Excessive invariance

МФТИ, 2020 год
Чубчева Александра

1 Введение

Машинное обучение, и в частности нейронные сети, имеют уязвимости в безопасности. Существуют алгоритмы генерации так называемых adversarial examples (далее - атаки). Они представляют собой входные векторы, на которых нейронная сеть выдает неправильный результат. В статье — приведен метод защиты от подобных атак. Я попыталась воспроизвести этот метод в упрощенном виде. В этом отчете изложены результаты проделанного эксперимента.

2 Метод атаки

В работе исследуются классификаторы изображений. Авторы статьи предполагали, что классификатор может быть инвариантен к некоторой части информации. А именно, если взять промежуточное представление входной картинки и подменить часть информации, получится успешная атака на нейронную сеть.
Промежуточное представление картинки z размерности n делится на две части: z_n и z_s. Семантическое подпространство z_s с размерностью C, равной числу классов, считаем основным источником информации для классификатора, на основе которой он выдает ответ. Составляющая из Nuisance подспространства размерности $n - C$ не оказывает влияния на результат работы классификатора. Это значит, что подмена этой части вектора не повлияет на ответ, выданный сетью.

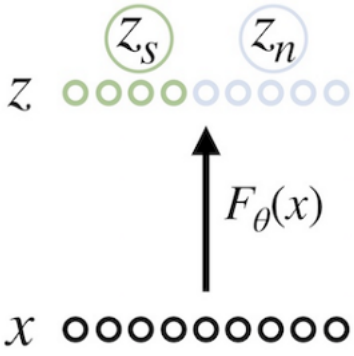

Figure 1: логиты z_s содержат информацию о классе, к логитам z_n сеть инвариантна

Такую атаку удалось сгенерировать как на MNIST, так и на Imagenet. Для этого рассматривались две картинки, исходная и целевая, принадлежащие разным классам. Логиты z_n исходной градиетнтым спуском приближались к логитам целевой картинки, z_s при этом оставались неизменными. В результате искуственно созданная картинка принадлежала к исходному классу, который определяется исключительно z_s. При

этом оказалось, что внешний вид изображения в основном определяет составляющая z_n, и визуально сгенерированная атака была похожа на целевую картинку. Следующая иллюстрация это поясняет.

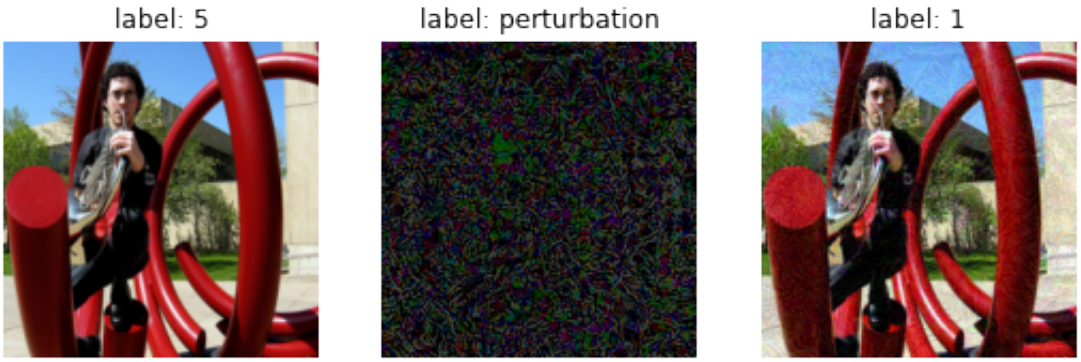

Figure 2: Пример атаки, сгенерированной на imagenet

3 Метод защиты

Атаки - это прекрасно, но хотелось бы уметь защищаться от них. Авторы предложили свой метод тренировки классификатора. Они использовали структуру, похожую на схему domain adaptation.

