Санкт-Петербургский государственный университет

Кафедра теории управления

Эбраль Александр Владимирович

Выпускная квалификационная работа бакалавра

Сегментация текста на изображениях

Направление 01.03.02

Прикладная математика и информатика

Научный руководитель,
кандидат технических наук,
доцент
Гришкин В.М.

Санкт-Петербург

2018

Содержание

Введение

В настоящее время большое распространение получила задача семантической сегментации - точного выделения объектов различных классов на изображениях. Она нашла применение во многих сферах:

- Автомобилестроение - классификация дорожных знаков, разметки, пешеходов и тд. на изображении с камеры автомобиля.
- Медицина - распознавание различных новообразований и отклонений на кт\мрт-снимках.
- Биология - исследование численности редких видов животных по снимкам со спутника.
- Различного рода аналитика

Данный список не следует считать полным, он приведен только лишь для показа широты применимости данной задачи.

Стоит отметить разницу задач обнаружения и сегментации объектов (рис. 1.1).

Рис. 1.1. Слева направо: исходное изображение, результат обнаружения текста, результат сегментации.

На рисунке выше видно, что, в случае задачи обнаружения, достаточно получить минимальный прямоугольник, описывающий объект, а в случае сегментации - четкий контур, что важно в текущей задаче, т.к. в противном

случае усложняется процесс отделения текста от фона.

Данная работа рассматривает сегментацию текста на сложном фоне, результаты которой предполагается использовать для различного рода аналитики. Суть задачи состоит в выделении на входных изображениях регионов, содержащих текстовую информацию, которая, в свою очередь, легче поддается различного рода анализу (анализ настроения текста, его тематики и т.д.). В таком виде задача применима в надзадаче - описания сцены, которая в последнее время становится все более актуальной, т.к. количество медиа-контента в сети растет, отсюда возникает потребность в его обработке и анализе.

В рамках данной работы не рассматривается последующее устранение геометрических искажений текста, также не предполагается оптического распознавания символов (OCR [1]) и самой аналитики на полученных текстовых областях.

Постановка задачи

1. Формальное описание

Требуется разработать алгоритм (или их ансамбль), который получает на вход K растровых изображений размерностью $X \times Y \times P$, где X и Y - кол-во пикселей по ширине и высоте, P - количество цветовых каналов. И для $\forall i \in \{1, .., K\}$, где i − номер входного изображения, выдает данные о расположении обнаруженных сегментов текста в виде:

$$\left\{ \left\{ \left(x_1^i, y_1^i\right), .., \left(x_{k1}^i, y_{k1}^i\right) \right\}^1, .., \left\{ \left(x_1^i, y_1^i\right), .., \left(x_{km}^i, y_{km}^i\right) \right\}^m, .. \right\},$$

где $\left(x_j^i \in [0, X], y_j^i \in [0, Y]\right)$ - координаты j-ой точки замкнутой m-ой ломаной, описывающей контур одного из символов алфавита, а также: $\left\{ p^i_1, .., p^i_m, .. \right\}$, где $p_m \in [0, 1]$ − вероятность принадлежности m-ого контура к некоторому классу (фона, текста или определенного символа).

Каждая m-ая текстовая область может быть повернута на любой из трех углов Эйлера (α, β, γ), относительно плоскости изображения. Также плоскость области может быть геометрически искажена из-за физических деформаций объекта, на котором она располагается. Валидной будем считать ту область, где человек все еще может распознать текст.

2. Формат входных данных

Для тестирования алгоритма используется некоторое число M заранее размеченных (как описано в п.1) изображений, содержащих регионы с текстом. Размеченным изображением считается такое, для которого в соответствие установлена матрица C размерности $X \times Y$, каждый элемент

которой задает класс соответствующего пикселя на изображении:

$$C = \left\{ c_{ij} : 0 \le i \le X,\ 0 \le j \le Y,\ 0 \le c_{ij} \le n,\ c_{ij} \in N,\ i \in N, j \in N \right\},$$

где n - число сегментируемых классов, N - множество натуральных чисел.

3. Оценка результатов

Для оценивания результатов работы алгоритма воспользуемся некоторым числом $K_{test} \le M$ предварительно размеченных изображений. Оценку будем проводить по метрике MIoU - (Mean Intersection of Union) - среднего пересечения к объединению:

$MIoU = \frac{\sum_{m=1}^{k} \frac{S^m_{overlap}}{S^m_{union}}}{k}$, где

$S^m_{overlap} = S^m_{pred} \cap S^m_{real}$,

$S^m_{union} = S^m_{pred} \cup S^m_{real}$,

а S^m_{pred} - площадь m-ой предсказанной области,

S^m_{real} - площадь m-ой реальной области.

Обзор публикаций

1. Подход

Для исследования выбирались статьи по теме, опубликованные в течение 10 последних лет. Для сравнения и отбора алгоритмов, описанных в данных статьях, использовалась метрика MIoU (описанная в п. 5 предыдущей главы), полученная на наборе данных Pascal VOC2012 [2] и отмеченная в таблице [3].

2. Исторический обзор

До того, как главную роль в компьютерном зрении взяло на себя глубокое обучение, для семантической сегментации использовались алгоритмы, основанные на решающих деревьях, такие как TextonForest [4] и RandomForest [5], которые давали $\sim 60\%\ MIoU$.

Позже, в 2014ом году, был предложен подход FCN (Fully Convolutional Networks) [6] с $\sim 67\%\ MIoU$, заключающийся в том, что полносвязные слои в классификационных сетях стали рассматривать как свертки (пример: $y_i = \sum_k x_{i+k} w_k$) с ядрами, которые охватывают их перцептивные области (рис. 3.1).

