Федеральное государственное бюджетное научное учреждение
«Федеральный исследовательский центр Институт прикладной физики им. А.В. Гапонова-Грехова Российской академии наук»

На правах рукописи

Ковальчук Андрей Викторович

РАЗРАБОТКА НЕЙРОМОРФНЫХ АЛГОРИТМОВ ПРИНЯТИЯ РЕШЕНИЙ В ЗАДАЧАХ РАСПОЗНАВАНИЯ ОБЪЕКТОВ НА ИЗОБРАЖЕНИИ

5.12.4 - Когнитивное моделирование (физико-математические науки)

Диссертация на соискание ученой степени
кандидата физико-математических наук

Научный руководитель:
доктор физико-математических наук
Яхно Владимир Григорьевич

Нижний Новгород – 2026

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Актуальность темы диссертации

Современные тенденции в области искусственного интеллекта и распознавания образов всё больше ориентируются на биологически вдохновленные подходы. Биоморфное проектирование интеллектуальных систем предполагает использование принципов организации и работы живых систем при создании алгоритмов и архитектур искусственного интеллекта (ИИ). Актуальность используемого подхода обусловлена несколькими обстоятельствами.

Во-первых, традиционные системы распознавания достигают своего предела эффективности на сложных и изменчивых данных [37, 171]. В тех случаях, когда условия выходят за рамки предусмотренных (новые виды шума, ранее невиданные классы образов, изменение свойств сенсоров), классические алгоритмы часто оказываются негибкими [77, 95]. Биоморфный же подход, напротив, предполагает адаптивность и устойчивость, свойственные биологическим системам [101]. Мозг животных умеет удивительно хорошо справляться с неполными или искаженными данными, дорисовывать недостающую информацию, переносить знания с одного типа задач на другой. Имитация этих свойств – через динамические нейроподобные ячейки, через автоподстройку алгоритмов – открывает путь к новым, более живучим системам распознавания.

Во-вторых, исследуемый нейроморфный подход вносит вклад в концепцию *интеллектуальных систем нового поколения*, где жёстко детерминистские алгоритмы сменяются на самообучающиеся и саморегулирующиеся [91]. Если ранее основной упор делался на увеличение глубины нейронных сетей и объёма данных для офлайн-обучения, то теперь все более ценными становятся свойства непрерывной адаптации, пояснимости решений, способности к автономному улучшению без внешнего вмешательства [153, 155]. Причины этого факта кроятся в достижении предела

возможностей извлечения информации с помощью статистических подходов и моделей. Биоморфная адаптивная система по своей природе не статична, она самонастраивается в процессе функционирования, и при этом её архитектура позволяет интерпретировать, какие именно модули и почему перестраиваются.

Наконец, с инженерной точки зрения, биоморфный подход предоставляет новые возможности оптимизации. Заимствуя принципы из нейробиологии (параллелизм обработки, событийное управление, избыточность и устойчивость к сбоям и т.д. [42, 75]), можно создавать системы распознавания, более экономичные и надежные, например, за счет снижения количества вычислений с помощью адаптивного управления алгоритмами.

Подводя итог, отметим, что актуальность разрабатываемого подхода определяется растущей потребностью в гибких биометрических системах, способных работать в условиях реального мира, с его непредсказуемостью и разнообразием сигналов.

Степень разработанности проблемы

Первые количественные модели биоэлектрической активности появились в работах Ходжкина и Хаксли, открывших путь к описанию распространения импульсов в возбудимых средах [97]. Их идеи были развиты в популяционных моделях Уилсона–Коуэна и в теориях латерального торможения Амари, объясняющих формирование устойчивых пространственных паттернов в коре мозга [25, 36, 184]. Позднее Чуа и Янг предложили клеточные нейронные сети как аппаратную дискретизацию непрерывных нейронных полей [54], что дало импульс широкому спектру приложений — от импульсных PCNN-фильтров до обработки изображений и распознавания образов [103, 149]. Тем не менее большинство работ сосредоточено либо на теоретической динамике, либо на низкоуровневой фильтрации. Возможности и роль динамических режимов, встроенных в

контур классификации в комплексных системах, остается еще слабо изученной.

