

Санкт-Петербургский государственный университет

Кафедра системного программирования

Мамро Никита Владимирович

Реализация алгоритмов выделения
вспомогательных объектов при измерении
штабелей круглого леса и лесовозов

Отчёт по учебной практике

Научный руководитель:
ст. преп. Смирнов М. Н.

Консультант:
инженер-программист 2-ой категории
ООО «Системы Компьютерного Зрения»
Куксенок Д. С.

Санкт-Петербург
2021

Оглавление

Введение	3
1. Постановка задачи	5
2. Обзор	6
2.1. Использование нейронных сетей	6
2.1.1. Возможные подходы	6
2.1.2. Доступные технологии	6
2.2. Классические методы сегментации	7
2.2.1. Пороговые методы	7
2.2.2. Метод водоразделов	8
2.2.3. Метод k средних	9
3. Испытание классических алгоритмов	11
3.1. Результаты работы пороговых методов	12
3.2. Результаты работы метода водоразделов	13
3.3. Результаты работы метода k средних	14
4. Реализация алгоритма для решения задачи	15
4.1. Идея алгоритма	15
4.2. Формирование набора данных	16
5. Оценка результатов работы алгоритма	18
6. Внедрение алгоритма в приложение	20
7. Заключение	22
Список литературы	23

Введение

В рамках работы предприятий лесной промышленности регулярно возникает задача измерения и учёта габаритов, объёма и коэффициента полноты для штабелей круглого леса и лесовозов. На данный момент существуют предприятия, на которых замеры производятся вручную, что может привести к неточности полученных результатов. Это является основной мотивацией получения инструментов, способных автоматизировать измерение леса. Одним из таких инструментов является мобильное приложение с функциональностью распознавания торцов бревен на фотографиях и использующее полученные результаты детекции и введенные пользователем дополнительные параметры для подсчёта необходимых характеристик.

При использовании приложения пользователи вынуждены вручную формировать некоторые параметры, получение которых приложением можно частично или полностью автоматизировать. Например, пользователь должен самостоятельно выделять прямоугольную область, содержащую уложенные на лесовозе брёвна на изображениях с видом лесовоза сбоку (Рис. 1) для определения габаритов пачки. На данный момент при начале разметки область располагается по центру изображения. Так как мобильное приложение призвано увеличить точность расчётов и минимизировать количество действий, которую работники предприятий вынуждены совершать вручную, автоматизация выделения области с брёвнами является одной из важных задач, стоящих при его разработке.

В отличие от распознавания, например, автомобильного номера на лесовозе выделение области с брёвнами не является распространённой тривиальной задачей. В случае выделения области задача может осложняться тем, что брёвна сливаются с задним планом или соседней пачкой.

Что может оказаться немаловажным, изображения делаются на камеры мобильных телефонов или планшетов, причём обычно недорогих, поскольку предприятие может не изъявить желание приобрести дорого-

стоящие устройства для каждого из сотрудников. В связи с этим нельзя предполагать, что изображения будут одинаково высокого качества, а также нужно обратить внимание на то, что необходима возможность использования алгоритма на мобильных устройствах, так как работа на лесных предприятиях может осуществляться в местах без доступа в Интернет.



Рис. 1: Изображение лесовоза на виде сбоку.

1. Постановка задачи

Цель данной работы — добавить в мобильное приложение функциональность для автоматического выделения необходимой области на изображениях с видом лесовозов сбоку, дающую большую точность, чем помещение прямоугольной области в центр изображения, которое используется на настоящий момент, для минимизации действий, которые должен выполнить пользователь при разметке.

Для достижения цели были поставлены следующие задачи:

- провести обзор алгоритмов выделения объектов на изображениях;
- выбрать и разметить данные;
- описать подход к решению задачи распознавания области на виде сбоку;
- реализовать алгоритм;
- внедрить полученный алгоритм в мобильное приложение.

2. Обзор

Для выделения объектов на изображениях могут быть использованы различные методы с использованием нейронных сетей. Кроме того существуют классические методы сегментации изображений, то есть поиска таких участков, каждый из которых характеризует один смысловой объект. Возможно такие методы могут помочь в решении поставленной задачи в качестве предварительной обработки.

2.1. Использование нейронных сетей

Одним из вариантов решения нетривиальных задач в области работы с изображениями является применение методов машинного обучения.

