МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики

Кафедра геоинформатики и информационной безопасности

**ОТЧЕТ ПО ПРАКТИКЕ**

|  |  |
| --- | --- |
| Вид практики: | производственная практика |
|  | (учебная, производственная) |
| Тип практики: | научно-исследовательская работа |

Сроки прохождения практики: с 01.09.2021 г. по 28.12.2021 г.

по направлению подготовки 10.05.03 Информационная безопасность автоматизированных систем

(уровень академического специалитета)

направленность (профиль) «Обеспечение информационной безопасности распределенных информационных систем.»

|  |  |
| --- | --- |
| Студент группы | № 6512-100503DА.Н.Коновалов |
| Руководитель практики  от университета | Мясников В.В. |

Дата сдачи 28.12.2021 г.

Дата защиты 28.12.2021 г.

Оценка \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Самара 2021

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева»

(Самарский университет)

Институт информатики, математики и электроники

Факультет информатики

Кафедра геоинформатики и информационной безопасности

**Индивидуальное задание на практику**

Студенту группы № 6512-100503DА.Н.Коновалову

|  |
| --- |
| Направление на практику оформлено приказом по университету 30.08.21 № 357-ПР на кафедру геоинформатики и информационной безопасности Самарского университета |
| (наименование профильной организации или структурного подразделения университета) |

|  |  |  |
| --- | --- | --- |
| **Планируемые результаты освоения образовательной программы (компетенции)** | **Планируемые результаты практики** | **Содержание задания** |
| ОК-1 | Знать:  основные направления, проблемы, теории и методы философии, содержание современных философских дискуссий по проблемам общественного развития  Уметь:  использовать положения и категории философии для оценивания и системного анализа различных социальных тенденций, фактов и явлений и моделирования процессов в научной  деятельности; Владеть: - навыками анализа текстов, имеющих философское содержание | Изучить сущность методов реконструкции изображений, а также теоретические материалы, служащие предпосылками к пониманию этой сущности. |
| ОК-2 | Знать:  базовые экономические понятия, законы функционирования экономики и поведения экономических агентов, показатели макроэкономического уровня развития страны, экономические показатели, используемые для оценки производственно- хозяйственной деятельности промышленных предприятий Уметь:  использовать понятийный аппарат экономической науки для описания экономических и финансовых процессов  Владеть: навыками использования экономических знаний в сфере личных финансов и профессиональной деятельности. | Проанализировать современную актуальность проблемы реконструкции изображений. |
| ОК-3 | Знать:  закономерности и этапы исторического процесса, основные исторические факты, даты, события и имена исторических деятелей России; основные события и процессы отечественной истории в контексте мировой истории  Уметь:  критически воспринимать, анализировать и оценивать историческую информацию, факторы и механизмы исторических изменений  Владеть: навыками анализа причинно-следственных связей в развитии российского государства и общества; места человека в историческом процессе и политической организации общества; навыками уважительного и бережного отношения к историческому наследию и культурным традициям. | Проанализировать историю проблематики реконструкции изображений. |
| ОК-5 | Знать:  основные закономерности взаимодействия человека и общества, специфику профессиональной деятельности; основы социологии, структуру общества и социальных институтов; основные этические понятия, историю этических учений, современное положение в сфере этического знания; основные понятия культурологии, типологию культур. Уметь:  определять место и роль профессии в социальной сфере, взаимосвязь с другими профессиями; создавать и поддерживать высокую мотивацию к выполнению профессиональной деятельности; ориентироваться в этической проблематике; выявлять основные черты и особенности культурно-исторических ценностей.  Владеть: методами выявления мотивов социального поведения; технологиями анализа и прогноза социокультурных процессов для решения практических профессиональных проблем. | Проанализировать современную актуальность проблемы реконструкции изображений. |
| ОК-9 | Знать:  основные средства и методы физического воспитания Уметь:  выбирать и применять методы и средства физической культуры для совершенствования основных физических качеств Владеть: навыками использования методов и средств физической культуры для обеспечения полноценной социальной и профессиональной деятельности. | Распределить время на выполнение плана по изучению реконструкции изображений, разработку программного обеспечения для подготовки данных для изучения и исследование эффективности методов распознавания на них. |
| ОПК-6 | Знать: правила поиска и содержание основных нормативно-правовых документов регулирующих работу в области обеспечения информационной безопасности информационных систем Уметь:  разрабатывать локальные и объектовые нормативно-правовые документы для обеспечения нормативно-правового сопровождения работ по обеспечению информационной безопасности на предприятии. Владеть: навыками систематизации и выбора необходимой нормативно-правовой информации согласно поставленным задачам в области обеспечения информационной безопасности автоматизированных систем. | Проанализировать нормативно-правовую базу, касающуюся реконструкции изображений. |
| ПК-1 | способностью осуществлять поиск, изучение, обобщение и систематизацию научно-технической информации, нормативных и методических материалов в сфере профессиональной деятельности, в том числе на иностранном языке | Изучить обоснования различных методов реконструкции изображений в различных источниках |
| ПК-16 | способностью участвовать в проведении экспериментально-исследовательских работ при аттестации автоматизированных систем с учетом нормативных документов по защите информации | Исследовать эффективность различных методов реконструкции изображений. |
| ПК-7 | способностью разрабатывать научно-техническую документацию, готовить научно-технические отчеты, обзоры, публикации по результатам выполненных работ | Собрать результаты и реализовать отчёт. |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Дата выдачи задания: | 01.09.2021 г. | | | | |
| Срок предоставления на кафедру отчета о практике: | | | | 28.12.2021 г. | |
| Руководитель практики от | | |  | |  |
| университета, д.ф.-м.н., профессор кафедры ГИиИБ | | |  | | Мясников В.В. |
|  | | | (подпись) | |  |
|  | | |  | |  |
| Задание принял к исполнению | | |  | |  |
| студент группы № 6512-100503D | |  | | | А.Н. Коновалов |
|  | | (подпись) | | |  |

**Рабочий график (план) проведения практики**

|  |  |  |
| --- | --- | --- |
| Дата (период) | Содержание задания | Результаты практики |
| 01.09.2021 – 01.11.2021 | Изучить теоретические материалы и историю развития, касающиеся тематики реконструкции изображений. Проанализировать современную актуальность и нормативно-правовую базу, касающуюся данной темы. Распределить время на выполнение плана по изучению методов восстановления исходных данных в изображениях. | Были изучены соответствующие теоретические материалы, история развития, нормативно-правовая база, сделан вывод об актуальности проблемы реконструкции изображений. Был изучен прогресс исследований в направлении реконструкции изображений. Распределено время на выполнение поставленных задач. |
| 02.11.2021 – 21.11.2021 | Поиск готового программного обеспечения для реконструкции изображений, выбор нескольких решений из найденных. | Было найдено готовое программное обеспечения для реконструкции изображений, выбрано нескольких решений из найденных. |
| 22.11.2021 –  15.12.2021 | Написание собственного программного обеспечения для подготовки базы данных изображений с вырезанными различными способами фрагментами и анализа качества реконструкции | Написано собственное программное обеспечение для подготовки базы данных изображений с вырезанными различными способами фрагментами и анализа качества реконструкции |
| 15.12.2021 –  28.12.2021 | Реконструирование изображений из базы данных различными способами, анализ качества проведенной реконструкции, создание отчета по проделанной работе | Разными способами реконструированы изображения из базы данных, проанализировано качество результатов, создан отчет по проделанной работе |

|  |  |  |
| --- | --- | --- |
| Руководитель практики от |  |  |
| университета, д.ф.-м.н., профессор кафедры ГИиИБ |  | Мясников  В. В. |
|  | (подпись) |  |

О Т Ч Е Т

о выполнении индивидуального задания

по научно-исследовательской работе

ВВЕДЕНИЕ

При прохождении практики по научно-исследовательской работе, руководителем были поставлены следующие задачи:

* Изучить теоретические материалы, касающиеся тематики реконструкции изображений;
* Проанализировать современную актуальность проблемы реконструкции изображений;
* Проанализировать историю проблематики реконструкции изображений;
* Распределить время на выполнение плана по изучению реконструкции изображений, поиск программного обеспечения по методам реконструкции изображений, разработку программного обеспечения для подготовки данных этим методам и исследование эффективности этих методов;
* Проанализировать нормативно-правовую базу, касающуюся реконструкции изображений;
* Найти программное обеспечение, реализующее несколько методов реконструкции изображений и при необходимости адаптировать его;
* Разработать программное обеспечение для подготовки данных выбранным программным решениям.
* Исследовать эффективность программного обеспечения на сгенерированных изображениях;
* Собрать результаты и реализовать отчёт.

Задания необходимо было выполнять последовательно в течение всего времени практики, предоставляя руководителю промежуточные отчеты.

ВЫПОЛНЕНИЕ ЗАДАНИЯ

Для выполнения поставленных руководителем задач было распределено время и были изучены теоретические материалы, касающиеся реконструкции изображений. В частности, были изучены материалы из [4]. На основании материалов [5] был оценён прогресс в области исследований по данной теме и история развития. Актуальность выбранной темы заключается в том, что реконструкция изображения позволяет восстанавливать утерянную информацию. Нормативно-правовая база, касающаяся данной темы заложена в № 152-ФЗ «О персональных данных» и в № 149-ФЗ «Об информации, информационных технологиях и о защите информации», так как информация, подлежащая восстановлению, может касаться данных законов. Определены цель и задачи исследования.

Целью и задачей исследования стало проведение анализа эффективности трех методов реконструкции изображения на соответствующих образцах.

В качестве объекта исследования выбраны методы реконструирования изображений на основе Contextual Residual Aggregation (CRA), Deep Fusion Network (DFN) и Hypergraph Image Inpainting (HII).

Код CRA: <https://github.com/Atlas200dk/sample-imageinpainting-HiFill>

Заранее обученная модель для СRA взята отсюда (places2):

<https://github.com/Atlas200dk/sample-imageinpainting-HiFill/tree/master/GPU_CPU/pb>

Код DFN: <https://github.com/hughplay/DFNet>

Заранее обученная модель для DFN взята отсюда (places2):

<https://drive.google.com/drive/folders/1lKJg__prvJTOdgmg9ZDF9II8B1C3YSkN>

Код HII: <https://github.com/GouravWadhwa/Hypergraphs-Image-Inpainting>

Заранее обученная модель для HII взята отсюда (places2):

<https://drive.google.com/drive/folders/1dk1zSm1FxZVaafOtvoud8aAdZ6Ubs4oU>

В качестве исходных изображений для создания в них вырезов и последующей реконструкции взят первые 1000 изображений отсюда:

<http://data.csail.mit.edu/places/places365/val_256.tar>

Предметом исследования стали изображения с вырезанными фрагментами, эффективность восстановления фрагментов, реконструкированные изображения.

