Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 Информационные системы и технологии, геоинформационные системы
Отделение школы (НОЦ) Отдел информационных технологий

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

Тема работы
Алгоритмы нейросетевой сегментации снимков дистанционного зондирования поверхности Земли

УДК 004.932.1.032.26:528.8

Студент

Группа	ФИО	Подпись	Дата
8ИМ6Б	Аркалыков Ерболат Усенович		

Руководитель

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Друки А.А.	к.т.н.		

КОНСУЛЬТАНТЫ:

По разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Старикова Е.В.	к.ф.н.		

По разделу «Социальная ответственность»

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОКД	Волков Ю.В.	к.т.н.		

ДОПУСТИТЬ К ЗАЩИТЕ:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Шерстнев В.С.	к.т.н.		

Томск – 2018 г.

Код результатов	Результат обучения (выпускник должен быть готов)	Требования ФГОС ВО (ФГОС 3+), критерии АИОР
Общепрофессиональные компетенции		
Р1	Воспринимать и самостоятельно приобретать, развивать и применять математические, естественнонаучные, социально-экономические и профессиональные знания для решения нестандартных задач, в том числе в новой или незнакомой среде и в междисциплинарном контексте.	Требования ФГОС 3+ (ОПК-1, ПК 8-12, ОК-4), критерий 5 АИОР (п. 1.1), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р2	Владеть и применять методы и средства получения, хранения, переработки и трансляции информации посредством современных компьютерных технологий, в том числе в глобальных компьютерных сетях.	Требования ФГОС 3+ (ОПК-5, ПК-7, ОК-3), критерий 5 АИОР (п. 1.1, 1.2), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р3	Демонстрировать культуру мышления, способность выстраивать логику рассуждений и высказываний, основанных на интерпретации данных, интегрированных из разных областей науки и техники, выносить суждения на основании неполных данных, анализировать профессиональную информацию, выделять в ней главное, структурировать, оформлять и представлять в виде аналитических обзоров с обоснованными выводами и рекомендациями.	Требования ФГОС 3+ (ОПК-2,6, ПК-1, ОК-1), критерий 5 АИОР (п. 1.2), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р4	Анализировать и оценивать уровни своих компетенций в сочетании со способностью и готовностью к саморегулированию дальнейшего образования и профессиональной мобильности. Владеть, по крайней мере, одним из иностранных языков на уровне социального и профессионального общения, применять специальную лексику и профессиональную терминологию языка.	Требования ФГОС 3+ (ОПК-3,4, ПК-2,3, ОК-2), критерий 5 АИОР (п. 1.6, п. 2.2), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Профессиональные компетенции		
Р5	Разрабатывать стратегии и цели проектирования, критерии эффективности и ограничения применимости, новые методы, средства и технологии проектирования геоинформационных систем (ГИС) или промышленного программного	Требования ФГОС 3+ (ПК-1,2,3, ОПК-2, ОК-1), критерий 5 АИОР (п.1.3), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.

	обеспечения.	
Р6	Планировать и проводить теоретические и экспериментальные исследования в области создания интеллектуальных ГИС и ГИС технологии или промышленного программного обеспечения с использованием методов системной инженерии.	Требования ФГОС 3+ (ПК-7-13, ОПК-1, ОК-4), критерий 5 АИОР (п. 1.4), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р7	Осуществлять авторское сопровождение процессов проектирования, внедрения и сопровождения ГИС и ГИС технологий или промышленного программного обеспечения с использованием методов и средств системной инженерии, осуществлять подготовку и обучение персонала.	Требования ФГОС 3+ (ПК-4,17, ОПК-6, ОК-4,7), критерий 5 АИОР (п. 1.5), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р8	Формировать новые конкурентоспособные идеи в области теории и практики ГИС и ГИС технологий или системной инженерии программного обеспечения. Разрабатывать методы решения нестандартных задач и новые методы решения традиционных задач. Организовывать взаимодействие коллективов, принимать управленческие решения, находить компромисс между различными требованиями как при долгосрочном, так и при краткосрочным планировании.	Требования ФГОС 3+ (ПК-5,6,14,15,16, ОПК-1,2, ОК-4), критерий 5 АИОР (п. 1.6), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Общекультурные компетенции		
Р9	Использовать на практике умения и навыки в организации исследовательских, проектных работ и профессиональной эксплуатации современного оборудования и приборов, в управлении коллективом.	Требования ФГОС 3+ (ОК-4,7, ПК-8-12, ОПК-1,6), критерий 5 АИОР (п. 2.1, п. 2.3, п. 1.5), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р10	Свободно пользоваться русским и иностранным языками как средством делового общения.	Требования ФГОС 3+ (ОК-3, ПК-7, ОПК-4,5), критерий 5 АИОР (п. 2.2), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р11	Совершенствовать и развивать свой интеллектуальный и общекультурный	Требования ФГОС 3+ (ОК-1,5, ПК-1, ОПК-2),

	уровень. Проявлять инициативу, в том числе в ситуациях риска, брать на себя всю полноту ответственности.	критерий 5 АИОР (п. 2.4, п. 2.5) , соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.
Р12	Демонстрировать способность к самостоятельному обучению новым методам исследования, к изменению научного и научно-производственного профиля своей профессиональной деятельности, способность самостоятельно приобретать с помощью информационных технологий и использовать в практической деятельности новые знания и умения, в том числе в новых областях знаний, непосредственно не связанных со сферой деятельности, способность к педагогической деятельности.	Требования ФГОС 3+ (ОК-2,6, ПК-2,3, ОПК-3), критерий 5 АИОР (п. 2.6), соответствующий международным стандартам EUR-ACE и FEANI. Запросы студентов, отечественных и зарубежных работодателей.

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 Информационные системы и технологии, геоинформационные системы
Отделение школы (НОЦ) Отдел информационных технологий

УТВЕРЖДАЮ:
Руководитель ООП

_____ _____ _____
(Подпись) (Дата) (Ф.И.О.)

ЗАДАНИЕ
на выполнение выпускной квалификационной работы

В форме:

магистерской диссертации

Студенту:

Группа	ФИО
8ИМ6Б	Аркалыков Ерболат Усенович

Тема работы:

Алгоритмы нейросетевой сегментации снимков дистанционного зондирования поверхности Земли	
Утверждена приказом директора (дата, номер)	№2740/с от 19.04.2018 г.

Срок сдачи студентом выполненной работы:	03.06.2018

ТЕХНИЧЕСКОЕ ЗАДАНИЕ:

Исходные данные к работе *(наименование объекта исследования или проектирования; производительность или нагрузка; режим работы (непрерывный, периодический, циклический и т. д.); вид сырья или материал изделия; требования к продукту, изделию или процессу; особые требования к особенностям функционирования (эксплуатации) объекта или изделия в плане безопасности эксплуатации, влияния на окружающую среду, энергозатратам; экономический анализ и т. д.).*	Спроектировать и реализовать программу для сегментирования снимков дистанционного зондирования Земли.

Перечень подлежащих исследованию, проектированию и разработке вопросов *(аналитический обзор по литературным источникам с целью выяснения достижений мировой науки техники в рассматриваемой области; постановка задачи исследования, проектирования, конструирования; содержание процедуры исследования, проектирования, конструирования; обсуждение результатов выполненной работы; наименование дополнительных разделов, подлежащих разработке; заключение по работе).*	• Аналитический обзор и сводное тестирование библиотек для построения неронных сетей; • Разбор алгоритмов для семантической сегментации; • Проектирование и разработка нейросети; • Обучение нейросети.
Перечень графического материала *(с точным указанием обязательных чертежей)*	Презентация работы

Консультанты по разделам выпускной квалификационной работы

(с указанием разделов)

Раздел	Консультант
Финансовый менеджмент, ресурсоэффективность и ресурсосбережение	Старикова Е.В.
Социальная ответственность	Волков Ю.В.
Раздел на иностранном языке	Диденко А.В.

Названия разделов, которые должны быть написаны на русском и иностранном языках:

Разделы введение, аналитический обзор, разработка программы, финансовый менеджмент, ресурсоэффективность и ресурсосбережение, социальная ответственность и заключение должны быть написаны на русском языке.

Раздел аналитический обзор должен быть написан на английском языке

Дата выдачи задания на выполнение выпускной квалификационной работы по линейному графику	

Задание выдал руководитель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Друки А.А.	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ6Б	Аркалыков Е.У.		

Министерство образования и науки Российской Федерации
федеральное государственное автономное образовательное учреждение
высшего образования
**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Школа Инженерная школа информационных технологий и робототехники
Направление подготовки 09.04.02 Информационные системы и технологии, геоинформационные системы
Уровень образования магистр
Отделение школы (НОЦ) Отдел информационных технологий
Период выполнения весенний семестр 2017/2018 учебного года)

Форма представления работы:

магистерской диссертации

КАЛЕНДАРНЫЙ РЕЙТИНГ-ПЛАН
выполнения выпускной квалификационной работы

Срок сдачи студентом выполненной работы:	03.06.2018 г.

Дата контроля	Название раздела (модуля) / вид работы (исследования)	Максимальный балл раздела (модуля)
09.01.2018	*Постановка задачи и анализ предметной области*	*10*
22.01.2018	*Подбор и изучение материалов по тематике*	*5*
12.03.2018	*Проектирование архитектуры приложения*	*15*
16.04.2018	*Разработка приложения для семантической сегментации снимков*	*25*
30.04.2018	*Тестирование конечного программного обеспечения*	*10*
14.05.2018	*Финансовый менеджмент, ресурсоэффективность и ресурсосбережение*	*10*
21.05.2018	*Социальная ответственность*	*10*
28.05.2018	*Оформление пояснительной записки*	*10*
03.06.2018	*Обязательное приложение на иностранном языке*	*5*

Составил преподаватель:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Друки А.А.	к.т.н.		

СОГЛАСОВАНО:

Руководитель ООП	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОИТ	Шерстнев В.С.	к.т.н.		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА
«ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРСОЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ»

Студенту:

Группа	ФИО
8ИМ6Б	Аркалыков Ерболат Усенович

Школа	ИШИТР	Отделение	ОИТ
Уровень образования	Магистр	Направление/специальность	09.04.02 Информационные системы и технологии

Исходные данные к разделу «Финансовый менеджмент, ресурсоэффективность и ресурсосбережение»:	
1. Стоимость ресурсов научного исследования (НИ): материально-технических, энергетических, финансовых, информационных и человеческих 2. Нормы и нормативы расходования ресурсов 3. Используемая система налогообложения, ставки налогов, отчислений, дисконтирования и кредитования	Работа с информацией, представленной в российских и иностранных научных публикациях, аналитических материалах, статистических бюллетенях и изданиях, нормативно-правовых документах; анкетирование; опрос, наблюдение
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
1. Оценка коммерческого потенциала, перспективности и альтернатив проведения НИ с позиции ресурсоэффективности и ресурсосбережения	Проведение предпроектного анализа, оценка потенциальных потребителей, определение возможных альтернатив проведения НТИ
2. Планирование и формирование бюджета научных исследований	Определение структуры и трудоёмкости работ в рамках НТИ, разработка графика проведения НТИ, планирование бюджета НТИ.
3. Определение ресурсной (ресурсосберегающей), финансовой, бюджетной, социальной и экономической эффективности исследования	Расчёт интегрального показателя финансовой эффективности, интегрального финансового показателя, интегрального показателя ресурсоэффективности для всех видов исполнения НТИ
Перечень графического материала *(в том числе таблиц)***:**	
1. Перечень работ и продолжительность их выполнения 2. Трудозатраты на выполнение проекта 3. Линейный график работ 4. Расчет затрат на материалы 5. Затраты на вознаграждение 6. Затраты на электроэнергию 7. Смета затрат на разработку проекта 8. Оценки научно-технического уровня НИР	

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОСГН	Старикова Е.В.	к.ф.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ6Б	Аркалыков Ерболат Усенович		

ЗАДАНИЕ ДЛЯ РАЗДЕЛА «СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ»

Студенту:

Группа	ФИО
8ИМ6Б	Аркалыков Ерболат Усенович

Школа	ИШИТР	Отделение	ОИТ
Уровень образования	Магистр	Направление/специальность	09.04.02 Информационные системы и технологии

Исходные данные к разделу «Социальная ответственность»:	
1. Характеристика объекта исследования (вещество, материал, прибор, алгоритм, методика, рабочая зона) и области его применения	Программное обеспечение осуществляет сегментацию и распознавание объектов на снимках дистанционного зондирования. Может быть востребовано в системах автоматической идентификации объектов: - Здания; - Дороги; - Растительность; - Автомобили.
Перечень вопросов, подлежащих исследованию, проектированию и разработке:	
2. Профессиональная социальная безопасность 2.1. Анализ вредных и опасных факторов, которые может создать объект исследования. 2.2. Анализ вредных и опасных факторов, которые могут возникнуть на рабочем месте при проведении исследований. 2.3. Обоснование мероприятий по защите исследователя от действия опасных и вредных факторов	2.1. Вредные производственные факторы, создаваемые объектом исследования, в процессе разработки и реализации аналогичны и сводятся к использованию ПК: - Недостаточное освещение; - Нарушение параметров микроклимата; - Монотонность работы. Опасные производственные факторы, создаваемые объектом исследования: - Поражение электрическим током; - Возникновение пожара. 2.2. Мероприятия по защите от вредных факторов согласно нормативным документам: - СанПиН 2.2.4.548-96 - Гигиенические требования к микроклимату производственных помещений; - СанПиН 2.2.2/2.4.1340–03 - Гигиенические требования к персональным электронно-вычислительным машинам и организации работы; - СП 52.13330.2011 - Естественное и искусственное освещение; - ГОСТ Р 12.1.019-2009 ССБТ - Система стандартов безопасности труда; - СНиП 2.01.02-85 - Противопожарные нормы.
3. Экологическая безопасность 3.1. Анализ влияния объекта исследования на окружающую среду.	3.1. Влияния объекта исследования в процессе разработки и применения на окружающую среду:

3.2. Обоснование мероприятий по защите окружающей среды.	- Утилизация комплектующих ПК. 3.2 Данный программный продукт можно использовать для идентификации построек нарушающих экологическое законодательство.
4. Безопасность в чрезвычайных ситуациях 4.1. Анализ вероятных ЧС, которые может инициировать объект исследований и могут возникнуть на рабочем месте при проведении исследований. 4.2. Обоснование мероприятий по предотвращению ЧС и разработка порядка действия в случае возникновения ЧС. 4.3. Специфика комплекса.	4.1. Вероятные ЧС, инициируемые объектом исследования и возникающие на рабочем месте: - Пожары и взрывы; - Социальные чрезвычайные ситуации. 4.2. Мероприятия по предотвращению наиболее типичной ЧС – пожара, согласно нормативным документам: - НПБ 105-03; - ППБ 01–03. 4.3. Программный комплекс помогает при борьбе с терроризмом: - Помогает отслеживать автомобили.
5. Правовые и организационные вопросы обеспечения безопасности 5.1. Специальные (характерные для проектируемой рабочей зоны) правовые нормы трудового законодательства. 5.2. Организационные мероприятия при компоновке рабочей зоны.	5.1. Описание правовых норм для работ, связанных с работой за ПЭВМ согласно следующим документам: − Трудовой кодекс Российской Федерации" от 30.12.2001 N 197-ФЗ (ред. от 30.12.2015). 5.2. Влияние реализации программы на работу компаний: − Позволяет экономить время и усилия, затрачиваемые на распознавания данных вручную; − Экономия денежных средств; − Высокая точность и надежность распознавания минимизируют количество ошибок в данных; − Оснащение рабочего места и обучение рабочего персонала.

Дата выдачи задания для раздела по линейному графику	

Задание выдал консультант:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент ОКД	Волков Ю.В.	к.т.н.		

Задание принял к исполнению студент:

Группа	ФИО	Подпись	Дата
8ИМ6Б	Аркалыков Ерболат Усенович		

РЕФЕРАТ

Выпускная квалификационная работа состоит из 79 страниц, содержит 23 рисунка и 18 таблиц.

Ключевые слова: алгоритмы, нейронные сети, семантическая сегментация, дистанционное зондирование земли.

Объектом исследования является семантическая сегментация данных.

Цель работы проектирование и реализация программы для сегментирования снимков дистанционного зондирования Земли.

В ходе работы был произведён обзор методов и алгоритмов семантической сегментации, разработан алгоритм, основанный на нейронной сети. Была выполнена программная реализация алгоритма, позволяющая получать выходное изображение с контурами зданий, дорог, автомобилей и растительности, на основе снимка дистанционного зондирования земли.

Возможные области применения исследования: картография, градостроительство, анализ землеиспользования и т.д.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Семантическая сегментация объектов на изображениях – одно из самых интенсивно развивающихся направлений в области информационных технологий. Данные системы находят применение в самых разных областях — от систем обеспечения безопасности до различных видов медицинской диагностики. Распознавание объектов на изображениях – это назначение объекту определенного класса по существенным признакам, отделяющим данных объект от остальных.

Целью данной работы является проектирование и реализация программы для сегментирования снимков дистанционного зондирования Земли.

Для достижения поставленной цели, необходимо было решить следующие задачи:

1. Выбрать библиотеку для реализации нейронной сети.
2. Разработать архитектуру нейронной сети.
3. Обучение нейронной сети.
4. Получить результаты.

1 АНАЛИТИЧЕСКИЙ ОБЗОР

1.1 Обзор алгоритмов семантической сегментации изображений

Основным подходом для семантической сегментации изображений является совокупность трёх алгоритмов: детекторов, дескрипторов и классификаторов. Данные алгоритмы определяют основные параметры изображения, выделяют объекты и классифицируют их. К основным параметрам изображения можно отнести яркость, цвет, текстуру, границы и углы объектов и тому подобное. Параметры изображения фиксируются с помощью алгоритмов-детекторов, после чего математически описываются при помощи алгоритмов-дескрипторов при этом получается выделение объектов на изображении. Далее необходимо определить к какому классу относятся объекты, этой процедурой занимаются алгоритмы классификаторы.

1.1.1 Детекторы и дескрипторы

Детектор SIFT.

Детектор SIFT (Scale invariant feature transform – масштабно независимое преобразование особенностей) основывается на использовании масштабируемых пространств – набора всевозможных, сглаженных определённым фильтром версий одного изображения. При использовании гауссового фильтра, данное масштабируемое пространство становится инвариантно к сдвигам, вращениям и не смещающему локальные экстремумы масштабу. Для улучшения инвариантности относительно масштаба, детектор использует исходное изображение, взятое в разных масштабах, путём построения пирамиды гауссианов: всё масштабируемое пространство разбивается на так называемые октавы, таким образом, чтобы следующая октава занимала в два раза больше пространства, чем предыдущая. В каждой октаве строится некоторое количество гауссианов изображения, из которых вычитаются разности гауссианов, составляющих отдельную пирамиду (рис. 1) [1]

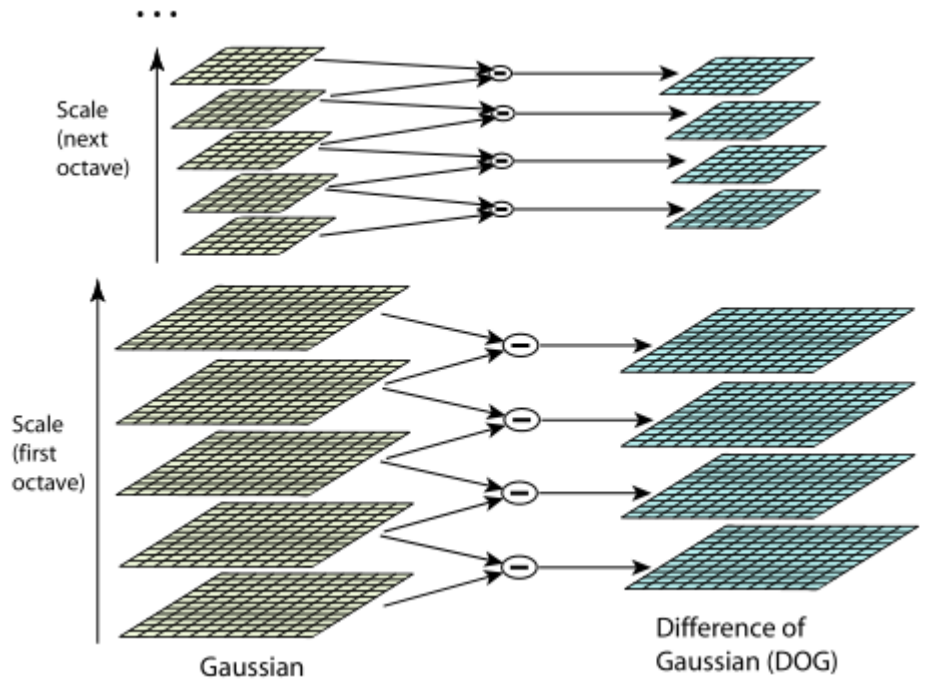

Рисунок 1 - Пирамиды гауссианов и разности гауссианов

Далее остаётся только определить особые (ключевые) точки. Главный критерий особой точки заключается в том, что она должна являться локальным экстремумов разности гауссианов. Для нахождения особых точек используется метод, показанный на рисунке 2. Для каждой точки изображения разности гауссианов проходит сравнение с соседними точками как в этом изображении, так и в соседних. Соответственно, если эта точка больше или меньше них, то она считается локальным экстремумом.

Далее происходит проверка этой точки: при помощи многочлена Тейлора, где выявляется смещение от точного экстремума; значения контраста самой разности гауссианов; для нахождения точки на границе объекта при помощи матрицы Гессе.

После этого вычисляется ориентация ключевой точки исходя из направления градиентов соседних точек. Теперь данная точка готова для описания её дескриптором.

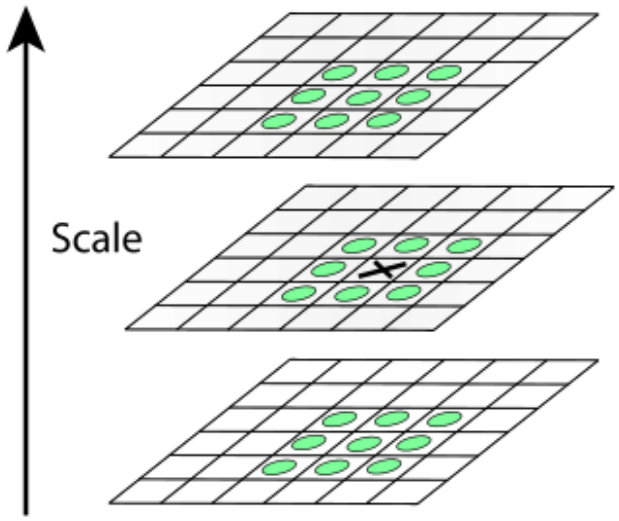

Рисунок 2 - Определение локального экстремума на пирамиде разности гауссианов

Детектор SURF.

Детектор SURF (Speeded up robust features – ускоренное выявление устойчивостей) является модернизацией детектора SIFT, но вместо функции Гаусса используется её приближение прямоугольным фильтром 9x9, что ускоряет получение результата алгоритма. Соответственно делит те же достоинства и недостатки, что и детектор SIFT. [1]

Детектор FAST.

Алгоритм FAST (Features from accelerated segment test – особенности из ускоренного теста сегментов) не требует вычисления производных по яркости, а сравнивается сама яркость в окружности от проверяемой точки. Сначала проводится быстрый тест, четырёх точек от исследуемой, а затем проверяются остальные. При этом в полном переборе не учитываются значения быстрого теста. Количество проверок и их последовательность определяются на обучающей выборке, а наличие угла определяются всего несколькими десятками проверок. Таким образом, алгоритм работает достаточно быстро, чтобы применятся в системах реального времени. [2,3]

Детектор MSER.

Алгоритм MSER (Maximally stable extremal regions – максимально стабильные экстремальные области). Идея алгоритма заключается в

определении интенсивности пикселов изображения и сравнение их с некоторым порогом (если интенсивность пикселя больше порога, считаем его белым, иначе чёрным). Таким образом, строятся пирамиды изображений, в начале которой стоят белые изображения, а в конце чёрные. Такая пирамида позволяет построить множество связных компонент интенсивности, которые будут инвариантны к аффинным преобразованиям. Данный алгоритм стабильно работает, но имеет неполную инвариантность относительно масштаба.

Дескриптор SIFT.

Помимо детектора, существует специальный дескриптор SIFT, который представляет из себя локальную гистограмму направлений градиентов изображения. Выходом дескриптора является вектор значений, описывающих окрестности этой ключевой точки.

Принцип его работы заключается в определении градиента изображения, его направления и модуля умноженного на вес, для каждого из четырёх секторов вокруг особой точки. Для каждого сектора составляется гистограмма направлений градиента, каждое вхождение которого взвешено модулем этого градиента.

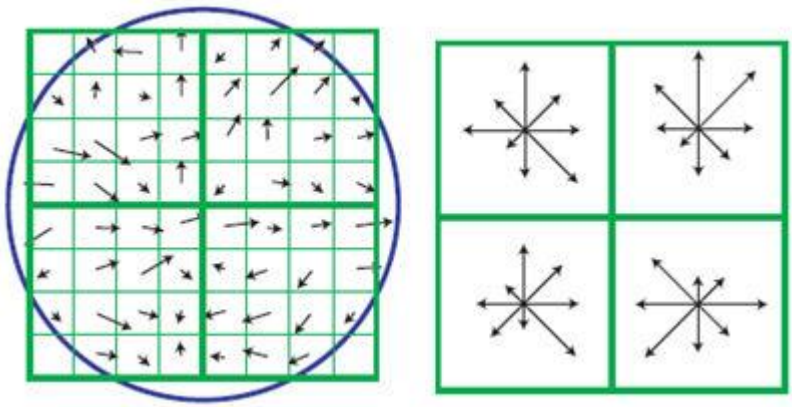

Рисунок 3 - Построение дескриптора SIFT

Таким образом, основными достоинствами связки детектора и дескриптора SIFT являются: инвариантность относительно масштабу и повороту изображения, устойчивость к изменению освещения и вычислительная эффективность. [1,2,3]

Дескриптор SURF.

Данный дескриптор похож на дескриптор SIFT, но вместо использования гистограмм взвешенных градиентов используются отклики исходного изображения на вейвлеты Хаара. Квадратная область строится вокруг точки интереса и ориентируется относительно предпочтительного направления и разделяется на квадратные сектора, в которых вычисляются отклики на вейвлеты Хаара, направленные вертикально и горизонтально направленные. Данные отклики взвешиваются и суммируются по каждому из секторов, образуя первую часть дескриптора.

Для того, чтобы учесть не только сам факт изменения яркости от точки к точке, но и сохранить информацию по направлению изменения яркости, используются суммы модулей яркости. Как и SIFT дескриптор, SURF инвариантен к изменению яркости и масштабу. [1,3]

Дескриптор HOG.

HOG (Histogram of oriented gradients – гистограмма ориентированных градиентов) дескриптор ключевых точек, основанный на подсчёте направлений градиента в локальных областях изображения. Алгоритм его работы схож с алгоритмом SIFT, но отличается тем, что вычисление ведётся на плотной сетке равномерно распределённых ячеек. Также этот алгоритм использует нормализацию локального контраста с целью увеличить точность.