2.1.1. Возможные подходы

В основе способов решения задач компьютерного зрения можно использовать свёрточные нейронные сети[2], изначально нацеленные на эффективное распознавание изображений. Такие сети работают по средствам выделения некоторых признаков изображения и их последующей обработки. Например, эта концепция использована в таких популярных алгоритмах, как YOLO[4] или RCNN[9]. Так как стоит задача распознавания одного крупного объекта на изображении, вряд ли придётся прибегать к перечисленным алгоритмам, однако использование свёрточной сети в качестве основы метода решения поставленной задачи может дать хороший результат.

2.1.2. Доступные технологии

При работе с машинным обучением имеет смысл обратить на следующие фреймворки: TensorFlow[14] и PyTorch[8].

Во время выбора фреймворка решающими факторами являлись сохранение единообразия с имеющимся в проекте кодом для алгоритма

детекции брёвен для возможности переиспользования результатов и возможность запускать алгоритм на мобильных устройствах. В проекте используется TensorFlow, а запуск алгоритма на целевых устройствах возможен с использованием TensorFlow Lite и PyTorch Mobile. Важно заметить, что на момент написания работы только TensorFlow предоставлял поддержку GPU, для PyTorch Mobile эта функциональность всё ещё находится в разработке.

С учётом сказанного выше в рамках данной работы был выбран TensorFlow.

2.2. Классические методы сегментации

Далее будут рассмотрены некоторые из распространённых[12] методов решения задачи сегментации изображений, которые можно будет использовать для достижения цели данной работы.

2.2.1. Пороговые методы

Один из самых простых способов решить задачу сегментации является использование порога как некоторого признака, который помогает разделить изображение на классы.

При рассмотрении гистограмм изображений можно заметить, что все пиксели сгруппированы около нескольких основных центров. Чтобы выделить такие группы можно выбрать некоторое T , после чего определить все точки на изображении такие, что $f(x, y) > T$, принадлежат фону, а оставшиеся — объекту. После этого можно, например, окрасить все точки объекта в черный цвет, а фона — в белый.

Сегментация может быть произведена при помощи порогового метода с глобальным порогом или же при помощи метода с адаптивным порогом[19]. В первом случае выбирается порог, после чего проверяется всё изображение.

Преимущества метода:

- простота.

Недостатки метода:

- если гистограмма не бимодальная, скорее всего не получится добиться хорошего результата;
- чувствителен к шумам.

2.2.2. Метод водоразделов

Ещё одним из алгоритмов, использующихся для сегментации изображений, является метод водоразделов (англ. watershed).

Изображение рассматривается, как функция от двух переменных $f(x, y)$, в качестве f , можно, например, взять интенсивность изображения, а x и y будут координатами пикселя. Также в качестве f можно брать градиент изображения.

После того, как будут найдены минимумы f , можно рассмотреть график функции как рельефную поверхность, после чего начать заполнять "водой" глобальный минимум, как изображено на иллюстрации в одной из статей о методе водоразделов[13] (Рис. 2). Далее "вода" будет заполнять локальные минимумы и образуются "дамбы", разделяющие изображение на сегменты.

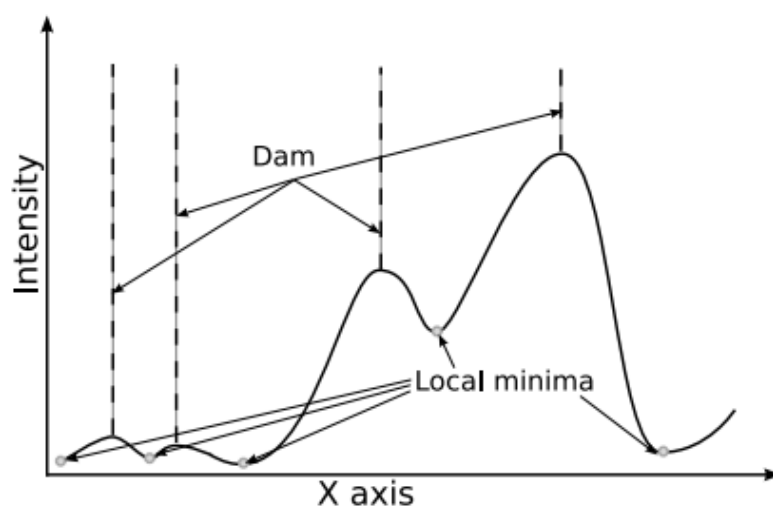


Рис. 2: Иллюстрация принципа работы метода водоразделов.

Преимущества алгоритма:

- в результате работы формируются закрытые связные области.

