Московский государственный университет имени М.В. Ломоносова

Факультет вычислительной математики и кибернетики

Кафедра математической физики

Загайнов Сергей Дмитриевич

Выделение автомобилей на изображениях с использованием слабой разметки

КУРСОВАЯ РАБОТА

Научный руководитель:

к.ф-м.н., н.с

А.В.Хвостиков

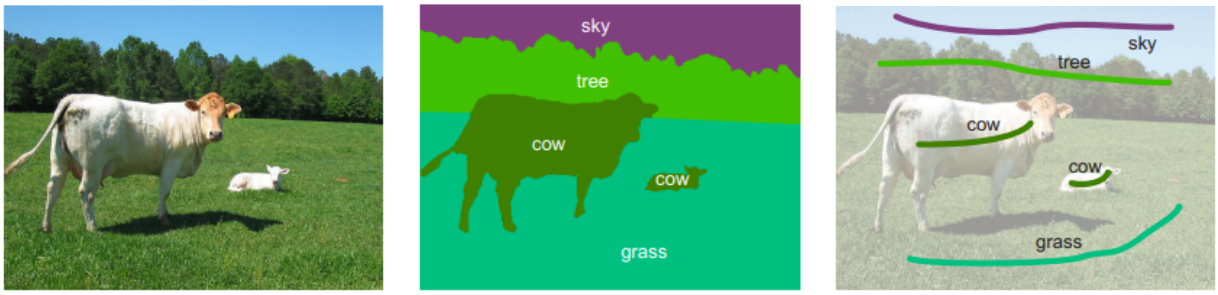
Москва, 2023

Оглавление

1. Введение

Для обученя нейронных сетей требуется большое количество данных, предварительно обработанных специалистами, например, для задачи сегментации объектов на изображении необходимо наличие референсной маски (англ. grount truth mask) для каждого изображения, на которых и будет обучаться нейронная сеть.

Но разметка данных специалистами требует больших либо денежных, либо временных затрат, поэтому для разметки данных можно использовать слабую разметку, так как она занимает гораздо меньше времени для получения. Существует большое количество различных вариантов слабой разметки, большая их часть представлена в статье [1]. В данной работе использовалась разметка штрихами (англ. scribble supervise) её отличие от полной разметки можно увидеть на Рис. 1.



(a) Исходное изображение (b) Полная Разметка (c) Разметка штрихами

Рис. 1: Разница между полной и слабой раметками.

В данной работе предлагается метод дообучения нейронной сети на слабо размеченных данных для решения задачи отделения автомобиля от фона. Эта задача важна, так как она может являться первым шагм для решения более сложных задач, чтобы рассматривать только часть изображения с авомобилем. Примером более сложной задачи является поиск повреждений на автомобиле, которая является важной для каршеринговых команий, чтобы облегчить работу людей, которые этим занимаются. Предлагаемый подход является общим и потенциально может быть применён для произвольной свёрточной нейронной сети для сегментации изображений.

2. Цель работы

Целью работы является разработка метода дообучения сегментирующей нейронной сети на слабо размеченных данных (использовалась разметка штрихами), для улучшения качества её работы на новых данных, отличающихся от тех, на которых она обучалась.

Задача состоит из следующих подзадач:

1. Обучение нейронной сети на относительно большом открытом набре данных.
2. Сбор нового набора изображений и его разметка штрихами для дообучения нейронной сети.
3. Реализация итеративного алгоритма дообучения нейронной сети (на каждой итерации происходит дообучение нейронной сети на новых масках).
4. Анализ полученных результатов.

3. Описание модельной задачи

В этом разделе даётся описание архитектуры свёрточной нейронной сети для сегментации и набора данных, на котором нейронная сеть была обучена, дообучена и протестирована.

3.1. Архитектура нейронной сети

В работе была обучена U-net подобная [2] свёрточная нейронная сеть для сегментации изображений. На Рис. 2 приведена архитектура этой нейронной сети. Она состоит из сужающегося пути (слева) и расширяющегося пути (справа). Сужающийся путь состоит из повторного применения двух сверток 3x3, за которыми следуют функция активации ReLU и операция Max Pooling для понижения разрешения в два раза. Каждый шаг в расширяющемся пути состоит из операции повышающей дискретизации карты признаков, за которой следуют: свертка 2x2, которая уменьшает количество каналов карты признаков; объединение с соответствующим образом обрезанной картой признаков из сужающегося пути; две 3x3 свертки, за которыми следует ReLU. На последнем слое используется свёртка 1x1 для сопоставления каждой 64-канальной карты признаков с желаемым количеством классов.

