МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Институт математики, физики и информационных технологий
(наименование института полностью)

Кафедра «Прикладная математика и информатика»
(наименование)

01.03.02 Прикладная математика и информатика
(код и наименование направления подготовки, специальности)

Системное программирование и компьютерные технологии
(направленность (профиль))

# ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему: «Увеличение разрешения изображений с использованием нейронных сетей»

Студент А.С. Негру
(И.О. Фамилия) (личная подпись)

Руководитель д.т.н,, доцент, С.В. Мкртычев
(ученая степень, звание, И.О. Фамилия)

Консультант К.А. Селиверстова
(ученая степень, звание, И.О. Фамилия)

Тольятти 2020

## Аннотация

Тема бакалаврской работы: «Увеличение разрешения изображений с использованием нейронных сетей».

В данной бакалаврской работе исследуются способы увеличения разрешения изображений на основе нейросетевых технологий.

Цель работы - разработка приложения, для повышения разрешения изображения с использованием глубоких нейронных сетей.

За основу взята генеративно-состязательная сеть SRGAN, позволяющая решать задачу увеличения разрешения изображений. В работе предложены изменения в структуре генератора и дискриминатора, повышающие скорость их работы. На языке Python разработано приложение, которое демонтирует возможности предложенного варианта нейронной сети при решении задачи увеличения разрешения изображений. Работа приложения протестирована на различных изображениях.

Структура бакалаврской работы представлена введением, тремя главами, заключением, списком литературы.

Во введении описывается актуальность проводимого исследования, дается краткая характеристика проделанной работы.

В первой главе проводится анализ перспектив развития методов по увеличению разрешения изображений.

Во второй главе описывается стандартные методы увеличения изображения, основанные на интерполяции. Здесь также рассматривается структура нейронной сети, позволяющей генерировать изображения высокого разрешения, описывается математическая модель сети.

В третьей главе дается описание разработанного приложения для увеличения размера изображений на основе.

В заключении представлены выводы по проделанной работе.

В работе использовано 10 рисунков, 14 формул, список литературы содержит 20 литературных источников. Объем бакалаврской работы составляет 40 страниц.

## Abstract

The subject of bachelor 's work: "Increasing the resolution of images using neural networks."

This bachelor 's paper explores ways to increase image resolution based on neural network technologies.

The goal of the work is to develop an application, to increase image resolution using convolutional neural networks.

Based on the generative-competitive network SRGAN, which allows to solve the problem of increasing the resolution of images. The paper proposes changes in the structure of the generator and discriminator, increasing the speed of their work. An application has been developed in Python that disassembles the capabilities of the proposed neural network variant when solving the problem of increasing image resolution. The application has been tested on various images.

The structure of bachelor's work is represented by introduction, three chapters, conclusion, list of literature.

The introduction describes the relevance of the study and provides a brief description of the work done.

The first chapter analyses the prospects for the development of methods to increase the resolution of images.

The second chapter describes standard image zoom methods based on interpolation. Also discussed here is the structure of a neural network capable of generating high-resolution images, and a mathematical model of the network is described.

The third chapter describes an application developed to increase the size of images based on.

The conclusion presents the conclusions on the work done.

The work used 10 drawings, 14 formulas, the bibliography contains 20 literature. The volume of bachelor's work is 42 pages.

**Содержание**

## Введение

Расширение сфер применения систем компьютерного зрения связано с потребностью автоматизации технических и социальных процессов. Благодаря их использованию решаются такие задачи как анализ медицинских изображений, анализ дорожно-транспортной обстановки. Алгоритмы анализа изображений, лежащие в основе работы таких систем, чувствительны к качеству входных изображений. Поэтому актуальным исследовательской задачей остается разработка и совершенствование алгоритмов повышения разрешения изображений.

В данной бакалаврской работе исследуются способы увеличения разрешения изображений на основе нейросетевых технологий.

В исследовании предложены модификации генеративно-состязательные сети SRGAN, заключающиеся в удалении слоев батч-нормализация в генераторе для повышения его производительности и снижении вычислительной сложности расчётов. Также предложены изменения по работе дискриминатора в части классификации изображений.

Цель работы - разработка приложения, для повышения разрешения изображения с использованием глубоких нейронных сетей.

На языке Python разработано приложение, которое демонтирует возможности предложенного варианта нейронной сети при решении задачи увеличения разрешения изображений. Работа приложения протестирована на различных изображениях.

## 1 Анализ проблемы повышения разрешения

Размер растрового изображения можно выразить в виде количества пикселей. При этом существуют такая величина, как разрешение изображения, которая выражается путем деления количества пикселей на площадь. При реализации систем компьютерного зрения в качестве входных данных часто отдается предпочтение изображениям с высоким разрешением. Это объясняется тем, что чем выше разрешение, тем больше мелких деталей можно различить на изображении.

Обработка изображений несколько последних десятилетий является активно развивающейся научной областью. Одной из активно решаемых задач в данной области является – повышение разрешения изображения. Это объясняется тем, что изображения с высоким разрешением имеют лучшие характеристики для восприятия человеком и позволяют проводить более качественный последующий анализ. Изначально считалось, что главным способом получения изображений с высоким разрешением является использование технически совершенного оборудования существующего на данный момент времени. Однако, при таком подходе возникают сложности, связанные со стоимостью оборудования и его техническими ограничениями. Для их преодоления был предложен подход, который состоит в получении изображения высокого разрешения из одного или нескольких изображений с низким разрешением.

Благодаря расширению сфер применения в последние несколько лет систем компьютерного зрения задача повышения разрешения изображений на основе вычислений все более актуальна. Для решения данной задачи перспективным является использование методов, основанных на глубоком машинном обучении, которые в последние годы доказали практическую значимость при решении различных научных задач.

Использование глубоких нейронных сетей при решении практических задач связано со следующими преимуществами:

– Простота в использовании. Нейронные сети могут учиться на примерах. Пользователь лишь подбирает данные для сети, после чего запускает обучение, а алгоритм обучение производит настройку параметров нейронной сети. При использовании нейронных сетей от аналитика требуется намного меньше действий, чем при использовании традиционных методов математической статистики.

– Широкий выбор функциональных возможностей. Глубокие нейронные сети - сильный метод моделирования, который позволяет осуществлять поиск и моделировать сложные зависимости данных. Работа нейронных сетей основана на нелинейном преобразовании данных. Стоит отметить, что линейное моделирование в течение длительного времени являлось основным методом моделирования во многих областях, т.к. для него разработано большинство методов оптимизации. Однако, большинство задач реального мира имеют неудовлетворительную линейную аппроксимацию, по этой причине растет популярность нейронных сетей как нелинейных моделей.

На основе вышесказанного можно сделать вывод, что с развитием систем компьютерного зрения растет потребность в получении изображений высокого разрешения. При этом функциональные возможности глубоких нейронных сетей в теории позволяют обрабатывать (дополнять) изображения, повышая их разрешение. В рамках данной бакалаврской работы планируется на практике реализовать приложение, реализующее глубокую нейронную сеть, позволяющую повышать разрешение изображений.

Целью данной работы является разработка приложения, для повышения разрешения изображения с использованием глубоких нейронных сетей.

Поставленную цель планируется достигнуть путем решения следующих задач:

1. Провести анализ проблем повышения разрешения изображений.

2. Разработать метод применения нейронных сетей для увеличения разрешения изображения.

3. Спроектировать и разработать приложение для увеличения разрешения изображений.

**Выводы по главе.**

С увеличение распространения систем компьютерного зрения растет потребность в получении изображений высокого разрешения. При этом функциональные возможности глубоких нейронных сетей в теории позволяют обрабатывать (дополнять) изображения, повышая их разрешение. На основе этих данных была сформулирована цель работы и определены задачи, требующие решения для ее достижения.

# 2 Применение нейронных сетей для увеличения разрешения изображений

## 2.1 Классические методы увеличения изображения

Рассмотрим классические, наиболее распространённые, методы увеличения разрешения изображения.

Основными методами, которые часто используют при решении задачи увеличения разрешения изображений, является методы, основанные на интерполяции.

Интерполяция – является подходом к нахождению промежуточных значений (в нашем случае цветов пикселей) по набору исходных известных данных. Фактически увеличение разрешения изображения на основе интерполяции связано с расчетом цвета новых появляющихся пикселей. Данные инструменты используются во всех редакторах растровых изображений, таких например, как Adobe Photoshop компании Adobe System Incorporated.

При увеличении разрешения изображений с использованием интерполяции могут возникнуть следующие артефакты:

- Волновые артефакты в местах изображения с резким изменением яркости. Такой дефект называется Ringing и появляется чаще всего на контрастных изображениях.
- Артефакты изображения, заключающиеся в появлении ярко выраженной ступенчатости диагональных границ на изображении. Такой артефакт носит название Aliasing.
- Наиболее часто встречаемый дефект при интерполяции – снижение четкости изображения. Данный дефект также носит название Unsharpening.

- В зависимости от типа применяемой интерполяции на изображении могут появляться субпиксельные сдвиги (Subpixel shift). Визуально такие артефакты обычно слабо заметены, но они оказывают влияние на формальные метрики оценки качества.

Применяемые алгоритмы интерполяции данных можно разделить на две категории:

- неадаптивные алгоритмы, обрабатывающие пиксели все пиксели на изображении идентично;
- адаптивные алгоритмы, которые подстраиваются под задачу интерполяции.

Адаптивные нелинейные методы получили малое распространение из-за узкой направленности их применения. Но в лицензированных программах можно увидеть применение коммерческих алгоритмов на основе адаптивных методов.

Чаще всего адаптивные алгоритмы позволяют избежать таких искажений, как overshooting или aliasing и позволяют сохранить резкость границ. Но часто могут создавать несвойственные исходному изображению одиночные пиксели или текстуры.