Традиционные методы качества (FAR/FRR и др.) фиксируют итоговую ошибку, но не позволяют судить о пригодности конкретного примера для данного алгоритма. В ответ на это в работах [28] предлагается цикл *кодирование – восстановление* с анализом невязки как внутреннего сигнала качества. Идея перекликается с автоэнкодерами [96] и разрежённым кодированием [140], а также с локальными байесовскими схемами Гельмгольца и мета-обучением [63]. Однако в существующих решениях отсутствуют формальные критерии для автоматического выбора алгоритма предобработки, что и закладывается в настоящей работе.

Механизмы коллективного распознавания и алгоритмы построения экспертных комитетов активно исследовались в работах Растригина [24], Мазурова [21], а также в более поздних исследованиях [102, 143, 174] и [52, 133, 148]. Принятие окончательного решения здесь осуществляется либо выбором наиболее подходящего классификатора с помощью заранее заданной меры компетенции, либо с помощью обучаемых правил аггрегирования решений коллектива.

Рост применения ИИ в критических областях обострил проблему непредсказуемых сбоев, связанных с нарушением гипотезы независимости и сдвигом распределений выборок [59, 93]. Полное переобучение дорого и может вносить новые ошибки, поэтому развиваются *корректоры* — внешние модули, обучающиеся на единичных ошибках и локально исправляющие вывод базовой модели [80]. Согласно теоремам стохастического разделения, в сверхвысоких размерностях такие ошибки отделимы линейными границами (эффект «благословения размерности») [82], а потому корректор можно строить по one/few-shot-схеме.

Цель и задачи исследования

Цель работы — разработка и исследование элементов нейроморфной системы, реализующих адаптивный выбор алгоритмов обработки и коррекцию решений на основе внутренней динамики и тернарного коллективного распознавания.

Для достижения поставленной цели решались следующие задачи:

1. Разработать для математических моделей нейронных сетей рекуррентные алгоритмы, обеспечивающие выделение дополнительных информативных признаков для задач классификации.

2. Разработать критерий «невязки» в процессе “кодирования-восстановления” как формализованную метрику оценки данных и алгоритмов и интегрировать его в модуль предобработки изображений отпечатков пальцев.

3. Разработать алгоритмы построения тернарного классификатора и протестировать их реализацию на экспериментальных данных.

4. Разработать методы коррекции решений каскадных детекторов объектов на изображениях, использующих малое количество примеров для обучения.

5. Продемонстрировать на реальных биометрических базах данных, что предложенные процедуры и алгоритмы обучения повышают точность распознавания целевых объектов на изображении и обеспечивают возможность обучения на малых количествах примеров.

Объект и предмет исследования

Объектом исследования являются нейроподобные системы для распознавания визуальных образов и принятия решений.

Предмет исследования — методы оценки качества данных, динамический выбор алгоритма обработки и корректировка решений в нейроморфных системах коллективного распознавания.

Научная новизна

1. Показано, что расширение признакового пространства с помощью динамики однородной нейроноподобной среды позволяет повысить качество алгоритмов принятия решений в задаче классификации лиц.

2. Впервые предложен критерий оценки качества изображения на основе динамики невязки в процессе «кодирование-восстановление», а также показано, что применение адаптивного выбора алгоритма бинаризации с помощью невязки повышает качество распознавания изображений отпечатков пальцев.

3. Разработан алгоритм построения тернарного классификатора (эксперта) *OnSVM*, обеспечивающий корректное распознавание в условиях частичной информативности и неопределенности, а также его дообучение.

4. Реализован алгоритм корректировки решений каскадного детектора объектов на изображениях, использующий внутренние состояния и признаки для повышения его точности.

5. Разработана методика тестирования алгоритмов корректировки решений каскадных детекторов на размеченных базах данных изображений.

Теоретическая и практическая значимость

В работе формализованы параметры латерально-ингибиторных функций связи типа латеральное торможение, введено понятие динамики невязки как индикатора качества сигнала, разработана теория тернарного классификатора, разработана модификация алгоритма бустинга M-Adaboost, предложена новая постановка потоковой машины опорных векторов (OnSVM) с максимизацией зазора, а также предложен подход к коррекции ошибок детекторов на основе динамических сверток.

Предложенные алгоритмы применимы при разработке биометрических систем распознавания лиц и отпечатков пальцев, детекторов объектов на видео, систем мониторинга и диагностики, а также для ИИ-модулей, работающих в условиях ограниченных данных и данных со смещением

распределений. Часть методов внедрена в НИОКТР “PhysioAssist”, НИОКТР “БСИИРОКО” и зарегистрирована как программное обеспечение.