Темой исследования стало: «Разработка и исследование методов реконструкции изображений и следов использования методов реконструкции».

Структура исследования представляет собой следующие этапы:

1. Описание объекта исследования.
2. Реализация программного обеспечения для подготовки данных.
3. Анализ эффективности методов восстановления.

В [4] описано, что постановка задачи реконструкции цифрового изображения в наиболее простой постановке может быть представлена следующим образом. Пусть функция яркости изображения

(1)

задана на множестве отсчетов , где пара , определяет вертикальный и горизонтальный размеры изображения, , а множество - множество целых чисел до N. Значениями функции яркости выступают величины из множества V, в качестве которого на практике обычно выступают множества или , где K – число «каналов» изображения (обычное цветное изображения имеет три спектральных канала, соответствующих различным длинам волн), а B – число требуемых для представления яркости отсчета битов. Область задания изображения – множество Ω – в задаче реконструкции изображения (inpainting) предполагается состоящей из двух непересекающихся областей: , где определяет множество отсчетов изображения с известными значениями функции яркости (1), а , напротив, с неизвестными. Таким образом, в простейшем варианте постановки задачи реконструкции без обучения необходимо доопределить значения функции (1) на множестве (- известные величины функции яркости):

В случае реконструкции с обучением, наряду в самим реконструируемым изображением в постановке задачи могут фигурировать другие изображения того же типа (например, изображения лиц или изображения зданий), заданные обычно на всем множестве Ω. При этом в наиболее общей постановке для различных изображений множества Ω задания функции яркости могут быть различными.

Согласно [4] все методы решения задачи реконструкции могут быть разделены на классические, то есть использующие математические модели, и использующие машинное обучение. В [5] описано широкое многообразие методов из второй категории, основанных на различных методах машинного обучения, включая нейронные сети, например, CNN, DCNN, PEN, GAN и т. д.

В данной работе будет рассмотрена эффективность трех методов, основанных на машинном обучении: Contextual Residual Aggregation (CRA), Deep Fusion Network (DFN) и Hypergraph Image Inpainting (HII).

Для реализации CRA согласно [1] используется нейронная сеть, причем изображение сначала делается более низкого разрешения, потом происходит восстановление фрагмента и затем изображение возвращается в исходное расширение, в результате изображение становится размытым. Далее происходит генерация компонент в области высоких частот специальным образом, формирующиеся исходя из соседних областей изображения, которым задаются определенные веса. Наконец, сгенерированные компоненты добавляются к увеличенному размытому изображению для получения более четкого результата. Поскольку сеть работает только с изображениями с низким разрешением, стоимость памяти и вычислительного времени значительно снижается. Более того, поскольку модель можно обучать с изображениями с низким разрешением, необходимость в наборах данных для обучения с высоким разрешением отпадает. Для улучшения качества восстановления и скорости вычислений в описании в дополнение могут применяться некоторые техники, например, конфигурации в виде тонких и глубоких слоев, распределение оценок внимания, многомерное распределение внимания и легковесные стробированные свертки (LWGC). Метод позволяет восстанавливать изображения размером до 8K с удовлетворительным качеством.

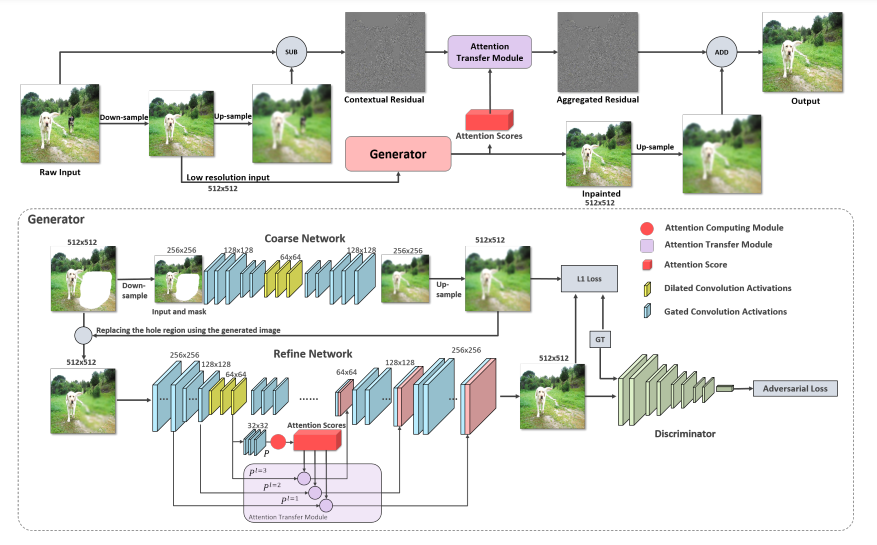


Рисунок 1 – Общий механизм CRA (в верхней части) и генератора (нижняя часть)

На рисунке 1 показан общий конвейер предлагаемого механизма CRA, в котором генератор - единственный обучаемый компонент в схеме. Учитывая высокое разрешение входного изображения, сначала понижается размер изображения до 512 × 512, а затем происходит его повышение до получения размытого изображения того же размера, что и исходное. Высота и ширина изображения не обязательно должны быть равны, но должны быть кратны 512.

В модуле вычисления внимания (Attention Computing Module) оценки внимания рассчитываются на основе близости к региону из высокоуровневой карты признаков (обозначенной буквой *P* на рисунке 1). *P* делится на фрагменты, и ACM вычисляет косинусное сходство между фрагментами внутри и снаружи отсутствующих областей:

где является i-м фрагментом, извлеченным из *P* снаружи маски, является j-м фрагментом, извлеченным из *P* внутри маски. Затем softmax-функция применяется к оценкам сходства, чтобы получить оценки внимания для каждого патча:

где *N* - количество фрагментов за пределами вырезанного региона. Размер каждого региона составляет 3 × 3, а *P* - 32 × 32, таким образом, может быть извлечено всего 1024 фрагмента. По всей матрице 1024 × 1024 сохраняются оценки близости между любой возможной парой фрагментов, хотя только часть из них полезна.

В модуле передачи внимания (Attention Tranfer Module), после получения оценок внимания из *P*, соответствующие вырезанные области в картах признаков нижнего уровня () могут быть заполнены контекстными фрагментами, взвешенными по оценкам внимания:

где l ∈ 1,2,3 - номер слоя, – i-й фрагмент, извлеченный из вне областей, охваченных маской, а – j-й фрагмент, который должен быть заполнен внутри замаскированных областей. N указывает количество контекстных фрагментов (фона). После расчета всех фрагментов внутри вырезанной области получается заполненная карта признаков . Поскольку размер карт признаков зависит от слоя, размер фрагментов должен соответственно меняться. Предполагая, что размер карты признаков равен 1282, а оценка внимания рассчитывается на основе 322 фрагментов, становится ясно что размеры фрагментов должны быть больше или равны (128/32)2 = 42, чтобы можно было охватить все пиксели. Если размеры фрагментов больше 4 × 4, то определенные пиксели перекрываются, что нормально, поскольку следующие уровни сети научатся адаптироваться.

Целью остаточного агрегирования (Residual Aggregation) является вычисление остатков для вырезанной области, чтобы можно было восстановить точные детали отсутствующего содержимого. Остатки для недостающего содержимого можно рассчитать путем агрегирования взвешенных контекстных остатков, полученных на предыдущих шагах:

где *R* - остаточное изображение, - это i-й фрагмент, извлеченный из контекстного остаточного изображения за пределами маски, а - j-й фрагмент, который должен быть заполнен внутри маски. Размеры фрагмента должны быть откалиброваны, чтобы точно покрывать все пиксели без перекрытия и гарантировать, что заполненные остатки согласованы с окружающими областями. После получения агрегированного остаточного изображения происходит его добавление к размытому изображению генератора с повышенной размером и получается более четкий результат.

На рисунке 1 также показана архитектура сети генератора. Используется двухступенчатая сетевая архитектура, где “грубая” (Coarse) сеть заполняет примерным образом отсутствующее содержимое, а “точная” (Refine) сеть генерирует уточнение результатов. Генератор берет изображение и двоичную маску, указывающую области вырезанных фрагментов, в качестве входных данных и строит законченное изображение. Ожидается, что размеры входа и выхода будут 512 × 512. Для увеличения поля восприятия и сокращения вычислений входные данные подвергаются понижению размера до 256 × 256 перед сверткой в ​​грубой сети, в отличие от уточненной сети, которая работает над размером 512 × 512. Прогноз грубой сети наивно смешивается с входным изображением путем замены вырезанной области в последнем на область первого в качестве входных данных для уточняющей сети. Сеть Refine вычисляет оценки контекстного внимания с помощью высокоуровневой карты признаков и выполняет передачу внимания на нескольких картах признаков более низкого уровня, таким образом, удаленная контекстная информация может быть заимствована на нескольких уровнях абстракции. Также применяются расширенные свертки (dilated convolutions) как в “грубой”, так и в “точной” сетях, чтобы еще больше увеличить размер полей восприятия.

В [2] декларируется проблема того, что восстановление изображения с помощью глубокого обучения обычно негармонично соединяет восстановленный фрагмент с существующим контентом, особенно в приграничной зоне. В данной статье эта проблема рассматривается с новой точки зрения создания плавного перехода и предлагается решение DFN. Вводится специальный “блок слияния” (fusion block), который создает гибкую карту альфа-композиции для объединения известных и неизвестных регионов (рисунок 3). Блок слияния не только обеспечивает плавное слияние восстановленного и существующего контента, но также предоставляет карту внимания, чтобы сосредоточить внимание сети на неизвестные пиксели. Таким образом, он создает мост для структурной и текстурной информации, так что информация может естественным образом распространяться из известного региона.

Блоки слияния встраиваются в несколько уровней декодера сети U-Net, что показано на рисунке 2. В сочетании с регулируемым механизмом ограничения потерь на каждом слое, достигается более точная структурная информация. Метод является высокопроизводительным, а также обеспечивает гармоничный фактурный переход, фактурную детализацию и смысловую структурную согласованность.

