Федеральное государственное автономное образовательное учреждение высшего образования
Национальный исследовательский университет
«Высшая школа экономики»

Факультет компьютерных наук

Баранова Дарья Дмитриевна

РАЗРАБОТКА СЕРВИСА ДЛЯ ГЕНЕРАЦИИ РЕАЛИСТИЧНЫХ ИЗОБРАЖЕНИЙ НА ОСНОВЕ ЭСКИЗА С ПОМОЩЬЮ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ

Выпускная квалификационная работа

студента образовательной программы магистратуры «Машинное обучение и высоконагруженные системы» по направлению подготовки *01.04.02 Прикладная математика и информатика*

Рецензент
ученая степень, ученое звание,
должность

Руководитель
ученая степень, ученое звание, должность

__Нарек Алвандян__
И.О. Фамилия

Москва, 2023 год

Аннотация

Работа содержит * глав, * страниц, * рисунков, * таблиц, * использованных источников, * приложений.

Оглавление

Введение

В настоящее время задача генерации изображений находится на острие развития науки. Множество задач сохраняют свою актуальность и требуют более тщательных исследований несмотря на недавно произошедшие прорывы в области и большое количество существующих нестандартных решений.

Данная работа посвящена задаче Sketch-to-Image – преобразованию заданного эскиза в изображение. Другими словами, необходимо сгенерировать реалистичное изображение на основе наброска (эскиза), сохраняя при этом определенную семантику и границы отдельных элементов. Эта задача играет важную роль в задачах синтеза и обработки изображений, таких как редактирование фотографий и раскрашивание. Синтез изображений на основе эскизов позволяет людям, не являющимся художниками, создавать реалистичные изображения без значительных художественных навыков или специальных знаний в области синтеза изображений. Разработанные решения могут использоваться в качестве мощных инструментов в индустрии художественного дизайна, или, к примеру, в качестве инструмента, помогающего в составлении реалистичного фоторобота подозреваемого.

Сложность задачи заключается в том, что элементы эскиза почти всегда будут сильно отличаться от границ соответствующих реалистичных зарисованных элементов. Поэтому необходимо контролировать генерацию, сохраняя баланс между реализмом и соответствием результата исходному эскизу. Для этого необходимо реализовать подход, который позволит регулировать при генерации уровень реализма и степень соответствия зарисовке. Дополнительной сложностью выступает необходимость генерировать детали, которые отсутствуют на эскизе, но необходимы для реалистичного изображения.

Разработанное решение должно быть прикладным, то есть являться комплексным законченным инструментом – продуктом, который можно использовать.

Таким образом, в качестве объекта исследования выступает готовое изображение, которое было получено в процессе генерации. А предметом исследования выступает процесс получения этого изображения, то есть процесс его генерации с помощью нейросетевой модели на основе заданного наброска.

Целью работы является создание системы генерации изображений на основе эскизов (или зарисовок), сохраняя при этом заданную семантику и задавая определенный уровень реализма.

Поставленная цель предполагает решение следующих задач:

– выполнить обзор существующих методов генерации изображений на основе эскизов;

– получить размеченные обучающие данные;

– разработать метод генерации изображения;

– экспериментально определить оптимальную архитектуру сети и оптимальную реализацию метода генерации;

– реализовать систему генерации в виде полноценного сервиса;

– выполнить тестирование системы.

Глава 1 Анализ существующих подходов

1.1 Существующие подходы генерации изображений по эскизу

Если рассматривать задачу в более общем виде, то можно её свести к Image-to-Image задаче, которая является ключевой во всей области компьютерного зрения. Большинство подходов к решению этой задачи основывается на Generative Adversarial Network (GAN), Variational Autoencoder (VAE) архитектурах, а также Diffusion моделях, совершивших крупный прорыв в области за последнее время. Задача Sketch-to-Image не является исключением, поэтому её существующие решения также основываются на указанных моделях с некоторыми модификациями в архитектуре. Рассмотри подробнее некоторые из них.

