Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ Информатика и системы управления

КАФЕДРА Программное обеспечение ЭВМ и информационные технологии

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К ВЫПУСКНОЙ КВАЛИФИКАЦИОННОЙ РАБОТЕ

НА ТЕМУ:

Метод повышения разрешения черно-белого изображения на основе генеративно-состязательных нейронных сетей

Студент ИУ7-85Б (Группа)	_____ (Подпись, дата)	К.В. Марабян (И.О.Фамилия)
Руководитель ВКР	_____ (Подпись, дата)	Н.В. Новик (И.О.Фамилия)
Консультант	_____ (Подпись, дата)	_____ (И.О.Фамилия)
Консультант	_____ (Подпись, дата)	_____ (И.О.Фамилия)
Нормоконтролер	_____ (Подпись, дата)	Ю.В.Строганов (И.О.Фамилия)

2020 г.

Реферат

Расчетно-пояснительная записка 59 с., 28 рис., 8 табл., 12 источников.

Объектом исследования является черно-белые изображения с низким разрешением. Цель работы – разработка генеративно-состязательной нейросети для увеличения разрешения черно-белого изображения, изменённого методом аугментации.

Для достижения заданной цели были решены следующие задачи:

- проведён анализ предметной области;
- проведён анализ нейронных сетей;
- проведён анализ методов аугментации изображений;
- модифицирована генеративно-состязательная нейронная сеть для повышения разрешения черно-белых изображений;
- разработан программный комплекс, реализующий выбранный метод;
- проанализирована эффективность работы программного продукта.

Для реализации программного продукта была использована генеративно-состязательная нейронная сеть, основанная на функции потерь, которая ближе к перцептивному подобию. В результате работы спроектировано и реализовано программное обеспечение, которое повышает разрешение для любого входного изображения.

Содержание

Глоссарий

В данной работе были использованы следующие термины с соответствующими определениями.

Батч – подмножество объектов, которые обрабатываются нейросетью прежде чем обновить параметры нейронной сети.

Эпоха – период обработки всех изображений из обучающей выборки;

Тензор – объект линейной алгебры, линейно преобразующий элементы одного линейного пространства в элементы другого.

Обозначения и сокращения

SRCNN – Super Resolution Convolutional Neural Network;

SRGAN – Super Resolution Generative Adversarial Network;

MSE – Mean Squared Error;

VDSR – Very Deep Super Resolution;

HR – High Resolution;

SR – Super Resolution;

LR – Low Resolution;

VGG – Visual Geometry Group.

Введение

Задача увеличения разрешения изображения очень актуальна в современном мире. Улучшение качества изображения полученных снимков применяют в медицине при диагностике, в системе правопорядка - для идентификации лиц, в картографии - для анализа аэрофотоснимков и т.д.

Увеличение разрешения требуется для получения качественного изображения высокого разрешения из изображения с малым разрешением. Для решения данной задачи часто применяют методы, относящиеся к классу Single image super-resolution (рус. суперразрешение на основе одного изображения). Однако применение методов интерполяции или наложение фильтров не дают качественного изображения [1].

Существуют различные методы улучшения качества, такие как бикубическая и билинейная интерполяция, глубокая остаточная сеть, оптимизированная для среднеквадратической ошибки, генеративно-состязательные нейронные сети и свёрточные нейронные сети.

Среди используемых методов применяются методы, основанные на нейросетях. Использование искусственных нейронных сетей (ИНС) позволяет получить результаты, которые превосходят стандартные алгоритмы и фильтры[2]. Причина, по которой ИНС до сих пор не сильно распространены в применении – это большие временные затраты на реализацию программного продукта, его обучение и тестирование.

Целью данной работы является создание программного комплекса для повышения разрешения черно-белого изображения на основе генеративно-состязательных нейронных сетей. Для достижения заданной цели необходимо провести анализ предметной области и методов повышения разрешения изображения, модифицировать выбранный метод под поставленные задачи, провести анализ эффективности работы

разработанного программного обеспечения на различных входных изображениях.

1 Аналитический раздел

1.1 Цель и задачи работы

Целью данной работы является создание программного комплекса для увеличения разрешения черно-белого изображения, изменённого методом аугментации (вращение, отражение, сдвиг, масштабирование). Для достижения заданной цели были решены следующие задачи:

- проанализирована предметная область и существующие методы увеличения разрешения черно-белых изображений;
- проанализированы способы малых преобразований в черно-белых изображениях;
- из существующих методов увеличения разрешения выбран метод, наиболее оптимальный для поставленной задачи;
- создано программное обеспечение на основе выбранного метода;
- проанализирована эффективности работы программного продукта.

1.2 Обзор существующих методов увеличения разрешения изображений

Существует много способов улучшения изображения, такие как бикубическая интерполяция, глубокая остаточная сеть, оптимизированная для среднеквадратической ошибки, Super Resolution Generative Adversarial Network (SRGAN), Super Resolution Convolutional Neural Network (SRCNN), Very Deep Super Resolution (VDSR), которые включают в себя шумоподавление, масштабирование изображения и цветовые настройки.

