РЕФЕРАТ

Выпускная квалификационная работа бакалавра состоит из 41 страницы, 4 рисунков, 6 таблиц, 17 используемых источников и 2 приложений.

АКУСТИЧЕСКИЙ КЕЙЛОГГЕР, КЛАВИАТУРНЫЕ АТАКИ, КЛАСТЕРИЗАЦИЯ, КЛАСТЕРИЗАЦИЯ КОНТЕКСТУАЛЬНЫХ ДАННЫХ, СКРЫТАЯ МАРКОВСКАЯ МОДЕЛЬ, АУДИО-ПРИЗНАКИ, КЛАССИФИКАЦИЯ БЕЗ УЧИТЕЛЯ

Работа состоит из введения, трех глав и заключения. Темой данной работы является кластеризация контекстуальных аудиоданных на примере звуков клавиатуры с помощью обучения без учителя. Целью данной работы является воспроизведение предыдущих результатов работ в области акустических клавиатурных атак без учителя и их улучшение с помощью современных подходов машинного обучения и представленной в данной работе нейросетевой архитектуры *Continuous Bag of Samples*, призванной улучшить качество кластеризации контекстуальных данных.

В первой главе описываются основные алгоритмы которые используются для выделения аудио признаков, снижения размерности и кластеризации данных, а так же некоторые предыдущие работы в области акустических клавиатурных атак.

Во второй главе описывается наша архитектура акустической клавиатурной атаки, в частности, представленная в данной работе новая нейросетевая архитектура *Continuous Bag of Samples* для улучшения результатов кластеризации контекстуальных данных.

В третьей главе описываются результаты наших экспериментов в кластеризации звуков клавиш и результаты восстановления напечатанного на клавиатуре текста из аудиозаписи печати, в том числе на разных клавиатурах и с разными микрофонами. Так же в ней описываются ограничения используемого подхода и предложения по возможным улучшениям в последующих работах на данную тему.

В нашей работе мы добились значительного улучшения распознавания напечатанного текста по звуку его печати по сравнению с предыдущими работами а так же представили нейросетевую архитектуру *Continuous Bag of*

Samples которая хорошо показала себя в задачи улучшения кластеризации контекстуальных данных.

ГЛОССАРИЙ

MFCC – Мел-кепстральные коэффициенты (mel-frequency cepstral coefficients).

FFT – Быстрое преобразование Фурье (fast Fourier transform).

HMM – Скрытая Марковская модель (hidden Markov model).

UMAP – Uniform Manifold Approximation and Projection.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

В современном мире, где значительная часть нашей жизни проходит онлайн: в социальных сетях, видеозвонках, удаленной учебе и работе, кибератаки представляют особую угрозу. Зная пароль только от вашей почты, злоумышленник может получить доступ к вашим накоплениям, узнать где вы живете и, возможно, получить секретную рабочую информацию.

В то же время мы сами даем злоумышленникам большинство информации которая им нужна. Сейчас им не обязательно взламывать вашу почту и узнавать где вы живете, если можно просто посмотреть фотографии в ваших соцсетях. Если не фильтровать информацию, которую мы публикуем в сети или же то что мы показываем в наших видеозвонках, мы подвергаем себя большой опасности.

Уже давно рассматривается возможность так называемых акустических атак, когда человек может узнать интересующую его информацию по звукам, которые издает тот или иной предмет. Так по звукам, которые издает клавиатура, можно понять что на ней было напечатано, в том числе пароли, номера и коды банковских карт. В 2000-х годах, когда о таких атаках впервые начали говорить, они не представляли большой опасности. Чтобы получить запись печати на клавиатуре злоумышленнику пришлось бы установить вам прослушивающее устройство, однако если бы у него был физический доступ к вашим месту жительства и компьютеру он бы смог получить интересующие его данные более надежным способом, нежели восстановлением текста из записи звука клавиатуры. Тогда как сейчас мы окружены сотней микрофонов. Кроме того что злоумышленник может взломать плохо защищенные записывающие устройства, мы сами даем им всю необходимую информацию. Как уже и говорилось выше любой видеозвонок, видеозапись выложенная в онлайн или стрим может стать источником утечки ваших паролей.

Так, можно выделить наиболее уязвимую категорию людей к таким атакам – онлайн стримеры. Они ведут онлайн трансляции того как играют в игры, красят фигурки, решают задачи. У них всегда включен микрофон, они разговаривают со зрителями и, что самое важное, часто печатают что-то на клавиатуре, в том числе пароли и детали банковских карт для покупок в сети.

Тема работы: кластеризация контекстуальных аудиоданных на примере звуков клавиатуры с помощью обучения без учителя.

Цель работы: воспроизведение предыдущих результатов работ в области акустических клавиатурных атак без учителя и их улучшение с помощью современных подходов машинного обучения и представленной в данной работе нейросетевой архитектуры *Continuous Bag of Samples*, призванной улучшить качество кластеризации контекстуальных данных.

Актуальность: из-за увеличения присутствия каждого человека в онлайне в последние несколько лет, а особенно после пандемии covid-19, опасность акустических клавиатурных атак сильно возрастает. Если такие атаки возможны, то для того чтобы от них эффективно защититься нужно знать как они устроены и какие у них есть слабые стороны. Кроме того нейросетевая архитектура textitContinuous Bag of Samples, представленная в данной работе, может быть переменена для улучшения кластеризации в других областях, например, в области распознавания речи.

В данном дипломе нами были воспроизведены и улучшены результаты предыдущих работ по распознаванию текста напечатанного на клавиатуре по его звуку, в частности представлен новый способ для улучшения кластеризации контекстуальных данных textitContinuous Bag of Samples и использованы современные алгоритмы понижения размерности данных, ранее не применяемые для решения данной задачи.

Полученные в данном дипломе результаты не предусматривают применение их в корыстных целях, а лишь являются примером того чего можно добиться в данной области с применением современных технологий. Все полученные результаты исследования можно использовать для защиты от таких акустических атак.

1. ОБЗОР ПРЕДМЕТНОЙ ОБЛАСТИ

1.1. Выделение признаков звука

Выделение признаков (feature extraction) – процесс преобразования данных, снижающий их размерность и предоставляющий более управляемое и информативное (с точки зрения полезного сигнала) их представление.

Аудиофайл хранит в себе информацию о частоте и амплитуде звука. Эта информация обычно состоит из N записей амплитуды. Число записей амплитуды в 1 секунду называется частотой дискретизации. Из такого представления аудиофайла можно восстановить исходную звуковую волну и воспроизвести ее на аудиомониторах.

Звуковая волна хранит в себе всю информацию о звуке, однако для решения задач по классификации или кластеризации она не подходит. Для работы со звуком нужно для начала преобразовать звуковую волну во что-то более информативное для этих алгоритмов, то есть выделить какие-то полезные для решаемой задачи признаки из этой звуковой волны.

Так, основным методом выделения признаков из звуковой волны является *быстрое преобразование Фурье* (FFT). Быстрое преобразование Фурье позволяет вычислять дискретное преобразование Фурье за время $O(n \log n)$. Таким образом мы можем получить частотное распределение для определенного малого отрезка звука, то есть его частотный спектр. Из подряд идущих спектров можно построить, так называемую спектрограмму (рисунок 1.1).

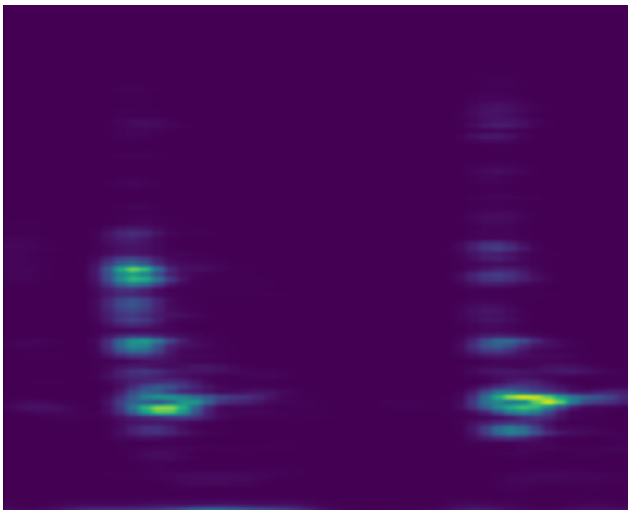

Рисунок 1.1 — Пример спектрограммы звука нажатия клавиши

Спектрограмма – это изображение, показывающее зависимость спектральной плотности мощности сигнала от времени.

