УДК 004.891

АЛГОРИТМЫ ПОВЫШЕНИЯ ЧИТАБЕЛЬНОСТИ ТЕКСТА НА ИЗОБРАЖЕНИЯХ НИЗКОГО КАЧЕСТВА С ПРИМЕНЕНИЕМ НЕЙРОСЕТЕВЫХ МОДЕЛЕЙ

Сачивко Н.С.
Студент факультета компьютерных систем и сетей БГУИР
skriepcoder@gmail.com

Калугина М.А.
Доцент кафедры информатики БГУИР, кандидат физико-математических наук, доцент
marina_kalugina@list.ru

Н.С. Сачивко

В 2020 году окончил Браславскую гимназию. Неоднократно становился победителем олимпиад по математике, информатике, финансовой грамотности, английскому языку. Интересуется информационной безопасностью, информатикой и математикой.

М.А. Калугина

Окончила Белорусский государственный университет. Область научных интересов связана с исследованием проблем метрической теории диофантовых приближений зависимых величин и приложений математических методов к нейросетевому анализу.

Аннотация. В рамках данной статьи рассмотрена проблема нечитаемого текста на изображениях низкого качества. Предложено два алгоритма повышения его читабельности: алгоритм увеличения размера изображения и алгоритм снижения степени его размытости. Описаны устройства моделей используемых нейросетей, приведены результаты работы предложенных алгоритмов.

Ключевые слова: машинное обучение, глубокое обучение, нейросети, нейросетевые модели, распознавание текста, увеличение изображений, повышение качества изображений, устранение размытия, повышение читабельности текста.

Введение. Не так давно фотография являлась непозволительной роскошью для многих людей, сам процесс съёмки занимал продолжительный период времени, а качество таких фотографий было достаточно низким. Затем появились фотоаппараты, использующие фотоплёнку; они уже были доступны большему количеству людей, но требовалось немало усилий и времени для получения готовых фотографий. Позже появились первые фотокамеры моментальной печати «*Polaroid*» с черно-белыми снимками.

Сегодня, пожалуй, каждый человек хоть раз в жизни сталкивался с цифровой фотографией в том или ином виде: полученной с помощью полноценного цифрового фотоаппарата, встроенной в иное устройство фотокамеры или даже сканера. Такой метод получения изображений и неразрывно связанных с ними видеозаписей встречается повсеместно: для любительской фотографии с использованием смартфонов, в криминалистической фотографии, для автоматической фиксации нарушений правил

дорожного движения, в целях оцифровки медицинских карт, для регистрации происшествий на дорогах (например, с помощью автомобильных видеорегистраторов). Безусловно, часто такие изображения (к слову, видеозаписи можно рассматривать как серию изображений) содержат важные сведения именно в текстовом виде, будь то номера автомобилей, рукописные медицинские сведения или важные документы.

К сожалению, не бывает идеальных систем фото- или видеофиксации, изображения не могут содержать в себе неограниченно детализированные сведения, а некоторые камеры способны производить лишь «замыленные» изображения низкого разрешения. Другими словами, всё в нашем мире конечно, в том числе и возможности цифровых фото- и видеокамер. Нередко текст на изображениях оказывается нечитаемым за счёт низкого разрешения последнего. Также не исключена ситуация, когда изображение получается размытым по той или иной причине. Если такое изображение попадает в систему, которая распознаёт текст в автоматическом режиме, то из-за низкого качества изображения возникает необходимость в его проверке человеком в ручном режиме. Впрочем, бывают случаи, когда не каждый человек способен распознать текст на таких изображениях – приходится либо заново производить фото- или видеосъёмку, если такая возможность имеется, либо просто пропускать изображение, что часто приводит к определённым потерям, а иногда и к критическим последствиям.

Один из подходов к решению проблемы распознавания текста на изображениях низкого качества – повышение качества изображения. Например, можно увеличить его разрешение, снизить степень размытости, устранить шум. Однако возникает проблема, связанная со сложностью данных операций: на самом деле, задача восстановления исходного изображения из размытого или из изображения с низким разрешением имеет слишком большое количество решений [1], то есть получить точно исходное изображение из «испорченного» на практике невозможно. Именно поэтому предлагается использовать нейросети и методы глубокого обучения для решения указанных задач.

Описание алгоритмов и обзор используемых технологий. Предполагается, что исходное изображение представлено в формате *RGB* и имеет размер по крайней мере 30×30 пикселей. В статье рассматривается два алгоритма: увеличения размера изображения и уменьшения степени его размытости.

Для помощи в решении поставленных задач взят язык программирования *Python*, библиотеки *Pillow*, *PyTorch* 2.1, *Torchvision* 0.16 и *Matplotlib*. В качестве аппаратного обеспечения используется среднестатистический домашний компьютер.

Важной особенностью библиотеки *Torchvision* является способ обработки изображений: встроенная функция *to_tensor* преобразует изображение в тензор, при этом стандартный диапазон яркости цветов от 0 до 255 приводится к диапазону $[0.0;\ 1.0]$. Этим и обусловлен выбор последнего слоя нейронных сетей: наиболее подходящим является слой активаций *Sigmoid*, ведь именно он обеспечивает корректный диапазон значений.

