Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Технология программирования

Бабкин Иван Алексеевич

Анализ предобработки и аугментации изображений для поиска аномалий при флюорографическом исследовании

Бакалаврская работа

Научный руководитель:
к.ф.-м.н., ст. преп. СПбГУ Салищев С. И.

Рецензент:
к. т. н., доцент ГУАП Ронжин А. Л..

Санкт-Петербург
2018

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Technology in Programming

Babkin Ivan

Analysis of pre-processing and augmentation of images for anomaly detection in preventive fluorographic

Graduation Thesis

Scientific supervisor:
senior lecturer Sergey Salishchev

Reviewer:
assoc. prof. Alexander Rohzhin

Saint-Petersburg
2018

Оглавление

Введение

Легочные заболевания являются одними из самых больших угроз для здоровья людей. От них умирает каждый шестой человек в мире. Ситуация не меняется с начала столетия, понизить опасность легочных болезней так и не удалось. Заболевания легких ведут к инвалидности и преждевременной смерти.

Одним из самых распространенных и массовых методов диагностики различных заболеваний легких является флюорография. Рентген легких назначается врачом как для профилактического медицинского осмотра дыхательной системы пациента, так и для постановки и уточнения диагноза при следующих заболеваниях:

- туберкулез;
- бронхит;
- злокачественные новообразования;
- пневмония;
- плеврит и др.

Анализ полученных результатов занимает у врачей много времени, при этом может быть допущена ошибка и врач неправильно поставит диагноз. Медицинские учереждения, проводящие ежегодное флюорографическое обследование, заинтересованы в разработке программы, которая по ренгенограммам могла бы определять, имеются ли в легких аномалии.

В данной работе было принято решение применить генеративный подход машинного обучения для выявления патологий по флюорографическим снимкам.

На данный момент перспективным подходом считаются автоэнкодеры. Автоэнкодер — архитектура нейронных сетей, которая переводит данные в скрытый слой меньшей размерности, а затем пытается восстановить исходные данные. При этом сеть автоматически обучается

выделять ключевые точки на изображении. Есть предположение, что автоэнкодер научится выделять аномалии на снимках в скрытом слое.

Был обучен сверточный автоэнкодер. В конце обучения потеря у сверточного автоэнкодера составила 0.008. Более подробнее о реализации модели сверточного автоэнкодера написано в разделе 4.1.

Далее, данные, полученные на выходе энкодера, подаются на вход методу опорных векторов (SVM). SVM классифицирует данные, разделяя их на два класса: легкие с патологиями и здоровые легкие. Подробная информация о реализации SVM представлена в разделе 4.2.

Для оценки качества классификации была посчитана площадь под ROC-кривой (receiver operating characteristic), которая составила 0.93. Подробное описание результатов представлено в разделе 5.

Возникает проблема в связи с отсутствием мощных машин для обучения нейронных сетей, нет возможности обучить достаточно сложную модель. Поэтому были предобработаны изображения: исключена лишняя информация, которая может негативно повлиять на обучение автоэнкодера, и уменьшена размерность до 128×128.

Для обработки и улучшения качества снимков было использовано выравнивание гистограммы изображения, а для выделения границ легких — оператор Собеля. Полное описание алгоритма изложено в разделе 3.

Новизна моего подхода заключается в том, что автоэнкрдер обучается для понижения размерности данных в задаче поиска аномалий на рентгенограммах. Обучение производится на неразмеченных предобработаных данных, которые предоставлены СПб НИИ пульмонологии.

1. Постановка задачи

Целью данной работы является обучение модели, выявляющей аномалии на предобработанных флюорографических снимках.

Для достижения этой цели были сформулированы следующие задачи:

- предобработать данные
- реализовать и обучить модели для поиска аномалий
- оценить качество полученной модели.

2. Обзор технологий

2.1. Существующие решения

На данный момент существуют решения, которые могут выявлять определенные заболевания с некоторой точностью. Многие работы, в которых получен достойный результат, используют снимки компьютерной томографии.

Авторы данной статьи [15] реализовали сегментацию раковых образований, а так же классификацию легочных узлов. Был реализован нечеткий алгоритм средних и метод опорных векторов. Обучение производилось на КТ снимках. Получена точность 97,6%.

В статье [5] производится обзор решений для распознавания легочных узлов на основе снимков компьютерной томографии.

Также была рассмотрена работа [12], в которой производится распознавание интерстициальные заболевания лёгких. Для классификации была обучена глубокая сверточная нейросеть. Полученная общая точность классификации 68.6%.

2.2. Выделение границ

Было проведено достаточно много исследований по теме выделения границ объекта на изображении. В частности, были проведены исследования по выделению контуров на изображении в медицине. В работе [17] было проведено сравнение методов Робертса, Прюитта и Собеля для выявления границ на КТ снимке человеческого мозга. Оператор Собеля в данной работе показыват лучшие результаты.

2.2.1. Оператор Собеля

Оператор Собеля [7] широко используется в обработке изображений, особенно в задачах обнаружения границ. Технически это дискретный оператор дифференцирования, вычисляющий приближение градиента функции интенсивности изображения. В каждой точке изображения

результатом оператора Собеля является либо соответствующий вектор градиента, либо норма этого вектора. Оператор Собеля является частной производной $f(x, y)$ как центральная вычислительная окрестность 3×3 в направлении x и y.

Чтобы уменьшить шум, определенный вес соответственно увеличивается в центральной точке, а его уравнения с цифровым градиентом могут описываться следующим образом:

$$\begin{aligned} G_x = &\{f(x+1, y-1) + 2f(x+1, y) + f(x+1, y+1)\} \\ &- \{f(x-1, y-1) + 2f(x-1, y) + f(x-1, y-1)\}, \end{aligned} \tag{1}$$

$$\begin{aligned} G_y = &\{f(x-1, y+1) + 2f(x, y+1) + f(x+1, y+1)\} \\ &- \{f(x-1, y-1) + 2f(x, y-1) + f(x+1, y-1)\} \end{aligned} \tag{2}$$

В целом, размер градиента определяется формулой:

$$g(x, y) = \sqrt{G_x^2 + G_y^2} \tag{3}$$

Далее, G_x и G_y используются для свертки изображения.