Основывается на допущении, что внешний вид и форма объекта может быть описана при помощи распределения градиентов интенсивности или их направлений. Поэтому при работе алгоритма, изображение разделяется на небольшие связные области, которые называются ячейками, и для каждой ячейки рассчитывается гистограмма направлений градиентов и направлений краёв для пикселов внутри ячейки. Выходом дескриптора является комбинация этих гистограмм. К достоинствам этого алгоритма можно отнести то, что при работе с локальными клетками, он обладает низкой чувствительностью к геометрическим и фотометрическим преобразованиям. [1,2]

1.1.2 Классификаторы

В машинном обучении представлено множество различных алгоритмов классификации и их модификаций, поэтому рассмотрим два класса из широко используемых классификаторов: мешок слов и машину опорных векторов.

Bag of words

Данный алгоритм относится к числу наиболее распространённых классов алгоритмов классификации изображений. Его название во многом определяет принцип его работы, ведь фактически он использует гистограмму вхождений отдельных шаблонов в изображение. [4]

В основном этот алгоритм использовался для классификации текстов, в котором также строилась гистограмма вхождения в документ слов из заранее заготовленного словаря, но может эффективно применяться и для других задач.

Основные шаги данного алгоритма:

Определить особые (ключевые) точки на изображении

Построить дескрипторы этих точек

Провести кластеризацию этих дескрипторов, принадлежащих к объектам обучающей выборки (то есть заполнить словарь «словами»)

Построить описание каждого изображения в виде нормированной гистограммы встречаемости «слов» (вычисление для каждого из кластеров количества отнесённых к нему особых точек изображения)

Построение классификатора, использующего описание с шага 4.

Для первых двух шагов могут быть использованы любые детекторы и дескрипторы, с тем условием, что они устойчивы к аффинным преобразованиям и изменениям в освещённости.

Основные недостатки данного классификатора в большом размере словаря, причём его размер не должен превышать некоторого значения, при котором он будет устойчив к шуму. Также, мешок слов никак не учитывает пространственную информацию об объекте, что при наличии схожих по дескрипторам особых точек разных объектов на изображении их описания могут совпасть. [5]

Машина опорных векторов

Машина опорных векторов (SVM – support vector machine) одна из наиболее популярных методик обучения классификаторов. Изначально данный метод применялся для задач бинарной классификации, но он легко обобщается и для задач с большим количеством классов, а также на задачи восстановления регрессии. Для обучения алгоритма предлагается строить линейный пороговый классификатор

$$\operatorname{sgn}\left(\sum_{j=1}^{n} w_j x_j - w_0\right). \quad (1)$$

Уравнение $\langle w, x\rangle = w_0$ описывает гиперплоскость, разделяющую классы. Если предположить, что выборка линейно разделима, то становится очевидно, что данная плоскость не единственная и существуют другие её положения, также разбивающие выборку. [4]

Идея метода заключается в выборе таких w и w_0которые были бы оптимальны с точки зрения какого-либо конкретного критерия. Изначально, метод классификации возник из эвристических предположений, но сейчас имеет теоретическое обоснование.

Для того, чтобы гиперплоскость более устойчиво разделяла классы, она должна отстоять от точек выборки на максимальном расстоянии. Это достигается, когда норма вектора w минимальна. Опорными векторами же называются множества, лежащие на границах областей, разделённых гиперплоскостью.

Алгоритм также легко обобщается для линейно неразделимых данных путём перехода из исходного пространства признаковых описания объектов к другому посредству большей размерности при помощи ядер функций.

К достоинствам данного алгоритма помимо распространённости и простоты можно отнести небольшую обучающую выборку, при которой классификатор будет выдавать приемлемый результат. Это же является и недостатком – алгоритм использует не всё множество, а лишь небольшую их часть на границах областей. [4, 5]

1.2 Искусственные нейронные сети

Искусственные нейронные сети (ИНС) – алгоритмы, моделирующие способ обработки информации человеческим мозгом. Представляют из себя распределённый параллельный процессор, состоящий из элементарных частиц обработки информации (нейронов) и связей между ними (синаптические веса), позволяющих накапливать знания из окружающей среды и использовать в процессе обучения. [6]

Искусственные нейронные сети состоят из слоёв нейронов и обладают различной архитектурой. Классическими сетями являются полносвязные, т.е. сети, в которых каждый нейрон одного слоя соединён с нейронами предыдущего слоя. Обработка информации нейроном в таких сетях ведётся посредством умножения всех входных сигналов на соответствующие синаптические связи, их суммирования и обработки функцией активации нейрона, после чего сигнал распространяется дальше по сети. Обучение сети заключается в перенастройке синаптических весов в зависимости от выхода сети.

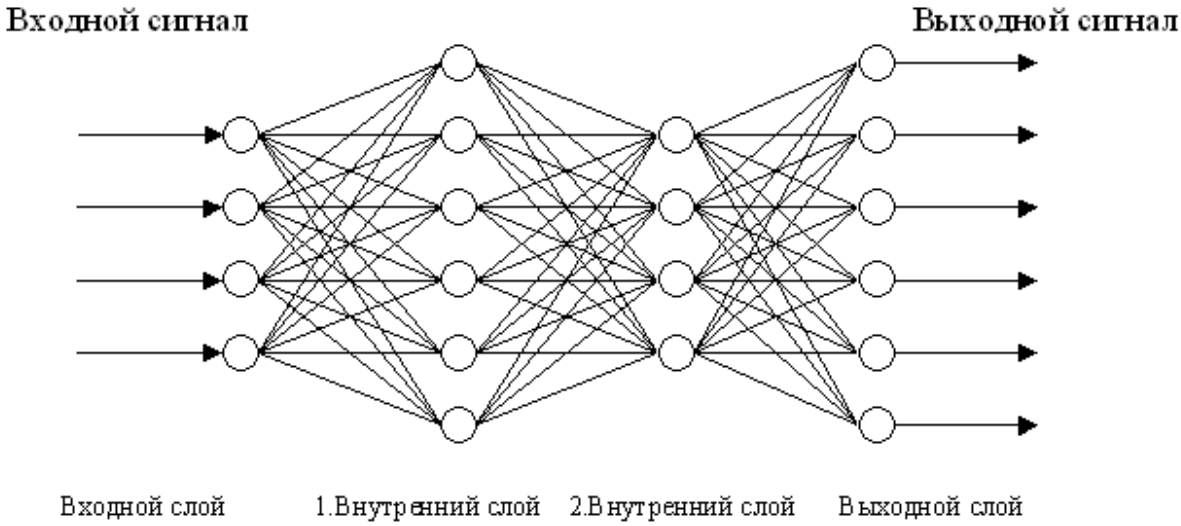

Рисунок 4 - Полносвязная ИНС с двумя внутренними слоями

Основным недостатком классических нейронных сетей для обработки изображений является большой размер входного вектора (т.е. каждый пиксель изображения). В следствии этого растёт количество нейронов каждого слоя и

для больших изображений она становится слишком объёмной и тяжёлой для обучения. Также классические нейронные сети не могут учесть топологию исходного изображения, так как принимают его целиком. [7]

Этих недостатков лишены свёрточные искусственные нейронные сети, которые обладают специальными свёрточными и субдискретизирующими слоями. Основная идея этих слоёв – создание разделяемых весов, т.е. часть нейронов использует одни и те же весовые коэффициенты и объединяются в карты признаков, каждый нейрон которой связан с предыдущим слоем. Каждый нейрон такого слоя выполняет операцию математической свёртки некоторой области предыдущего слоя и такой слой соответственно называется свёрточным и моделирует некоторые особенности человеческого зрения, отвечая за обнаружение конкретного признака. Слои субдискретизации отвечают за уменьшение входного вектора в некоторое количество раз (обычно в 2 раза).

Таким образом, свёрточные нейронные сети обладают гораздо меньшим количеством настраиваемых весовых коэффициентов, что позволяет сети обучатся обобщению информации, а не запоминанию. Такие сети устойчивы к изменению масштаба, сдвигу и повороту.

1.2.1 Нейронная сеть

Биологическая нейронная сеть обеспечивает возможность эффективного взаимодействия человека с окружающей средой путём выполнения различных задач: распознавания образов; обработки сигналов органов чувств; моторики и многих других. При этом, данные задачи зачастую выполняются многократно и последовательно в пределах сотен миллисекунд, что намного быстрее, чем решение аналогичных задач на современных компьютерах.

Это связано с тем, что способы обработки информации человеческим мозгом сильно отличаются от методов обработки информации вычислительных машин, благодаря особой структуре человеческого мозга, состоящей из большого числа *нейронов* (около 10 миллиардов), связанных в единую сеть.

Каждый нейрон состоит из ядра, тела клетки и отростков (*дендритов*, принимающих входные сигналы, и *аксонов*, передающих), соединяющих их друг с другом, и основная задача каждого из них – передать электрохимический импульс по всей сети. Это обеспечивается тем, что биологическая нейронная сеть обладает большой степенью связности – каждый нейрон соединён несколькими тысячами других и каждое такое соединение обладает *силой синаптической связи*, определяющей усиление или ослабление импульса при передаче между этими нейронами. Таким образом, структура человеческого мозга не статична и постоянно изменяется, настраивая эти связи и позволяя строить собственные правила, основываясь на «опыте». Иными словами, структура человеческого мозга позволяет обучатся. [7,8]

1.2.2 Модель нейрона

Нейрон – основная единица обработки информации в ИНС. Представляет из себя простой процессор, преобразующий некоторые входные сигналы в выходной по определённому правилу. Данный выходной сигнал поступает на другие нейроны по взвешенным связям. Общая структура представлена на рисунке 5.

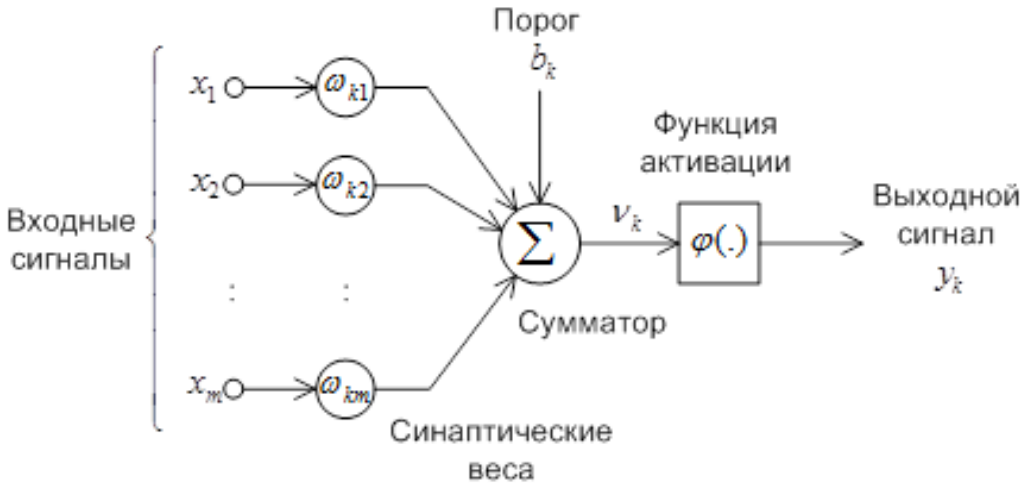

Рисунок 5 - Модель нейрона

В модели нейрона можно выделить три основные элемента [6]:

1. Наличие взвешенных связей – *синапсов*, обладающих весом. Таким образом, каждый входной сигнал x_j, поступающий на вход j нейрона k, умножается на вес ω_{kj}. В отличие от биологической нейронной сети, эти веса могут быть как положительными, так и отрицательными.
2. *Сумматор* – складывает входные сигналы с учётом весов синапсов. В модели также присутствует *пороговый элемент*, отражающий усиление или ослабление сигнала, подаваемого на функцию активации.
3. *Функция активации (сжатия)* – функция, ограничивающая амплитуду выходного сигнала. Обычно регулирует сигнал в интервалах от 0 до 1 или от -1 до 1.

Опишем функционирование нейрона при помощи математических уравнений:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \; ; \tag{2}$$

$$y_k = \varphi(u_k + b_k) \; , \tag{3}$$

где $x_1, x_2, \dots, x_m$ – входные сигналы; $w_{k1}, w_{k2}, \dots, w_{km}$– синаптические веса нейрона; u_k – линейная комбинация входов; $\varphi()$ – функция активации; y_k – выход нейрона. Постсинаптический потенциал вычисляется по формуле 3 и объясняется эффектом аффинного преобразования от порогового элемента b_k.

$$v_k = u_k + b_k \; . \tag{4}$$

Данный порог является внешним параметром для нейрона k и может быть преобразован, как новый синапс со входным сигналом $x_0 = +1$ и весом $w_{k0} = b_k$. Таким образом, уравнения 1 и 2 придут к виду:

$$u_k = \sum_{j=0}^{m} w_{kj} x_j \; ; \tag{5}$$

$$y_k = \varphi u_k \; . \tag{6}$$

1.2.3 Функции активации

Функции активации (сжатия, возбуждения) – представляет из себя функцию, вычисляющую выходной сигнал искусственного нейрона. Одними из самых распространённых типов функций являются:

1. *Пороговая функция* или функция единичного скачка. Соответственно, если входное значение ниже порогового, то значение функции активации равно минимуму, иначе – максимальному. Иногда называется функцией Хэвисайда. Математическое описание:

$$\varphi(v)=\begin{cases}1, & v\geq 0\\ 0, & v<0\end{cases}. \qquad (7)$$

Линейный порог или гистерезис. Эта функция кусочно-заданная и имеет три интервала, два из которых являются минимальным и максимальным порогом, а третий возрастает. [6,7]

$$\varphi(v)=\begin{cases}1, & v\geq +\frac{1}{2}\\ |v|, & +\frac{1}{2}>v>-\frac{1}{2}\\ 0, & v\leq -\frac{1}{2}\end{cases}. \qquad (8)$$

Здесь коэффициент усиления в линейно области оператора предполагается равным единицу и это функцию можно рассматривать как аппроксимацию нелинейного усилителя. Имеется две особые формы: *линейный сумматор* – линейная область оператора не достигает порога насыщения; *пороговая функция* – если коэффициент усиления линейно области принят бесконечно большим. [6]

2. *Сигмоидальная функция.* Является самой распространённой при создании ИНС. Принимает не только строгие значения 0 или 1, но и

бесконечное множество в интервале от 0 до 1. Примером такой функции может стать *логистическая функция*:

$$\varphi(v)=\frac{1}{1+\exp(-av)}\ , \tag{9}$$

где a – параметр наклона сигмоидальной функции. Настраивая этот параметр, можно изменить крутизну самой функции. В некоторых ситуациях требуется симметричность относительно начала координат, соответственно область значений функции должна находиться на отрезке от -1 до +1. Такая функция называется *сигнум* и её примером может стать *гиперболический тангенс*:

$$\varphi(v)=\tanh(v)\ . \tag{10}$$

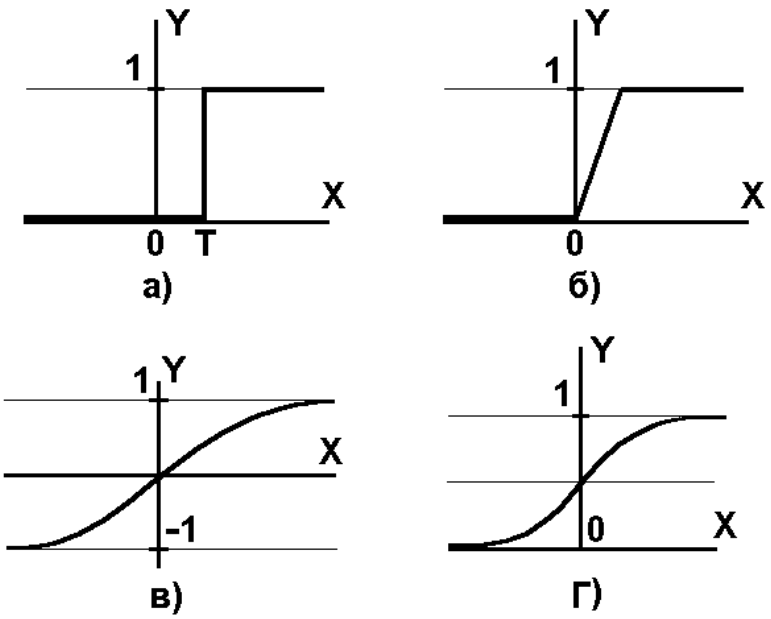

Рисунок 6 - Виды функций активаций: пороговая функция (а); гистерезис (б); гиперболический тангенс (в); логистическая функция (г)

Рассмотренные выше типы функций преобразуют входной сигнал в выходной по точно определённым для всех значениях этого сигнала. Однако, может возникнуть необходимость использовать вероятностный подход и перейти к стохастической модели нейрона. В такой модели нейрон может

находиться в состоянии -1 и +1 и переключатся с некоторой вероятностью наступления такого события. Тогда описать такую модель можно будет следующей формулой [6]:

$$\varphi(v)=\begin{cases} +1, & \text{с вероятностью } P(v) \\ -1, & \text{с вероятностью } 1-P(v). \end{cases} \quad (11)$$

где x – состояние нейрона; $P(v)$ – вероятность активации нейрона. Данная вероятность также описывается сигмоидальной функцией:

$$P(v)=\frac{1}{1+\exp\left(-\frac{v}{T}\right)}, \quad (12)$$

где T – описывает эффект синаптического шума. Если этот параметр стремиться к нулю, то стохастический нейрон примет пороговую форму активации.

1.2.4 Архитектуры ИНС

Выделим два основных класса нейросетевых структур.

Сети прямого распространения (ацикличные)

Полносвязные сети. В таких сетях распространение информации происходит от нейронов *входного слоя* и передаётся далее к нейронам *выходного слоя*. В простейшем случае такие сети имеют один слой нейронов, который является выходным – эти сети называют *однослойными*. Если сеть имеет дополнительные слои, которые называют *скрытыми слоями*, то такая сеть называется *многослойной* и чем больше скрытых слоёв, тем больше синаптических связей и взаимодействие нейронов, тем самым больше возможностей для обработки больших данных. Одной из первых и простых нейронных сетей является *перцептрон* – однослойная ИНС прямого распространения с пороговой функцией активации. [6]

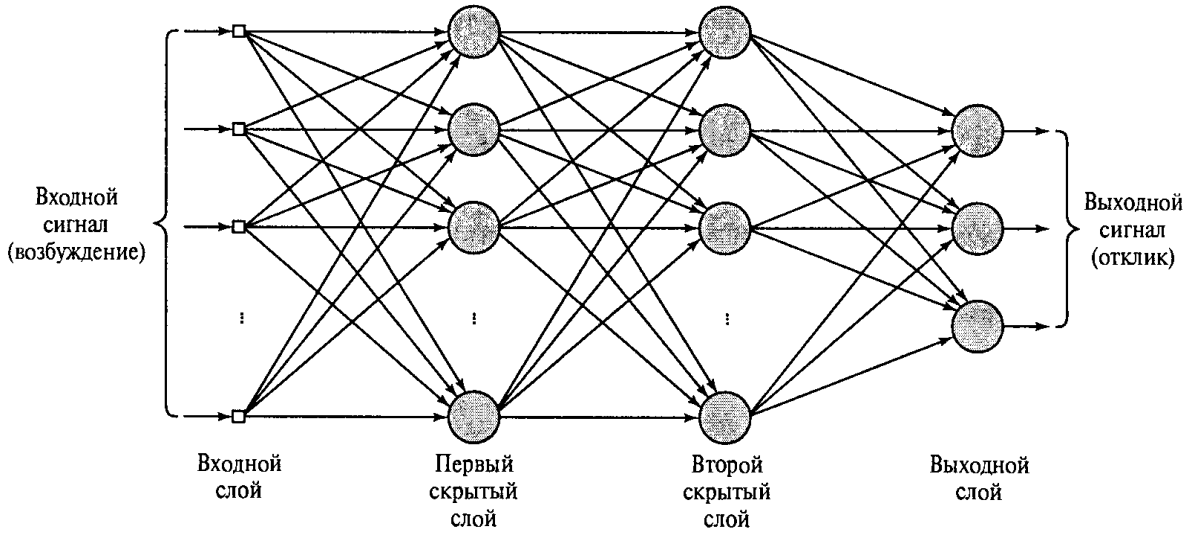

Рисунок 7 - ИНС прямого распространения с двумя скрытыми слоями

Свёрточные сети. В полносвязных сетях нейроны каждого слоя связаны со всеми предыдущими нейронами, что бывает неудобно или не эффективно при решении некоторых задач. Свёрточные нейронные сети – особый вид нейросетей влючающий в своей структуре *свёрточные слои, слои субдискретизации* и *полносвязные слои*.

Такая архитектура сетей включается в себя три парадигмы: *локальное восприятие*; *разделяемые веса*; *субдискретизация*. [6,7]

Под локальным восприятием подразумевается, что на вход одного нейрона поступает не все выходы предыдущего слоя, а лишь некоторая определённая их часть.

Под разделяемыми весами подразумевается, что для большинства связей используется небольшой набор весов, называемых *ядрами*. Ядро представляет из себя матрицу небольших размеров, которая применяется ко входному вектору или слою нейронов. Например, если входной вектор является изображением, состоящим из пикселей, то ядро применяется к изображению посредством математической операции свёртки. Суть этой операции заключается в поэлементном умножении фрагмента изображения на матрицу ядра, суммирование полученных значений и запись результата в аналогичную позицию выходного изображения, которое называется *картой признаков*, так

как свёртка ядром будет давать степень схожести фрагмента с фильтром или признаком чего-либо на изображении.

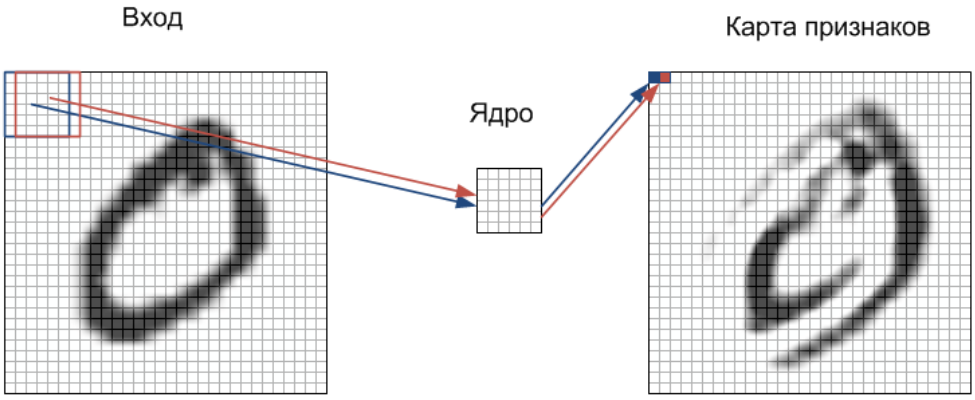

Рисунок 8 - Применения ядра к изображению

Соответственно, слой сети, выполняющий операцию свёртки, называется свёрточным.

Суть субдискретизации заключается в уменьшении пространственной размерности изображения, обычно в 2 раза. То есть исходное изображение усредняется, обеспечивая инвариантность относительно масштаба. Соответственно. Слои субдискретизации занимаются этой операцией. [7,8]

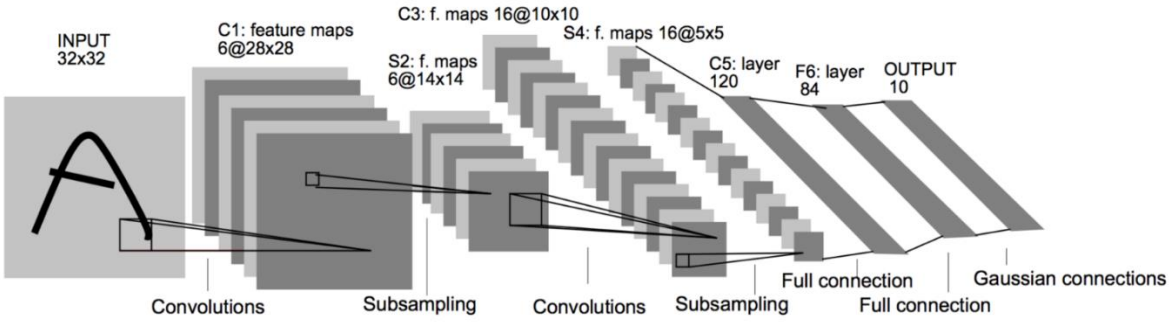

Рисунок 9 - Структура свёрточной нейронной сети

Рекуррентные сети

Это сети, обладающие хотя бы одной *обратной связью*. Понятие обратной связи характерно для систем, в которых выходной сигнал влияет на входные сигналы. Обратные связи существуют в нервной системе почти любого животного и позволяют усиливать внешние сигналы, сигналами, циркулирующими внутри системы. Таким образом, в рекуррентных сетях

нейроны могут обмениваться информацией друг с другом и это важное отличие от сети прямого распространения – обратные связи позволяют организовать некоторое подобие памяти, что позволяет хранить некоторую информацию о предыдущем состоянии сети, а, следовательно, позволяет анализировать последовательности данных, для которых важен порядок их значений, к примеру, музыкальные композиции или состояние биржевого рынка. Одной из первых рекуррентных ИНС была *сеть Хопфилда*, практически реализующая одну ячейку ассоциативной памяти. Такая сеть состояла из одного скрытого слоя связанных друг с другом нейронов. [6,7]

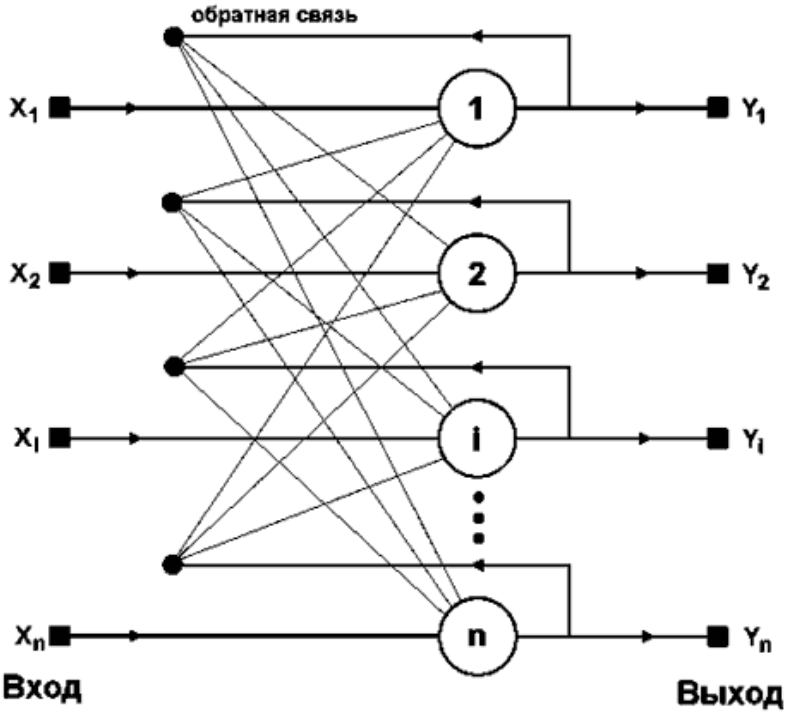

Рисунок 10 - Рекуррентная сеть Хопфилда

Сети радиально-базисных функций. Такие функции образуют один из простейших классов и могут быть использованы в разных сетях (как многослойных, так и однослойных)

1.2.5 Обучение ИНС

Способность обучаться на основе внешних данных и тем самым повышать собственную производительность является одой из важных особенностей ИНС.