Функция активации *Sigmoid* определяется следующим образом:

$$s(x) = \frac{1}{1 + e^{-x}}.$$

Также будет применяться функция активации *ReLU*:

$$r(x) = \max(0, x).$$

Описание модели для увеличения размера изображения. Первая задача, которую необходимо решить, – это увеличение размера изображения. Для получения результата было принято решение использовать комбинацию свёрточной нейронной сети

(*Convolutional Neural Network* или *CNN*) и такой же сети, работающей в обратном процессе (*Deconvolutional Neural Network* или *DNN*).

Поскольку исходное изображение представлено в формате *RGB*, количество каналов на входе и на выходе нейросети принимается равным трём.

В ходе тестирования было установлено, что модель со следующими слоями является наиболее оптимальной (стрелкой указывается переход от исходного количества каналов к количеству каналов на выходе слоя):

1 *Deconvolution*: $3 \rightarrow 75$, ядро 6×6, шаг 2.
2 *ReLU*.
3 *Convolution*: $75 \rightarrow 75$, ядро 4×4, шаг 1.
4 *ReLU*.
5 *Convolution*: $75 \rightarrow 75$, ядро 4×4, шаг 1.
6 *ReLU*.
7 *Deconvolution*: $75 \rightarrow 3$, ядро 6×6, шаг 2.
8 *Sigmoid*.

Такая модель обеспечивает увеличение размера изображения в 4 раза.

Стоит отметить, что размеры ядра свёртки и шага для слоёв «деконволюции» (то есть слоёв, которые работают в обратном процессе по отношению к слоям свёртки, англ. «*deconvolution*») выбраны неслучайно: всё для того, чтобы избежать эффекта «шахматной доски». Если бы размер ядра свёртки не делился без остатка на шаг «деконволюции», то в результате этого процесса мог бы возникнуть «неравномерный нахлёст» (англ. «*uneven overlap*») фрагментов, который и привёл бы к возникновению указанного эффекта [2]. Такой эффект заметен невооружённым глазом, как представлено на рисунке 1.

Рисунок 1. Пример эффекта «шахматной доски» [3]

Выбор набора данных для обучения модели для увеличения размера изображения. Изначально планировалось использовать набор изображений, содержащих части страниц общедоступных документов [4] – пример такого изображения представлен на рисунке 2.

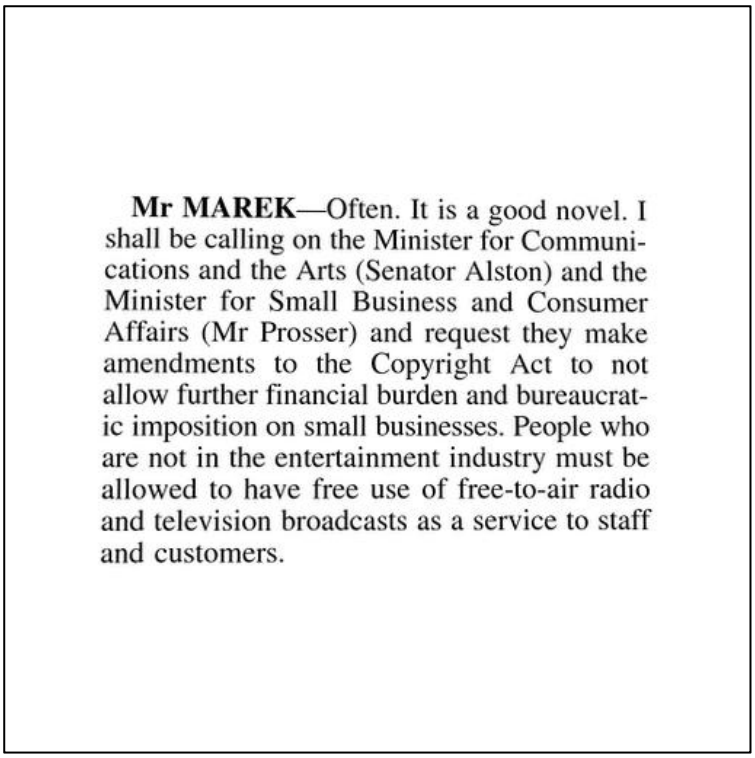

Рисунок 2. Пример исходного изображения (*BMVC*)

Такой набор данных, безусловно, хорош в случае, когда модель должна обрабатывать только документы. Но у такой модели есть один существенный недостаток: она преобразует все изображения, в том числе цветные, в чёрно-белые. По этой причине было решено использовать другой набор данных.

Например, существует набор синтетически сгенерированных символов [5] – они также являются чёрно-белыми, но все изображения в наборе содержат всего два цвета: либо полностью белый, либо полностью чёрный, что позволяет перекрасить их. Пример исходного символа из данного набор представлен на рисунке 3.