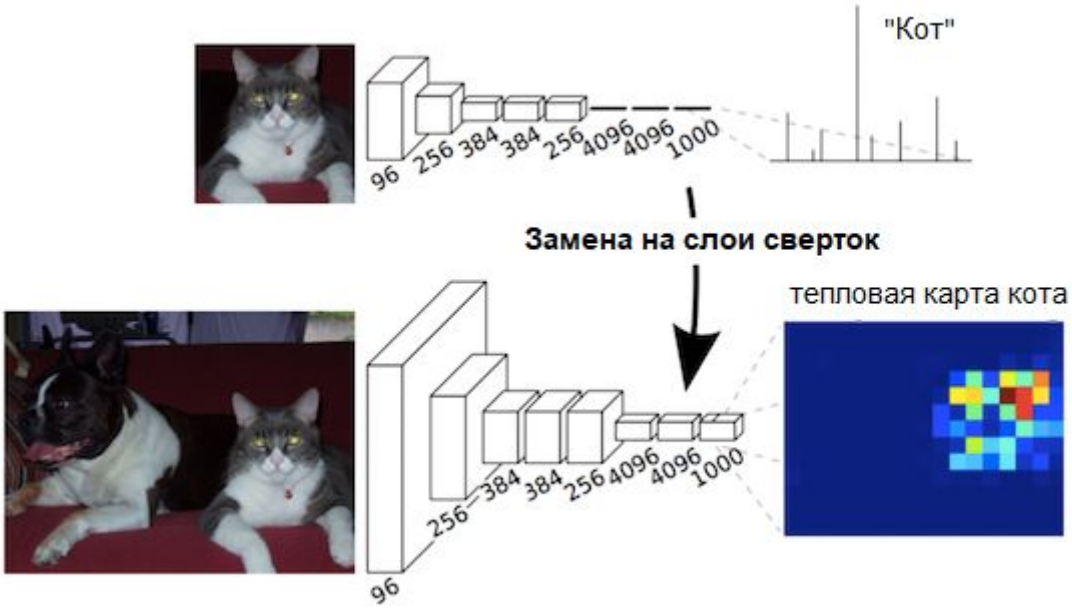

Рис. 3.1. Схема FCN.

Стоит отметить, что при данном подходе на слоях пулинга теряется много информации, а последующее увеличение дискретизации таких данных приводит к достаточно грубой сегментации.

Следующий подход появился в конце 2015 года - Dilated Convolutions [7] (также известный как Atrous Convolutions), он имеет результат $\sim 75\%$ $MIoU$. Его основой является использование слоев расширенной свертки (рис. 3.2) следующего вида: $y_i = \sum_k x_{i+rk} w_k$, где r - шаг между сверточными весами, она позволяет увеличивать перцептивное поле без уменьшения пространственной размерности, которое произошло бы при использовании классической свертки размерности, охватывающей аналогичное поле.

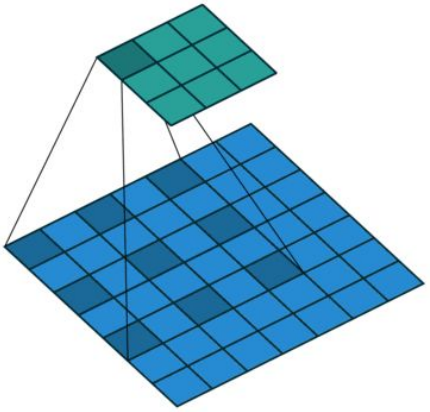

Рис. 3.2. Визуализация применения расширенной свертки.

В данной работе убраны последние два слоя пулинга сети VGG (рис. 3.3) и последующие им слои свертки заменены на слои расширенной свертки.

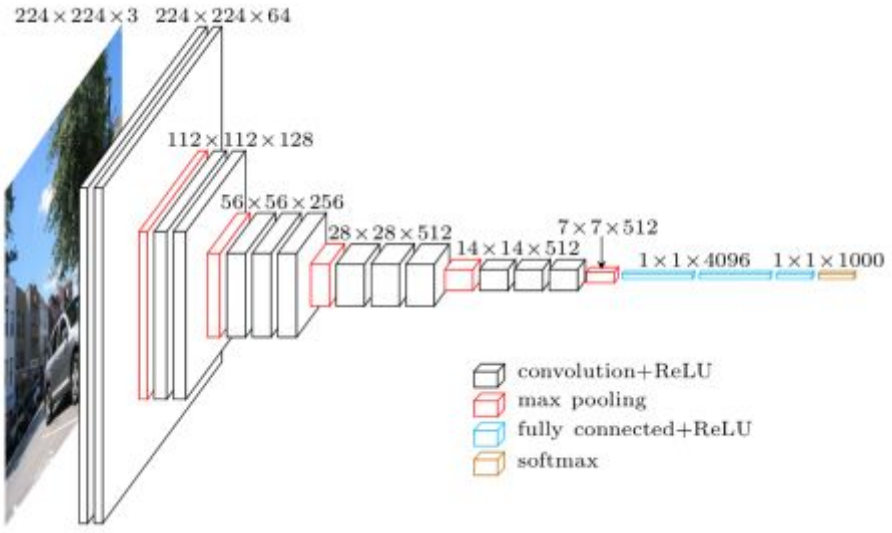

Рис. 3.3. Структура сверточной нейронной сети VGG.

Следующим и одним из последних на данный момент шагом развития подходов к сегментации стал метод DeepLab v3 [8] $\sim 86\%\, MIoU$, опубликованный в июне 2017. Он повторяет предыдущий подход, добавляя к нему ASPP (atrous spatial pyramid pooling), т.е. параллельное использование нескольких расширенных сверток с различными шагами r между сверточными весами, а также последующую конкатенацию параллельных

изображений в одно. Схема работы показана на рисунках 3.4 - 3.5.

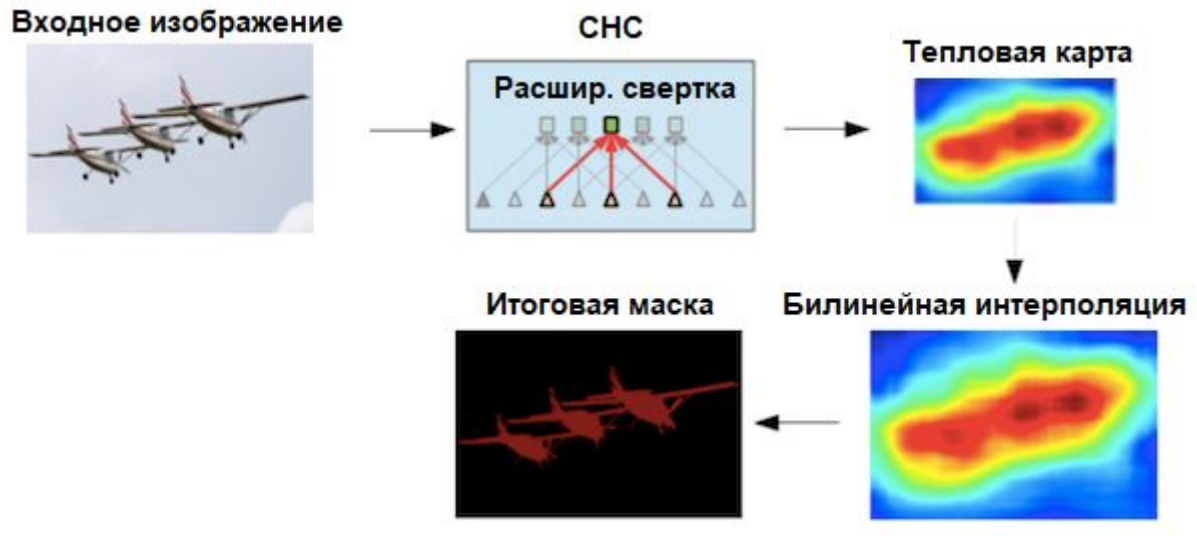

Рис. 3.4. Схема работы алгоритма DeepLab v3.