Обучение – процесс, в котором свободные параметры нейронной сети настраиваются посредством моделирования среды, в которую эта сеть встроена. Тип обучения определяется способом подстройки этих параметров. [6]

Исходя из определения можно выделить последовательность при обучении ИНС: подача стимулов из внешней среды; изменение свободных параметров сети; изменение ответа сети на последующие возбуждения. Данная последовательность называется *алгоритмом обучения*, каждый из которых отличается способом настройки синаптических весов нейронов и способом связи обучаемой нейросети с окружающим миром. Алгоритмы обучения итеративны и каждая итерация называется *эпохой*.

Существует две основные парадигмы обучения нейросети: обучение с учителем и без него.

Обучение с учителем

Концепция такого обучения заключается в том, что учитель готовит для нейросети специальных пары *входы и эталонные выходы*. Предположим, что из окружающей среди поступает некоторые входные значения. Обладая знаниями, учитель может сформировать желаемый отклик на эти входные значения и передать ИНС. Параметры сети будут корректироваться на основе тих значений и значений *сигнала ошибки* – разности между желаемым сигналом и текущим выходным сигналом сети. Процесс корректировки является пошаговым и необходим для имитации действий учителя, что позволяет передать знания учителя в полном объёме. После окончания обучения, ИНС может действовать самостоятельно. [6,7]

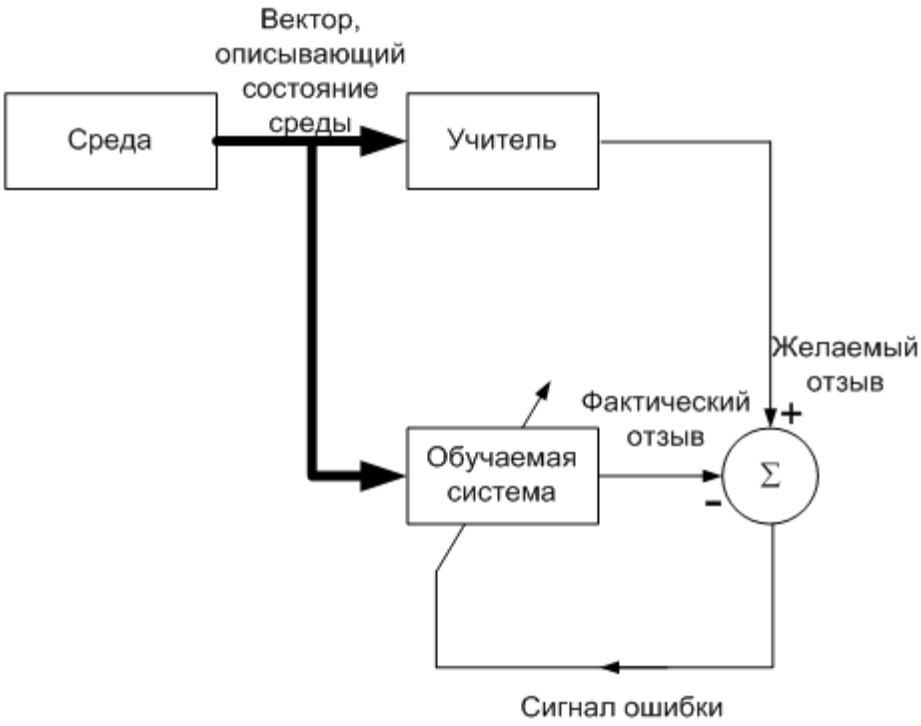

Рисунок 11 - Диаграмма обучения с учителем

Обучение без учителя

При таком обучении, как видно из названия, обучение сети и настройка весовых коэффициентов происходит без контроля учителя. Такой поход не включает в себя маркированных примеров.

Обучение с подкреплением (*нейродинамическое программирование*). При таком обучении генерация выходных сигналов осуществляется со взаимодействием с внешней средой. На рисунке 6 представлена диаграмма, описывающая данный алгоритм. Существует две важные функции: *оценки* и *подкрепления*. Функция оценки определяет, какой результат приемлем в продолжительном периоде, в то время как функция подкрепления решает, что приемлемо в данный момент. При таком обучении сеть запоминает соответствия между ситуациями и действиями, которые должны выполнятся. [6,8]

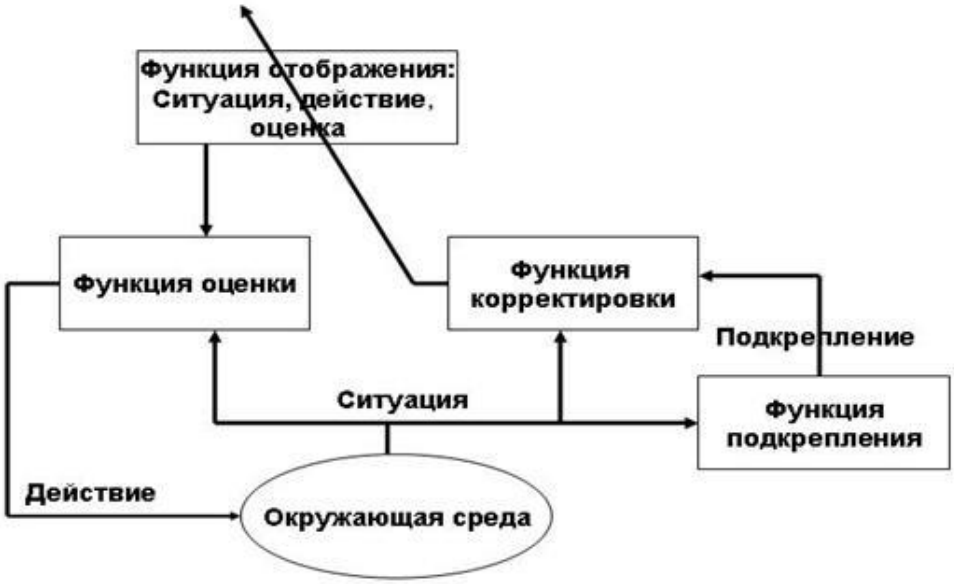

Рисунок 12 - Диаграмма обучения с подкреплением

Методы обучения

Алгоритмы обучения подразделяются на детерминированные и стохастические:

Детерминированные – настройка параметров ИНС ведётся итеративно и основывается на текущем состоянии сети.

Стохастические – настройка параметров ИНС ведётся случайным образом, но сохраняются лишь те изменения, что привели к улучшению работы нейронной сети.

Метод Хэбба. Один из самых первых методов обучения, основанный на биологических исследованиях, предполагающий, что прочность связи между нейронами усиливается, если оба нейрона одновременно активны. Таким образом, изменение веса связи зависит только от пары нейронов – при синхронном возбуждении связь упрочняется, а при асинхронном наоборот.

При начале обучения данным алгоритмом всем весовым коэффициентам присваиваются некоторые случайные значения. При подаче входного сигнала, происходит обработка и выход, на основании которого производится изменение весовых коэффициентов по описанному выше правилу. [6,9]

Конкурентное обучение. При данном методе обучения выходные нейроны не могут быть активированы одновременно, как в сети, обученной методом Хэбба, а соревнуются за право быть активированными. Причём синаптические веса распределяются так, что входные данные приводят к различной реакции разных нейронов. Таким образом, каждый выходной нейрон в таких сетях отвечает за некоторую группу или класс признаков и является детектором признака. Перенастройка весов производиться только для победившего в соревновании нейрона. [7,8]

Коррекция ошибок. Модель использует алгоритм обучения с учителем, где входные данные сопровождаются желаемыми выходными. Для настройки весов производится сравнение полученных выходных данных с желаемыми. Рассмотрим один из них:

Алгоритм обратного распространения ошибки. Своё название алгоритм получил благодаря тому, что ошибка, вычисляемая на каждой итерации, распространяется по ИНС от выхода ко входу с целью перенастройки синаптических весов. В процессе обучения сети, при подаче входного вектора, выход сети сравнивается с выходом из обучающей выборки, формируя ошибку $\delta = d - y$, где d – эталонный выход, а y – выход сети. Данная ошибка обусловлена не только погрешностью выходных нейронов, но и всех скрытых нейронов. [9]

Корректировка синаптических весов производится по правилу Видроу – Хоффа (дельта правило):

$$\Delta w_i = \eta \delta x_i , \quad (13)$$

где η – коэффициент скорости обучения; x_i – значения входа; δ – ошибка.

Новый синаптический вес рассчитывается по следующей формуле:

$$w_i(k+1) = w_i(k) + \Delta w_i , \quad (14)$$

где k – номер итерации обучения.

Для выходной ошибки используют обычно среднеквадратическую:

$$E=\frac{1}{2}\sum_{j}(y_j-d_{jj})^2. \quad (15)$$

Тогда в соответствии с градиентным спуском получим:

$$\Delta w_{ij}=-\eta\frac{\partial E}{\partial w_{ij}}, \quad (16)$$

где w_{ij} – текущий вес связи i нейрона j.

Покажем, что для линейного нейрона справедливо

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial w_{ij}}. \quad (17)$$

Тогда подставив (17) в (16) получим, что

$$\Delta w_{ij}=-\eta\frac{\partial E}{\partial w_{ij}}\frac{\partial y_i}{\partial w_{ij}}. \quad (18)$$

В следствии этого и (13) получится, что

$$\frac{\partial E}{\partial y_j}=-\delta_j. \quad (19)$$

Когда выход нейрона формируется при помощи активационной функции ($f(S)$) выражения (17) и (19) примут вид:

$$\frac{\partial E}{\partial w_{ij}}=\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}\frac{\partial S_j}{\partial w_{ij}};\ \delta_j=-\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}. \quad (20\text{-}21)$$

Продифференцировав выражение (15) и учитывая формулы (20-21) получаем:

$$\delta_j=\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}=(y-d_j)f'(S). \quad (22)$$

Все сомножители правой части выражения (20) известны и определяют величину коррекции веса (23), а с учётом того, что синаптические веса меняются в направлении функции ошибки, получим направление обратное по знаку производной функции ошибки (24).

$$\frac{\partial E}{\partial w_{ij}}-(y-d_j)f'(S)x_i;\ \Delta w_{ij}=\delta_j x_i. \quad (23\text{-}24)$$

Полученная ошибка δ_j является ошибкой выходного нейрона. Чтобы получить ошибку скрытого нейрона их вес нужно корректировать в соответствии с формулой (24) по вкладу в ошибку следующего слоя. Тем самым, чем больше ошибка на следующем слое и чем больше синаптический вес связи, соединяющей эти два нейрона, тем больше будет ошибка на выходе скрытого нейрона. Таким образом, получаем ошибку скрытого нейрона:

$$\delta_j = f'(S) \sum_k \delta_k w_{kj} \ . \qquad (25)$$

Метод градиентного спуска

Алгоритм обратного распространения ошибки использует метод *градиентного спуска.*

Идея этого оптимизационного алгоритма заключается в движении к локальным экстремумам функции по направлению наиболее быстрого убывания или возрастания функции. Для этого находится градиент – вектор определяющий направления возрастания функции:

$$\text{grad}\varphi = \nabla\varphi = \frac{\partial\varphi}{\partial x} i + \frac{\partial\varphi}{\partial y} j + \frac{\partial\varphi}{\partial z} k \ . \qquad (26)$$

В случае искусственных нейронных сетей градиент вычисляется как сумма градиентов, вызванных каждым элементом обучающей выборки. Такая разновидность градиентного спуска называется *пакетной*. [7,8]

В отличие от пакетного градиентного спуска, существует *стохастический* (*оперативный*) градиентный спуск, отличие которого в том, что значение градиента аппроксимируется градиентом функции стоимости, вычисленной на одном случайном элементе обучающей выборки.

Существует также третий вариант, который совмещает в себе пакетный и стохастический методы – *mini-batch*. [9] Данный метод аппроксимирует значение градиента по *n* случайно выбранным элементам обучающей выборки.

Из достоинств данного алгоритма можно вынести лёгкую реализацию, возможность использовать множество функций потерь, может быть использован для больших данных. Из недостатков, данный алгоритм используется в основном для выпуклых функций, так как для других функций

находит лишь локальные экстремумы. Также в некоторых ситуациях возможна расходимость, при выборе слишком большой скорости обучения, или скорость сходимости окажется слишком мала, притом присутствует вероятность переобучения. Для решения данных проблем существуют оптимизаторы градиентного спуска. Рассмотрим некоторые из них:

Nesterov Accelerated Gradient – данный метод оптимизации основан на идеи накопления импульса, т.е. при длительном движении в одном направлении, скорость будет сохраняться некоторое время после. Для этого необходимо хранить несколько предыдущих значений и вычислять среднее. Вычисление среднего значения занимает слишком много памяти для большого количества вхождений, поэтому используется оценка среднего. Данные значения домножаются на коэффициент сохранения γ. [12]

$$v_t = \gamma v_{t-1} + (1 - \gamma)x \, . \qquad (27)$$

Adagrad – адаптивный градиент (adaptive gradient). Это семейство алгоритмов, основная идея которого заключается в том, чтобы сохранять некоторые редко встречаемые признаки для защиты их от шума. Для этого создаётся некоторая величина например сумма квадратов обновлений или их модули для некоторого параметра искусственной нейронной сети. На основе этой величины регулируются обновления элементов – часто встречаемые обновляются реже, освобождая место для редко встречаемых, тем самым получается адаптивная скорость обучения или затухание скорости обучения. [12]

Adam – adaptive moment estimation. Данный метод оптимизации сочетает в себе накопление импульса, рассмотренное в методе нестерова, а также хранение частоты изменения градиента, схожее с adagrad. Таким образом, данный метод обладает преимуществами обоих рассмотренных методов. [12]

1.3 Обзор и сравнение библиотек.

TensorFlow

TensorFlow — открытая программная библиотека для машинного обучения, разработанная компанией Google для решения задач построения и тренировки нейронной сети с целью автоматического нахождения и классификации образов, достигая качества человеческого восприятия Применяется как для исследований, так и для разработки собственных продуктов Google. Основное API для работы с библиотекой реализовано для Python, также существуют реализации для C++, Haskell, Java и Go.

Torch.

Torch — библиотека для научных вычислений с широкой поддержкой алгоритмов машинного обучения. Разрабатывается Idiap Research Institute, New York University и NEC Laboratories America, начиная с 2000г., распространяется под лицензией BSD.

Библиотека реализована на языке Lua с использованием C и CUDA. Быстрый скриптовый язык Lua в совокупности с технологиями SSE, OpenMP, CUDA позволяют Torch показывать неплохую скорость по сравнению с другими библиотеками. На данный момент поддерживаются операционные системы Linux, FreeBSD, Mac OS X. Основные модули также работают и на Windows.

Keras

Keras — открытая нейросетевая библиотека, написанная на языке Python. Она представляет собой надстройку над фреймворками Deeplearning4j, TensorFlow и Theano.[1][2] Нацелена на оперативную работу с сетями глубинного обучения, при этом спроектирована так, чтобы быть компактной, модульной и расширяемой. Она была создана как часть исследовательских усилий проекта ONEIROS (англ. Open-ended Neuro-Electronic Intelligent Robot Operating System),[3] а ее основным автором и поддерживающим является Франсуа Шолле (фр. François Chollet), инженер Google.

Для сравнения библиотек использован набор данных CIFAR-10, содержащий в себе 10 классов изображений[13]. Всего в наборе данных содержится 60000 цветных изображений размером 32 на 32 пикселя, где для каждого класса соответственно 6000 изображений. Классы, входящие в CIFAR-10: самолёты, автомобили, грузовики, птицы, кошки, олени, собаки, лягушки, лошади и корабли (Рис. 13).

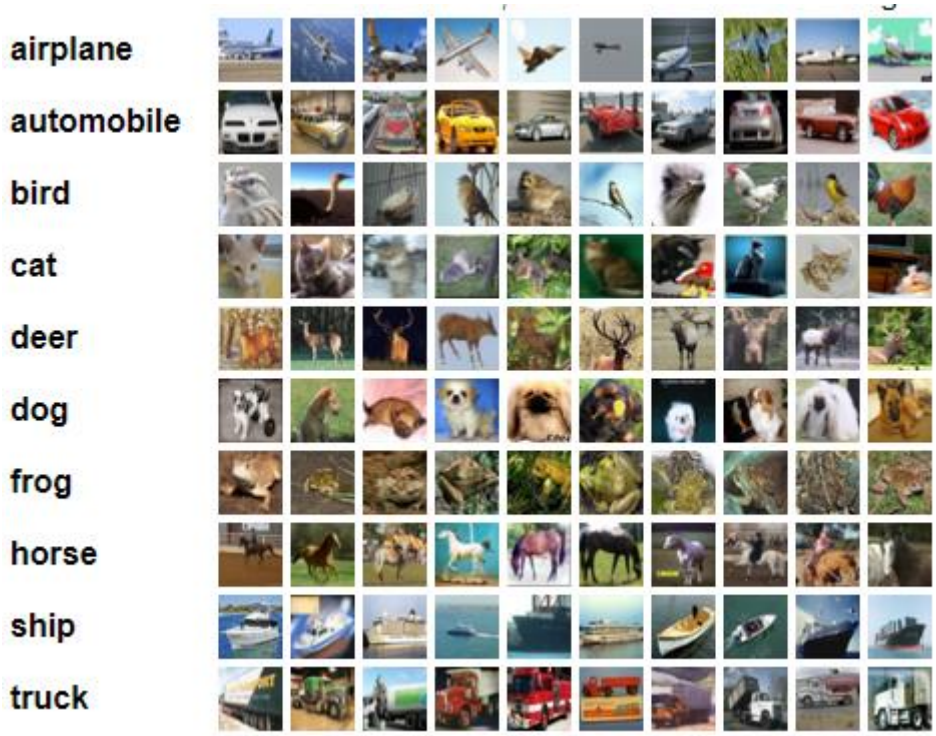

Рисунок 13 -Примеры изображений набора данных CIFAR 10

В качестве архитектуры для тестирования использована VGG 16 разработанная в Оксфорде в 2014 году (Рис. 14).

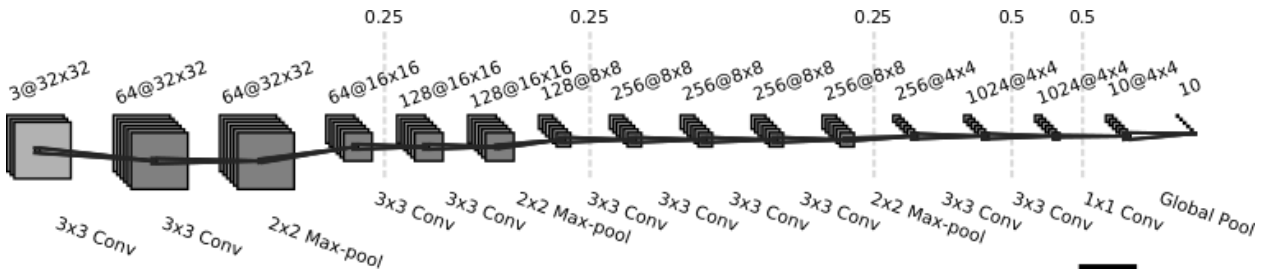

Рисунок 14 -Архитектура сети для тестирования библиотек

По результатам времени обучения, быстрее всего обучилась библиотека Pytorch – 168 секунд, незначительно от неё отстаёт библиотека TensorFlow – 173 секунды и самое долгое время показывает библиотека Keras – 252 секунды

(Рис. 15). По точности библиотеки Pytorch и TensorFlow показывают одинаковые результаты – 78%, Keras на один процент меньше – 77% (Рис. 16)[14].

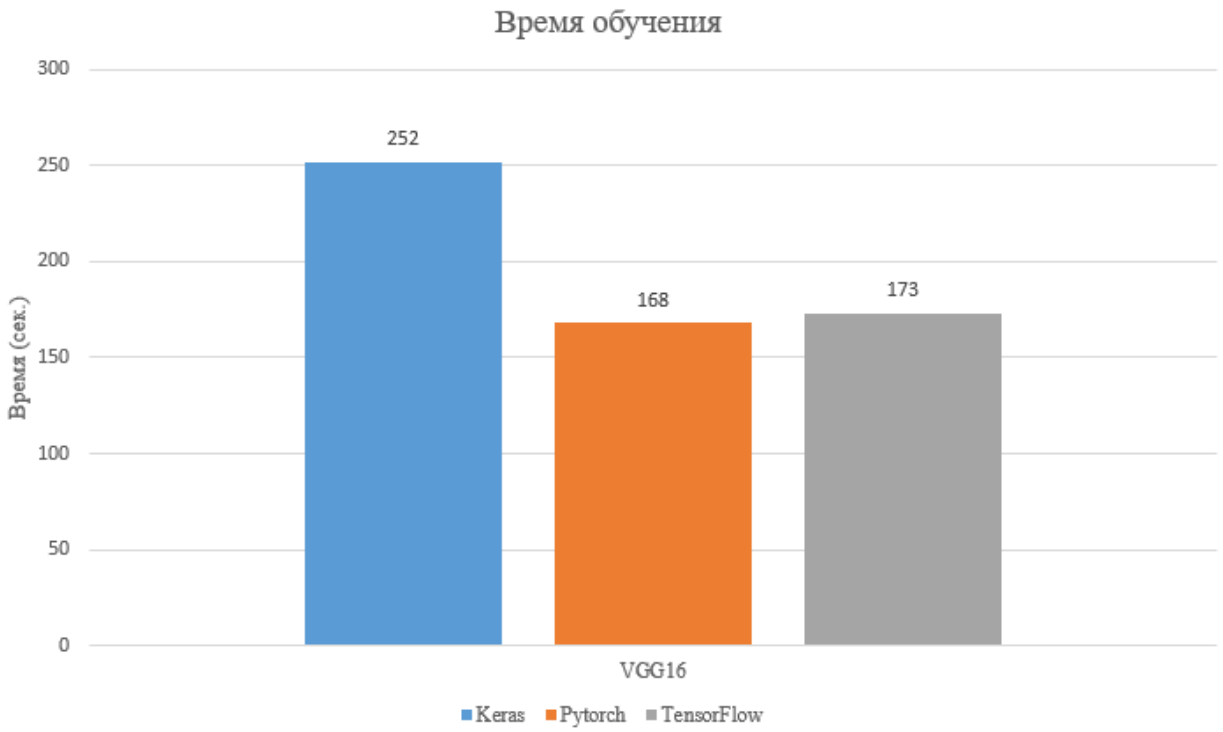

Рисунок 15 -Сравнение обучения библиотек

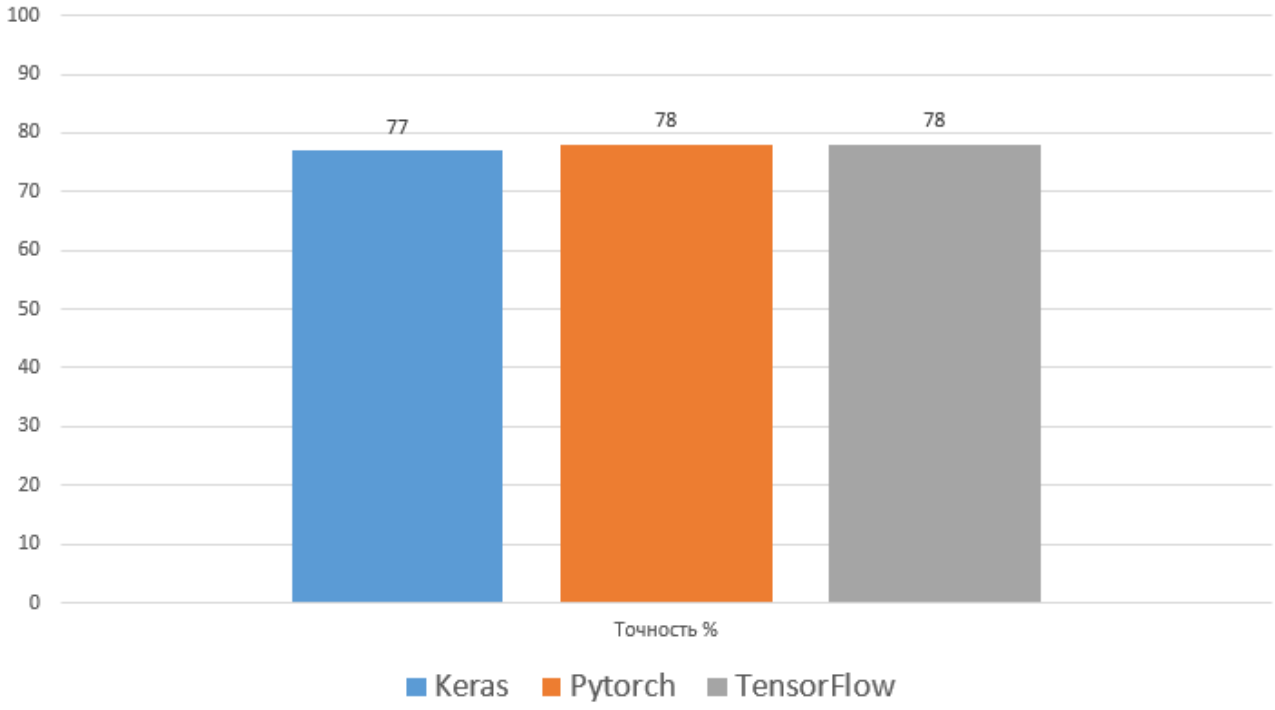

Рисунок 16 - Точность результатов обучения

Таблица 1 – Сводные характеристики библиотек.

Библиотека	Keras	PyTorch	TensorFlow
Авторы	François Chollet	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan	Google Brainteam
Распространяется под лиценззией	MIT license	BSD license	Apache 2.0
Открытый исходный код	+	+	+
Поддерживаемые ОС	Linux, macOS, Windows	Linux, macOS, Windows	Linux, macOS, Windows,Android
Написана на языке	Python	Python, C, CUDA	C++, Python, CUDA
Языковой интерфейс	Python, R	Python	Python (Keras), C/C++, Java, Go, R, Julia
Поддержка CUDA	+	+	+
Автоматическое дифференцирование	+	+	+
Есть заранее натренированные модели	+	+	+
Поддержка рекуррентных нейросетей	+	+	+
Поддержка свёрточных нейросетей	+	+	+
Многопоточные вычисления	+	+	+

2 РАЗРАБОТКА ПРОГРАММЫ

2.1 Построение архитектуры.

В качестве архитектуры использована полная свёрточная сеть (Fully convolutional network).