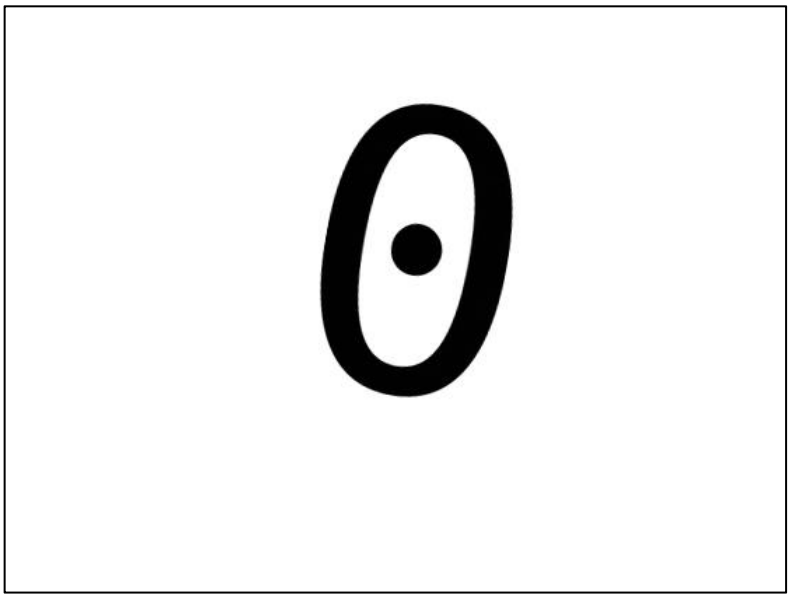

Рисунок 3. Пример исходного изображения (*Synthetic Character Set*)

С указанным набором данных были проделаны следующие операции:

1 Все изображения обрезаны так, чтобы по краям не оставалось белого фона.

2 Чёрный цвет в 20% случаев заменён на случайный.

3 Полученные изображения объединены в коллекции, причём в 20% случаев белый фон заменён случайным цветом, а в 50% случаев итоговое изображение повёрнуто на случайный угол.

4 Итоговые коллекции обрезаны до одинакового размера.

Примеры изображений из полученного набора (назовём его «исходный набор») представлены на рисунке 4.

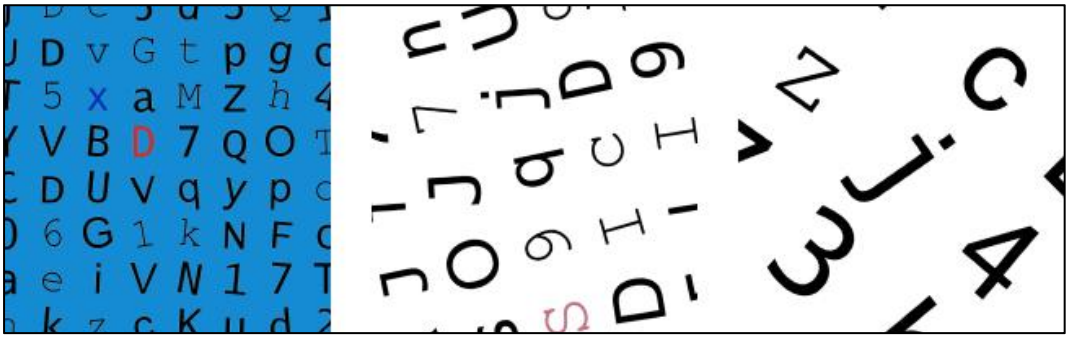

Рисунок 4. Пример изображений исходного набора

Изображения из исходного набора были уменьшены в 4 раза с использованием следующих методов:

– *BOX*;
– *HAMMING*;
– *LANCZOS*;
– *BICUBIC*;
– *BILINEAR*.

Примеры изображений из полученного набора (назовём его «итоговый набор») представлены на рисунке 5.

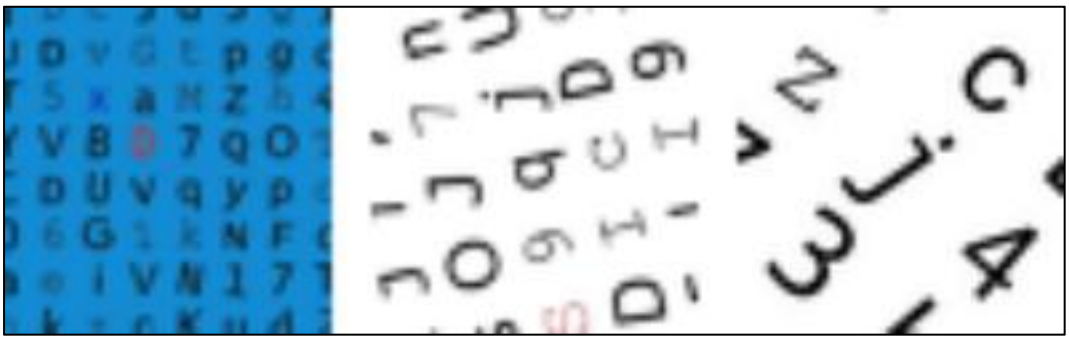

Рисунок 5. Пример изображений итогового набора

Для обучения созданной модели изображения из исходного и итогового наборов были собраны в пары, 70% таких пар отобраны случайным образом непосредственно для обучения модели, остальные 30% используются для валидации результатов.