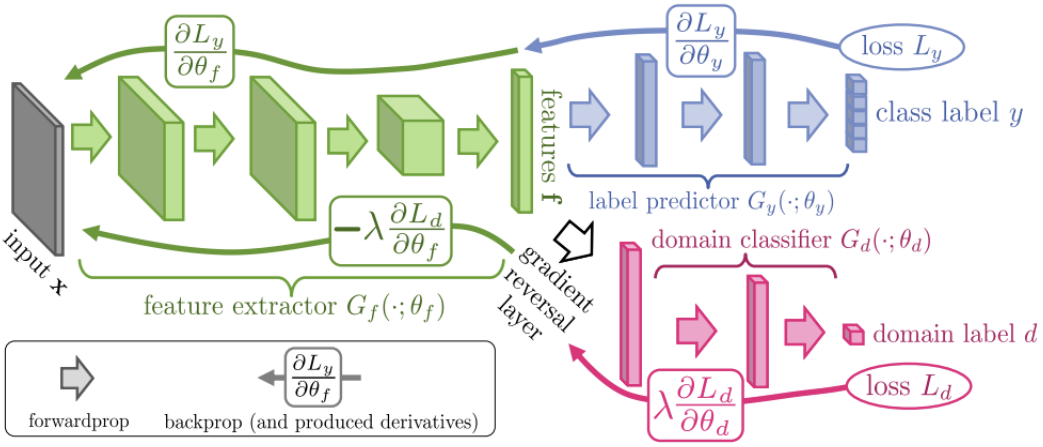

Figure 3: Схема модели, использующей domain adaptation

Сначала нужно получить сжатое представление входной картинки. В статье авторы использовали обратимую сеть Revnet, мы же взяли простой автоэнкодер. На этом сжатом представлении тренируются два классификатора: первый использует z_s и он должен давать правильный ответ, второй использует z_n и "выжимает" оттуда информацию. Это означает, что второй классификатор должен в идеале иметь accuracy $\frac{1}{C}$, то есть фактически отвечать наугад. Для тренировки авторы предложили свою функцию потерь, названную independence cross-entropy loss:

$$L = \sum_{i=1}^{C} -y_i \log F_\theta^{z_s}(x)_i + \sum_{i=1}^{C} \log D_\theta^{z_n}(F_\theta^{z_s}(x)_i$$

Первое слагаемое - cross entropy первого классификатора, второе слагаемое - бинарная cross entropy второго классификатора, взятая с противоположным знаком (хотим максимизировать ее, чтобы второй классификатор предсказывал как можно хуже).
В статье утверждается, что сеть, натренированная таким методом, будет более устойчива к adversarial examples.

4 Результаты

В своей работе я использовала несколько упрощенную модель, а именно автоэнкодер вместо revnet для получения сжатого представления и датасет MNIST. Для проверки эффективности метода я сравнивала работу двух моделей. Первая - простой классификатор, принимающий на вход z_s и использующий cross-entropy loss. Вторая модель построена по схеме, приведенной выше, содержит два классификатора и использует independence cross-entropy loss.
Эксперимент имел следующую схему: выбираются две произвольные картинки из разных классов. Начинаем генерировать атаку градиентным спуском и на каждой его итерации проверяем ответы сравниваемых классификаторов. Считаем, что атака успешна, если классификатор выдал ответ, отличный от класса исходной картинки.
Об успешности метода domain adaptation можно говорить в том случае, если классификатор с domain adaptation показывает успех атаки на более поздних итерациях, чем простой классификатор.
На гистограммах показано распределение номера итерации, начиная с которого атака начала действовать.

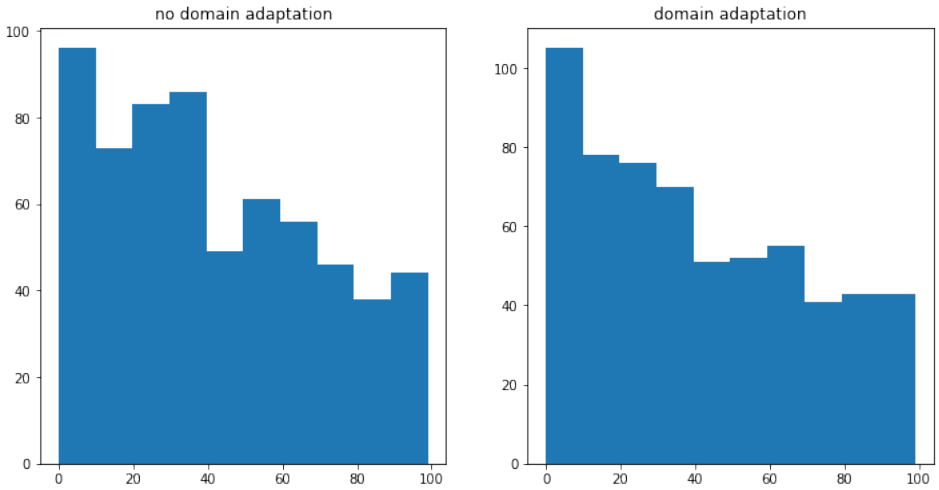

Figure 4: Пример полученных диаграмм

Как видно, модели с domain adaptation и без показывают примерно одинаковые результаты. На данном примере подтвердить эффективность метода из статьи не удалось.