Министерство образования и науки Российской Федерации

Санкт-Петербургский политехнический университет Петра Великого

1. —
2. Институт кибербезопасности и защиты информации

**КУРСОВОЙ ПРОЕКТ**по дисциплине «Основы искусственного интеллекта»

|  |  |  |
| --- | --- | --- |
| Выполнил  студент гр. 4851001/90202 |  | Судаков Д. И. |
| Проверил преподаватель | <*подпись*>    <*подпись*> | Крундышев В. М. |
|  |  |  |

Санкт-Петербург

2022

**Содержание**

[Введение 3](#_Toc120574694)

[1 Теоретическая часть 4](#_Toc120574695)

[1.1 Наивный Байесовский классификатор 4](#_Toc120574696)

[1.2 Jacobian Saliency Map attack (JSMA) 6](#_Toc120574697)

[2 Практическая часть 9](#_Toc120574698)

[2.1 Тестирование ИИ 9](#_Toc120574699)

[2.2 Реализация атаки JSM 14](#_Toc120574700)

[2.3 Реализация защиты ИИ 20](#_Toc120574701)

[3 Вывод 22](#_Toc120574702)

[4 Список литературы 23](#_Toc120574703)

## **Введение**

Искусственный интеллект (ИИ), как правило, базируется на машинном обучении и способен решить множество задач из реального мира. В то время, как размышление, принятие решений и т.п. сравнительно со способностями человеческого мозга у машин далеки от идеала, в недавнее время было сделано несколько важных открытий в области технологий ИИ и связанных с ними алгоритмов. Важную роль играет увеличивающееся количество доступных для обучения ИИ больших выборок разнообразных данных.

Область ИИ пересекается со многими другими областями, включая математику, статистику, теорию вероятностей, физику, обработку сигналов, компьютерное зрение, психологию, лингвистику и науку о мозге. Мотивация развития технологий ИИ состоит в том, что задачи, зависящие от множества переменных факторов, требуют очень сложных решений, которые трудны к пониманию и сложно алгоритмизируются вручную.

Современные технологии машинного обучения и ИИ совместно с правильно подобранными и подготовленными «тренировочными» данными для систем могут позволить научить компьютеры «программировать» не хуже, чем человек.

# **1 Теоретическая часть**

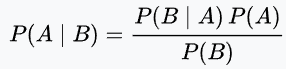
## **1.1 Наивный Байесовский классификатор**

Наивный байесовский классификатор — это вероятностный алгоритм машинного обучения, основанный на применении теоремы Байеса и используемый в самых разных задачах классификации.

Теорема Байеса — это простая математическая формула, используемая для вычисления условных вероятностей.

Условная вероятность — это вероятность наступления одного события при условии, что другое событие уже произошло.

Формула для определения условной вероятности:



Она показывает, как часто происходит событие A при наступлении события B, обозначается как P(A|B) и имеет второе название «апостериорная вероятность». При этом мы должны знать:

как часто происходит событие B при наступлении события A, что обозначается в формуле как P(B|A);

какова вероятность того, что A не зависит от других событий, обозначаемая в формуле как P(A);

какова вероятность того, что B не зависит от других событий. В формуле она обозначается как P(B).

Вкратце можно сказать, что теорема Байеса — это способ определения вероятности исходя из знания других вероятностей.

Основным допущением наивного байесовского алгоритма является то, что каждая характеристика вносит независимый и равный вклад в конечный результат.

Далее рассмотрим эти допущения на примере.

* Предположим, что среди параметров нет зависимых друг от друга. Например, цвет не имеет никакого отношения к типу или стране производства автомобиля. Отсюда первое допущение, что эти параметры являются независимыми;
* Второе допущение: у каждого параметра одинаковая важность. Например, зная только цвет автомобиля, нельзя предсказать страну его производства. Ведь ни один из параметров не имеет большего или меньшего значения по сравнению с другими, и все вносят равный вклад в результат.