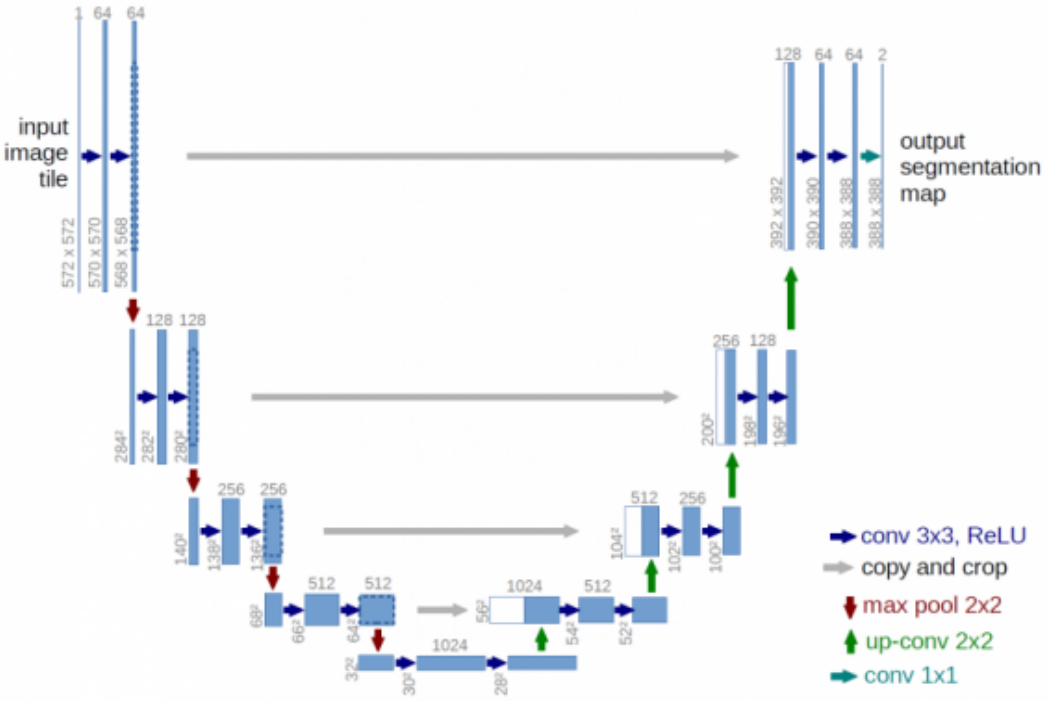


Рис. 2: Архитектура U-net. Каждый синий прямоугольник соответствует многоканальной карте признаков. Количество каналов приведено в верхней части квадрата. Размер XxY приведен в нижнем левом краю прямоугольника. Белые прямоугольники представляют собой копии карты признаков. Стрелки обозначают различные операции.

3.2. Используемые наборы данных

Для первичного обучения нейронной сети использовался набор данных Carvana Image Masking Challenge [1], состоящий из 5088 изображений с полной разметкой и 100064 изображений без разметки размера 1280 x 1918 пикселей. Пример можно увидеть на рис. 3

(a) Исходное изображение (b) Референсная маска сегментации

Рис. 3: Исходное изображение и его референсная маска сегментации из набора Carvana.

Для дообучения нейронной сети использовались изображения реальных автомобилей, сфотографированных на мобильный телефон, всего было сфотографировано 110 автомобилей, 100 изображений использовались для дообучения и 10 для тестирования. Исходный размер изображений 4032 x 3024 пикселей, для ускорения работы алгоритма изображения были приведены к размеру 1024 x 768 пикселей. Далее для изображений, используемых для дообучения, была получена разметка штрихами. Для изображений, используемых для тестирования, была получена полная разметка, причём из-за того, что разметка производилась вручную, на ней есть неточности. Изображения размечались с помощью сервиса supervisely.com. Пример можно увидеть на рис. 4 и рис. 5

3.3. Обучение нейронной сети

Нейронная сеть была обучена на наборе данных Carvana Image Masking Challenge. Обучающая выборка была разбита на тренировочную выборку, состоящую из 5037 изображений, и валидационную выборку, состоящую из 51 изображения. Для ускорения обучения изображения были приведены к размеру 256 x 384 пикселя. Для обучения использовался оптимизатор Adam[2] со скоростью обучения (англ. learning rate) = 10^{-3} и $L_{CCE} + 0.3 * L_{Dice}$ в качестве функции потерь, где L_{CCE} - это категоральная кросс-энтропия (англ. Categorical Cross-Entropy) и L_{Dice} - функция потерь Сёренсена (англ. dice loss).