Целью выше перечисленных методов является воссоздание изображения высокого разрешения путём увеличения масштаба

изображения низкого разрешения таким образом, чтобы текстурные детали в восстановленных SR данных не терялись.

Оптимизация алгоритмов SR обычно заключается в минимизации среднеквадратичной ошибки (англ. mean squared error, MSE) между восстановленным изображением HR и оригинальным. Данная операция максимизирует пиковое отношение сигнал-шум (англ. peak signal-to-noise ratio, PSNR), которое является общей мерой, используемой для оценки и сравнения алгоритмов SR. Однако способность MSE (и PSNR) определять перцептивно значимые различия, такие как высокая детализация текстуры, очень ограничена, поскольку они определяются на основе пиксельных различий изображений.

Одним из наиболее часто используемых методов является бикубическая интерполяция, так как не требует больших временных затрат на реализацию. Но использование данного метода приводит к искажению изображения или снижает визуальное качество данных по сравнению с оригинальным изображением. Более сложные методы (SRGAN, SRCNN) используют внутренние сходства изображения или наборы данных изображений с низким разрешением и их аналоги с высоким разрешением для повышения эффективности работы программного продукта.

В последние годы было предложено много методов увеличения разрешения изображений с использованием нейронных сетей, так как они дают хорошее качество оптимизированных изображений [2]. Рассмотрим существующие методы в глубоком обучении (методы машинного обучения, позволяющие обучать нейронную сеть предсказывать результат по набору входных данных):

– SRCNN: SRCNN был первым методом глубокого обучения, который превзошел традиционный (бикубическую интерполяцию). Это сверточная нейронная сеть, состоящая из 3 слоев свертки:

извлечение и представление патчей, нелинейное отображение и реконструкция;

– Very Deep Super Resolution: VDSR использует структуру, подобную SRCNN, но идет глубже, чтобы достигнуть более высокой точности;
– GAN: GAN - это класс алгоритмов искусственного интеллекта, состоящий из двух сетей (генератора и дискриминатора), противопоставляющий одну сеть другой (таким образом, «состязательной»), используемых в бесконтрольном машинном обучении.

В машинном обучении два основных класса моделей - генеративная и дискриминаторная. Дискриминаторная модель — это модель, которая различает два (или более) различных класса данных. Генеративной модели ничего не известно о классах данных. Её цель состоит в том, чтобы генерировать новые данные, соответствующие распределению обучающих данных.

Генератор получает некоторые данные из распределения вероятностей, а дискриминатор определяет, поступает ли на вход изображение из истинного набора обучающих данных или же из сгенерированных данных. Генератор пытается оптимизировать данные, чтобы они соответствовали истинным данным обучения. На рисунке 1.1 показана базовая архитектура GAN.

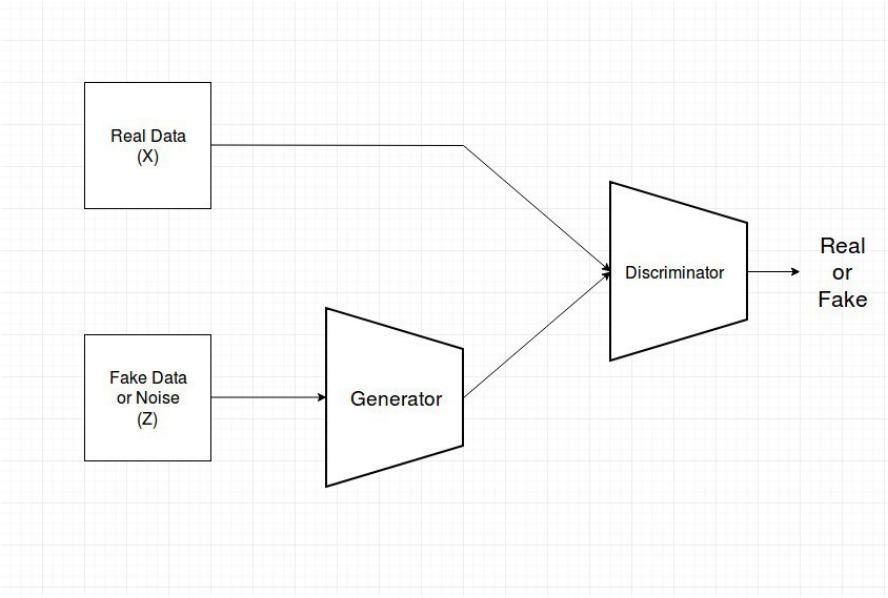

Рисунок 1.1 – Базовая архитектура GAN

Дискриминатор и генератор обучаются одновременно, и когда генератор обучен, он знает достаточно о распределении обучающих выборок, чтобы генерировать новые выборки, которые обладают очень похожими свойствами.

1.2.1 Бикубическая интерполяция

Бикубическая интерполяция — это двумерная система использования кубических сплайнов или другой полиномиальной техники для настройки резкости и увеличения цифровых изображений [3]. Этот метод обычно используется в программах (Photoshop и других) для редактирования компьютерных изображений, ретушерами и редакторами при масштабировании или повторной дискретизации изображения. При интерполяции изображения происходит искажение пикселей от одной сетки к другой.