Недостатки алгоритма:

- алгоритм может давать чрезмерную сегментацию изображений [6];

Чтобы избавиться от проблемы чрезмерной сегментации, необходима дополнительная предварительная обработка изображений. Можно промаркировать области, привязав их к ближайшим минимумам. В таком дамба будет строиться только в случае объединения помеченных областей, иначе будет происходить слияние регионов. Тем не менее в рамках поставленной задачи можно предположить, что метод не даст хороших результатов. Проблемы возникнут как в случаях, когда брёвна будут частично закрыты внешними деталями прицепов, так и в случаях, когда пачка с брёвнами сливается с другими объектами или задним фоном.

2.2.3. Метод k средних

Для сегментации можно использовать методы кластерного анализа[20], например, метод k средних. В случае работы с изображениями суть кластеризации заключается в разбиении всех N пикселей на k кластеров, чтобы отклонение точек кластеров от центров кластеров было минимально. В случае задачи сегментации кластеры есть различные цвета.

Преимущества метода:

- простота;
- гарантированная сходимость.

Недостатки метода:

- нужно вручную выбирать подходящее значение k ;
- результат зависит от выбора центроидов;

- необходимо дополнительно выделять нужный кластер;
- медленный.

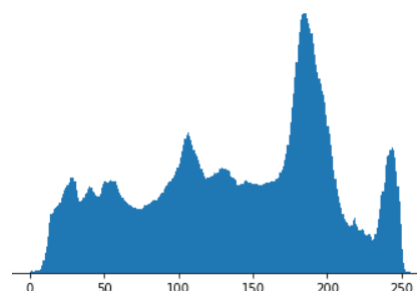
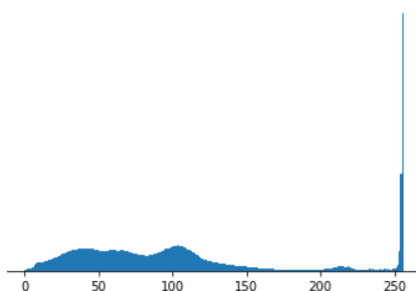
3. Испытание классических алгоритмов

С целью применения рассмотренных алгоритмов были выбраны изображения, использовавшиеся при осуществлении реальных замеров с использованием мобильного приложения. Для того, чтобы опробовать работу методов, было выбрано использовать язык Python, как язык, используемый для исследований внутри проекта, и библиотеку OpenCV[7], включающую в себя реализацию необходимых алгоритмов. Для оценки качества сегментации была использована метрика IoU[11]. Также производился запуск на смартфоне Xiaomi Redmi 4X с использованием библиотеки JavaCV для измерения времени работы алгоритмов.

В качестве примеров для получения представления о работе методов были использованы следующие два изображения лесовозов, на первом из которых есть только одна пачка и брёвна отчётливо видны, а на втором две пачки и припорошенные снегом брёвна:



Соответствующие гистограммы:



3.1. Результаты работы пороговых методов

Для выделения области, содержащей брёвна, на виде сбоку, был использован пороговый метод Оцу[18], содержащийся в библиотеке OpenCV. При использовании данного метода значение порога определяется автоматически посредством минимизации внутриклассовой дисперсии.

Глядя на изображения и их гистограммы, можно предположить, что для первого лесовоза произойдёт отделение кузова с колёсами от фона, а во втором скорее всего не получится достичь даже такого результата. Для второго изображения можно ожидать ещё более плохого результата, так как на гистограмме достаточно много пиков.

На рисунках 3 и 4 можно увидеть результат работы пороговых методов на изображениях в градациях серого с использованием параметра `THRESH_OTSU`, где изображение окрашивается в черный и самый яркий цвет.

Для первого изображения значение IoU равняется 0.43, а для второго — 0.37. Время работы на смартфоне не превышало 100мс.

Так как данный метод быстро работает и позволяет частично отделить лесовоз или даже пачку от заднего фона, можно попробовать использовать его в качестве предварительной обработки.



Рис. 3: Результат для первого лесовоза.



Рис. 4: Результат для второго лесовоза.

3.2. Результаты работы метода водоразделов

Для использования алгоритма была использована библиотека `skimage`[16]. Метод был применён с использованием маркировки областей, чтобы избежать избыточную сегментацию.

На рисунках 5 и 6 представлены результаты работы алгоритма для двух выбранных лесовозов без и с использованием модификации.

Для первого изображения значение IoU равняется 0.42, а для второго — 0.34.

Метод не показал результата лучше, чем пороговый, а также, как и предполагалось, найденные области могут не только содержать много лишних пикселей, но и не учитывать искомые пиксели, поэтому он не рассматривается в качестве предобработки в рамках представленной работы.