[1] https://www.kaggle.com/c/carvana-image-masking-challenge

(a) Исходное изображение

(b) Разметка штрихами: жёлтый обозначает автомобиль, зелёный обозначет фон

Рис. 4: Изображение и его разметка штрихами для дообучения.

$$L_{CCE} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c})$$

Здесь: M - количество классов; y - индикатор того, что метка класса c верна для наблюдения o; p - предсказанная вероятность того, что наблюдение o принадлежит классу c.

$$S_{Dice} = \frac{2|X \cap Y|}{|X| + |Y|}$$

$$L_{Dice} = 1 - S_{Dice}$$

Здесь: X - истинная маска; Y - предсказанная маска.

Обучение длилось 5 эпох, размер обучающего пакета 32, размер валидационного пакета 8. График зависимости функции потерь от номера эпохи можно увидеть на рис. **??**. Максимальный достигнутый S_{Dice} - это коэффеициент Сёренсена (англ. Dice score) на валидации 0.9889. Пример предсказания нейронной сети можно увидеть на рис. 6.

(a) Исходное изображение

(b) Полная разметка: жёлтый обозначает автомобиль, зелёный обозначет фон, фиолетовый обозначет неточности разметки (не присвоен никакой класс)

Рис. 5: Изображение и его полная разметка для тестирования.

(a) Исходное изображение

(b) Исходное изображение

(c) Предсказание нейронной сети: жёлтый обозначает автомобиль, фиолетовый обозначает фон

(d) Предсказание нейронной сети: жёлтый обозначает автомобиль, фиолетовый обозначает фон

Рис. 6: Результаты работы нейронной сети, обученной на наборе данных Carvana.

4. Метод дообучения нейронной сети

В данной работе предлагается метод дообучения предварительно обученной свёрточной нейронной сети с использованием разметки штрихами[3].

Предлагаемый подход состоит из следующих шагов:

1. Предобрабока каждого изображения из дообучающей выборки:
   1.1 Кластеризация изображений на суперпиксели.
   1.2 Нахождение текстурной и цветовой гистограмм для каждого суперпикселя.
   1.3 Поиск соседей для каждого суперпикселя (соседи - это граничащие суперпиксели).
   1.4 Поиск суперпикселей, которые пересекаются с разметкой штрихами.
2. Применение алгорима разреза графа (англ. graph cut) для получения масок для дообучения (после нулевой итерации с учётом предсказаний нейронной сети, нулевая итерация - это самое первое применение шагов 2. и 3.).
3. Дообучение нейронной сети на полученных масках.
4. Переход на шаг 2. останавливаемся на третей итерации.

4.1. Предобрабока каждого изображения из дообучающей выборки

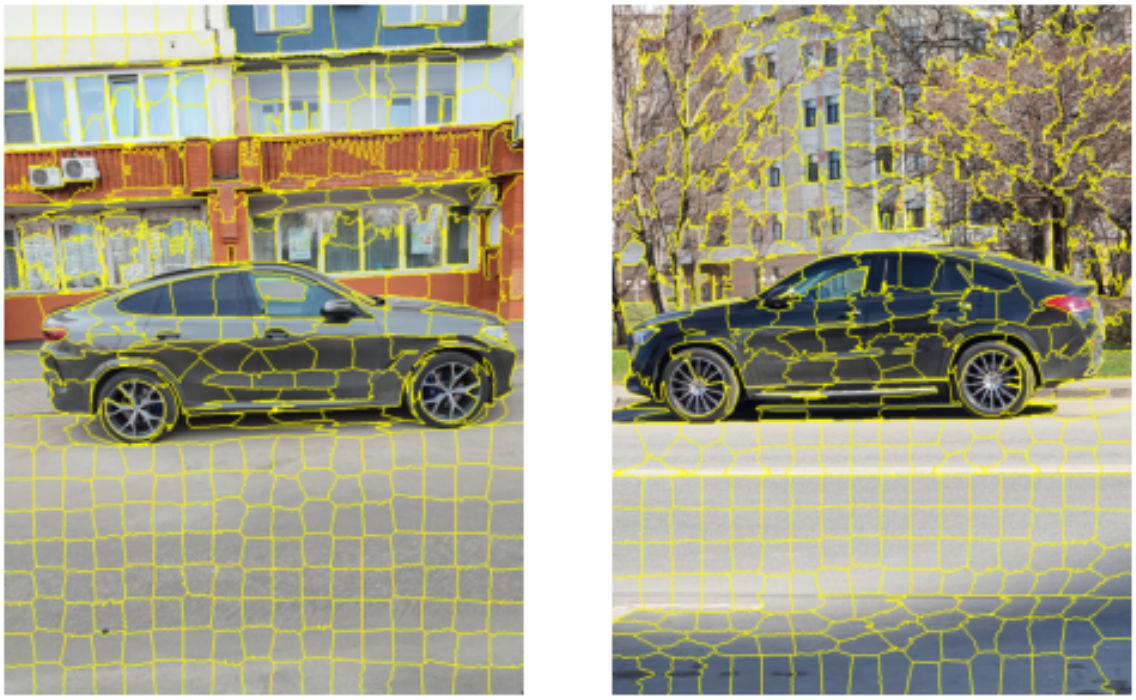
4.1.1. Кластеризация изображений на суперпиксели