2.2.2. Гауссовский фильтр

Фильтр Гаусса [4] интенсивно используется в области обработки изображения. Наиболее распространенное применение — подавление шума. Двумерный цифровой гауссовский фильтр можно определить следующим образом:

$$G(x, y) = \frac{1}{\sqrt{2\pi\sigma}} exp(-(x^2 + y^2/2\sigma^2)), \tag{4}$$

где σ^2 — дисперсия фильтра Гаусса. При использовании гауссовского фильтра для подавления шума, большая дисперсия фильтра эффективна для сглаживания шума, но в то же время происходит искажение тех частей изображения, где происходят резкие изменения яркости пикселей.

2.3. Автоэнкодер

Автоэнкодер [1] — алгоритм обучения без учителя, который переводит входные данные в пространство меньшей размерности, после чего на выходе получает данные размерности, равной размерности входных данных. Автоэнкодер используется для сжатия данных, визуализации, классификации и т. д. Входные данные сжимаются энкодером до размерности скрытого слоя, а затем декодер пытается восстановить их. Таким образом автоэнкодер обучается выделять полезные признаки входных данных.

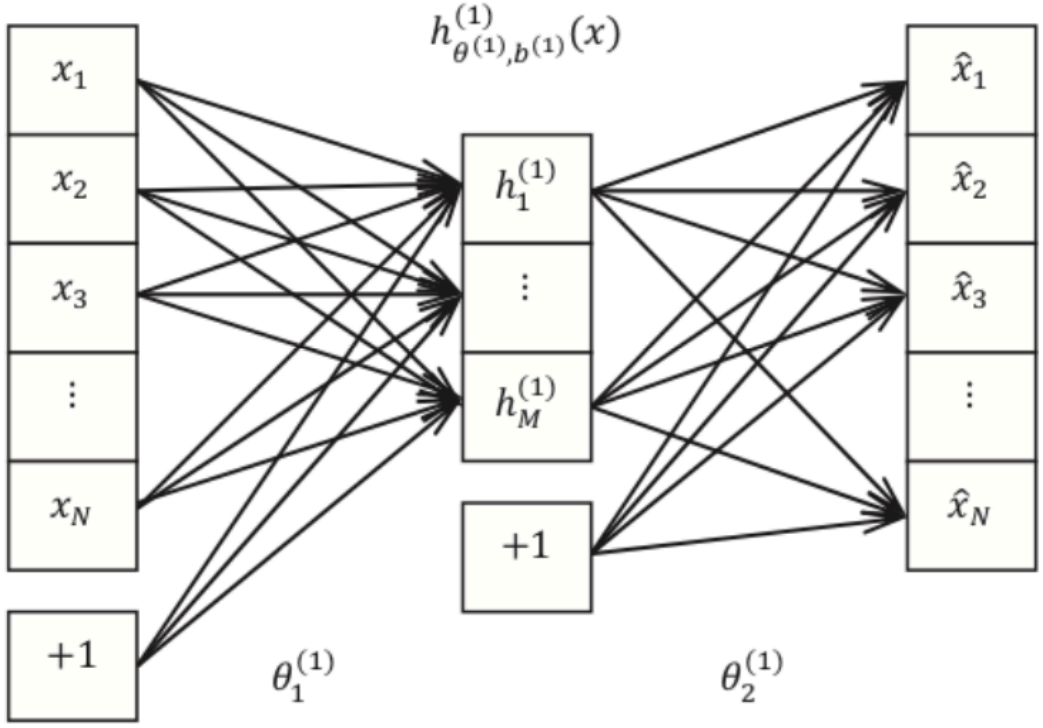


Рис. 1: Модель автоэнкодера.

Автоэнкодер (рис. 1) пытается обучить функцию $h_{\theta,b}(x) \approx x$, где θ и b — веса сети и смещения, полученные соответственно во время обучения, x — данные, подаваемые на вход. Другими словами, он пытается узнать приближенную функцию идентичности, такую что выходной $\hat{x}$ и входной x были бы одинаковы. Ограничивая скрытый слой, можно получить выделение интересных свойств объекта.

Сложный автоэнкодер представляет собой нейронную сеть, состоящую из нескольких слоев разреженных автоэнкодеров.

2.3.1. Сверточный автоэнкодер

Сверточный автоэнкодер [11] — автоэнкодер, состоящий из сверточных слоев. Процедура сверточного преобразования входных данных в скрытый слой называется сверточным энкодером. Затем выходные значения энкодера восстанавливаются посредством обратной сверточной операции, которая называется сверточным декодером.

На вход сверточному автоэнкодеру подаются данные $x \in R^{n \times l \times l}$, где n — это количество входных каналов, $l \times l$ — это размер изображения. На выходе автоэнкодера мы должны получить $\hat{x} \in R^{n \times l \times l}, \hat{x} \approx x$.

Сверточный автоэнкодер также можно использовать для понижения размерности входных данных с выделением главных свойств объектов.

2.4. Метод опорных векторов

Ключевой концепцией метода опорных векторов (SVM) [9], которые первоначально были разработаны для задач двоичной классификации, является использование гиперплоскостей для определения границ принятия решений, разделяющих точки данных разных классов. SVM способны обрабатывать как простые линейные задачи классификации, так и более сложные, то есть нелинейные проблемы классификации. Как разделимые, так и неразделимые задачи обрабатываются SVM в линейном и нелинейном случаях. Идея SVM заключается в том, чтобы сопоставить исходные точки данных входного пространства с пространством большей размерности таким образом, чтобы проблема классификации стала проще в пространстве функций. Отображение выполняется с помощью подходящего выбора функции ядра.