Примеры неадаптивные линейных алгоритмов:

- метод ближайшего соседа, заключающийся в определении цвета пикселей на основе цветов соседних пикселей, отвечающих критерию близости, рассчитываемому по одной из метрик расстояния между объектами;
- билинейная интерполяция – расширение линейной интерполяции для функций двух переменных. В этом случае линейная интерполяция сначала в одном направлении, потом – перпендикулярно;
- бикубическая интерполяция – расширение кубической интерполяции для функций двух переменных, значения которой заданы на двумерной регулярной сетке. Поверхность, полученная в результате бикубической

интерполяции, является гладкой функцией на границах соседних квадратов, в отличие от поверхностей, полученных в результате билинейной интерполяции или интерполяции методом ближайшего соседа;

- сплайн-интерполяция. Сплайном является функция, область определения которой разбита на конечное число отрезков, на каждом из которых она совпадает с некоторым алгебраическим многочленом (полиномом).

В спектральном представлении используется разложение изображения в некоторый новый базис (например, двумерное дискретное преобразование Фурье, или дискретное косинусное преобразование), и обратное к нему. Этот метод удобен тем, что мелкие детали соответствуют высокочастотным компонентам базиса, и за счет этого становится возможным восстанавливать мелкие детали изображения, которые возможно получить за счет интерполяции с нескольких изображений низкого разрешения.

## 2.2 Генерирование изображений с высоким разрешением с использованием глубоких нейронных сетей

Искусственные нейронные сети принадлежат сфере машинного обучения.

Машинное обучение – это раздел науки об искусственном интеллекте, который изучает алгоритмы способные обучаться на заданном наборе данных.

Алгоритмы машинного обучения принято делить на 2 категории:

- алгоритмы обучения без учителя – принадлежат классу задач обработки данных, в которых известны только значения атрибутов исходного множества объектов. Алгоритмы из данной категории, в ходе своей работы, сравнивают заданные объекты с целью поиска в них закономерностей и зависимостей;

• алгоритмы обучения с учителем – для таких алгоритмов подразумевается наличие пар «входные данные – правильный ответ». Алгоритм восстанавливает связь между входными данными и ожидаемыми результатами. На основе изученной информации алгоритм обучается создавать результат максимально схожий с правильным результатом тестовой выборки.

Искусственные нейронные сети – это системы, преобразующие информацию подобно процессам, происходящим в человеческом мозге. Обрабатываемая информация имеет численный характер, позволяя использовать нейронные сети в качестве модели объекта с неизвестными характеристиками.

Одной из главных функциональных особенностей нейронных сетей является способность к обучению. Обучение нейронных сетей заключается в итерационном процессе подбора весовых коэффициентов, обеспечивающих ее требуемую реакцию сети на входные сигналы.

Обученная нейронная сеть моделирует зависимости между входными и выходными данными, содержащиеся в обучающей выборке.

Настроенная и обученная нейронная сеть может возвращать правильный результат на данных, отсутствовавших в обучающей выборке. Некоторые типы нейронных сетей могут работать с неполных или искаженных данными.

Структурной единицей нейронной сети является нейрон представляющий из себя аналог биологического нейрона. С точки зрения математики искусственный нейрон это нелинейная функция, обрабатывающая все входные сигналы и посылающая результат на единственный выход нейрона. Эта нелинейная функция называется активационной функцией или функцией активации.

Искусственный нейрон включает в себя группу синапсов, которые представляют собой однонаправленные связи, соединенные с предыдущими нейронами.

Выходная связь нейрона с синапсами следующих нейронов называется аксон.

Каждый синапс можно охарактеризовать с помощью весов W. Это аналог электрической проводимости биологического нейрона.

Для того чтобы параллельно обрабатывать сигналы нейроны объединяются в слои в зависимости от их функционирования. Слои принято нумеровать слева направо. Первый слой называется входным.

Нейроны этого слоя принимают входные данные и передают их следующему слою. Последний слой называется выходным, т.к. в качестве выходных данный он выдает результат работы нейронной сети. Слои, находящиеся между входным и выходным слоем, называются скрытыми или внутренними. В этих слоях происходит большая часть обработки данных.

Когда выходные сигналы нейронов одного слоя передаются всем нейронам следующего слоя, слой, получающий эти сигналы, называется полносвязным.

При обучении для определения корректности результата работы применяется функция потерь. В задаче обучения с учителем функция потерь показывает разницу между реальными данными и данными полученными в результате работы нейронной сети.

Если у нейронной сети большое количество слоев ее определяют в категорию глубокого обучения(DL).

Одной из основных характеристик сети является ее модель. Охарактеризовать эту модель можно по видам нейронов, структуре модели и способам обучения.

При решении задачи масштабирования изображений используются разные модели, однако лучше всего себя зарекомендовали: сверточные нейронные сети и генеративно-состязательные сети.

Сверточные нейронные сети успешно применялись в задаче обработки изображений еще в прошлом веке [1]. Но только недавно они обрели большую популярность, началом которой послужила демонстрация значительного превосходства над другими методами в задаче классификации изображений [2]. Большую роль в этом прогрессе сыграла эффективная реализация обучения на современных мощных графических процессорах и легкий доступ к данным для обучения моделей с огромным количеством параметров.

В основе сверточных нейронных сетей лежат следующие идеи:

- каждый нейрон слоя на вход получает сигнал от локального рецептивного поля в предыдущем слое. Рецептивное поле это слой нейронов, воспринимающих внешние сигналы и передающими их следующему слою. Рецептивное поле непосредственно взаимодействует со следующим слоем;
- все скрытые слои нейронной сети состоят из множества карт признаков, где все нейроны имеют общие веса;
- за каждым слоем свёртки идет вычислительный слой, осуществляющий локальное усреднение карт и пространственную подвыборку.

При вычислении сети нейроны выполняют операцию свертки области предыдущего слоя, которая состоит из множества нейронов связанных нейронами слоя. Такие слои нейронной сети называются сверточными слоями.

Также в сверточной нейронной сети могут присутствовать слои субдискретизации и полносвязные слои. Слои субдискретизации уменьшают размерности пространства карт признаков. Этот слой позволяет ускорить дальнейшие вычисления. Субдискретизации могут быть применены благодаря тому, что в такой архитектуре нейронных сетей не так важно значение признака, как сам факт его наличия.

Выходной слой сверточной нейронной сети всегда полносвязный. Нейроны выходного слоя имеют полное соединение со всеми функциями активации прошлого слоя.

Все виды слоев могут чередоваться в произвольном порядке. Это позволяет создавать карты признаков из карт признаков, т.е. распознавать сложные иерархические признаки. А значит позволяет от конкретных особенностей входных данных к абстрактным деталям.

При обучении нейронных сетей обычно используется метод обратного распространения ошибки или его модификации. Суть этого метода заключается в том, что распространение сигналов ошибки происходит от выходов сети к ее входам, в направлении обратном прямому распространению сигналов.

В задаче увеличения разрешения изображений методы на основе сверточной нейронной сети вскоре продемонстрировали превосходные результаты [3], где предложен метод (SRCNN), состоящий из 4 этапов (рисунок 1):

Первый этап – увеличение размера изображения. Прежде всего, нужно увеличить размер исходного изображения до нужных размеров. Как говорилось ранее, это чаще всего делают с помощью бикубической интерполяции;

Второй этап – выделение патчей и их представление. На этом этапе проводится получение отдельных участков из изображения и представление их в виде многомерных векторов содержащих карты признаков размером равным размерности вектора по формуле (1):

$$F_1(y) = \max(0, W_1 * y + B_1) \quad (1)$$

где $W_1$ – фильтр, $B_1 \in \mathrm{R}^{n1}$ – вектор сдвига, а символом «$*$» обозначена операция свертки. При этом:

$$W_1 = c \times f_1 \times f_1 \quad (2)$$

где $c$ – число каналов на изображении, $f_1$ – размер фильтра, $n_1$ – число операций свертки, а взятие поэлементного максимума осуществляется с использованием функция активации ReLU сверточной нейронной сети [4]. Веса $B_1$ представляют собой $n_1$-мерный вектор, каждый элемент которого сопоставлен с элементом фильтра $W_1$.