Суперпиксели - это пиксели объединённе в группы на основании схожести характеристик (например, интенсивность). Суперпиксели могут быть полезны из-за следующих свойств:

1. Суперпиксели разбивают всё изображение на непересекающиеся множества и в объединении дают всё изображение.
2. Суперпиксели несут больше информации чем пиксели, так как являются объединением нескольних пикселей.
3. Суперпиксели легко интерпретируемы, так как принадлежащие им пиксели имеют похожие визуальные характериситки.

4. Суперпиксели предоставляют удобное и компактное представление изображений, что может быть полезно в вычислительно затратных задачах.

Существует большое число алгоритмов для поиска суперпикселй [4]. В данной работе используется алгоритм SLIC (Simple Linear Iterative Clustering) [5]. Пример работы алгоритма SLIC, реализованного в библиотеке skimage[6] можно увидеть на рис. 7.



(a) Пример 1 (b) Пример 2

Рис. 7: Суперпиксели, полученные с помощью алгоритма SLIC, границы суперпикселей выделены жёлтым.

4.1.2. Нахождение текстурной и цветовой гистограмм для каждого суперпикселя

Для получения масок понадобится сравнивать похожесть соседних суперпикселей в функции похожести (3), а для этого по каждому суперпикселю строятся нормированные (площадь под графиком равна единице) текстурная и цеветовая гистограммы, интерпретируемые как вектор признаков. Далее эти гистограммы будут выступать в качестве аргументов функции похожести, их достаточно построить один раз.

Цветовая гистограмма строится для RGB изображений, то есть для каждого канала строится своя гистограмма интенсивностей, используя 25 бинов, далее эти гистограммы объединяются и нормируются.

Текстурная гисторамма строится для RGB изображений, то есть для каждого канала строятся свои гистограммы градиентов в вертикальном и горизонтальном направлениях, используя 10 бинов для каждого направления, далее эти шесть гистограмм объединяются и нормируются.

4.1.3. Поиск соседей для каждого суперпикселя

Назовём соседями суперпикселей граничащие с ним суперпиксели. Для каждого суперпикселя нужно найти соседей, так как похожесть вычисляется только между суперпикселем и его соседями. В данной работе это самая медленная часть метода. Поэтому реализация алгоритма SLIC из skimage[6] была выбрана неслучайно, один из параметров алогоритма позволяет задать верхнюю оценку на количество суперпикселей, что позволяет контролировать время поиска соседей.

4.1.4. Поиск суперпикселей, которые пересекаются с разметкой штрихами

Последним этапом предобработки дообучающей выборки является поиск суперпикселей, которые пересекаются с разметкой штрихами. То есть мы получаем инфрмацию о том, к какому объекту (фону или автомобилю) относить размеченные суперпиксели.

4.2. Применение алгорима разреза графа (англ. graph cut) для получения масок для дообучения

Введём обозначения: X - изображение из дообучающей выборки; $\{x_i\}$ - множество непересекающихся суперпикселей изображения X, дающих в объединении всё X; C - это количество размеченных классов в разметке штрихами (в данной работе их было два: автомобиль и фон); s_k - это множество пикселей, размеченных в разметке штрихами одним цветом (то есть отнесенных к одному классу); S - это множество s_k; y_i - это метка i-ого суперпикселя; θ - это веса нейронной сети.