Метод опорных векторов [13] является контролируемым алгоритмом обучения для классификации и регрессии. Обучаясь на наборе p-мерных векторов в векторном пространстве, SVM находит разделяющую гиперплоскость, которая разбивает векторное пространство на два подмножества векторов. Существует критерий качества гиперплоскости: она должна максимизировать границу между этими подмножествами.

Предположим, что мы имеем p-мерные векторы. Каждый из них принадлежит к одному из двух классов. Мы можем найти много $(p-1)$-мерных гиперплоскостей, которые классифицируют такие векторы, но есть только одна гиперплоскость, которая максимизирует разницу между двумя классами. Такая гиперплоскость называется гиперплоскостью максимального предела, и именно ее подразумевают под классификатором SVM.

2.5. Метод оценки качества

В двоичной проблеме решения классификатор определяет примеры как положительные или отрицательные. Решение, принятое классификатором, может быть представлено в структуре, известной как матрица ошибок. Матрица ошибок имеет четыре категории:

- true positives (TP) — это количество примеров, правильно помеченных как положительные
- false positives (FP) — это количество примеров, неправильно помеченных как положительные
- true negatives (TN) — это количество примеров, правильно помеченных как отрицательные
- false negatives (FN) — это количество примеров, неправильно помеченных как отрицательные.

Матрицу ошибок можно использовать для построения точек в пространстве ROC (Receiver Operating Characteristic) или в пространстве PR (Precision-Recall) [6]. Учитывая матрицу ошибок, мы можем определить метрики, используемые в каждом пространстве.

В пространстве ROC график определяет False Positive Rate (FPR) по оси x и True Positive Rate (TPR) по оси y. FPR измеряет долю отрицательных примеров, которые ошибочно классифицируются как положительные. TPR измеряет долю положительных примеров, которые помечены правильно. Формулы расчета TPR и FPR:

$$TPR = \frac{TP}{TP + FN}, \qquad FPR = \frac{FP}{FP + TN}.$$

В пространстве PR по оси x полнота (recall), по оси y точность (precision). Под полнотой подразумевается то же самое, что и TPR, тогда как точность показывает долю примеров классифицированых как положительные, которые действительно являются положительными. Формулы расчета precision и recall:

$$Precision = \frac{TP}{TP + FP}, \qquad Recall = \frac{TP}{TP + FN}.$$

Кривые ROC и PR обычно строятся для оценки качества алгоритма машинного обучения на наборе данных. Большее значение площади под графиком показывает лучшую способность модели классифицировать данные.

3. Предобработка изображений

Важной частью перед обучением модели является предобработка изображений.

По причине отсутствия доступа к мощным машинам, нет возможности обучать сложные модели, которые с высокой точностью могли бы выделить ключевые точки на снимках флюорографии. Поэтому требуется обработка изображения, которая позволила бы убрать лишние «фичи».

Для начала, белые области сверху и снизу изображения закрашивались в черный цвет (рис. 2) для того, чтобы не размывалась граница между легкими и данной областью. Просматривался каждый пиксель изображения, начиная с каждого из углов, если значение пикселя находится в промежутке от 128 до 255, то значение пикселя устанавливалось 0. Когда такого пикселя справа и сверху ненаходится, то просмотр останавливался.

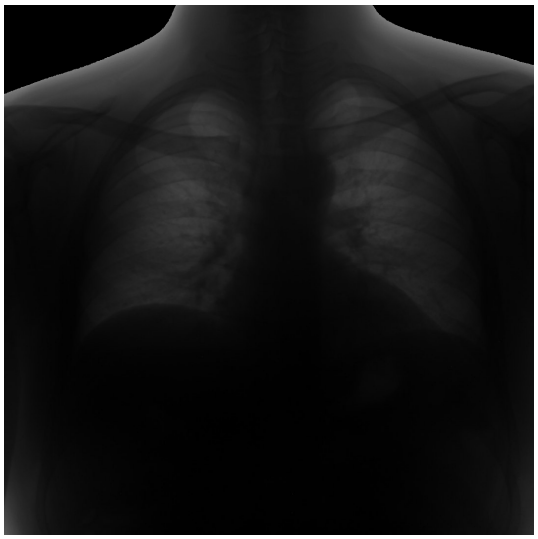
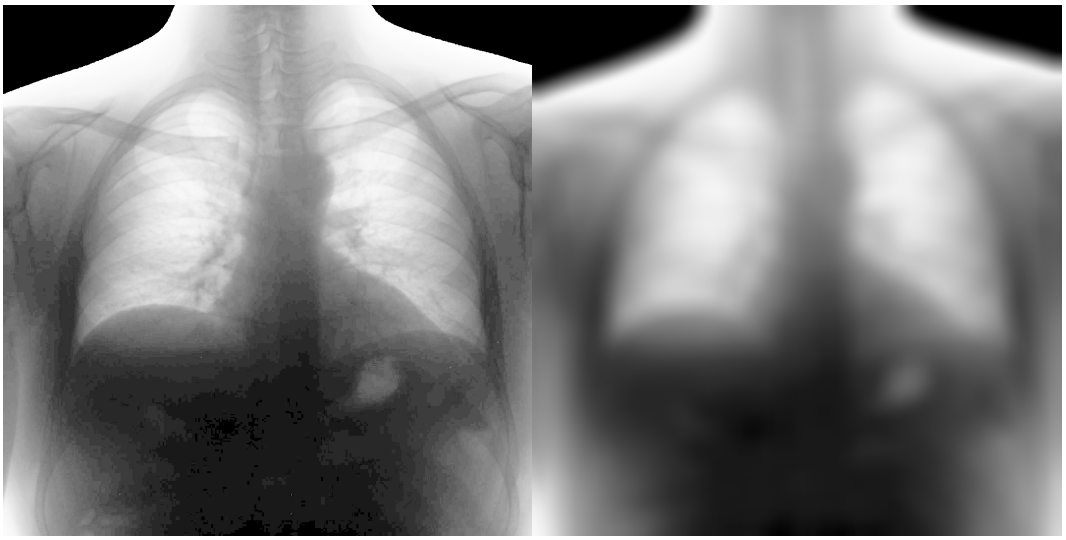


Рис. 2: Закрашивание белых областей в черный цвет.