Рис. 5: Результат без предварительной обработки.



Рис. 6: Результат с предварительной обработкой.

3.3. Результаты работы метода k средних

При использовании алгоритма k средних центроиды выбирались случайным образом, а k было решено выбирать исходя из визуальной оценки изображений, чтобы посмотреть, как будет вести себя алгоритм при разных k .

Ниже представлены результаты работы алгоритма для двух изображений при k , равных 5, 7 и 20 слева направо соответственно после выделения кластеров. По результатам исследований[17] устанавливается, что часто k выбирается в промежутке от 2 до 10. Так как каждый кластер будет соответствовать какому-то цвету, нет смысла выбирать очень маленькие значения k , поэтому было решено посмотреть, как алгоритм будет себя вести при таких средних значениях, как 5 и 7, и достаточно высоком значении $k = 20$.

Как можно видеть на полученных результатах (Рис. 7, Рис. 8), действительно оказалось, что малые значения k не дали достаточно хорошего результата.

Для первого изображения значение IoU равняется 0.78, а для второго — 0.58.

Уже при $k = 7$ время работы алгоритма превышало минуту на мобильном устройстве, что непозволительно долго, в связи с чем было принято решение отказаться от данного метода при решении поставленной задачи.

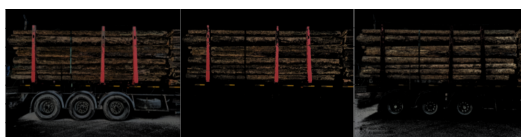


Рис. 7: Результат для первого лесовоза.



Рис. 8: Результат для второго лесовоза.

4. Реализация алгоритма для решения задачи

После изучения возможных решений для выделения объекта на изображении был составлен алгоритм решения поставленной задачи.

4.1. Идея алгоритма

Прямоугольную область можно описать четырьмя значениями: координатами левой нижней вершины и правой верхней. Тогда решение поставленной задачи сводится к предсказанию некоторых четырёх значений от 0 до 1, соответствующих искомым значениям координат относительно разрешения изображения.

Для решения задачи предсказания было решено использовать модель на основе некоторой существующей архитектуры сверточной нейронной сети. Сначала изображение проходит через слои свёрток и пулингов для выделения признаков — фич. После чего результат проходит через полносвязные слои для выделения четырёх значений, и в конце с использованием функции активации на выходе получаются 4 значения от 0 до 1.

На изображении (Рис. 9) проиллюстрирован ход работы алгоритма. Будем считать, что необходимо получить значения a , b , c , d , отвечающие координатам x_{min} , y_{min} , x_{max} , y_{max} прямоугольной области. Пусть изначальное предсказание — четвёрка a_1 , b_1 , c_1 , d_1 , задача во время обучения — минимизировать среднеквадратичное отклонение от верного результата. На изображении зелёным цветом отмечена ожидаемая область, красным цветом отмечены области, полученные в процессе минимизации функции потерь, и жёлтым — окончательное предсказание, которое получается с использованием обученного алгоритма.

В качестве основы для алгоритма были опробованы такие архитектуры сверточных сетей, как SimpleNet[10], сочетающую в себе легковесность и высокую результативность[3], и MobileNet[1], являющуюся эффективной с точки зрения времени работы на мобильных устройствах

и при этом не сильно менее легковесной. SimpleNet была реализована с использованием текста оригинальной статьи, а для MobileNet было использовано готовое решение, которое предоставляет TensorFlow.