Методы и методология

В диссертационной работе использован комплекс методов когнитивного, математического и численного моделирования, ориентированный на построение и исследование нейроморфных алгоритмов принятия решений в задачах распознавания изображений.

Основные положения, выносимые на защиту

1. Расширение признакового пространства с помощью динамики однородной нейроноподобной среды позволяет повысить точность алгоритмов классификации лиц. В работе показано, что использование 40 параллельных нейроноподобных сред с функцией связи типа латеральное торможение (5 масштабов × 8 ориентаций) и учёт временной эволюции активности среды приводят к существенному снижению ошибки верификации. Ошибка базового алгоритма снижается с EER ~13% до ~9.5% при использовании 4-х дискретных шагов нейроноподобной среды, что соответствует итоговой точности распознавания ~90–91% при четырёх эталонных изображениях на класс.

2. На примере задачи классификации отпечатков пальцев показано, что по динамике невязки в циклах «кодирование–восстановление» можно строить адаптивные алгоритмы предобработки, автоматически выбирающие между активным нейроноподобным и пассивным линейно-пороговым фильтром. Эксперименты показали, что активный фильтр показывает ошибку EER порядка 11 %, пассивный — около 13 %, тогда как адаптивный выбор алгоритма по невязке снижает ошибку до 9–10 %, что соответствует уменьшению доли неверных классификаций на 10–15 % по сравнению с лучшим одиночным фильтром.

3. Введён и математически формализован тернарный классификатор с областью компетенции, позволяющий локально ограничивать область уверенного распознавания и снижать ошибки в каскадных детекторах при добавлении новых объектов или атрибутов без изменения основной модели. В задаче классификации атрибутов лиц, предложенные алгоритмы построения тернарных экспертов M-Adaboost и OnSVM показывают заметный рост качества: тернарный режим снижает ошибку на 5–14% по сравнению с бинарным. При этом доля объектов, по которым эксперт возвращает ответ «не знаю», остаётся ограниченной, позволяя безопасно передавать сложные случаи в последующие каскады детекторов, что обеспечивает расширение области компетенции системы без ухудшения базовой точности.

4. Решения каскадных детекторов, использующих модифицированные признаки Цензуса, могут быть скорректированы с помощью линейных правил по малому числу примеров ошибок, существенно меньшему, чем число верных предсказаний. На базах «Цифры» и «Идентификаторы» показано, что обучение FN-корректоров по выборке пропущенных срабатываний (FN) позволяет увеличить полноту детектирования цифр с 0.94 до 0.98 при практически неизменной точности (0.92 → 0.91). Для задачи детектирования идентификаторов корректоры по небольшому числу ложных срабатываний (FP) повышают точность с 0.36 до 0.64 при сохранении высокой полноты (Recall ≈ 0.98).

Обоснованность и достоверность

Достоверность результатов подтверждена многократными экспериментами на широко известных биометрических базах данных, таких как *MNIST* [64], *OptDigits* [108], *PenDigits* [35], *FERET* [104], *FG-NET+* [145], а также на специально собранных датасетах изображений лиц, изображений отпечатков пальцев, фрагментов фасадов домов и базы изображений лиц различной этнической принадлежности. Для валидации разработанных алгоритмов применены методы перекрестной проверки, многократной

повторной выборки, сравнение с эталонными реализациями *LIBSVM* [53] *и AdaBoost*. В диссертации представлены графики сходимости, карты невязок, ROC-кривые и сравнения ошибок, подтверждающие корректность моделей.

Публикации автора по теме диссертации

Всего по теме диссертации опубликовано 10 статей, из них: 8 статей в научных изданиях, индексируемых в базах данных Web of Science и Scopus, 8 статей в научных изданиях, индексируемых в базе RSCI.