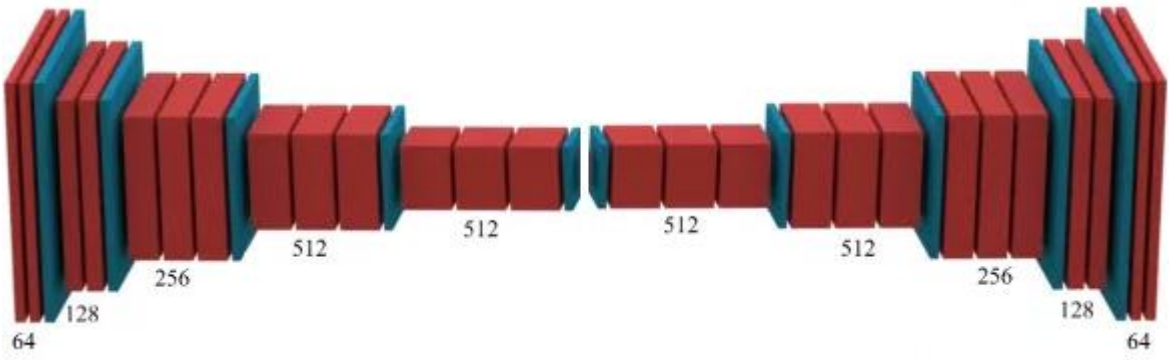

Рисунок 17 - Ахитектура полной свёрточной сети

Использованы персептроны, у которых нет лимита активации. Они просто выдают линейную форму от своих входов:

$$o(x_0, \ldots, x_n) = \sum_{0}^{n} w_i x_i. \quad (28)$$

Т.е. у такого персептрона не два, а континуально много возможных значений.

Для функции обратного распространения ошибки был выбран стохастический градиентный спуск[15].

Алгоритм работы метода - необходимо найти персептрон, который минимизирует ошибку. Пусть есть m тестовых примеров x_i^j с верными ответами t^j, j = 1..m. Ошибка — из статистики, среднеквадратичное отклонение:

$$E(w_0, \ldots, w_n) = \frac{1}{2} \sum_{j=1}^{m} (t_j - o(x_0^j, \ldots, x_n^j))^2. \quad (29)$$

Нужно минимизировать функцию E на пространстве возможных весов $\{w_i\}$.

Чтобы минимизировать нелинейную функцию от нескольких аргументов нужно двигаться в сторону, обратную градиенту. Градиент — направление, в котором достигается наибольший прирост значений:

$$\nabla E(w_0, \ldots, w_n) = \left[\frac{\partial E}{\partial w_0}, \frac{\partial E}{\partial w_1}, \ldots, \frac{\partial E}{\partial w_n}\right] \quad (30)$$

Изменения весов достигаются следующим образом:

$$w_i \leftarrow w_i - \eta \frac{\partial E}{\partial w_i}, \quad (31)$$

Все веса изменяются после каждого тестового примера.

2.2 Обучение

Для обучения нейронной сети был использован набор данных с сайта ISPRS.com для города Фáйинген Германия (Рис. 18). Пространственное разрешение снимков 9см на пиксель. Каналы снимков составляют ближний инфракрасный-красный-зелёный (IRRG). Город поделён на сектора, каждый из которых имеет свой уникальный номер (Рис. 19).

Обучающие примеры представляют собой набор снимков и маску. В зависимости от цвета, каждый элемент маски представляет собой определённый объект на снимке (рис 20). Дороги и тротуары – белый, здания – синий, деревья – зелёный, трава и кусты – бирюзовый, автотранспорт – жёлтый. Всего было использовано 12 снимков для обучения и 4 снимка для проверки. Используемые снимки имеют разрешение около 2000 на 2500 пикселей[17]. Разрешение для каждого снимка указано в таблице 2. Обучение производилось на 50 эпохах по 2000 итерации в каждой, где использовались фрагменты 256 на 256 пикселей (Рис.21). В обучении участвовали 12 снимков, а для контрольной проверки использовались снимки под номером 5,15,21,30.

Рисунок 18 - Снимок из набора данных города Фа́йинген

Таблица 2 – Разрешение обучающих снимков

TOP	Ncol	Nrow
top_mosaic_09cm_area1	1919	2569
top_mosaic_09cm_area3	2006	3007
top_mosaic_09cm_area5	1887	2557
top_mosaic_09cm_area7	1887	2557
top_mosaic_09cm_area11	1893	2566
top_mosaic_09cm_area13	2818	2558
top_mosaic_09cm_area15	1919	2565
top_mosaic_09cm_area17	2336	1281
top_mosaic_09cm_area21	1903	2546
top_mosaic_09cm_area23	1903	2546
top_mosaic_09cm_area26	2995	1783
top_mosaic_09cm_area28	1917	2567
top_mosaic_09cm_area30	1934	2563
top_mosaic_09cm_area32	1980	2555
top_mosaic_09cm_area34	1388	2555
top_mosaic_09cm_area37	1996	1995

Рисунок 19 -Сектора города Фа́йинген

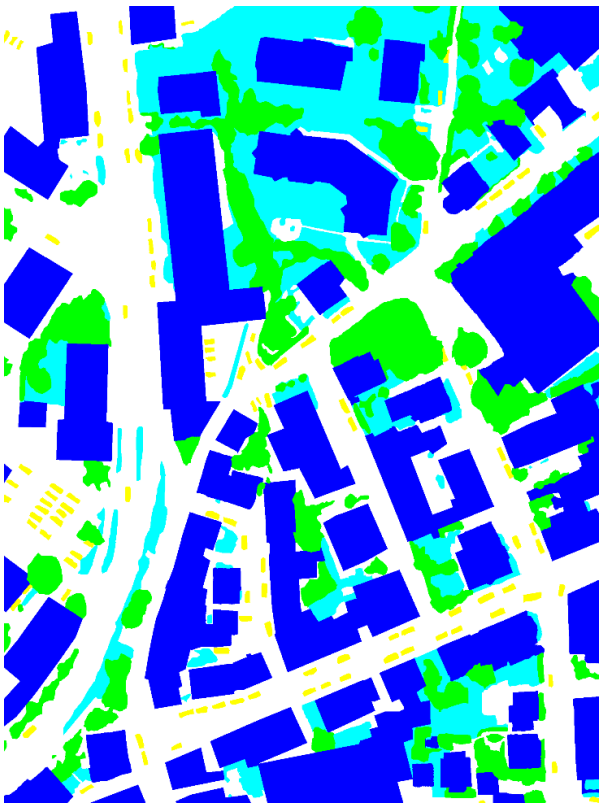

Рисунок 20 - Снимок маски из набора данных города Фа́йинген

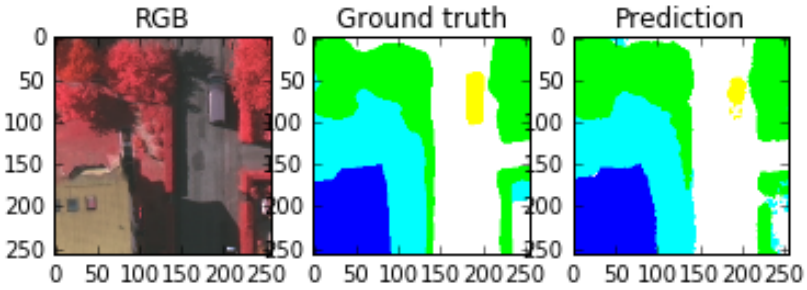

Рисунок 21 - Процесс обучения. Слева оригинальный снимок, по середине обучающий пример, справа предсказанный результат

2.3 Результаты

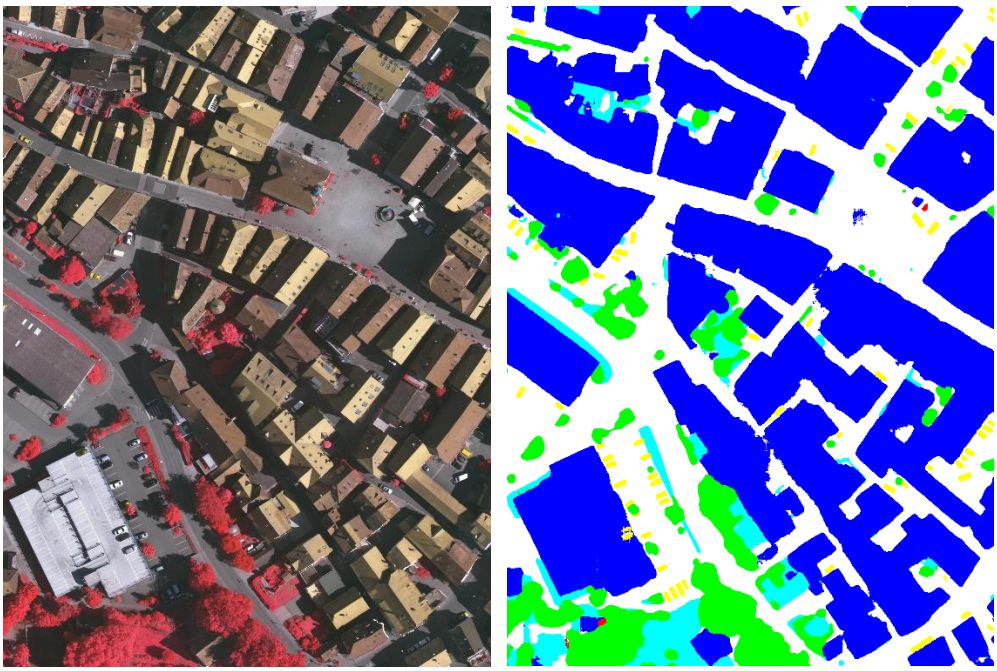

Рисунок 22 - Полученый результат, слева тестовый снимок, справа предсказанная маска.

Итоговая точность составила 89,84%, для дорог: 90,89%, зданий 94,65%, деревья: 89,89%, кусты и трава: 80,62%, автомобили: 89,32%.

Задача получения объектов на аэрокосмических снимках уже давно используется, поэтому существуют алгоритмы, её реализующие. Из таблицы 3 видно, что разработанная система незначительно уступает существующим аналогам по таким показателям как дороги, здания, трава. Однако, по таким показателям как деревья и машины она выигрывает. Существующие аналоги

являются закрытыми коммерческими системами, поэтому проанализировать алгоритмы, используемые в них, не представляется возможным. так же существующие системы являются достаточно дорогостоящими, что так же усложняет их использование среди широкого круга пользователей.

Таблица 3 - Сравнение с аналогами

Название	Дороги	Здания	Трава и кусты	Деревья	Машины	Итого
CVEO3	91	93	81.3	88.3	83	88.6
CASRS2	90.6	93.2	82	88.8	76.6	88.7
WUH_W5	92.7	95.3	83.6	89.2	85.8	90.4
BKHN10	92.9	96	84.6	89.8	88.8	91
BKHN11	92.9	96	84.6	89.9	88.6	91
LLYT1	91.5	93.7	81.5	88.4	78.9	89
Разработанная программа	90.8	94.7	80.6	89.9	89.3	89.8

3 ФИНАНСОВЫЙ МЕНЕДЖМЕНТ, РЕСУРС ЭФФЕКТИВНОСТЬ И РЕСУРСОСБЕРЕЖЕНИЕ

3.1 Экономическая концепция и реализация научного проекта

Разработанный алгоритм осуществляет сегментацию и распознавание объектов на снимках дистанционного зондирования Земли. Системы такого рода используются в картографии. Все большее применение распознавание объектов находит и в частном бизнесе: в геологии, в ГИС системах.

3.2 Календарный план выполнения работы

В рамках планирования научного проекта был построен календарный план проекта в виде линейного графика в таблице (таблица 3.1) и его иллюстрация работы в виде диаграммы Ганта (таблица 3.2). Диаграммы Ганта строилась с разбивкой по месяцам и декадам (10 дневки) который обхватывает весь период времени выполнения научного проекта.

Таблица 3.1 - Календарный план проекта

Дата контроля	Название	Длительность, дни	Дата начала работ	Дата окончания работ	Состав участников
1	Постановка целей и задач	2	09.01.	11.01	НР
2	Анализ предметной области	8	11.01	22.01	И
3	Исследование и выбор методов сегментации	6	22.01	29.01	И, НР
4	Исследование и выбор методов распознавания	6	29.01	5.02	И, НР
5	Выбор программного обеспечения, библиотеки, языка программирования	6	5.02	12.02	И, НР
6	Сбор обучающей и тестовой выборки	6	12.02	19.02	И
7	Проектирование архитектуры приложения	21	19.02	16.03	И
8	Разработка приложения для семантической сегментации снимков	32	16.03	30.04	И
9	Тестирование конечного программного обеспечения	11	30.04	14.05	И, НР

10	Оформление пояснительной записки	6	14.05	21.05	И
11	Оформление графического материала	10	21.05	1.06	И
12	Подведение итогов	2	1.06	3.06	И, НР
Итого: Инженер Научный руководитель		116 114 33	9.01	3.06	

Перечисленные работы в таблице 3.1, 3.2 были выполнены специалистами:

1) И – инженер – исполнитель ВКР – Аркалыков Ерболат Усенович;

2) НР – научный руководитель – доцент отдела информационных технологий Друки Алексей Алексеевич.

Таблица 3.2 – Диаграмма Ганта

Код	Вид работ	Ис-пол-ни-тели	Т, кол, дн.	Продолжительность выполнения работ, месяц															
				1			2			3			4			5			6
				1	2	3	1	2	3	1	2	3	1	2	3	1	2	3	1
1	Постановка целей и задач	НР	2																
2	Анализ предметной области	И	8																
3	Исследование и выбор методов сегментации	И, НР	6																
4	Исследование и выбор методов распознавания	И, НР	6																
5	Выбор программного обеспечения, библиотеки, языка программирования	И, НР	6																
6	Сбор обучающей и тестовой выборки	И	6																
7	Проектирование архитектуры приложения	И	21																

8	Разработка приложения для семантической сегментации снимков	И	32																
9	Тестирование конечного программного обеспечения	И, НР	11																
10	Оформление пояснительной записки	И	6																
11	Оформление графического материала	И	10																
12	Подведение итогов	И, НР	2																

3.3 Бюджет научного исследования

Общий бюджет научно исследовательской работы состоит из 4 видов планируемых расходов: материальные расходы, вознаграждения, отчисления на социальные нужды и накладные расходы.

3.3.1 Материальные затраты

Материальные затраты – элемент себестоимости продукта. В ходе выполнения научно-исследовательской работы для разработки алгоритма не были приобретены услуги. Использованные программы и библиотеки бесплатны для учащихся в заведениях высшего образования и находятся в открытом доступе. Результаты по сделанным материальным затратам представлены на таблице 3.3.

Таблица 3.3 – Материалы

Наименование	Марка, размер	Количество	Цена за единицу, руб	Сумма, руб
Ручка	Maxriter	2	75	150
Карандаш	Maxriter	1	30	30
Бумага А4	SvetoCopy	100	3	300

Всего за материалы	480
Транспортно-заготовительные расходы (3-5%)	24
Итого	504

К материальным затратам так же относится и затраты на электроэнергию. Затраты на электроэнергию рассчитываются по формуле:

$$С_{эл.об.} = Р_{об} \cdot t_{об} \cdot Ц_э, \quad (3.1)$$

где $Р_{об}$ – мощность, потребляемая оборудованием, кВт; $t_{об}$–время работы оборудования, час; $Ц_э$, – тариф на 1 кВт□час. Для ТПУ $Ц_э$ = 5.8 руб./кВт · час.

Время работы оборудования вычисляется на основе итоговых данных таблицы 2 для инженера из расчета, что продолжительность рабочего дня равна 8 часов.

$$t_{об} = Т_{РД} \cdot K_t, \quad (3.2)$$

где K_t – коэффициент использования оборудования по времени, K_t = 0,9.

Мощность, потребляемая оборудованием, определяется по формуле:

$$Р_{об} = Р_{ном} \cdot K_C, \quad (3.3)$$

где K_C – коэффициент загрузки. Для технологического оборудования малой мощности K_C = 1 [36].

Итоги расчет затрат на электроэнергию приведены на таблице 3.4.

Таблица 3.4 – Затраты на электроэнергию технологическую

Наименование оборудования	Время работы оборудования $t_{об}$, час	Потребляемая мощность $Р_{об}$, кВт	Затраты $С_{эл.об.}$, руб.
Персональный компьютер инженера	228	0,09	119,016

Материальные затраты с учетом затрат на электроэнергию составляют 588 рублей 56 копеек.

3.3.2 Вознаграждения

В ТПУ оклады распределены в соответствии с занимаемыми должностями, например, ассистент, ст. преподаватель, доцент, профессор. Базовый оклад $З_б$ определяется исходя из размеров окладов, определенных штатным расписанием предприятии. В соответствие с утвержденным Положением об оплате труда в Томском политехническом университете ставки почасовой оплаты для научного руководителя с ученой степенью доцента ровна $З_{дн.науч.р}$ = 300 рублей, для инженера незащищенных специалистов $З_{дн.инж}$ = 175 рублей.

Основная заработная плата ($З_{осн}$) инженера и научного руководителя рассчитывается по следующей формуле:

$$З_{осн} = З_{поч} * Т_р \quad (3.4)$$

где $З_{осн}$ – основная заработная плата работника;

$Т_р$ – продолжительность работ, выполняемых научно-техническим работником, раб. час;

$З_{поч}$ – почасовая оплата работника, руб.

При расчете учитывалось, что работа велась в период 09.01- 03.06, всего 116 дней при средней загрузке 2 часа в день. Затраты времени на выполнение работы по каждому исполнителю брались из таблицы 2:

- научный руководитель – 33 дней, всего $Т_{р.науч.р}$ = 66 часов.
- инженер – 114 дней, всего $Т_{р.инж}$ = 228 часов.

$$З_{осн.науч.р} = 300 * 66 = 19800 \text{ рублей},$$

$$З_{осн.инж} = 175 * 228 = 39900 \text{ рублей}$$

Общая сумма заработной платы инженера и научного руководителя составило 59 700 рублей.

3.3.3 Отчисления на социальные нужды

Ставка взноса в социальные фонды при выполнении научных работ установлен в размере 27,1% от заработной платы. Размер взноса рассчитываются по формуле:

$$С_{соц} = С_{ЗП} \cdot 0{,}271\%, \quad (3.5)$$

где $С_{ЗП}$ – размер заработной платы.

$$С_{соц} = 59\ 700 \cdot 0{,}271 = 16\ 178{,}7 \text{ руб.}$$

3.3.4 Накладные расходы

Накладные расходы составляют 40% от всей суммы затрат. Расчет накладных расходов ведется по следующей формуле:

$$С_{нак} = S_{об}.сум.зат * 0{,}4\% \ , \qquad (3.6)$$

где $S_{об}$.сум.зат – общая сумма всех затрат составляет 75 878,7 рублей.

$$С_{нак} = 75\ 878{,}7 * 0{,}4\% = 30\ 351{,}48$$

3.3.5 Расчет общей себестоимости разработки

Проведя расчет сметы затрат на разработку, можно определить общую стоимость разработки проекта «Разработка и программная реализация алгоритма сегментации и распознавания автомобильных номеров» (таблица 3.6).

Таблица 3.6 – Смета затрат на разработку проекта

Статья затрат	**Условное обозначение**	**Сумма, руб.**
1. Материальные затраты	$С_{МАТ}$	504
2. Вознаграждения	$С_{ЗП}$	59 700
3. Отчисления в социальные нужды	$С_{СОЦ}$	16 178,7
4. Расходы на электроэнергию	Э	119,016
5. Накладные расходы	СПРОЧ	30 351,48
Итого:		**106 853,196**

Таким образом, расходы на разработку составили $С$ = 106 853,196 рублей.

3.4 Оценка экономической эффективности проекта

Экономический эффект характеризуется в частности выраженной в стоимостных показателях экономией живого общественного труда, а также в возможности применения полученных знаний для создания новых разработок.

Экономическая эффективность проекта обусловлена возрастающей необходимостью в системах автоматизации процессов распознавания номерных

пластин. Конечная система в виде авто-подключаемого модуля или библиотеки с возможностью сегментирования и распознавания номерных пластин является востребованной и актуальной на сегодняшний день. Использованной подобной системы в коммерческой и социальной сфере позволяет решать такие задачи, как оперативное распознавание номерной пластины, а также систематизация и проверка полученных результатов.

Подводя итог вышесказанному, экономический эффект от реализации проекта может быть выражен в снижении затрат на покупку подобных систем от сторонних разработчиков, а также повышение эффективности распознавания пластин за счет экономии времени при замене ручной обработки на автоматизированную.

3.4.1 Оценка научно-технического уровня НИР

Научно-технический уровень характеризует, в какой мере выполнены работы и обеспечивается научно-технический прогресс в данной области. Для оценки научной ценности, технической значимости и эффективности, планируемых и выполняемых НИР, используется метод бальных оценок.

Сущность метода заключается в том, что на основе оценок признаков работы определяется коэффициент ее научно-технического уровня по формуле:

$$K_{\text{НТУ}} = \sum_{i=1}^{3} R_i \cdot n_i, \qquad (3.7)$$

где $K_{\text{НТУ}}$ – коэффициент научно-технического уровня;

R_i – весовой коэффициент i-го признака научно-технического эффекта;

n_i – количественная оценка i-го признака научно-технического эффекта, в баллах.

Таблица 3.7 – Весовые коэффициенты признаков НТУ

Признак НТУ	**Примерное значение весового коэф-та *ni***	
1. Уровень новизны	Систематизируются и обобщаются сведения, определяются пути дальнейших исследований	0,4
2. Теоретический уровень	Разработка алгоритма	0,1

3. Возможность реализации	Время реализации в течение первых лет	0,5

Таблица 3.8 – Баллы для оценки уровня новизны

Уровень новизны	**Характеристика уровня новизны**	**Баллы**
Принципиально новая	Новое направление в науке и технике, новые факты и закономерности, новая теория, вещество, способ	8 – 10
Новая	По-новому объясняются те же факты, закономерности, новые понятия дополняют ранее полученные результаты	5 – 7
Относительно новая	Систематизируются, обобщаются имеющиеся сведения, новые связи между известными факторами	2 – 4
Не обладает новизной	Результат, который ранее был известен	0

Таблица 3.9 – Баллы значимости теоретических уровней

Теоретический уровень полученных результатов	**Баллы**
1. Установка закона, разработка новой теории	10
2. Глубокая разработка проблемы, многоспектральный анализ, взаимодействия между факторами с наличием объяснений	8
3. Разработка способа (алгоритм, программа и т. д.)	6
4. Элементарный анализ связей между фактами (наличие гипотезы, объяснения версии, практических рекомендаций)	2
5. Описание отдельных элементарных факторов, изложение наблюдений, опыта, результатов измерений	0,5

Таблица 3.10 – Возможность реализации научных, теоретических результатов по времени и масштабам

Время реализации	**Баллы**
В течение первых лет	10
От 5 до 10 лет	4
Свыше 10 лет	2

Результаты оценок признаков научно-технического уровня приведены в таблице 3.11.

Таблица 3.11 – Количественная оценка признаков НИОКР

Признак научно-технического эффекта НИР	**Характеристика признака НИОКР**	***Ri***
Уровень новизны	Систематизируются, обобщаются имеющиеся сведения, новые связи между известными факторами	00,4
Теоретический уровень	Разработка способа (алгоритм, программа мероприятий, устройство, вещество и т.п.)	00,1
Возможность реализации	Время реализации 5-10 лет	00,5

Обоснование оценки признаков НИОКР приводится в таблице 3.12.

Таблица 3.12 – Оценки научно-технического уровня НИР

Фактор НТУ	**Значимость**	**Уровень фактора**	**Выбранный балл**	**Обоснование выбранного балла**
Уровень новизны	0,4	Относительно новая	4	Облегчит обработку машиночитаемых бланков
Теоретический уровень	0,1	Разработка способа	6	Разработана и реализован алгоритм, решающий поставленные в рамках задания задачи
Возможность реализации	0,5	В течение первых лет	7	Полученный продукт находится на стадии тестирования и внедрения

Исходя из оценки признаков НИОКР, показатель научно-технического уровня для данного проекта составил:

$$К_{нту} = 0{,}4 * 4 + 0{,}1 * 6 + 0{,}5 * 7 = 5{,}7$$

Таблица 3.13 – Оценка уровня научно-технического эффекта

Уровень НТЭ	**Показатель НТЭ**
Низкий	1-4
Средний	4-7
Высокий	8-10

Таким образом, исходя из данных в таблице 14, проект «Алгоритмы нейросетевой сегментации снимков дистанционного зондирования поверхности Земли» имеет средний уровень научно-технического эффекта.

4 СОЦИАЛЬНАЯ ОТВЕТСТВЕННОСТЬ

Программное обеспечение осуществляет сегментацию и распознавание объектов на снимках дистанционного зондирования Земли. Может быть востребовано в картографии:

Здания;

Дороги;

Автомобили;

Деревья и растительность.

Разработка системы велась исключительно при помощи компьютера. Однако использование средств вычислительной техники накладывает целый ряд вредных факторов на человека, что впоследствии снижает производительность его труда и может привести к существенным проблемам со здоровьем сотрудника.

Данный раздел посвящен анализу вредных и опасных факторов производственной среды как для разработчиков, так и для пользователей разработанной системы.

Все производственные факторы классифицируются по группам элементов: физические, химические, биологические и психофизические. Для данной работы целесообразно рассмотреть физические и психофизические вредные и опасные факторы производства, характерные для рабочей зоны программиста, разработчика приложения, пользователя. Выявленные факторы представлены в таблице 4.1.

Таблица 4.1 – Вредные и опасные производственные факторы при выполнении работ за ПЭВМ

Источник фактора, наименование видов работ	Факторы (по ГОСТ 12.0.003-74)		Нормативные документы
	Вредные	Опасные	
1) Работа за ПК	1) Недостаточная освещенность рабочей зоны; 2) Умственное перенапряжение; 3) Монотонный режим работы.	1) Опасность поражения электрическим током; 2) Опасность возникновения пожара.	1) СН 2.2.4/2.1.8.56296; 2) СанПиН 2.2.4.548-96; 3) СанПиН 2.2.2/2.4.134003; 4) СП 52.13330.2011; 5) ГОСТ Р 12.1.019-2009 ССБТ; 6) СНиП 21-0197.

4.1 Профессиональная и социальная безопасность

Работа связана с анализом и теоретической разработкой алгоритмов сегментации и распознавания объектов на снимках дистанционного зондирования Земли. В ходе выполнения исследования и при дальнейшей эксплуатации разработанного алгоритма на рабочем месте могут возникнуть следующие вредные и опасные факторы:

1) несоответствия нормам условий освещения рабочего места;
2) несоответствия нормам условий температуры рабочего места;
3) повышенное напряжение электрической цепи;
4) возможности возникновения пожара
5) монотонность работы при разработке алгоритма.

Также данный алгоритм может быть использован в корыстных целях, как часть программного обеспечения. Для предотвращения или уменьшения количества подобных происшествий необходимо обеспечение гарантий защиты конфиденциальных данных владельцев автомобилей с помощью комплекса технических и юридических мер.