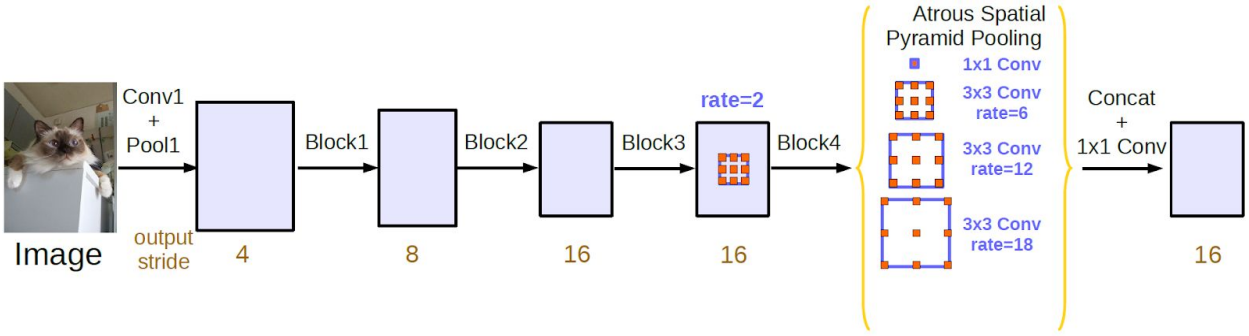

Рис. 3.5. Схема свертки в DeepLab v3.

3. Выводы

Отмечается достаточно последовательное развитие подходов к сегментации, результаты которых также последовательно растут. Для основы последующей работы было решено выбрать алгоритм DeepLab v3, поскольку он включает почти все исторические наработки в этой области за последнее время, а также среди многих сегментационных алгоритмов он имеет одно из самых высоких значений по метрике $MIoU \sim 86\%$, которое было получено

для набора данных Pascal VOC2012 [2-3], включающего в себя 20 различных классов.

Глава 1. Структура сверточной нейронной сети

В публикации [8] предложен алгоритм DeepLab v3, основанный на сверточной нейронной сети ResNet [10], как упоминалось в пункте 2 предыдущей главы, данный подход показывает высокие результаты на наборе данных Pascal VOC2012 [2], состоящем из 21 класса, которые можно разбить на четыре основных надкласса: люди, транспорт, животные, предметы быта. Все они имеют существенное отличие от класса, используемого в настоящей работе, - текста. Отличие заключается в том, что приведенные выше классы в большинстве своем представляют собой достаточно существенные по площади замкнутые области кадра, текст же состоит из отдельных тонких линий. Потому в указанной публикации свертки сети ResNet сконфигурированы так, что разрешение входного изображения отличается от выходного в шестнадцать раз. Такой подход недопустим для сегментации текста, т.к. геометрическая структура символов попросту не сохраняется (что также было подтверждено опытным путем). Учитывая данные отличия, а также то, что символы текста представляют собой более простую геометрическую структуру, нежели классы набора данных [2], было решено составить собственную DeepLab-подобную структуру сети на основе ResNet с ASPP на конце.

1.1. Структура сети для сегментации текста

В качестве исходных положений для составления новой топологии сети были взяты следующие пункты:

1. Отношение разрешения входного изображения к разрешению выходного не должно быть больше 4-5, иначе возникает проблема, описанная в первом абзаце данной главы.
2. Сеть должна умещаться в памяти современных видеокарт, что,

учитывая предыдущий пункт, накладывает ограничения на количество сверток на каждом слое и глубину сети.

3. Символы текста имеют достаточно простую геометрическую структуру.

В качестве СНС было решено оставить ResNet, т.к. каждый блок данной сети (рис. 1.1) содержит два пути прохода изображения: короткий и через несколько последовательных сверток.

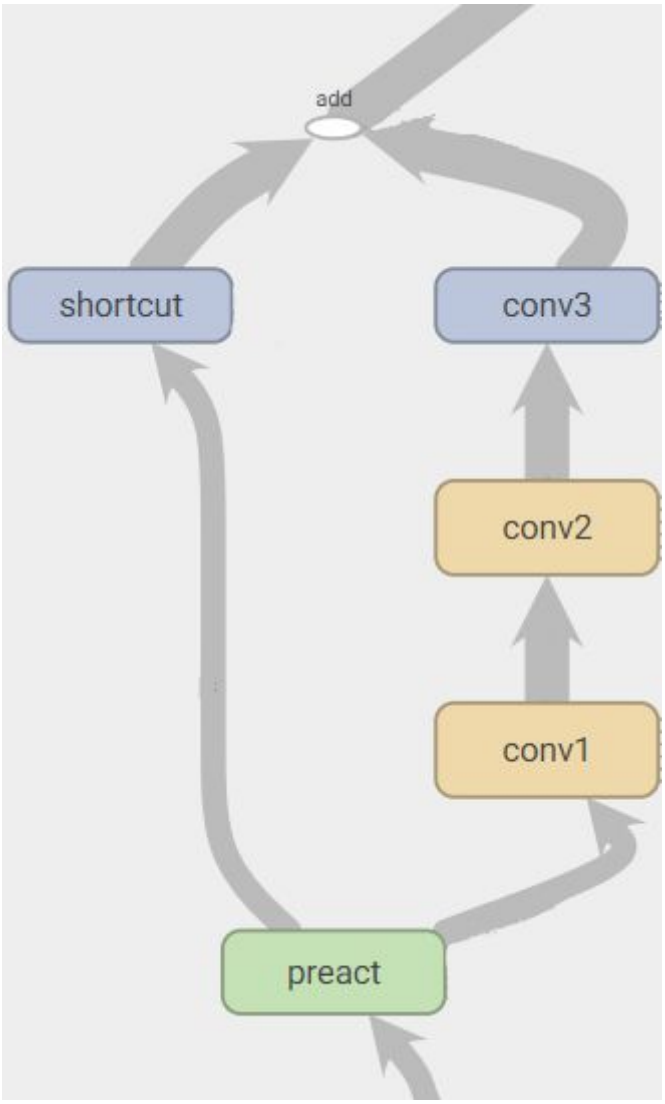

Рис.1.1. Схематичное представление блока сети ResNet (короткий путь слева).