Существует два общих алгоритма интерполяции, адаптивный и неадаптивный. Адаптивные методы зависят от интерполируемого объекта, тогда как неадаптивные методы обрабатывают пиксели одинаково. В общем случае бикубическая интерполяция может быть выполнена с использованием полиномов Лагранжа, кубических сплайнов или

кубических алгоритмов свертки. Поскольку при увеличении масштаба изображения пиксели добавляются, а не вычитаются, детали могут быть потеряны. Чтобы сохранить резкость и детализацию, для получения наиболее близкого значения интенсивности, каждый пиксель должен быть аппроксимирован окружающими его пикселями. Данное действие похоже на дублирование пикселя, с целью заполнить пространство, созданное в изображении путем увеличения масштаба.

Предположим, что значения функции f и производные от него f_x, f_y и $f_{x,y}$ известны в четырёх углах (0, 0), (0, 1), (1, 0) и (1, 1) единичного квадрата. Затем интерполированную поверхность можно записать в виде:

$$p(x,y)=\sum_{i=0}^{3}\sum_{j=0}^{3} a_{i,j}x^i y^j \tag{1.1}$$

Эту формулу можно представить следующим образом:

1) $f(0, 0) = p(0, 0) = a_{00}$,
2) $f(0, 1) = p(0, 1) = a_{00} + a_{10} + a_{20} + a_{30}$,
3) $f(0, 1) = p(0, 1) = a_{00} + a_{01} + a_{02} + a_{03}$,
4) $f(0, 1) = p(0, 1) = \sum_{i=0}^{3}\sum_{j=0}^{3} a_{i,j}$.

Необходимо определить значения 16 коэффициентов p(x,y). Это основная формула, которая создаёт интерполированную поверхность 2D-изображения. Задача состоит в том, чтобы взять значения в точке p(x, y) на сетке и интерполировать их так, чтобы приблизить значение окружающих её точек.

1.2.2 Метод VDSR

VDSR - это одна из самых быстрых моделей, которая также способна обеспечить высокоточное построение изображений. Этот метод основан на VGG-net и становится эффективным в результате остаточного обучения и отсечение градиента. Сеть VDSR состоит из 20 сверточных слоев [4]. Входное и выходное изображение имеют одинаковый размер. Это достигается путем заполнения нулей в каждой свертке. Ключевым элементом здесь является остаточное обучение, которое применяется путем добавления входного изображения к выходу из последнего сверточного слоя. Таким образом, сеть узнает только разницу между низким и высоким разрешением. Это имеет смысл, потому что оба изображения имеют одинаковые низкие частоты и поэтому не должны рассматриваться в процессе обучения.

Размер фильтра каждой свертки, кроме первой, составляет 3x3x64. Таким образом, воспринимающее поле сети составляет 41x41 пиксель. Каждая свертка, за исключением последней, генерирует 64 карты объектов, некоторые из которых визуализируются на графике выше. Для обучения используется увеличение объема данных с помощью вращения и переворачивания. Для увеличения скорости и уменьшения размера сети обучающие данные разлагаются на патчи размером 41x41. Это также помогает увеличить объем обучающих данных. На рисунке 1.2 показана архитектура VDSR.