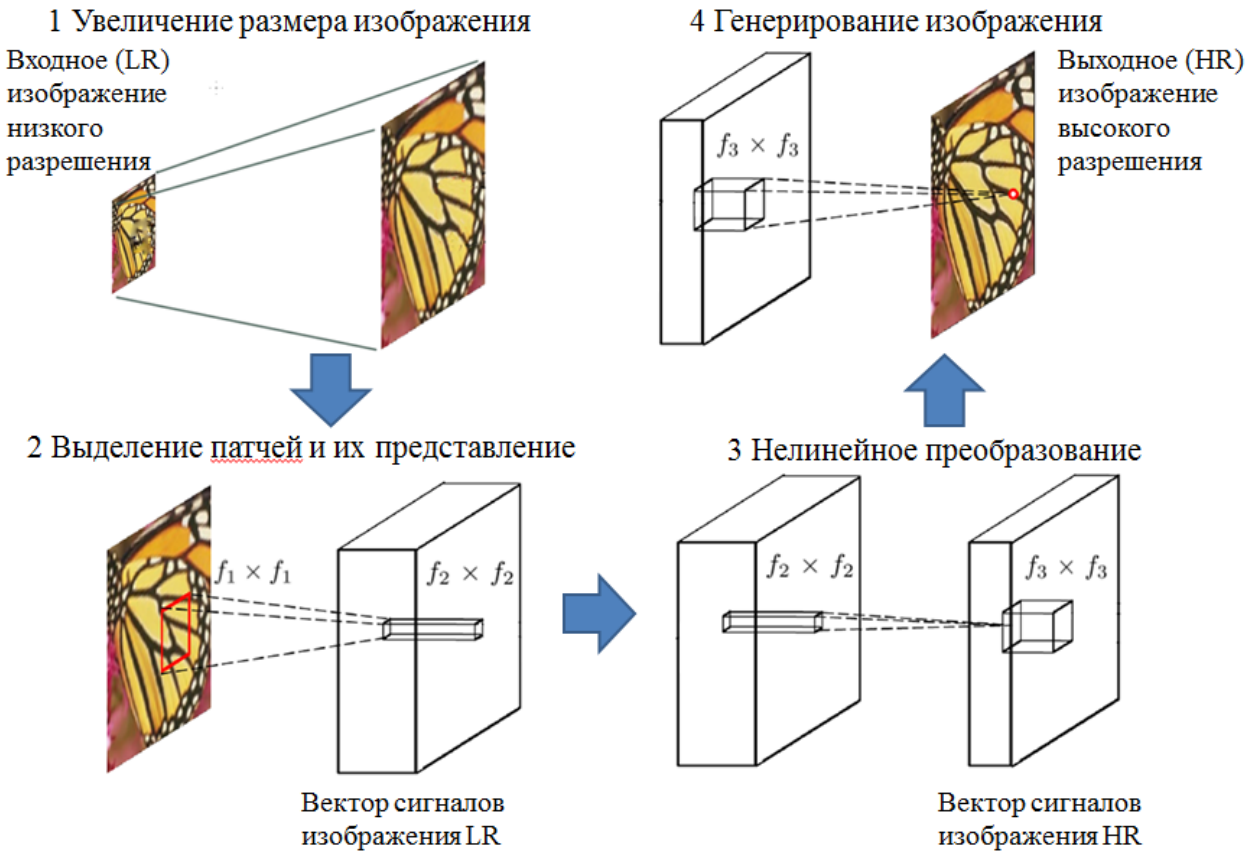


Рисунок 1 – Схема сверточной нейронной сети для решения задачи увеличения разрешения

Третий этап – нелинейное преобразование. Здесь под преобразованием понимается отображение каждого многомерного вектора на другие векторы. Каждый отображенный вектор представляет патч высокого разрешения, а вектора содержат новый набор карт признаков.

Преобразование одного $n_1$-мерного вектора в другой $n_2$-мерный вектор осуществляется по формуле:

$$F_2(y) = \max(0, W_2 * y + B_2) \quad (3)$$

где $W_2 \in R^{n1 \times 1 \times 1 \times n2}$ – фильтр, $B_2 \in R^{n2}$ – вектор сдвига.

Четвертый этап – восстановление (генерирование) изображения высокого разрешения.

Отображения высокого разрешения каждого патча из предыдущего шага используются для генерации изображения высокого разрешения. На этом шаге из $n_2$-мерного вектора строится искомое изображение высокого разрешения по формуле (4):

$$F(y) = W_3 * F_2(y) + B_3 \quad (4)$$

где $W_3 \in R^{n2 \times f2 \times f2 \times c}$ – фильтр, $B_3 \in R^c$ – вектор сдвига.

Генеративно-состязательные сети представляют собой систему из двух нейронных сетей, выполняющих роли генератора и дискриминатора.

Нейронная сеть, выполняющая роль генератора, пытается сгенерировать образы, которые смогут обмануть дискриминатор и убедить его в том, что предоставляемые образы не были созданы искусственно. Нейронная сеть, выполняющая роль дискриминатора, пытается как можно точней отличать реальные образы из целевой выборки от сгенерированных образов.

Вся сложность обучения этой модели заключается в том, что, во-первых, необходимо обучать сразу две нейронных сети, а во-вторых, нужно хорошо настроить баланс между ними. Генератор обучается создавать более качественный результат, дискриминатор обучается распознавать реальные данные от сгенерированных. Если одна из сетей будет обучена лучше другой, то модель в целом не будет работать корректно.

Модель обучается методом обратного распространения ошибки. Для обучения используется следующий алгоритм:

- делается подготовка реальных данных. В задаче увеличения изображений – это исходные изображения;

- генерируется шум, на базе которого генератор генерирует данные;
- создается набор данных для обучения дискриминатора. Он состоит из настоящих данных, подготовленных на шаге 1 (им присваивается метка 1) и подделок от генератора, полученных на предыдущем шаге (им присваивается метка 0);
- проводится обучение дискриминатора, в процессе которого на вход подаются реальные данные и созданные генератором;
- проводится обучение генератора. На данном этапе обучается генератор, а обучение дискриминатора отключено. На вход подается шум, на выходе ожидается получение отметки 1, т. е. распознавание дискриминатором данных как реальных. При обучении генератора не используются реальные изображения, а используется только отметка дискриминатора.

Рассмотрим пример генеративно-состязательные сети для задачи увеличения разрешения (SRGAN).

Входные данные – изображения с низким разрешением - $I^{LR}$. Результат работы нейронной сети - $I^{SR}$. Изображение высокого разрешения (без обработки) - $I^{HR}$. $I^{LR}$ будем получать уменьшением $I^{HR}$ в n раз.

В результате обучения будет получена генерирующая функция G.

Генератор будет обучаться как сверточная нейронная сеть прямого распространения с параметрами $\Theta_G$ (5):

$$\theta_G = \{ W_{1:L} ; b_{1:L} \}, \quad (5)$$

где $W_{1:L}$ и $b_{1:L}$ — это веса и смещения нейронной сети глубиной в L слоев. Этот параметр вычисляется с помощью оптимизации составной функции потерь $l_{SR}$. Для обучения нейронной сети нужно значение:

$$\hat{\theta}_G = \min \frac{1}{N} l_{SR}(G(I_n^{LR}), I_n^{HR}) \quad (6)$$

Функция потерь $l_{SR}$ (7)

$$l_{SR} = l_{MSE} + 10^{-3} l_{GEN}, \quad (7)$$

где $l_{MSE}$ - средняя квадратическая ошибка при сравнении сгенерированного изображения $I^{SR}$ с его оригиналом $I^{HR}$,

Соревновательная функция потерь $l_{GEN}$ рассчитывается по (8):

$$L_{GEN}=\sum_{n=1}^{N} \log(D(G(I^{LR}))), \quad (8)$$

где $D(G(I^{LR}))$ - вероятность распознавания дискриминатором того, что созданное генератором изображение $G(I^{LR})$ является изображением высокого разрешения $I^{HR}$.

При сравнении предполагается изображений не MSE, а предобученную нейронную сеть VGG-19, т.к. результаты ее работы больше соответствуют тому, как человеческий глаз воспринимает разницу между двумя изображениями. [5]

Стоит также отметить, что применение нормализации к входным данным нейронной сети позволяет увеличить эффективность работы алгоритма.

Архитектура генеративной нейронной сети G состоит из следующих блоков: 64 карты признаков, за которыми следуют слои нормализации, 2 сверточных слоя с ядром 3x3 и 2 субпиксельных сверточных слоя. Функция активации - Parametric ReLU (9):

$$f(x) = \begin{cases} x, & x > 0 \\ ax, & x \leq 0 \end{cases}, \quad (9)$$

где x – входной сигнал, a – коэффициент вычисляемый в результате обучения модели.

Архитектура генератора показана на рисунке 2.

На изображении 2.2 применены следующие обозначения:

- Input $I^{LR}$ – изображение низкого разрешения поступающее на вход генератору;
- Conv – сверточный слой с n картами признаков и шагом равным s;
- ReLU – активационная функция Parametric ReLU;
- BN – Batch Normalization, батч-нормализация;

- skip connection – дополнительные связи;
- B residual blocks – остаточные блоки;
- Elementwise sum – поэлементная сумма;
- PixelShuffler x2 – субпиксельные сверточные слои;
- $I^{SR}$ – результат работы нейронной сети.

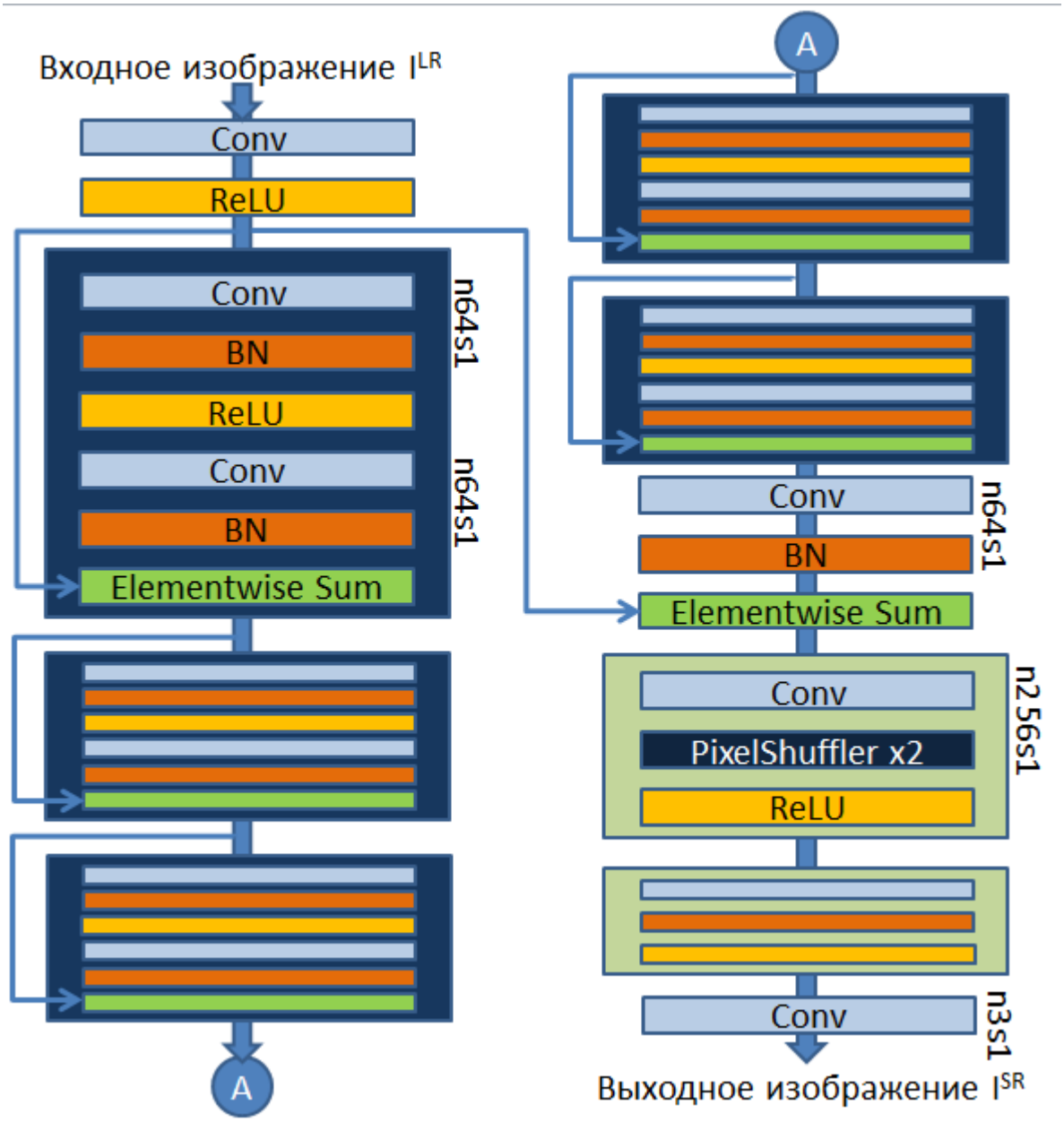


Рисунок 2 – Архитектура генератора SRGAN

Чтобы отличить изображения высокого разрешения $I^{HR}$ от сгенерированных изображений $I^{SR}$, производится обучение дискриминатора D.