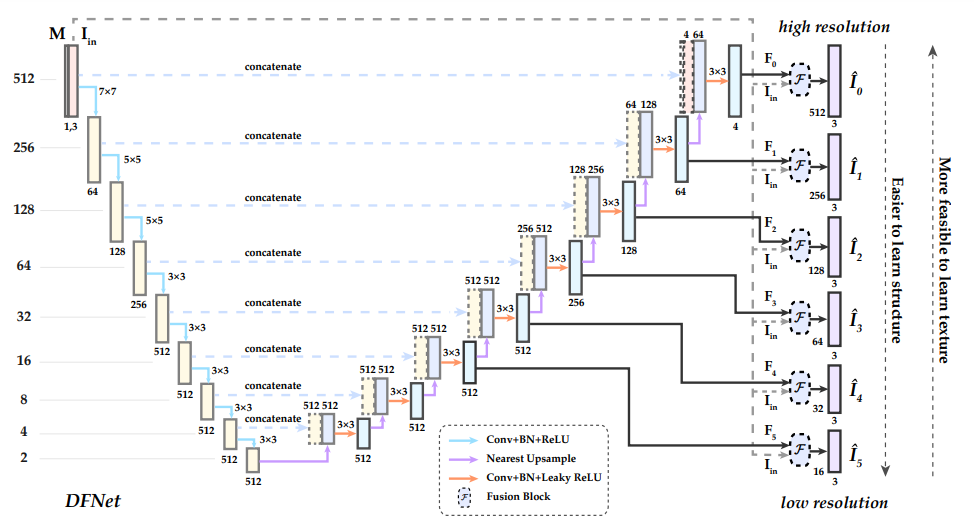


Рисунок 2 – общая структура DFNet.

Сеть U-Net, согласно [6] содержит сверточную и разверточную части. На каждом шаге количество каналов признаков удваивается. Сверточная часть похожа на обычную свёрточную сеть, она содержит два подряд свёрточных слоя 3×3, после которых идет слой ReLU и пулинг с функцией максимума 2×2 с шагом 2. Каждый шаг разверточной части содержит слой, обратный пулингу, который расширяет карту признаков, после которого следует свертка 2×2, которая уменьшает количество каналов признаков. После идет конкатенация с соответствующим образом обрезанной картой признаков из сжимающего пути и две свертки 3×3, после каждой из которой идет ReLU. Обрезка нужна из-за того, что мы теряем пограничные пиксели в каждой свёртке. На последнем слое свертка 1×1 используется для приведения каждого 64-компонентного вектора признаков до требуемого количества классов.

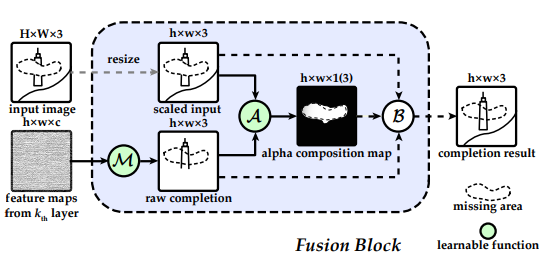


Рисунок 3 – Блок слияния получает сырой вариант заполнения отсутствующего региона из карт признаков на выходе обучаемой функции M, а также строит карту альфа-композиции с помощью функции A. Далее происходит объединение полученного варианта заполнения с масштабированным входным изображением с помощью функции смешивания B.

В [3], описывающем HII, предлагается при разработке метода реконструкции опираться на гиперграфы. Большинство существующих подходов реконструкции используют механизм внимания для изучения глобального контекста изображения. Этот механизм внимания семантически правдоподобный, но генерирует расплывчатые результаты из-за невозможности зафиксировать глобальный контекст. В данной работе описано использование свертки гиперграфов на пространственных объектах для использования в обучении сложных взаимоотношения между данными. Для этого вводится обучаемый механизм для подключения узлов, использующий гиперребра для свертки гиперграфа, показанный на рисунке 4.

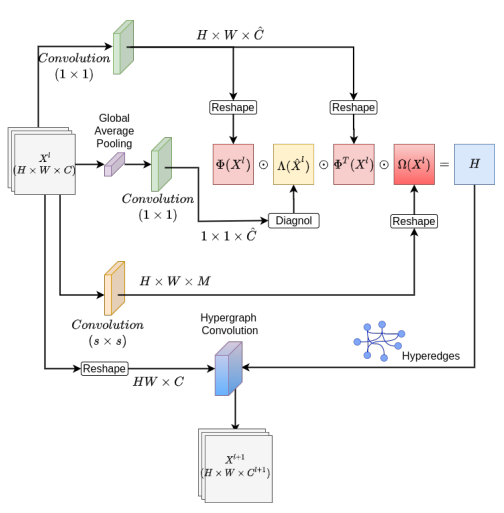


Рисунок 4 – Обучаемый слой гиперграфа. Сначала вычисляется матрица инцидентности H с использованием информации, собранной из входных характеристик, а затем вычисляется свертка гиперграфа с использованием вычисленной матрицы инцидентности.

Структура гиперграфов используется во многих задачах компьютерного зрения для моделирования ограничений высокого порядка на данные, которые не могут быть размещены в традиционной структуре графа. В отличие от парных связей в графах, гиперграф содержит гиперребро, которое может соединять две или более вершины. Гиперграф определяется как , где – набор вершин, – набор гиперребер, . Матрица инцидентов определяется как:

Для данного гиперграфа *G* степень вершины и степень гиперребра также определяются как и .

Матрица инцидентности *H*, степень вершины *D* и степень гиперребра *B* используются для вычисления нормализованной матрицы Лапласа гиперграфа , причем . Это симметричная положительно полуопределенная матрица и ее собственное разложение можно использовать для получения полного набора ортонормированных собственных векторов и диагональной матрицы , содержащая соответствующие неотрицательные собственные значения. Мы можем определить преобразование Фурье гиперграфа , которое преобразует сигнал в спектральную область, определенную на базисе *Φ*, также известный как базис Фурье. С обобщением теоремы о свертке на структурированное пространство гиперграфов, свертка по сигналу может определяться как:

Где является функцией коэффициентов Фурье. Однако для вычисления свертки сигнала *x* потребуется вычислить собственные векторы матрицы Лапласа.

Далее предлагается использовать параметризацию усеченными многочленами Чебышева до K-го порядка, таким образом определяя сверточную операцию над сигналом гиперграфа как:

Где вектор коэффициентов полинома Чебышева, а полином Чебышева. В этой формуле исключена операция вычисления собственных векторов матрицы Лапласа. Она может быть еще больше упрощена ограничением К = 1. Оно аппроксимируется ≈ 2 из-за масштабируемости нейронных сетей, следовательно, сверточная операция над сигналом гиперграфа принимает вид:

где *θ* - единственный коэффициент Чебышева, оставшийся после взятия K = 1 многочленов Чебышева.

Для полученного сигнала в виде гиперграфа , где это размерность вектора признаков на входе слоя под номером *l*, операция свертки в многослойной сверточной сети гиперграфов обобщается как:

Где - обучаемый параметр, а *σ* - нелинейная функция активации. Матрица инцидентности *H* кодирует структуру гиперграфа, которая в дальнейшем используется для распространения информации между узлами гиперграфа. Следовательно, можно легко увидеть, что более качественные соединения гиперребер могут привести к лучшему обмену информацией между узлами, дополнительно улучшая законченное изображение.

Чтобы преодолеть ограниченное восприимчивое поле архитектур CNN, недавние исследования преобразуют карты пространственных характеристик в структуру на основе графа и выполняют свертку графа, чтобы зафиксировать глобальную взаимосвязь между данными. Легко заметить, что простые графы являются частным случаем гиперграфов, где каждое гиперребро соединяет только два узла. Простой граф может легко представить парные отношения между данными, но уже труднее с помощью него представить сложные отношения между пространственными характеристиками изображения, поэтому предлагается использовать гиперграфы вместо графов. Чтобы преобразовать пространственные признаки в графоподобную структуру, каждый пространственный объект рассматривается как узел, имеющий вектор признаков размера *c*, .

В недавних исследованиях для задачи визуальной классификации матрица инцидентности *H* формируется с использованием евклидова расстояния между элементами изображений. Чтобы лучше фиксировать внутрипространственную структуру изображения, предлагается улучшенная матрица инцидентности, которая может научиться фиксировать долгосрочные внутрипространственные зависимости. Вместо евклидова расстояния между пространственными объектами используется взаимная корреляция пространственных объектов для вычисления вклада каждого узла в каждое гиперребро. Матрица инцидентности *H* содержит информацию о вкладе каждого узла в каждое гиперребро и выражается как:

Где - это линейное встраивание входных функций, за которым следует нелинейная функция активации (функция ReLU), – размерность вектора признаков после линейного встраивания, - диагональная матрица, которая помогает лучше изучить метрику расстояния между узлами для матрицы инцидентности *H* и, наконец, – величина, помогающая определить вклад каждого узла для каждого гиперребра, m - количество гиперребер в гиперграфе. , и - матрицы, зависящие от данных. реализуется сверткой 1 × 1 со входными признаками, реализуется путем объединения глобального среднего значения по каналам, за которым следует свертка 1 × 1 и используется Ω (X) для захвата глобальной взаимосвязи функций для создания лучших гиперребер (реализовано с использованием фильтра s × s, s = 7).

Матрица инцидентности вычисляется как показано на рисунке 4, и ее можно сформулировать как:

Где - карта признаков, созданная после глобального объединения входных признаков, , , - обучаемые параметры для линейного вложения. Чтобы избежать отрицательных значений в матрице инцидентности *H*, которые могут привести к мнимым значениям в матрицах степеней, в матрице инцидентности используются модули. Тогда слой свертки гиперграфа на пространственных признаках формулируется как:

Где – обучаемые параметры, – нелинейная функция активации, – входные признаки, – выходные признаки.

На рисунке 5 показано использование в HII двухступенчатой сетевой архитектуры от грубого к точному. “Грубая” сеть примерно заполняет недостающую область, которая наивно смешивается с входным изображением, а затем уточняющая сеть предсказывает более точные результаты с резкими краями. В уточненной сети используется слой гиперграфа с высокоуровневыми картами функций, чтобы увеличить восприимчивое поле сети и получить отдаленную глобальную информацию об изображении. Используются расширенные свертки для грубой и точной сетей, чтобы еще больше увнличить восприимчивое поле. В дополнение использовались стробированные свертки, чтобы улучшить нашу производительность на определенной нерегулярной маске. Также удаляются все слои пакетной нормализации и заменяются все слои свертки закрытой сверткой, с помощью которой обеспечивается локальная согласованность в завершенном изображении.

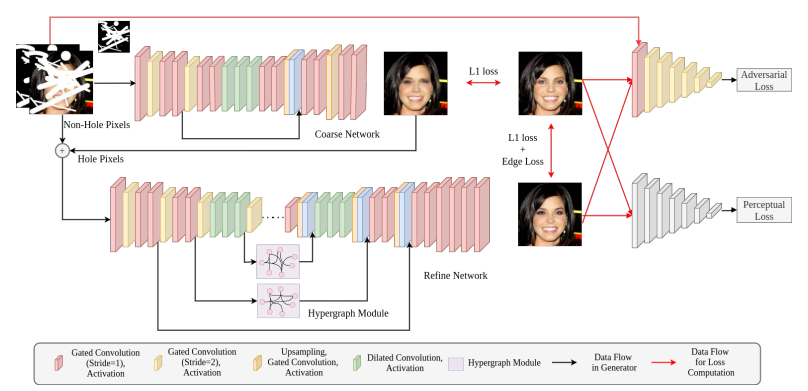


Рисунок 5 – “Грубая” сеть генерирует примерное заполнение недостающих областей. Позже “точная” сеть на основе свертки гиперграфа генерирует окончательно завершенное изображение высокого качества.