Апробация результатов

Основные результаты диссертационной работы были представлены на международных и российских научных конференциях: VI всероссийской научно-технической конференции “Нейроинформатика – 2004” (Москва, 2004), VII всероссийской научно-техническая конференции “Нейроинформатика – 2005” (Москва, 2005), 13-й всероссийской конференции “Математические методы распознавания образов”, (Зеленогорск, 2007), X всероссийской научно-технической конференции “Нейроинформатика – 2008” (Москва, 2008), XI всероссийской научно-технической конференции “Нейроинформатика – 2009” (Москва, 2009), XV международной научно-технической конференции «Информационные системы и технологии» (Нижний Новгород, 2009), Всероссийской конференции "Нелинейная динамика в когнитивных исследованиях" (Нижний Новгород, 2009), XII всероссийской научно-технической конференции “Нейроинформатика – 2010” (Москва, 2010), 13-й всероссийской научно-технической конференции “Нейроинформатика – 2011” (Москва, 2011), 3-всероссийской конференции «Нелинейная динамика в когнитивных исследованиях – 2013» (Нижний Новгород, 2013), 15-й всероссийской научно-технической конференции “Нейроинформатика – 2013” (Москва, 2013), 19-й международной научно-технической конференции “Нейроинформатика – 2017” (Москва, 2017), 4-й международной конференции “Нейротехнологии и Нейроинтерфейсы”

(Калининград, 2022), 25-й международной научно-технической конференции “Нейроинформатика – 2023” (Москва, 2023).

Алгоритмы и методы, описание которых приведено в главах данной работы, использованы в следующих НИОКТР, грантах и РИД:

НИОКТР “Автоматическая интеллектуальная система оценки биомеханики верхних и нижних конечностей”. П2030/2024-СП4 тема№9. Рег. номер 124091900013-6 (Россия, 2024).

Свидетельство о государственной регистрации программ ЭВМ № 2025610744 “PhysioAssist - Автоматическая интеллектуальная система оценки биомеханики верхних и нижних конечностей” (Россия, 2025).

НИОКТР “Биоморфная система искусственного интеллекта для распознавания образов с коррекцией ошибок (БСИИРОКО)” КодИИ-117137 (Россия, 2022-2025).

НИОКТР “Intelektinė paslaugų teikimo procesų valdymo sistema Magnetca Golf Video Capturing, skirta kompleksiniam golfo paslaugų valdymui” (“Интеллектуальная система видеозахвата ‘Magnetca Golf Video Capturing’ для комплексного управления гольф-сервисами”) 01.2.1-LVPA-T-848-01-0016 (Литва, 2019-2022).

CRDF Грант «Разработка методов обработки и принятия решений по биометрическим данным» (“Development of Application of Pattern Recognition Technology”) RMO-10214 BNL (США, 2000–2009)

Структура и объем диссертации

Диссертационная работа состоит из введения; основной части, состоящей из четырех глав, в которых изложено содержание диссертации; выводов; списка литературы. Общий объем работы составляет 183 страницы машинописного текста, 51 рисунок, 18 таблиц, 42 формулы, 2 приложения и 190 цитируемых работ.

Благодарности

Автор выражает глубокую признательность и сердечно благодарит **Полевую С. А.** за ценные дискуссии, поддержку и вдохновение; научного руководителя **Яхно В. Г.** за стратегическое руководство; светлой памяти **Тельных А. А.** — за помощь в реализации алгоритмов и неизменный энтузиазм; **Шемагиной О. В.**, **Нуйдель И. И.** и **Парину С. Б.**— за помощь, идеи и советы на всех этапах исследования.

ГЛАВА 1. ИСПОЛЬЗОВАНИЕ ДИНАМИКИ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЙ В НЕЙРОНОПОДОБНЫХ СРЕДАХ В ЗАДАЧАХ КЛАССИФИКАЦИИ

Моделирование нейроноподобных сред – распределённых активных систем, состоящих из множества взаимодействующих элементов, – направление в исследовании процессов обработки информации в нервной системе, которое концентрируется на макроуровне обработки, опуская физические детали передачи информации. Ещё в середине XX века появились первые модельные описания: так, классическая работа Ходжкина и Хаксли [97] дала количественное описание генерации нервного импульса с помощью системы дифференциальных уравнений, успешно описав механизм возникновения потенциала действия в мембране нейрона. В 1970-е годы были предложены модели, описывающие динамику целых популяций нейронов. В частности, Уилсон и Коуэн [184] разработали математическую модель активности локальных групп возбуждающих и тормозных нейронов, заложив основу для понимания устойчивости нейронных сетей и возникновения колебательных режимов в коре головного мозга. Последующие исследования учли пространственные эффекты: Сбитнев с соавторами рассмотрел распространение спайков в статистических нейронных ансамблях, введя представление о фазовых переходах в нейронной среде [25], а Амари [36] исследовал формирование паттернов активности в нейронных полях с латеральным торможением, показав, как локальное конкурентное взаимодействие приводит к возникновению устойчивых пространственных структур в нервной ткани. В отечественной научной школе одним из важных шагов стала работа Кудряшова и Яхно, предложивших модель распределённой нейронной среды с нарастающим числом связей между возбудимыми элементами. Эта модель объяснила появление волновых процессов – фронтов возбуждения, автоколебаний, диссипативных структур – аналогичных тем, что наблюдаются в коре головного мозга и гиппокампе [20].