Для того, чтобы получить из разметки штрихами маски для дообучения нейронной сети, в данной работе используется алгоритм разреза графа [7] из библиотеки PyMaxflow. А именно разрез графа использовался для минимизации функционала энергии [8] вида:

$$\sum_i \psi_i^{scr}(y_i|X,S) + \sum_i \psi_i^{net}(y_i) + \sum_{i,j} \psi_{ij}(y_i,y_j|X) \tag{1}$$

Суммирование идёт по всем суперписелям изображения.

$$\psi_i^{scr}(y_i) = \begin{cases} 0 & \text{if } y_i = c_k \text{ and } x_i \bigcap s_k \neq \varnothing \\ -\log\left(\frac{1}{C}\right) & \text{if } x_i \bigcap S = \varnothing \\ \infty & \text{otherwise} \end{cases} \tag{2}$$

Здесь первое и третье условия значят, что если суперпиксель x_i пересекается с s_k, то он имеет нулевой вес, если относится к k-тому классу, и бесконечно большой вес, если - к любому другому. Второе условие значит, что если суперпиксель x_i не пересекается с S (то есть мы не знаем к какому классу относится суперпиксель), то ему может быть присвоена любая метка с равной вероятностью.

$$\psi_{ij}(y_i, y_j|X) = [y_i \neq y_j] \exp\left\{ -\frac{\|h_c(x_i) - h_c(x_j)\|_2^2}{\delta_c^2} - \frac{\|h_t(x_i) - h_t(x_j)\|_2^2}{\delta_t^2} \right\} \tag{3}$$

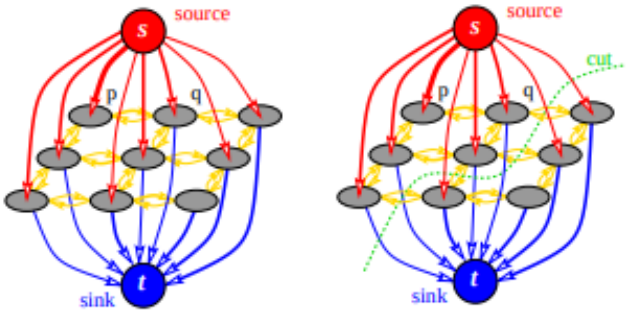
Здесь $[\cdot]$ - это функция-индикатор; параметры δ_c и δ_t равны соответственно 25 и 10; h_c и h_t - это соответственно цветовая и текстурная гистограммы (полученные ранее), интерпретируемые как векторы признаков. Это слагаемое отвечает за похожесть двух соседних суперпикселей, оно тем меньше, чем слабее отличаются суперпиксели.

$$\psi_i^{net}(y_i) = -\log P(y_i|X, \theta) \tag{4}$$

$\log P(y_i|X, \theta)$ означет логарифм вероятности того, что суперпиксель x_i будет иметь метку y_i, иными словами, это попиксельная сумма логарифмов вероятности того, что пиксель внутри суперпикселя x_i имеет метку y_i, то есть это просто сумма логарифмов вероятности предсказаний нейронной сети, которая считается для пикселей внутри суперпикселя x_i.

Одним из вариантов минимизации функционала (1) является применение алгоритма разреза графа, но чтобы его применить нужно инициализировать граф $G = (V, E)$. Опишем построение графа G для решаемой задачи (для двух классов). Введём обозначения. $V(G)$ - множество вершин, вершины могут быть терминальные и нетерминальные. К терминальным вершинам относятся две вершины, называемые исток (англ. source) (обозначает фон) и сток (англ. sink) (обозначает объект, в данном случае автомобиль). К нетерминальным вершинам относятся все суперпиксели изображения. E - это множество рёбер, рёбра также могут быть терминальные и нетерминальные. Терминальные рёбра - это рёбра, проводящиеся от истока ко всем нетерминальным вершинам, с весами $\psi_i^{scr}(y_0|X, S) + \psi_i^{net}(y_0)$, если предсказания нейронной сети учитываются, и с весами $\psi_i^{scr}(y_0|X, S)$, если предсказания нейрон-