Так, спектрограммы широко используются в сфере распознавания голоса [2] и многих других областях машинного обучения, не говоря уже о сфере стандартной звуковой обработки, подавление шума и т.д..

Стоит упомянуть, что спектрограммы редко используются в их чистом виде. Чаще всего их преобразовывают мел-фильтрами в *мел-спектрограммы*. Мел-спектрограмма это обычная спектрограмма у которой по оси Y не частота в Гц, а мелы.

Кроме спектрограмм, в сфере распознавания голоса повсеместно используются *мел-кепстральные коэффициенты (MFCC)*. Они строятся применением мел-фильтров к *кепстру*. *Кепстр (cepstrum)* [3] – функция обратного преобразования Фурье от логарифма спектра мощности сигнала. Такое преобразование может показаться странным, однако мел-кепстральные коэффициенты использованные в качестве аудио признаков показывают эффективность в решении многих задачах, превышающую таковую при использование мел-спектрограмм.

Для получения признаков звуков клавиши используются как мел-спектрограммы [8], так и MFCC [13]. Однако, как было показано в работе *Keyboard acoustic emanations revisited* [13], применение MFCC для задачи различия звуков разных клавиш подходит лучше мел-спектрограмм, что было еще раз доказано в нашей работе.

1.2. Кластеризация

В основе задачи по определению напечатанного на клавиатуре текста по издаваемому ею звуку лежит кластеризация звуков отдельных клавиш. Дадим определение термину кластеризация:

Кластеризация - задача по группировке объектов множества на подмножества. Объекты из одного подмножества (кластера) должны иметь больше общих признаков чем объекты из разных подмножеств.

Постановка задачи кластеризации

Пусть X – множество объектов, Y – множество идентификаторов (классов). На множестве X задана функция расстояния между объектами $\rho(x, x')$. Необходимо разбить множество X на такие подмножества, для которых расстояние между объектами одного класса будет меньше расстояния между объектами разных классов по метрике ρ.

Для решения этой задачи применяют алгоритмы кластеризации.

Алгоритмы кластеризации

Алгоритм кластеризации – функция $a : X \to Y$, которая любому объекту из множества $x \in X$ ставит в соответствие метку $y \in Y$.

Рассмотренные выше определения относятся к, так называемой, *четкой* (hard) кластеризации. Однако существует еще *нечеткая* (soft или fuzzy) кластеризация. При нечеткой кластеризации каждый объект может принадлежать сразу к нескольким классам и функция кластеризации a приобретает вид $a : X \to Y$, где $Y \in \mathbb{R}^N$, N– количество классов и $y \in Y$ описывает вероятностное распределение принадлежности $x \in X$ к одному из N классов.

Применение методов нечеткой кластеризации не показало должных результатов в наших экспериментах, поэтому сконцентрируемся на алгоритмах четкой кластеризации (далее просто кластеризации).

Самым распространенным алгоритмом кластеризации является метод *K-средних* (K-Means) [7, 12]. Он стремится минимизировать суммарное квадратичное отклонение точек кластеров от центров этих кластеров:

$$V = \sum_{i=1}^{k} \sum_{x \in S_i} (x - \mu_i)^2 \tag{1.1}$$

где k – число кластеров,
S_i – полученные кластеры,
$i = 1, 2, \ldots, k$,
а μ_i – центры масс всех векторов x из кластера S_i.

Из-за того что класс точки по сути определяется по ближайшему к ней центру масс, метод K-средних не предусматривает того что 1 кластер

будет лежать внутри другого в исходном пространстве множества X. Это ограничивает применение этого алгоритма в некоторых задачах, но несмотря на это для наших целей он подходит.

Кроме того стоит отметить что результаты кластеризации зависят от начальной инициализации центров масс, которые потом будут подвинуты для минимизации выражения 1.1. Из-за этой недетерминированности, для получения более хороших результатов стоит запускать этот алгоритм несколько раз и выбирать из получившихся кластеризаций лучшую.

Метрики качества кластеризации

Для того чтобы определить насколько хорошо была произведена кластеризация используются метрики качества кластеризации. Их можно разделить на 2 основных типа:

– **Внешние меры оценки качества** Данные меры используют дополнительные знания о кластеризуемом множестве: распределение по кластерам, количество кластеров и т.д.

– **Внутренние меры оценки качества** Данные меры оценивают качество структуры кластеров опираясь только непосредственно на нее, не используя внешней информации.

Внешние меры оценки качества гораздо информативнее и говорят о том насколько хорошо мы в действительности смогли кластеризировать данные. Однако когда мы кластеризируем данные для решения какой-либо прикладной задачи, мы не знаем ответа заранее, поэтому внешние меры подходят только для валидации.

Внутренние же меры оценки качества основываются на том, насколько сформированными оказались кластеры, как сильно они сгруппированы, на том какое расстояние между кластерами. В общем случае мы хотим чтобы расстояние между кластерами было максимальным, а расстояние между точками в каждом кластере минимальным.

К популярным внешним мерам оценки качества относят:

- Индекс Rand;

- Индекс Adjusted Rand;
- Homogeneity score.

К внутренним:

- Calinski Parabasz score [4];
- Silhouette Score [11].

В нашей работе мы будем использовать *homogeneity score* и *индекс rand* как основные метрики качества кластеризации, так как исходя их экспериментов, они больше всего влияют на последующее качество восстановленного текста.

Во время же решения задачи, для определения лучшей кластеризации из нескольких попыток больше подходит *Silhouette Score*, так как, опять же, исходя из экспериментов, высокое значение Silhouette Score зачастую ведет к увеличению индекса rand и homogeneity score, чего не наблюдается в должной мере при использовании Calinski Parabasz score.

1.3. Снижение размерности

Для улучшения качества кластеризации многомерных данных (сотни и более измерений) часто применяют снижение размерности данных.

Снижение размерности – это преобразование данных, заключающиеся в снижение количества переменных. При этом в лучшем случае мы хотим сохранить всю информацию исходных данных, или же потерять *минимальное* количество полезной информации.

Постановка снижения размерности

Пусть $X \in \mathbb{R}^N$ – множество размерности N, тогда преобразование $a : X \in \mathbb{R}^N \to Y \in \mathbb{R}^M$, где $M < N$ – преобразование снижения размерности.

Снижение размерности может быть разделено на 2 основных типа:

- **Отбор признаков** Выделение самых значащих переменных. Все остальные просто не используются.
- **Выделение признаков** Создание нового набора переменных на основе исходных переменных, размер которого меньше исходного.

Рассмотрим основные методы снижения размерности.

Метод главных компонент

Метод главных компонент (PCA) – один из основных способов уменьшить размерность данных, потеряв наименьшее количество информации. Является методом линейного снижения размерности данных.

Метод состоит в поиске n главных компонент множества точек $x_1, x_2, \ldots, x_m \in \mathbb{R}^n$. Первая главная компонента – это такая линия L_n, для которой выполняется условие:

$$\sum_{i=1}^{m} \operatorname{dist}^2(x_i, L_n) \to \min . \tag{1.2}$$

Ортонормированный вектор a_1 называется вектором первой главной компоненты и однозначно задает L_n, для которой выполняется условие выше.

Для поиска всех главных компонент можно воспользоваться методом Фреше:

1. Централизуются данные (вычитанием среднего):

$$x_i := x_i - \overline{X}. \tag{1.3}$$

   Теперь $\sum_{i=1}^{m} x_i = 0$.