Развитие вычислительных технологий и нейрокибернетики в 1980-х привело к появлению аппаратных и алгоритмических реализаций нейроноподобных сред. Например, Чуа и Янг предложили концепцию клеточных нейронных сетей (*Cellular Neural Networks*), в которых непрерывно-динамические уравнения нейронного поля реализуются на дискретной матрице взаимосвязанных элементарных процессоров [54]. Такая клеточная нейронная сеть представлялась как программируемый клеточный автомат, способный к параллельной обработке изображения в режиме реального времени. Модели Чуа и Янга и их последователей показали, что распределённые нейроподобные системы могут эффективно выполнять задачи фильтрации сигналов и распознавания образов за счёт локальных взаимодействий между узлами сети. Фактически, уравнения, описывающие классическую однородную нейроноподобную среду с возбуждающими и тормозными связями, можно свести к одной нелинейной интегро-дифференциальной модели [36]. Её дискретный аналог – разностная схема – реализует эволюцию во времени на сетке ячеек и тем самым формирует основу для нейроморфных алгоритмов обработки изображений.

Особое развитие получили так называемые резервуарные вычисления, являющиеся обобщением идей распределённой обработки в нейронных системах [169]. Эти модели включают в себя динамическую, слабо обучаемую нелинейную среду (резервуар), на которую подаётся внешний сигнал, в то время как обучение производится только на выходных связях. Важной особенностью резервуарных моделей является способность превращать временные зависимости во входных данных в различимые пространственные представления, пригодные для последующей классификации, например с помощью моделей echo state networks (ESN) и liquid state machines (LSM) [130]. Резервуарные вычисления, как и клеточные нейронные сети Чуа и Янга, используют идею активной динамической среды для фильтрации и преобразования информации, но при этом подчёркивают временную эволюцию как ключевой механизм обработки. Это делает их особенно

эффективными для задач, связанных с анализом временных сигналов, в том числе в системах речевого распознавания, нейроинтерфейсах и адаптивных сенсорных сетях. Таким образом, исторически были заложены основы нейроподобных методов обработки информации, и к началу XXI века появилось множество вариантов моделей распределенной активности нейронов.

Важно отметить, что идеи, возникшие более полувека назад, сохраняют актуальность и поныне. Современные исследования продолжают развивать нейроподобные алгоритмы и применять их на практике. Например, активно исследуются импульсные нейронные сети, имитирующие вспышечную активность коры [65, 103], и показывающие высокую эффективность в анализе образов. В последние годы были созданы десятки модификаций импульсных нейросетей (PCNN) – моделей, изначально предложенных Джонсоном с соавторами в 1994 году для обработки изображений [103, 149]. Как отмечает Лю с соавторами в обзорной работе [128], модели PCNN и их производные успешно применяются для решения самых разных задач компьютерного зрения: распознавания образов, выделения признаков, сегментации изображений, подавления шума, шифрования изображений и многого другого [18]. Важное преимущество таких нейроподобных фильтров – отсутствие необходимости в долгом обучении на больших выборках: они сразу готовы выделять существенные черты благодаря заложенным принципам, имитирующим работу биологических нейронных сетей. Кроме того, благодаря развитию элементной базы, стали появляться аппаратные реализации нейроноподобных систем: к примеру, современные электронные компоненты (мемристоры и резонансно-туннельные диоды) используются для построения аналоговых и смешанных схем клеточных нейронных сетей, позволяющих выполнять аппаратную параллельную обработку изображений [66]. Это подтверждает, что нейроподобная фильтрация – подход, вдохновленный нейрофизиологией, – продолжает оставаться востребованным и перспективным методом обработки данных даже в эпоху глубокого обучения.