Этот подход позволяет не терять те данные, которые обратились в 0 на свертках. Данный факт полезен для текста, т.к. потери части тонких линий,

из которых состоит текст, могут сказаться на общем качестве сегментации.

Обычно для сегментации сложных объектов многих классов используется несколько блоков структуры, приведенной на рисунке 1.1, но учитывая исходные положения, обозначенные выше, было решено протестировать сеть, состоящую из одного такого блока. Здесь попытка удовлетворения первого и второго пункта ведет к уменьшению числа сверточных слоев, что также оправдывается пунктом 3.

Блок ASPP был составлен из четырех сверток: одна с ядром 1×1, три расширенных с ядром 3×3, схема представлена на рисунке 1.2. Для расширенных сверток в качестве шагов r между сверточными весами были выбраны шаги со значениями: $\{2, 4, 6\}$. В публикации [8] использовались шаги $\{6, 12, 18\}$, трехкратное уменьшение шагов сделано по причине того, что текст состоит из тонких линий, а свертка с ядром 3×3 и шагом 6 покрывает линию шириной в 15 пикселей, что выглядит достаточным для сегментации текста.

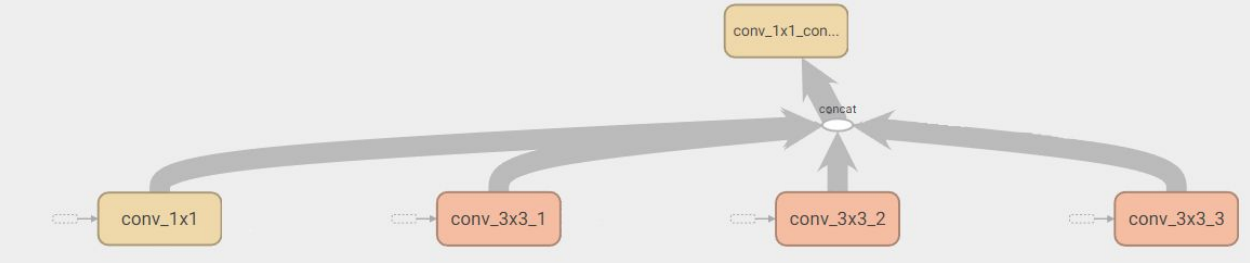

Рис.1.2. Структура ASPP.

Итоговая конфигурация сети подобрана так, чтобы входное изображение сворачивалось всего лишь двукратно, при этом на вход ASPP поступает 2048 вариантов сверток исходного изображения. Исходный код проекта доступен в публичном репозитории [11] на странице автора данной ВКР.

Глава 2. Подготовка данных

Для обучения сети, описанной в главе 1, необходим предварительно размеченный набор изображений (формат разметки описан в главе “постановка задачи”). Есть три возможных варианта его получения:

1. Подбор изображений с текстом, ручная разметка оных с помощью дополнительного ПО.
2. Использование готового набора данных.
3. Автоматическая генерация.

Первый вариант не рассматривался, т.к. сильно затратен по времени и может иметь низкое качество разметки, произведенной вручную. По второму варианту, в силу специфичности задачи, имеется только один публичный набор данных - “Chars74K” [9], но количество и качество изображений, а также аннотаций текста в нем достаточно низкое. Потому было решено автоматически сгенерировать набор изображений с текстом.

2.1. Размерность входных данных

В пункте 1 постановки задачи размерность входного изображения была определена как $X \times Y \times P$, но в реальной жизни все эти три параметра могут меняться, поэтому далее будем считать, что $X = X_{fixed}$, $Y = Y_{fixed}$, $P = P_{fixed}$, т.е. каждое изображение дополнительно предобработано и приведено к некоторым фиксированным значениям пространственных размерностей. Также, возвращаясь к контексту задачи, отметим, что число каналов может быть сведено к $P_{fixed} = 1$ (перевод в градации серого посредством линейной свертки цветовых каналов

$Ch_{grayscale} = \sum_{i=1}^{P} \lambda_i Ch_i, \ \sum_{i=1}^{P} \lambda_i = 1$) без влияния на конечный результат. Это допустимо за счет независимости восприятия текста от цвета его печати (пример - рис. 2.1 а-б).

Рис. 2.1 (а). Цветной текст.

Рис. 2.1 (б). Черно-белый текст.

2.2. Алгоритм генерации изображений

В качестве исходных положений было принято, что текст, содержащийся на изображении, может иметь любой:

1. Фон
2. Размер
3. Шрифт
4. Угол наклона (по любому из углов Эйлера)
5. Цвет
6. Позицию на изображении

В качестве алфавита были взяты русские символы нижнего регистра (а-б), цифры (0-9) и часто используемые знаки препинания ('.', ',', ';', '!', ':', '-'). Отсюда итоговый размер алфавита составил 49 символов. Верхний регистр русского алфавита не использовался, т.к. в большинстве своем он повторяет нижний, только в большем масштабе, а также это вдвое увеличило бы число классов, что, в свою очередь, негативно сказалось бы на результатах.

В качестве источника текста использовалась последовательная случайная выборка символов из выше обозначенного алфавита. Для тестовых изображений был использован словарь существительных русского языка, это сделано в большей мере для проверки сегментации на реальном распределении символов (тут под распределением имеется в виду соседство различных символов), а также из эстетических соображений.

Текст на изображениях печатался с использованием одного из 30 свободных шрифтов формата TrueType, разнообразие шрифтов важно для проверки прикладной применимости алгоритма, т.к. в реальном мире используется

достаточно большое число различных шрифтов.

В качестве фоновых изображений было использовано 5000 изображений различных тематик (природа, 3д графика, портреты, фото), не содержащих текст (рис. 2.2).

Рис. 2.2. Примеры фоновых изображений.

Алгоритм:

1. Фоновое изображение переводится в градации серого.
2. Изображение из п. 1 разбивается на 4 равных прямоугольника, покрывающие всю площадь.
3. В каждом из таких прямоугольников случайным образом выбираются координаты точки, в которой находится левый верхний угол текста.
4. Случайным образом выбирается один из шрифтов.
5. Случайной выборкой из алфавита генерируется строка.
6. Цвет текста выбирается отличным от фона по яркости.
7. Символы печатаются на прозрачном изображении (alpha = 0), равном по разрешению одному прямоугольнику на исходном фоне, также печатается еще на одном на изображении, где вместо значения яркости пикселя записывается класс каждого пикселя - создается маска для

обучения.