2. Отыскивается первая главная компонента как решение задачи:

$$a_1 = \underset{\|a_1\|=1}{\operatorname{argmin}} \left(\sum_{i=1}^{m} \|x_i - a_1(a_1, x_i)\|^2 \right). \tag{1.4}$$

   если решение не единственно, то осуществляется выбор одного из них.

3. Из данных вычитается проекция на первую главную компоненту:

$$x_i := x_i - a_1 \left(a_1, x_i\right) . \tag{1.5}$$

4. Отыскивается вторая главная компонента как решение задачи:

$$a_2 = \underset{\|a_2\|=1}{\operatorname{argmin}} \left(\sum_{i=1}^{m} \|x_i - a_2(a_2, x_i)\|^2 \right). \tag{1.6}$$

Если решение не единственно, то выбирается одно из них.

Далее процесс продолжается, то есть на шаге $2k-1$ вычитается проекция на $(k-1)$-ю главную компоненту (к этому моменту проекции на предшествующие $(k-2)$ главные компоненты уже вычтены).

На шаге $2k$ определяется k-я главная компонента как решение задачи:

$$a_k = \underset{\|a_k\|=1}{\operatorname{argmin}} \left(\sum_{i=1}^{m} \|x_i - a_k(a_k, x_i)\|^2 \right) \tag{1.7}$$

(если решение не единственно, то выбирается одно из них).

Пространство построенное из первых главных компонент несет в себе наибольшее количество информации. Это основано на гипотезе о наличии «сигнала» (сравнительно малая размерность, относительно большая амплитуда) и «шума» (большая размерность, относительно малая амплитуда). С этой точки зрения метод главных компонент работает как фильтр: сигнал содержится в основном в проекции на первые главные компоненты, а в остальных компонентах пропорция шума намного выше.

Однако, если «сигнал» не сильно сосредоточен в первых главных компонентах, для сохранения информации из оригинального пространства, необходимо брать очень много главных компонентов, что означает, что мы не сможем сильно понизить размерность данных. Так PCA не подходит для данных с очень большими размерностями.

Хоть PCA и зарекомендовал себя как самый надежный и популярный метод снижения размерности, в нашей задаче размерность признаков звука клавиши имеет 1000 измерений. Эксперименты показали, что при уменьшении размерности с помощью только PCA не удается добиться приемлемых результатов кластеризации.

t-SNE

t-SNE или стохастическое вложение соседей с t-распределением [14] – это алгоритм машинного обучения для снижения размерности, в частности для визуализации многомерных данных (снижение размерности до 2-х 3-х компонент). Является алгоритмом нелинейного снижения размерности.

Алгоритм t-SNE состоит из двух главных шагов.

1. Сначала t-SNE создаёт распределение вероятностей по парам объектов высокой размерности таким образом, что похожие объекты будут выбраны с большой вероятностью, в то время как вероятность выбора непохожих точек будет мала.
2. Затем t-SNE определяет похожее распределение вероятностей по точкам в пространстве малой размерности и минимизирует расстояние Кульбака—Лейблера между двумя распределениями с учётом положения точек.

Исходный алгоритм использует евклидово расстояние между объектами, как базу измерения сходства, однако зачастую для приемлемого результата нужно выбрать метрику более подходящую для конкретной задачи. Так для сравнения векторов признаков звука больше подходит метрика L1, нежели L2 (евклидово расстояние).

Опыт показывает, что t-SNE очень хорошо справляется со своей задачей. Так, если применить t-SNE к набору рукописных цифр MNIST, и понизить размерность данных до 2-х, на графике точки принадлежащие одним классу будут находиться рядом.

Однако у t-SNE есть свои минусы:

- t-SNE достаточно медленный. Существуют методы оптимизации этого алгоритма для случая понижения размерности до 2-х или 3-х измерений, однако если у нас есть много данных и мы хотим понизить размерность до, скажем, 60 измерений, алгоритм может работать до нескольких часов.
- t-SNE не сохраняет информацию о расстоянии между кластерами, что делает его выбор для задачи снижения размерности для последующей кластеризации спорным.

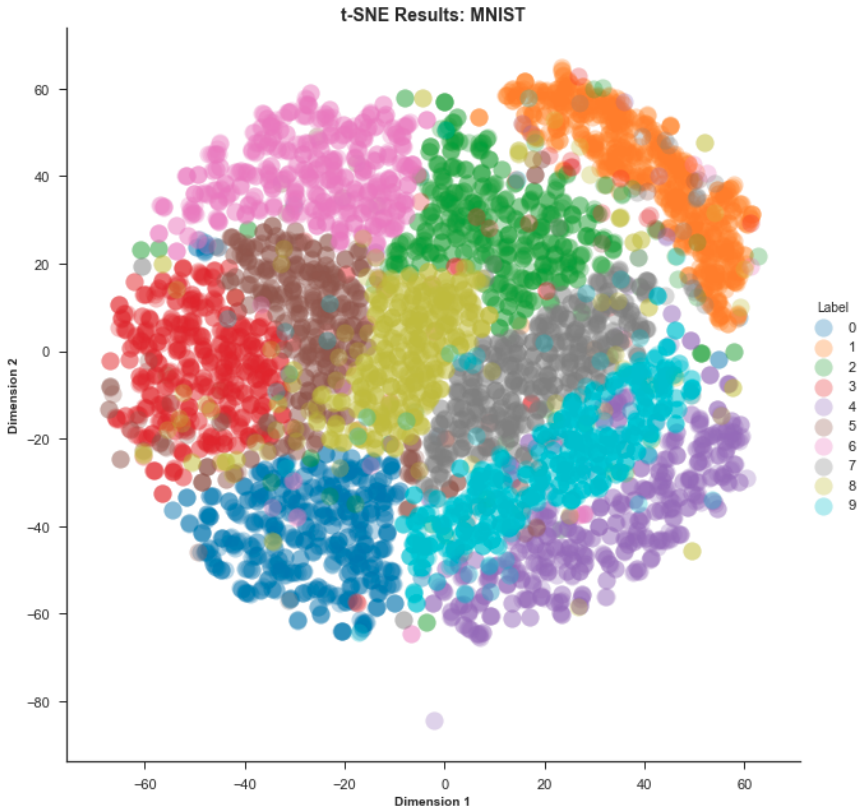

Рисунок 1.2 — Визуализация MNIST с помощью t-SNE

Из-за этих минусов t-SNE чаще всего используют только для визуализации данных и их анализа.

Uniform Manifold Approximation and Projection

Uniform Manifold Approximation and Projection (UMAP) [9] – еще один алгоритм нелинейного снижения размерности данных.

UMAP был создан в 2018 году, как попытка получить алгоритм, который работает так же хорошо как t-SNE, но лишен его главных минусов, а точнее алгоритм который работает быстрее и сохраняет расстояние между кластерами.

Так UMAP почти для всех задач показывает чуть более хорошие результаты чем t-SNE в визуализации данных, а также его можно применять как алгоритм для понижения размерности с целью последующей кластеризации такими алгоритмами как K-Means, что является ключевым его преимуществом над t-SNE для нашей задачи.

Кроме того, как и заверялось создателями, UMAP работает значительно

быстрее t-SNE на большом наборе данных и хорошо масштабируется для снижения размерности до 10 и более измерений.

1.4. Акустические клавиатурные атаки

Акустические клавиатурные атаки – разновидность клавиатурных атак с целью получения напечатанного текста по звуку, исходящему от клавиатуры при его печати.

В первой значимой работе по этой теме – *Keyboard acoustic emanations* [1] было показано, что звуки каждой клавиши на клавиатуре компьютера, банкомата и т.д. немного различаются, и что можно использовать эти различия для определения нажатой клавиши по ее звуку. В этой работе исследователи применяли быстрое преобразование Фурье для получения признаков нажатия клавиш и натренировали нейросеть для их классификации.