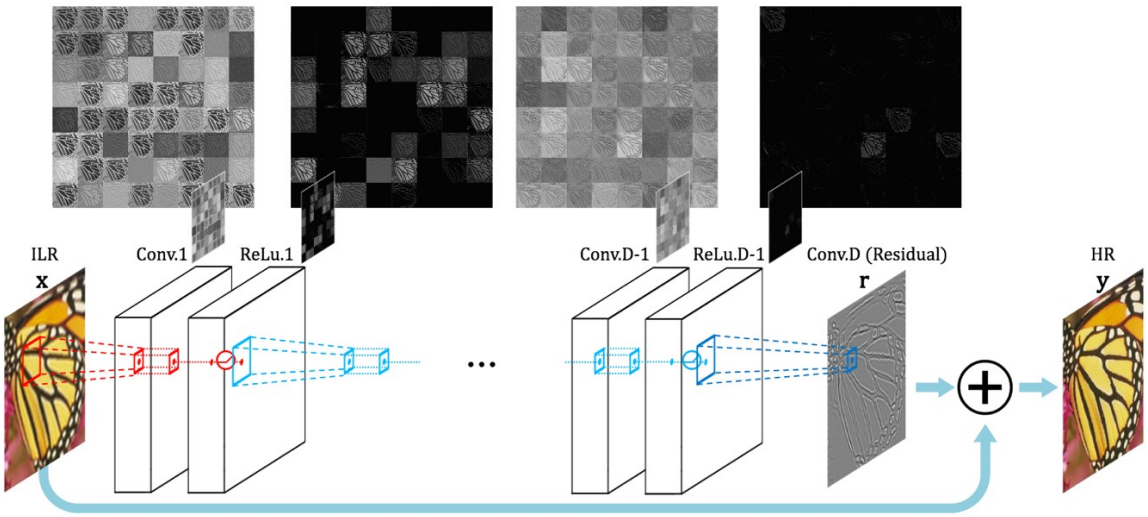

Рисунок 1.2 – Архитектура VDSR

Для того, чтобы сеть сходилась, используется отсечение градиента и L2-регуляризация. Скорость обучения снижается каждые 20 эпох в 0,1 раза, что улучшает производительность. VDSR укладывает 18 сверточных слоев с активацией ReLU, в дополнение к входному сверточному слою, который работает на входном изображении, и выходному сверточному слою, используемому для реконструкции изображения. Рецептивное поле сети увеличивается пропорционально глубине сети, позволяя сети использовать все больше и больше соседних пикселей для более точного предсказания деталей изображения.

Данная сеть состоит из серии блоков, разделяющих одни и те же гипер-параметры (размер фильтра, количество фильтров, количество слоев).

Существуют две эквивалентные конструкции для строительных блоков:

1) 32 ветви 3-слойных сверточных слоев, которые агрегируются (соединяются) путем суммирования в конце блока;

2) сгруппированная операция свертки выполняется на среднем слое блока, в котором входные каналы разбиваются на 32 группы для

отдельных сверток, а затем объединяются в качестве выходов слоя.

В качестве функции потерь используется среднеквадратическая ошибка:

$$MSE = \frac{1}{2}\|y - f(x)\|^2 \tag{1.2}$$

где f (x) - прогнозируемое изображение, а y - целевая истина. Потери усредняются по всему тренировочному набору.

1.2.3 Метод SRCNN

Сверточные нейронные сети (CNN) успешно применяются в областях компьютерного зрения, таких как увеличение разрешения изображений, обнаружение объектов [5], распознавание лиц [6] и обнаружение пешеходов [7]. Несколько факторов имеют центральное значение в этом прогрессе:

1) эффективная реализация обучения на современных мощных графических процессорах;
2) предложение выпрямленного линейного блока (ReLU), который значительно ускоряет конвергенцию, но при этом обеспечивает хорошее качество;
3) легкий доступ к обилию данных для обучения более крупных моделей.

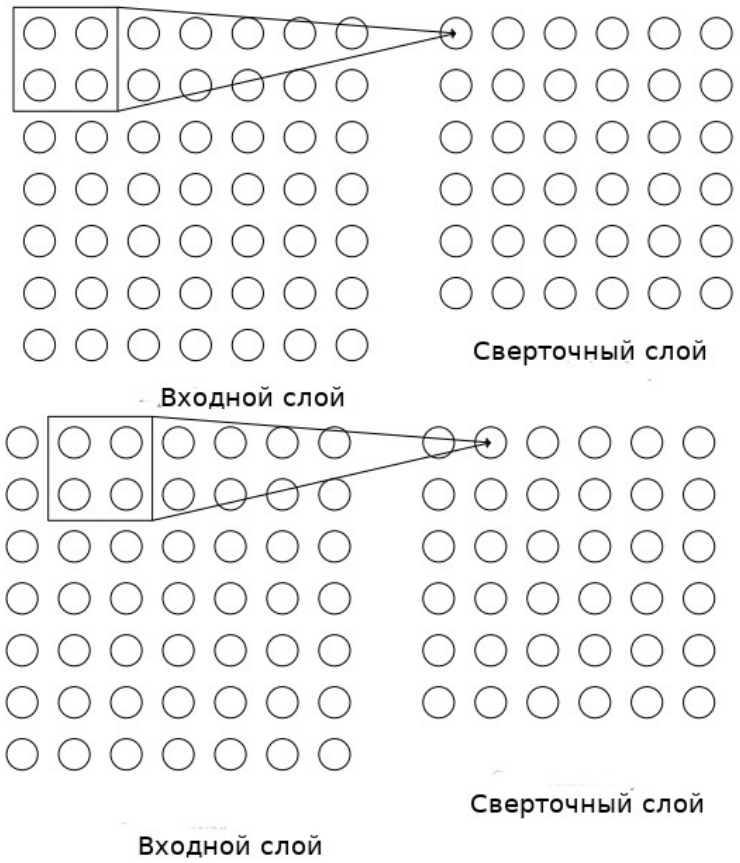

Рисунок 1.3 — Связывание входного слоя с первым скрытым слоем

На рисунке 1.3 продемонстрирована связь входного слоя с первым скрытым слоем.

SRCNN состоит из 3 сверточных слоев с размерами фильтров 9x9 для первого слоя, 1x1 для второго слоя и 3x3 для последнего слоя. Первый слой генерирует 64 карты объектов, второй-32, а последний - выходные данные. Фильтры первого слоя могут быть интерпретированы как детекторы объектов, таких как углы, линии и т. д.