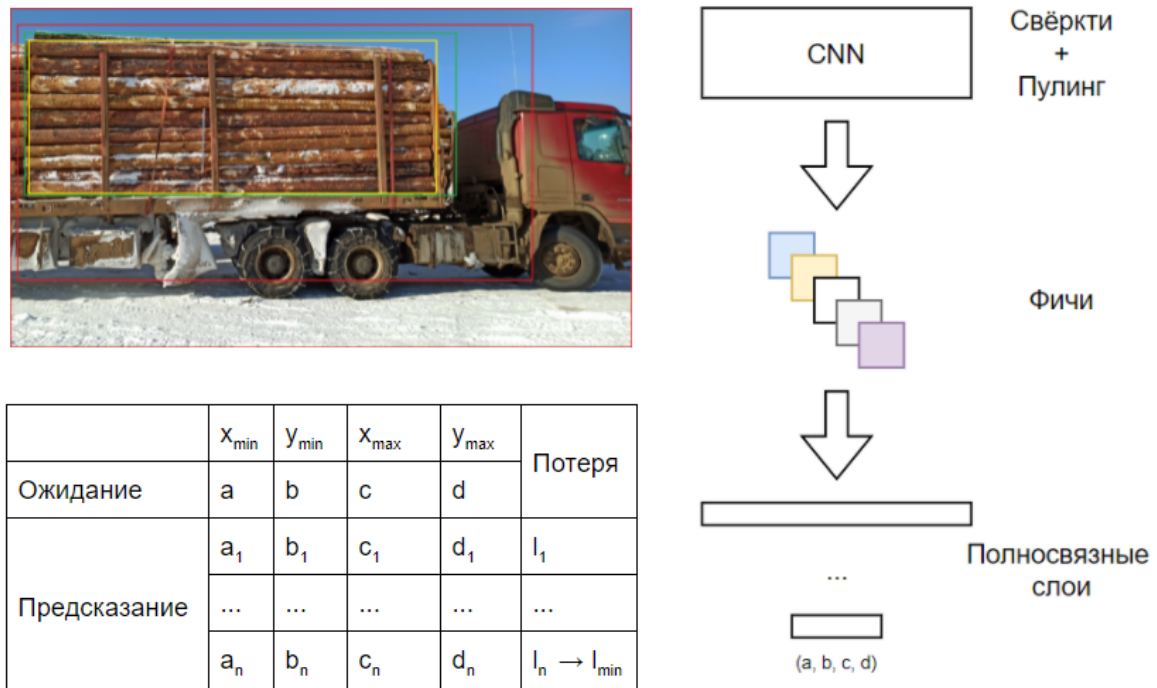


Рис. 9: Иллюстрация концепции использованного алгоритма.

4.2. Формирование набора данных

Для обучения нейронной сети, а также для оценки качества реализованного алгоритма необходим набор данных, содержащих изображения с лесовозом на виде сбоку и информацию о координатах левой нижней и правой верхней вершин прямоугольной области, корректно выделяющей пачку брёвен. Для разметки изображений использовался инструмент LabelImg[5], который позволяет в графическом редакторе отмечать необходимую прямоугольную область, после чего сохранять информацию о 4 координатах в XML файл.

Во время исследований выяснилось, что после обучения на данных без предварительной аугментации алгоритм ищет прямоугольную область в районе центра изображения, даже если пачка сильно смещена.

В связи с этим было решено сгенерировать дополнительные изображения (Рис. 10) на основе имеющихся, в которых пачка будет смещена. Также при аугментации у изображений случайным образом менялась яркость.



Рис. 10: Пример изображений, полученных из исходных при аугментации.

5. Оценка результатов работы алгоритма

После выбора наилучших результатов обучения, было проведено сравнение работы алгоритмов на мобильном устройстве между собой, а также с текущим решением — выделением прямоугольной области по центру изображения.

Сравнение результатов проводилось на 114 тестовых изображениях с использованием моделей сетей в формате tflite для мобильных устройств после квантизации с использованием шестнадцатибитных float-ов. На тестовых изображениях не было заметно ухудшения качества работы алгоритма после квантизации, но при этом было достигнуто значительное уменьшение размера модели.

Для оценки качества использовалась метрика IoU. Также сравнивались размеры модели и время работы на мобильном устройстве. Результаты работы алгоритма можно увидеть в таблице (Табл. 1), где при измерении времени также указано значение максимального отклонения от среднего. Распределение значений IoU представлено на графике (Рис. 11). В результатах значения, соответствующие названию Dummy, являются результатом помещения прямоугольной области в центр экрана, которое используется в приложении сейчас. Для классических методов сегментации не указывается время работы, так как они дают значения IoU ниже, чем предъявленный алгоритм на основе нейронных сетей.

Среднее значение	Dummy	SimpleNet (+Otsu)	MobileNet (+Otsu)	Классические методы
IoU	0.47	0.71 (0.69)	0.77 (0.73)	Не более 0.5
Размер, кб	—	2549	12346	—
Время первой детекции, мс	—	2300 $\pm$ 200	795 $\pm$ 60	—
Время последующих детекций, мс	—	1642 $\pm$ 160	159 $\pm$ 40	—

Таблица 1: Таблица результатов работы алгоритма.

Время первой детекции после запуска приложения всегда больше времени последующих детекций, что обусловлено тем, что модель должна загрузиться в память.