Однако такой тип атаки можно отнести к *атаке с учителем*, так как для того чтобы натренировать классификатор у нас должен быть набор данных на котором мы будет тренировать данный классификатор, то есть нужно иметь физический доступ к клавиатуре. Если набор данных звуков клавиатур банкоматов можно получить непосредственно от самого банкомата, то чтобы получить размеченные данные о звуке клавиатуры, нужно иметь доступ к этой клавиатуре, что не всегда представляется возможным.

Гораздо опаснее выглядят *атаки без учителя*. Для таких атак нужны только неразмеченные данные, то есть для восстановления текста по звуку печати нужен только звук этой печати. Такие атаки могут быть куда опаснее, так как злоумышленникам не обязательно будет иметь физический доступ к вашей клавиатуре для совершения такой атаки.

В работе *Keyboard acoustic emanations revisited* [13] был впервые предложен метод акустической клавиатурной атаки без учителя. Он состоит из 6 основных эпапов:

1. Выделение звуков клавиш в исходной аудиозаписи;
2. Выделение вектора признаков для каждой клавиши;
3. Кластеризация звуков;

4. Применение скрытой Марковской модели (HMM) для получения знаков из полученных кластеров;
5. Применение лингвистической модели для исправления ошибок допущенных на предыдущем этапе;
6. Тренировка классификатора по отфильтрованным данным полученным на предыдущем этапе, и его применение на всех звуках клавиш.

Последний этап повторяется несколько раз, так как после классификации мы получаем большую точность распознавания, а значит можем использовать еще больше данных и еще лучше натренировать классификатор.

Этим методом исследователи смогли добиться 92% точности классификации клавиш и 96% точности после применения к финальному результату классификации лингвистической модели для исправления ошибок.

Наше исследование основывается на этой работе и улучшает процесс кластеризации алгоритмами машинного обучения не доступными исследователям оригинальной работы, а так же предлагает совершенно новый метод улучшения кластеризации контекстуальных данных, какими и являются звуки клавиш при печати текста.

Большинство последующих работ по этой теме [5, 8] фокусировались на задаче классификации, то есть на атаках с учителем.

Кроме методов основанных на различие звука каждой клавиши есть и другие способы определить нажатую клавишу. Так в работе *Keyboard Acoustic Triangulation Attack* [6] был предложен способ определения клавиши по ее положению в пространстве. Положение нажатой клавиши определяется задержкой звука от момента нажатия клавиши, до момента попадания в 2 микрофона стоящих рядом с клавиатурой. Такая атака представляется наиболее маловероятной, так как для ее реализации нужно иметь доступ к сразу двум микрофонам жертвы. Но не смотря на это, способ достаточно интересный и заслуживает внимания.

2. АРХИТЕКТУРА АКУСТИЧЕСКОЙ КЛАВИАТУРНОЙ АТАКИ

2.1. Обзор архитектуры

Наш метод акустической атаки основан на работе *Keyboard acoustic emanations revisited* [13] и использует те же основные этапы с некоторыми важными изменениями, в частности на этапе кластеризации и классификации.

Основные этапы

1. **Выделение звуков клавиш из исходной аудиозаписи.** На этом этапе мы должны выделить из всей аудиозаписи звуки только нажатий клавиш. Этот этап почти полностью повторяет такой же этап из исходной работы. Для решения этой задачи мы используем оконную сумму спектральной мощности. Данный алгоритм не идеален, так он не может отличить громкий сторонний звук от звука клавиши. Кроме того, если скорость печати высока, данный алгоритм либо пропускает клавиши, либо же при других настройках «находит» больше звуков клавиш, чем было напечатано. В оригинально работе результаты полученные на данном этапе подвергались дополнительной ручной правке. В нашей работе этого не проводилось для проверки того каких результатов распознавания можно добиться без какой-либо ручной доработки. Улучшение этого этапа может значительно улучшить качество распознавания исходного текста в «полевых условиях».
2. **Выделение признаков из звука нажатия клавиш.** На этом этапе мы находим MFCC звуков клавиш и понижаем размерность их признаков. Выбор MFCC вместо FFT был обоснован в секции 1.1. В отличие от оригинальной работы, размерность нашего вектора признаков гораздо больше, так как мы рассматриваем более большой промежуток времени и используем 100 MFCC коэффициентов, тогда как в оригинальной работе было 16. Наши эксперименты с использованием UMAP показывают значительное улучшение качества кластеризации с использованием большего числа MFCC коэффициентов. В оригинальной статье не было

использовано понижение размерности, хотя и было упомянута возможность использования PCA. Мы же используем UMAP для уменьшения размерности признаков, который вместе с увеличенным числом MFCC коэффициентов и позволяет достичь результатов кластеризации сильно превосходящих таковые в оригинальной работе.

3. **Кластеризация звуков.** После уменьшения размерности вектора признаков при помощи UMAP мы применяем новый, представленный в данной работе, метод улучшения кластеризации контекстуальных данных Continuous Bag of Samples (CBoS). После его применения к данным мы производим кластеризацию с помощью алгоритма K-Means.
4. **Применение скрытой Марковской модели (HMM) для получения знаков из полученных кластеров.** Данный этап почти не отличается от аналогичного в оригинальной работе. С помощью дискретной HMM, используя матрицу переходов основанную на статистики языка распознаваемого текста мы получаем исходный текст из кластеров полученных на предыдущем этапе. Матрицу переходов мы инициализируем не только по статистике биграмм большого корпуса текста, мы добавляем в матрицу переходов специальный символ $ который означает отсутствие буквы, на тот случай если на этапе выделения звуков клавиш из общей аудиозаписи были допущены ошибки. Кроме того была автоматизирована инициализация матрицы вероятностей эмиссии с помощью кластеризации преобразованных на предыдущем этапе данных на 2 класса. Из за того что у большинства клавиатур звук пробела сильно отличается от всех остальных, такая кластеризация достаточно точно отделяет звуки пробелов, а значит мы можем узнать какие кластеры с наибольшей вероятностью были порождены пробелом. В оригинальной работе это делалось вручную. Автоматическая инициализация матрицы вероятностей влияет только на скорость обучения HMM и вероятности того что она что-то распознает, но не на точность.
5. **Применение лингвистической модели.** После получения предварительной версии текста в нем исправляются ошибки при помощи словаря и статистики биграмм большого корпуса текста. В отличие от оригинальной работы используется модель биграмм, а не триграмм, для

улучшения результатов на сложных текстах, так как триграммы сильно страдают от проблемы нулевого вхождения даже с достаточно большим корпусом текста на несколько миллионов слов.

6. **Тренировка классификатора.** На данном этапе мы используем текст с исправленными на предыдущем этапе ошибками как разметку наших данных и тренируем на них классификатор. Мы размечаем только те данные, в которых мы больше уверены. Для этого мы берем из текста только те слова которые есть в словаре, а значит в них менее вероятно есть ошибки. Кроме того на первом проходе мы берем только слова длинной в 4 и более символов, так как вероятность того что слово из 4-х символов которое оказалось в словаре будет содержать случайную ошибку распознавания ниже чем у слов с меньшим количеством букв. В нашей работе мы используем в качестве классификатора «случайный лес» вместо линейного дискриминантного анализа из оригинальной работы, так как по результатам наших экспериментов он показывает более хорошие результаты классификации
7. **Цикл обратной связи.** Мы повторяем шаги 4 и 5 по очереди до тех пор, пока точность не перестанет расти. На данном этапе точность классификатора составляет примерно 86-97% (для разных клавиатур).

Теперь более подробно расскажем об отдельных этапах и частях нашего подхода, которые отличаются от оригинальной работы.

2.2. Выделение признаков и понижение размерности

Мы используем MFCC для получения вектора признаков из звуковой волны. Мы берем первые 100 MFCC коэффициентов с десятью шагами в 110 семплов с окном в 1024 семпла на одноканальном аудиосигнале с частотой дискретизации 41000 Гц. Десяти шагов хватает для того чтобы захватить как звук касания клавиши, так и звук ее нажатия. Подробнее о составляющих звука клавиши было описано в *Keyboard acoustic emanations* [1]. В оригинальной работе было использовано только 16 MFCC коэффициентов, однако данные, представленные в первой части таблицы 2.1, демонстрирующей ре-

зультаты кластеризации алгоритмом K-Means для первых 16 и первых 100 MFCC коэффициентов, показывают что увеличение количества MFCC коэффициентов значительно улучшают результаты кластеризации, что приводит к более качественному восстановлению исходного напечатанного текста.