На вход дискриминатору в случайном порядке подаются или изображения $I^{HR}$ или $I^{SR}$. Тогда должен получиться 0 в случае, если изображение было распознано как $I^{SR}$ и 1 в случае, если распознано как $I^{HR}$.

Сеть состоит из 8 сверточных слоев с ядром 3x3, количество которых увеличивается в 2 раза с 64 до 512, за которыми следуют 2 полносвязных слоя.

В качестве активационной функции сверточных слоев используется Leaky ReLU с a = 0,02:

$$f(x) = \begin{cases} x, & x > 0 \\ 0{,}02x, & x \leq 0, \end{cases} \quad (10)$$

где x – входной сигнал.

Активационной функцией для второго полносвязного слоя служит сигмоида (11):

$$\sigma(x) = \frac{1}{1+e^{-x}} \quad (11)$$

где x – входной сигнал.

В дискриминаторе, как и в генераторе, используется батч-нормализация.

Архитектура дискриминатора показана на рисунке 3.

На изображении 2.3 применены следующие обозначения:

- Input – входные данные, изображения созданные генератором и изображения высокого разрешения;
- Conv – сверточный слой с *n* картами признаков и шагом равным *s*;
- Leaky ReLU – активационная функция Leaky ReLU;
- BN – Batch Normalization, батч-нормализация;
- Dense – полносвязный слой;

- Sigmoid – активационная функция сигмоида;
- $I^{HR}$, $I^{SR}$ – изображение высокого и изображение низкого разрешения, а также метка класса изображения, т.е. настоящее изображение высокого разрешения или созданное генератором.

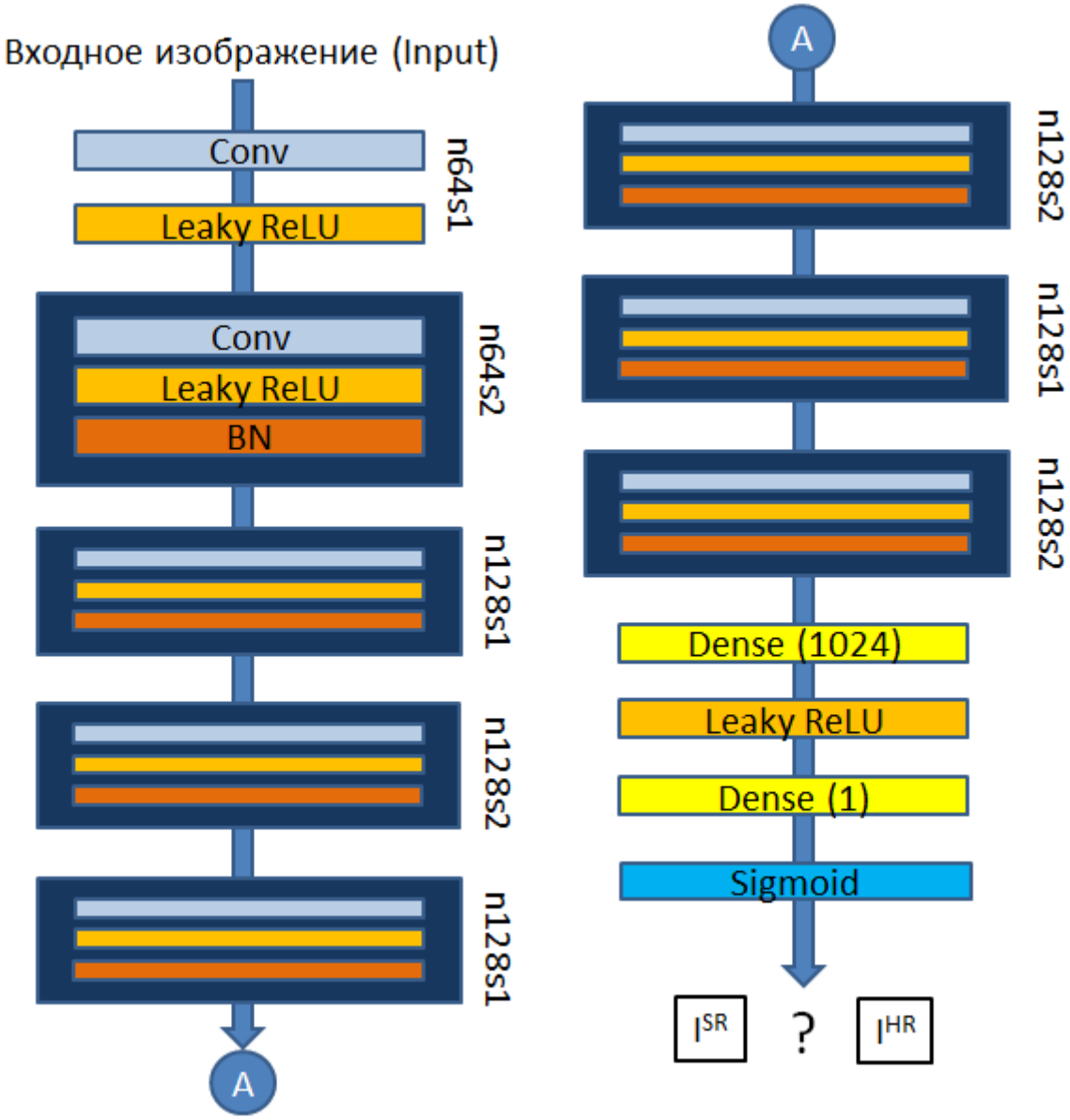


Рисунок 3 – Архитектура дискриминатора SRGAN

SRGAN показывает хорошие результаты, но его качество все еще недостаточно хорошее и может иметь артефакты в результате работы. Для того чтобы исправить это и увеличить производительность работы сети SRGAN был переработан в ESRGAN.

Главная цель состоит в том, чтобы улучшить качество для SR. Для этого нужно подкорректировать архитектуру сети, затем улучшить дискриминатор и подправить функцию потерь, а для балансировки качества ввести сетевую интерполяцию.

Для того чтобы улучшить качество изображения в архитектуре генератора SRGAN из всех блоков, будут удалены слои BN.

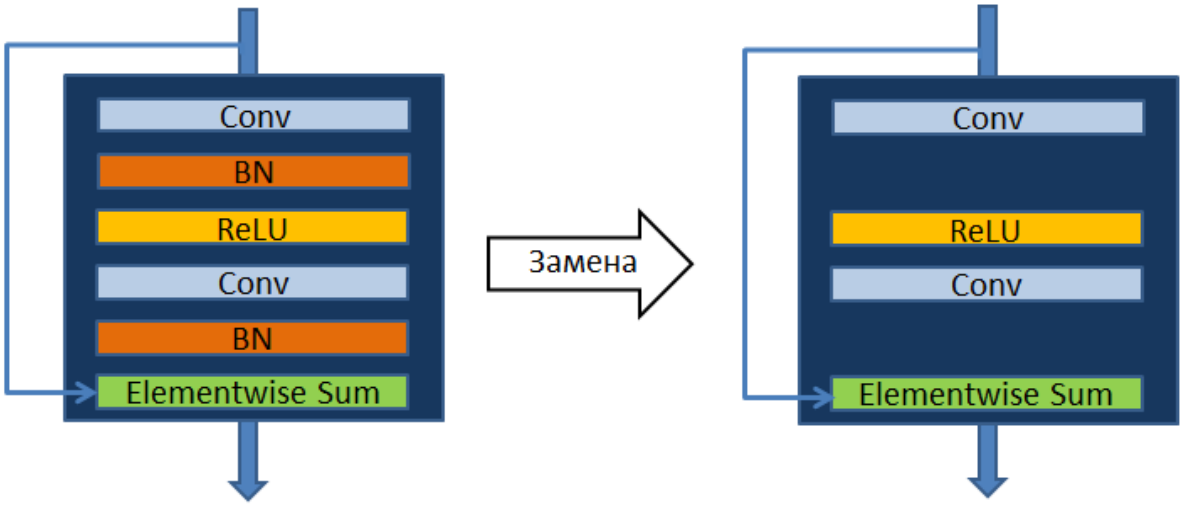


Рисунок 4 – Изменения в архитектуре генератора SRGAN

Удаление BN слоев повышает производительность и снижает вычислительную сложность в различных PSNR-ориентированных задачах. BN слои нормализуют функции, используя среднее значение и дисперсию в процессе обучения. Когда обучающие и тестовые наборы данных сильно отличается, BN слои, как правило, выдают артефакты в результате, и ограничивают возможность обобщения.