1.1.1 GANs

1.1.1.1 Conditional Adversarial Networks (pix2pix)

В этой статье моделью генерации выступала GAN с conditional дополнением. В качестве входных данных модель принимала не только случайные данные, как это обычно происходит у GAN моделей, но и дополнительное условие (condition), с помощью чего обусловливалось входное изображение. Входным изображением не обязательно выступал набросок, так как задача в исследовании ставилась иначе – создать «переводчик» изображений из одной области в другую. В качестве области может выступать RGB-изображение, карта границ, карта семантических меток и т. д. [https://arxiv.org/abs/1611.07004]

Такой подход может частично решать поставленную в данной работе задачу. Для этого исходной областью будет выступать карта границ объекта, а целевой областью – RBG-изображение (см. рисунок 1).

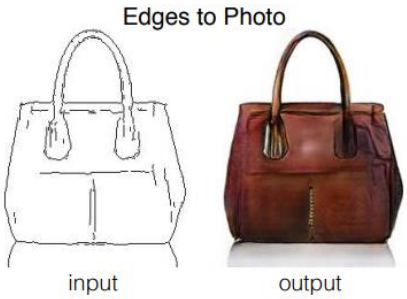

Рис. 1 Результат работы pix2pix модели при переводе изображения из карты границ в RGB-изображение

Недостатком данного решения является сильное ограничение на соответствие границ объекта на сгенерированном изображении граничным линиям эскиза. Такое ограничение сильно портит качество генерации в задаче генерации по эскизу, а не по карте границ, так как границы эскиза зачастую условны и редко совпадают с очертаниями соответствующего реалистичного объекта.

1.1.1.2 Contextual GAN

Указанный недостаток был также выделен и исправлен в рамках другой более новой работы [https://arxiv.org/pdf/1711.08972.pdf]. По словам авторов, её главное преимущество заключается именно в отсутствии упомянутого строгого ограничения на пиксельное соответствие для границ.

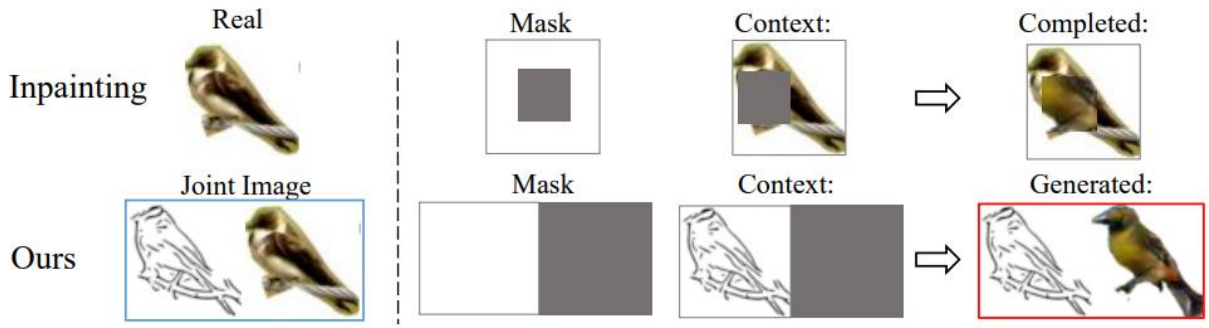

Рис. 2 Иллюстрация подхода Contextual GAN. Эскиз используется в качестве контекста

Новый подход заключается в том, что теперь генератор будет пытаться восстанавливать изображение, как схожим образом это происходит в задаче Inpainting. В традиционной Inpainting задаче поврежденная часть входного изображения дополняется с использованием окружающего содержимого изображения в качестве контекста. Для иллюстрации приведен рисунок 2. При дорисовке поврежденной части изображения птицы в верхнем ряду немаскированная часть птицы являются контекстуальной информацией. По аналогии, теперь «испорченной» частью будет выступать все изображение, которое необходимо сгенерировать, а «контекст» будет предоставляется входным эскизом (нижний ряд рисунка 2).