Структура программы, разработанной с целью подготовки данных для проверки эффективности методов реконструкции, состоит из ряда модулей. Прежде всего можно выделить модуль, содержащий функции генерации масок по размерам входного изображения и параметру, от которого зависит доля пространства, занимаемой вырезанным фрагментом в изображении. Маски вырезов имеют следующие формы: прямоугольники в случайном месте, решетка в виде чередующихся полос, всплески пикселей на основе шума соль и перец, вырез отсечения некой доли изображения, вырез рамки вокруг изображения. Модуль генерации масок используется модулем генерации изображений с вырезанными фрагментами и масками этих вырезов в БД в формате системы папок. Модуль позволяет делать эти операции сразу для всей выборки целых изображений. Также имеются вспомогательные скрипты для генерации системы папок, переименования выборки целых изображений для удобства и единый модуль инициализации всей БД с нуля, агрегирующий эти скрипты.

Был разработан модуль подсчета метрик качества SSIM и PSNR реконструированных изображений в сравнении с исходными для сравнения эффективности изучаемых методов. Модуль выводит результаты для хранения в отдельные файлы, а затем из этих файлов данные считываются другим скриптом, который показывает результаты в виде графиков в удобном виде. Для метода HII был создан отдельный специальный скрипт, который нужно запускать перед вышеописанными модулями с целью привести результирующие изображения к более удобному для анализа виду (выделить последнее изображение из 4 склеенных в каждом случае реконструкции).

Структура генерируемой базы данных изображений с вырезанными фрагментами и масками этих вырезов для этих изображений выглядит следующим образом:

* в папке images лежат папка с выборкой исходных изображений для выреза фрагментов;
* в папке fragmented лежат выборки изображений с вырезами;
* в папке masks лежат выборки с масками вырезов, каждая из которых соответствует определенному изображению в выборке;
* в папке results лежат выборки с изображениями восстановленными разными методами.

В каждой папке выборка в соответствии с используемой в работе одна - places2. В каждой папке выборки имеются подпапки, последняя часть имени которых - название маски, которая применялась к выборке изображении:

* frame - вырез в виде рамки, отсекающей информацию по краям изображений;
* grid - вырез в виде решетки, то есть чередующихся полос;
* noise - вырез, образованный из шума соль и перец;
* rectangle - вырез в виде прямоугольников в случайных местах изображений;
* half - вырез в виде отсечения некой доли от изображения.

В каждых из папок, включающих вышеназванные названия в своих именах, содержатся папки с названиями в виде чисел. Число означает параметр, от которого зависит занимаемая маской доля в изображении, то есть ее размер. В данной работе для примера генерируются изображения со следующие значения размеров масок вырезов: frame 0.1, grid 0.1, noise 0.5, half 0.5, rectangle 0.1, 0.2, 0.5.

Для вышеописанных параметров масок были получены следующие результаты для каждого из трех методов реконструкции. В целях наглядности графики вычислены для первых ста изображений из исследуемой выборки. Также для наглядности приведены примеры вариантов реконструкции определенного изображения из выборки.

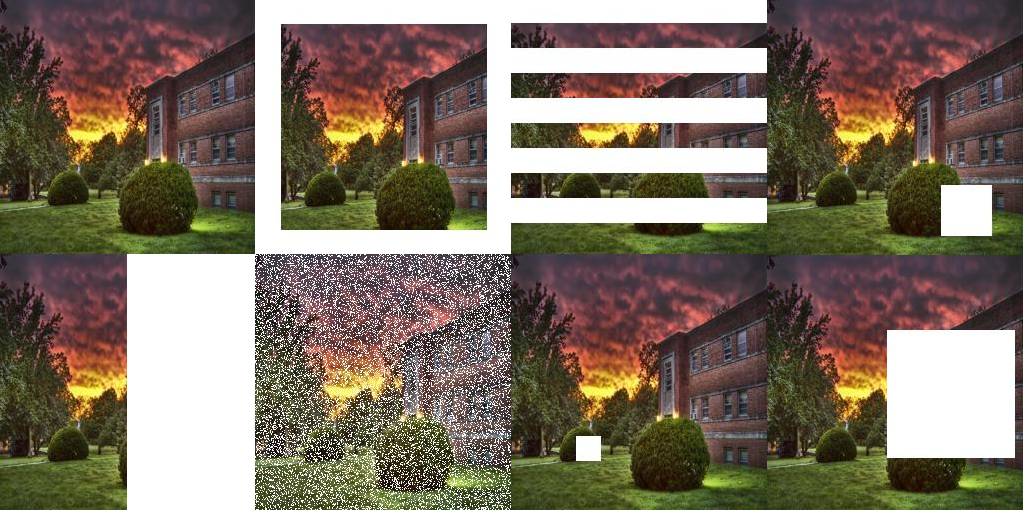


Рисунок 6 – исходное изображение и результаты вырезания областей разными способами



Рисунок 7 – результаты для случая frame с параметром 0.1

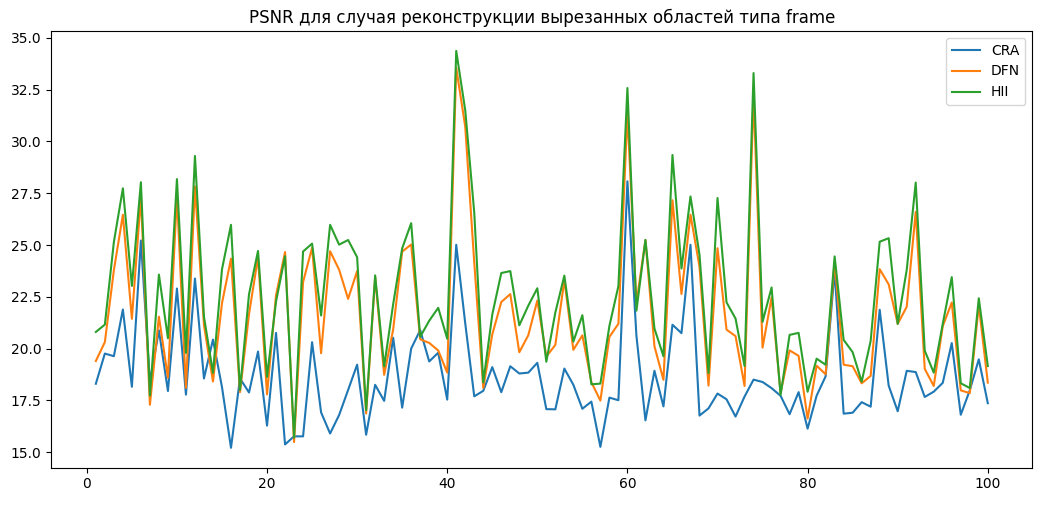


Рисунок 8 – результаты для случая frame с параметром 0.1



Рисунок 9 – пример реконструирования для случая frame с параметром 0.1. Здесь и далее слева направо: CRA, DFN, HII