В дополнении к улучшенной архитектуре, также используется несколько методов для облегчения обучения глубокой сети:

- остаточное масштабирование;
- уменьшение инициализации, т.к. эмпирическая и остаточная архитектура легче обучаются когда начальное отклонение параметра становится меньше.

Помимо улучшений генератора, был также улучшен дискриминатор, теперь это Relativistic average Discriminator RaD($D_{Ra}$). В отличии от стандартного дискриминатора D в SRGAN, который оценивает вероятность того, что входное изображение x является реальным, релятивистский дискриминатор пытается предсказать вероятность того, что изображение стало более реалистичным, рисунок 5.


Рисунок 5 – Модификация дискриминаторами SRGAN

Дискриминатор SRGAN основывался на (12):

$$D(x) = \sigma(C(x)), \tag{12}$$

где $\sigma$ – функция сигмоиды, C(x) – не трансформированный выход дискриминатора.

После модификации SRGAN работа дискриминатора будет осуществляться по формуле (13).:

$$D_{Ra}(x_r, x_f) = \sigma(C(x_r) - E_{xf}[C(x_f)]), \tag{13}$$

где $E_{xf}$ – среднее для всех минибатчей.

Ошибка дискриминатора определяется по выражению (14):

$$L_D^{Ra} = -E_{x_r}\left[\log\left(D_{Ra}(x_r, x_f)\right)\right] - E_{x_f}\left[\log\left(1 - D_{Ra}(x_f, x_r)\right)\right] \tag{14}$$

Состязательная потеря генератора имеет рассчитывается по формуле (15) :

$$L_G^{Ra} = -E_{x_r}\left[\log\left(1 - D_{Ra}(x_r, x_f)\right)\right] - E_{x_f}\left[\log\left(D_{Ra}(x_f, x_r)\right)\right], \quad (15)$$

где $x_f = G(x_i)$ и $x_i$ состоят из входных изображений LR. $x_r$ и $x_f$ содержатся в состязательной потери для генератора.

Для того, чтобы удалить шум на изображении в результате работы сети и при этом не потерять качество, была использована гибкая и эффективная сетевая интерполяция. Сначала тренируется PSNR-ориентированная сеть $G_{PSNR}$, после чего получается GAN на основе $G_{GAN}$ с тонкой настройкой. Интерполируя все соответствующие параметры этих двух сетей получаем интерполированную модель $G_{INTERP}$, имеющая вид (16):

$$\theta_G^{INTERP} = (1 - \alpha) * \theta_G^{PSNR} + \alpha * \theta_G^{GAN}, \quad (16)$$

где $\theta_G^{INTERP}, \theta_G^{PSNR}, \theta_G^{GAN}$ – параметры $G_{INTERP}$, $G_{PSNR}$, $G_{GAN}$ и $\alpha$ – параметр интерполяции $\in [0, 1]$.

Такая сетевая интерполяция имеет два преимущества. Во-первых, интерполированная модель способна обеспечить лучше результаты для любого возможного $\alpha$ без потери качества. Во-вторых, появилась возможность настраивать качество и точность без повторного обучения модели.

## Выводы по главе

Рассмотрены классические методы увеличения разрешения изображения, основанные на интерполяции. Также описаны технологии генерирования изображений с высоким разрешением, основанные на использовании глубоких нейронных сетей. Предложена структура нейронной сети, подходящая для решения задачи увеличения разрешения изображений.

# 3 Программная реализация приложения для нейросетевого увеличения разрешения изображений

## 3.1 Подготовка данных

При настройке искусственных нейронных сетей важно подобрать качественный материал для обучения и тестирования. Для того, чтобы алгоритм хорошо произвел обработку, изображения должны быть одного формата, качества и размера.

Одним из простых способов получения данных LR является ухудшение данных HR. Это часто делается с помощью размытия или добавления шума. Особенности JPEG и аномалии квантования также могут быть использованы для ухудшения изображения.

Есть свободные ресурсы с базами изображений, например CIFAR-10, MNIST, Labelme. Большинство изображений там в виде .gz или .zip архивов.

Для обучения искусственной нейронной сети лучше использовать изображения с большим количеством деталей и в цветовом пространстве RGB, т.к. этот формат является одним из наиболее подходящих для хранения реалистичных изображений с большим количеством деталей.

## 3.2 Средства разработки программного обеспечения

Для решения задач машинного обучения и обработки изображений используются множество языков программирования. Существует много различных библиотек, в которых есть наиболее часто используемые методы и алгоритмы.

В качестве языка программирования выбран Python. Язык обладает следующими преимуществами:

- популярность в академической среде при использовании задач, связанных с машинным обучением и интеллектуальной обработкой данных. Поэтому приложение с использованием этого языка будет понятна и доступна большому кругу исследователей;
- наличие большого числа компонентов, упрощающих работу с алгоритмами машинного обучения. В Python существует много библиотек для работы с изображениями. Стоит отметить легкость установки библиотек с помощью встроенных в язык менеджеров пакетов;
- Python - это язык программирования высокого уровня, который позволяет быстро проектировать прототипы программ позволяя сократить время разработки;
- Python поддерживает модульность благодаря тому, что каждый файл в проекте может быть выполнен как отдельный скрипт или может быть использованным при импорте отдельных методов.

Но у Python есть и минусы. Из-за того, что он интерпретируемый язык программирования, реализованные на нем алгоритмы, могут работать медленней, чем на компилируемых языках программирования. Однако этот минус частично компенсируется использованием библиотек, в которых была произведена оптимизация с использованием исполняемого кода на других языках.

Для программной реализации были использованы следующие компоненты:

- библиотека NumPy, для использования математических функций в промежуточных расчётах;
- библиотека OpenCV для применения различных преобразований над изображение;
- библиотека PyTorch, для моделирования работы глубоких нейронных сетей;
- библиотека PyQt, для создания графического интерфейса.

NumPy – это библиотека, которая включает в себя множество методов по работе с многомерными массивами данных. Помимо работы с массивами, в данной библиотеке реализовано множество таких полезных вещей, как функции линейной алгебры, преобразования Фурье, генерации случайных чисел и т.д. [13]

Как уже упоминалось ранее в этой главе, Python, являясь интерпретируемым языком программирования, как правило, обрабатывает сложные математические алгоритмы медленнее, чем компилируемые языки. Авторы NumPy постарались разрешить эту проблему, предложив модуль с исполняемым кодом функций на языках C и Fortran.

Библиотека NumPy настолько популярна, что зачастую интегрирована в другие библиотеки. Так, например, функции из библиотеки OpenCV могут обрабатывать массивы из библиотеки NumPy.

OpenCV – это библиотека с открытым исходным кодом, содержащая наборы специальных типов данных и методы для работы с изображениями. Написана и оптимизирована на C/C++, также имеет интерфейсы для Python, Java, Ruby, Lua и других языков.

Эта библиотека представляет из себя набор модулей. Каждый из этих модулей связан с определенной областью компьютерного зрения.

Алгоритмы, реализованные в этой библиотеке, протестированы и оптимизированы, а значит, подобно библиотеке NumPy, позволяют частично уменьшить потери производительности, связанные с использованием интерпретатора Python. Библиотека включает в себя более 1000 функций и алгоритмов.

OpenCV является достаточно широко используемой библиотекой благодаря ее свободной лицензии BSD. Она применяется такими компаниями, как NVidia, WillowGarage, Intel, Google. Компании NVidia и WillowGarage частично спонсируют ее разработку. [7]

PyTorch — библиотека машинного обучения для языка Python с открытым исходным кодом, созданная на базе Torch. Используется для

решения различных задач: компьютерное зрение, обработка естественного языка. Разрабатывается преимущественно группой искусственного интеллекта Facebook. Также вокруг этого фреймворка выстроена экосистема, состоящая из различных библиотек, разрабатываемых сторонними командами: Fast.ai упрощающая процесс обучения моделей, Pyro модуль для вероятностного программирования от Uber, Flair для обработки естественного языка. [6]

PyTorch предоставляет две основные высокоуровневые модели:

- тензорные вычисления (по аналогии с NumPy) с развитой поддержкой ускорения на GPU;
- глубокие нейронные сети на базе системы autodiff.

В настоящее время почти все приложения, которые создаются для конечного пользователя, имеют GUI.

Под графическим интерфейсом пользователя (GUI) подразумеваются все те окна, кнопки, текстовые поля для ввода, скроллеры, списки, радиокнопки, флажки и др., которые пользователь видит на экране, открывая то или иное приложение. Через них пользователь взаимодействует с программой и управляете ею.

Существует множество библиотек GUI. Tkinter далеко не самая популярная, хотя с ее использованием написано не мало проектов. Установочный файл Python обычно уже включает пакет Tkinter в составе стандартной библиотеки наряду с другими модулями.

Tkinter хорошо подходит для создания небольших графических модулей, модальных окон и легкого пользовательского интерфейса, но для больших проектов больше подходит библиотека PyQt. PyQt представляет собой инструмент для взаимодействия с графическим фреймворком Qt, выполненный в виде расширения Python.

PyQt также включает в себя Qt Designer (Qt Creator) — дизайнер графического интерфейса пользователя. Программа pyuic генерирует Python

код из файлов, созданных в Qt Designer. Это делает PyQt очень полезным инструментом для быстрого прототипирования. Кроме того, можно добавлять новые графические элементы управления, написанные на Python, в Qt Designer.

### 3.3 Структура приложения

Структура приложения показана на рисунке 6.