ной сети не учитываются, а также рёбра, проводящиеся от всех нетермнальных вершин к стоку, с весами $\psi_i^{scr}(y_0|X,S) + \psi_i^{net}(y_1)$, если предсказания нейронной сети учитываются, и с весами $\psi_i^{scr}(y_1|X,S)$, если предсказания нейронной сети не учитываются. Нетерминальные рёбра - это рёбра, проводящиеся между соседними суперпикселями, с весами равными $\psi_{ij}(y_i, y_j|X)$. После применения алгоритма разреза графа к полученному графу он разобъётся на два графа, один из которых будет содержать исток и нетерминальные вершины, а второй - сток и оставшиеся нетерминальные вершины (множества нетерминальных вершин этих графов непересекаются). И теперь те нетерминальные вершины (суперпиксели), которые относятся к графу истока, мы считаем фоном, а те, которые относятся к графу стока, - автомобилем. Пример графа и его разреза можно увидеть на рис. 8.



(a) Граф G (b) Разрез графа G

Рис. 8: Граф и его разрез.

4.3. Дообучение нейронной сети на полученных масках

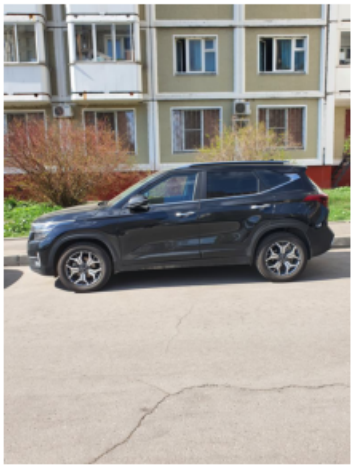
Дообучение присходит итерационно. На нулевой итерации с помощью разреза графа получаются маски без учёта предсказания нейронной сети, то есть решается задача минимизации функционала (1), без слагаемого $\psi_i^{net}(y_i)$. После получения масок для дообучающей выборки, выполняется дообучение нейронной сети (изменение весов нейронной сети θ со скоростью обучения = 10^{-5}). После нулевой итерации с помощью разреза графа получаются маски с учётом предсказания нейронной сети, то есть решается задача минимизации функциионала (1). После получения масок для дообучающей выборки, производится дообучение нейронной сети (изменение весов нейронной сети θ со скоростью обучения = 10^{-5}). В данной работе выполнялось три итерации алгоритма, так так в ходе экспериментов было выяснено, что далее наблюдаются незначительные улучшения.

Смысл итеративности в том, что на каждой итерации улучшается точность предсказаний нейронной сети, поэтому разрез графа, учитывающий эти предсказания, на каждой

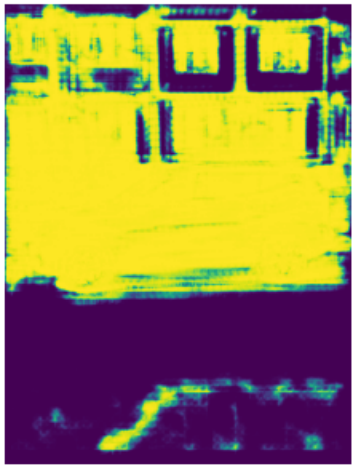
следующей итерации даёт более точные маски для дообучения, что и позволяет улучшить предсказания нейронной сети после дообучения.

5. Результаты

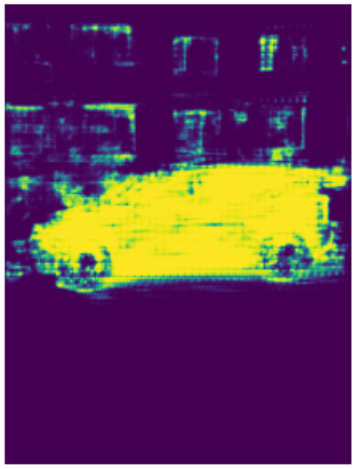
На рис. 9 приведены приведены результаты работы алгоритма алгоритма для не дообученной нейронной сети и для нейронной сети после каждой итерации алгоритма.