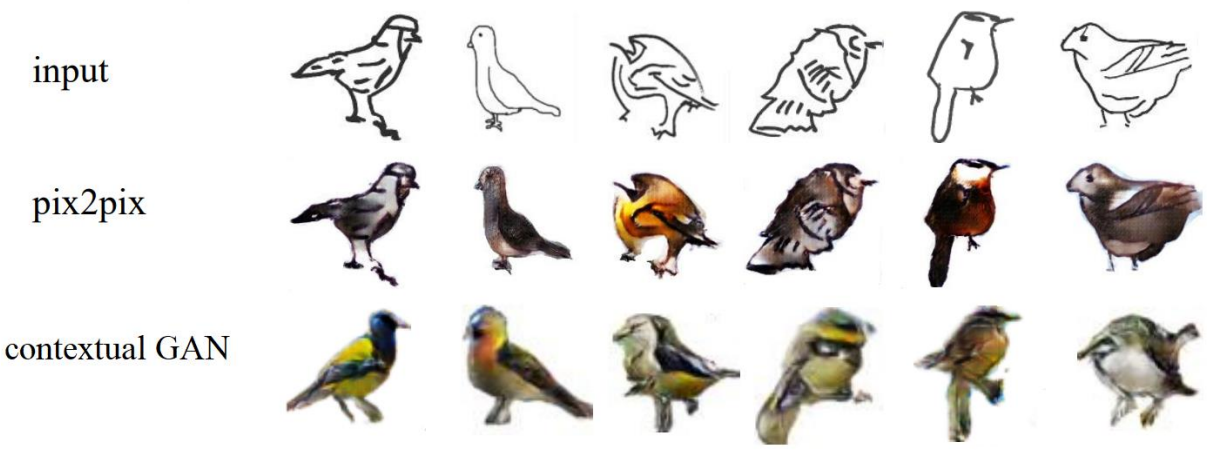

Рис. 3 Сравнение результатов работы Contextual GAN и Pix2Pix на примере генерации птиц

1.1.2 Diffusions

1.1.2.1 DiSS [https://arxiv.org/pdf/2208.12675.pdf]

В работе используется стандартная архитектура диффузионной модели. Модификации состоят в следующем:

1. У входа модели увеличено число размерностей. Это сделано для того, чтобы дополнительно подавать на вход изображение эскиза и изображение цветовой карты.

2. В процессе обучения модели в этих входных вспомогательных данных заменяется часть изображения на серые пиксели. С помощью этой модификации контролируется степень соответствия генерируемого изображения каждому из условий (условию близости к эскизу и условию близости к карте цветов).

3. Степень реализма достигается путем последовательного применения операций даунсэмплинга (downsampling) и апсэмплинга (upsampling) на каждом итеративном шаге модели. Таким образом сохраняется заданное соотношение эскиз-цвет и регулировка реализма происходит независимо.

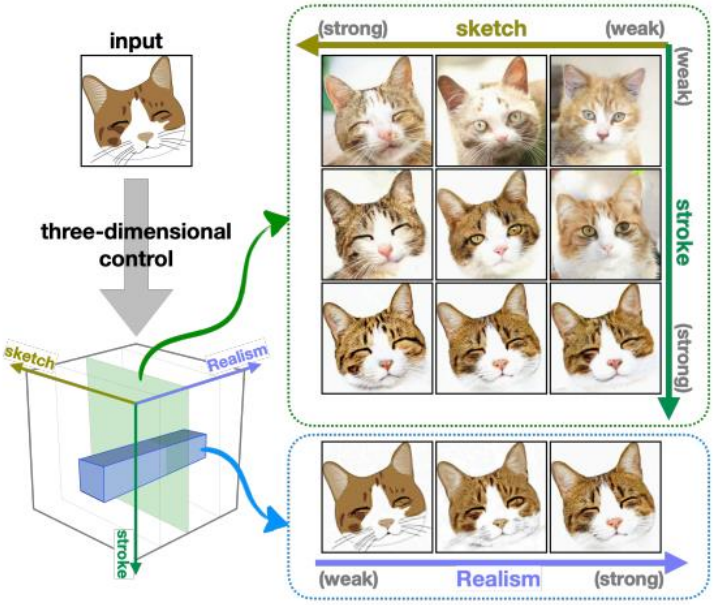

Рис.4 Пример полученных результатов работы модели DiSS при разных соотношениях соответствия эскиз-цвет-реализм

1.1.2.2 MLP Latent Guidance Predictor [https://arxiv.org/pdf/2211.13752v1.pdf]

Метод позволяет сгенерировать изображение на основе эскиза и текстового описания. В результате могут получиться разные результаты для одного эскиза – разница будет формироваться на основе различного текстового описания.