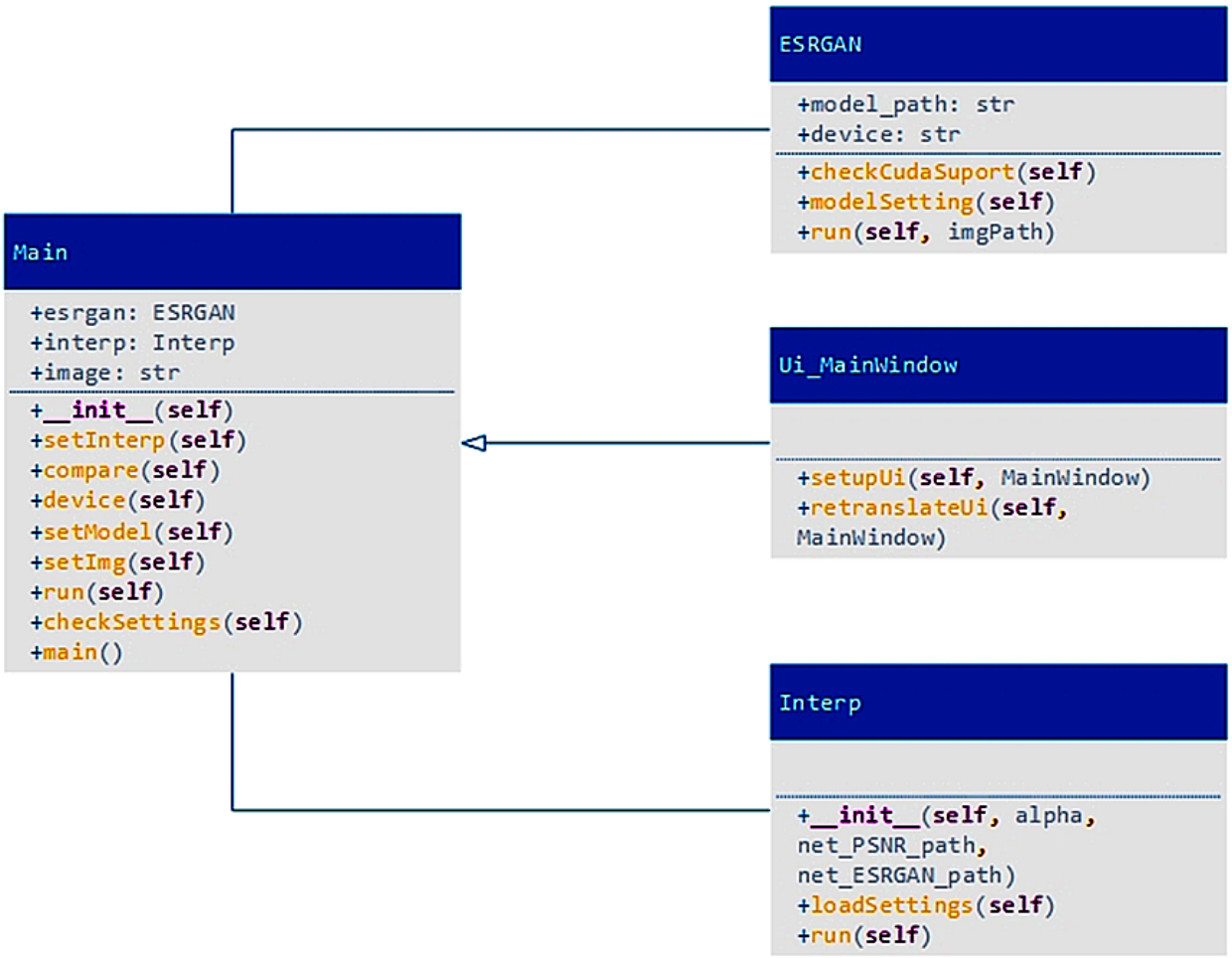


Рисунок 6 – Структура приложения

Приложение состоит из 4 классов:

- Interp – класс создающий модель с интерполяцией. Конструктор в качестве аргумента принимает 3 параметра: 2 пути моделей и аргумент

интерполяции a находящийся в диапазоне от 0 до 1. Метод run запускает создание модели.

- ESRGAN – класс работающий с исходным изображением имеет 2 поля: путь до модели и режим работы. Метод checkCudaSuport проверяет возможность работы приложения на видеокарте и возвращает результат проверки, если технология cuda доступна режим по умолчанию будет выбран с ней, в противном случае приложение будет работать на центральном процессоре. Метод modelSetting задает настройки для выполнения работы. И метод run запускает преобразование LR в HR сохраняя результат в папку и возвращает путь до него.
- Ui_MainWindow – класс автоматически создаваемый PyQt5. Содержит в себе все элементы графического интерфейса и его верстку задаваемые методом Ui_MainWindow. Также создает экшены для меню в методе retranslateUi.
- Main – главный класс наследуется от Ui_MainWindow и является запускаемым для приложения. В конструкторе класса экшенам меню присваиваются эвенты для обработки действий пользователя и первоначальная настройка приложения. Метод main инициализирует форму приложения и запускает ее отображение. Метод checkSettings проверяет настройки приложения на корректность и задает отображаемые в форме подсказки. Эвент setInterp вызывает диалоговое окно с помощью tkinter для ввода значения интерполяции от 0 до 1, при вводе числа выходящего за этот диапазон в качестве аргумента будет выбрано значение 1. Эвент compare служит лишь для сравнения результата работы приложения с классическим методом увеличения разрешения. Эвент device служит для изменения режима работы приложения между вычислением на cpu и cuda. Эвент setModel вызывает диалоговое окно для выбора файла модели с разрешением .pth из папки models если такая существует и задает эту модель для вычисления в классе esrgan. Эвент setImg вызывает диалоговое окно для выбора

изображения низкого разрешения. Эвент run запускает метод run в esrgan засекая время его выполнения и выводит результат работы в приложении, также показывает время выполнения работы.

### 3.4 Графический интерфейс приложения

При запуске приложения появляется главное окно программы показанное на рисунке 7.

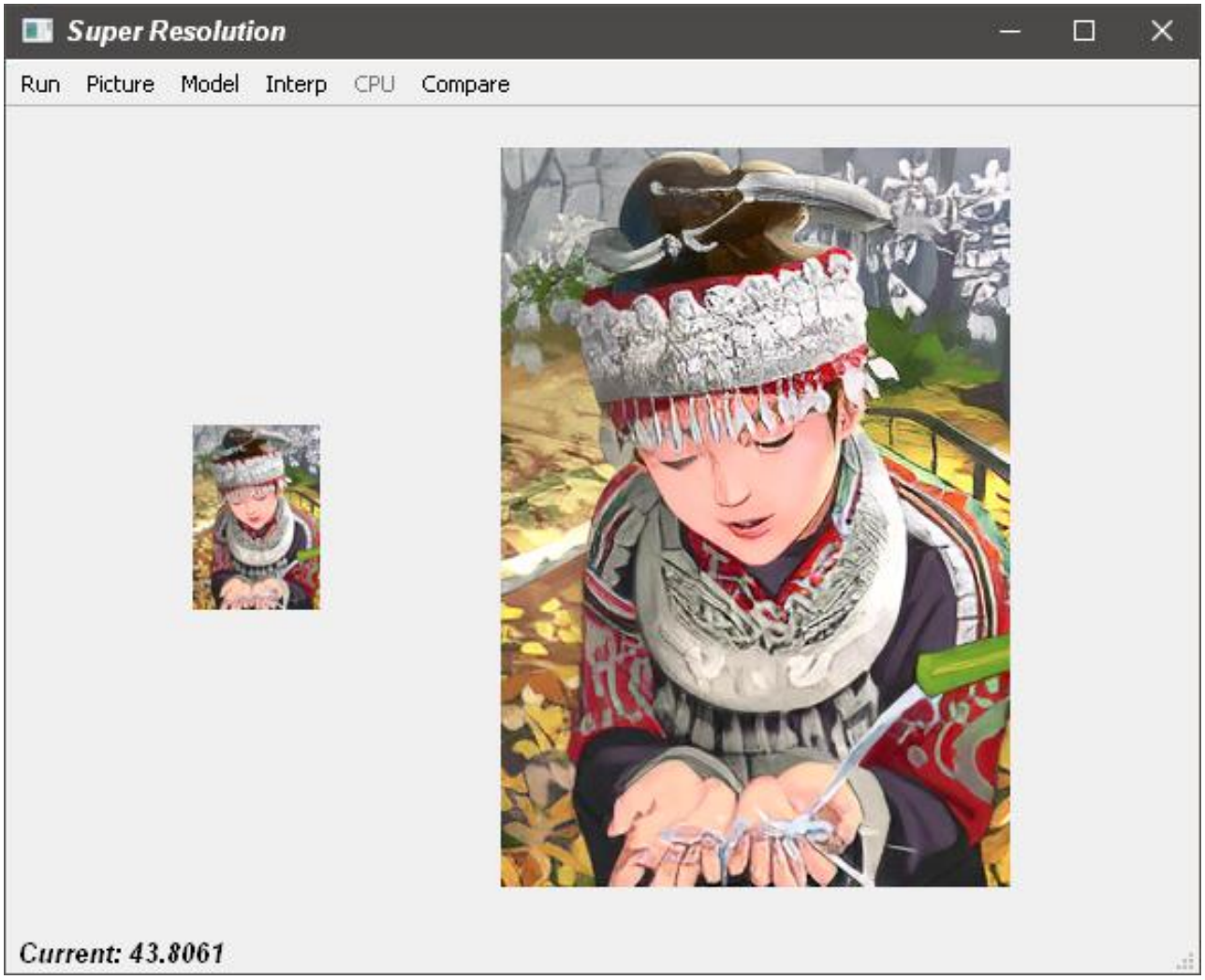


Рисунок 7 – Главное окно приложения

Главная форма представляет из себя рабочую область над которым расположено меню приложения, состоящее из:

- Run – запуск работы нейросети. По умолчанию неактивно и нажать на него можно только при выбранной модели нейронной сети и выбранном изображении для обработки;
- Picture – открытие диалогового окна для выбора изображения на обработку;
- Model – открытие диалогового окна для выбора модели нейронной сети;
- Interp – открытие диалогового окна для ввода параметра интерполяции и последующее создание модели с ней.
- CPU/CUDA – выбор режима работы приложений, а именно где будут проводиться вычисления на процессоре или видеокарте. Если технология CUDA не доступна для приложения кнопка будет всегда неактивна;
- Compare – с помощью бикубической интерполяции разрешение исходного изображения будет подогнано под разрешение результирующего изображения, для сравнения работы классического метода увеличения разрешения и работы нейронной сети.