Из результатов видно, что полученный алгоритм превосходит по качеству выделение пачки по средствам построения прямоугольной области в центре экрана. Несмотря на то, что модель на основе SimpleNet значительно меньше по размеру, она уступает в качестве выделения области и времени работы модели на основе MobileNet. Вес модели в 12.5 мегабайт можно считать достаточно небольшим с учётом того, что в процессе взаимодействия с приложением пользователь генерирует большие объёмы данных посредством загрузки изображений, поэтому было решено интегрировать именно модель на основе MobileNet в имеющееся приложение. Также исходя из оценки качества работы алгоритма было принято решение остановиться на алгоритме без учёта предварительной обработки.

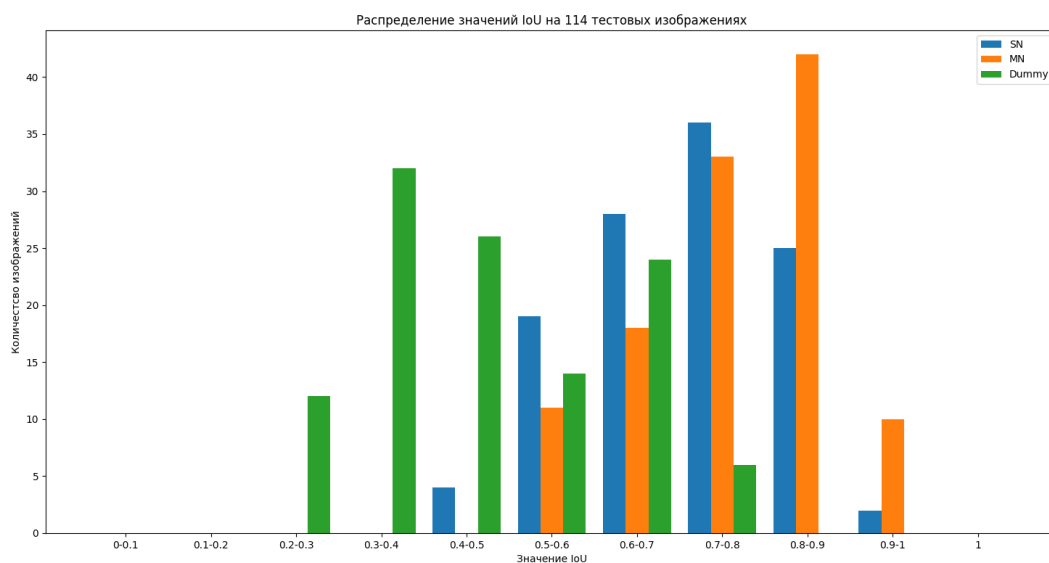


Рис. 11: График, иллюстрирующий распределение значений IoU.

6. Внедрение алгоритма в приложение

Преобразованную при помощи TensorFlow Lite модель можно запустить на мобильном устройстве с использованием интерпретатора[15].

За счёт того, что на экранах разметки (Рис. 12) пользователь сначала должен отметить эталон, и только после область сбоку, скорость работы алгоритма можно считать достаточной, и при переходе на экран с разметкой области сбоку, предположительные координаты прямоугольной области уже будут посчитаны.