Кроме того стоит упомянуть что на качество полученных признаков сильно влияют посторонние звуки на записи, такие как шум микрофона или шум компьютерного вентилятора. Для улучшения качества распознавания на записях с заметным шумом можно использовать различные алгоритмы по удалению шума, что мы и делали для некоторых зашумленных записей.

Используя первые 100 MFCC коэффициентов и 10 шагов мы получаем 1000 признаков. Их уже можно попытаться кластеризировать как и было сделано в оригинальной работе, но мы можем лучше. Для начала нужно уменьшить размерность получившегося вектора признаков. Первым что мы попробовали был PCA, однако использование первых 100, 250, 500, 750 и 900 главных компонент почти не дало никакого прироста в качестве кластеризации. Но не смотря на это мы все же смогли улучшить кластеризацию при помощи PCA. Для этого мы поступили не совсем обычно, вместо того чтобы взять n первых главных компонент PCA, мы убрали первую главную компоненту, что привело к значительному приросту в метриках кластеризации. Такой прирост может быть обусловлен с тем что мы исключили измерение с сигналом с самой большой амплитудой, который могло отвечать за громкость или что-то еще, что сильно меняется от нажатия к нажатию, но при этом не помогает отличить звук одной клавиши от другой. Исключение 2-х и более первых главных компонент не дает такого же сильного прироста, так как уже начинает теряться полезная нам информация.

Не смотря на этот положительный прирост в метриках мы все равно не используем PCA. Самые лучшие результаты продемонстрировали понижение размерности с помощью алгоритма UMAP. Так в нашей работе мы снижаем размерность до 32 измерений, что сопровождается очень значительным приростом в метриках кластеризации. Это обусловлено робастностью алгоритма, а так же скорее всего тем фактом, что в отличие от PCA, UMAP является алгоритмом нелинейного снижение размерности. Удаление первой главной компоненты перед использованием UMAP не дает положительных

результатов и только ухудшает метрики кластеризации. Кроме того после исключения первой главной компоненты не получается точно определить звуки пробелов, что необходимо для инициализации матрицы вероятностей эмиссий. Результаты сравнения алгоритмов снижения размерности представлены во второй части таблицы 2.1

Таблица 2.1 — Сравнение качества кластеризации данных алгоритмом K-Means для различных наборов признаков и способов уменьшения размерности

	Adjusted rand score	Homogeneity Score
16 MFCC коэффициентов	0.253	0.61
100 MFCC коэффициентов	**0.326**	**0.742**
MFCC + PCA (без первой главной компоненты)	0.380	0.826
MFCC + UMAP	**0.454**	**0.926**
MFCC + PCA (без первой главной компоненты) + UMAP	0.452	0.916

2.3. Continuous Bag of Samples

После понижения размерности вектора признаков мы получаем уже хорошо кластеризируемые данные. Однако ни один алгоритм кластеризации не берет в расчет контекстуальность данных. Контекстуальнось в данном случае означает что у данных, в нашем случае звуков клавиш, есть порядок, вероятность принадлежности звука к тому или иному классу зависит не только от самого звука, но и от данных идущих до и после него. Это следует из предположения о том, что был напечатан осмысленный текст на каком-либо языке, а у букв языка есть некоторая структурированность. Так после буквы *r* в английском языке скорее будет идти *a*, нежили *h*.

В данной работе мы представляем новую нейросетевую архитектуру *Continuous Bag of Samples* которая направленна на улучшение кластеризации контекстуальных данных.

CBoS работает похоже на *Continuous Bag of Words*(CBoW) для обучения Word2Vec [10]. CBoW используется для получения векторного представления слов, которое будет иметь в себе «смысл» этого слова. Так, если взять вектор для слова *king*, вычесть из него вектор слова *man* и добавить *woman*, то получится вектор наиболее близкий к слову *queen*. Архитектура CBoW состоит из энкодера E и декодера E'. Энкодер кодирует one-hot и вектора слов стоящих рядом с целевым в векторное представление, затем находится средний вектор из всех закодированных слов контекста и он уже декодируется декодером E' с целью получить one-hot представление целевого слова (рисунок 2.1). Если натренировать такую сеть *предсказывать* целевые слова по их контексту, то энкодер E будет хорошо выполнять задачу по преобразованию слов в их векторное представление, где близкие по смыслу слова будут находится рядом в векторном пространстве.

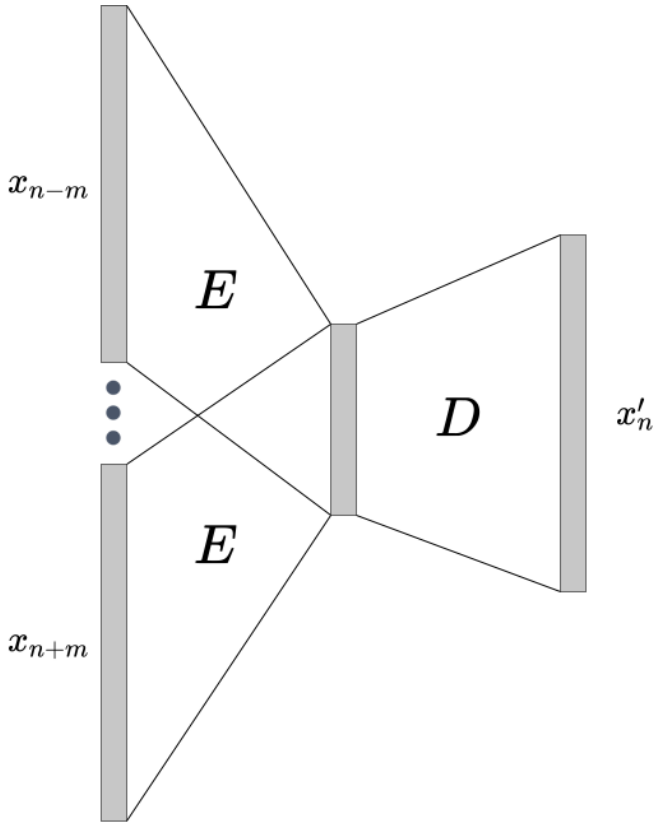

Рисунок 2.1 — Архитектура CBoW. E– энкодер. D– декодер. x– one-hot векторы слов, m– размер контекста

В CBoS мы используем очень похожую архитектуру, однако мы пытаемся по уже имеющимся векторам признаков контекста предсказать целевой вектор признаков (рисунок 2.2). Так если достаточно обучить такую нейросеть

предсказывать целевой вектор признаков по векторам признаков контекста, мы получим энкодер, которые преобразует вектора признаков таким образом, что в получившемся векторном пространстве они кластеризируются лучше чем в исходном.