В результате работы приложения из исходного изображения низкого разрешения(слева) было получено изображение высокого разрешения(справа) за время отображающееся под рабочей областью.

### 3.5 Результаты тестирования приложения

Приведем несколько примеров работы приложения. Примеры показаны на рисунках 8 – 10.

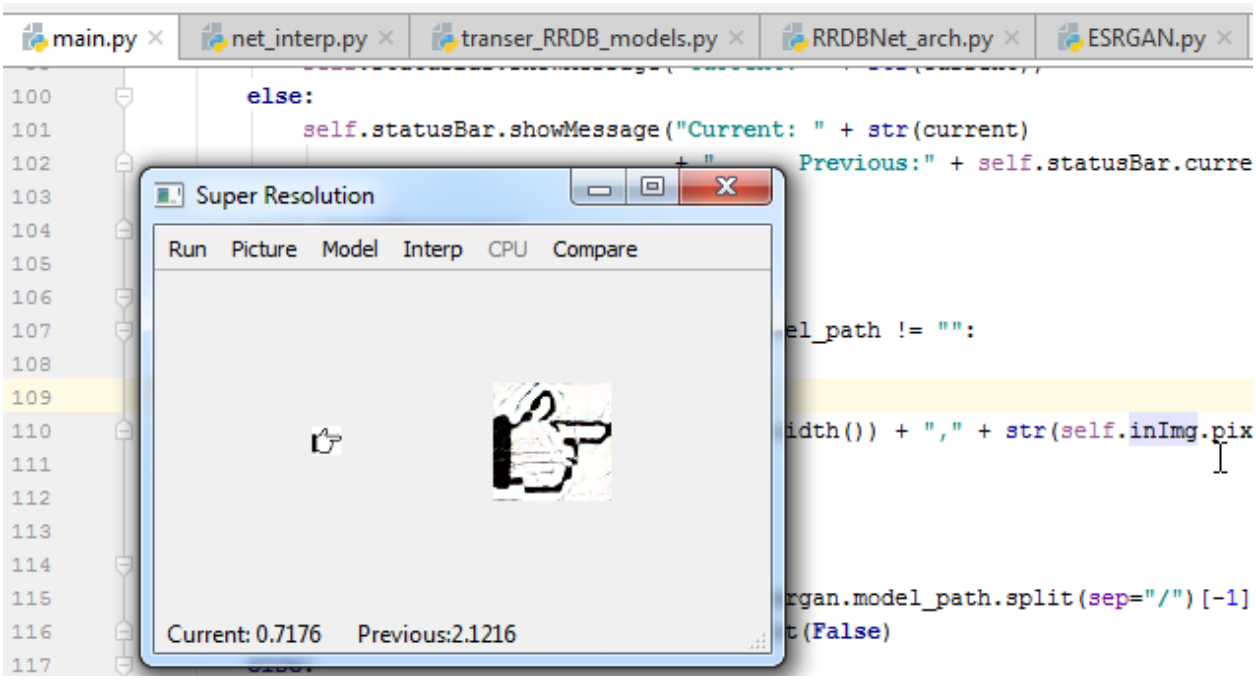


Рисунок 8 – Увеличение разрешения простого двухцветного изображения в 4 раза (скорость обработки 0,7176 секунды)

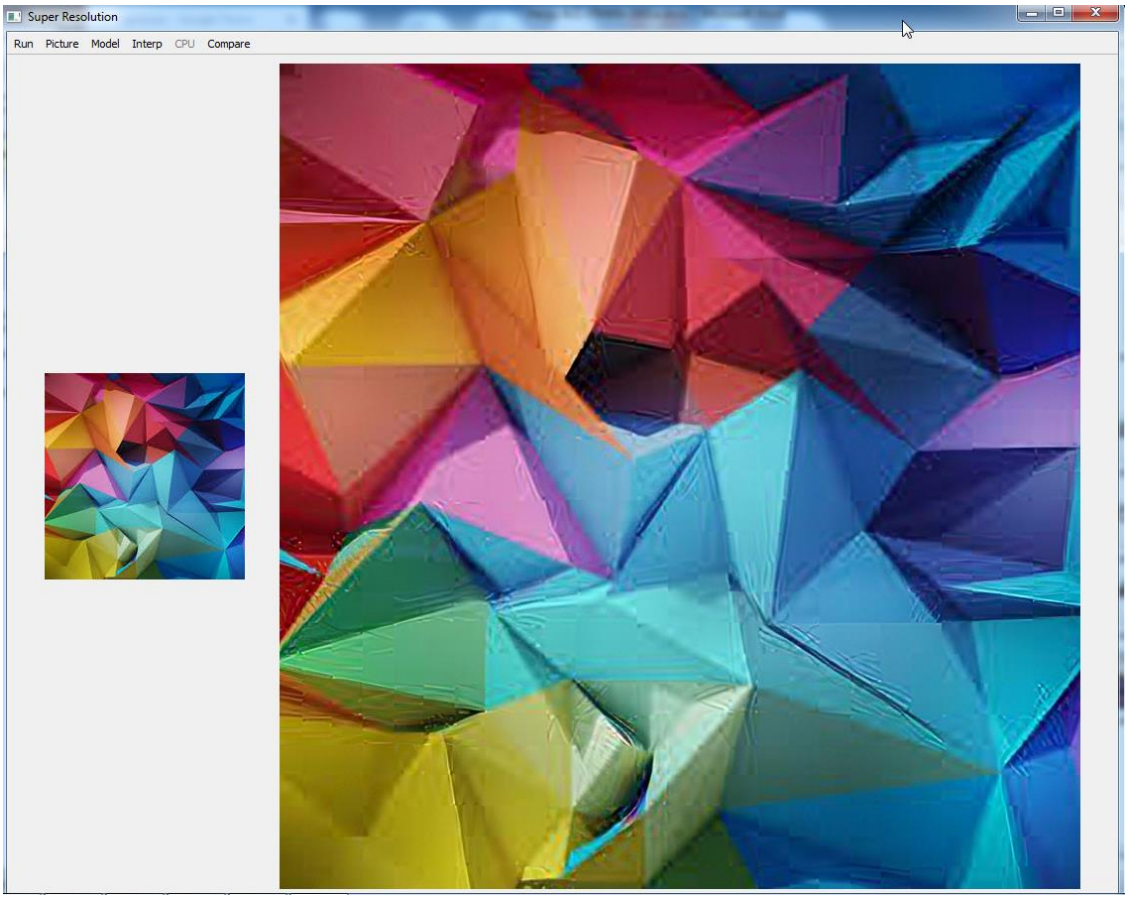


Рисунок 9 – Увеличение разрешения простого цветного узора в 4 раза (скорость обработки 168 секунд)

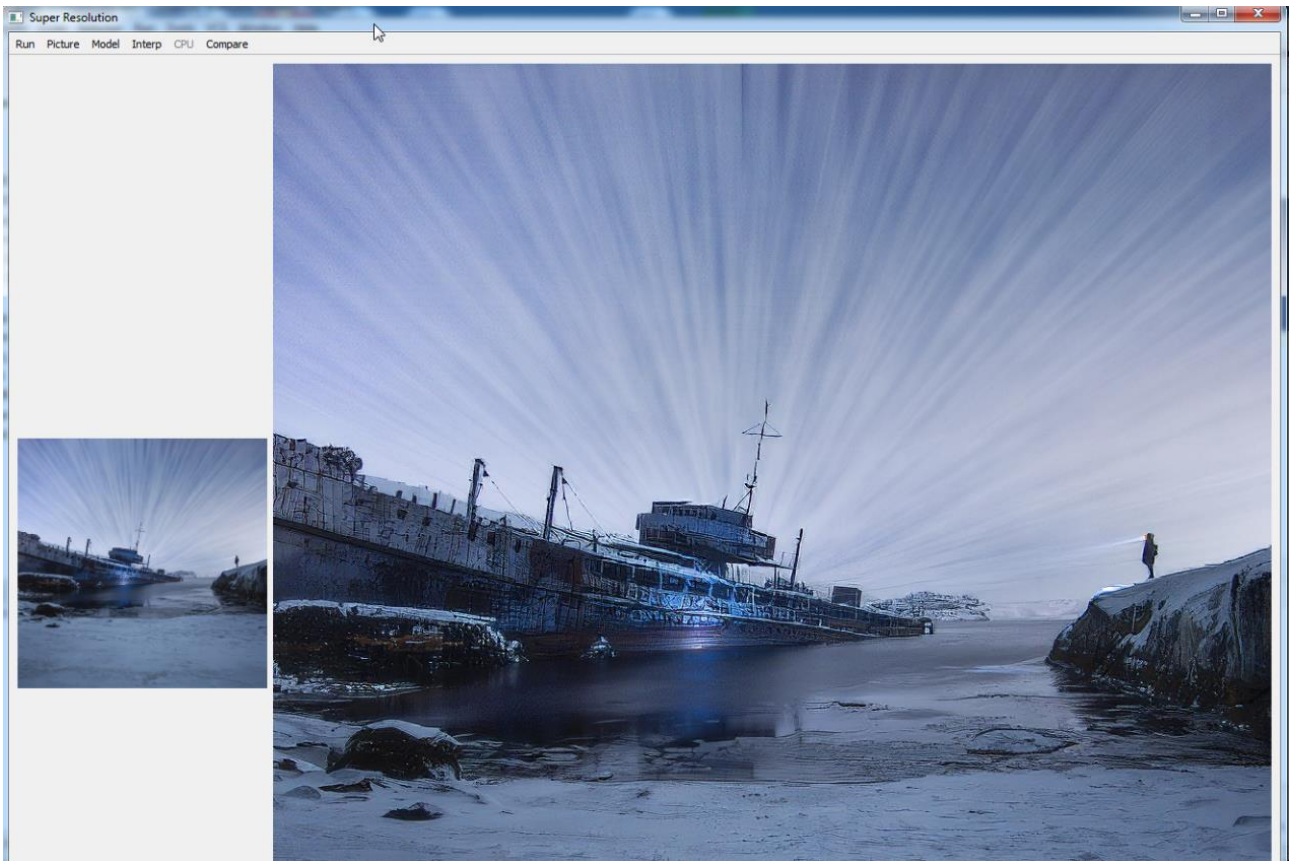


Рисунок 10 – Увеличение разрешения сложного цветного изображения в 4 раза (скорость обработки 329 секунд)

Как видно, из рисунков 8-10 разработанное приложение успешно справляется с поставленной задачей.