Стоит отметить, что данные не были нормализованы. Хоть процесс нормализации и позволял достигнуть меньшей ошибки модели, но это также зачастую приводило к искажению цветов изображения в процессе обработки, что в итоге негативно сказывалось на восприятии человеком результата.

Обучение модели для увеличения размера изображения и достигнутые результаты. Модель обучалась из начального состояния, которое определялось случайным образом с помощью метода инициализации Ксавье (*Xavier* или *Glorot initialization*) [6].

Для обучения использовался модифицированный алгоритм стохастического градиентного спуска, именованный *Adam*. В ходе тестирования установлено, что именно данный алгоритм обеспечивает хорошую сходимость. В качестве меры ошибки была выбрана среднеквадратичная ошибка (*MSELoss*), которая определяется следующим образом:

$$l(X,Y) = \frac{\sum_{i=1}^{N}(X_i - Y_i)^2}{N},$$

где $N = |X| = |Y|$; X – исходный набор данных, Y – итоговый набор данных.

Модель обучалась в течение 600 эпох (одна эпоха означает, что все данные из созданного набора были использованы для обучения ровно один раз). График зависимости среднеквадратичной ошибки (оранжевым – ошибка на данных для валидации, синим – на тренировочных данных) от эпохи представлен на рисунке 6.

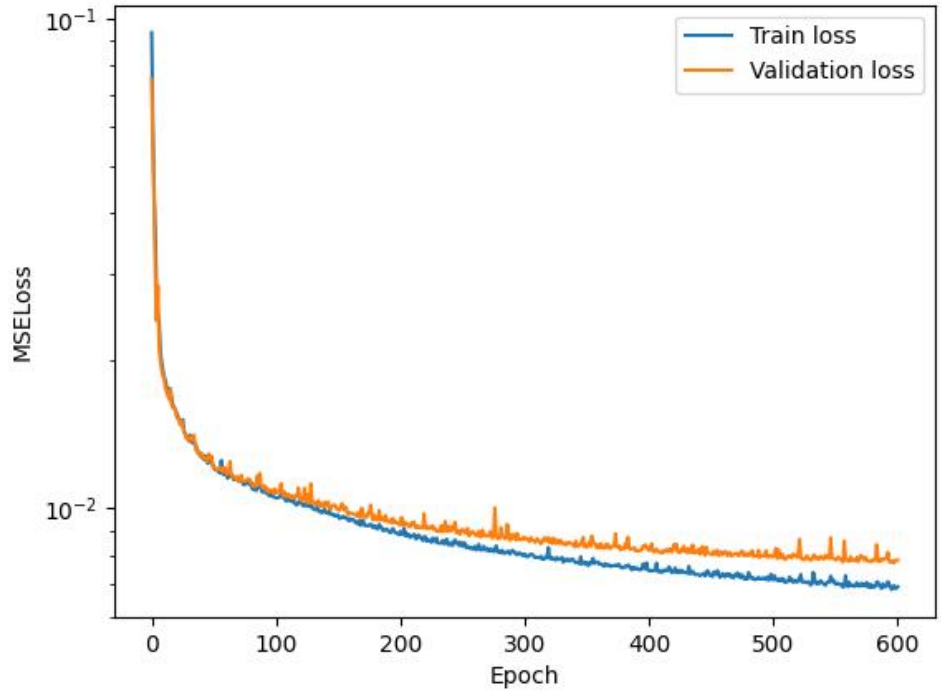

Рисунок 6. Ошибка на валидационных (оранжевым) и тренировочных (синим) данных, логарифмическая шкала

Пример обработки уменьшенных изображений представлен на рисунке 7, более детальное сравнение – на рисунке 8.