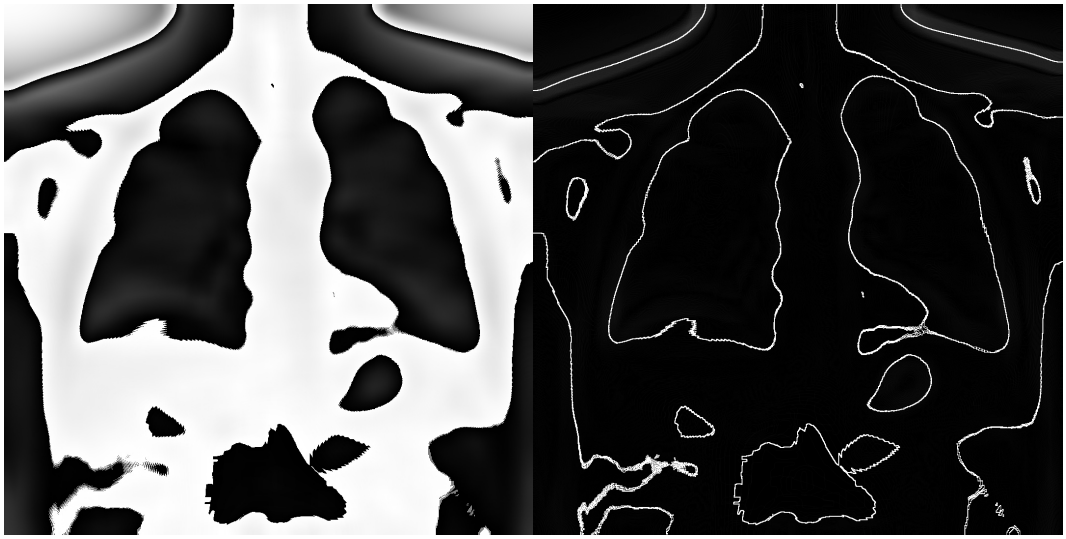
Далее, чтобы сделать снимок более подходящим для последующей обработки, необходимо было улучшить качество изображения путем выравнивания его гистограммы (рис. 3а). Можно заметить, что легкие стали более выражены на снимке.

Также необходимо применить фильтр для удаления нежелательного



(a) Выравнивание гистограммы изображения.

(b) Размытие изображения.



(c) Применение гауссовсого фильтра.

(d) Применение оператора Собеля.

Рис. 3

шума с изображения. Было принято решение использовать гауссовский фильтр (рис. 3c) с дисперсией $\sigma = 65$.

Для того, чтобы автоэнкодер выделял только важные ключевые точки на изображении, было принято решение выделять границы легких. Для решения данной задачи был применен оператора Собеля (рис. 3d). После его применения, границы легких получались слишком резкими. Поэтому для сглаживания границ применялось размытие изображения (рис. 3b).

Изображения уменьшаются до размера 128×128, для ускорения обу-

чения и уменьшения потребления оперативной памяти.