Задача состоит в том, чтобы восстановить из Y изображение F (Y), которое должно быть как можно больше похоже на основное изображение с высоким разрешением - X. Для удобства представления обозначим изображение "низкого разрешения" - Y, хотя оно имеет тот же размер, что и X. Для восстановления изображения F применяются следующие три операции:

1) извлечение и представление патчей (англ. patch extraction and representation): эта операция извлекает (перекрывает) патчи из

изображения Y с низким разрешением и представляет каждый патч в виде вектора высокой размерности. Эти векторы представляют собой набор карт объектов, число которых равно размерности векторов;

2) нелинейное отображение (англ. non-linear mapping): эта операция нелинейно отображает каждый вектор высокой размерности на другой вектор высокой размерности. Каждый отображенный вектор концептуально является представлением патча высокого разрешения. Эти векторы содержат еще один набор карт объектов;
3) реконструкция (англ. reconstruction): эта операция агрегирует вышеприведенные патч - представления с высоким разрешением для получения окончательного изображения с высоким разрешением. Этот образ, как ожидается, будет похож на основную истину X.

1.2.3.1 Извлечение и представление патчей

Идея восстановления изображений состоит в том, чтобы плотно извлекать патчи и затем представлять их набором предварительно подготовленных баз, таких как PCA (Principal component analysis). Это эквивалентно свертыванию изображения с помощью набора фильтров, каждый из которых является основой. Оптимизация этих баз включается в оптимизацию сети. Первый слой выражается операцией F_1:

$$F_1(Y) = max\left(0, W_1 * Y + B_1\right) \quad (1.3)$$

где W1 и B1 представляют собой фильтры и смещения, а операция '*' обозначает операцию свертки. Здесь W1 соответствует n1 фильтрам поддержки c × f1 × f1, где c - количество каналов во входном изображении, f1 - пространственный размер фильтра. W1 применяет n1 свертки к изображению, и каждая свертка имеет размер ядра c × f1 × f1. Выходные данные состоят из n1 карт объектов. B1 – это n1 - мерный вектор, каждый

элемент которого связан с фильтром. Применяется выпрямленная линейная единицу (ReLU, max(0, x)) к откликам фильтра.

1.2.3.2 Нелинейное отображение

Первый слой извлекает n1 - мерный объект для каждого патча. Во второй операции сопоставляется каждый из этих n1 - мерных векторов в n2 - мерный. Эти действия эквивалентны применению фильтров n2, которые имеют тривиальную пространственную поддержку 1 × 1. Эта интерпретация справедлива только для фильтров 1×1. Работа второго слоя заключается в следующем:

$$F_2(Y) = max\left(0, W_2 * F_2(Y) + B_2\right) \quad (1.4)$$

Здесь W2 содержит n2 фильтров размером n1 × f2 × f2, а B2 является n2 - мерным. Каждый из выходных n2 - мерных векторов концептуально является представлением патча высокого разрешения, который будет использоваться для реконструкции.

Можно добавить больше сверточных слоев, чтобы увеличить нелинейность. Но это может увеличить сложность модели (n2 × f2 × f2 × n2 параметров для одного слоя), и, следовательно, требует больше времени на обучение.

1.2.3.3 Реконструкция

В традиционных методах предсказанные перекрывающиеся участки высокого разрешения часто усредняются для получения окончательного полного изображения. Усреднение можно рассматривать как заранее определенный фильтр на множестве карт объектов. Исходя из этого определяется сверточный слой для получения конечного изображения с высоким разрешением:

$$F(Y)=W_3*F_2(Y)+B_3 \quad (1.5)$$

Здесь W3 соответствует c-фильтрам размера n2 × f3 × f3, а B3 – c-мерному вектору.

Фильтры действуют как фильтры усреднения, если представления патчей высокого разрешения находятся в области изображения (т. е. можно просто изменить каждое представление, чтобы сформировать патч). Если представления патчей высокого разрешения находятся в некоторых других областях (например, коэффициенты в терминах некоторых базисов), это значит, что W3, являющийся набором линейных фильтров, ведет себя так, как если бы сначала коэффициенты проецировались на область изображения, а затем усреднялись.

Все вышеприведенные три операции приводят к одной и той же форме сверточного слоя. В совокупности они формируют сверточную нейронную сеть. Все фильтрующие веса и смещения должны быть оптимизированы.