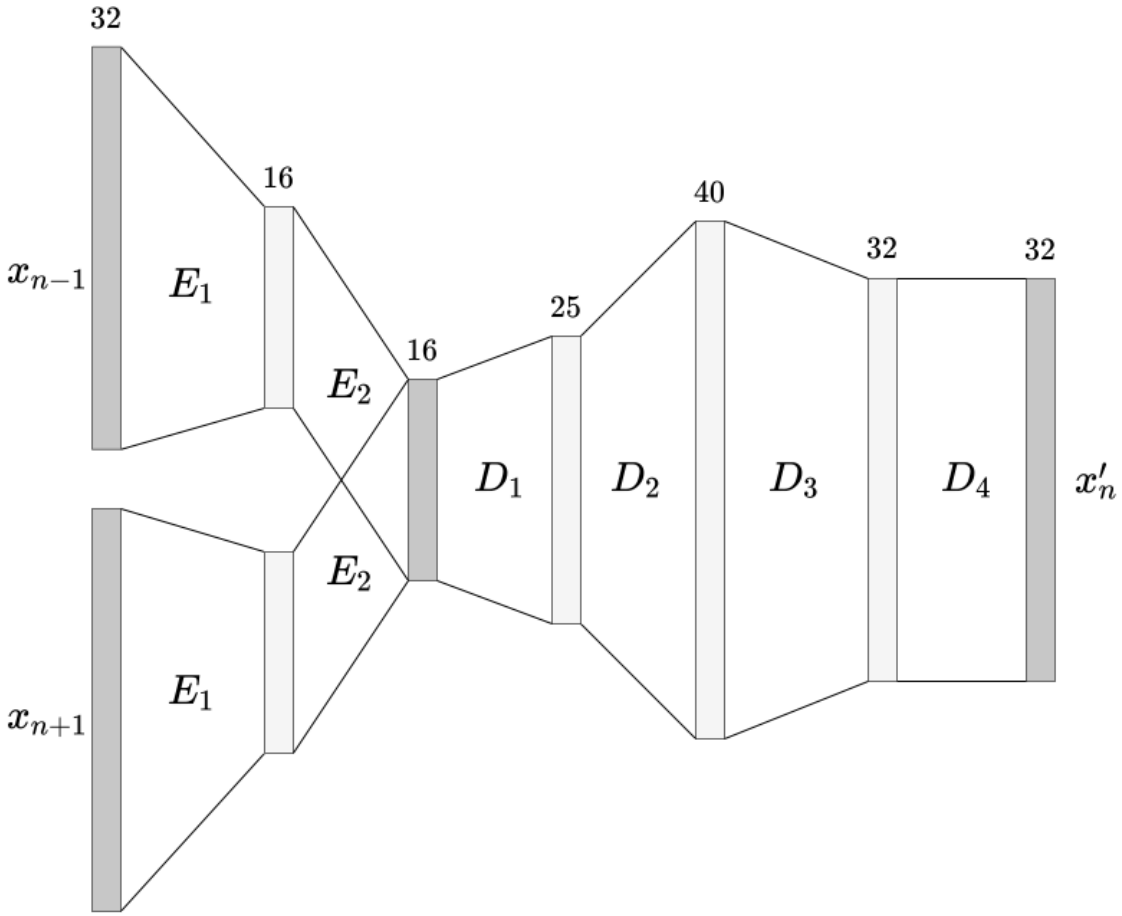

Рисунок 2.2 — Архитектура CBoS. E– полносвязные слои энкодера, D– полносвязные слои декодера, x– векторы признаков.

Для энкодера мы используем 2 полносвязных слоя с функциями активации ELU. Для декодера мы используем 4 полносвязных слоя с функциями активации ELU. Закодированная размерность 16 признаков. В качестве функции ошибки используется $L1$, так как эксперименты показывают, что она дает более хорошие результаты кластеризации чем MSE.

В таблице 2.2 представлены средние значения метрик кластеризации признаков звука клавиш по 30 запускам кластеризации методом K-Means на 50 классов для оригинального векторного пространства, векторного пространства получившегося после применения энкодера из CBoS и векторного пространства получившегося после применения энкодера из автоэнкодера.

Автоэнкодер был добавлен для того чтобы убедиться, что результаты кластеризации были улучшены именно из-за того что нейросеть поняла контекст, а не только из-за еще большего уменьшения размерности данных.

Результат так же зависит от инициализации, иногда нужно перезапустить обучение для достижения более хороших показателей. Так же наблюдается переобучение, поэтому нужно периодически проверять результаты кластеризации или найти подходящее время обучения для решаемой задачи.

Таблица 2.2 — Сравнение качества кластеризации данных алгоритмом K-Means на 50 классов для методов CBoS и Autoencoder с исходным пространством признаков

	Adjusted rand score	Homogeneity Score
Исходные признаки	0.459	0.925
CBoS	**0.615**	**0.931**
Autoencoder	0.449	0.922

2.4. Восстановление текста

Восстановление знаков из классов

После кластеризации наступает этап восстановления знаков из полученных классов. Для этого мы пользуемся скрытой Марковской моделью, где скрытыми параметрами являются знаки текста, а наблюдаемыми – полученные классы. В качестве матрицы переходов используется статистика символьных биграмм на отфильтрованном корпусе википедии [15], состоящем только из маленьких английский букв, точки, запятой и пробела. Отфильтрованный корпус состоит из 10598733 символов. Кроме того в матрицу переходов добавлен символ $ который обозначает отсутствие буквы в случае ошибки на этапе выделения звуков клавиш из аудиозаписи. Вероятность появления этого символа зависит от качества аудиосигнала, скорости печати и качества подбора гиперпараметров для выделения звуков клавиш. В наших экспериментах мы считали вероятность появления $ после любого настоящего символа 0,5% и 8% после самого себя.

Матрица вероятности эмиссии класса определенной буквой получается в процессе обучения HMM EM алгоритмом. Мы автоматически инициализируем матрицу эмиссий, определяя для каких классов наиболее вероятна эмиссия пробела. В оригинальной работе было показано, что это увеличивает шанс на успешное восстановление исходного текста. Для этого мы кластеризируем данные на всего 2 класса. Звуки пробелов очень отличаются от звуков других клавиш, поэтому такая кластеризация хорошо справлялась с таким разделением на всех клавиатурах которые мы использовали в наших экспериментах. Так, тот класс из двух, который имеет меньшее количество вхождений в текст мы считаем пробелом. Далее мы инициализируем матрицу эмиссий таким образом, что чем больше соотношение количества автоматических определенных пробелов ко всем объектам класса, тем больше шанс эмиссии пробела. При такой инициализации EM для HMM в наших экспериментах всегда восстанавливает текст, в то же время без нее, текст успешно восстанавливался только 1 раз из 10-и при удачной случайной начальной инициализации матрицы эмиссий. Остальные вероятности инициализируются случайно, из-за чего при каждом перезапуске алгоритма получается разная точность восстановления. Наиболее вероятная последовательность символов определяется алгоритмом Витерби.

Для определения лучшего результата получено с помощью HMM мы не можем использовать точность, так как при восстановлении текста из звука его печати мы не знаем что конкретно было напечатано. Поэтому вместо точности мы используем следующую аппроксимацию: мы считаем количество букв в тех словах из распознанных HHM, которые нашлись в словаре и имеют длину 4 символа или больше. Шанс получить настоящее слово длинной более трех символов из случайных букв достаточно низкий, поэтому чем лучше HMM справился с задачей восстановления текста, тем больше должно быть таких слов. Это не самый точный способ, так как иногда более точное распознавание дает результат такого подсчета букв меньший чем для распознавания более маленькой точности из-за случайной ошибки в длинном слове, которой не было в восстановлении с чуть меньшей точностью, однако на практике такая аппроксимация работает достаточно хорошо и дает одну из лучших точностей их всех перезапусков алгоритма EM для HMM. Лучший резуль-

тат, полученный на данном этапе, составляет 90% точности восстановления исходного текста.

Лингвистическая модель

Для лингвистической проверки используются словарь и скрытая Марковская модель для биграмм слов. Словарь [16] состоит из 97648 самых используемых английских слов, имен и слов, начинающихся с большой буквы (которые были заменены на такие же слова с маленькой буквой). Статистика биграмм собирается на основе brown, webtext, gutenberg, reuters корпусов текста доступных из Python библиотеки nlkt [17]. Длинна объединенного корпуса текста составляет 4441437 слов.

В скрытой Марковской модели на биграммах в качестве скрытых параметров берутся исправленные слова которые мы хотим получить, в качестве наблюдаемых – слова полученные на предыдущем этапе. Вместо вероятности перехода берется статистика биграмм (ненормализированная). Если словосочетания в биграммах нет, берется число 0.5. Вероятность эмиссии считается так же как и в оригинальной работе перемножением вероятностей изменения букв. В оригинальной работе вероятность сохранения любой буквы равнялась 0.5, а все остальные распределялись равномерно. Мы же инициализируем матрицу вероятности изменения букв по статистике языка, где наиболее частые буквы имеют больший шанс остаться самими собой, чем буквы редко встречающиеся в языке, так как при кластеризации более часто встречающиеся буквы кластеризируются лучше редко встречающихся. Однако данная инициализация используется только после этапа с выделением текста их кластеров. При применении лингвистической модели к результатам классификации используется матрица вероятности изменения букв как в оригинальной работе.