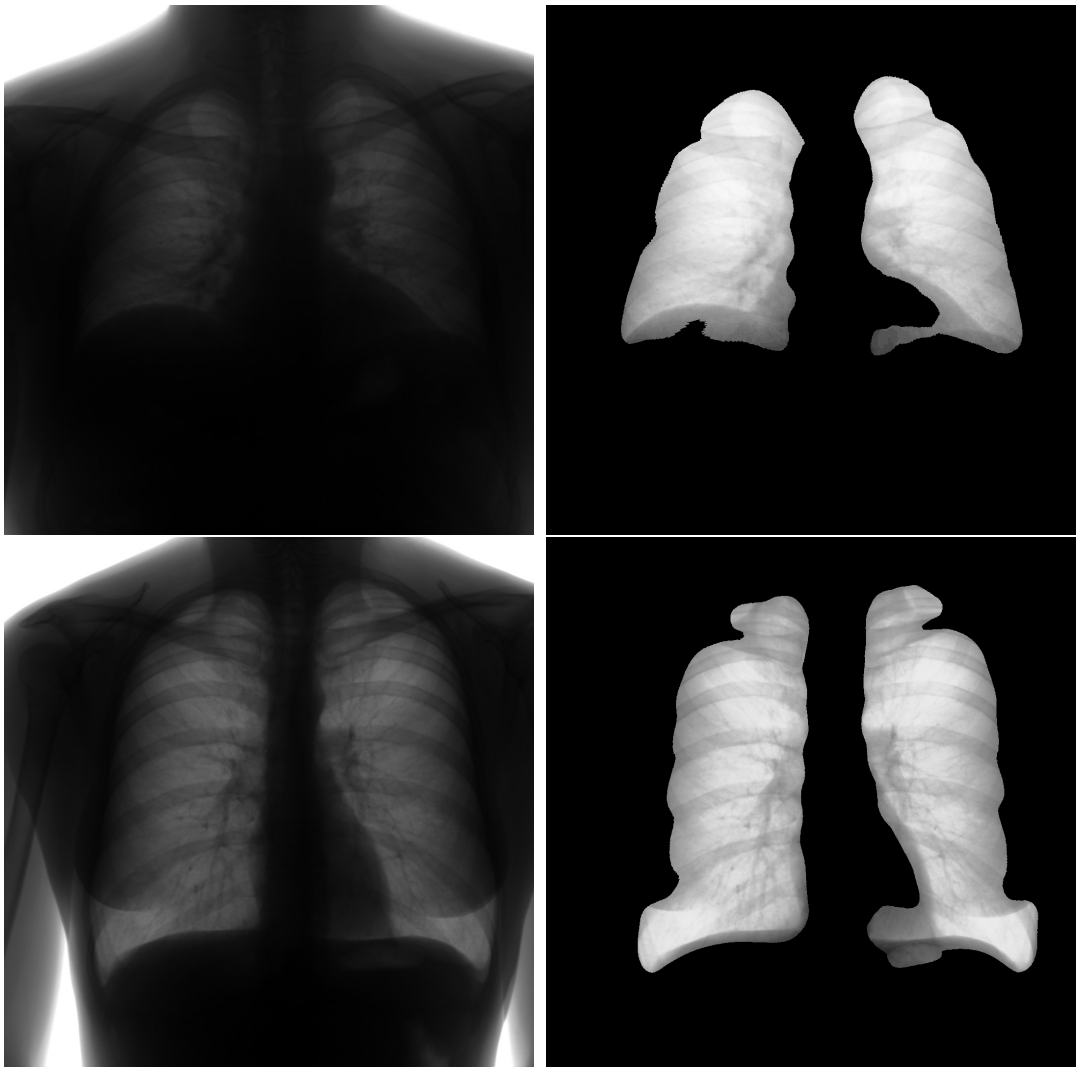


Рис. 4: Работа алгоритма предобработки изображений

Полный реализованный алгоритм предобработки изображений:

- закрашивание белых областей изображения сверху и снизу
- выравнивание гистограммы
- размытие изображения
- гауссовский фильтр
- оператор Собеля
- выделение контуров

- закрашивание областей вне контура в черный цвет
- уменьшение размера изображения до 128×128.

На рис. 4 показана работы данного алгоритма. Видно, что легкие стали лучше различимы. Так же на изображении остутвуют другие детали: на снимке присутствуют только легкие.

Алгоритм реализоаван на языке Python с использованием библиотек OpenCV [14] и Scipy [8].

4. Реализованные модели

Автоэнкодер — это генеративная модель, сотоящая из энкодера и декодера. Энкодер переводит входные данные в скрытый слой, а декодер пробует восстановить исходное изображение. Суть автоэнкодера в том, что он пытается выделить важные признаки в данных.

4.1. Сверточный автоэнкодер

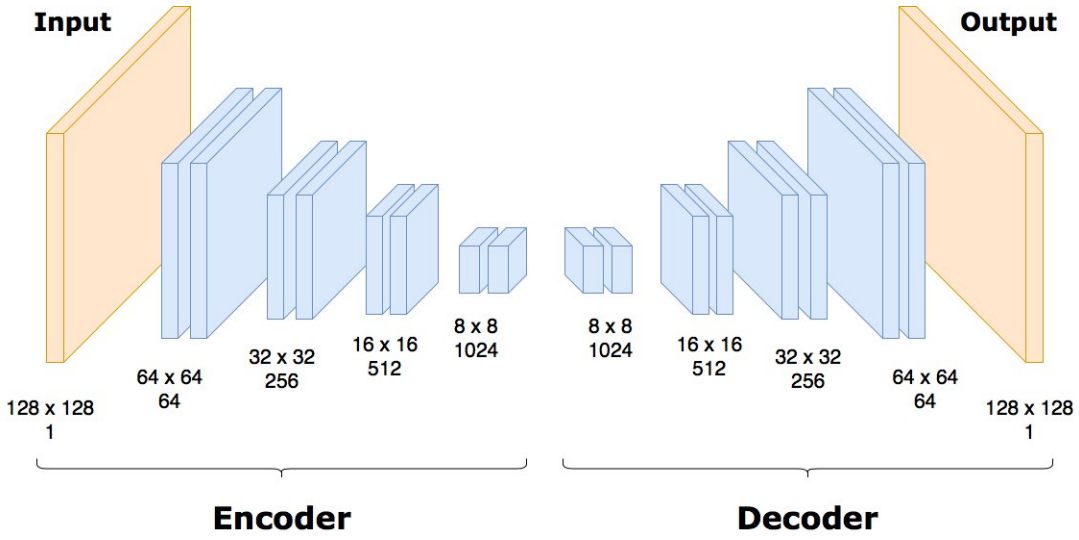


Рис. 5: Модель сверточного автоэнкодера.

Сверточный автоэнкодер — генеративная модель, состоящая из энкодера и декодера. Энкодер и декодер состоят из сверточных слоев. На рис. 5 показана реализованная модель.

На вход автоэнкодеру подается изображение размера 128x128 с количеством каналов 1. Далее размер изображений на каждом слое уменьшается, при этом увеличивается количество фильтров. Скрытый слой имеет размер 8x8 с количеством фильтров 1024. После каждого сверточного слоя используется batch normalization для ускорения обучения. Далее на выходе слоя применятеся функция активации ReLU: $f(x) = max(0, x)$. На выходе декодера использовалась функция активации Tanh: $g(x) = \frac{sinh(x)}{cosh(x)}$.

Модель реализована на языке Python с использованием фреймворка Pytorch [2]. Модель была обучена на данных в количестве 11295 сним-

ков. В качетсве метода оптимизации был выбран метод Адама [10]. Потеря автоэнкодера составила 0.008 (рис. 7).

Обучение производилось на сервере Microsoft Azure и заняло 3 дня со скоростью обучения 1e-2. Конфигурация описана в приложении А.