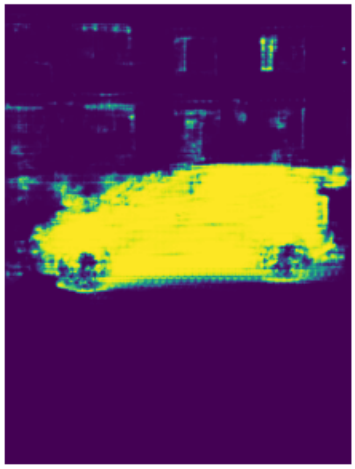
(a) Исходное изображение



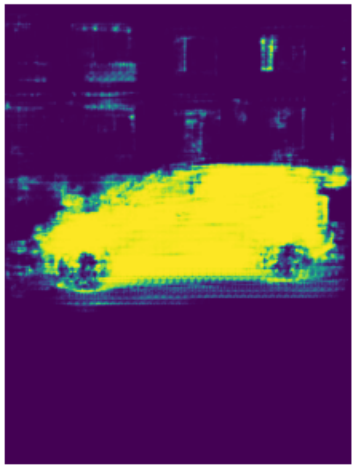
(b) Предсказание не дообученной нейронной сети



(c) Предсказание дообученной нейронной сети после 0-ой итерации



(d) Предсказание дообученной нейронной сети после 1-ой итерации



(e) Предсказание дообученной нейронной сети после 2-ой итерации

Рис. 9: Результаты работы алгоритма. На всех изображениях с результатами работы нейронной сети жёлтый обозначет автомобиль, фиолетовый обозначет фон.

Значения метрик, посчитанных на тестовой выборке, которая была собрана и размечена собственноручно, приведены в таблице 1. Видим, что значения метрик увеличиваются на каждой итерации.

	Мера Сёренсена (Dice score)	IOU
Не дообученная нейронная сеть	0.53	0.37
Дообученная нейронная сеть после 0-ой итерации	0.83	0.72
Дообученная нейронная сеть после 1-ой итерации	0.85	0.75
Дообученная нейронная сеть после 2-ой итерации	0.86	0.76

Таблица 1: Значения метрик.

6. Заключение

Была обучена сегментирующая U-net подобная нейронная сеть на наборе данных Carvana Image Masking Challenge. Собран и размечен набор данных, состоящий из 110 изображений автомобилей с улиц Москвы. Разработан итерационный алгоритм дообучения свёрточной нейронной сети на 100 изображениях из собранного набора данных со слабой разметкой (штрихами). Метод был протестирован на тестовой выборке, состоящей из 10 изображений автомобилей.

6.1. Программная реализация

Предлагаемый метод был программно реализован на языке Python с использованием следующих библиотек numpy, tensorflow, sklearn, PyMaxflow, skimage.

6.2. Дальнейшее развитие

Дальнейшим развитием работы может стать ускорение алгоритма (ускорение поиска соседей суперпикселей) и обобщение метода для работы более чем с двумя классами.

Список литературы

[1] A Survey on Label-efficient Deep Segmentation: Bridging the Gap between Weak Supervision and Dense Prediction / Shen W., Peng Z., Wang X., Wang H., Cen J., Jiang D., Xie L., Yang X., and Tian Q. // arXiv preprint arXiv:2207.01223. — 2022. — P. 1–22.

[2] Ronneberger O., Fischer P., Brox T. U-net: Convolutional networks for biomedical image segmentation // Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 / Springer. — 2015. — P. 234–241.

[3] Scribblesup: Scribble-supervised convolutional networks for semantic segmentation / Lin D., Dai J., Jia J., He K., and Sun J. // Proceedings of the IEEE conference on computer vision and pattern recognition. — 2016. — P. 3159–3167.

[4] Stutz D., Hermans A., Leibe B. Superpixels: An evaluation of the state-of-the-art // Computer Vision and Image Understanding. — 2018. — Vol. 166. — P. 1–27.

[5] SLIC superpixels compared to state-of-the-art superpixel methods / Achanta R., Shaji A., Smith K., Lucchi A., Fua P., and Süsstrunk S. // IEEE transactions on pattern analysis and machine intelligence. — 2012. — Vol. 34, no. 11. — P. 2274–2282.

[6] scikit-image: image processing in Python / Van der Walt S., Schönberger J. L., Nunez-Iglesias J., Boulogne F., Warner J. D., Yager N., Gouillart E., and Yu T. // PeerJ. — 2014. — Vol. 2. — P. 1–18.

[7] Boykov Y., Kolmogorov V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision // IEEE transactions on pattern analysis and machine intelligence. — 2004. — Vol. 26, no. 9. — P. 1124–1137.

[8] Boykov Y., Veksler O., Zabih R. Fast approximate energy minimization via graph cuts // IEEE Transactions on pattern analysis and machine intelligence. — 2001. — Vol. 23, no. 11. — P. 1222–1239.