Стоит упомянуть, что в нашей работе мы получаем значительно более маленький прирост в точности после применения лингвистической модели по сравнении с таковым в оригинальной работой. В первою очередь это связано с методом замены слов и плохим выделением звука клавиш из исходной аудиозаписи. В настоящем алгоритме замены слов – слово заменяется **только** на слово такой же длинны. Если же длина слова с ошибкой отличается

от исходного, алгоритм ищет наиболее вероятную замену для слова длины такового с ошибкой, в связи с чем меняются почти все символы. Так в слове «possibly» можно допустить ошибку и не напечатать букву «s», из чего получим «posibly». Однако алгоритм замены слов не сможет восстановить слово «possibly» из-за различия в длине, и по его мнению лучшей заменой будет слово «visibly», что только приведет к увеличению ошибки, чтобы избежать в таком случае замены всего слова мы ставим ограничение на то сколько букв в процентном соотношение от слова можно заменить. В оригинальной работе процесс выделением звука клавиш из исходной аудиозаписи сопровождался ручной проверкой и исправлением результатов алгоритмического определения, из-за чего все длины слов были правильными (если только не была сделана опечатка). В нашем же случае выделение звука клавиш из исходной аудиозаписи полностью автоматическое, что приводит к ошибкам длины слов, что в свою очередь приводит к ухудшению результатов лингвистической модели.

Одним из способов решения этой проблемы проблемы может стать определение более вероятных замен другим алгоритмом, который учитывает возможность изменения длины слова. Таким алгоритмом может быть расстояние Левенштейна, однако его применение выходит за рамки данной работы. Кроме того улучшение алгоритма выделение звука клавиш из исходной аудиозаписи так же должно сильно улучшить результаты, что опять же выходит за рамки данной работы.

Классификация и цикл обратной связи

Для классификации используется алгоритм случайный лес. Данный алгоритм имеет высокую точность для наших, однако он не способен дать вероятностное распределение принадлежности звука к той или иной клавише, необходимой для ускорения подбора паролей, поэтому для подбора паролей нужен алгоритм который способен дать это вероятностное распределение. Таким алгоритмом может быть логистическая регрессия. Финальная точность в таблицах показана для классификации только для случайного леса. В секции

3.2 продемонстрировано сравнение качества классификации разными алгоритмами.

Для тренировки классификатора текст предварительно фильтруется на те знаки в которых мы наиболее уверены. Для букв алфавита мы выбираем все буквы из тех предсказанных слов которые есть в словаре, так как они наиболее вероятно были предсказаны точно. Со знаками препинания и пробелами все сложнее. Из-за того что нельзя проверить их словарем мы просто доверяемся предыдущим шагам и отдаем все знаки препинания и пробелы на классификацию с надеждой на то что классификатор сможет даже с ошибками регуляризировать результаты.

Во время цикла обратной связи мы берем предыдущие результаты классификации, фильтруем их и снова обучаем классификатор. Так при каждом проходе мы все больше увеличиваем точность классификации, а значит увеличиваем как количество, так и качество данных для повторной тренировки классификатора. На этом этапе так же можно применять лингвистическую модель после каждой классификации, что увеличивает количество отфильтрованных данных из того что при исправлении текста с помощью лингвистической модели мы заменяем неправильные слова на те что есть в словаре. Однако на данном этапе лингвистическая модель почти не улучшает результаты распознавания из-за проблем описанных в 2.4.

3.1. Результаты кластеризации и их влияние на точность восстановления текста

Улучшения качества кластеризации влияет на качество восстановленного текста. В таблице 3.1 предоставлены средние результаты точности восстановления текста из кластеров с помощью HMM за 50 запусков, а также средние результаты метрик кластеризации для разных наборов признаков. В данном случае мы использовали кластеризацию на 55 классов.

Таблица 3.1 — Сравнение качества восстановления текста для различных наборов признаков

	Adjusted rand score	Homogeneity Score	Медианная точность	Максимальная точность
MFCC	0.317	0.766	0.720	0.756
MFCC + UMAP	0.426	0.932	0.890	0.901
MFCC + UMAP + CBoS	**0.481**	**0.937**	**0.895**	**0.905**

3.2. Финальные результаты восстановления текста на каждом этапе

В таблице 3.2 представлены результаты восстановления текста после этапа восстановления текста из кластеров с помощью HMM и после этапа цикличного применения классификатора. В таблице для каждого этапа есть строка точности для псевдослучайного текста (без применения лингвистической модели) и для откорректированного лингвистической моделью. Так же в таблице представлены лучшие результаты точности на тех же этапах из работы *Keyboard acoustic emanations revisited* [13]. Как уже говорилось в секции 2.4 наши результаты применения лингвистической модели не дают такого прироста в точности как они давали в оригинальной работе, даже иногда ухудшают процент точности из-за того, что они вручную правили

результаты выделения звука клавиш из исходной аудиозаписи. Но даже не смотря на это, в нашей работе мы на каждом этапе получили результаты восстановления псевдослучайного текста выше, чем в оригинальной работе, а также гораздо более высокие результаты после этапа восстановления текста из кластеров с помощью HMM. Наши лучшие результаты для восстановления псевдослучайного текста даже выше чем результаты применения лингвистической модели в оригинальной работе. Стоит отметить, что данное сравнение не совсем корректно, так как в наших и их экспериментах были разные клавиатуры и микрофоны, что очень сильно может повлиять на результат, поэтому для сравнения мы специально взяли лучшие результаты из нашей и их работы, однако все равно стоит интерпретировать данные результаты сравнения с большой осторожностью.

Таблица 3.2 — Сравнение посимвольной точности восстановления текста с оригинальной работой

		Наши результаты	Результаты оригинальной работы
Восстановление из кластеров	Псевдослучайный текст	**0.902**	0.762
	Лингвистическая модель	**0.912**	0.872
Цикл классификации	Псевдослучайный текст	**0.965**	0.924
	Лингвистическая модель	0.955	**0.963**

В приложении А приведен пример отрывка исходного и восстановленного на разных этапах текста.

Сравнение различных алгоритмов классификации

В таблице 3.3 продемонстрированы результаты классификации различными алгоритмами после фильтрации данных полученных после этапа восстановления текста из кластеров. Приведены результаты для 3-х проходов обучения на предыдущем отфильтрованном результате восстановления и

классификации. Как видно из таблицы – случайный лес демонстрирует себя лучше всего.

Таблица 3.3 — Сравнение качества восстановления текста для различных наборов признаков

	Линейный дискриминантный анализ	Случайный лес	Логистическая регрессия
Первый проход	0.928	**0.944**	0.935
Второй проход	0.931	**0.962**	0.937
Третий проход	0.928	**0.963**	0.937

Так как для эффективного перебора пароля требуется вероятностная классификация, а не четкая, которую дает случайный лес, после цикла классификации алгоритмом случайный лес, на отфильтрованных финальных данных тренируется классификатор логистическая регрессия, который дает точность предсказания **0.95**.

3.3. Сравнение качества восстановления текста с различными условиями записи звука

Нами был напечатан 1 текст на трех различных экспериментальных установках:

1. На механической клавиатуре «Razer Blackwidow Chroma v.2» с использованием качественного конденсаторного микрофона.
2. На механической клавиатуре «Roccat ROC-12-761-BK / MX Black» с использованием микрофона вебкамеры.
3. На клавиатуре ноутбука «MacBook Pro 16 2019» с использованием встроенного в ноутбук микрофона.