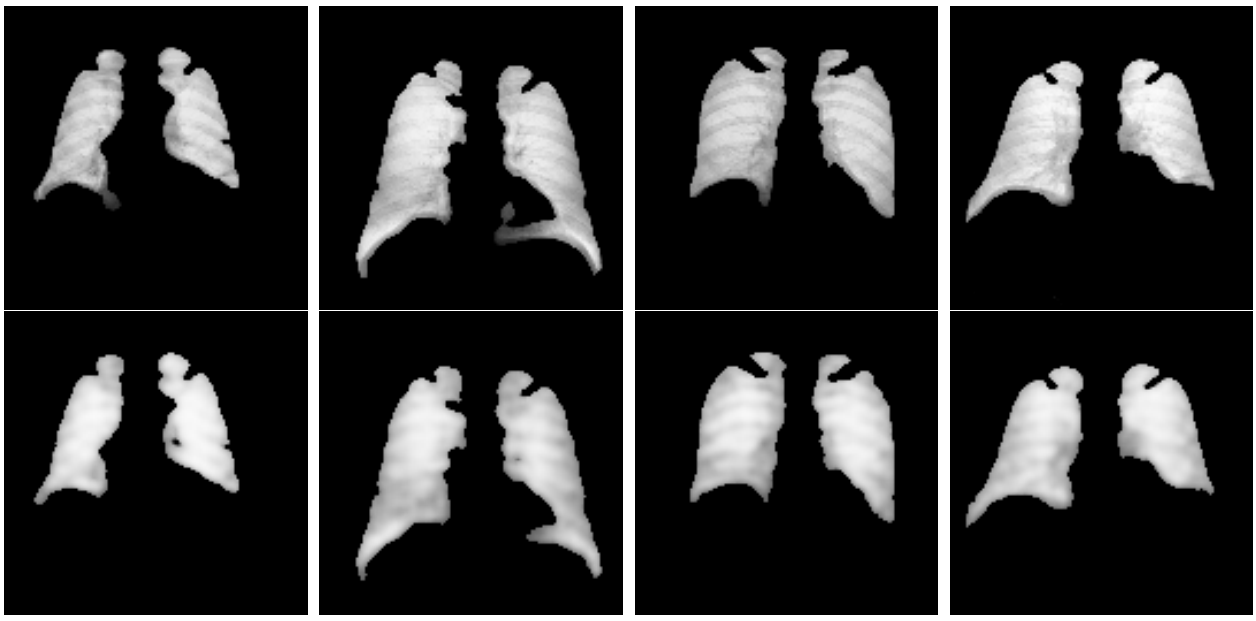


Рис. 6: Изображения, сгенерированные сверточным автоэнкодером.

На рис. 6 показаны результаты работы обученной модели. Изображения получаются немного размытыми, но при этом вся важная информация сохранется на снимках.

4.2. Метод опорных векторов

Для классификации изображений по признаку наличия аномалий был реализован метод опорных векторов (SVM). Для модели было определено два класса: легкие с аномалиями и легкие без аномалий. Для класса легких с аномалиями использовались снимки легких с различ-

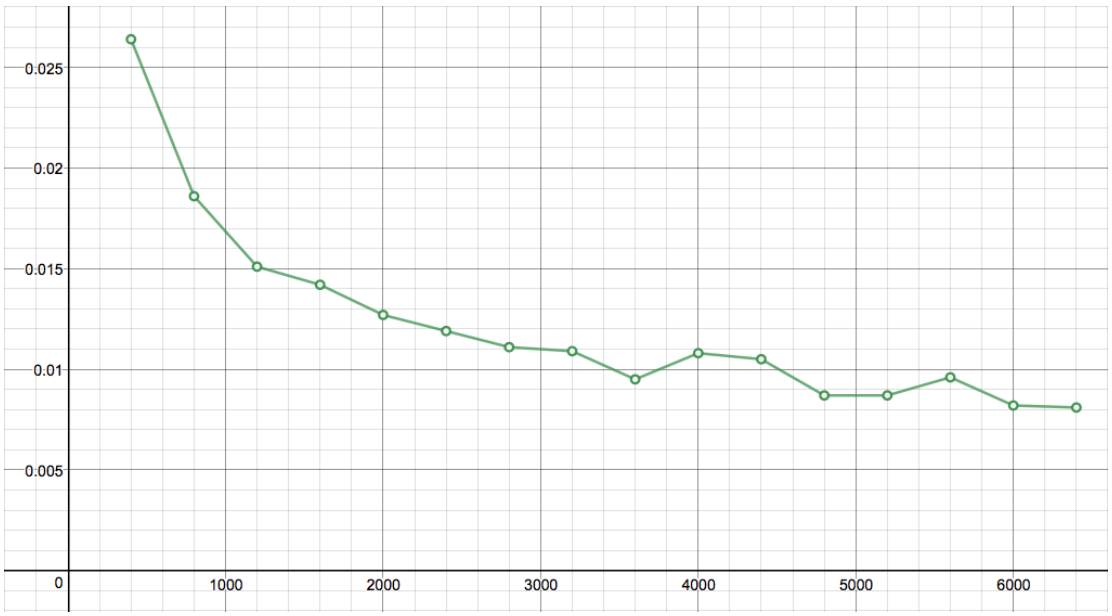


Рис. 7: График потерь сверточного автоэнкодера.

ными заболеваниями, для класса норм использовались неразмеченные снимки, так как считается, что практически все легкие в неразмеченных данных без патологий.