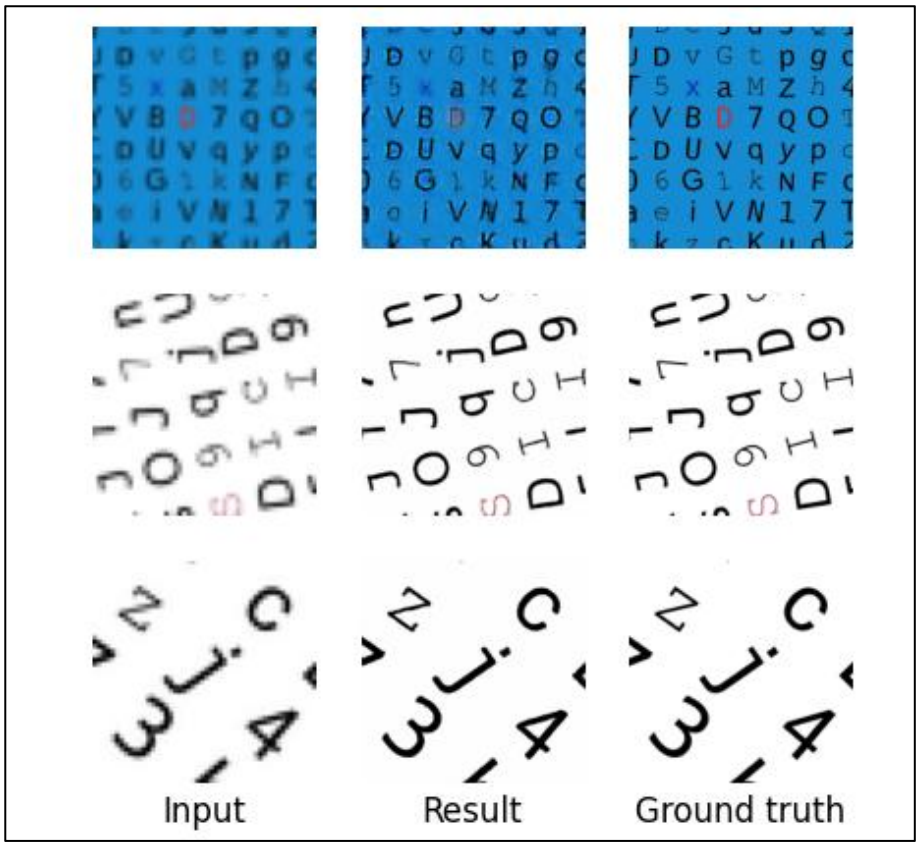

Рисунок 7. Пример обработки уменьшенных изображений

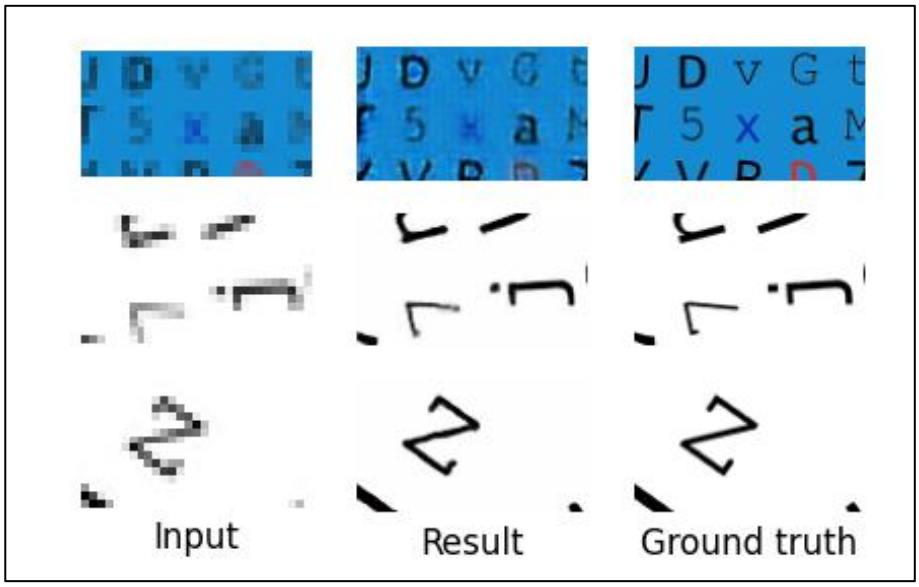

Рисунок 8. Детальное сравнение результатов обработки уменьшенных изображений

Видно, что модель корректно выполняет свою функцию: текст становится более чётким при увеличении размера изображения. К сожалению, при работе с цветным фоном результат несколько хуже, чем при работе с белым фоном. Вероятно, это можно исправить путём увеличения объёма набора данных для тренировки, добавлением большего количества цветных изображений и обучением модели в течение большего количества эпох.

Описание модели для снижения степени размытости изображений. Данная модель призвана решить задачу снижения степени размытости изображения. Для этого было принято решение использовать свёрточную нейронную сеть (*Convolutional Neural Network* или *CNN*).

Как и ранее, исходное изображение представлено в формате *RGB*, в связи с чем количество каналов на входе и на выходе нейросети принимается равным трём.

Архитектура данной нейросети схожа с архитектурой ранее рассмотренной сети, но имеет существенные отличия (стрелкой указывается переход от исходного количества каналов к количеству каналов на выходе слоя):

1 *ReflectionPad*: по 4 пикселя снизу, сверху, слева и справа.
2 *Convolution*: $3 \rightarrow 75$, ядро 4×4, шаг 1.
3 *ReLU*.
4 *Convolution*: $75 \rightarrow 75$, ядро 2×2, шаг 1.
5 *ReLU*.
6 *Convolution*: $75 \rightarrow 75$, ядро 1×1, шаг 1.
7 *ReLU*.
8 *Convolution*: $75 \rightarrow 75$, ядро 2×2, шаг 1.
9 *ReLU*.
10 *Convolution*: $75 \rightarrow 3$, ядро 4×4, шаг 1.
11 *Sigmoid*.