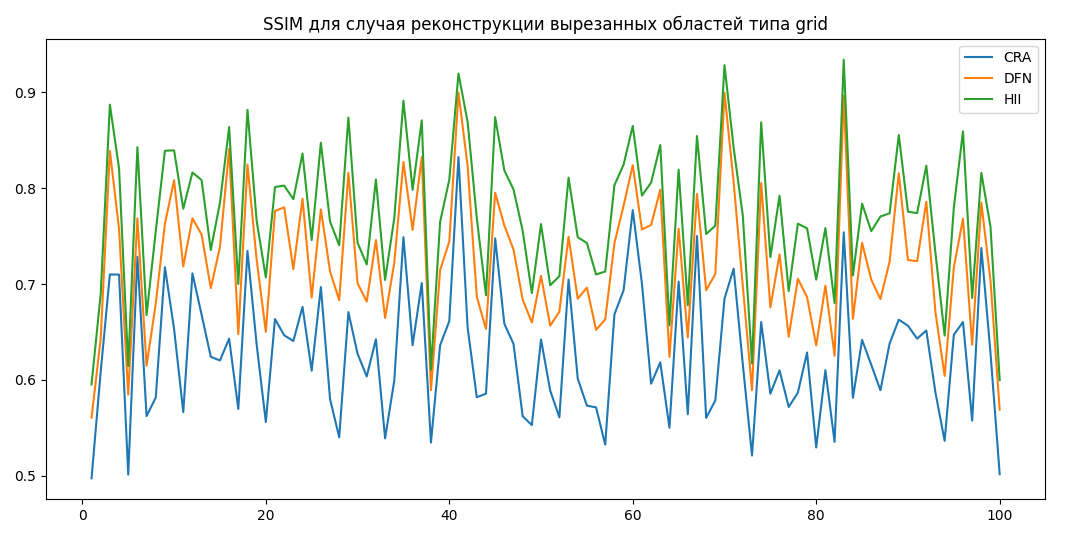


Рисунок 10 – результаты для случая grid с параметром 0.1

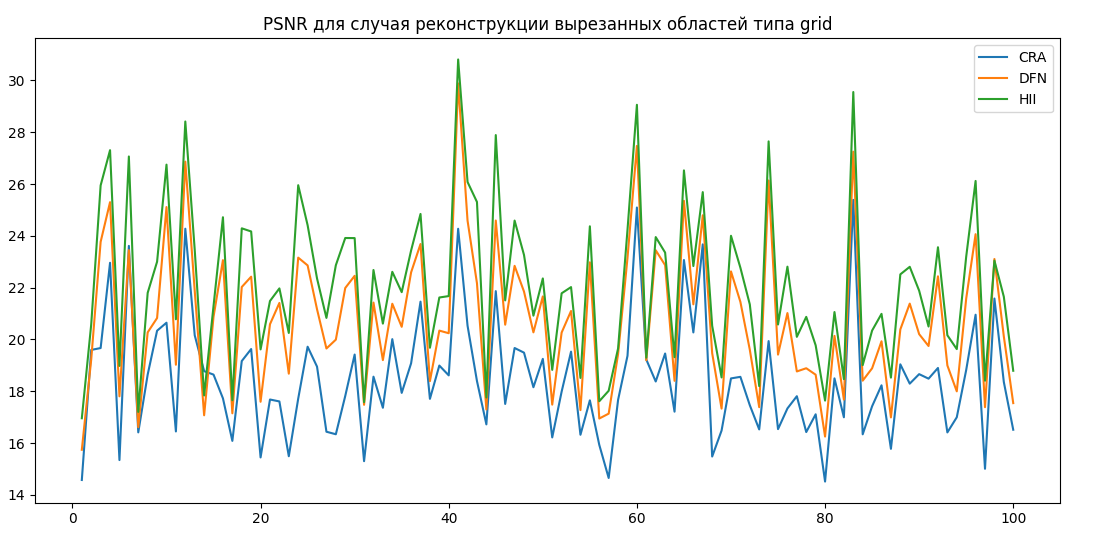


Рисунок 11 – результаты для случая grid с параметром 0.1



Рисунок 12 – пример реконструирования для случая grid с параметром 0.1

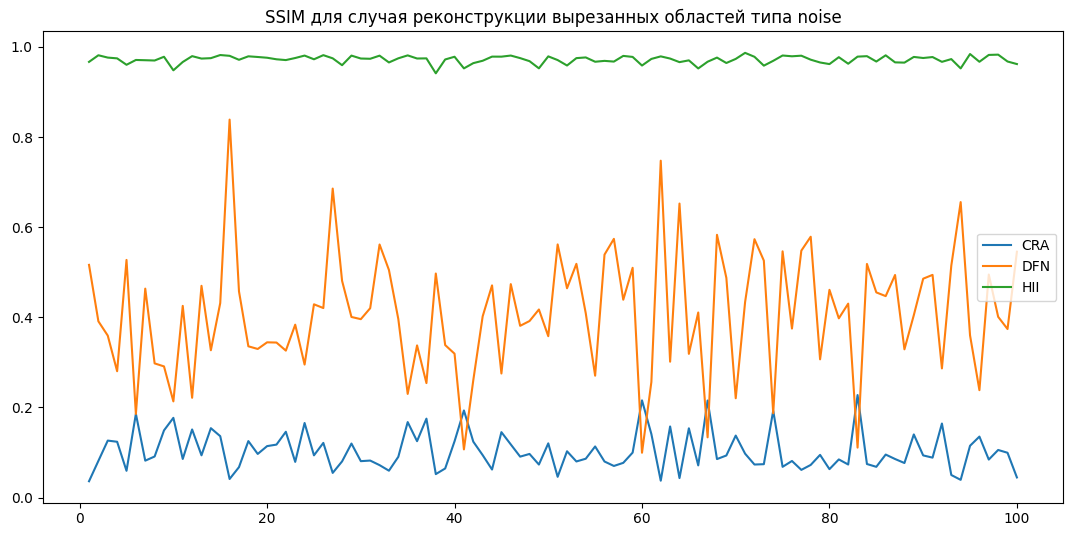


Рисунок 13 – результаты для случая noise с параметром 0.5

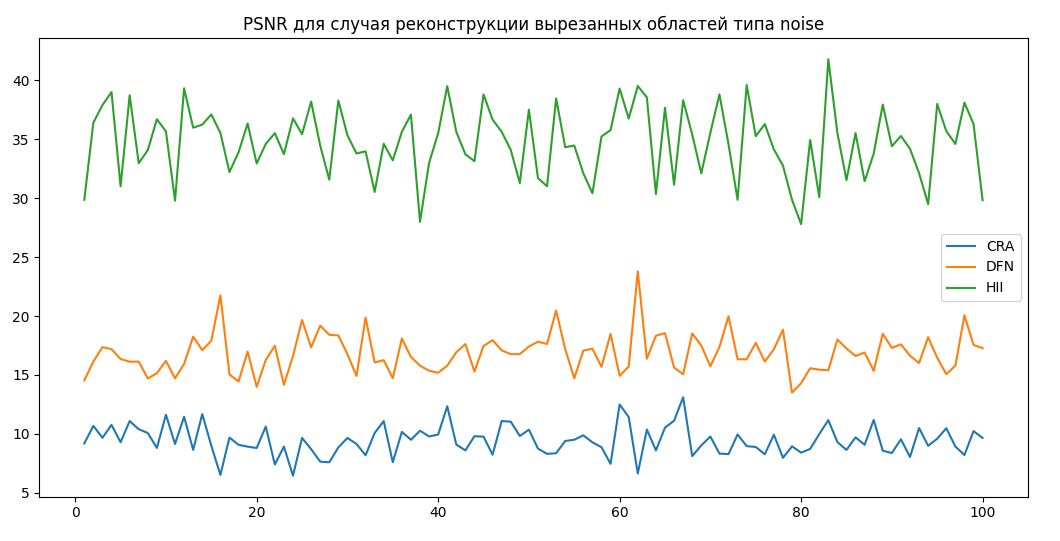


Рисунок 14 – результаты для случая noise с параметром 0.5



Рисунок 15 – пример реконструирования для случая noise с параметром 0.5

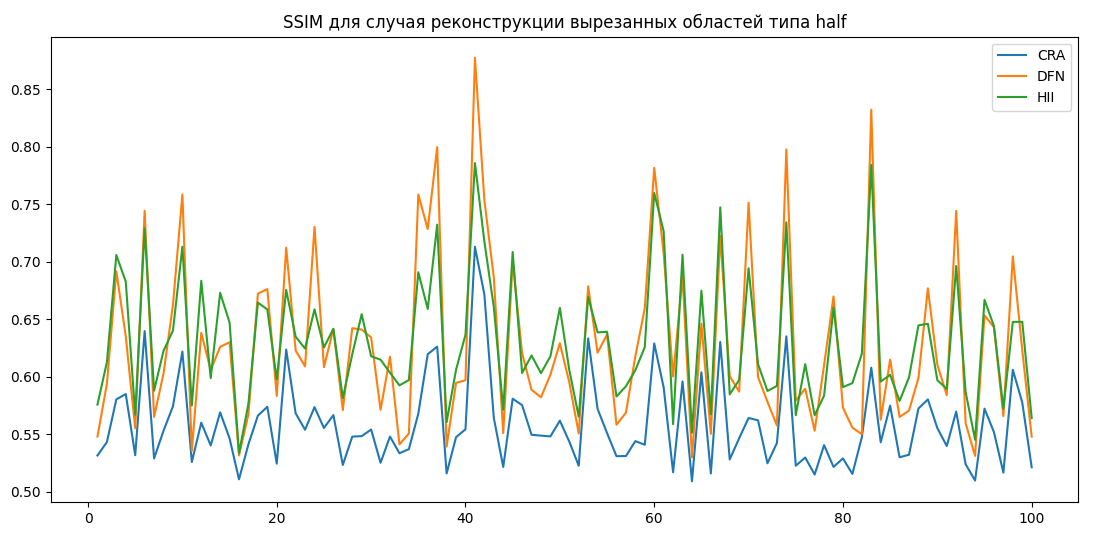


Рисунок 16 – результаты для случая half с параметром 0.5



Рисунок 17 – результаты для случая half с параметром 0.5

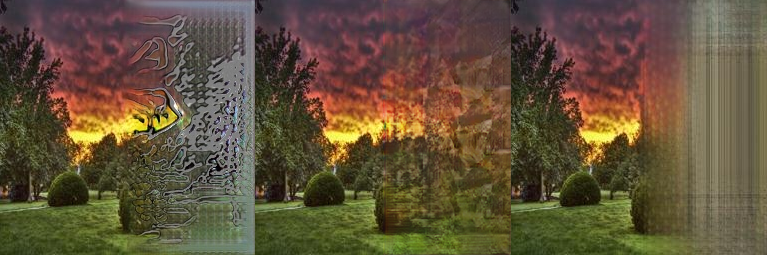


Рисунок 18 – пример реконструирования для случая half с параметром 0.5

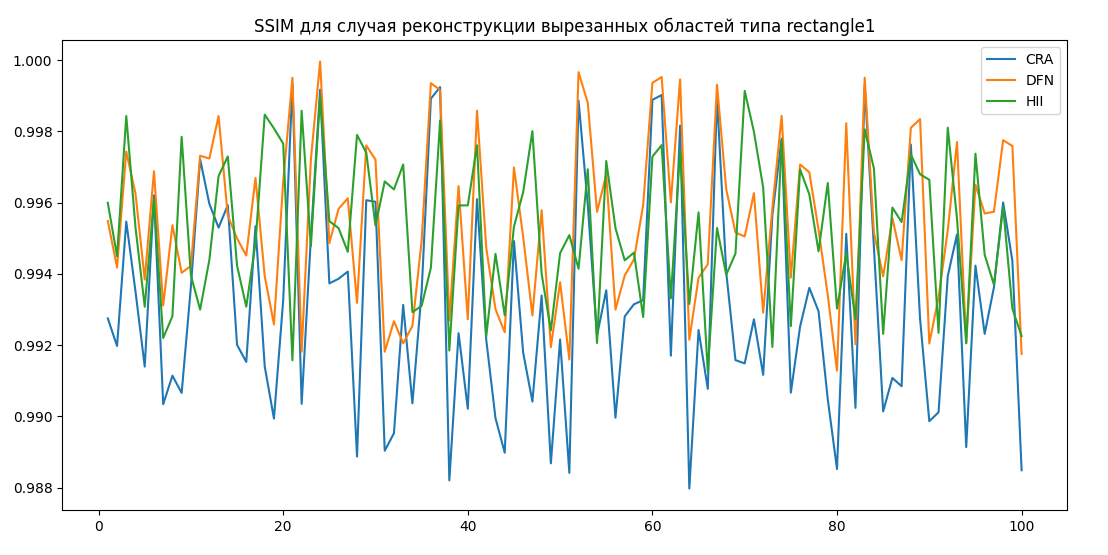


Рисунок 19 – результаты для случая rectangle с параметром 0.1

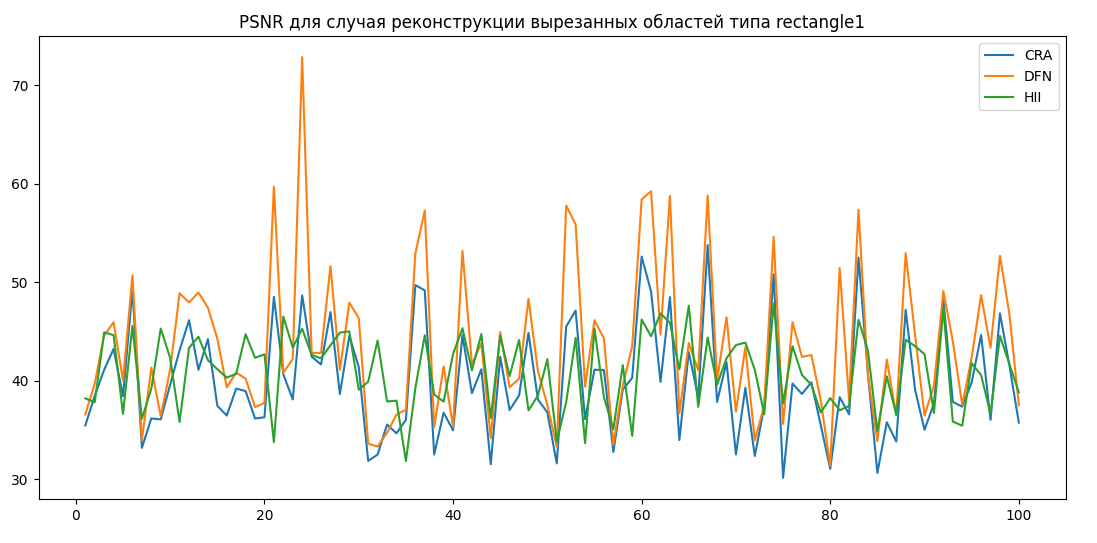


Рисунок 20 – результаты для случая rectangle с параметром 0.1



Рисунок 21 – пример реконструирования для случая rectangle с параметром 0.1

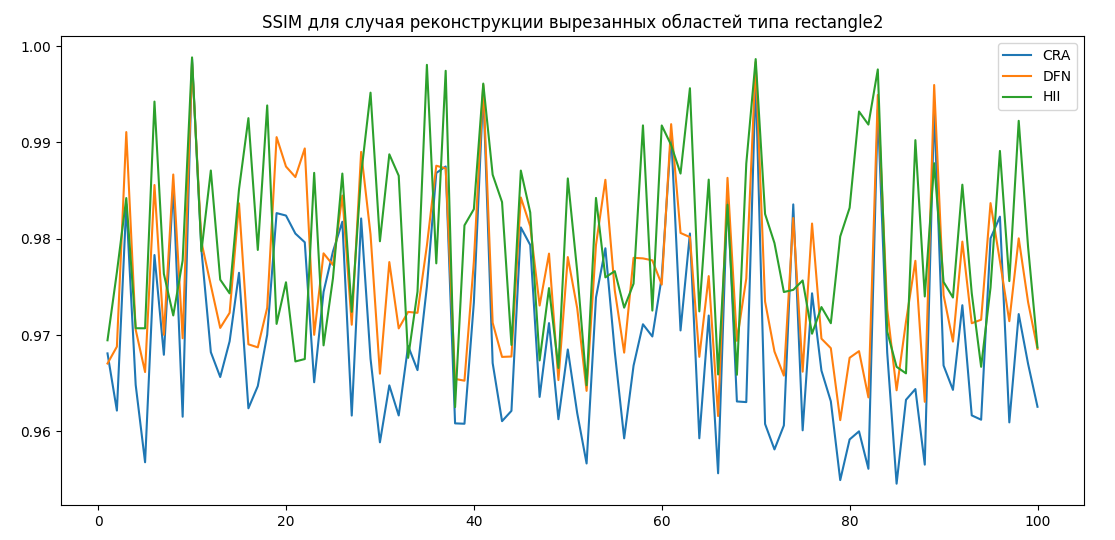


Рисунок 22 – результаты для случая rectangle с параметром 0.2

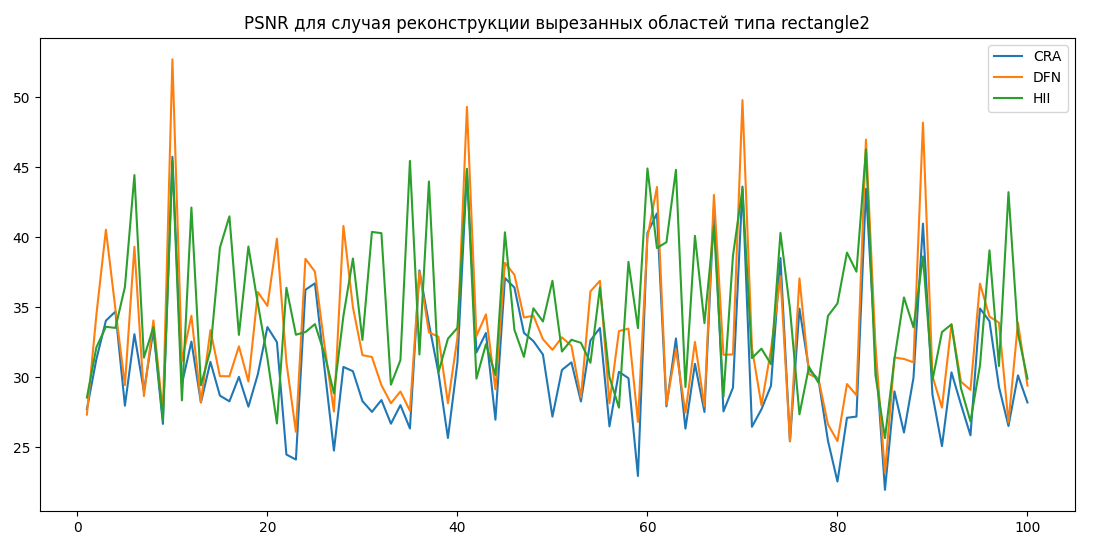


Рисунок 23 – результаты для случая rectangle с параметром 0.2



Рисунок 24 – пример реконструирования для случая rectangle с параметром 0.2

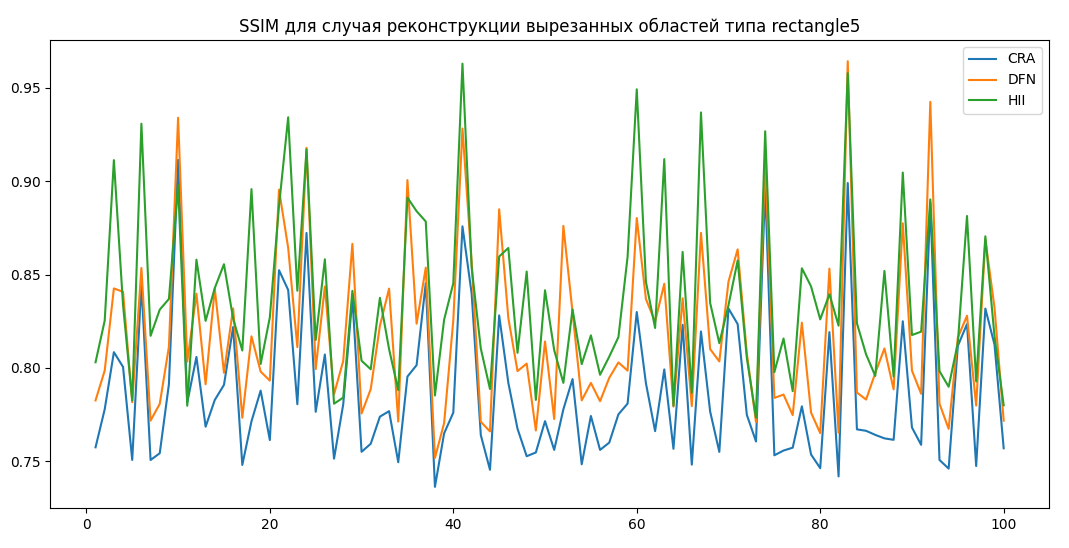


Рисунок 25 – результаты для случая rectangle с параметром 0.5

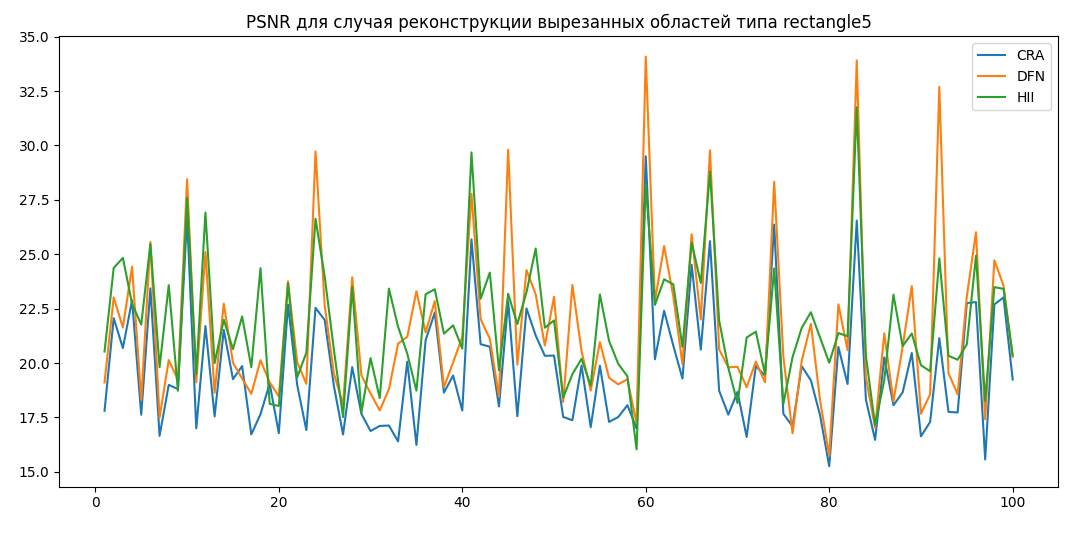


Рисунок 26 – результаты для случая rectangle с параметром 0.5