Также допущения наивного байесовского алгоритма, как правило, некорректны в реальных ситуациях. Допущение о независимости всегда некорректно, но часто хорошо работает на практике. Поэтому алгоритм и называется наивным. [1]

**Типы наивного байесовского классификатора:**

1. Мультиномиальный метод

Здесь векторы признаков представляют собой значения частотности, то есть частоту, с которой генерируются те или иные события посредством мультиномиального распределения. Это модель событий, обычно используемая для классификации документов;

1. Метод Бернулли

В многомерной модели событий Бернулли характеристики являются независимыми логическими значениями (двоичными переменными), которыми описываются входные данные. Подобно мультиномиальной модели, эта модель широко применяется в задачах классификации документов, где используется не частотность термина (т. е. частота встречаемости слова в документе), а бинарные характеристики встречаемости терминов (т. е. встречается слово в документе или нет);

1. Метод Гаусса

Предполагается, что непрерывные значения всех характеристик имеют распределение Гаусса, то есть нормальное распределение. При нанесении на график получается колоколообразная кривая, симметричная относительно средней значений характеристик.

Наивные байесовские алгоритмы часто используются при анализе эмоциональной окраски текстов, фильтрации спама, в рекомендательных системах и т.д. Они легко и быстро внедряются, но их самый большой недостаток заключается в сложности соблюдения требования о независимости предикторов.

# **1.2 Jacobian Saliency Map attack (JSMA)**

Атака на карту значимости, основанная на значениях матрицы Якоби представляет собой семейство методов состязательной атаки для обмана моделей классификации. Алгоритм пытается изменить как можно меньше пикселей. Для достижения этой цели он использует карту значимости, которая показывает влияние каждого пикселя на результат классификации. Большое значение означает, что изменение этого пикселя окажет значительное влияние на результат классификации. Алгоритм JSMA выбирает наиболее важный пиксель на основе карты значимости и изменяет его, чтобы увеличить вероятность целевого класса атаки. Процесс повторяется до тех пор, пока сеть не будет обманута или не будет достигнуто максимальное количество измененных пикселей (в этом случае атака была неудачной). [2]

Метод JSM работает в три этапа:

1. **Вычисление прямой производной DNN**

Пусть F обозначает функцию, изученную DNN во время обучения. Матрица Якоби получается производной от F по входным пикселям. Если F — вектор вероятности, полученный после применения слоя softmax для m-классов, а X — входные данные, то матрица Якоби будет такой, как показано на рисунке 4.

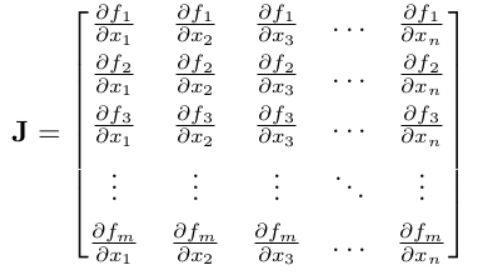


Рисунок 4 – Матрица Якоби

Запись (i,j) в матрице Якоби показывает скорость изменения вероятности для класса i относительно входного пикселя xj входа.

1. **Построение состязательных карт значимости**

После того, как матрица Якоби вычислена, следующим шагом является вычисление состязательной карты значимости. Эта карта указывает, какие входные функции должны быть нарушены, чтобы сеть можно было эффективно обмануть. Не все пиксели при возмущении будут способствовать неправильной классификации, и мы хотим возмущать как можно меньше пикселей, чтобы получить неправильную классификацию. Карта значимости выбирает пиксели, которые с наибольшей вероятностью будут способствовать неправильной классификации. Карта существенности вычисляется для каждого входного признака xi по следующей формуле:

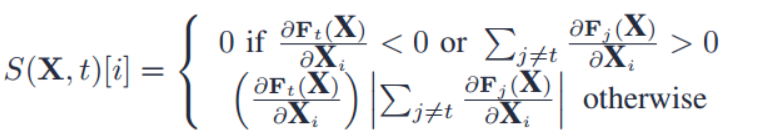


Рисунок 5 - Вычисление карты значимости состязательности

Интуитивно, приведенная выше формула выбирает только те пиксели, возмущение которых увеличивает вероятность того, что изображение будет ошибочно классифицировано как целевой класс, и в то же время снижает вероятность для всех других классов.