## Выводы по главе

На языке программирования Python разработано приложение с графическим интерфейсом позволяющее увеличивать разрешения изображения на основе работы глубоких нейронных сетей. Для моделирования работы слоев нейронной сети используется библиотека PyTorch. Работа приложения протестирована на различных изображениях.

## Заключение

В результате написания бакалаврской работы были сделаны следующие выводы:

1. Алгоритмы анализа, используемые в системах компьютерного зрения, чувствительны к качеству изображения. Увеличение разрешения изображений возможно за счет использования более совершенного аппаратного обеспечения, но это является дорогостоящим решением проблемы. Другим путем является предобработка изображений с помощью алгоритмов, позволяющих повышать их разрешение. Поэтому актуальным исследовательской задачей остается разработка и совершенствование алгоритмов повышения разрешения изображений.

2. Анализ литературы показал, что при решении задачи увеличения изображений лучше всего себя показывают сверточные нейронные сети и генеративно-состязательные сети.

3. В исследовании предложены модификации генеративно-состязательные сети SRGAN, заключающиеся в удалении слоев батч-нормализация в генераторе для повышения его производительности и снижении вычислительной сложности расчётов. Также предложены изменения по работе дискриминатора в части классификации изображений.

4. На языке Python разработано приложение, которое демонтирует возможности предложенного варианта нейронной сети при решении задачи увеличения разрешения изображений. Работа приложения протестирована на различных изображениях.

## Список используемой литературы

1. Аверин, Д.С. Разработка структуры нейронной сети для увеличения разрешения изображения / Д.С. Аверин, Е.М. Марков // Информационные технологии в науке, промышленности и образовании: Сборник трудов региональной научно-технической конференции 31 мая 2018 г. – Ижевск : Ижевский государственный технический университет имени М.Т. Калашникова, 2018. – С. 107-113. – Текст : непосредственный.

2. Блажевич, С.В. К вопросу о методах повышения качества цифровых изображений / С.В. Блажевич, Е.С. Селютина // Научные ведомости Белгородского государственного университета. Серия: Математика. Физика. – Федеральное государственное автономное образовательное учреждение высшего образования «Белгородский государственный национальный исследовательский университет», 2015. – С. 5-12. – Текст : непосредственный.

3. Болотин, Е.В. Применение нейронных сетей для задачи улучшения качества изображений и видеоматериалов / Е.В. Болотин, Е.М. Марков // Гагаринские чтения: Сборник тезисов докладов XLV Международной молодежной научной конференции 16-19 апреля 2019 г. – Москва : Московский авиационный институт (национальный исследовательский университет), 2019. – С. 197-198. – Текст : непосредственный.

4. Ганьшин, К.Ю. Суперразрешение облаков точек, захватываемых Microsoft KINECT / К.Ю. Ганьшин, С.А. Ржевский // Наука: прошлое, настоящее, будущее: сборник статей Международной научно-практической конференции в 3 частях, 25 июня 2017 г. – Уфа : Общество с ограниченной ответственностью "Аэтерна", 2017. – С. 21-25. – Текст : непосредственный.

5. Гаранина, М.С. Получение суперпиксельного изображения методом суперразрешения / М.С. Гаранина, С.В. Машкин // Физика для пермского края: Материалы региональной научно-практической

конференции студентов, аспирантов и молодых ученых 20-29 апреля 2019 г. (под общей редакцией Н. Н. Картавых). – Пермь : Пермский государственный национальный исследовательский университет, 2019. – С. 131-136. – Текст : непосредственный.

6. Кокошкин, А.В. Оценка ошибок синтеза изображений с суперразрешением на основе использования нескольких кадров / А.В. Кокошкин, В.А. Коротков, К.В. Коротков, Е.П. Новичихин // Компьютерная техника. – Федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королёва» (Самарский университет), 2017. – С. 11-21. – Текст : непосредственный.

7. Крылов, А.С. Регуляризирующие методы интерполяции изображений / А.С.Крылов, А.В.Насонов. – М : АРГАМАК-МЕДИА, 2014. – 100 с. – Текст : непосредственный.

8. Минаев, С.А. Увеличение разрешения изображения с применением выделенных образцов / С.А. Минаев, С.А. Ржевский // Технические науки: проблемы и решения, сборник статей по материалам XXXI международной научно-практической конференции, Москва, 21 января 2020 г. – Москва : Общество с ограниченной ответственностью "Интернаука", 2020. – С. 5-8. – Текст : непосредственный.

9. Насонов А.В. Развитие методов повышения качества изображений лиц в видеопотоке / А.В. Насонов, А.С. Крылов, О.С. Ушмаев // Информатика и ее применения, 2009. – Т. 3, Вып. 1. – С.19-28. – Текст : непосредственный.

10. Сергеева, А.О. Проблема повышения качества изображения и её решение / А.О. Сергеева, Г.А. Чайникова // European Science Forum: сборник статей Международной научно-практической конференции 18 апреля 2019 г. – Петрозаводск : Международный центр научного партнерства «Новая Наука» (ИП Ивановская Ирина Игоревна), 2019. – С. 106-113. – Текст : непосредственный.

11. Шевкунов, И.А. Вычислительное пиксельное суперразрешение в безлинзовой осевой цифровой голографии / И.А. Шевкунов, Н.В. Петров, В.Я. Катковник // Сборник научных трудов VII Международная конференция по фотонике и информационной оптике, Москва, 24-26 января 2018 г. – Москва : Национальный исследовательский ядерный университет "МИФИ", 2018. – С. 568-569. – Текст : непосредственный.

12. Camponez, M.O. A Closed Form Algorithm for Superresolution / Marcelo O. Camponez, Evandro O. T. Salles, Mário Sarcinelli-Filho // ISVC 2011: Advances in Visual Computing – 7th International Symposium, ISVC 2011, Las Vegas, NV, USA, September 26-28, 2011. Proceedings, Part II. – Springer-Verlag Berlin Heidelberg, 2011. – pp. 338-347. – Text : direct.

13. Goldlücke, B.A Superresolution Framework for High-Accuracy Multiview Reconstruction / Bastian Goldlücke, Daniel Cremers // DAGM 2009: Pattern Recognition – 31st DAGM Symposium, Jena, Germany, September 9-11, 2009. Proceedings. – Springer-Verlag Berlin Heidelberg, 2009. – Text : direct.

14. Klatzer, T. Trainable Regularization for Multi-frame Superresolution / Teresa Klatzer, Daniel Soukup, Erich Kobler, Kerstin Hammernik, Thomas Pock // GCPR: German Conference on Pattern Recognition – 39th German Conference, GCPR 2017, Basel, Switzerland, September 12–15, 2017, Proceedings. – Springer International Publishing AG, 2017. – pp. 90-100. – Text : direct.

15. Peterson, L.E. Superresolution MUSIC Based on Marčenko-Pastur Limit Distribution Reduces Uncertainty and Improves DNA Gene Expression-Based Microarray Classification / Leif E. Peterson // CIBB 2013: Computational Intelligence Methods for Bioinformatics and Biostatistics – 10th International Meeting, CIBB 2013, Nice, France, June 20-22, 2013, Revised Selected Papers. – Springer International Publishing Switzerland, 2014. – pp. 194-209. – Text : direct.

16. Rivadeneira, R.E. Thermal Image SuperResolution Through Deep Convolutional Neural Network / Rafael E. Rivadeneira, Patricia L. Suárez, Angel D. Sappa, Boris X. Vintimilla // ICIAR: International Conference on Image Analysis and Recognition, 16th International Conference, ICIAR 2019, Waterloo,

ON, Canada, August 27–29, 2019, Proceedings, Part II – pringer Nature Switzerland AG, 2019. – pp. 417-426. – Text : direct.

17. Torii, A. PDE Based Method for Superresolution of Gray-Level Images / A. Torii, Y. Wakazono, H. Murakami, A. Imiya // CAIP 2003: Computer Analysis of Images and Patterns – 10th International Conference, CAIP 2003, Groningen, The Netherlands, August 25-27, 2003. Proceedings. – Springer-Verlag Berlin Heidelberg, 2003. – pp. 706-713. – Text : direct.

18. Vlasenko, A. Superresolution and Denoising of 3D Fluid Flow Estimates / Andrey Vlasenko, Christoph Schnörr // DAGM: Joint Pattern Recognition Symposium – 31st DAGM Symposium, Jena, Germany, September 9-11, 2009. Proceedings. – Springer-Verlag Berlin Heidelberg, 2009. – pp. 482-491. – Text : direct.

19. Wang, T. SuperResolution Image Reconstruction Using a Hybrid Bayesian Approach / Tao Wang, Yan Zhang, Yong Sheng Zhang // ICONIP: International Conference on Neural Information Processing – 13th International Conference, ICONIP 2006, Hong Kong, China, October 3-6, 2006. Proceedings, Part II. – Springer-Verlag Berlin Heidelberg, 2006. – pp. 412-419. – Text : direct.

20. Zhang, Y. Limited Recurrent Neural Network for Superresolution Image Reconstruction / Yan Zhang, Qing Xu, Tao Wang, Lei Sun // ICONIP 2006: Neural Information Processing – 13th International Conference, Hong Kong, China, October 3-6, 2006. Proceedings, Part II. – Springer-Verlag Berlin Heidelberg, 2006. – pp. 304-313. – Text : direct.