4.1.1 Искусственное освещение

Рабочее освещение предназначено для создания нормальных условий видения на рабочих местах при выполнении трудовых процессов. Согласно санитарно-гигиеническим требованиям рабочее место разработчика должно освещаться естественным и искусственным освещением.

Для освещения помещений искусственным светом следует использовать, как правило, наиболее экономичные разрядные лампы. Использование ламп накаливания для общего освещения допускается только в случае невозможности или технико-экономической нецелесообразности использования разрядных ламп [23].

Рабочая зона должно освещаться так, чтобы в процессе работы:

− не ощущаться раздражение глаз;

− иметь хорошее видение;

− исключалось прямое попадание лучей источника света в глаза.

Гигиенические требования к производственному освещению, основанные на психофизических особенностях восприятия света и его влияния на организм человека, могут быть сведены к следующим:

−спектральный состав света, создаваемого искусственными источниками, должен приближаться к солнечному;

−уровень освещенности должен быть достаточным и соответствовать гигиеническим нормам, учитывающим условия зрительной работы;

−должна быть обеспечена равномерность и устойчивость уровня освещенности в помещении во избежание частой переадаптации и утомления

зрения. В то же время, по имеющимся данным, при длительной работе в равномерно освещенном пространстве может нарушиться восприятие формы объектов, реализующееся, в конечном счете, в зрительных галлюцинациях;

−освещение не должно создавать блескости как самих источников света, так и других предметов в пределах рабочей зоны.

По нормам освещенности СанНиП 23-05-95 (СанНиП 23-05-2010) и отраслевым нормам, работа разработчика относится к четвертому разряду зрительной работы средней точности. В таблице 4.1 приведены нормы освещенности при искусственном освещении.

Таблица 4.2 - Нормы освещенности при искусственном освещении

Характеристика зрительной работы	Наименьший или эквивалентный размер объекта различения, мм.	Разряд зрительной работы	Освещённость, лк
			Искусственное освещение
Малой точности	от 0.3 до 0.5	III	200

4.1.2 Микроклимат

Существует нормативный документ «Гигиенические требования к микроклимату производственных помещений. СанПиН 2.2. 4.548-96». Все требования, перечисленные в данном документе должны соблюдаться на рабочем месте.

Согласно этому документу, требования по температуре для работников умственного труда при восьмичасовом рабочем дне:

1. 23-25°С – наиболее подходящая температура летом;
2. 22-24°С – оптимум в холодное время года;
3. 1-2°С – допустимое отклонение от нормы температуры;
4. 3-4 °С – допустимое колебание температуры в продолжении рабочего дня.

К мероприятиям по оздоровлению воздушной среды в производственном помещении относятся: правильная организация вентиляции и кондиционирования воздуха, отопление помещений. Вентиляция может осуществляться естественным и механическим путём. В помещении должны подаваться следующие объёмы наружного воздуха: при объёме помещения до 20 м3 на человека – не менее 30 м3 в час на человека; при объёме помещения более 40 м3 на человека и отсутствии выделения вредных веществ допускается естественная вентиляция [24].

В аудитории 407 ИК присутствует принудительная вентиляция. Её наличие обусловлено требованиями по объёму воздуха в помещении приходящегося на одного работника – более 40 м3. В зимнее время в помещении предусмотрена система водяного отопления со встроенными нагревательными элементами и терморегуляторами.

4.1.3 Монотонность работы

Монотонность – однообразное повторение рабочих операций. Опасность монотонности заключается в снижении внимания к процессу производства, быстрой утомляемости и снижении интереса к трудовому процессу, что влияет на безопасность труда в целом. Монотонность сопровождается скукой, апатией к выполнению трудовой деятельности [26].

Данную работу относим к монотонной так, как и при исследовании, разработки и при эксплуатации данного алгоритма как часть программного обеспечения является однообразным трудом в условиях однообразной рабочей обстановки. Например, при разработке программист, сидя за компьютером 8 часов пишет код программы.

Основные отрицательные моменты нерациональной организации заключаются в следующем:

- нерациональное чередование периодов работы и отдыха
- отсутствие кратности операций

- развитие заболеваний таких как варикоз, Сколиоз, Тромбоз, Синдром хронической усталости [26].

Решающее значение имеет неукоснительное соблюдение техники безопасности труда, контроля за трудовым процессом и чередование периодов труда и отдыха.

4.1.4 Поражение электрическим током

Поражение электрическим током является опасным производственным фактором и, поскольку программист имеет дело с электрооборудованием, то вопросам электробезопасности на его рабочем месте должно уделяться особое внимание. Нормы электробезопасности на рабочем месте регламентируются СанПиН 2.2.2/2.4.1340-03, вопросы требований к защите от поражения электрическим током освещены в ГОСТ Р 12.1.019-2009 ССБТ.

Электробезопасность – система организационных и технических мероприятий и средств, обеспечивающих защиту людей от вредного и опасного воздействия электрического тока, электрической дуги, электромагнитного поля и статического электричества.

Опасность поражения электрическим током усугубляется тем, что человек не в состоянии без специальных приборов обнаружить напряжение дистанционно.

Помещение, где расположено рабочее место оператора ПЭВМ, относится к помещениям без повышенной опасности ввиду отсутствия следующих факторов: сырость, токопроводящая пыль, токопроводящие полы, высокая температура, возможность одновременного прикосновения человека к имеющим соединение с землей металлоконструкциям зданий, технологическим аппаратам, механизмам и металлическим корпусам электрооборудования.

Основным организационным мероприятием по обеспечению безопасности является инструктаж и обучение безопасным методам труда, а

также проверка знаний правил безопасности и инструкций в соответствии с занимаемой должностью применительно к выполняемой работе.

К мероприятиям по предотвращению возможности поражения электрическим током относятся:

– С целью защиты от поражения электрическим током, возникающим между корпусом приборов и инструментом при пробое сетевого напряжения на корпус, корпуса приборов и инструментов должны быть заземлены;

– При включенном сетевом напряжении работы на задней панели корпуса приборов должны быть запрещены;

– Все работы по устранению неисправностей должен производить квалифицированный персонал; – Необходимо постоянно следить за исправностью электропроводки.

4.1.5 Возникновение пожара

Причиной загорания может быть:

1) короткие замыкания в блоке питания или высоковольтном блоке дисплейной развертки;

2) несоблюдение правил пожарной безопасности;

3) наличие горючих компонентов: документы, двери, столы, изоляция кабелей и т.п.

Возникновение пожара возможно предотвратить путем осуществления соответствующих инженерно-технических мероприятий при проектировании и эксплуатации технологического оборудования, энергетических и санитарно-технических установок, а также соблюдением установленных правил и требований пожарной безопасности.

Согласно нормам технологического проектирования, в зависимости от характеристики используемых в производстве веществ и их количества, по пожарной и взрывной опасности помещения подразделяются на категории А, Б, В, Г, Д.

В случае возникновения пожара на территории предприятия действия всех работников должны быть направлены на немедленное сообщение о нем в пожарную охрану, обеспечение безопасности людей и их эвакуации, а также тушение возникшего пожара. Для оповещения людей о пожаре должны использоваться тревожные или звуковые сигналы [29].

Для предупреждения возникновения пожара необходимо соблюдать следующие правила пожарной безопасности:

1) исключение образования горючей среды (герметизация оборудования, контроль воздушной среды, рабочая и аварийная вентиляция);

2) применение при строительстве и отделке зданий несгораемых или трудно сгораемых материалов.

Необходимо проводить в аудитории следующие пожарно-профилактические мероприятия:

1) организационные мероприятия, касающиеся технического процесса с учетом пожарной безопасности объекта;

2) эксплуатационные мероприятия, рассматривающие эксплуатацию имеющегося оборудования;

3) технические и конструктивные, связанные с правильным размещением и монтажом электрооборудования и отопительных приборов.

Организационные мероприятия:

1) противопожарный инструктаж обслуживающего персонала;

2) обучение персонала правилам техники безопасности;

3) издание инструкций, плакатов, планов эвакуации.

Эксплуатационные мероприятия:

1) соблюдение эксплуатационных норм оборудования;

2) обеспечение свободного подхода к оборудованию;

3) содержание в исправности изоляции токоведущих проводников.

Технические мероприятия:

1) соблюдение противопожарных мероприятий при устройстве электропроводок, оборудования, систем отопления, вентиляции и освещения.

2) профилактический осмотр, ремонт и испытание оборудования.

Кроме устранения самого очага пожара, нужно своевременно организовать эвакуацию людей. На рисунке 23 показан план эвакуации из производственного здания, где проводилась разработка программы.

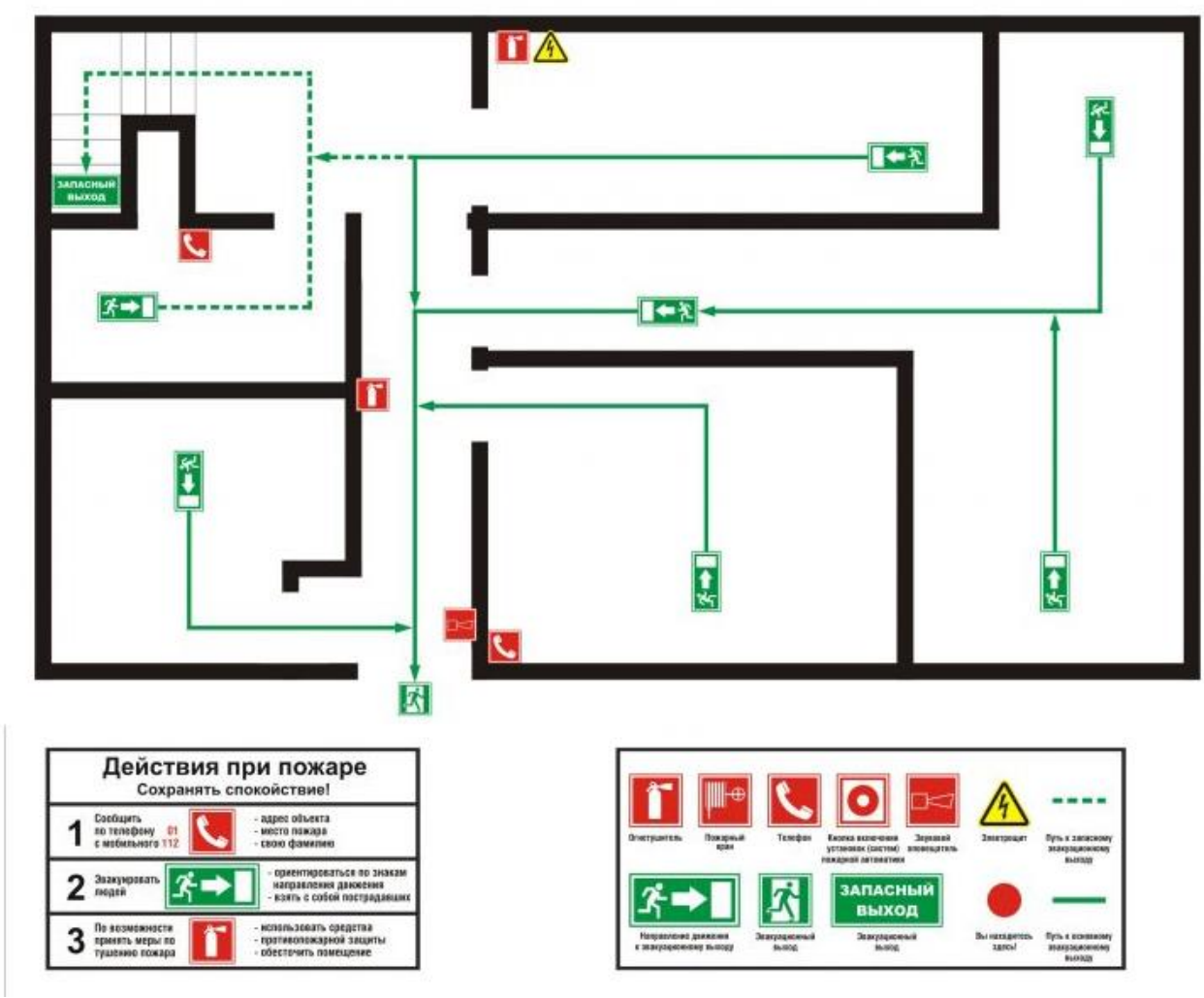

Рисунок 23 - План эвакуации

4.2 Экологическая безопасность

Для уменьшения вредного влияния на литосферу необходимо производить сортировку отходов и обращаться в службы по утилизации для дальнейшей переработки или захоронения.

В основном, организации, занимающиеся приёмом и утилизацией ртуть содержащих отходов, принимают люминесцентные лампы в массовых

количествах. Лампа состоит из электронного блока — выгодный компонент для реставрации и утилизации; колба и цоколь также ценное сырье. По стране утилизацией «ртутных» ламп занимаются более 50 фирм, но единственное их условие — деньги, которые вы должны заплатить за вывоз.

Такие лампы нельзя выкидывать в мусоропровод или уличные контейнеры, а нужно отнести в свой районный ДЕЗ (Дирекция единичного заказчика) или РЭУ (Ремонтно-эксплуатационное управление), где есть специальные контейнеры. Там они принимаются бесплатно, основанием должна служить утилизация в соответствии с Управлением Федеральной службы по надзору в сфере защиты прав потребителей и благополучия человека по Томской области. Пункты приёма отработавших свой срок люминесцентных ламп по городам можно найти в интернете.

Переработка макулатуры представляет собой многоэтапный процесс, цель которого заключается в восстановлении бумажного волокна и, зачастую, других компонентов бумаги (таких как минеральные наполнители) и использование их в качестве сырья для производства новой бумаги.

Организации, занимающиеся покупкой сломанных компьютеров на запчасти, готовы платить за запчасти деньги, которые они сэкономят на покупке новых деталей, необходимых для ремонта. Такие организации принимают даже битую и залитую чем-то технику. Компьютерная техника (или ее компоненты) может также заинтересовать тех, кто скупает старые платы и радиодетали для получения из них после переработки драгоценных и редких металлов. Многие сетевые гипермаркеты электронной техники периодически устраивают программу утилизации. Условия такие: за старую бытовую технику вам предложат неплохую скидку на последующую покупку в этом магазине. Также можно самостоятельно отвезти сломанный компьютер в пункт приема металлолома не составит труда. Такие точки приема есть в каждом городе.

Также следует отменить, что разрабатываемая в рамках ВКР система позволяет создавать сложные по структуре документы технических задании для компаний разрабатывающие информационные системы, вести

документооборот и переписку между участниками непосредственно в самой системе в электронном формате, что экономит бумагу.

4.3 Безопасность в чрезвычайных ситуациях

4.3.1 Пожары и взрывы

Объект исследования является программный продукт, для использования которого необходим ПК.

На рабочем месте программиста и эксплуататора возможно возникновение следующих чрезвычайных ситуаций техногенного характера:

- Пожары и взрывы;

- Социальные чрезвычайные ситуации.

Для обеспечения безопасности в чрезвычайных ситуациях каждый сотрудник организации должен:

- должен быть ознакомлен с инструкцией по пожарной безопасности,

- пройти инструктаж по технике безопасности

- строго соблюдать его

Работнику запрещается:

- использовать электроприборы в условиях, не соответствующих требованиям инструкций изготовителей

- использовать электропровода и кабели с поврежденной или потерявшей защитные свойства изоляцией.

- электроустановки и бытовые электроприборы по окончании рабочего времени должны быть обесточены

- недопустимо хранение легковоспламеняющихся, горючих и взрывчатых веществ, использование открытого огня в помещениях офиса.

Перед уходом из служебного помещения работник обязан провести его осмотр, закрыть окна, и убедиться в том, что в помещении отсутствуют

источники возможного возгорания, все электроприборы отключены и выключено освещение [24].

Работник при обнаружении пожара или признаков горения (задымление, запах гари, повышение температуры и т.п.) должен:

- Немедленно прекратить работу и вызвать пожарную охрану по телефону «01», сообщив при этом адрес, место возникновения пожара и свою фамилию;

- Принять по возможности меры по эвакуации людей и материальных ценностей;

- Отключить от сети закрепленное за ним электрооборудование;

- Приступить к тушению пожара имеющимися средствами пожаротушения;

- Сообщить непосредственному или вышестоящему начальнику и оповестить окружающих сотрудников;

- При общем сигнале опасности покинуть здание согласно «Плану эвакуации людей при пожаре и других ЧС».

4.3.2 Социальные чрезвычайные ситуации

Чрезвычайные ситуации бывают техногенного, природного, биологического, социального или экологического характера.

При работе в кабинете могут возникнуть следующие классификации чрезвычайных ситуаций:

– преднамеренные/непреднамеренные;

– техногенные: взрывы, пожары, обрушение помещений, аварии на системах жизнеобеспечения/природные – связанные с проявлением стихийных сил природы.

– экологические – это аномальные изменения состояния природной среды, такие как загрязнения биосферы, разрушение озонового слоя, кислотные дожди/ антропогенные – являются следствием ошибочных действий людей.

– биологические – различные эпидемии, эпизоотии, эпифитотии;

– комбинированные.

При разработке системы наиболее вероятной чрезвычайной ситуацией является пожар, так как в современных ЭВМ очень высокая плотность размещения элементов электронных схем, в непосредственной близости друг от друга располагаются соединительные провода и кабели, при протекании по ним электрического тока выделяется значительное количество теплоты, при этом возможно оплавление изоляции и возникновение возгорания. Возникновение других видов ЧС маловероятно [23].

Обеспечение пожарной безопасности учреждений прежде всего достигается установлением жесткого противопожарного режима и обучением обслуживающего персонала и учащихся мерам пожарной безопасности и действиям во время пожара.

Территория образовательного учреждения, а также участки, прилегающие к нему, должны своевременно очищаться от горючих отходов, мусора, которые следует собирать на специально выделенных площадках в контейнеры или ящики, а затем вывозить на свалку.

Важно контролировать состояние дорог, проездов, подъездов и проходов к зданиям, следить за тем, чтобы они ничем не загромождались, а в зимнее время регулярно очищались от снега и льда.

В зданиях, относящихся к объектам с массовым пребыванием людей, особое внимание должно уделяться содержанию путей эвакуации. Каждое здание должно иметь не менее двух эвакуационных выходов: если один из них отрезан огнем, для спасения используется другой. Запасные выходы должны быть свободны и иметь надпись: «Запасный выход». Все двери эвакуационных выходов свободно открываются в сторону выхода из помещений.

На случай отключения электроэнергии у обслуживающего персонала должны быть электрические фонари – не менее одного на каждого работника дежурного персонала.

На каждом этаже здания на видном месте должен быть вывешен план эвакуации с этажа (здания). На плане эвакуации кроме путей выхода (стрелками) указываются места размещения средств пожаротушения, телефонов.

Необходимо проводить следующие пожарно-профилактические мероприятия:

а) Организационные мероприятия:

1) противопожарный инструктаж обслуживающего персонала;

2) обучение персонала правилам техники безопасности;

б) Эксплуатационные мероприятия:

1) соблюдение эксплуатационных норм оборудования;

2) обеспечение свободного подхода к оборудованию.

3) содержание в исправности изоляции токоведущих проводников.

в) Технические мероприятия:

1) соблюдение противопожарных мероприятий при устройстве электропроводок, оборудования, систем отопления, вентиляции и освещения. В здании должны присутствовать порошковые огнетушители, установлен рубильник, обесточивающий все помещение. Если возгорание произошло в электроустановке, для его устранения должны использоваться углекислотные огнетушители или порошковые;

2) профилактический осмотр, ремонт и испытание оборудования.

В случае возникновения пожара сотрудники должны предпринять следующие меры [25]:

– сообщить о пожаре в пожарную охрану, задействовать систему оповещения;

– задействовать план эвакуации (открыть запасные двери и включить светоуказатели эвакуационных путей);

– вывести людей в безопасное место в соответствии с планом эвакуации;

– проверить поименно, все ли эвакуированы;

– приступить к тушению пожара первичными средствами;

– встретить пожарные подразделения и сообщить, где могли остаться люди, как туда можно подойти;

– принять меры к эвакуации имущества.

Для тушения пожаров необходимо применять углекислотные и порошковые огнетушители, которые обладают высокой скоростью тушения, большим временем действия, возможностью тушения электроустановок, высокой эффективностью борьбы с огнем. Воду разрешено применять только во вспомогательных помещениях [26].

4.4 Организационные мероприятия обеспечения безопасности

4.4.1 Организационные мероприятия при компоновке рабочей зоны

Рабочее место должно быть организовано так, чтобы иметь возможность максимально удобно выполнять работу. Должны учитываться размеры рабочей зоны, стула, стола и зоны для свободного передвижения во врем работы.

Неправильная организация рабочего места может привести к получению работником производственной травмы или развитие профессионального заболевания [27].

Организация инструментов и рабочего места при выполнении работ в положении сидя должна обеспечивать оптимальное положение работающего, которое достигается регулированием высоты рабочей поверхности, высоты сидения, оборудованием пространства для размещения ног и высотой подставки для ног. Оптимальные параметры рабочего места при работе с ЭВМ представлены в таблице 4.3:

Таблица 4.3 - Оптимальные параметры рабочего места при работе с ЭВМ

Параметры	Значение параметра	Реальные значения
Высота рабочей поверхности стола	От 600 до 800 мм	700
Высота от стола до клавиатуры	Около 20 мм	20

Высота клавиатуры	600-700, мм	600
Удаленность клавиатуры от края стола	Не менее 80 мм	100
Удаленность экрана монитора от глаз	500-700, мм	500
Высота сидения	400-500, мм	500
Угол наклона монитора	0-30, град.	20
Наклон подставки ног	0-20, град.	0

Влияние реализации программы на работу компаний:

- Позволяет экономить время и усилия, затрачиваемые на распознавания данных вручную;

- Экономия денежных средств;

- Высокая точность и надежность распознавания минимизируют количество ошибок в данных

- Требуется оснастить рабочее место и обучить рабочий персонал [28].

ЗАКЛЮЧЕНИЕ

Результатом работы является программа для сегментации данных дистанционного зондирования Земли.

Произведён анализ популярных библиотек для построения свёрточных нейронных сетей. Сделан выбор по использованию библиотеки посредством сравнительного анализа.

Построен алгоритм и реализована нейронная сеть способная распознавать объекты на снимках дистанционного зондирования, а именно: дороги, здания, деревья, траву и автомобили.

При сравнении с другими алгоритмами, разработанная нейросеть показала хорошие результаты и конкурентоспособный уровень.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. The Inria Aerial Image Labeling addresses a core topic in remote sensing: the automatic pixelwise labeling of aerial imagery // [Электронный ресурс] // https://project.inria.fr/aerialimagelabeling/

2. Елизаров А.И., Афонасенко А.В. Методика построения систем распознавания автомобильного номера // Известия Томского политехнического университета. – 2006. – Т. 309. – № 8. – С. 118–121.

3. Le Cun Y., Bengio Y. Convolutional networks for images, speech and time series // The handbook of brain theory and neural networks. – 1998. – V. 7. – № 1. – P. 255–258.

4. Болотова, Ю.А., Спицын В.Г. Сравнение способов обучения модели HTM для задачи распознавания цифр // Молодежь и современные информационные технологии: сборник трудов IX Всероссийской научно_практической конференции студентов, аспирантов и молодых ученых. – Томск: Изд_во СПБ Графикс, 2011. – Т. 1. – С. 252–253.

5. Гонсалес Р., Вудс Р. Цифровая обработка изображений. – М.:Техносфера, 2005. – 1072 с.

6. Болотова, Ю.А., Спицын В.Г., Кермани А.К. Распознавание символов на цветном фоне на основе иерархической временной модели с предобработкой фильтрами Габора // Электромагнитные волны и электронные системы. – 2012. – Т. 16. – № 1. – С. 14–19.

7. Bundzel M., Hashimoto S. Object identification in dynamic images based on the memory-prediction theory of brain function //Journal of Intelligent Learning Systems and Applications. –2010. – V. 2. – № 4. – P. 212–220.

8. 2D Semantic Labeling - Vaihingen data // [Электронный ресурс] // http://www2.isprs.org/commissions/comm3/wg4/2d-sem-label-vaihingen.html

9. Hansen D.W., Hansen J.P., Nielsen M. Eye typing using Markov and active appearance models // Applications of computer vi_ sion. – 2002. – V. 12. – P. 132–136.

10. Rowley H.A., Baluja S., Kanade T. Neural network-based face detection // Pattern anal. mach. intell. – 1998. – V.20. – P. 23–38.

11. Le Cun Y., Huang F., Bottou L. Learning Methods for Generic Object Recognition with Invariance to Pose and Lighting // Proceedings of CVPR'04. – Washington, DC, USA: IEEE Computer Society, 2004. – P. 97–104.

12. George D., Hawkins J. A hierarchical bayesian model of invariant pattern recognition in the visual cortex // Proceedings. 2005 IEEE International Joint Conference on Neural Networks. –Montreal, Canada: IEEE Computer Society, 2005. – V. 3. –P. 1812–1817.

13. Ивашечкин А.П., Василенко А.Ю., Гончаров Б.Д. Методы нахождения особых точек изображения и их дескрипторов // Молодой ученый. – 2016. – № 15. – С. 138–140.

14. Park S., Yoo J.H. Realtime face recognition with SIFT based local feature points for mobile devices // The 1st International Conference on Artificial Intelligence, Modelling and Simulation(AIMS 13). – Malaysia, 2013. – P. 304–308.

15. Tawfiq A., Ahmed J. Object detection and recognition by usingenhanced Speeded Up Robust Feature // International Journal of Computer Science and Network Security. – 2016. – V.16. – № 4. – P. 66–71.

16. Tore V., Chawan P.M. FAST Clustering Based Feature Subset Selection Algorithm for High Dimensional Data // International Journal of Computer Science and Mobile Computing. – 2016. – V. 5. – № 7. – P. 234–238.

17. Mammeri A., Boukerche A., Khiari E. MSERbased text detection and communication algorithm for autonomous vehicles // IEEE Symposium of Computers and Communication. – Messina, Italy, 2016. – P. 456–460.

18. Dalal N., Triggs B. Histograms of Oriented Gradients for Human Detection // IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). – San Diego, USA, 2005. – V. 1. – P. 886–893.

19. Mohey D.E. Enhancement BagofWords Model for Solving the Challenges of Sentiment Analysis // International Journal of Advanced Computer Science and Applications. – 2016. – V. 7. – № 1. – P. 244–251.

20. Kecman V., Melki G. Fast online algorithms for Support Vector Machines // IEEE South East Conference (SoutheastCon 2016). – Virginia, USA, 2016. – P. 26–31.

21. Хайкин С. Нейронные сети: полный курс, 2е изд. – М.: Изд-во «Вильямс», 2006. – 1104 с.

22. Saqib R., Asad N., Omer I.: Automated Number Plate Recognition Using Hough Lines and Template Matching // Proceedings of the World Congress on Engineering and Computer Science 2012 Vol I WCECS 2012, October 24-26, 2012, San Francisco, USA.