1. **Изменить пиксели**

Вычислив состязательную карту значимости, нужно выбрать пиксель с наивысшим значением значимости и возмутить этот пиксель на величину тета.

Алгоритм повторяет шаги 1-3 до тех пор, пока модель не классифицирует входные данные как целевой класс или пока не будет изменено максимальное количество пикселей. Ниже приведено несколько изображений, которые после атаки ошибочно классифицируются искусственным интеллектом как класс «7».



Рисунок 6 - Примеры изображений после JSMA

От подобных атак существует множество методов защиты. В данной работе будет применен метод случайного зашумления данных. Суть данного метода заключается в том, что перед каждым слоем свертки как на этапе обучения, так и на этапе тестирования добавляется шумовой слой, что увеличивает стойкость ИИ к различным атакам и в том числе JSM. [3]

# **2 Практическая часть**

## **2.1 Тестирование ИИ**

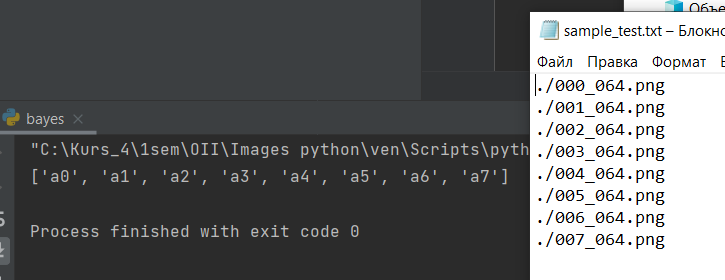
 За основу был взят наивный байесовский классификатор, который предсказывает, чьё лицо изображено на фотографии. Он использует мультиномиальный метод, то есть признаки (пиксели в данном случае) представляют собой значения частотности, относительно которых строится мультиномиальное распределение. В процессе обучения для каждой фотографии вычисляется распределение. Затем во время тестирования также вычисляется распределение, а затем оно сравнивается со всеми возможными вариантами, полученными при обучении. При наибольшей схожести распределений делается вывод о принадлежности объекта(фото) к классу(человеку). На рисунке 7 показан пример работы наивного байесовского классификатора.

Рисунок 7 – Пример работы наивного байесовского классификатора

ИИ обучался на 64 фотографиях 8 разных людей. Названия фотографий состоят из двух трехзначных чисел, разделенных символом ‘\_’, первое число – принадлежность к классу(человеку), второе число – порядковый номер фотографии в этом классе. В обучающем наборе названия классов соответствуют порядку изображений, то есть изображения 000\_000.png – 000\_063.png принадлежат классу a0; 001\_000.png – 001\_063.png – классу a1 и так далее. Затем ИИ был проверен на фотографиях, которые не использовались при обучении. Рассматривая рис.7, можно убедиться, что ИИ работает корректно.

Рисунок 8 – Вероятность ошибки 1-го рода

Рисунок 9 – Вероятность ошибки 2-го рода (a7)

Рисунок 10 – Вероятность ошибки 2-го рода (a6)

Рисунок 11 – Вероятность ошибки 2-го рода (a5)

Рисунок 12 – Вероятность ошибки 2-го рода (a4)

Рисунок 13 – Вероятность ошибки 2-го рода (a3)

Рисунок 14 – Вероятность ошибки 2-го рода (a2)

Рисунок 15 – Вероятность ошибки 2-го рода (a1)

Рисунок 16 – Вероятность ошибки 2-го рода (a0)

## **2.2 Реализация атаки JSM**

Суть данной атаки заключается в том, что вычисляются пиксели, которые имеют наибольшую значимость при обучении ИИ. Так как в наивном байесовском классификаторе нет зависимости и приоритетов признаков (в нашем случае пикселей), то вычисление карт значимости не имеет смысла и для реализации атаки необходимо лишь изменять пиксели таким образом, чтобы частота конкретных пикселей больше не позволяла классификатору отличать лица.