Рис.5 Примеры различной генерации при одном эскизе и разном текстовом описании

Основная идея подхода в том, что к готовой диффузионной модели вида text-to-image добавляется MLP сеть, которая будет выступать помощником в процессе генерации изображения.

Добавленный MLP обучается предсказывать истинный эскиз по переданным ему активационным слоям диффузионной модели. После этого во время обратного диффузионного процесса на каждом шаге t вычисления моделью шума для какого-то изображения x_t из обучающего батча, выходы её активационных слоев будут

снова подаваться на вход MLP. Перцептрон в свою очередь должен генерировать на основе полученных активаций соответствующий эскиз E˜ (точнее, эмбединг эскиза). Исходя из этого эскиза E˜ и истинного эскиза e при вычислении расшумленного изображения x_{t-1} будет также учитываться антиградиент $-\nabla_{x_t} L$, где $L(E˜, E(e)) = ||E˜ - E(e)||^2$. Таким образом, диффузионная модель будет стараться соблюдать указанные границы эскиза при генерации. Исходной моделью до модификации выступает предобученная latent diffusion model (Stable Diffusion).

1.2 Вывод

В этой главе было проведено исследование нескольких существующих подходов для решения задачи генерации реалистичного изображения на основе скетча. На основе сравнения результатов можно сделать ряд выводов. Во-первых, можно заметить, что качество изображений, полученных диффузионными моделями, сильно превышает качество изображений, сгенерированных моделями GAN архитектуры. Поэтому будущее решение будет основываться на архитектуре диффузионной модели. Во-вторых, решение 1.2.2 представляется более предпочтительным по сравнению с решением 1.2.1, так как кажется более простым и интересным подходом. При этом, это самое новое исследование, у него до сих пор не опубликован исходный код и недоступны веса обученной модели. В работе описаны хорошие полученные результаты, а также решение удовлетворяет сформулированным требованиям, а именно – существует механизм контроля степени соответствия эскиз-реализм. Обучение модели занимает достаточно мало времени, так как используется предобученная модель диффузии, и необходимо обучить только вспомогательную MLP сеть.

Таким образом, решение предлагается строить на основе архитектуры, описанной в исследовании [https://arxiv.org/pdf/2211.13752v1.pdf].

Глава 2 Разработка методов и проектирование системы

Опишем подробнее предлагаемый метод для генерации изображения по переданному скетчу, а также архитектуру будущей системы.

2.1 Описание модели и метода

2.1.1 Diffusion model

Основная идея диффузионной модели состоит в том, что с помощью итеративного прямого диффузионного процесса поданное на вход изображение «разрушается», смешиваясь с шумом до тех пор, пока оно целиком не станет состоять из белого шума. После этого происходит обратный диффузионный процесс – процесс восстановления изображения или его «расшумление», который также происходит итеративно. [https://arxiv.org/abs/1503.03585] Таким образом, имея правильно обученную сеть, мы можем подавать ей на вход изображение, состоящее полностью из белого шума, и получать на выходе сгенерированное изображение. Рассмотрим подробнее прямой и обратный диффузионные процессы.

Прямой процесс заключается в том, что исходное изображение x_0 постепенно зашумляется, проходя определенное количество итераций t={0,..,T}, пока на последней итерации T изображение x_T не начнет содержать только белый шум, то есть пока не будет полностью состоять из данных нормального распределения N(0, I). Таким образом, обозначим как $q(x_t|x_{t-1})$ функцию зашумления изображения x_{t-1} до состояния x_t.

При обратном процессе изображение, наоборот, расшумляется, проходя итерации от x_T до x_0. На шаге i модель получает изображение x_i на вход, и возвращает его в менее зашумленном состоянии x_{i-1}, таким образом постепенно генерируя его. Итеративность процесса генерации позволяет добиться лучшего качества, а также отслеживать и контролировать генерацию на каждом шаге. Обозначим $p(x_{t-1}|x_t)$ как функцию расшумления изображения, то есть перевода из состояния x_t в состояние x_{t-1}.