8. Случайно выбираются 3 угла Эйлера из диапазона [-20; 20] градусов и изображение из п. 5 трансформируется в соответствии с данными углами.
9. Изображения конкатенируются с фоном с помощью смешивания по альфа-каналу.

Итоговое изображение и маски приведены на рисунках 2.3 а-б.

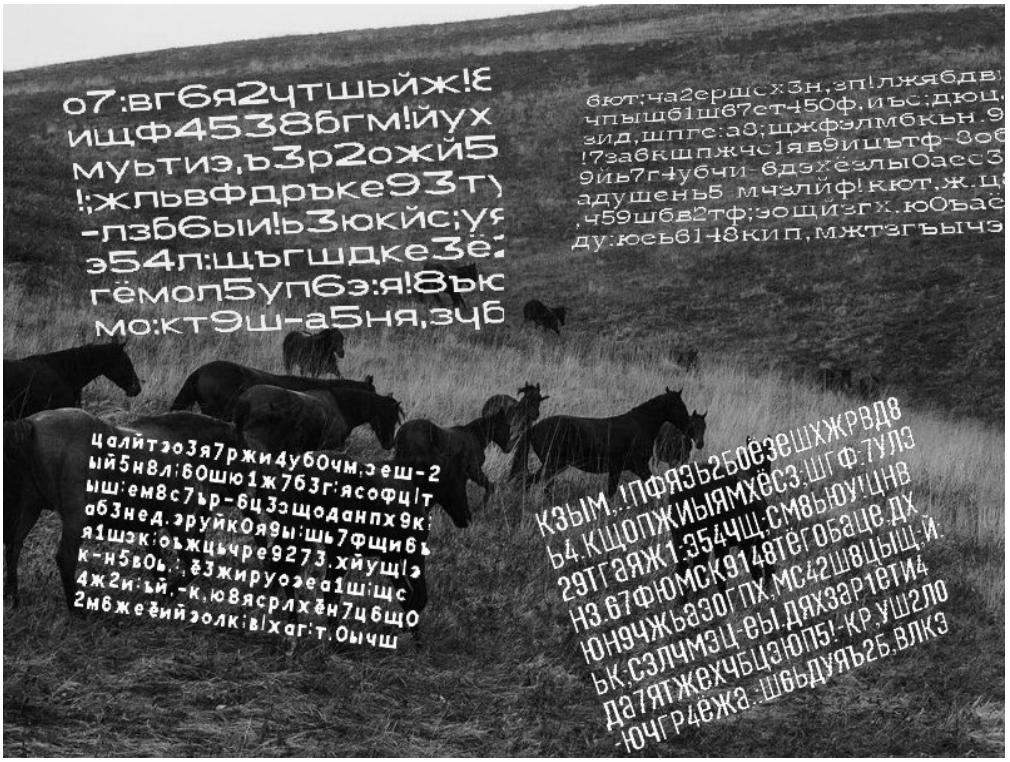

Рис. 2.3 (а). Пример сгенерированного изображения.

Рис. 2.3 (б). Бинарная маска (слева) и многоклассовая маска (справа).

В целях визуализации каждому классу, закрепленному за символами, установлен уникальный цвет (см. рис. 2.4)

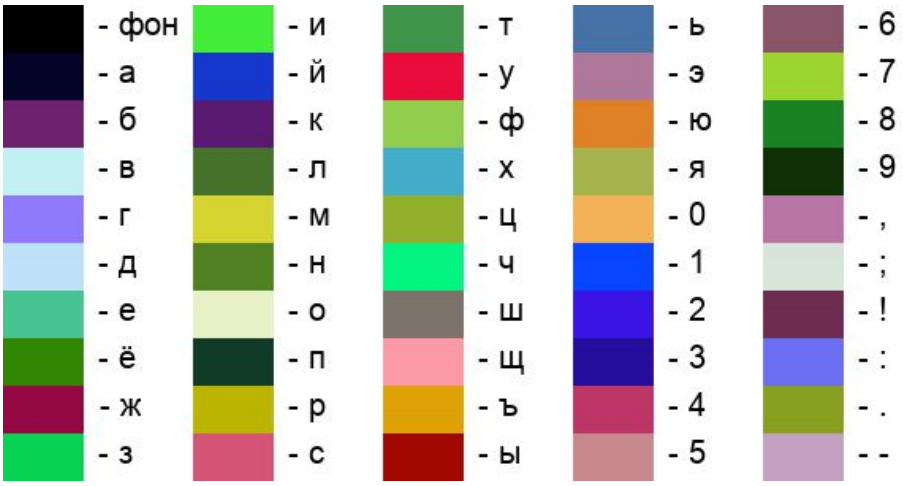

Рис. 2.4. Таблица соответствия цвета символу для многоклассовой маски.

Для бинарной маски фон обозначен черным, а текст белым цветом.

2.3. Предобработка изображений

Как уже отмечалось в пункте 1 данной главы, одним из первых шагов предобработки изображения для задачи сегментации текста является перевод его в градации серого. Кроме этого, существует еще одно свойство - яркость

текста обычно выбирается отличной от яркости фона (рис. 2.5).

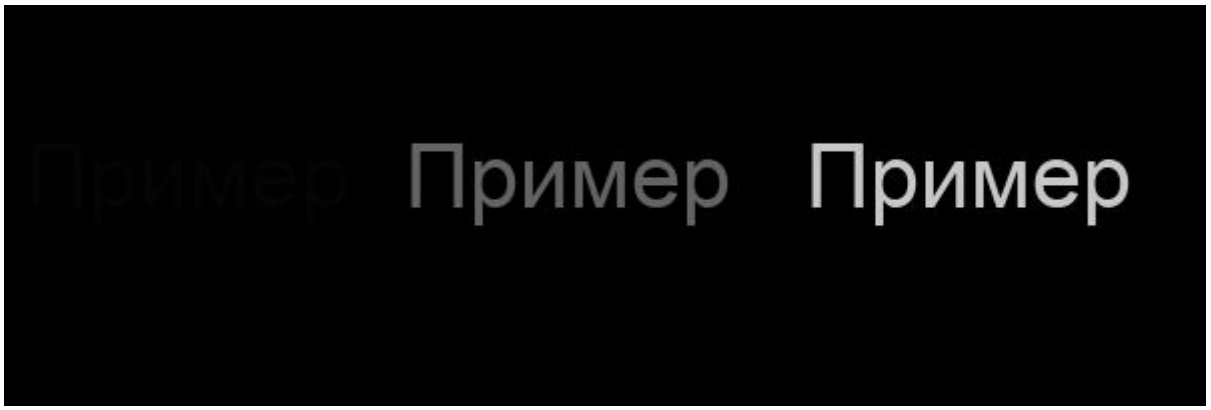

Рис. 2.5. Пример текста различной яркости. Слева яркость минимально отличается от фона, текст почти нечитаем.