Для эмуляции атаки была реализована программа, которая случайным образом изменяет определенное количество пикселей. На вход программа получает число – процент от общего числа пикселей для изменения. То есть если подать на вход число 10, то 10% пикселей данного изображения будут изменены на случайные пиксели. Идея заключается в том, что, меняя пиксели, изменяется частотность всех пикселей, следовательно, изменяется распределение. Значит вероятность того, что распределение будет похоже на какое-нибудь другое и ИИ ошибочно идентифицирует человека возрастает, и атака, соответственно, с большей вероятностью будет успешной. Также область изменения вычислялась с помощью нормального распределения с матожиданием в центре изображения и следовательно – лица. Данное решение было реализовано с целью увеличения эффективности атаки, ведь так изменяемая часть изображения будет больше влиять на распределение, потому что, условно говоря фон, волосы, уши и прочие атрибуты на краях изображений есть почти на всех фото и изменение этих пикселей будет равномерно влиять на все распределения, эффективнее же изменять одно конкретное распределение, на основе которого нейронная сеть чаще делает вывод. На рисунках 17 и 18 было изменено 10% и 50% пикселей соответственно.



Рисунок 17 – Изменение 10% пикселей



Рисунок 18 – Изменение 50% пикселей

Изображение выглядит как люди, ткань

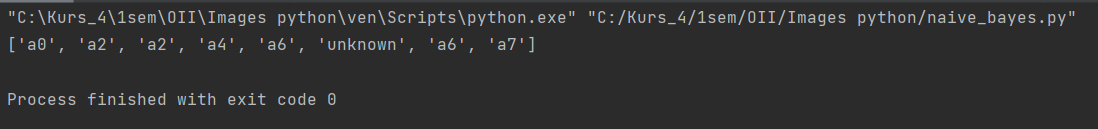
Автоматически созданное описаниеТакже для наглядности в набор обучающих данных ИИ была добавлена фотография, состоящая на 100% из случайных пикселей (рисунок 19). Идея заключается в том, что при изменении достаточно большого числа пикселей исходного изображения распределения оказываются ближе к «белому шуму», чем к любой из фотографий с лицами. Данное изображение при обучении было помечено как ‘unknown’.

Рисунок 19 – Изменение 100% пикселей

Изображение выглядит как текст, в позе, человек, женский

Автоматически созданное описание Далее была протестирована атака. На первом фото было изменено 5% пикселей, на втором 6%, третьем 10%, четвертом 50%, пятом 51%, шестом 65%, а на седьмом и восьмом ничего не было изменено. На рисунке 20 показаны фото после изменения пикселей. Результат определения ИИ представлен на рисунке 21.

Рисунок 20 – Изменение пикселей

Рисунок 21 – Результат ИИ после атаки

На основании тестирования можно сделать вывод о том, что при изменении менее чем 5-7% пикселей ИИ с большой вероятностью продолжает правильно определять человека на фото. При изменении 7-15% пикселей наивный байесовский классификатор уже достаточно часто допускает ошибки. Далее, соответственно, чем больше процент измененных пикселей, тем выше вероятность ошибки. Также стоит отметить, что при изменении больше половины пикселей ИИ начинает относить распределение на атакованном изображении к распределению, принадлежащему классу unknown (рис. 10).

Далее, класс ‘unknown’, отвечающий за “белый шум” был удалён. Ниже приведена таблица зависимости результатов классификации от процента измененных пикселей подаваемого на вход изображения.

Таблица 1 – Зависимости результатов классификации от процента измененных пикселей подаваемого на вход изображения

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Класс  % | a0 | a1 | a2 | a3 | a4 | a5 | a6 | a7 |
| 0 | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| 10 | 0 | 0 | 5 | 3 | 4 | 5 | 6 | 7 |
| 20 | 2 | 0 | 5 | 2 | 4 | 2 | 0 | 7 |
| 30 | 2 | 0 | 5 | 2 | 4 | 2 | 0 | 0 |
| 40 | 2 | 0 | 5 | 2 | 4 | 2 | 0 | 0 |
| 50 | 2 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |
| 60 | 0 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |
| 70 | 2 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |
| 80 | 0 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |
| 90 | 0 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |
| 100 | 0 | 0 | 5 | 2 | 5 | 2 | 0 | 0 |

Изучив таблицу можно сделать вывод о том, что 0-ой класс ошибочно выявляется как 2-ой класс, причем если изменить более 50% пикселей, он начнет правильно идентифицироваться. 1-ый, 6-ой и 7-ой классы ошибочно выявляются как 0-ой класс. 3-ий и 5-ый классы ошибочно выявляются как 2-ой класс, причем 2-ой класс выявляется как 5-ый. 4-ый класс оказался самым стойким, так как он начинает ошибочно относиться к 5-ому классу только при изменении более 50% пикселей.