В архитектуре данной сети используется новый слой: *ReflectionPad*. Необходимость его применения объясняется особенностью слоя свёртки: фактически для каждого канала применяется фильтр, в результате чего значения результирующей матрицы являются суммами произведений элементов фильтра на элементы исходной матрицы. Нетрудно установить, что в случае, когда используется размер фильтра больший, чем 1×1, результирующая матрица имеет меньший размер, чем исходная.

С целью сохранения исходного размера изображения оно искусственно расширяется на 4 пикселя во все стороны, причём расширяется неслучайно, а путём отражения частей изображения, расположенных на его границе. Для наглядности на рисунке 9 приведён пример использования *ReflectionPad* на 2 пикселя во все стороны для изображения размером 4×4 пикселя.

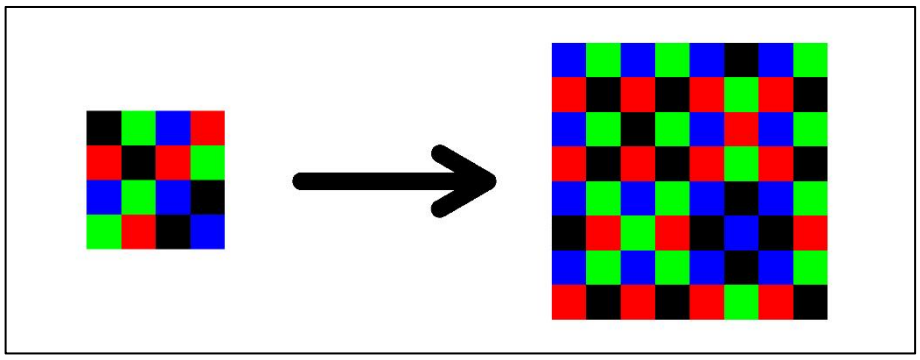

Рисунок 9. Пример использования *ReflectionPad*

Выбор набора данных для обучения модели для снижения степени размытости изображений. На данном этапе было решено использовать некоторую часть из набора данных с *British Machine Vision Conference* (*BMVC*) [4], а именно сгенерированные функции рассеяния точки.

Функция рассеяния точки (англ. *Point Spread Function* или *PSF*) представляет собой изображение, которое было бы получено при наблюдении точечного источника света в

некоторых условиях. Именно эти условия и можно описать с помощью *PSF*. В частности, такие функции удобно применять при моделировании размытия.

Итак, из набора *BMVC* случайным образом были выбраны функции рассеяния точки. Пример выбранных функций представлен на рисунке 10.

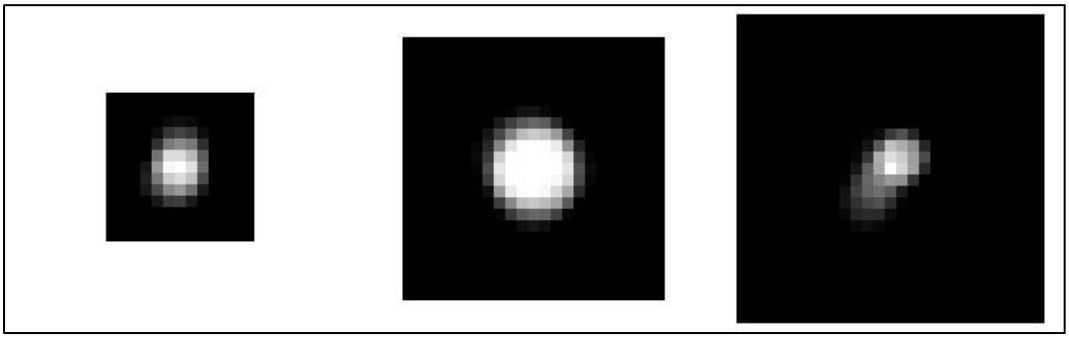

Рисунок 10. Пример используемых *PSF*

Для применения *PSF* достаточно конвертировать рассмотренные представления функции в виде изображения в матрицу, состоящую из нормированных яркостей пикселей, и применить к исходному изображению фильтр, ядром которого выступает указанная матрица.

В качестве исходного был использован набор, аналогичный тому, что использовался для обучения модели для увеличения изображения (см. рисунок 4). К каждому изображению из исходного набора была применена соответствующая ему случайным образом выбранная функция рассеяния точки. Примеры изображений из полученного набора (назовём его «итоговый набор») представлены на рисунке 11.