Рисунок 27 – пример реконструирования для случая rectangle с параметром 0.5

В целом, исходя из полученных результатов можно отметить, что HII-метод показал себя с наилучшей стороны в большинстве вариантов вырезания фрагментов в изображениях. Особенно заметна разница в случае восстановления зашумленного изобюражения. В некоторых вариантах визуальнее чуть лучше работает DFN. CRA же в большинстве случаев оказался менее оптимальным решением по сравнению с двумя этими методами, а в некоторых случаях и вовсе не справлялся.

ЗАКЛЮЧЕНИЕ

В течение практики по научно-исследовательской работе успешно выполнены поставленные задачи: были изучены темы в области методов восстановления исходных данных в изображениях и проанализирована эффективность трех из них. Для этой задачи было реализовано ПО для генерации подходящих образцов изображений. Далее была выполнена проверка эффективности выбранных методов на созданных образцах запуском каждого из них с помощью заранее обученных моделей. По итогам работы был сделан вывод о том, что наиболее эффективным методом реконструкции изображений из трех выбранных является HII.

За время прохождения практики освоены необходимые компетенции, в частности овладел основами государственной политики РФ в сфере информационной безопасности в условиях современного информационного общества, знаниями об устройстве методов восстановления исходных данных в изображениях, опытом разработки программного обеспечения, касающейся данной тематики.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Yi, Zili and Tang, Qiang and Azizi, Shekoofeh and Jang, Daesik and Xu, Zhan. Contextual Residual Aggregation for Ultra High-Resolution Image Inpainting. – IEEE/CVF, 2020.