Результаты посимвольной точности восстановленного текста для всех условий записи представлены в таблице 3.4. Стоит еще раз напомнить, что низкие результаты для некоторых установок по большей части связаны с качеством выделения звуков клавиш из аудиозаписи, так как из-за этого пропускаются некоторые символы или же наоборот попадают нежелательные звуки как в сами слова, так и между ними. Из-за этого лингвистическая модель не

приносит почти никаких результатов, статистика символьных биграмм для напечатанного текста нарушается, а в классификатор попадают звуки не относящиеся к клавишам. В связи со всем вышеперечисленным, мы наблюдаем наихудшие результаты восстановления текста именно для клавиатуры ноутбука (3-я установка), так как именно с ней были самые большие проблемы при выделении звуков клавиш из исходной аудиозаписи.

На звуке из 2-й экспериментальной установки четко слышен шум микрофона. Для улучшения результатов восстановления исходного текста аудиозапись подверглась удалению шума. Данная предобработка увеличивает финальную точность распознавания примерно на 5%.

Таблица 3.4 — Сравнение качества восстановления псевдослучайного текста с различными условиями записи звука

	1-я установка	2-я установка	3-я установка
Восстановление из кластеров	0.902	0.809	0.7
Цикл классификации	**0.965**	0.936	0.866

3.4. Ограничения настоящего подхода и дальнейшие исследования

Главные ограничения нашего подхода такие же как и в оригинальной работе *Keyboard acoustic emanations revisited* [13]. Наш метод не распознает клавиши модификаторы, такие как *shift*, *caps lock* или *backspace*. Некоторая модификация алгоритма способна начать распознавать и их, так как в наших аудио записях печати, содержащих *shift*, эта клавиша нажималась отдельно от остальных клавиш, а значит ее можно выделить из аудиозаписи и так же класстеризировать и восстановить. Для восстановления нужна будет модификация матрицы переходов в HMM модели, в которую будут включена клавиша *shift*.

Еще одним ограничением для применения нашего метода на практике является плохой алгоритм выделения клавиш из аудиозаписи, о чем мы уже неоднократно говорили. Звуки клавиш достаточно легко визуально определить на спектрограмме, поэтому возможно для задачи их выделения из

аудиозаписи может подойти конвалюционная нейросеть, которую нужно будет обучить на звуках нескольких клавиатур, и, скорее всего, этого будет достаточно для качественного определения звуков для того же типа клавиатур. В любом случае это выходит за рамки этого исследования и достойно отдельной работы, так как может сделать акустическую клавиатурную атаку еще более реализуемой в реальных «полевых» условиях.

ЗАКЛЮЧЕНИЕ

В выпускной квалификационной работе бакалавра мы воспроизвели результаты предыдущих исследований в области акустических клавиатурных атак без учителя и улучшили их современными алгоритмами машинного обучения, не доступными исследователям прошлых работ, такими как UMAP для понижения размерности вектора признаков, а так же представили новый нейросетевой подход для улучшения кластеризации контекстуальных данных.

Наши эксперименты говорят о том, что представленный в данной работе метод до сих пор не готов к применению в реальных условиях, однако если решить 2 основные проблемы описанные в секции 3.4, уже в скором времени данные акустические атаки могут стать очень опасными. Главной группой риска в данном случае будут являются онлайн стримеры, которые зачастую имеют хорошие студийные микрофоны и механические клавиатуры. Именно на такой тестовой установке мы и получили самые лучшие результаты восстановления текста. Кроме того в интернете уже доступны многочасовые записи стримов из которых можно легко достать десятки минут, а то и часов звуков печати на клавиатуре, которые могут содержать конфиденциальные данные, такие как номера и пин-коды банковских карт и пароли от различных интернет аккаунтов.

Представленная в данной работе нерестовая архитектура *Continuous Bag of Samples* (CBoS) показала свою эффективность в улучшении результатов кластеризации в рамках решаемой задачи. Интересно было бы попробовать CBoS в других задачах с контекстуальными данными где нужна качественная кластеризация или классификация, например распознавание голоса.

В приложении Б приведена ссылка на страницу GitHub с написанным нами программным комплексом, используемым для получения всех вышеописанных результатов.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Asonov D., Rakesh A. Keyboard Acoustic Emanations, 2004
2. Badshah A.M., Abdul and Ahmad, Jamil and Rahim, Nasir and Baik, Sung. Speech Emotion Recognition from Spectrograms with Deep Convolutional Neural Network, 2017
3. Bogert B.P., Healy M. J. R., and Tukey J. W., The Quefrency Alanysis of Time Series for Echoes: Cepstrum, Pseudo Autocovariance, Cross-Cepstrum and Saphe Cracking, Proceedings of the Symposium on Time Series Analysis (M. Rosenblatt, Ed) Chapter 15, 209-243. New York: Wiley, 1963.
4. Calinski T., Harabasz J., A dendrite method for cluster analysis, Communications in Statistics-theory and Methods 3 (1), 1974 1–27.
5. Halevi T., Saxena N. Keyboard acoustic side channel attacks: exploring realistic and security-sensitive scenarios, 2014
6. Hiu Yan F. Keyboard Acoustic Triangulation Attack BY Au, 2019
7. Lloyd S.P. Least square quantization in PCM's. Bell Telephone Laboratories Paper, 1957
8. Martinasek Z., Clupek V. Acoustic Attack on Keyboard Using Spectrogram and Neural Network, 2015
9. McInnes L., John Healy, James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction
10. Mikolov, T., Corrado, G., Chen K., Dean J. Efficient Estimation of Word Representations in Vector Space. Proceedings of the International Conference on Learning Representations (ICLR 2013), 1–12.
11. Rousseeuw P.J., Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. Journal of Computational and Applied Mathematics, 20:53 – 65, 1987.
12. Steinhaus H. Sur la division des corps materiels en parties. Bull. Acad. Polon. Sci., C1. III vol IV: 801—804., 1957
13. Zhuang L., Zhou F., and Tygar J. D., Keyboard acoustic emanations revisited, in Proceedings of the 12th ACM conference on Computer and communications security, 2005

14. van der Maaten L.J.P., Hinton G.E. Visualizing Data Using t-SNE // Journal of Machine Learning Research. — 2008. — November
15. Full-text corpus data – https://www.corpusdata.org/wiki/samples/text.zip
16. SCOWL (Spell Checker Oriented Word Lists) – http://wordlist.aspell.net
17. NLTK (Natural Language Toolkit) – https://www.nltk.org

ПРИЛОЖЕНИЕ А

Отрывок оригинального текста

tea has a stimulating effect in humans primarly due to its caffeine cintent. tea plants are native to east asia and probably originated in the borderland of southwestern china and northern burma. an early credible record of tea drinking dates to the third century ad, in a medical text written nby hua tuo. it was popularised as a recreational drink suring the chinese tang dynasty, and tea drinking subsequently spread to east asian countries.

Отрывок восстановленного из классов текста

В тексте могут быть добавлены или пропущены буквы из за ошибок выделения звуков клавиши из исходной аудиозаписи

ted has a stimulating effect in jumans primarly bue to its coffeine cintent. ted plants are catife to hast asid and procally orivinated in the corderland of southwestern chind and corthern burma. an herly credicke record of ted drinking bates to the third century ad, in a medical tect writhen byby jud tus. it was popllarised as a recreational brin, suring the chinese tang donasty, and ted brinking subshouently spread to hast asian countries.

Отрывок восстановленного классификатором текста

В тексте могут быть добавлен или пропущены буквы из за ошибок выделения звуков клавиши из исходной аудиозаписи

tea has a stimulating effect in humans primarls due to its caffeine cintent. tea plants are native to east asia and probably originated in the borderland of southwestern china and northern burma. an early credible record of tea dringing dates to the third century ad. in a medical text written nyby hua tuo, it was popularises as a recreational bring suring the chinese tang dynasty. and tea bringing subseouently spread to east asian countries.

ПРИЛОЖЕНИЕ Б

Ссылка на проект на GitHub:
https://github.com/AlexN1ght/Acoustic-Keylogger