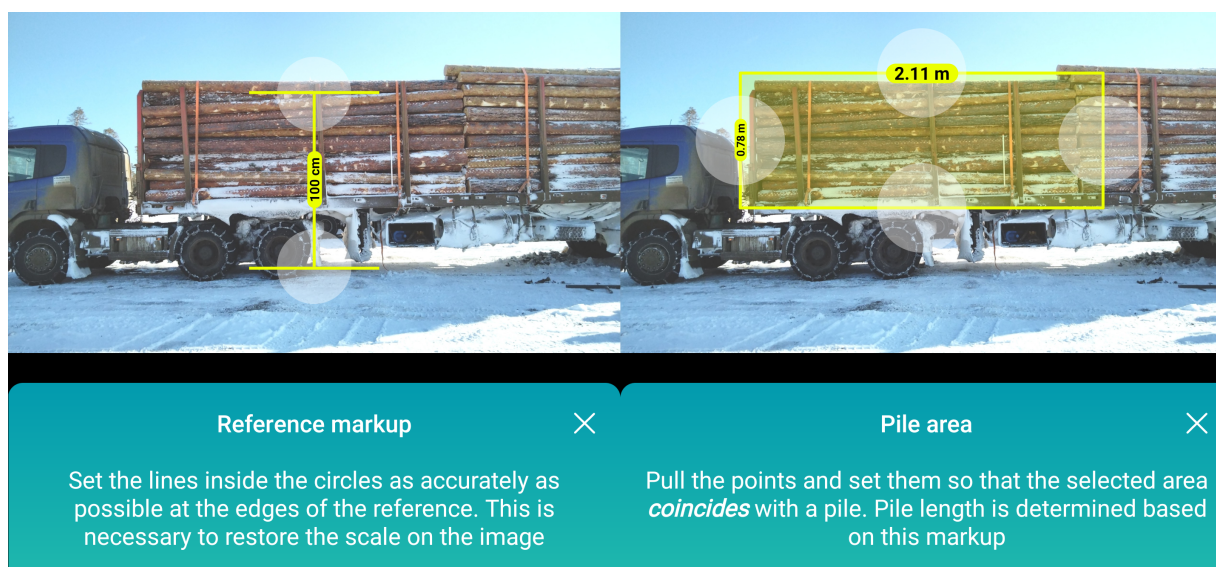


Рис. 12: Экраны с разметкой изображения.

Модель хранится в приложении в зашифрованном виде и дешифруется перед загрузкой. За шифрование и дешифрование отвечает уже имеющийся в проекте код, который используется для существующей модели, отвечающей за детекцию торцов брёвен.

Согласно архитектуре приложения действия пользователя на том или ином экране обрабатывает соответствующая модель представления. За более сложную логику отвечают объекты-интеракторы, методы которых вызываются из вью моделей. В рамках решаемой задачи нужно было реализовать метод для детекции границ пачки в `DetectionInteractor` (Рис. 13). Интерактор использует объект типа `DetectorSelector`, который возвращает нужную реализацию интерфейса `Detector` в соответствии с тем, для какого изображения необходимо

осуществить детекцию. Для выделения пачки с брёвнами был добавлен новый класс, реализующий интерфейс `Detector`, который принимает изображение и с использованием интерпретатора TensorFlow Lite возвращает 4 значения, которые соответствуют координатам точек, необходимых для добавления прямоугольной области на экране разметки.

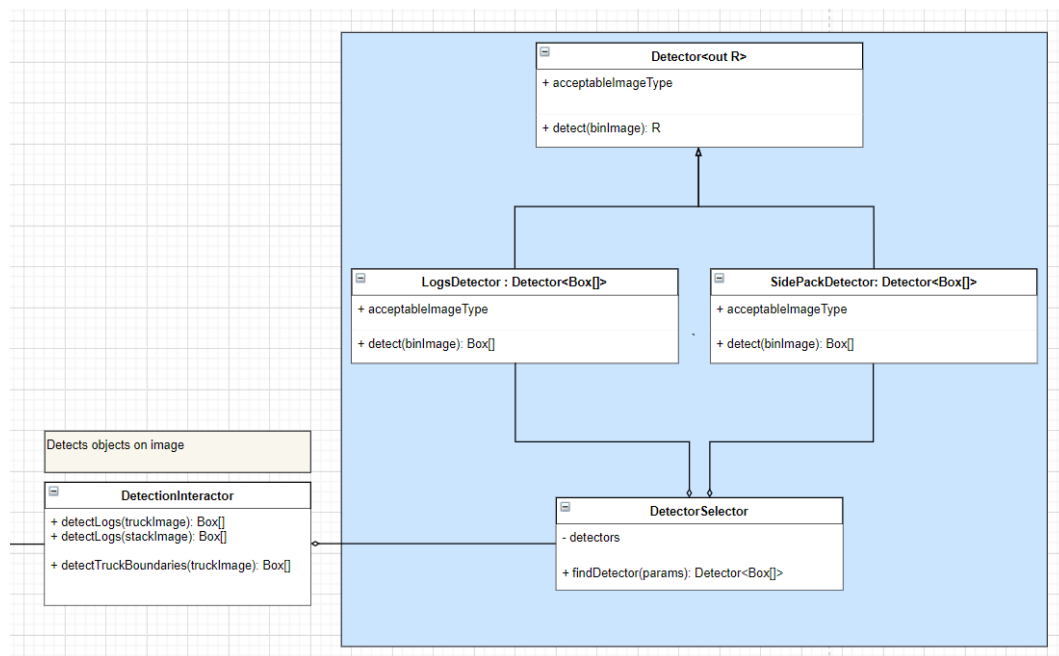


Рис. 13: Диаграмма классов, отвечающих за детекцию.

7. Заключение

Были достигнуты поставленные цели:

- проведён обзор алгоритмов выделения объектов на изображениях;
- описан алгоритм решения задачи нахождения пачки на изображении с видом лесовоза сбоку;
- собраны и размечены данные для применения алгоритма;
- реализован описанный алгоритм;
- алгоритм интегрирован в мобильное приложение.

Исходный код загружен в закрытый репозиторий проекта.