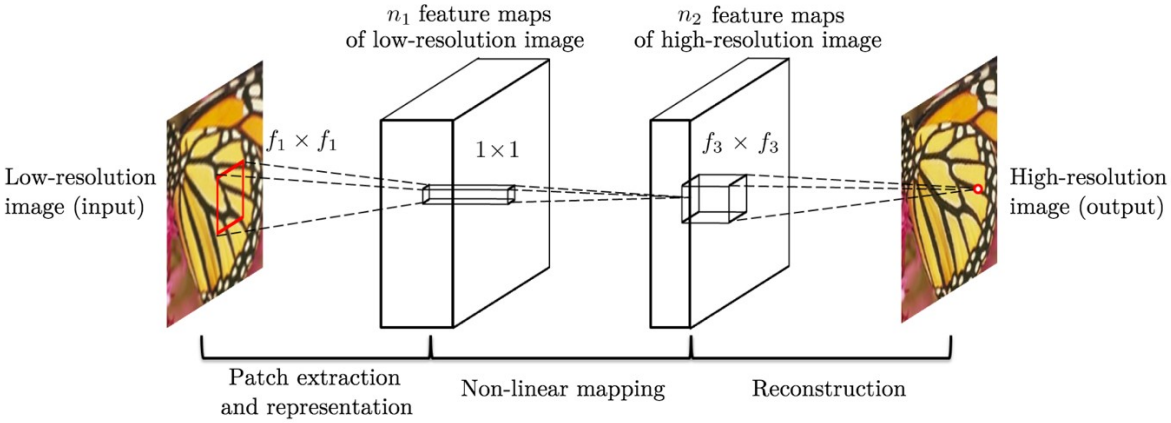

Рисунок 1.4 – Архитектура SRCNN

На рисунке 1.4 показана архитектура SRCNN.

$$L(\theta)=\frac{1}{n}\sum_{i=0}^{n}\left\|F(Y_i;\theta)-X_i\right\|^2 \tag{1.6}$$

Для SR функция потерь L является средним значением среднеквадратичной ошибки (MSE) для обучающих выборок (n), что является своего рода стандартной функцией потерь.

1.2.4 Метод SRGAN

Рассмотренные выше способы увеличения разрешения изображения имеют хорошее быстродействие и точность, однако плохо справляются с задачей восстановления без искажения мелких деталей текстуры из изображения с низким разрешением. Большинство алгоритмов сосредоточены на минимизации среднеквадратичной ошибки реконструкции. Результаты имеют высокие пиковые отношения сигнал-шум (PSNR), что свидетельствует о хорошем качестве изображения, но полученные изображения часто не имеют высокочастотных деталей и являются перцептивно неудовлетворительными[8], они не достигают точности, ожидаемой в изображениях с высоким разрешением. Предыдущие способы пытаются установить сходство в пиксельном пространстве, что приводит к перцептивно неудовлетворительным результатам или размытым изображениям. Эту проблему можно решить использованием генеративно-состязательного подхода.

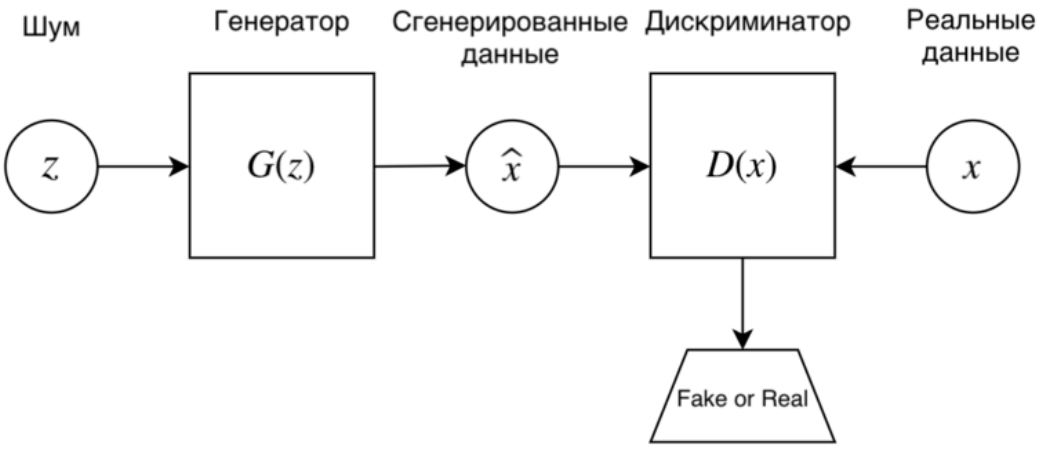

Рисунок 1.5 – Архитектура SRGAN

На рисунке 1.5 показана архитектура SRGAN, состоящая из генератора и дискриминатора. Входными данными служат изображения с шумом. Обозначим изображение, которое мы получаем в результате работы нейронной сети, как SR Images (I^{SR}), изображение низкого разрешения, которое необходимо увеличить в n раз, как LR Images (I^{LR}). Изображение высокого разрешения (без обработки) обозначим как HR Images (I^{HR}). I^{LR} получается в результате уменьшения I^{HR} в n раз.

В результате обучения необходимо получить генерирующую функцию G. Генератор обучается как сверточная нейронная сеть прямого распространения с параметрами ϴG:

$$\theta_G = \{W_{1:L}; b_{1:L}\} \quad (1.7)$$

где $W_{1:L}$ и $b_{1:L}$ это, соответственно, веса и смещения нейронной сети глубиной в L слоев. Параметр ϴG вычисляется путем оптимизации составной функции потерь l_{SR}.