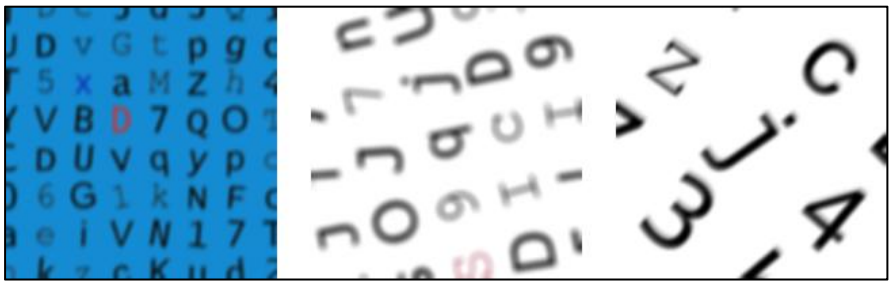

Рисунок 11. Пример изображений итогового набора

Для обучения созданной модели изображения из исходного и итогового наборов были собраны в пары, 70% таких пар отобраны случайным образом непосредственно для обучения модели, остальные 30% используются для валидации результатов. Как и в случае с моделью для увеличения размера изображения, данные не были нормализованы.

Обучение модели для снижения степени размытости изображений и достигнутые результаты. Как и в ранее рассмотренном случае, модель обучалась из начального состояния, которое определялось случайным образом с помощью метода инициализации Ксавье (*Xavier* или *Glorot initialization*) [6].

В качестве меры ошибки была выбрана среднеквадратичная ошибка (*MSELoss*). Для обучения использовался тот же модифицированный алгоритм стохастического градиентного спуска: *Adam*. Аналогично предыдущей модели данный алгоритм показал хорошую скорость сходимости и позволил снизить итоговую ошибку.

Модель обучалась на протяжении 1000 эпох. График зависимости среднеквадратичной ошибки (оранжевым – ошибка на данных для валидации, синим – на

тренировочных данных) от эпохи представлен на рисунке 12.

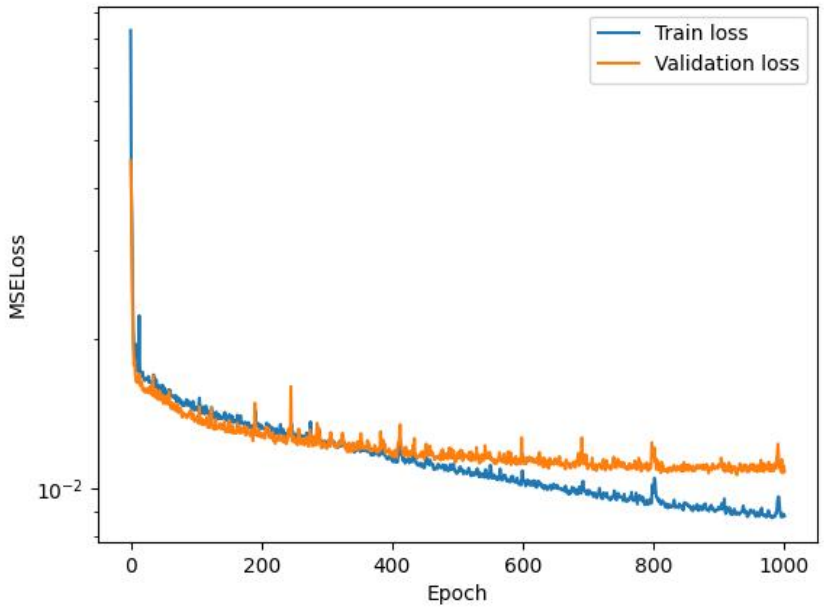

Рисунок 12. Ошибка на валидационных (оранжевым) и тренировочных (синим) данных, логарифмическая шкала

Видно, что ошибка на данных для валидации на последних эпохах практически не уменьшалась, в то время как ошибка на тренировочных данных продолжала падать. Это может свидетельствовать о переобучении модели. Впрочем, нельзя сказать, что ошибка на валидационных данных начала расти, да и, как будет видно далее, нельзя утверждать о плохом в том или ином смысле поведении модели на таких данных.

Пример обработки уменьшенных изображений представлен на рисунке 13, более детальное сравнение – на рисунке 14.