Это позволяет воспользоваться детектором границ Кэнни [12], но здесь возникает следующая проблема: маска, содержащая найденные границы, является бинарной (рис 2.6), т.е. состоит только из двух различных значений яркости пикселей, потому все границы, которые будут найдены на фоне за текстом, будут приводить к зашумлению входных данных сети, т.к. никаких отличий по значению пикселей контура буквы и оных на границах фона нет. На рисунке 2.6 такая проблема возникает в нижней части буквы “в”, что зашумляет ее внутреннюю границу.

Чтобы исправить эту проблему было решено применить к маске границ морфологическое расширение с ядром 5×5 и полученную маску поэлементно умножить на исходное изображение (рис. 2.7). Таким образом мы избавляемся от большого процента фона, но при этом оставляем информацию о яркости в окрестности границ, что в свою очередь упрощает задачу отделения границ фона от границ текста.

Данный подход к предобработке упрощает сегментацию класса “фон”, который обычно является разнообразным и оттого требует обучение большого числа сверток для его сегментации. Подход упрощается настолько,

что это достаточно хорошо укладывается с положениями о структуре сети из пункта 1 главы 1, т.к. для выделения фона, состоящего из пикселей нулевой яркости, требуется уже куда меньше вычислений.

Рис. 2.6. Исходное изображение (слева) и маска границ (справа).

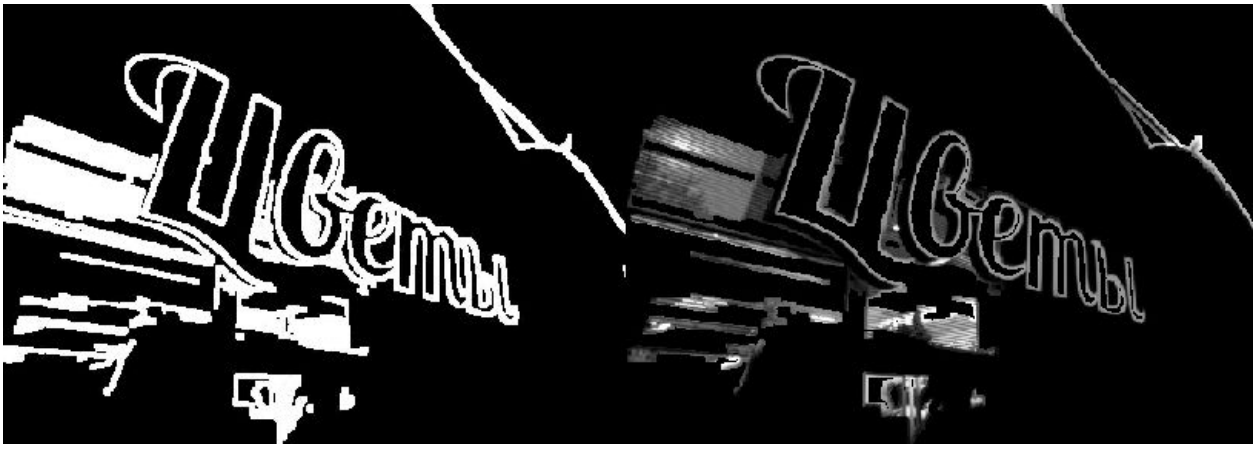

Рис. 2.7. Маска границ после морфологического расширения (слева) и результат поэлементного умножения данной маски на исходное изображение (справа).

Глава 3. Тестирование

Прежде чем переходить непосредственно к тестированию, необходимо отметить, что поставленную в данной работе задачу можно считать успешно решенной в том случае, если предложенный алгоритм сможет выполнять бинарную сегментацию типа текст-фон с точностью $\geq 50\%\, MIoU$, этого достаточно для последующей постобработки и OCR на выделенной области посредствам сторонних алгоритмов. Также будет рассмотрен вариант многоклассовой сегментации, где за каждым символом алфавита, описанного в пункте 2 предыдущей главы, будет закреплен отдельный класс. В теории это позволит более точно провести постобработку области, т.к. зная класс символа мы имеем представление о его геометрической форме, а значит появляется возможность более полно обратить искажения выделенной области, что может быть полезно для последующего уточнения с помощью OCR.

Для тестирования структуры сети, предложенной в главе 1, было сгенерировано 5000 изображений по алгоритму, описанному в пункте 2 предыдущей главы. Набор данных для обучения сети сформирован случайной выборкой 4000 изображений, тестовый набор составлен из оставшейся тысячи изображений. Для каждого из изображений сгенерирована как бинарная, так и многоклассовая маска. В ходе обучения сети точность сегментации будет измеряться по метрике $MIoU$ полученной на тестовом наборе данных.

3.1. Бинарная сегментация

Для обучения СНС необходим подбор нескольких параметров:

- Коэффициент скорости обучения (*lr* от англ. *learning rate*), данный

параметр позволяет управлять величиной изменения весов на каждой итерации обучения, в алгоритме обратного распространения ошибки он вводится как коэффициент при градиенте.

- Количество изображений после обработки которых происходит обратное распространение ошибки по сети.

- Максимальное количество итераций обучения сети или некий критерий останова.

Коэффициент скорости обучения был подобран опытным путем и равен $lr = 5 \cdot 10^{-3}$, больший коэффициент приводил к падению точности, меньший - к увеличению времени обучения, но при этом не добавлял точности.

Количество изображений было ограничено аппаратными ограничениями и потому равнялось единице. При большем числе изображений возникала нехватка памяти на видеокарте с 4гб видеопамяти (Nvidia GeForce GTX 1050 Ti).

В качестве критерия останова было выбрано условие: $MIoU_i - MIoU_{i-10000} \leq 0.01$, где i – номер текущей итерации.

В ходе обучения сети были получены показатели метрики IoU, представленные на рисунке 3.1. Максимальное значение метрики составило: $MIoU = 76,7\%$, $IoU_{\text{текст}} = 58,6\%$, $IoU_{\text{фон}} = 94,8\%$. Что позволяет говорить о том, что предложенная структура сети подходит для решения поставленной задачи.