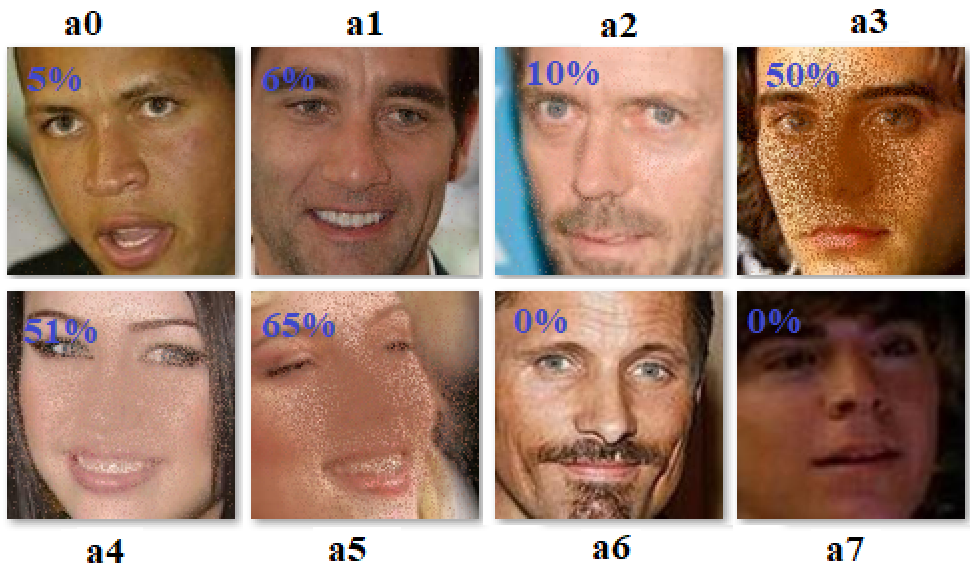
Стоит отметить, что для реальной атаки замена пикселей на случайные не подходит, так как нарушается свойство скрытности. В связи с данным обстоятельством, скрипт атаки был модифицирован таким образом, чтобы исходные пиксели в атакуемом изображении заменялись на пиксели, отличающиеся от предыдущих не более чем на 10% в каждом из 3-х значений палитры RGB.

Рисунок 22 – Свойство скрытности атаки

Далее приведена таблица, показывающая для каждого класса, какой минимальный процент пикселей необходимо изменить в атакуемом изображении, чтобы ИИ допустил ошибку при классификации.

Таблица 2 – Граничные значения для ошибочной идентификации

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Класс | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| % | 20±3 | 3±2 | 2±1 | 19±3 | 44±3 | 12±3 | 17±3 | 23±3 |

На основании таблицы 1 был сделан график, на котором видно, к каким классам ИИ чаще всего относит изображения при ошибке. Таким образом, можно наблюдать, сколько раз ошибочно был выявлен 0 класс, 1 класс, 2 класс и т.д. График приведен на рисунке 23.

Рисунок 23 – График ошибочно выявленных классов

По приведенному графику можно сделать вывод, что, как правило, ИИ ошибочно относит полученное на вход изображение к нулевому, второму и пятому классу, к остальным же классам ИИ никогда на практике не относит данное изображение ни при одном варианте атаки.