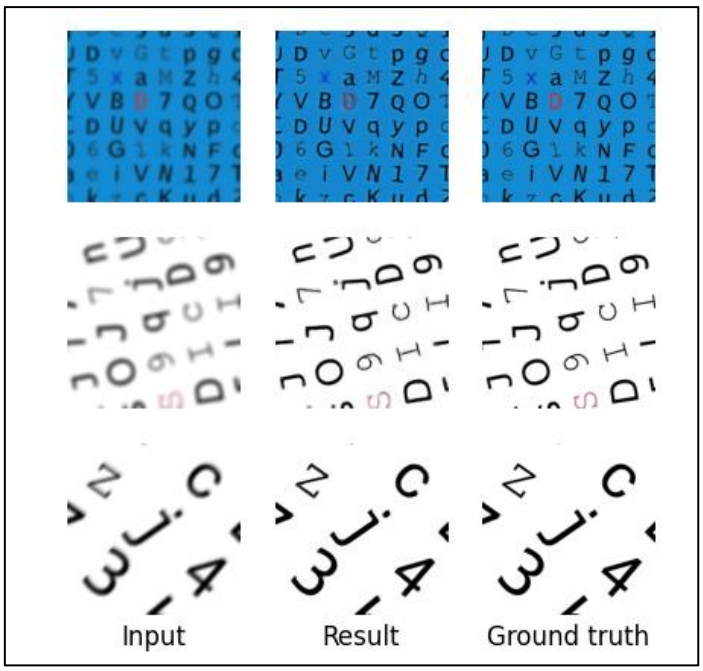

Рисунок 13. Пример обработки размытых изображений

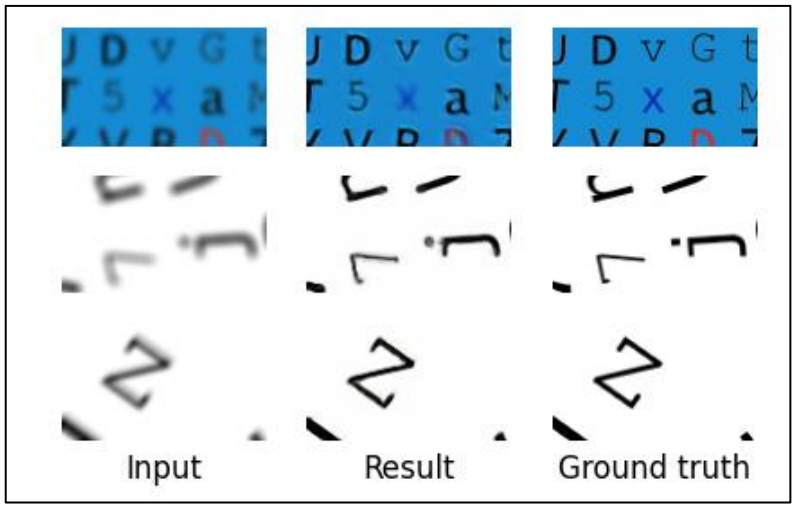

Рисунок 14. Детальное сравнение результатов обработки размытых изображений

Комбинирование обученных моделей. Безусловно, можно применять полученные модели независимо друг от друга, однако зачастую низкое качество изображения подразумевает именно комбинацию размытости и низкого разрешения. По результатам проведённых экспериментов предлагается сначала увеличивать размер изображения, а затем снижать степень его размытости.

Заключение. Основными показателями, влияющими на качество текстовых изображений, являются разрешение и степень размытости. Предложенные алгоритмы успешно решают задачи увеличения размера изображения в 4 раза и снижение степени размытости текста. Примечательно, что рассмотренные нейросети могут обрабатывать изображения практически любых размеров (за исключением самых маленьких). Достигнутые результаты теоретически можно улучшить, например, путём модификации архитектуры нейронных сетей, изменения используемых наборов данных, использования профессионального оборудования для быстрого обучения более глубоких сетей, пред- или постобработки изображений с помощью иных методов.

Список литературы

[1] X. Xu, D. Sun, J. Pan, Y. Zhang, H. Pfister and M. -H. Yang, "Learning to Super-Resolve Blurry Face and Text Images," 2017 IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 2017, pp. 251-260, doi: 10.1109/ICCV.2017.36.

[2] Odena, et al., "Deconvolution and Checkerboard Artifacts", Distill, 2016. http://doi.org/10.23915/distill.00003.

[3] Donahue J., Krähenbühl P., Darrell T. Adversarial feature learning //arXiv preprint arXiv:1605.09782. – 2016.

[4] BMVC OCR test data [Электронный ресурс]. — Режим доступа: https://www.fit.vutbr.cz/~ihradis/CNN-Deblur/BMVC_OCR_test_data.tar.gz. — Дата доступа: 25.11.2023.

[5] Synthetic Character Set [Электронный ресурс]. — Режим доступа: https://www.kaggle.com/datasets/shreyasubbu/synthetic-character-set. — Дата доступа: 10.12.2023.

[6] Glorot, X. & Bengio, Y. (2010). Understanding the difficulty of training deep feedforward neural networks. Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics, in Proceedings of Machine Learning Research 9:249-256. https://proceedings.mlr.press/v9/glorot10a.html.

Авторский вклад

Сачивко Никита Сергеевич – постановка задачи исследования, создание архитектуры нейросетевых моделей, обучение созданных моделей, анализ полученных результатов.

Калугина Марина Алексеевна – руководство исследованием по повышению читабельности текста на изображениях низкого качества с применением нейросетевых моделей.

ALGORITHMS FOR ENHANCING READABILITY OF TEXT IN LOW-QUALITY IMAGES USING DEEP NEURAL NETWORK MODELS

N.S. Sachivko
Student of BSUIR

M.A. Kalugina
Associate Professor of Informatics Department of the BSUIR

Abstract. This work addresses the problem of unreadable text in low-quality images. Two algorithms have been proposed to improve its readability: an algorithm for upscaling an image and an algorithm for deblurring it. The architecture of neural network models has been described; the results of the proposed algorithms have been presented.

Keywords: machine learning, deep learning, neural networks, neural network models, text recognition, image upscaling, image enhancing, deblurring, enhancing text readability, improving text readability.