2. Hong, Xin and Xiong, Pengfei and Ji, Renhe and Fan, Haoqiang. Deep Fusion Network for Image Completion. – ACM, 2019.

3. Wadhwa, Gourav and Dhall, Abhinav and Murala, Subrahmanyam and Tariq, Usman. Hyperrealistic Image Inpainting With Hypergraphs. – IEEE/CVF, 2021.

4. В.В. Мясников, А.В. Веричев. Реконструкция изображения как задача квадратичного программирования. – ИТНТ-2020, Самара, 2020.

5. David Josue Barrientos Rojas, Bruno Jose Torres Fernandes, Sergio Murilo Maciel Fernandes. A Review on Image Inpainting Techniques and Datasets. – IEEE, 2020.

6. Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. – University of Freiburg, Germany, 2015.

ПРИЛОЖЕНИЕ

Код программы

Файл с параметрами работы (config.py)‎

masked\_color = 0

non\_masked\_color = 255

dataset = "places2\\"

methods\_inpaint = ["CRA", "DFN", "HII"]

abs\_path = "C:\\Users\\maste\\Documents\\materials\\" # Указать здесь папку, где будет находится БД изображений

abs\_path\_masks = abs\_path + "masks\\"

abs\_path\_images = abs\_path + "images\\"

abs\_path\_fragmented = abs\_path + "fragmented\\"

abs\_path\_result = abs\_path + "results\\"

path\_mask\_frame = "mask\_frame\\"

path\_mask\_noise = "mask\_noise\\"

path\_mask\_rectangle = "mask\_rectangle\\"

path\_mask\_grid = "mask\_grid\\"

path\_mask\_half = "mask\_half\\"

path\_image\_frame = "image\_frame\\"

path\_image\_noise = "image\_noise\\"

path\_image\_rectangle = "image\_rectangle\\"

path\_image\_grid = "image\_grid\\"

path\_image\_half = "image\_half\\"

path\_recovery\_frame = "recovery\_frame\\"

path\_recovery\_noise = "recovery\_noise\\"

path\_recovery\_rectangle = "recovery\_rectangle\\"

path\_recovery\_grid = "recovery\_grid\\"

path\_recovery\_half = "recovery\_half\\"

params\_small = [0.05, 0.1, 0.2]

params = [0.1, 0.2, 0.3, 0.4, 0.5]

params\_img = [str(x) + "\_" for x in params]

params\_small\_img = [str(x) + "\_" for x in params\_small]

Файл, агрегирующий в себе все функции, необходимые для инциализации БД изображений (generate\_database.py)‎

from config import \*

from image\_gen import image\_gen\_main, get\_list\_images

from initial\_data\_dirs import initial\_dirs

from format\_name import rename\_images\_to\_format

from shutil import copyfile

def copy\_files(source\_dir, destination\_dir):

sources = get\_list\_images(source\_dir)

for i in range(len(sources)):

print("Copying " + str(i) + " image...")

copyfile(source\_dir + sources[i], destination\_dir + sources[i])

if \_\_name\_\_ == '\_\_main\_\_':

source\_dir = "C:\\Users\\maste\\Documents\\images\\" # Указать здесь исходную папку с изображениями в любом месте

initial\_dirs()

copy\_files(source\_dir, abs\_path\_images + dataset)

rename\_images\_to\_format()

image\_gen\_main()

Файл создания необходимых папок для хранения изображений (initial\_data\_dirs.py)‎

import os

from config import \*

def initial\_dirs():

os.mkdir(abs\_path)

os.mkdir(abs\_path\_masks)

os.mkdir(abs\_path\_images)

os.mkdir(abs\_path\_fragmented)

os.mkdir(abs\_path\_result)

os.mkdir(abs\_path\_masks + dataset)

os.mkdir(abs\_path\_images + dataset)

os.mkdir(abs\_path\_fragmented + dataset)

os.mkdir(abs\_path\_result + dataset)

path\_masks = [path\_mask\_frame, path\_mask\_noise, path\_mask\_rectangle, path\_mask\_grid, path\_mask\_half]

path\_fragmented = [path\_image\_frame, path\_image\_noise, path\_image\_rectangle, path\_image\_grid, path\_image\_half]

path\_results = [path\_recovery\_frame, path\_recovery\_noise, path\_recovery\_rectangle, path\_recovery\_grid, path\_recovery\_half]

for p in path\_masks:

os.mkdir(abs\_path\_masks + dataset + p)

for p in path\_fragmented:

os.mkdir(abs\_path\_fragmented + dataset + p)

for i in range(len(methods\_inpaint)):

os.mkdir(abs\_path\_result + dataset + methods\_inpaint[i] + "\\")

for p in path\_results:

for i in range(len(methods\_inpaint)):

os.mkdir(abs\_path\_result + dataset + methods\_inpaint[i] + "\\" + p)

for p in [path\_recovery\_noise, path\_recovery\_rectangle, path\_recovery\_half]:

for i in range(len(methods\_inpaint)):

for v in params\_img:

os.mkdir(abs\_path\_result + dataset + methods\_inpaint[i] + "\\" + p + v)

for p in [path\_recovery\_frame, path\_recovery\_grid]:

for i in range(len(methods\_inpaint)):

for v in params\_small\_img:

os.mkdir(abs\_path\_result + dataset + methods\_inpaint[i] + "\\" + p + v)

if \_\_name\_\_ == '\_\_main\_\_':

initial\_dirs()

Файл для переименования исходных изображений в удобные имена для обработки (format\_name.py)

import os

from config import \*

from image\_gen import get\_list\_images

def rename\_images\_to\_format():

path = abs\_path\_images + dataset

list\_images = get\_list\_images(path)

for i in range(0, len(list\_images)):

os.rename(path + list\_images[i], path + str(i + 1) + '.jpg')

print(str(i + 1) + ' renamed')

if \_\_name\_\_ == '\_\_main\_\_':

rename\_images\_to\_format()

Файл функций генерации масок вырезов для каждого отдельного изображения (mask\_func.py)

import numpy as np

from PIL import Image

import random

from config import \*

fill = [masked\_color, masked\_color, masked\_color]

background = [non\_masked\_color, non\_masked\_color, non\_masked\_color]

def mask\_frame(shape, param):

h1, h2 = shape[0], shape[1]

mask = np.zeros(shape)

mask.fill(non\_masked\_color)

h1\_value = int(h1 \* param)

h2\_value = int(h2 \* param)

mask[0:h1\_value, :, :] = fill

mask[h1 – h1\_value:h1, :, :] = fill

mask[:, 0:h2\_value, :] = fill

mask[:, h2 – h2\_value: h2, :] = fill

return mask

def mask\_rectangle(shape, param):

h1, h2 = shape[0], shape[1]

mask = np.zeros(shape)

mask.fill(non\_masked\_color)

h1\_value = int(h1 \* param)

h2\_value = int(h2 \* param)

coord1 = random.randint(0, h1 – h1\_value)

coord2 = random.randint(0, h2 – h2\_value)

mask[coord1:coord1 + h1\_value, coord2:coord2 + h2\_value, :] = fill

return mask

def mask\_half(shape, param):

h1, h2 = shape[0], shape[1]

mask = np.zeros(shape)

mask.fill(non\_masked\_color)

mask[:, h2 – int(h2 \* param):h2, :] = fill

return mask

def mask\_grid(shape, param): #, regime=2):

h1, h2 = shape[0], shape[1]

h1\_value = int(h1 \* param)

# h2\_value = int(h2 \* value)

mask = np.zeros(shape)

mask.fill(non\_masked\_color)

# if regime == 1:

h\_value = int(h1 \* param)

start = 0

end = h\_value

while end <= h1:

mask[start:end, :, :] = fill

start += int(2 \* h1\_value)

end += int(2 \* h1\_value)

# else:

# h\_value = int(h2 \* value)

# start = 0

# end = h\_value

# while end <= h2:

# mask[:, start:end, :] = fill

# start += int(2 \* h2\_value)

# end += int(2 \* h2\_value)

return mask

def mask\_noise(shape, param):

mask = np.zeros(shape)

mask.fill(non\_masked\_color)

noise = np.random.choice((0, 1, 2), size=(shape[0], shape[1]), p=[param, (1 – param) / 2., (1 – param) / 2.])

mask[noise == 1] = fill

return mask

Файл генерации изображений с вырезанными разными способами фрагментами (image\_gen.py)

from mask\_func import \*

import os

from os import path

def get\_list\_images(path):

lst = []

for root, s ar, files in os.walk(path):

for filename in files:

lst.append(filename)

return lst

def apply\_mask(img, mask):

new\_img = img.copy()

new\_img[mask == masked\_color] = 255

return new\_img

def img\_open(path):

img = Image.open(path)

img = img.convert(‘RGB’)

img = np.asarray(img)

return img

def img\_save(img, path):

result = Image.fromarray(img.astype(np.uint8))

result.save(path)

def apply\_mask\_to\_list\_images(func\_mask, dir\_read, dir\_save\_image, dir\_save\_mask, params, params\_name):

lst = get\_list\_images(dir\_read)

for img\_name in lst:

print(‘\n’ + img\_name + ‘ func: ‘ + func\_mask.\_\_name\_\_)

img = img\_open(dir\_read + img\_name)

for i in range(len(params)):

print(params[i], end=’ ‘)

mask = func\_mask(img.shape, params[i])

dir = dir\_save\_mask + params\_name[i] + “\\”

if not path.exists(dir):

os.mkdir(dir)

img\_save(mask, dir + img\_name)

result = apply\_mask(img, mask)

dir = dir\_save\_image + params\_name[i] + “\\”

if not path.exists(dir):

os.mkdir(dir)

img\_save(result, dir + img\_name)

def image\_gen\_main():

path\_masks = [path\_mask\_half, path\_mask\_rectangle, path\_mask\_noise]

path\_masks\_small\_param = [path\_mask\_frame, path\_mask\_grid]

path\_images = [path\_image\_half, path\_image\_rectangle, path\_image\_noise]

path\_images\_small\_param = [path\_image\_frame, path\_image\_grid]

func\_masks = [mask\_half, mask\_rectangle, mask\_noise]

func\_masks\_small\_param = [mask\_frame, mask\_grid]

dir\_read = abs\_path\_images + dataset

dir\_save\_mask = abs\_path\_masks + dataset

dir\_save\_image = abs\_path\_fragmented + dataset

for i in range(len(func\_masks)):

apply\_mask\_to\_list\_images(func\_masks[i], dir\_read,

dir\_save\_image + path\_images[i],

dir\_save\_mask + path\_masks[i],

params, params\_img)

for i in range(len(func\_masks\_small\_param)):

apply\_mask\_to\_list\_images(func\_masks\_small\_param[i], dir\_read,

dir\_save\_image + path\_images\_small\_param[i],

dir\_save\_mask + path\_masks\_small\_param[i],

params\_small, params\_small\_img)

if \_\_name\_\_ == ‘\_\_main\_\_’:

image\_gen\_main()