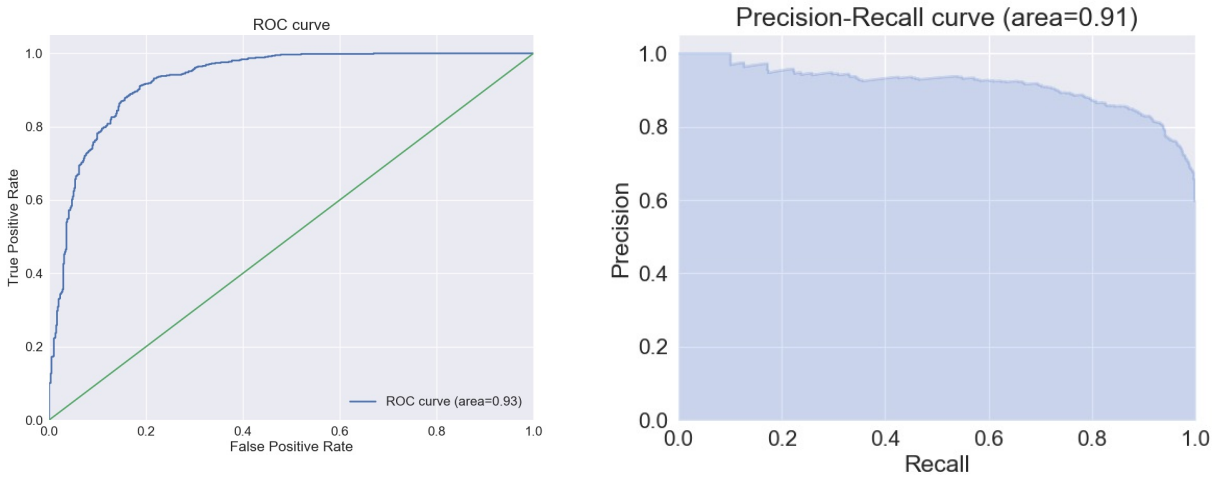
Изначально изображения обрабатывались энкодером сверточного автоэнкодера, затем выходные данные энкодера подавались на вход классификатору.

Так как входные данные не являются линейно-разделимыми, было принято решение использовать ядро RBF [3]. Было обучено несколько конфигураций SVM: для каждого C = 0.1, 1 и 10 были взяты $\gamma = 0.0001, 0.001, 0.010.1$, где C — это параметр регуляризации, γ — это ширина RBF ядра. Лучие результаты были получены для C = 10 и $\gamma = 0.01$.

Метод опорных векторов был реализован на языке Python с использование библиотеки sklearn [16].

Обучение производилось на сервере Microsoft Azure. Конфигурация описана в приложении А.

5. Результаты



(a) ROC-кривая

(b) PR-кривая

Рис. 8: Оценка качества метода опорных векторов, обученного на предобработанных данных.

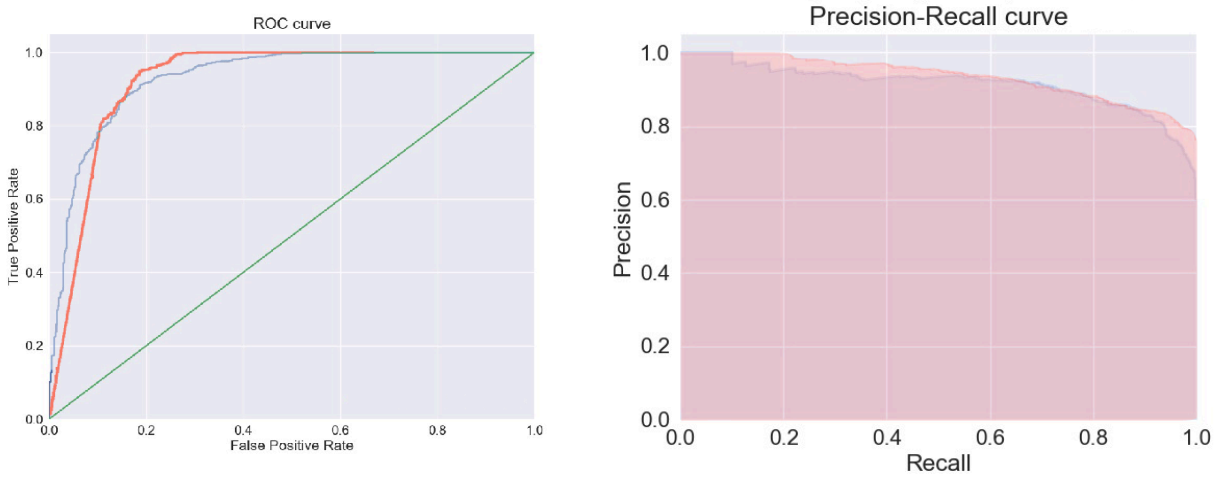
Тестовая выборка состояла из 1230 снимков легких: 615 неразмеченных снимков как здоровые легкие и 615 сников с легкими с патологиями.

Для обученной модели SVM была построена ROC-кривая и посчитана площадь под графиком. График изображен на рис. 8a. По оси x идут значения False Positive Rate, по оси y – True Positive Rate. Площадь под графиком ROC-кривой составила 0.93.

Таблица 1: Сравнение качества модели, обученной на разных данных.

	AUC-ROC	AUC-PR
Предобработанные данные	0.93	0.91
Непредобработанные данные	0.92	0.94

Также была построена PR-кривая и посчитана площадь под ней. График изображен на рис. 8b. По оси x идут значения полноты, по оси y – точности. Площадь под графиком PR-кривой составила 0.91.



(a) ROC-кривая

(b) PR-кривая

Рис. 9: Оценка качества метода опорных векторов. Красным цветом обозначена модель, обученная на данных, не подвергшихся предватрительной обработке, синим цветом — на предобработанных данных.

Для модели SVM, обученной на данных, не подвергшихся предватрительной обработке, также были построены ROC-кривая (рис. 9a) и PR-кривая (рис. 9b). AUC-ROC = 0.92, AUC-PR = 0.94.