23. Hung K., Hsieh C.: A Real-Time Mobile Vehicle License Plate Detection and Recognition // Tamkang Journal of Science and Engineering, Vol. 13, No. 4, pp. 433442 (2010).

24. Zhu, Siyu, "An End-to-End License Plate Localization and Recognition System" (2015) // Department of Electrical Engineering Kate Gleason College of Engineering Rochester Institute of Technology Rochester, New York March 2015.

25. Долин П.А. Справочник по технике безопасности. М.: Энергоатомиздат, 1984 г. – 824 с.

26. Трудовой кодекс Российской Федерации от 30.12.2001 N 197-ФЗ (ред. от 3.07.2016) // Электронный фонд правовой и нормативно-технической документации. URL: http://docs.cntd.ru/document/901807664 (дата обращения: 15.05.2017).

27. ГОСТ Р 50923-96 Дисплеи. Рабочее место оператора. Общие эргономические требования и требования к производственной среде. Методы измерения // Электронный фонд правовой и нормативнотехнической документации. URL: http://docs.cntd.ru/document/1200025975 (дата обращения: 16.05.2017).

28. ГОСТ 22269-76 Система "Человек-машина". Рабочее место оператора. Взаимное расположение элементов рабочего места. Общие эргономические требования // Электронный фонд правовой и нормативно-технической документации. URL: http://docs.cntd.ru/document/1200012834 (дата обращения: 16.05.2017).

Приложение А

Листинг программы

```
import numpy as np
from skimage import io
from glob import glob
from tqdm import tqdm_notebook as tqdm
from sklearn.metrics import confusion_matrix
import random
import itertools

import matplotlib.pyplot as plt
%matplotlib inline

import torch
import torch.nn as nn
import torch.nn.functional as F
import torch.utils.data as data
import torch.optim as optim
import torch.optim.lr_scheduler
import torch.nn.init
from torch.autograd import Variable


WINDOW_SIZE = (256, 256) # Patch size
STRIDE = 32 # Stride for testing
IN_CHANNELS = 3
FOLDER = "./ISPRS_dataset/"
BATCH_SIZE = 5 # Number of samples in a mini-batch

LABELS = ["roads", "buildings", "low veg.", "trees", "cars", "clutter"] # Label names
N_CLASSES = len(LABELS) # Number of classes
WEIGHTS = torch.ones(N_CLASSES) # Weights for class balancing
CACHE = True

DATASET = 'Vaihingen'
MAIN_FOLDER = FOLDER + 'Vaihingen/'
DATA_FOLDER = MAIN_FOLDER + 'top/top_mosaic_09cm_area{}.tif'
LABEL_FOLDER = MAIN_FOLDER + 'gts_for_participants/top_mosaic_09cm_area{}.tif'
ERODED_FOLDER = MAIN_FOLDER + 'gts_eroded_for_participants/top_mosaic_09cm_area{}_noBoundary.tif'

# ISPRS color palette
palette = {0 : (255, 255, 255), # Impervious surfaces (white)
           1 : (0, 0, 255),     # Buildings (blue)
           2 : (0, 255, 255),   # Low vegetation (cyan)
           3 : (0, 255, 0),     # Trees (green)
           4 : (255, 255, 0),   # Cars (yellow)
           5 : (255, 0, 0),     # Clutter (red)
           6 : (0, 0, 0)}       # Undefined (black)

invert_palette = {v: k for k, v in palette.items()}

def convert_to_color(arr_2d, palette=palette):
    """ Numeric labels to RGB-color encoding """
    arr_3d = np.zeros((arr_2d.shape[0], arr_2d.shape[1], 3), dtype=np.uint8)

    for c, i in palette.items():
        m = arr_2d == c
        arr_3d[m] = i

    return arr_3d

def convert_from_color(arr_3d, palette=invert_palette):
    """ RGB-color encoding to grayscale labels """
    arr_2d = np.zeros((arr_3d.shape[0], arr_3d.shape[1]), dtype=np.uint8)

    for c, i in palette.items():
        m = np.all(arr_3d == np.array(c).reshape(1, 1, 3), axis=2)
        arr_2d[m] = i

    return arr_2d


def get_random_pos(img, window_shape):
    """ Extract of 2D random patch of shape window_shape in the image """
    w, h = window_shape
    W, H = img.shape[-2:]
    x1 = random.randint(0, W - w - 1)
```

```
    x2 = x1 + w
    y1 = random.randint(0, H - h - 1)
    y2 = y1 + h
    return x1, x2, y1, y2

def CrossEntropy2d(input, target, weight=None, size_average=True):
    """ 2D version of the cross entropy loss """
    dim = input.dim()
    if dim == 2:
        return F.cross_entropy(input, target, weight, size_average)
    elif dim == 4:
        output = input.view(input.size(0),input.size(1), -1)
        output = torch.transpose(output,1,2).contiguous()
        output = output.view(-1,output.size(2))
        target = target.view(-1)
        return F.cross_entropy(output, target,weight, size_average)
    else:
        raise ValueError('Expected 2 or 4 dimensions (got {})'.format(dim))

def accuracy(input, target):
    return 100 * float(np.count_nonzero(input == target)) / target.size

def sliding_window(top, step=10, window_size=(20,20)):
    """ Slide a window_shape window across the image with a stride of step """
    for x in range(0, top.shape[0], step):
        if x + window_size[0] > top.shape[0]:
            x = top.shape[0] - window_size[0]
        for y in range(0, top.shape[1], step):
            if y + window_size[1] > top.shape[1]:
                y = top.shape[1] - window_size[1]
            yield x, y, window_size[0], window_size[1]

def count_sliding_window(top, step=10, window_size=(20,20)):
    """ Count the number of windows in an image """
    c = 0
    for x in range(0, top.shape[0], step):
        if x + window_size[0] > top.shape[0]:
            x = top.shape[0] - window_size[0]
        for y in range(0, top.shape[1], step):
            if y + window_size[1] > top.shape[1]:
                y = top.shape[1] - window_size[1]
            c += 1
    return c

def grouper(n, iterable):
    """ Browse an iterator by chunk of n elements """
    it = iter(iterable)
    while True:
        chunk = tuple(itertools.islice(it, n))
        if not chunk:
            return
        yield chunk

def metrics(predictions, gts, label_values=LABELS):
    cm = confusion_matrix(
            gts,
            predictions,
            range(len(label_values)))

    print("Confusion matrix :")
    print(cm)

    print("---")

    # Compute global accuracy
    total = sum(sum(cm))
    accuracy = sum([cm[x][x] for x in range(len(cm))])
    accuracy *= 100 / float(total)
    print("{} pixels processed".format(total))
    print("Total accuracy : {}%".format(accuracy))

    print("---")

    # Compute F1 score
    F1Score = np.zeros(len(label_values))
    for i in range(len(label_values)):
        try:
            F1Score[i] = 2. * cm[i,i] / (np.sum(cm[i,:]) + np.sum(cm[:,i]))
        except:
            # Ignore exception if there is no element in class i for test set
```

```
            pass
    print("F1Score :")
    for l_id, score in enumerate(F1Score):
        print("{}: {}".format(label_values[l_id], score))

    print("---")

    # Compute kappa coefficient
    total = np.sum(cm)
    pa = np.trace(cm) / float(total)
    pe = np.sum(np.sum(cm, axis=0) * np.sum(cm, axis=1)) / float(total*total)
    kappa = (pa - pe) / (1 - pe);
    print("Kappa: " + str(kappa))
    return accuracy

# Dataset class

class ISPRS_dataset(torch.utils.data.Dataset):
    def __init__(self, ids, data_files=DATA_FOLDER, label_files=LABEL_FOLDER,
                           cache=False, augmentation=True):
        super(ISPRS_dataset, self).__init__()

        self.augmentation = augmentation
        self.cache = cache

        # List of files
        self.data_files = [DATA_FOLDER.format(id) for id in ids]
        self.label_files = [LABEL_FOLDER.format(id) for id in ids]

        # Sanity check : raise an error if some files do not exist
        for f in self.data_files + self.label_files:
            if not os.path.isfile(f):
                raise KeyError('{} is not a file !'.format(f))

        # Initialize cache dicts
        if self.cache:
            self.data_cache_ = {}
            self.label_cache_ = {}


    def __len__(self):
        # Default epoch size is 10 000 samples
        return 10000

    @classmethod
    def data_augmentation(cls, *arrays, flip=True, mirror=True):
        will_flip, will_mirror = False, False
        if flip and random.random() < 0.5:
            will_flip = True
        if mirror and random.random() < 0.5:
            will_mirror = True

        results = []
        for array in arrays:
            if will_flip:
                if len(array.shape) == 2:
                    array = array[::-1, :]
                else:
                    array = array[:, ::-1, :]
            if will_mirror:
                if len(array.shape) == 2:
                    array = array[:, ::-1]
                else:
                    array = array[:, :, ::-1]
            results.append(np.copy(array))

        return tuple(results)

    def __getitem__(self, i):
        # Pick a random image
        random_idx = random.randint(0, len(self.data_files) - 1)

        # If the tile hasn't been loaded yet, put in cache
        if random_idx in self.data_cache_.keys():
            data = self.data_cache_[random_idx]
        else:
            # Data is normalized in [0, 1]
            data   =   1/255   *   np.asarray(io.imread(self.data_files[random_idx]).transpose((2,0,1)),
dtype='float32')
            if self.cache:
```

```
                self.data_cache_[random_idx] = data

        if random_idx in self.label_cache_.keys():
            label = self.label_cache_[random_idx]
        else:
            # Labels are converted from RGB to their numeric values
            label       =       np.asarray(convert_from_color(io.imread(self.label_files[random_idx])),
dtype='int64')
            if self.cache:
                self.label_cache_[random_idx] = label

        # Get a random patch
        x1, x2, y1, y2 = get_random_pos(data, WINDOW_SIZE)
        data_p = data[:, x1:x2,y1:y2]
        label_p = label[x1:x2,y1:y2]

        # Data augmentation
        data_p, label_p = self.data_augmentation(data_p, label_p)

        # Return the torch.Tensor values
        return (torch.from_numpy(data_p),
                torch.from_numpy(label_p))

class MyNet(nn.Module):
    # MyNet network
    @staticmethod
    def weight_init(m):
        if isinstance(m, nn.Linear):
            torch.nn.init.kaiming_normal(m.weight.data)

    def __init__(self, in_channels=IN_CHANNELS, out_channels=N_CLASSES):
        super(MyNet, self).__init__()
        self.pool = nn.MaxPool2d(2, return_indices=True)
        self.unpool = nn.MaxUnpool2d(2)

        self.conv1_1 = nn.Conv2d(in_channels, 64, 3, padding=1)
        self.conv1_1_bn = nn.BatchNorm2d(64)
        self.conv1_2 = nn.Conv2d(64, 64, 3, padding=1)
        self.conv1_2_bn = nn.BatchNorm2d(64)

        self.conv2_1 = nn.Conv2d(64, 128, 3, padding=1)
        self.conv2_1_bn = nn.BatchNorm2d(128)
        self.conv2_2 = nn.Conv2d(128, 128, 3, padding=1)
        self.conv2_2_bn = nn.BatchNorm2d(128)

        self.conv3_1 = nn.Conv2d(128, 256, 3, padding=1)
        self.conv3_1_bn = nn.BatchNorm2d(256)
        self.conv3_2 = nn.Conv2d(256, 256, 3, padding=1)
        self.conv3_2_bn = nn.BatchNorm2d(256)
        self.conv3_3 = nn.Conv2d(256, 256, 3, padding=1)
        self.conv3_3_bn = nn.BatchNorm2d(256)

        self.conv4_1 = nn.Conv2d(256, 512, 3, padding=1)
        self.conv4_1_bn = nn.BatchNorm2d(512)
        self.conv4_2 = nn.Conv2d(512, 512, 3, padding=1)
        self.conv4_2_bn = nn.BatchNorm2d(512)
        self.conv4_3 = nn.Conv2d(512, 512, 3, padding=1)
        self.conv4_3_bn = nn.BatchNorm2d(512)

        self.conv5_1 = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_1_bn = nn.BatchNorm2d(512)
        self.conv5_2 = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_2_bn = nn.BatchNorm2d(512)
        self.conv5_3 = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_3_bn = nn.BatchNorm2d(512)

        self.conv5_3_D = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_3_D_bn = nn.BatchNorm2d(512)
        self.conv5_2_D = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_2_D_bn = nn.BatchNorm2d(512)
        self.conv5_1_D = nn.Conv2d(512, 512, 3, padding=1)
        self.conv5_1_D_bn = nn.BatchNorm2d(512)

        self.conv4_3_D = nn.Conv2d(512, 512, 3, padding=1)
        self.conv4_3_D_bn = nn.BatchNorm2d(512)
        self.conv4_2_D = nn.Conv2d(512, 512, 3, padding=1)
        self.conv4_2_D_bn = nn.BatchNorm2d(512)
        self.conv4_1_D = nn.Conv2d(512, 256, 3, padding=1)
        self.conv4_1_D_bn = nn.BatchNorm2d(256)
```

```
        self.conv3_3_D = nn.Conv2d(256, 256, 3, padding=1)
        self.conv3_3_D_bn = nn.BatchNorm2d(256)
        self.conv3_2_D = nn.Conv2d(256, 256, 3, padding=1)
        self.conv3_2_D_bn = nn.BatchNorm2d(256)
        self.conv3_1_D = nn.Conv2d(256, 128, 3, padding=1)
        self.conv3_1_D_bn = nn.BatchNorm2d(128)

        self.conv2_2_D = nn.Conv2d(128, 128, 3, padding=1)
        self.conv2_2_D_bn = nn.BatchNorm2d(128)
        self.conv2_1_D = nn.Conv2d(128, 64, 3, padding=1)
        self.conv2_1_D_bn = nn.BatchNorm2d(64)

        self.conv1_2_D = nn.Conv2d(64, 64, 3, padding=1)
        self.conv1_2_D_bn = nn.BatchNorm2d(64)
        self.conv1_1_D = nn.Conv2d(64, out_channels, 3, padding=1)

        self.apply(self.weight_init)

    def forward(self, x):
        # Encoder block 1
        x = self.conv1_1_bn(F.relu(self.conv1_1(x)))
        x = self.conv1_2_bn(F.relu(self.conv1_2(x)))
        x, mask1 = self.pool(x)

        # Encoder block 2
        x = self.conv2_1_bn(F.relu(self.conv2_1(x)))
        x = self.conv2_2_bn(F.relu(self.conv2_2(x)))
        x, mask2 = self.pool(x)

        # Encoder block 3
        x = self.conv3_1_bn(F.relu(self.conv3_1(x)))
        x = self.conv3_2_bn(F.relu(self.conv3_2(x)))
        x = self.conv3_3_bn(F.relu(self.conv3_3(x)))
        x, mask3 = self.pool(x)

        # Encoder block 4
        x = self.conv4_1_bn(F.relu(self.conv4_1(x)))
        x = self.conv4_2_bn(F.relu(self.conv4_2(x)))
        x = self.conv4_3_bn(F.relu(self.conv4_3(x)))
        x, mask4 = self.pool(x)

        # Encoder block 5
        x = self.conv5_1_bn(F.relu(self.conv5_1(x)))
        x = self.conv5_2_bn(F.relu(self.conv5_2(x)))
        x = self.conv5_3_bn(F.relu(self.conv5_3(x)))
        x, mask5 = self.pool(x)

        # Decoder block 5
        x = self.unpool(x, mask5)
        x = self.conv5_3_D_bn(F.relu(self.conv5_3_D(x)))
        x = self.conv5_2_D_bn(F.relu(self.conv5_2_D(x)))
        x = self.conv5_1_D_bn(F.relu(self.conv5_1_D(x)))

        # Decoder block 4
        x = self.unpool(x, mask4)
        x = self.conv4_3_D_bn(F.relu(self.conv4_3_D(x)))
        x = self.conv4_2_D_bn(F.relu(self.conv4_2_D(x)))
        x = self.conv4_1_D_bn(F.relu(self.conv4_1_D(x)))

        # Decoder block 3
        x = self.unpool(x, mask3)
        x = self.conv3_3_D_bn(F.relu(self.conv3_3_D(x)))
        x = self.conv3_2_D_bn(F.relu(self.conv3_2_D(x)))
        x = self.conv3_1_D_bn(F.relu(self.conv3_1_D(x)))

        # Decoder block 2
        x = self.unpool(x, mask2)
        x = self.conv2_2_D_bn(F.relu(self.conv2_2_D(x)))
        x = self.conv2_1_D_bn(F.relu(self.conv2_1_D(x)))

        # Decoder block 1
        x = self.unpool(x, mask1)
        x = self.conv1_2_D_bn(F.relu(self.conv1_2_D(x)))
        x = F.log_softmax(self.conv1_1_D(x))
        return x

# instantiate the network
net = MyNet()

import os
```

```
import urllib
import urllib.request

vgg16_weights = torch.load('./vgg16_bn-6c64b313.pth')
mapped_weights = {}
for k_vgg, k_MyNet in zip(vgg16_weights.keys(), net.state_dict().keys()):
    if "features" in k_vgg:
        mapped_weights[k_MyNet] = vgg16_weights[k_vgg]
        print("Mapping {} to {}".format(k_vgg, k_MyNet))

try:
    net.load_state_dict(mapped_weights)
    print("Loaded VGG-16 weights in MyNet !")
except:
    # Ignore missing keys
    pass

net.cuda()

# Load the datasets
all_files = sorted(glob(LABEL_FOLDER.replace('{}', '*')))
all_ids = [f.split('area')[-1].split('.')[0] for f in all_files]

train_ids = ['1', '3', '23', '26', '7', '11', '13', '28', '17', '32', '34', '37']
test_ids = ['5', '21', '15', '30']

print("Tiles for training : ", train_ids)
print("Tiles for testing : ", test_ids)

train_set = ISPRS_dataset(train_ids, cache=CACHE)
train_loader = torch.utils.data.DataLoader(train_set,batch_size=BATCH_SIZE)

base_lr = 0.01
params_dict = dict(net.named_parameters())
params = []
for key, value in params_dict.items():
    if '_D' in key:
        # Decoder weights are trained at the nominal learning rate
        params += [{'params':[value],'lr': base_lr}]
    else:
        # Encoder weights are trained at lr / 2 (we have VGG-16 weights as initialization)
        params += [{'params':[value],'lr': base_lr / 2}]

optimizer = optim.SGD(net.parameters(), lr=base_lr, momentum=0.9, weight_decay=0.0005)
# We define the scheduler
scheduler = optim.lr_scheduler.MultiStepLR(optimizer, [25, 35, 45], gamma=0.1)

def      test(net,      test_ids,      all=False,      stride=WINDOW_SIZE[0],      batch_size=BATCH_SIZE,
window_size=WINDOW_SIZE):
    # Use the network on the test set
    test_images =  (1 / 255 * np.asarray(io.imread(DATA_FOLDER.format(id)), dtype='float32') for id in
test_ids)
    test_labels = (np.asarray(io.imread(LABEL_FOLDER.format(id)), dtype='uint8') for id in test_ids)
    eroded_labels = (convert_from_color(io.imread(ERODED_FOLDER.format(id))) for id in test_ids)
    all_preds = []
    all_gts = []

    # Switch the network to inference mode
    net.eval()

    for  img,  gt,  gt_e  in  tqdm(zip(test_images,  test_labels,  eroded_labels),  total=len(test_ids),
leave=False):
        pred = np.zeros(img.shape[:2] + (N_CLASSES,))

        total = count_sliding_window(img, step=stride, window_size=window_size) // batch_size
        for   i,   coords   in   enumerate(tqdm(grouper(batch_size,   sliding_window(img,   step=stride,
window_size=window_size)), total=total, leave=False)):
            # Display in progress results
            if i > 0 and total > 10 and i % int(10 * total / 100) == 0:
                    _pred = np.argmax(pred, axis=-1)
                    fig = plt.figure()
                    fig.add_subplot(1,3,1)
                    plt.imshow(np.asarray(255 * img, dtype='uint8'))
                    fig.add_subplot(1,3,2)
                    plt.imshow(convert_to_color(_pred))
                    fig.add_subplot(1,3,3)
                    plt.imshow(gt)
                    clear_output()
                    plt.show()
```

```
            # Build the tensor
            image_patches = [np.copy(img[x:x+w, y:y+h]).transpose((2,0,1)) for x,y,w,h in coords]
            image_patches = np.asarray(image_patches)
            image_patches = Variable(torch.from_numpy(image_patches).cuda(), volatile=True)

            # Do the inference
            outs = net(image_patches)
            outs = outs.data.cpu().numpy()

            # Fill in the results array
            for out, (x, y, w, h) in zip(outs, coords):
                out = out.transpose((1,2,0))
                pred[x:x+w, y:y+h] += out
            del(outs)

        pred = np.argmax(pred, axis=-1)

        # Display the result
        clear_output()
        fig = plt.figure()
        fig.add_subplot(1,3,1)
        plt.imshow(np.asarray(255 * img, dtype='uint8'))
        fig.add_subplot(1,3,2)
        plt.imshow(convert_to_color(pred))
        fig.add_subplot(1,3,3)
        plt.imshow(gt)
        plt.show()

        all_preds.append(pred)
        all_gts.append(gt_e)

        clear_output()
        # Compute some metrics
        metrics(pred.ravel(), gt_e.ravel())
        accuracy = metrics(np.concatenate([p.ravel() for p in all_preds]), np.concatenate([p.ravel()
for p in all_gts]).ravel())
    if all:
        return accuracy, all_preds, all_gts
    else:
        return accuracy


from IPython.display import clear_output

def train(net, optimizer, epochs, scheduler=None, weights=WEIGHTS, save_epoch = 5):
    losses = np.zeros(1000000)
    mean_losses = np.zeros(100000000)
    weights = weights.cuda()

    criterion = nn.NLLLoss2d(weight=weights)
    iter_ = 0

    for e in range(1, epochs + 1):
        if scheduler is not None:
            scheduler.step()
        net.train()
        for batch_idx, (data, target) in enumerate(train_loader):
            data, target = Variable(data.cuda()), Variable(target.cuda())
            optimizer.zero_grad()
            output = net(data)
            loss = CrossEntropy2d(output, target, weight=weights)
            loss.backward()
            optimizer.step()

            losses[iter_] = loss.data[0]
            mean_losses[iter_] = np.mean(losses[max(0,iter_-100):iter_])

            if iter_ % 100 == 0:
                clear_output()
                rgb = np.asarray(255 * np.transpose(data.data.cpu().numpy()[0],(1,2,0)), dtype='uint8')
                pred = np.argmax(output.data.cpu().numpy()[0], axis=0)
                gt = target.data.cpu().numpy()[0]
                print('Train (epoch {}/{}) [{}/{} ({:.0f}%)]\tLoss: {:.6f}\tAccuracy: {}'.format(
                    e, epochs, batch_idx, len(train_loader),
                    100. * batch_idx / len(train_loader), loss.data[0], accuracy(pred, gt)))
                plt.plot(mean_losses[:iter_]) and plt.show()
                fig = plt.figure()
                fig.add_subplot(131)
                plt.imshow(rgb)
                plt.title('RGB')
```

```
                fig.add_subplot(132)
                plt.imshow(convert_to_color(gt))
                plt.title('Ground truth')
                fig.add_subplot(133)
                plt.title('Prediction')
                plt.imshow(convert_to_color(pred))
                plt.show()
            iter_ += 1

            del(data, target, loss)

        if e % save_epoch == 0:
            # We validate with the largest possible stride for faster computing
            acc = test(net, test_ids, all=False, stride=min(WINDOW_SIZE))
            torch.save(net.state_dict(), './MyNet256_epoch{}_{}'.format(e, acc))
    torch.save(net.state_dict(), './MyNet_final')


train(net, optimizer, 10, scheduler)

net.load_state_dict(torch.load('./MyNet_final'))
_, all_preds, all_gts = test(net, test_ids, all=True, stride=32)

for p, id_ in zip(all_preds, test_ids):
    img = convert_to_color(p)
    plt.imshow(img) and plt.show()
    io.imsave('./inference_tile_{}.png'.format(id_), img)
```

Приложение Б

Раздел на иностранном языке

Раздел 1

Analytical Review

Студент:

Группа	ФИО	Подпись	Дата
8ИМ6Б	Аркалыков Е.У.		

Консультант ОИТ:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Мирошниченко Евгений Александрович	к.т.н.		

Консультант – лингвист ОИЯ:

Должность	ФИО	Ученая степень, звание	Подпись	Дата
Доцент	Диденко Анастасия Владимировна	к.ф.н.		

ANALYTICAL REVIEW

The basic approach for semantic segmentation of images is a combination of three algorithms: detectors, handles and classifiers. These algorithms determine the basic image parameters, mark out objects and classify them. Basic picture settings include brightness, color, texture, borders and corners of objects, and the like. The parameters of the image are detected using algorithms-detectors, and are then mathematically described using the algorithms of descriptors that result in a new marking out of objects in the image. Then it is necessary to define to what class the objects belongs; this procedure is engaged in algorithms of classifiers.

1. Overview of semantic image segmentation algorithms

1.1 Detectors and descriptors

The SIFT detector (scale invariant feature transform) is based on the use of scalable spaces- a set of all sorts of filters-smoothed versions of one image. When using a Gaussian filter, this scalable space becomes invariant to shifts, rotations and not displacing local extrema of the scale. To improve the invariance to scale, the detector uses the original image, taken at different scales by building a pyramid of gaussians: all scalable space is divided into so-called octaves, so that the next octave take two times more space than the previous one. In each octave there is a certain number of gaussians images, from which the difference of gaussians constituting a separate pyramid is subtracted (Fig. 1) [1]

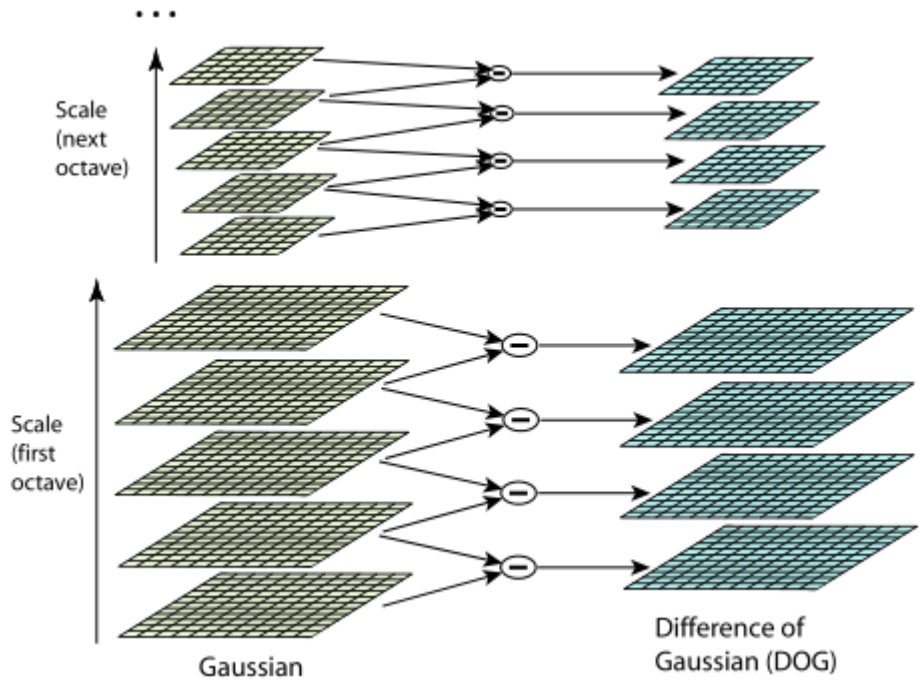

Figure 1 - Pyramids of gaussians and the difference of gaussians

It remains only to determine the special (key) points. The main criterion for a singular point is that it should represent local extrema of the gaussians difference. To locate specific points the method shown in Figure 2 is used. For each point in the image the difference of gaussians is compared with neighboring points both in this image, and in the images next to it. Accordingly, if this point is greater or smaller than these, it is considered a local extrema.