Для обучения нейронной сети находим значение $\widehat{\theta_G}$:

$$\widehat{\theta_G} = arg\min_{\theta_G} \frac{1}{N} \sum_{n=1}^{N} l^{SR}\left(G_{\theta_G}\left(I_n^{LR}\right), I_n^{HR}\right) \quad (1.8)$$

Далее определяется дискриминантная сеть D_{θ_D}, которая оптимизируется поочередно вместе с G_{θ_G} для решения задачи состязательности min-max [11]:

$$\min_{\theta_G} \max_{\theta_D} E_{I^{HR} \sim p_{train}(I^{HR})}\left[\log D_{\theta_D}\left(I^{HR}\right)\right] + ¿ E_{I^{LR} \sim p_{train}(I^{LR})} ¿¿ \quad (1.9)$$

Общая идея этой формулировки заключается в том, что она позволяет обучить генеративную модель G с целью «обмануть» дифференцируемый дискриминатор D, который обучен отличать сверхразрешённые изображения от реальных изображений [9]. При таком подходе генератор может научиться создавать решения, максимально приближенные к реальным образам, которые трудно классифицировать по D.

В основе генераторной сети G лежат остаточные блоки B с идентичной компоновкой. Используется два сверточных слоя с небольшими ядрами размером 3×3 и 64 картами объектов, за которыми следуют слои пакетной нормализации и параметрическая функция активации. Увеличивается разрешение входного изображения с помощью двух обученных субпиксельных слоев свертки.

Для того чтобы отличить реальные изображения HR от сгенерированных образцов SR, обучается сеть дискриминаторов. Архитектура показана на рисунке 1.6. Используется активация LeakyReLU (α = 0.2) [10] и избегается максимальное объединение по всей сети. Дискриминаторная сеть обучается решению задачи максимизации по

формуле 1.9. Она содержит восемь сверточных слоев с ядрами размером 3 × 3 и карты признаков, увеличивающихся в 2 раза с 64 до 512 в каждом слое. Шаговые свертки используются для уменьшения разрешения изображения каждый раз, когда количество объектов удваивается. За полученными 512 картами признаков следуют два плотных слоя и конечная сигмовидная активационная функция [10] для получения вероятности классификации образцов.

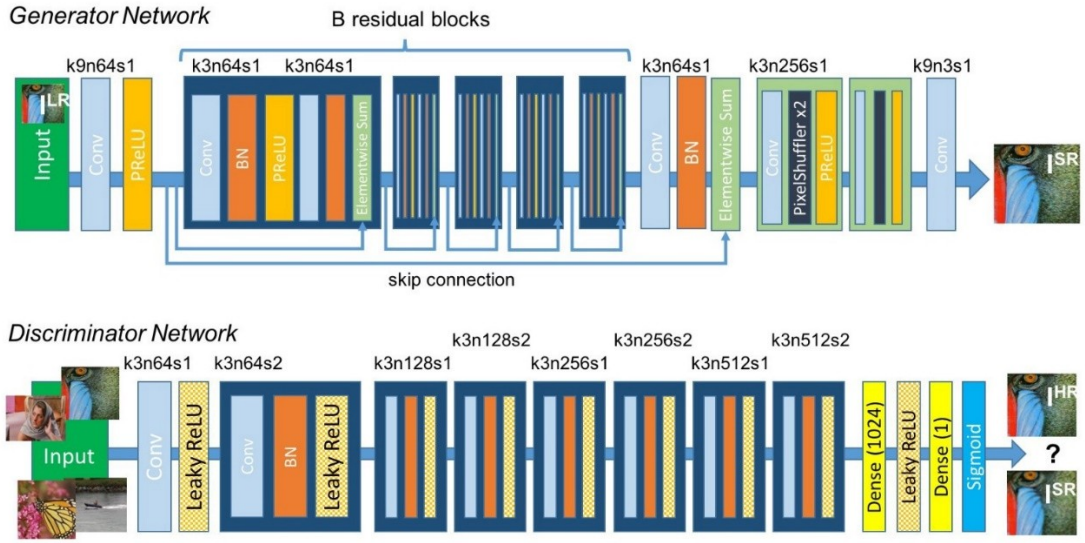

Рисунок 1.6 – Архитектура генераторной и дискриминаторной сети с соответствующим размером ядра (k), количеством карт признаков (n) и шагом (s), указанным для каждого сверточного слоя.

На рисунке 1.6 изображены архитектура генератора и дискриминатора.

Генератор:

- Input I^{LR} – изображение с низким разрешением, которое генератор использует в качестве входных данных;
- Conv – сверточный слой с n картами признаков (англ. feature maps) и шагом (англ. stride), равным s;

- ReLU – активационная функция Parametric ReLU;
- BN – батч-нормализация (англ. batch normalization);
- skip connection – дополнительные связи;
- B residual blocks – остаточные блоки;
- Elementwise sum – поэлементная сумма;
- PixelShuffler x2 – субпиксельные сверточные слои;
- I^{SR} – изображение высокого разрешения, полученное в результате работы нейронной сети.

Дискриминатор:

- Input – входные данные, изображения высокого разрешения и изображения, созданные генератором;
- Conv – сверточный слой с n картами признаков (feature maps) и шагом (stride), равным s;
- Leaky ReLU – активационная функция Leaky ReLU;
- BN – батч-нормализация (англ. batch normalization);
- Dense – полносвязный слой;
- Sigmoid – активационная функция сигмоида;
- I^{HR}, I^{SR} – изображение с высоким и изображение с низким разрешениями соответственно;
- результат работы дискриминатора, определившего класс изображения – настоящее изображение высокого разрешения или оно создано генератором.