Модель 1, обученная на предобработанных данных и модель 2, обученная на данных, которые не были предобработаны, показали примерно одинаковые результаты (таблица 1). Высокие показатели качества модели 1 можно обосновать тем, что на снимках с аномалиями, которые использовались для обучения, присутствует вспомогательный для врачей текст, а на снимках, помеченных как нормы, такого текста нет. Поэтому есть предположение, что автоэнкодер может считать текст за аномалии.

В модели 2 производится выделение границ легких. Поэтому на изображении не остается лишней информации, которая могла бы определиться автоэнкодером как аномалия.

Заключение

В рамках данной выпускной квалификационной работы были выполнены следующие задачи:

- реализован алгоритм предобработки изображений: улучшается качество изображения выравниванием гистограммы и выделяются контуры легких оператором Собеля
- обучены состязательный и сверточный автоэнкодер с потерями 0.045 и 0.008 соответсвенно, обучена модель метода опорных векторов с параметрами C $= 10$, $\gamma = 0.01$
- для оценки качества классификатора были построены ROC-кривая и PR-кривая и посчитана площадь под ними. Полученные значения: AUC-ROC $= 0.93$, AUC-PR $= 0.91$.

Список литературы

[1] Ashfaqur Rahman Daniel Smith James Hills-Greg Bishop-Hurley-Dave Henry-Richard Rawnsley. A comparison of autoencoder and statistical features for cattle behaviour classification // Neural Networks (IJCNN). — 2016.

[2] Automatic differentiation in PyTorch / Adam Paszke, Sam Gross, Alban Chintala et al. — 2017.

[3] Bor-Chen Kuo Member IEEE Hsin-Hua Ho-Cheng-Hsuan Li-Chih-Cheng Hung Member IEEE, Jin-Shiuh Taur Senior Member IEEE. A Kernel-Based Feature Selection Method for SVM With RBF Kernel for Hyperspectral Image Classification // IEEE JOURNAL OF SELECTED TOPICS IN APPLIED EARTH OBSERVATIONS AND REMOTE SENSING. — 2014. — Vol. 7. — P. 317–326.

[4] Deng G., Cahill L. W. An adaptive Gaussian filter for noise reduction and edge detection. — 1993. — P. 1615–1619 vol.3.

[5] Igor Rafael S.Valente Paulo CésarCortez Edson Cavalcanti Neto-José Marques Soares-Victor Hugo C.de Albuquerque João Manuel R.S.Tavares. Automatic 3D pulmonary nodule detection in CT images: a survey // Computer Methods and Programs in Biomedicine. — February 2016. — Vol. 124. — P. 91–107.

[6] Jesse Davis Mark Goadrich. The relationship between Precision-Recall and ROC curves // ICML '06 Proceedings of the 23rd international conference on Machine learning. — 2006. — P. 233–240.

[7] Jin-Yu Zhang, Yan Chen, Xian-Xiang Huang. Edge detection of images based on improved Sobel operator and genetic algorithms. — 2009. — April. — P. 31–35.

[8] Jones Eric, Oliphant Travis, Peterson Pearu. SciPy: Open Source Scientific Tools for Python. — 2001. — URL: `http://www.scipy.org`.

[9] Jan Luts Fabian Ojeda Raf Van de Plasa-Bart De Moor-Sabine Van Huffel Johan A.K.Suykens. A tutorial on support vector machine-based methods for classification problems in chemometrics // Analytica Chimica Acta. — 2010. — Vol. 665. — P. 129–145.

[10] Kingma Diederik P., Ba Jimmy. Adam: A Method for Stochastic Optimization // CoRR. — 2014. — Vol. abs/1412.6980. — 1412.6980.

[11] Min Chen Senior Member Xiaobo Shi Yin Zhang-Senior Member IEEE-Di Wu Mohsen Guizani Fellow. Deep Feature Learning for Medical Image Analysis with Convolutional Autoencoder Neural Network // IEEE Transactions on Big Data. — 2017.

[12] Mingchen Gao Ulas Bagci Le Lu Aaron Wu-Mario Buty Hoo-Chang Shin Holger Roth Georgios Z. Papadakis Adrien Depeursinge Ronald M. Summers Ziyue Xu, Mollura Daniel J. Holistic classification of CT attenuation patterns for interstitial lung diseases via deep convolutional neural networks // Computer Methods in Biomechanics and Biomedical Engineering: Imaging & Visualization. — 2016. — Vol. 6. — P. 1–6.

[13] Nguyen Loc. Tutorial on Support Vector Machine // Applied and Computational Mathematics. — 2017. — Vol. 6. — P. 1–15.

[14] Itseez. — The OpenCV Reference Manual, 2.4.9.0 edition, 2014. — April. — URL: `http://opencv.org/`.

[15] Sakthivel K Jayanthiladevi A Kavitha C. Automфtic detection of lung cancer nodules by employing intelligent fuzzy cmeans and support vector machine // Biomedical Research, Computational Life Sciences and Smarter Technological Advancement. — 2016.

[16] Scikit-learn: Machine Learning in Python / G. Pedregosa, F. and Varoquaux, A. Gramfort, V. Michel et al. // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825–2830.

[17] Бондаренко А. Ю. Адамов В. Г. Анализ методов определения контуров изображения // Международный научно-исследовательский журнал. — 2015. — Vol. 8. — P. 13–16.

A. Конфигурация сервера

Обучение моделей провизводилось на сервере Microsoft Azure с конфигурацией:

- Model name: Intel(R) Xeon(R) CPU E5-2673 v4 @ 2.30GHz
- CPU MHz: 2294.683
- BogoMIPS: 4589.36
- L1 cache: 32K
- L2 cache: 256K
- L3 cache: 51200K
- RAM: 16G