Next, this point is checked: with the help of Taylor polynomial, the offset from the exact extrema is identified as well as the contrast value of the guassians difference in order to find a point on the boundary of the object by using of Hesse's matrix.

After that the orientation of the key point is calculated based on directions of nearby points gradients. Now this point is ready to be described by its descriptor.

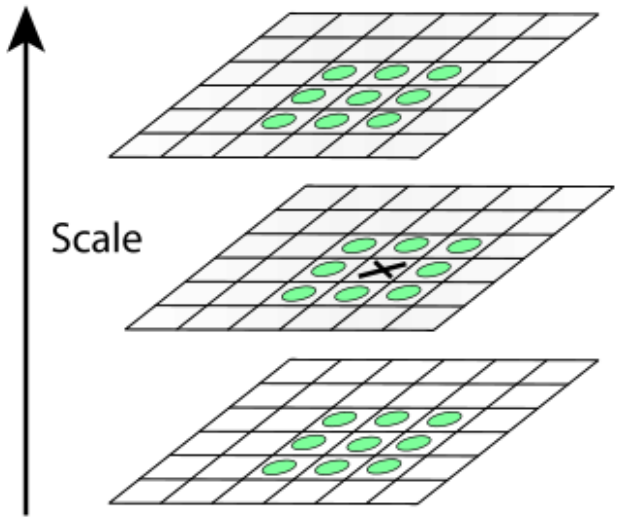

Figure 2 - Determination of local extremum on the Pyramid of Gaussians Difference

The SURF detector (speeded up robust features) is a modernisation of the SIFT detector, but instead of the Gauss function its approximation is used by a rectangular filter 9x9, which accelerates the result of the algorithm. Accordingly, it shares the same advantages and disadvantages as the SIFT detector. [1]

The fast (features from accelerated segment test) algorithm does not require the calculation of the brightness derivatives, but rather the brightness in the circle from the point is checked. First, a rapid test is conducted being four points away from the investigated one, and then the rest are checked. In this case, the values of the quick test are not counted in the full enumeration. The number of checks and their sequence are determined on the training sample, and the presence of the angle is determined by only a few dozen checks. Thus, the algorithm works quickly enough to be used in real-time systems. [2.3]

The idea behind the MSER algorithm (Maximally stable extremal regions) is to determine the intensity of the image pixels and compare them to a certain threshold (if the pixel intensity is greater than the threshold, consider it white or black). Thus, the pyramids of images are built, at the beginning of which there are white images, and at the end black. Such a pyramid allows to construct a set of the connected intensity component which will be invariant to affine transformations. This algorithm works consistently, but has partial invariance regarding scale.

In addition to the detector, there is a special SIFT descriptor, which is a local histogram of the image gradient directions. The output descriptor is a vector of values describing the surrounding area of this key point.

The principle of its operation is to determine the gradient of the image, its direction and module multiplied by the weight for each of the four sectors around the singular point. For each sector there is a histogram of gradient directions, each occurrence of which is weighed by the module of this gradient.

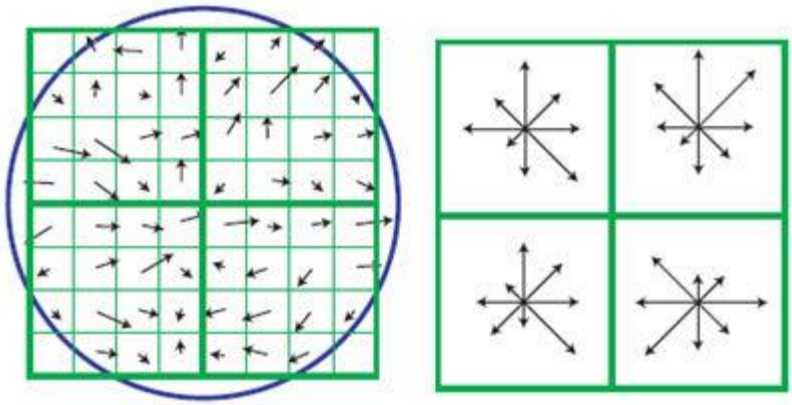

Figure 3 - The construction of SIFT descriptor

Thus, the main advantages of ligaments and SIFT descriptor are: invariance regarding the scale and rotation of the image, resistance to illumination change and computational efficiency. [1,2,3]

The SURF descriptor is similar to SIFT descriptor, but instead of using weighted histogram of gradients it uses feedback from the original image on Haar wavelets. Square area is built around the point of interest and is oriented relative to the preferred direction of the square and is divided into square sectors in which a response to the Haar wavelets aimed vertically and horizontally is calculated. Data responses are weighed and summarized for each of the sectors forming the first part of the descriptor.

In order to take into account not only the fact of change of brightness from a point to point, but also to retain information on the direction of change of brightness, it uses the sum of absolute luminance. As well as SIFT descriptor, SURF is invariant to changes in brightness and scope. [1,3]

HOG (Histogram of oriented gradients) descriptor of key points, based on calculating directions of the gradient in local areas of the image. Its algorithm is similar to the SIFT algorithm, but differs in the way the computation is carried out on a dense grid of evenly distributed cells. This algorithm also uses local contrast normalization to increase accuracy.

It is based on the assumption that the appearance and shape of an object can be described using the distribution of intensity gradients or directions. Therefore, when the algorithm is working, the image is divided into small cohesive areas called cells, and a histogram of gradient directions and edges for pixels within a cell is calculated for each cell. The output of the descriptor is the combination of these histograms. The advantages of this algorithm include the fact that when working with local cells, it has a low sensitivity to geometric and photometric transformations. [1,2]

1.2 Classifiers

Bag of words algorithm is among the most common classes of algorithms for image classification. Its name largely determines the principle of its operation, because it actually uses the histogram of occurrences of certain patterns in the image. [4]

Basically, this algorithm was used to classify texts, which also builds a histogram of the occurrences within the document of words from the previously prepared dictionary, but can be efficiently used for other tasks.

Main steps of this algorithm are:

1. Defining specific (key) points in the image;
2. Building the descriptors of these points;
3. Conducting clustering of these descriptors belonging to the objects of training sample (that is, to fill the dictionary "words");
4. Building a description of each image as a normalized histogram of "words" occurrence (calculation for each of clusters the quantity of special points of an image classified to it);
5. Building a classifier that uses description from step 4.

For the first two steps any detectors and descriptors can be used if they are resistant to affine transformations and changes in illumination.

The main drawback of this classifier in a large vocabulary is size, and its size should not exceed a certain value at which it is resistant to noise. Also, the bag-of-words does not take into account spatial information about the object that the presence of similar descriptors of singular points of different objects in the image descriptions may be the same. [5]

Support vector machine (SVM) is one of the most popular methods of training classifiers. Initially this method was used for the tasks of binary classification, but it is easily generalized for problems with a large number of classes, as well as the problem of recovering the regression. To train the algorithm it is proposed to build a linear threshold classifier

$$\operatorname{sgn}\left(\sum_{j=1}^{n} w_j x_j - w_0\right). \quad (1)$$

The equation $\langle w, x\rangle = w_0$ describes a hyperplane that separates the classes. If we assume that the sample is linearly separable, then it becomes obvious that this plane is not the only one and there are other provisions also in the split sample. [4]

The idea of the method is to select such$w$ andw_0 those which would be optimal from the perspective of any particular criterion. Initially, the method of classification arose from heuristic assumptions, but now has a theoretical justification.

In order for the hyperplane to be more stable shared classes, it must stand from the sampling points at maximum distance. This is achieved when the norm of the vector is $w$minimal. The support vectors are called the set lying on the boundaries of the regions separated by a hyperplane.

The algorithm is also easily summarized for linearly inseparable data by moving from the original space of the characteristic descriptions of objects to another medium with the help of function cores. The advantages of this algorithm in addition to the prevalence and ease can be attributed to the small training sample, in which the classifier will give an acceptable result. This is also a drawback – the algorithm does not use all the set, but only a small fraction on the boundary. [4, 5]

2. Artificial Neural Networks

Artificial Neural networks (ANN) are the algorithms that simulate the way information is processed by a human brain. They represent a distributed parallel processor, consisting of elementary particles of information processing (neurons) and connections between them (synaptic weight), allowing to accumulate knowledge from the environment and used in the process of training. [6]

Artificial neural networks consist of layers of neurons and have different architectures. Classical networks are fully connected, i.e. a network in which every neuron of one layer is connected with neurons of the previous layer. Processing of information by a neuron in such networks is carried out by multiplying all input signals on corresponding synaptic communications, their summation and processing by function of activation of a neuron, after which the signal spreads further over the network. Network training consists of readjusting the synaptic weights depending on the network output.

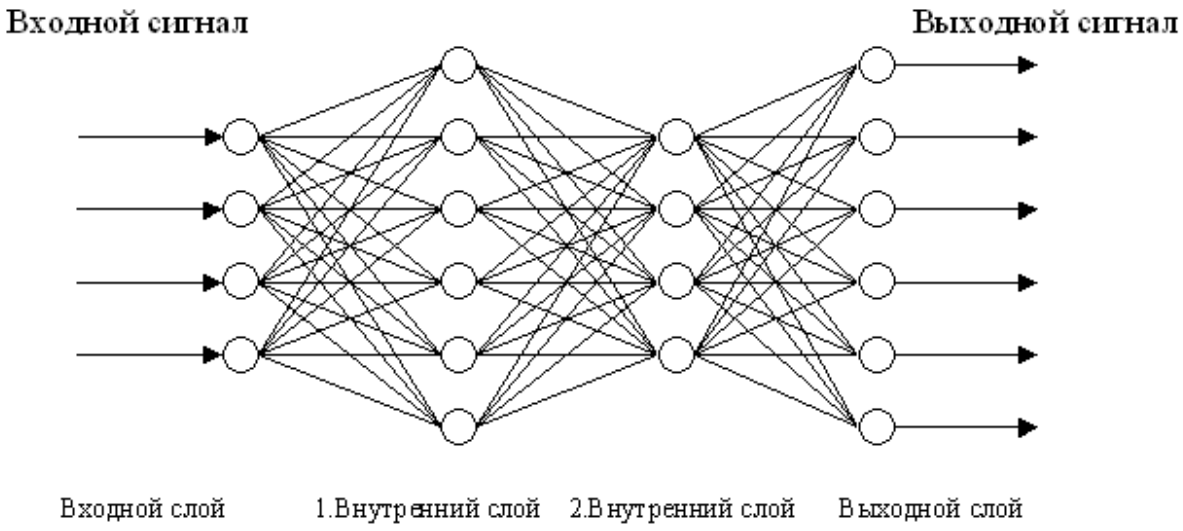

Figure 4 - A fully connected ANN with two internal layers

The main disadvantage of the classical neural networks for image processing is the large size of the input vector (i.e., each pixel of the image). As a result of the growing number of neurons in each layer and for large images it gets too bulky and heavy for training. Classic neural network can take into account the topology of the original image, as taking it as a whole. [7]

Convolutional neural networks are deprived of these disadvantages, which have special convolutional and subdirectory layers. The basic idea of these layers is creating shared weights, i.e. some of the neurons use the same weighting factors and are combined into a map of features, each neuron is connected to the previous layer. Each neuron of such layer performs the operation of mathematical convolution of some area of the previous layer and such layer is accordingly called convolutional and models some peculiarities of human vision, responsible for detection of a specific characteristic. The subsampling layers are responsible for the reduction of the input vector into a certain number of times (usually twice).

Thus, convolutional neural networks have a much lower number of adjustable weights that allows the network to learn the synthesis of information, not memorization. Such networks are resistant to scale, shift and rotation.

2.1 Neural network

Biological neural network provides the possibility of effective human interaction with the environment by performing a variety of tasks: pattern recognition; signal processing of the sensory organs; motor skills and many others. In this case, these tasks are often performed repeatedly and sequentially within hundreds of milliseconds, which is much faster than solving similar problems on modern computers.

This is due to the fact that the methods of information processing by the human brain are very different from the methods of processing information of computer machines, due to the special structure of the human brain, consisting of a large number of *neurons* (about 10 billion), connected to a single network. Each neuron consists of the nucleus, the body of the cell and the sprouts (*dendrites*, receiving input signals, and *axons* transmitting) connecting them to each other, and the main task of each of them is to transmit electrochemical impulse all over the network. This is ensured by the fact that the biological neural network has a high degree of connectivity-each neuron is connected by several thousands of others and each such connection has the *power of synaptic communication*, determining the amplification

or weakening of the impulse transmission between these neurons. Thus, the structure of the human brain is not static and constantly changing, setting these connections and allowing to build its own rules based on the "experience". In other words, the structure of the human brain allows learning. [7.8]

2.2 Neuron Model

A neuron is the basic unit of information processing in ANN. It is a simple processor that transforms some input signals into output by a certain rule. This output signal is delivered to other neurons by weighted connections. The general structure is shown in Figure 5.

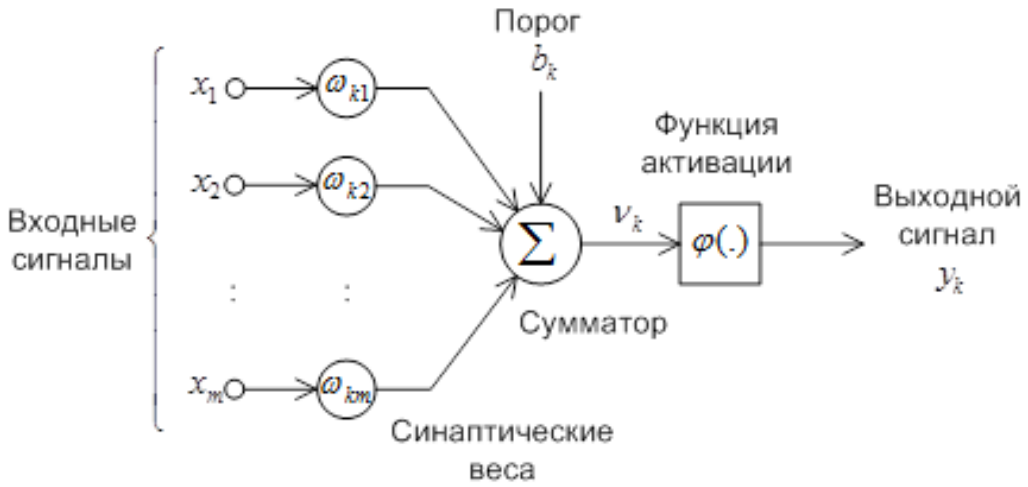

Figure 5 - Neuron model

In the model of a neuron it is possible to distinguish three basic elements [6]:

4. Presence of the suspended bonds – *synapses* possessing weight. Thus, each input signalx_j coming to the input$j$ of a neuron$k$ is multiplied by weightω_{kj}. Unlike the biological neural network, these weights can be both positive and negative.
5. *Adder* adds input signals to the balance of synapses. There is also a *threshold element* in the model that reflects the amplification or weakening of the signal supplied to the activation function.

6. *The activation function (compression)* is a function that limits the amplitude of the output signal. Usually adjusts the signal in intervals from 0 to 1 or from-1 to 1.

Next there will be the functioning of the neuron using mathematical equations described:

$$u_k = \sum_{j=1}^{m} w_{kj} x_j \; ; \tag{2}$$

$$y_k = \varphi(u_k + b_k) \; , \tag{3}$$

where $x_1, x_2, \dots, x_m$– input signals; $w_{k1}, w_{k2}, \dots, w_{km}$– synaptic of the neuron's weight; u_k– a linear combination of inputs; $\varphi()$– activation function; y_k - the output of the neuron. The postsynaptic potential is computed by the formula 3 and is due to the effect of the affine transformation from the threshold element b_k.

$$v_k = u_k + b_k \; . \tag{4}$$

This threshold is an external parameter for the neuron $k$and can be transformed as a new synapse with input signal $x_0 = +1$and weight$w_{k0} = b_k$. Thus, equations 1 and 2 would look this way:

$$u_k = \sum_{j=0}^{m} w_{kj} x_j \; ; \tag{5}$$

$$y_k = \varphi u_k \; . \tag{6}$$

2.3 Activation functions

Activation functions (compression, excitation)-is a function that calculates the output signal of an artificial neuron. One of the most common types of functions are:

3. *A threshold function* or a function of a single jump.

Accordingly, if the input value is below the threshold, the value of the activation function is equal to the minimum, or the maximum. It is sometimes referred to as the Heavyside function.

Mathematical Description of this function:

$$\varphi(v)=\begin{cases}1, & \&v\geq 0\\ 0, & \&v<0\end{cases}. \qquad (7)$$

Linear threshold or hysteresis function is piecewise given and has three intervals, two of which are the minimum and maximum threshold, and the third increases. [6,7]

$$\varphi(v)=\begin{cases}1, & v\geq +\frac{1}{2}\\ |v|, & +\frac{1}{2}>v>-\frac{1}{2}\\ 0, & v\leq -\frac{1}{2}\end{cases}. \qquad (8)$$

Here the gain in the linear region of the operator is assumed to be a unit and this function can be viewed as an approximation of the nonlinear amplifier. There are two special forms: *a linear combiner* is a linear area of the operator which does not reach the saturation threshold; *the threshold function* - if the amplification factor of a linearly region is adopted as infinitely large. [6]

4.Sigmoidal function is the most common when creating an ANN. It accepts not only strict values of 0 or 1, but also infinite set in the interval from 0 to 1. The logistic function can be *an example of such a function*:

$$\varphi(v)=\frac{1}{1+\exp(-av)}, \qquad (9)$$

where a is the slope parameter of sigmoidal function. By adjusting this parameter, it is possible to change the steepness of the function itself. Some situations require the symmetry relative to the origin, respectively, the range of values of the function must be in the interval from -1 to +1. This function is called *Signum* and its example can be *hyperbolic tangent*:

$$\varphi(v)=\tanh(v). \qquad (10)$$

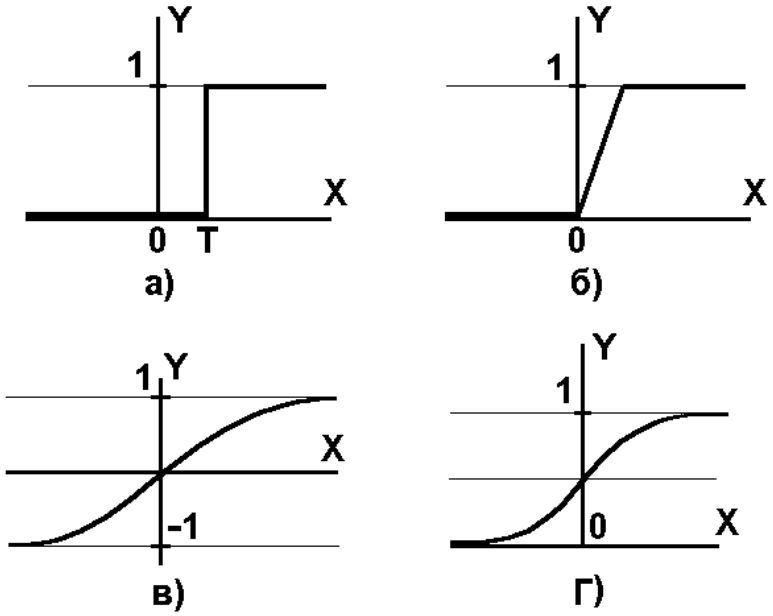

Figure 6 - Types of activation functions: threshold function (a) hysteresis (b); hyperbolic tangent (V); logistic function (g)

The types of functions described above convert the input signal into output by precisely defined values of this signal. However, it may be necessary to use the probabilistic approach and move to the stochastic model of the neuron. In such a model the neuron can be in the state-1 and + 1 and switch with some probability of occurrence of such event. The following formula can be used to describe such a model [6]:

$$\varphi(v)=\begin{cases} +1, & \text{с вероятностью } P(v) \\ -1, & \text{с вероятностью } 1-P(v). \end{cases} \qquad (11)$$

Where$x$ is the state of the neuron; $P(v)$is the probability of activating the neuron. This probability is also described by a sigmoidal function:

$$P(v)=\frac{1}{1+\exp\left(-\frac{v}{T}\right)}, \qquad (12)$$

where T– describes the effect of synaptic noise. If this parameter aspires to zero, the stochastic neuron will accept the threshold form of activation.

2.4 ANN Architectures

In fully-connected networks the propagation of information occurs from neurons of an *input layer* and is passed further to neurons of an *output layer*. In the simplest case such networks have one layer of neurons which is the output-these networks are called *single-layer*. If the network has additional layers that are called *hidden layers*, then the network is called *multilayer* and the more there are hidden layers, the more there are synaptic links and interaction of neurons, thus there are more possibilities for processing big data. One of the first and simple neural networks is *perceptron* – single-layered ANN direct distribution with a threshold activation function. [6]

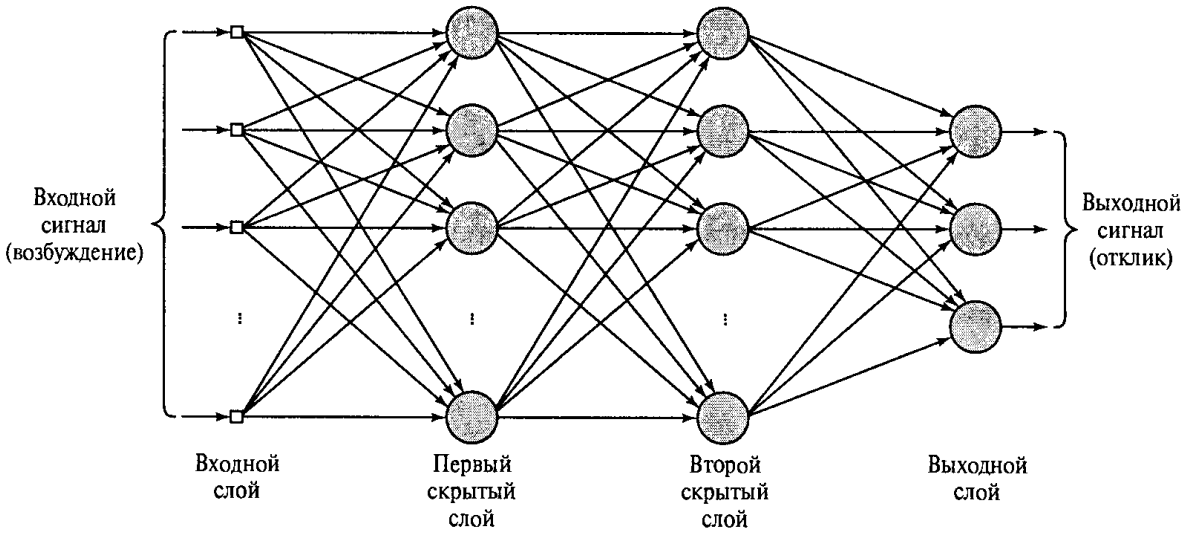

Figure 7 - ANN of direct spread with two hidden layers

In fully-connected networks, the neurons of each layer are connected to all previous neurons that is inconvenient or ineffective when solving certain tasks. Convolutional neural networks-are a special type of neural networks located very close to the structure of the *convolutional layers, subsampling layers* and *fully-connected layers*. This architecture of networks includes three paradigms: *local perception*; *partial weight*; *subsampling*. [6.7]

Local perception implies that the input of one neuron comes all the exits of the previous layer, but only some defined part thereof. Shared weights imply that a small set of weights, called *cores*, is used for most relationships. The core consists of a matrix of small size, which is applied to the input vector or layer of neurons. For

example, if the input vector is an image consisting of pixels, the kernel is applied to the image through a mathematical operation convolution. The essence of this operation consists in elemental multiplication of a fragment of an image on a matrix of a kernel, summation of the received values and record of the result in the similar position of an output image which is called a *map of traits* as convolution kernel will give the degree of similarity of the fragment with a filter or a sign of something in the image. Accordingly, the network layer that performs the convolution operation is called convolutional.