Функция перцептивной потери имеет вид:

$$l^{SR} = l_X^{SR} + 10^{-3} l_{Gen}^{SR} \quad (1.10)$$

где l_X^{SR} – пиксельная потеря, а $10^{-3} l_{Gen}^{SR}$ – состязательные потери.

Пиксельная потеря MSE рассчитывается как:

$$l_{MSE}^{SR} = \frac{1}{r^2 WH} \sum_{x=1}^{rW} \sum_{y=1}^{rH} \left(I_{x,y}^{HR} - G_{\theta_G}\left(I^{LR}\right)_{x,y} \right)^2 \quad (1.11)$$

Это часто используемая цель оптимизации для SR изображений, на которую опираются многие современные подходы. Но у данного метода есть недостаток - при достижении высокого пикового отношения сигнала к шуму (англ. peak signal-to-noise ratio, PSNR) решения задач оптимизации MSE часто не имеют высокочастотного содержания, что приводит к перцептивно неудовлетворительным решениям с чрезмерно гладкими текстурами.

Состязательные потери l_{Gen}^{SR} определяются на основе вероятностей работы дискриминатора $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ по всем обучающим выборкам следующим образом:

$$l_{Gen}^{SR} = \sum_{n=1}^{N} -\log D_{\theta_D}(G_{\theta_G}(I^{LR})) \quad (1.12)$$

где $D_{\theta_D}(G_{\theta_G}(I^{LR}))$ - вероятность того, что восстановленное изображение $G_{\theta_G}(I^{LR})$ является HR изображением.

1.3 Аугментация изображений

Аугментация данных – это метод создания дополнительных обучающих данных из имеющейся обучающей выборки. Для достижения хороших результатов нейронные сети должны обучаться на очень большом наборе тестовых данных.

Существует методики аугментации обучающей выборки:

1. заранее выполнить все необходимые преобразования, существенно увеличив размер набора обучающих данных;
2. выполнять преобразование непосредственно перед подачей в модель машинного обучения.

Первая методика предпочтительна для небольшой обучающей выборки, так как в конечном итоге набор данных увеличится на коэффициент, равный количеству выполняемых преобразований (например, если перевернуть все изображения из обучающей выборки, она увеличится в два раза).

Вторая методика предпочтительна для большого набора обучающей выборки. Так как современные выборки состоят из сотен тысяч изображений, хранение таких данных требует большого количество памяти, что не всегда выполнимо на практике.

Объём и разнообразие обучающей выборки влияют на качество результата обучения нейросети.

Аугментация изображения состоит из следующих действий [12]:

– вращение;
– горизонтальное сдвиг;
– вертикальный сдвиг;
– масштабирование;
– горизонтальное отражение;
– вертикальное отражение;
– разрез изображения в горизонтальном направлении;

В разработке программного продукта была использована вторая методика аугментации, так как ЭВМ, использованная для обучения нейронной сети имеет малый объём видеопамяти. Аугментация происходила непосредственно перед подачей изображения на вход генератору и состояла из трёх действий, которые применяются к каждому

изображению единовременно: горизонтальное отражение, масштабирование и вращение.

1.4 Формализация постановки задачи

Основываясь на приведенном выше анализе предметной области можно сформулировать постановку задачи, решаемой в данной работе.

Для выполнения поставленной задачи было необходимо спроектировать и реализовать программный продукт, осуществляющий увеличение разрешения черно-белых изображений, изменённых с помощью метода аугментации. На вход программный продукт получает оригинальное цветное или черно-белое изображение формата PNG, затем применяет к нему фильтр, который преобразует полученные данные в черно-белые изображения, то есть цветовой характеристикой будет являться оттенок серого цвета, а не RGB модель, затем сжимает полученное изображение, тем самым получается изображение с низким разрешением. На выходе программный продукт генерирует изображение с увеличенным разрешением формата PNG.

Для функционирования генеративно-состязательной нейронной сети необходимо провести её обучение с использованием готового набора изображений. Для достижения высоких результатов обучения данный набор должен состоять из различных изображений, количество этих изображений влияет на результат работы нейронной сети. В качестве генеративно-состязательной нейронной сети выбран SRGAN, так как имеет высокую эффективность работы среди описанных сетей.

Так же необходимо провести исследование эффективности работы генеративно-состязательной нейронной сети, которая натренирована на аугментированных изображениях, подав в качестве входных данных не изменённое изображение.

1.5 Вывод

В аналитическом разделе был проведён анализ предметной области, в ходе которого сформулирована основная задача, проанализированы существующие методы увеличения разрешения изображений и выбран вариант решения для поставленной задачи. Рассмотрены методы аугментации изображений, реализован метод, подходящий под требования и возможности ЭВМ, на котором происходит обучение нейронной сети.