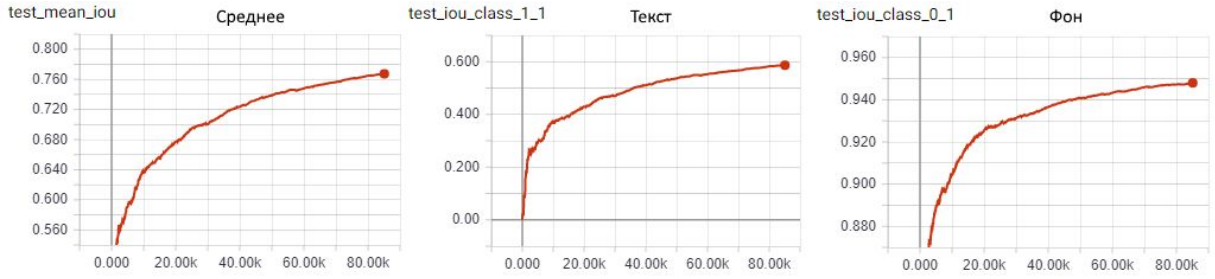

Рис. 3.1. Графики роста метрики IoU в ходе обучения. Слева MIoU, в центре IoU для класса текста, справа IoU для фона.

Примеры бинарной сегментации приведены на рисунках 3.2 - 3.3.

Рис. 3.2. Пример сегментации. Слева направо: исходное изображение, реальная маска, предугаданная маска.

Рис. 3.3. Пример менее качественной сегментации. Слева направо: исходное изображение, реальная маска, предугаданная маска.

3.2. Многоклассовая сегментация

Для обучения СНС на 50 классах (49 символов и фон) был использован набор данных, описанный в предыдущем пункте, за тем исключением, что были

использованы многоклассовые маски.

Эмпирически было установлено, что коэффициент скорости обучения, использованный для бинарной классификации ($lr = 5 \cdot 10^{-3}$), слишком велик. Потому новый коэффициент был взят на порядок меньше: $lr = 5 \cdot 10^{-4}$, что увеличило время обучения, но при этом рост метрики $MIoU$ был более устойчивым. Остальные параметры обучения не изменялись.

В ходе обучения сети были получены показатели метрики $MIoU$, представленные на рисунке 3.4. Максимальное значение метрики составило: $MIoU = 47{,}6\%$. Поклассовые графики значений IoU не приводятся в силу их большого количества.

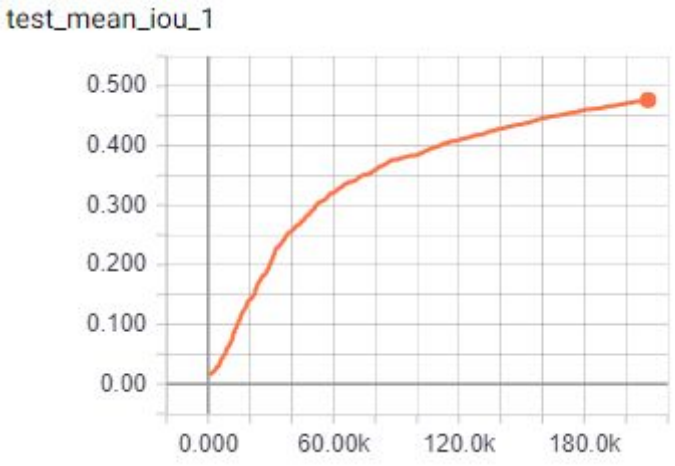

Рис. 3.4. График роста метрики MIoU в ходе обучения.

Примеры сегментации приведены на рисунках 3.5 - 3.6.

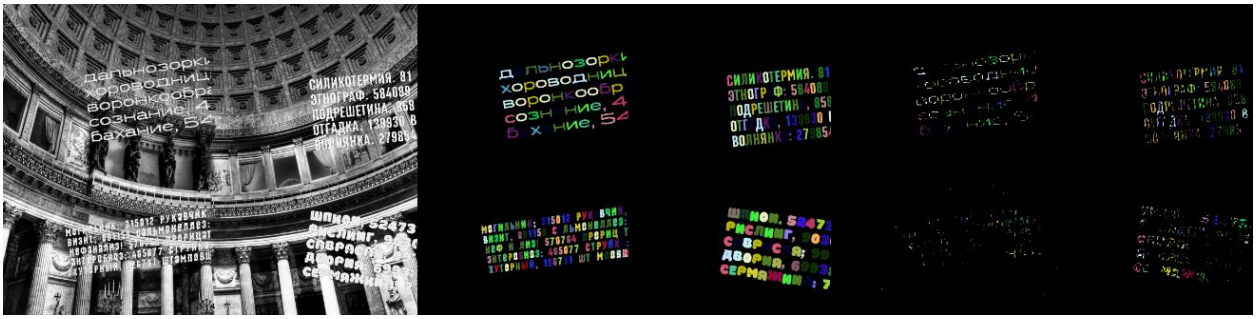

Рис. 3.5. Пример сегментации. Слева направо: исходное изображение, реальная

маска, предугаданная маска.

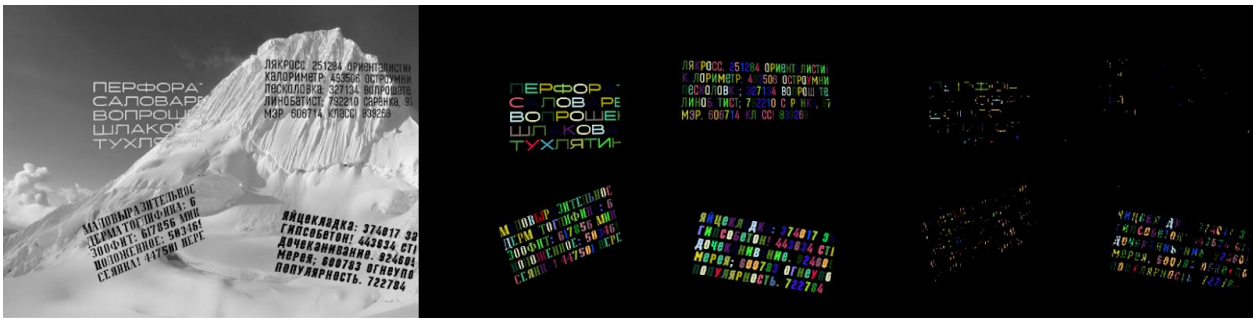

Рис. 3.6. Пример менее качественной сегментации. Слева направо: исходное изображение, реальная маска, предугаданная маска.