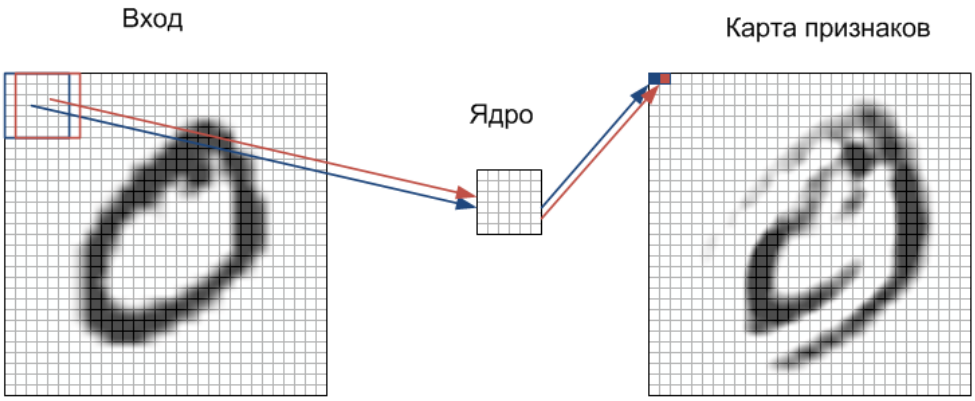

Figure 8 - Applying the core to the image

The essence of subsampling is to reduce the spatial dimension of the image, usually twice. That is, the original image is averaged, providing invariance regarding scale. Respectively. Subsampling layers are engaged in this operation. [7.8]

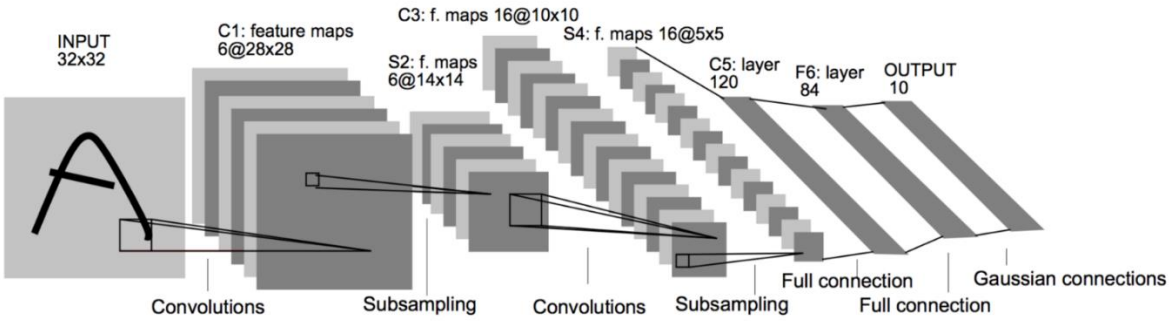

Figure 9 - Structure of the convolutional neural network

Recurrent network is possessing at least one feedback. The concept of feedback is typical for systems in which the output signal affects input signals. Feedbacks exist in the nervous system of almost any animal and allow to amplify

external signals, signals circulating inside the system. Thus, in recurrent networks, neurons can exchange information with each other and this is an important difference from a direct distribution network – feedbacks allow to organize some semblance of memory that allows to store some information about The previous state of the network, and, therefore, allows you to analyze the data sequences for which the order of their values, for example, musical compositions or the state of the stock market is important. One of the first recurrent of ANN was *the Hopfield network*, which practically implements one cell of associative memory. This network consisted of one hidden layer of neurons connected to each other. [6.7]

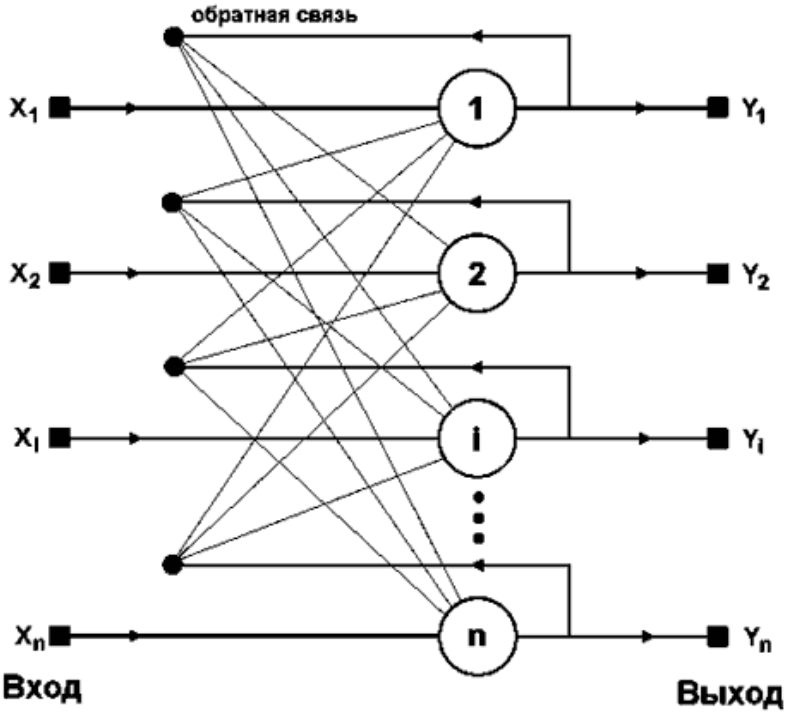

Figure 10 - Hopfield's Recurrent Cetwork

Network of radial basis functions. Such functions form one of the simplest classes and can be used in different networks (like the multilayer and single layer)

2.5 ANN Training

The ability to train on the basis of external data and thereby improve their own performance is one of the most important features of ANN.

Training – the process in which free parameters of a neural network are adjusted by modeling the environment in which the network is embedded. The type of training is determined by the way these parameters are built. [6]

Based on the definition it is possible to distinguish the sequence in ANN training: supply of incentives from the external environment; change of free network parameters; change the network response to subsequent excitation. This sequence is called the *teaching algorithm*, each of which differs in the way of setting the synaptic weights of neurons and the way of communication of the trainee network with the surrounding world. The teaching algorithms are iterative and each iteration is called *an epoch*.

There are two main paradigms of neural network training: learning with and without a teacher. The concept of teacher training is that the teacher prepares for the neural network special pairs of *inputs and reference outputs*. Let us assume that some input values are coming from among the surrounding ones. With knowledge, the teacher can generate the desired response to these input values and transmit the ANN. Network settings will be adjusted on the basis of quietly *signal values and errors*-the difference between the desired signal and current output signal network. The adjustment process is incremental and necessary to simulate the actions of the teacher, which allows to convey the knowledge of the teacher in full. After completing the training, ANN can act independently. [6.7]

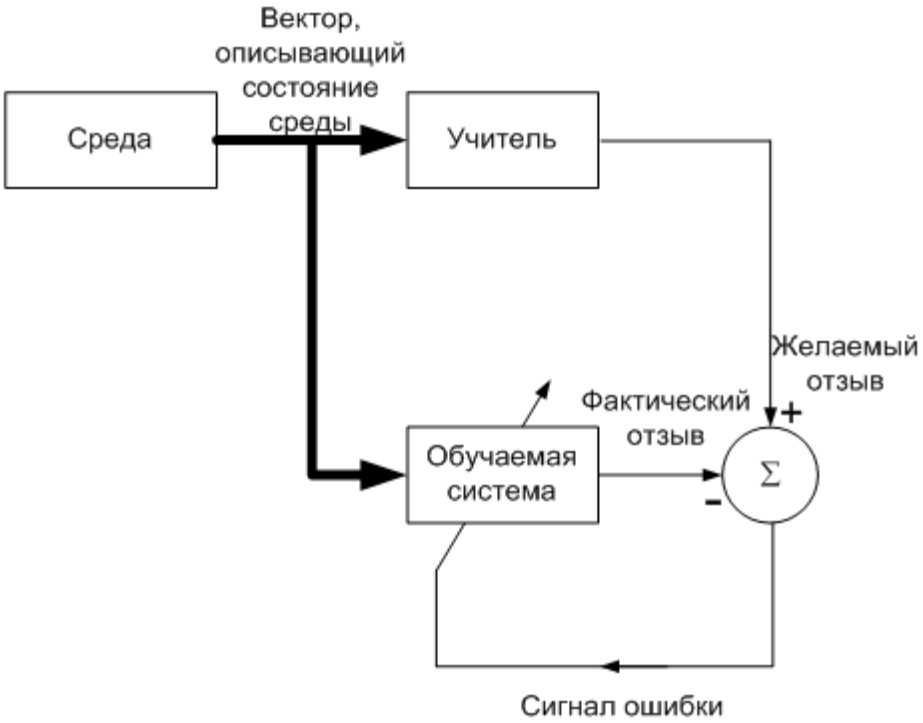

Figure 11 - Teacher Training Diagram

Teaching with reinforcement (neurodynamic programming). In this training, output signals are generated with interaction with the external environment. Figure 6 shows a diagram describing this algorithm. There are two important functions: *evaluation and reinforcement.* The evaluation function determines which result is acceptable in a long period, while the reinforcement function decides what is acceptable at the moment. With this training, the network remembers the correspondence between the situations and the actions necessary to be performed. [6.8]

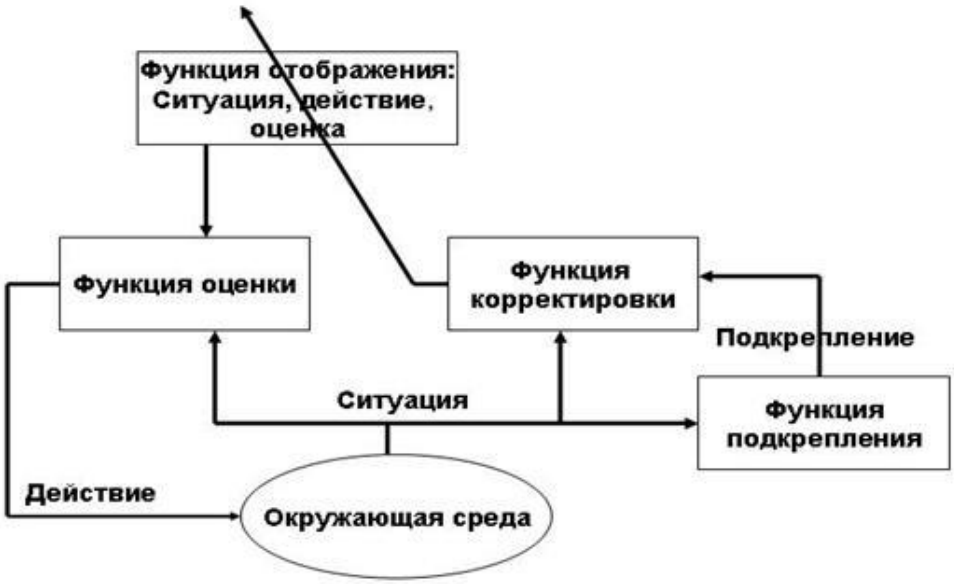

Figure 12 - Teacher Training Diagram

Learning algorithms are divided into deterministic and stochastic:

1. Deterministic-setting options to ANN is iterative and is based on the current state of the network.
2. Stochastic-setting options ANN is conducted randomly, but only those changes that have resulted improvement in a neural network.

Hebb's Method is one of the first methods of learning based on biological studies, suggesting that the strength of connections between neurons is enhanced if both neurons are simultaneously active. Thus, the change of weight of connection

depends only on a pair of neurons-at synchronous excitation communication strengthened, and at asynchronously on the contrary.

At the beginning of training with this algorithm, all weights are assigned some random values. When the input signal is submitted, the processing and output is performed, on the basis of which the weighting factors are changed according to the above-mentioned rule. [6.9]

As competitive method of training output neurons cannot be activated simultaneously, as in the network trained by the Hebb's method, and compete for the right to be activated. Moreover, the synaptic weights are distributed so that the input data result in different reactions of different neurons. Thus, each output neuron in such networks is responsible for some group or class of traits and is a characteristic detector. Readjustment of weights is made only for the winning neuron in the competition. [7.8]

Correction of errors is the model that uses the learning algorithm with the teacher, where the input data is accompanied by the desired output. To adjust the weights, the output data is compared with the desired results. Let us consider one of them:

Back propagation algorithm error got its name due to the fact that the error calculated on each iteration is spread by ANN from the exit to the input in order to reconfigure the synaptic weights. In the course of training the network, when submitting the input vector, the output of the network is compared with the output from the training sample, forming an error$\delta = d - y$, where$d$-the reference output, and-$y$the output of the network. This error is caused not only by the error of the output neurons, but also by all hidden neurons. [9]

Adjustment of synaptic weights is made according to the Widrow – Hoff rule (Delta rule):

$$\Delta w_i = \eta \delta x_i \ , \qquad (13)$$

whereη is the coefficient of speed training;x_i -the values of the input;δ -error.

The new synaptic weight is calculated according to the following formula:

$$w_i(k+1)= w_i(k)+ \Delta w_i \ , \tag{14}$$

where$k$ is the iteration number of training.

For the output error an average squared error is commonly used:

$$E=\frac{1}{2}\sum_j (y_j-d_{jj})^2. \tag{15}$$

Then, in accordance with a gradient descent we get:

$$\Delta w_{ij}= -\eta \frac{\partial E}{\partial w_{ij}} \ , \tag{16}$$

wherew_{ij} is the current weight connection$i$ of neuron$j$.

There will be shown that that is true for linear neurons

$$\frac{\partial E}{\partial w_{ij}}= \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial w_{ij}} \ . \tag{17}$$

Then substituting (17) (16) we will get

$$\Delta w_{ij}= -\eta \frac{\partial E}{\partial w_{ij}}\frac{\partial y_i}{\partial w_{ij}} \ . \tag{18}$$

As a consequence of this and (13) it will be

$$\frac{\partial E}{\partial y_j}= -\delta_j \ . \tag{19}$$

When the output of a neuron is formed by using the activation function ($f(S)$) equation (17) and (19) will look like this:

$$\frac{\partial E}{\partial w_{ij}}= \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}\frac{\partial S_j}{\partial w_{ij}}; \ \delta_j=- \frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j} \ . \tag{20-21}$$

Having differentiated the expression (15) and considered equations (20-21), we receive:

$$\delta_j=\frac{\partial E}{\partial y_j}\frac{\partial y_j}{\partial S_j}=\left(y-d_j\right)f'(S) \ . \tag{22}$$

All the cofactors of the right part of the expression (20) are known and determine the amount of weight correction (23), and taking into account that the synaptic weights the change in the direction of the error function, we get the reverse direction on the sign of the derivative error function (24).

$$\frac{\partial E}{\partial w_{ij}} - (y - d_j) f'(S) x_i;\ \Delta w_{ij} = \delta_j x_i \ . \qquad (23\text{-}24)$$

The resulting errorδ_j is an error of an output neuron. To get the error of a hidden neuron their weight should be adjusted according to the formula (24) on the contribution to the error of the next layer. Thus, the larger the error on the next layer and the more the synaptic weight of the connection linking these two neurons is, the greater will be the error at the output of the hidden neuron. Thus, we get the error of the hidden neurons:

$$\delta_j = f'(S) \sum_k \delta_k w_{kj} \ . \qquad (25)$$

The algorithm of error back propagation uses gradient descent. The idea of this optimization algorithm is to move to the local extrema of the function in the direction of the fastest descending or ascending function. For this purpose, there is a gradient-a vector defining directions of an ascending function:

$$\text{grad}\varphi = \nabla\varphi = \frac{\partial \varphi}{\partial x} i + \frac{\partial \varphi}{\partial y} j + \frac{\partial \varphi}{\partial z} k \ . \qquad (26)$$

In the case of artificial neural networks, the gradient is calculated as the sum of the gradients caused by each element of the training sample. This kind of gradient descent is called *batch*. [7.8]

In contrast to batch gradient descent, there is a *stochastic* gradient descent (*operational*), except that the value of the gradient is approximated by the gradient of the function value, calculated on the same random element tutorial sample.

There is also a third option, which combines a batch and Stochastic methods-*mini-batch*. [9] this method approximates the value of the gradient for *n* randomly selected elements of the training set.

From advantages of this algorithm it is possible to carry out easy realization, possibility to use many functions of losses, and that it can be used for big data. From the drawbacks, this algorithm is mainly used for convex functions, since other functions only local extrema are found. Also, in some situations, a possible divergence when choosing training rate which is too large or the convergence rate

will be too small, though there is the possibility of overfitting. To solve these problems, there is a gradient descent optimizer. Let us consider one of them:

1. Nesterov Accelerated Gradient method optimization-this method is based on the idea of stockpiling impulse, i.e. with the long-term movement in one direction, the speed will be maintained for some time after. To do this, you must store several previous values and calculate the average. Calculating the average value takes too much memory for a large number of occurrences, so the average is used. Multiply these values by a factor of preservationγ. [12]

$$v_t = \gamma v_{t-1} + (1 - \gamma)x \, . \qquad (27)$$

2. Adagrad is an adaptive gradient. It is a family of algorithms, the main idea of which is to keep some rare signs to protect them from noise. This creates a certain value for example the sum of the squares of the upgrades or modules for a parameter of an artificial neural network. On the basis of this value the updates of elements are regulated-frequently met ones are updated less often, freeing up some place for seldom one, thus the adaptive speed of training or attenuation of speed of training is obtained. [12]

3. Adam-adaptive moment estimation. This optimization method combines the accumulation of the pulse, considered in the Nesterov method, as well as storing the frequency of the gradient change, similar to the Adagrad. Thus, this method has the advantages of both methods considered. [12]

1.4 3**. Review and comparison of libraries.**

TensorFlow is an open program library for machine training developed by Google to solve the problems of building and training a neural network to automatically locate and classify images, achieving the quality of human perception. It is applied to both research and development of Google's own products. The basic API for working with the library is implemented for Python, and there are also implementations for C++, Haskell, Java and Go.

Torch is a library for scientific computing with broad support for machine training algorithms. Developed by the Idiap Research Institute, New York University and NEC Laboratories America since 2000, it is distributed under the BSD license.

The library is implemented in Lua language using C and CUDA. The fast Lua scripting language combined with SSE, OpenMP, CUDA Technologies allow Torch to show good speed in comparison with other libraries. Currently, Linux, FreeBSD, Mac OS X operating systems are supported. The main modules also work on Windows.

Keras is an open neural library written in Python. It is built on frameworks, Deeplearning4j, TensorFlow and Theano. [1] [2] focuses on operational work with networks of deep training, while designed to be compact, modular and extensible. It was created as part of research efforts of ONEIROS project (Open-ended Neuro-Electronic Intelligent Robot Operating System), [3] and its main sponsor and supporting is François Cholle (FR. François Chollet), a Google engineer.

For comparison, the libraries used a dataset CIFAR-10, which contains 10 classes of images[13]. The dataset contains 60000 color images 32 by 32 pixels, where each class has 6000 images respectively. Classes included in CIFAR-10: planes, cars, trucks, birds, cats, deer, dogs, frogs, horses and ships (Fig. 13). As architecture for testing used by VGG 16 developed in Oxford in the year 2014 (fig. 14).

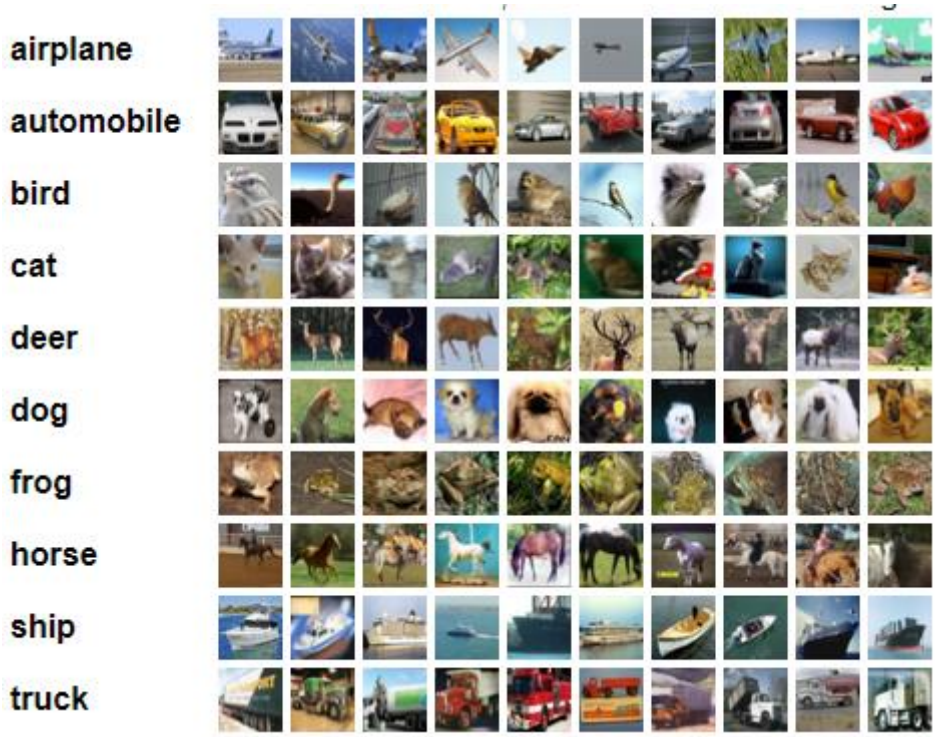

Figure 13 - Examples of the images in the CIFAR 10 dataset

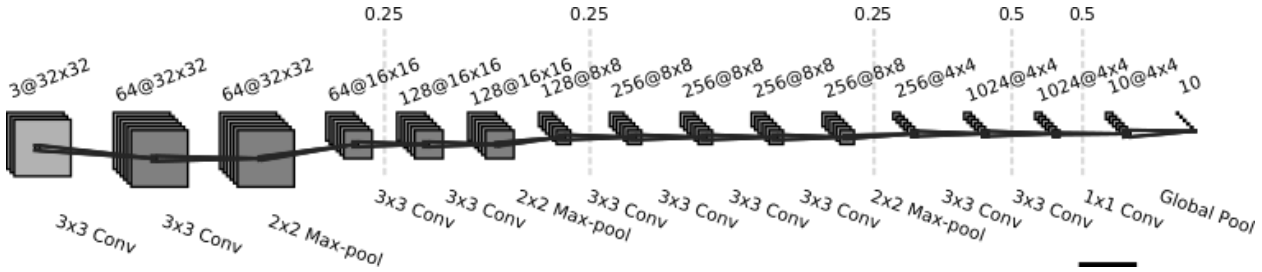

Figure 14 - Network architecture for testing libraries

According to the results of the training time, the Pytorch library was the fastest, 168 seconds, and the library TensorFlow-173 seconds and the longest time shows the library Keras-252 seconds (Fig. 15). According to the accuracy of the library Ppytorch and TensorFlow show the same results – 78%, Keras one percent less – 77% (Fig. 16) [14].

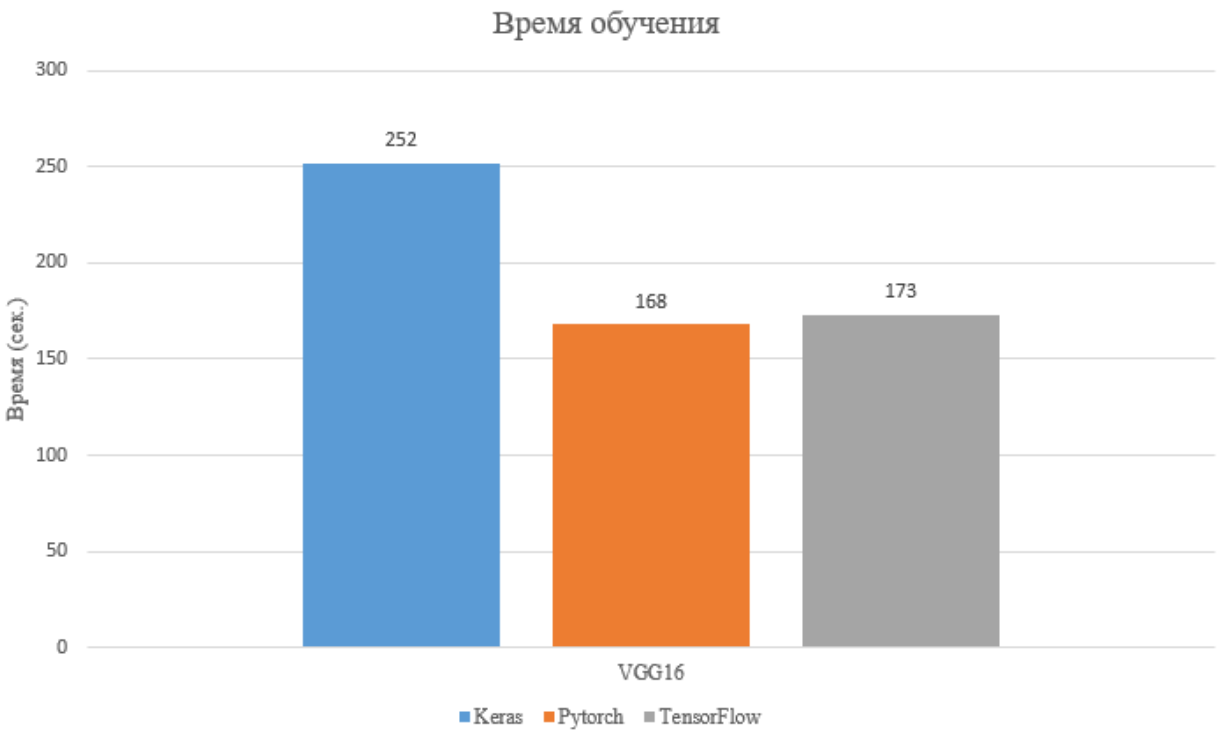

Figure 15 - Comparison of training libraries

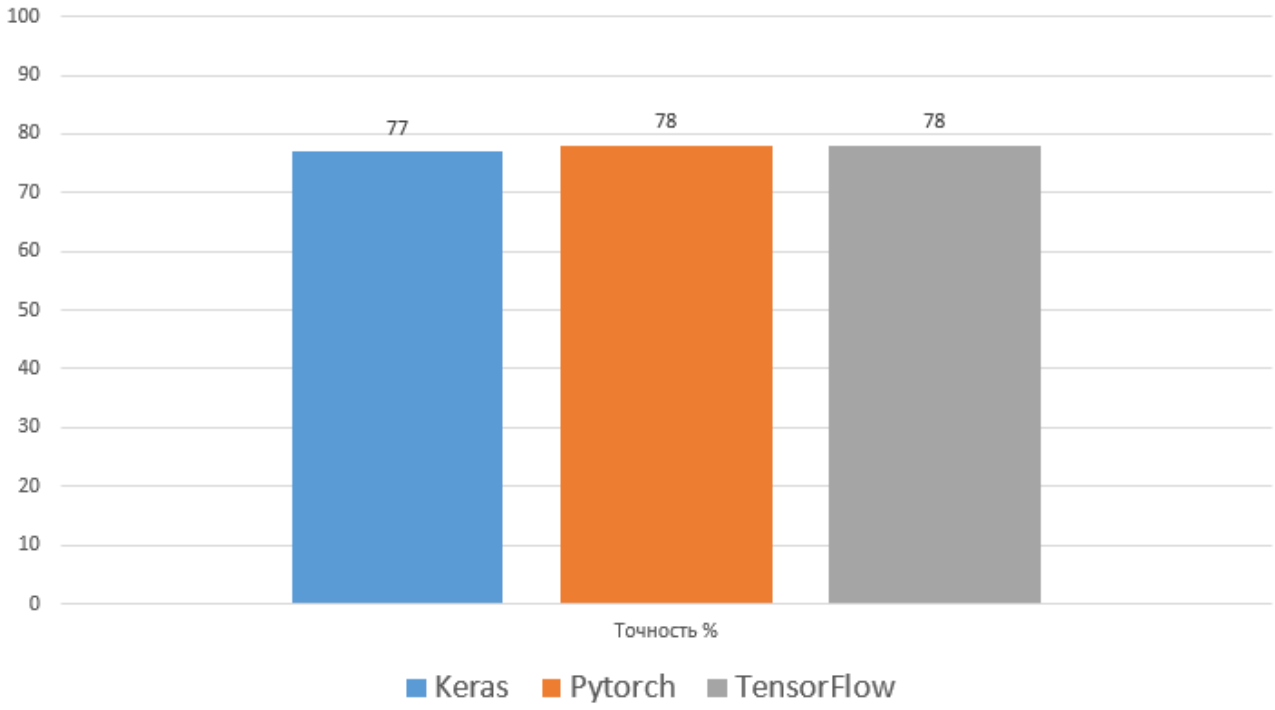

Figure 16 - Accuracy of training results

Table 1-Summary of library features.

Library	Keras	PyTorch	TensorFlow
Authors	François Chollet	Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan	Google Brainteam
Distributed under the license	MIT license	BSD license	Apache 2.0
Open source	+	+	+
Supported OS	Linux, macOS, Windows	Linux, macOS, Windows	Linux, macOS, Windows,Android
Written in language	Python	Python, C, CUDA	C++, Python, CUDA
Language interface	Python, R	Python	Python (Keras), C/C++, Java, Go, R, Julia
CUDA Support	+	+	+
Automatic differentiation	+	+	+
There are pre-trained models	+	+	+
Supports recurrent neural networks	+	+	+
Supports convolutional neural networks	+	+	+
Multi-threaded calculations	+	+	+