Согласно введенным обозначениям, будем иметь:

$$q(x_t|x_{t-1}) = N(x_t, \sqrt{1-\beta_t}x_{t-1}, \beta_t I),$$

где:

$N(\cdot)$ – нормальное распределение,

$\sqrt{1-\beta_t}x_{t-1}$ – матожидание,

β_tI – дисперсия,

β_t – коэффициент, отвечающий за постепенное изменение шума в изображении, $\beta_t \in [0,1]$.

Также можно свернуть все итеративные переходы, соединив все преобразования в одну итерацию и избавившись от промежуточных вычислений:

$$q(x_t|x_0) = N(x_t, \sqrt{\bar{a}_t}x_0, (1-\bar{a}_t)I),$$

где:

$\alpha_t = 1 - \beta_t$,

$\bar{a}_t = \prod_{s=1}^{t} a_s$.

Теперь подробнее опишем обратный процесс.

$$p(x_{t-1}|x_t) = N(x_{t-1}, \mu_\theta(x_t,t), \Sigma_\theta(x_t,t)),$$

$$\mu_\theta(x_t,t) = \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{\beta_t}{\sqrt{1-\bar{a}_t}}\epsilon(x_t,t)),$$

где:

$\epsilon(x_t,t)$ – шум, $\epsilon \sim N(0,I)$

$\Sigma_\theta(x_t,t)$ считается зафиксированной величиной и может быть установлена как β_t (в дальнейших работах этот параметр будет также учиться моделью).

Для реализации описанного процесса необходимо создать сеть, которая будет предсказывать шум, который присутствует на изображении [https://arxiv.org/abs/2006.11239]. После этого на каждой итерации необходимо вычитать предсказанный шум из входного изображения, делая его таким образом менее зашумленным. Функция потерь L_t будет являться стандартной среднеквадратичной ошибкой:

$$L_t = \|\epsilon - \epsilon(x_t,t)\|^2$$

Можно выразить x_{t-1} через x_t:

$$x_{t-1} = \begin{cases} \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{a}_t}}\epsilon(x_t,t)\right) + \sqrt{\beta_t}\epsilon, & t > 1 \\ \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{\beta_t}{\sqrt{1-\bar{a}_t}}\epsilon(x_t,t)\right), & t = 1 \end{cases}$$

Algorithm 1 Training

1: **repeat**
2: $\mathbf{x}_0 \sim q(\mathbf{x}_0)$
3: $t \sim \text{Uniform}(\{1, \dots, T\})$
4: $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
5: Take gradient descent step on
$$\nabla_\theta \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_\theta(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1-\bar{\alpha}_t}\boldsymbol{\epsilon}, t) \right\|^2$$
6: **until** converged

Algorithm 2 Sampling

1: $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$
2: **for** $t = T, \dots, 1$ **do**
3: $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ if $t > 1$, else $\mathbf{z} = \mathbf{0}$
4: $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left(\mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}} \boldsymbol{\epsilon}_\theta(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$
5: **end for**
6: **return** $\mathbf{x}_0$

Рис.6 Алгоритмы для процедуры обучения и выборки для диффузионной модели

Нужно заметить, что количество шума на каждой итерации может и будет отличаться, так как это позволяет достичь лучшего качества. При линейном характере изменения доли шума на изображении наблюдается несколько проблем, а именно:

1. На последних итерациях процесса, когда изображение приближается к тому, чтобы стать белым шумом и утратить последние элементы содержательной информации, визуально изменения уже не кажутся значительными (см рисунок). Можно сказать, что последняя треть или даже половина итераций уже выглядит как полностью зашумленное изображение.

2. При этом, на первых итерациях зашумления ключевая содержательная информация слишком быстро утрачивается.

Рис.7 Зашумление при линейном (вверху) и косинусном (внизу) подходе соответственно на каждой итерации t от 0 до T. В последней четверти линейного графика изображения представляют собой почти чистый шум, тогда как косинусный график добавляет шум медленнее.