Примеры в более высоком разрешении приведены в приложении к ВКР.

3.3. Выводы

Предложенная структура сети позволяет производить бинарную сегментацию текста, при этом, в случае многоклассовой сегментации, точность сильно падает. Оба подхода можно комбинировать и использовать параллельно для уточнения результатов друг друга, в таком случае имеется знание о достаточно точной маске текста, а также о классах некоторых символов внутри данной маски. Такой подход может позволить узнать как искажен текст (например, в плане поворота) и обратить эти искажения, а итоговую выровненную маску передать программе классического OCR для распознавания на ней символов.

Глава 4. Заключение

Задача, поставленная в рамках данной выпускной квалификационной работы, состояла в создании алгоритма сегментации текста на сложном фоне. Для этого подразумевалось решение сразу нескольких задач: создание алгоритма сегментации на базе имеющихся наработок в данной области, исследование возможных путей предобработки изображений, содержащих текст, а также создание набора данных для тестирования и обучения. В ходе ее решения были получены следующие результаты:

1. Найден актуальный алгоритм сегментации на основе которого предложена структура сверточной нейронной сети для сегментации текста.
2. Проведено исследование свойств печатного текста, результатом которого стал метод предобработки, позволяющий существенно снизить число число входных данных, а также обнулить большую нетекстовую часть изображения.
3. Предложен алгоритм генерации набора данных для обучения.
4. Проведено тестирование предложенной структуры сети для бинарной и многоклассовой сегментации, а также подтверждена ее работоспособность.

Исходя из всего вышеперечисленного, можно утверждать, что поставленная задача была решена полностью.

Список литературы

[1] Оптическое распознавание символов. URL: https://ru.wikipedia.org/wiki/Оптическое_распознавание_символов

[2] Набор данных Pascal VOC2012. URL: http://host.robots.ox.ac.uk/pascal/VOC/voc2012/

[3] Pascal VOC2012 Leaderboard. URL: http://host.robots.ox.ac.uk:8080/leaderboard/displaylb.php?cls=mean&challengeid=11&compid=6&submid=12345

[4] Semantic Texton Forests for Image Categorization and Segmentation. Jamie Shotton, Matthew Johnson, Roberto Cipolla. 2008. URL: http://mi.eng.cam.ac.uk/~cipolla/publications/inproceedings/2008-CVPR-semantic-texton-forests.pdf

[5] Real-Time Human Pose Recognition in Parts from Single Depth Images. Jamie Shotton, Andrew Fitzgibbon, Mat Cook, Toby Sharp, Mark Finocchio, Richard Moore, Alex Kipman, Andrew Blake. 2011. URL: http://www.cse.chalmers.se/edu/year/2011/course/TDA361/Advanced%20Computer%20Graphics/BodyPartRecognition.pdf

[6] Fully Convolutional Networks for Semantic Segmentation. Jonathan Long, Evan Shelhamer, Trevor Darrell. 2014. URL: https://arxiv.org/abs/1411.4038

[7] Multi-Scale Context Aggregation by Dilated Convolutions. Fisher Yu, Vladlen Koltun. 2015. URL: https://arxiv.org/abs/1511.07122

[8] Rethinking Atrous Convolution for Semantic Image Segmentation. Liang-Chieh Chen, George Papandreou, Florian Schroff, Hartwig Adam. 2017. URL: https://arxiv.org/abs/1706.05587

[9] Набор данных “Chars74k”. URL: http://www.ee.surrey.ac.uk/CVSSP/demos/chars74k

[10] Deep Residual Learning for Image Recognition. Kaiming He, Xiangyu Zhang,

Shaoqing Ren, Jian Sun. 2015. URL: https://arxiv.org/abs/1512.03385

[11] Репозиторий с исходным кодом. URL:

https://github.com/AlexEbral/text-seg

[12] A Computational Approach to Edge Detection. John Canny. 1986. URL:

https://perso.limsi.fr/vezien/PAPIERS_ACS/canny1986.pdf

Приложение

Здесь приведены результаты сегментации для рисунка 1.

Рис. 1. Исходное изображение.

Пример бинарной сегментации для изображения приведенного на рисунке 1:

Рис. 2. Реальная маска.

Рис. 3. Предугаданная маска.

Пример многоклассовой сегментации для изображения приведенного на рисунке 1:

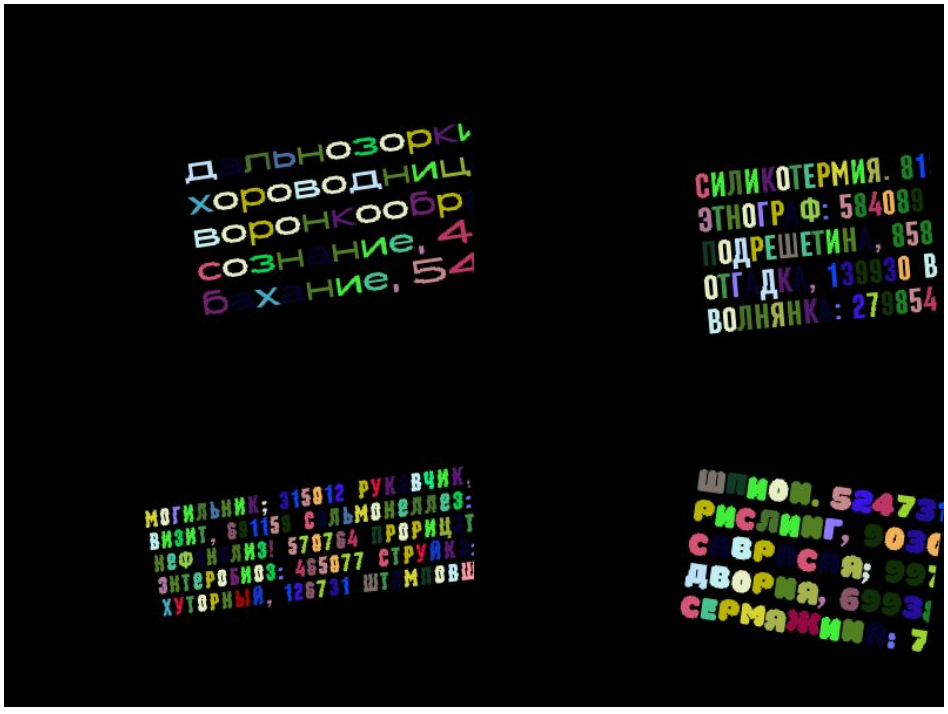

Рис. 4. Реальная маска.

Рис. 5. Предугаданная маска.