# **2.3 Реализация защиты ИИ**

Реализация защиты основана на том, что во всём наборе данных, на котором обучается наивный байесовский классификатор, будет заранее изменен определенный процент пикселей. То есть если изменить 10% пикселей у всех изображений из набора данных, то во время атаки распределение уже не будет заметно отличаться от распределения в обучающих изображениях и с бо́льшим успехом ИИ правильно определит класс. В этой части класс, отвечающий за “белый шум” был также удалён. Затем для каждого подаваемого на вход изображения было выбрано, сколько процентов пикселей необходимо изменить, чтоб ИИ точно ошиблась:

* 0 класс: изменено 20% пикселей
* 1 класс: изменено 5% пикселей
* 2 класс: изменено 2% пикселей
* 3 класс: изменено 19% пикселей
* 4 класс: изменено 48% пикселей
* 5 класс: изменено 12% пикселей
* 6 класс: изменено 17% пикселей
* 7 класс: изменено 27% пикселей

Результат классификации при такой конфигурации атаки представлен на рисунке 24.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 24 – Ошибочное выявление ИИ

Изображение выглядит как текст

Автоматически созданное описаниеЗатем заранее было изменено 10% пикселей во всех изображениях обучающего набора данных. Результат классификации при той же конфигурации атаки после применения защиты представлен на рисунке 25.

Рисунок 25 – Попытка защиты ИИ путем зашумления 10% пикселей

Таким образом, при изменении 10% пикселей получилось защитить 2, 5, 6, 7 классы от данного варианта атаки. Затем то же самое было сделано с изменением 20% и 30% пикселей соответственно. Результаты представлены на рисунках 26-27 соответственно.

Изображение выглядит как текст

Автоматически созданное описание

Рисунок 26 – Попытка защиты ИИ путем зашумления 20% пикселей

Изображение выглядит как текст, часы, устройство, темный

Автоматически созданное описание

Рисунок 27 – Попытка защиты ИИ путем зашумления 30% пикселей

В результате можно сделать вывод, что при зашумлении в 20% пикселей удалось также защитить и 3-ий класс, а в случае с 30% даже 1-ый класс, однако 0-ой и 4-ый классы защитить так и не удалось. Скорее всего это связано с более сильной чувствительностью к отдельным пикселям из-за слишком большой разницы между изображениями одного класса при обучении.

# **3 Вывод**

В ходе выполнения данной работы были изучены и применены на практике знания об искусственном интеллекте и машинном обучении с точки зрения безопасности.

Для исследования были выбраны следующие механизмы: наивный байесовский классификатор в качестве ИИ, JSM в качестве алгоритма атаки и зашумление данных в качестве защиты. Уже на этапе тестирования модели выяснялось, что такая комбинация атаки и классификатора является не самой удачной, так как метод наивного байеса не предполагает наличия наиболее значимых пикселей, которые вычисляются в алгоритме JSM, однако данный факт не помешал выяснить наиболее уязвимые области изображений с лицами и успешно произвести атаку. Также при проведении изменении пикселей был учтён принцип скрытности, то есть в большинстве случаев атака проводится практически незаметно для пользователя.

Результаты применения такой защиты, как зашумление данных, оказались наиболее удачными, хоть и не безупречными - большую часть классов удалось защитить от реализованной атаки, однако 2 класса всё же остались чувствительны к атакам. Это объясняется неудачным набором обучающих данных для этих классов: на всех фото хоть и изображен один и тот же человек, но фон, аксессуары, ракурсы и т.д. сильно контрастируют от одного фото к другому, что приводит к сильной чувствительности ИИ к каждому конкретному пикселю, что делает зашумление данных менее эффективным.

Таким образом, можно сделать вывод о том, что выбор обучающих данных крайне важен для безопасности ИИ. Причем это не только уменьшает шанс ошибок первого рода и стойкость к атакам, но и позволяет более успешно применять различные механизмы защиты, такие как, например, зашумление данных.

# **4 Список литературы**

[1] Geoffrey I Webb; Naïve Bayes; Encyclopedia of Machine Learning and Data Mining; 2017, 3, 556-576.

[2] Combey, T.; Loison, A.; Faucher, M.; Hajri, H. Probabilistic Jacobian-Based Saliency Maps Attacks*.*Mach. Learn. Knowl. Extr. 2020, 2, 558-578.

[3] Christian Meurisch and Max Mühlhäuser. 2021. Data Protection Via Noise in AI Services: A Survey. ACM Comput. Surv. 54, 2, Article 40, 2021, 3, 559-593.