Исходя из этого, в качестве альтернативы был предложен [https://arxiv.org/abs/2102.09672] косинусный характер изменения доли шума на изображении.

Архитектура сети представляет собой U-Net. На вход сети также будет подаваться номер итерации, представленный особым образом. Это важная деталь, так как, как было указано выше, в зависимости от номера итерации на изображении будет присутствовать разная доля шума.

2.1.2 Stable diffusion model

Stable Diffusion model является усовершенствованием классической diffusion модели. Она является latent diffusion model (LDM), что означает, что модель работает не с изображениями напрямую, а с их закодированными представлениями в латентном пространстве.

Stable Diffusion состоит из 3 частей: вариационного автоэнкодера (VAE), U-Net и дополнительного текстового энкодера, так как генерирует изображение на основе не только изображения, но и текстового описания.

Кодировщик VAE сжимает изображение из пространства пикселей в скрытое пространство меньшего размера, сохраняя основную семантику изображения и отбрасывая наименее фундаментальную информацию. Это позволяет увеличить скорость работы сети, так как размер данных уменьшился. Декодировщик VAE генерирует выходное изображение, преобразуя представление обратно в пространство пикселей.

Для эмбединга текста используется предварительно обученная сеть CLIP [https://arxiv.org/abs/2103.00020].

2.1.3 MLP

В качестве вспомогательной подсети в архитектуру будет также включен MLP. Его задача заключается в учете границ эскиза, который предварительно переведен в латентное пространство.

На вход перцептрону подаются активационные слои Stable Diffusion модели. На выходе требуется, чтобы перцептрон выдавал представление максимально похожее на представление эскиза. Формально, для входного тензора изображения w и контекста c на итерации t обозначим соединенные (сконкатенированные) активации выбранных внутренних слоев сети $\{l_1, \ldots, l_n\}$ как:

$$F(w|c,t) = [l_1(w|c,t), \ldots, l_n(w|c,t)]$$

Тогда входное измерение MLP представляет собой сумму количества каналов выбранных слоев. На выходе ожидается, что перцептрон вернет изображение

аналогичного размера, которое получается на выходе диффузионной сети после очередной итерации.

Также перцептрон должен уметь работать с входными данными с любой итерации диффузионной модели, поэтому дополнительно ему на вход подается номер итерации t.

После того, как перцептрон будет обучен, сеть будет считаться обученной полностью и готовой для генерации. Для того, чтобы на итерации t, получив входное изображение x_{t-1}, сгенерировать изображение x_t, необходимо, чтобы диффузионная сеть предсказала шум $\epsilon(x_t, t)$, который присутствует на изображении. Во время этого предсказания у модели должны считаться и сохраняться активации $F(w|c, t)$ выбранных внутренних слоев, которые используются MLP. На их основе перцептрон генерирует соответствующие латентное представление эскиза E˜, вычисляется мера похожести полученного эскиза E˜ на истинный e: L(E˜, E(e)) = ||E˜ − E(e)||^2, а затем считается соответствующий антиградиент $-\nabla_{x_t}\mathrm{L}$. Тогда новое изображение на выходе будет рассчитываться как:

$$\widetilde{x_{t-1}} - 1 \;=\; x_{t-1} - \;\alpha * \nabla_{x_t}\mathrm{L}$$

Где:

α – коэффициент, который задает степень похожести итогового изображения на эскиз. Эмпирически было выяснено, что лучше себя показывает нормированный коэффициент по формуле:

$$\alpha = \frac{\|x_t - x_{t-1}\|_2}{\|\nabla_{x_t}\mathrm{L}\|_2}\,\beta$$

В этом случае коэффициентом выступает уже β, который будет иметь порядок $O(1)$.

2.2 Обучающая выборка и входные данные

Во всех экспериментах использовался датасет Imagenet [https://image-net.org/static_files/papers/imagenet_cvpr09.pdf] с именами классов в качестве текстового описания. Соответствующие изображения эскизов были сгенерированы с помощью модели предсказания краев из [https://arxiv.org/abs/2108.07009], а затем дискретизированы с порогом 0,5.

2.3 Проектирование системы в рамках создания телеграм бота