Дополнительный файл для приведения выходных изображений, полученных методом HII к формату, удобному для анализа качества (HII\_format\_results.py)

import cv2

from config import \*

from image\_gen import get\_list\_images

# отсекает в выводе HII финальное решение (четвертое изображение из склеенных)

def extract\_final\_images\_from\_HII(path):

images = get\_list\_images(path)

for name in images:

img = cv2.imread(path + name)

h1, h2 = img.shape[0], img.shape[1]

img = img[:, h2 - int(h2 \* 0.25):h2, :]

cv2.imwrite(path + name, img)

if \_\_name\_\_ == '\_\_main\_\_':

path\_hii\_frame = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_frame + '0.1\_\\'

path\_hii\_grid = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_grid + '0.1\_\\'

path\_hii\_noise = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_noise + '0.5\_\\'

path\_hii\_half = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_half + '0.5\_\\'

path\_hii\_rectangle1 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.1\_\\'

path\_hii\_rectangle2 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.2\_\\'

path\_hii\_rectangle5 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.5\_\\'

extract\_final\_images\_from\_HII(path\_hii\_frame)

print('frame formated')

extract\_final\_images\_from\_HII(path\_hii\_grid)

print('grid formated')

extract\_final\_images\_from\_HII(path\_hii\_noise)

print('noise formated')

extract\_final\_images\_from\_HII(path\_hii\_half)

print('half formated')

extract\_final\_images\_from\_HII(path\_hii\_rectangle1)

print('rectangle 0.1 formated')

extract\_final\_images\_from\_HII(path\_hii\_rectangle2)

print('rectangle 0.2 formated')

extract\_final\_images\_from\_HII(path\_hii\_rectangle5)

print('rectangle 0.5 formated')

Файл, анализирующий SSIM и PSNR исходных изображений и изображений с восстановленными фрагментами с последующей записью выходные файлы (gen\_stat\_data.py)

import cv2

from skimage.metrics import structural\_similarity as compare\_ssim

from config import \*

from image\_gen import get\_list\_images

import pickle

# from skimage import io

def ssim(path1, path2):

img1 = cv2.cvtColor(cv2.imread(path1), cv2.COLOR\_BGR2GRAY)

img2 = cv2.cvtColor(cv2.imread(path2), cv2.COLOR\_BGR2GRAY)

return compare\_ssim(img1, img2, full=False)

def psnr(path1, path2):

img1 = cv2.cvtColor(cv2.imread(path1), cv2.COLOR\_RGB2BGR)

img2 = cv2.cvtColor(cv2.imread(path2), cv2.COLOR\_RGB2BGR)

return cv2.PSNR(img1, img2)

def calc\_metric(path1, path2):

ssim\_values = []

psnr\_values = []

images = get\_list\_images(path1)

images\_masked = get\_list\_images(path2)

for i in range(len(images)):

ssim\_values.append(ssim(path1 + images[i], path2 + images\_masked[i]))

psnr\_values.append(psnr(path1 + images[i], path2 + images\_masked[i]))

return ssim\_values, psnr\_values

def dump\_metric(output\_filename, path\_etalon, path\_three\_method):

with open(output\_filename, 'wb') as f:

for path in path\_three\_method:

print(path)

ssim\_values, psnr\_values = calc\_metric(path\_etalon, path)

pickle.dump(ssim\_values, f)

pickle.dump(psnr\_values, f)

if \_\_name\_\_ == '\_\_main\_\_':

path\_etalon = abs\_path\_images + dataset

path\_cra\_frame = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_frame + '0.1\_\\'

path\_cra\_grid = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_grid + '0.1\_\\'

path\_cra\_noise = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_noise + '0.5\_\\'

path\_cra\_half = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_half + '0.5\_\\'

path\_cra\_rectangle1 = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_rectangle + '0.1\_\\'

path\_cra\_rectangle2 = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_rectangle + '0.2\_\\'

path\_cra\_rectangle5 = abs\_path\_result + dataset + 'CRA\\' + path\_recovery\_rectangle + '0.5\_\\'

path\_dfn\_frame = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_frame + '0.1\_\\' + 'result\\'

path\_dfn\_grid = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_grid + '0.1\_\\' + 'result\\'

path\_dfn\_noise = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_noise + '0.5\_\\' + 'result\\'

path\_dfn\_half = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_half + '0.5\_\\' + 'result\\'

path\_dfn\_rectangle1 = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_rectangle + '0.1\_\\' + 'result\\'

path\_dfn\_rectangle2 = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_rectangle + '0.2\_\\' + 'result\\'

path\_dfn\_rectangle5 = abs\_path\_result + dataset + 'DFN\\' + path\_recovery\_rectangle + '0.5\_\\' + 'result\\'

path\_hii\_frame = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_frame + '0.1\_\\'

path\_hii\_grid = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_grid + '0.1\_\\'

path\_hii\_noise = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_noise + '0.5\_\\'

path\_hii\_half = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_half + '0.5\_\\'

path\_hii\_rectangle1 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.1\_\\'

path\_hii\_rectangle2 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.2\_\\'

path\_hii\_rectangle5 = abs\_path\_result + dataset + 'HII\\' + path\_recovery\_rectangle + '0.5\_\\'

output\_filename\_frame = 'frame.pickle'

output\_filename\_grid = 'grid.pickle'

output\_filename\_noise = 'noise.pickle'

output\_filename\_half = 'half.pickle'

output\_filename\_rectangle1 = 'rectangle1.pickle'

output\_filename\_rectangle2 = 'rectangle2.pickle'

output\_filename\_rectangle5 = 'rectangle5.pickle'

dump\_metric(output\_filename\_frame, path\_etalon, [path\_cra\_frame, path\_dfn\_frame, path\_hii\_frame])

print('frame dumped')

dump\_metric(output\_filename\_grid, path\_etalon, [path\_cra\_grid, path\_dfn\_grid, path\_hii\_grid])

print('grid dumped')

dump\_metric(output\_filename\_noise, path\_etalon, [path\_cra\_noise, path\_dfn\_noise, path\_hii\_noise])

print('noise dumped')

dump\_metric(output\_filename\_half, path\_etalon, [path\_cra\_half, path\_dfn\_half, path\_hii\_half])

print('half dumped')

dump\_metric(output\_filename\_rectangle1, path\_etalon, [path\_cra\_rectangle1, path\_dfn\_rectangle1, path\_hii\_rectangle1])

print('rectangle 0.1 dumped')

dump\_metric(output\_filename\_rectangle2, path\_etalon, [path\_cra\_rectangle2, path\_dfn\_rectangle2, path\_hii\_rectangle2])

print('rectangle 0.2 dumped')

dump\_metric(output\_filename\_rectangle5, path\_etalon, [path\_cra\_rectangle5, path\_dfn\_rectangle5, path\_hii\_rectangle5])

print('rectangle 0.5 dumped')

Файл, читающий данные о SSIM и PSNR исходных изображений и изображений с восстановленными фрагментами из выходных файлов и выводящий это в виде графиков (show\_stat\_data.py)

import matplotlib.pyplot as plt

import pickle

def load\_metric(output\_filename):

with open(output\_filename, 'rb') as f:

ssim1\_values = pickle.load(f)

psnr1\_values = pickle.load(f)

ssim2\_values = pickle.load(f)

psnr2\_values = pickle.load(f)

ssim3\_values = pickle.load(f)

psnr3\_values = pickle.load(f)

return ssim1\_values, psnr1\_values, ssim2\_values, psnr2\_values, ssim3\_values, psnr3\_values

def plot\_result(resuls1, resuls2, resuls3, text):

N = len(resuls1)

N = int(N / 10)

plt.title(text)

plt.plot(list(range(1, N + 1)), resuls1[:N], label='CRA')

plt.plot(list(range(1, N + 1)), resuls2[:N], label='DFN')

plt.plot(list(range(1, N + 1)), resuls3[:N], label='HII')

plt.legend()

plt.show()

def gen\_graphic(output\_filename, maskname):

plot\_result(ssim1\_values, ssim2\_values, ssim3\_values, 'SSIM для случая реконструкции вырезанных областей типа ' + maskname)

plot\_result(psnr1\_values, psnr2\_values, psnr3\_values, 'PSNR для случая реконструкции вырезанных областей типа ' + maskname)

if \_\_name\_\_ == '\_\_main\_\_':

output\_filename\_frame = 'frame.pickle'

output\_filename\_grid = 'grid.pickle'

output\_filename\_noise = 'noise.pickle'

output\_filename\_half = 'half.pickle'

output\_filename\_rectangle1 = 'rectangle1.pickle'

output\_filename\_rectangle2 = 'rectangle2.pickle'

output\_filename\_rectangle5 = 'rectangle5.pickle'

gen\_graphic(output\_filename\_frame, 'frame')

gen\_graphic(output\_filename\_grid, 'grid')

gen\_graphic(output\_filename\_noise, 'noise')

gen\_graphic(output\_filename\_half, 'half')

gen\_graphic(output\_filename\_rectangle1, 'rectangle1')

gen\_graphic(output\_filename\_rectangle2, 'rectangle2')

gen\_graphic(output\_filename\_rectangle5, 'rectangle5')

**ОТЗЫВ О ПРОХОЖДЕНИИ ПРАКТИКИ**

|  |  |
| --- | --- |
| Вид практики | производственная практика |
|  | (учебная, производственная, преддипломная) |

|  |  |
| --- | --- |
| Тип практики | научно-исследовательская работа |
|  | (учебная, производственная, преддипломная) |

Сроки прохождения практики: с 01.09.2021 г. по 28.12.2021 г.

по направлению подготовки 10.05.03 Информационная безопасность автоматизированных систем

(уровень академического специалитета)

направленность (профиль) «Обеспечение информационной безопасности распределенных информационных систем»

студентом группы № 6512-100503DА.Н.Коноваловым

|  |  |  |
| --- | --- | --- |
| № п/п | Критерии оценки | Оценка  (по 5-балльной шкале) |
| 1. | Общая систематичность и ответственность работы в ходе практики |  |
| 2. | Достижение планируемых результатов практики |  |
| 3. | Корректность в сборе, анализе и интерпретации представляемых данных |  |
| 4. | Степень личного участия и самостоятельности практиканта в представляемом отчете о практике |  |
| 5. | Качество оформления отчетной документации |  |
|  | **ИТОГОВАЯ ОЦЕНКА[[1]](#footnote-1)\*** |  |

|  |  |  |
| --- | --- | --- |
| Руководитель практики от |  |  |
| университета, д.ф.-м.н., профессор кафедры ГИиИБ |  | Мясников В.В. |
|  | (подпись) |  |

1. \* Итоговая оценка выставляется как средняя арифметическая оценок по пяти критериям оценки [↑](#footnote-ref-1)