Neural Networks,* Conference: 2017 IEEE Symposium on Security and Privacy (SP), pp. 39-57, 2017.

[21] Hossein Hosseini, Yize Chen, Sreeram Kannan, Baosen Zhang, Radha Poovendran, *Blocking Transferability of Adversarial Examples in Black-Box Learning Systems,* arXiv preprint arXiv:1703.04318, 2017.

[22] Maya Kabkab, Pouya Samangouei, and Rama Chellappa, *Task-aware compressed sensing with generative models,* AAAI Conference on Artificial Intelligence, 2018.

[23] Dongyu Meng, Hao Chen, *MagNet: a Two-Pronged Defense against Adversarial Examples,* the 2017 ACM SIGSAC Conference, 2017.