Список литературы

- [1] Andrew G. Howard Menglong Zhu Bo Chen Dmitry Kalenichenko Weijun Wang-Tobias Weyand Marco Andreetto Hartwig Adam.
- [2] Das Angel. Convolution Neural Network for Image Processing — Using Keras // towardsdatascience.com. — URL: <https://towardsdatascience.com/convolution-neural-network-for-image-processing-using-keras-dc3> (online; accessed: 29.05.2021).
- [3] Elia Velasco-Montero Jorge Fernandez-Berni Ricardo Carmona-Galan Angel Rodriguez-Vazquez. Optimum Selection of DNN Model and Framework for Edge Inference // core.ac.uk.
- [4] Joseph Redmon Santosh Divvala Ross Girshick Ali Farhadi. You Only Look Once: Unified, Real-Time Object Detectio // arxiv.org. — URL: <https://arxiv.org/pdf/1506.02640.pdf> (online; accessed: 29.05.2021).
- [5] LabelImg // github.com. — URL: <https://github.com/tzutalin/labelImg> (online; accessed: 27.05.2021).
- [6] Niket Amoda Ramesh K Kulkarni. Efficient Image Segmentation Using Watershed Transform // IJCST. — URL: <http://ijcst.com/vol42/2/niket.pdf> (online; accessed: 28.03.2021).
- [7] OpenCV // OpenCV-Python Tutorials. — URL: <https://docs.opencv.org/master> (online; accessed: 28.03.2021).
- [8] PyTorch // pytorch.org. — URL: <https://pytorch.org/> (online; accessed: 27.05.2021).
- [9] Ross Girshick Jeff Donahue Trevor Darrell Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation // openaccess.thecvf.com. — URL: https://openaccess.thecvf.com/content_cvpr_2014/papers/

Girshick_Rich_Feature_Hierarchies_2014_CVPR_paper.pdf
(online; accessed: 30.05.2021).

- [10] Seyyed Hossein Hasanpour Mohammad Rouhani Mohsen Fayyaz Mohammad Sabokrou. Let's keep it simple, using simple architectures to outperform deppere and more complex architectures // arxiv.org. — URL: <https://arxiv.org/pdf/1608.06037.pdf> (online; accessed: 27.05.2021).
- [11] Sheremet Oleksii. Intersection over union (IoU) calculation for evaluating an image segmentation model // towardsdatascience.com. — 2019.
- [12] Song Yuheng, Yan Hao. Image Segmentation Algorithms Overview // arxiv.org. — URL: <https://arxiv.org/ftp/arxiv/papers/1707/1707.02051.pdf> (online; accessed: 27.03.2021).
- [13] Sébastien Derivaux Germain Forestier Cédric Wemmert Sébastien Lefèvre. Supervised image segmentation using watershedtransform, fuzzy classification and evolutionarycomputation // HAL. — URL: <https://hal.archives-ouvertes.fr/hal-00512749v2/document> (online; accessed: 28.03.2021).
- [14] TensorFlow // tensorflow.org. — URL: <https://www.tensorflow.org> (online; accessed: 27.05.2021).
- [15] TensorFlow Lite Interpreter // tensorflow.org. — URL: https://www.tensorflow.org/lite/api_docs/java/org/tensorflow/lite/Interpreter (online; accessed: 28.05.2021).
- [16] Watershed segmentation // scikit-image. — URL: https://scikit-image.org/docs/dev/auto_examples/segmentation/plot_watershed.html (online; accessed: 28.03.21).
- [17] Xin Zheng, Qinyi Lei, Run Yao, Yifei Gong, Qian Yin. Image segmentation based on adaptive K-means algorithm // Springer. —

URL: <https://link.springer.com/content/pdf/10.1186/s13640-018-0309-3.pdf> (online; accessed: 28.03.2021).

- [18] Yousefi Jamileh. Image Binarization using Otsu Thresholding Algorithm // researchgate.net. — URL: https://www.researchgate.net/publication/277076039_Image_Binarization_using_Otsu_Thresholding_Algorithm (online; accessed: 27.05.2021).
- [19] В. Чичварин Н. Пороговые методы // Национальная библиотека им. Н. Э. Баумана. — URL: [https://ru.bmstu.wiki/Пороговые_методы_\(Сегментация\)](https://ru.bmstu.wiki/Пороговые_методы_(Сегментация)) (дата обращения: 28.03.2021).
- [20] Кластерный анализ // Лаборатория статистических исследований Кубанского государственного университета. — URL: http://statlab.kubsu.ru/sites/project_bank/claster.pdf (дата обращения: 03.28.2021).