Данная глава посвящена применению нейроноподобной динамической фильтрации в задаче распознавания образов, в частности – классификации изображений лиц. В ней рассмотрены примеры использования как однородных, так и неоднородных нейроноподобных сред для фильтрации изображений, а также описана экспериментальная система распознавания, основанная на нейроподобных признаках. Показано, что динамика распределённой нейронной системы может использоваться для преобразования изображений с целью выделения информативных признаков и последующей классификации [163]. Ниже излагаются конкретные примеры и результаты, подтверждающие актуальность нейроподобных методов фильтрации и распознавания образов.

1.1 Динамические уравнения нейроноподобной среды

Балансные уравнения распределённой нейроноподобной среды с возбуждающими и тормозными элементами записываются следующим образом [20, 22, 32]:

$$\begin{aligned}\tau_u\frac{\partial u}{\partial t}&=-u+S_1F_1\left[-T_1+\alpha_1\int_\Omega \Phi_{uu}(x-r)\cdot u(r,t)dr-\beta_1\int_\Omega \Phi_{uv}(x-r)\cdot v(r,t)dr+u_{ex}\right]\\ \tau_v\frac{\partial v}{\partial t}&=-v+S_2F_2\left[-T_2+\alpha_2\int_\Omega \Phi_{vu}(x-r)\cdot u(r,t)dr-\beta_2\int_\Omega \Phi_{vv}(x-r)\cdot v(r,t)dr+v_{ex}\right]\end{aligned} \quad (1.1)$$

, где:

- $u(x,t)$, $v(x,t)$ — усреднённая активность возбуждающих и тормозных нейронов по пространственной координате x соответственно, отражающая среднюю частоту нервных импульсов;
- τ_u, τ_v — характеристические времена, в течение которых активность нейронов сохраняется;
- $\Phi_{uu}(x)$, $\Phi_{uv}(x)$, $\Phi_{vu}(x)$, $\Phi_{vv}(x)$ — функции пространственной связи, характеризующие влияние возбуждающих и тормозных нейронов друг на друга и между собой;
- $F_1(u)$, $F_2(v)$ — нелинейные функции активации, описывающие процесс генерации импульсов;

- $u_{ex}(x,t), v_{ex}(x,t)$ — внешние стимулы, поступающие на возбуждающие и тормозные элементы;
- $S_1, S_2, T_{10}, T_{20}, \alpha_1, \alpha_2, \beta_1, \beta_2$ — параметры характеризующие плотность возбуждающих и тормозных нейронов в популяциях, пороги возбуждения, плотности расположения возбуждающих и тормозных связей.

На основе данной модели были получены следующие результаты [20, 29, 31]:

- Определены ключевые параметры, влияющие на тип динамических режимов активности нейронной сети.
- Выведена формула для расчета скорости распространения областей повышенной активности.
- Установлены условия для возникновения различных волновых режимов (фронты возбуждения, импульсная активность, источники волн, диссипативные структуры, автоколебания).
- Проведен анализ устойчивости автоколебаний к пространственным возмущениям, показавший их устойчивость.

При условии, что влияние торможения на себя минимально и описывается линейно, а также характеристическое время изменения активности торможения достаточно большое ($\tau_v \gg \tau_u$) система (1.1) упрощается до однокомпонентной нейроноподобной модели [139]:

$$\tau_u \frac{\partial u}{\partial t} = -u + S_1 F_1\left[-T_1 + \alpha_1 \int_\Omega W(x, x-r) \cdot u(r,t) dr + u_{ex}\right] \qquad (1.2)$$

, где $W(x,x')$ — обобщённая функция пространственной связи.

Дискретный аналог однокомпонентной модели (1.2) имеет вид:

$$u^{n+1} = u^n \left(1 - \frac{\Delta t}{\tau_u}\right) + \frac{\Delta t}{\tau_u} \cdot S_1 \cdot F\left(-T_1 + \sum_{x' \in \Omega} W(x,x') u^n(x') + I^n(x)\right) \quad (1.3)$$

, где $\nu^n(x)$ — активность элемента сети в точке x на шаге n;

Δt — шаг дискретизации, позволяющий численно интегрировать процесс во времени;

$I^n(x)$ – входной сигнал.

По виду пространственной связи $W(x, x')$ однокомпонентную нейроноподобную модель (1.2) и ее дискретную версию (1.3) можно разделить на два типа:

- **Однородная -** имеет одинаковую функцию связи $W(x')$ для любой координаты x.
- **Неоднородная -** характеризуется тем, что функция связи $W(x, x')$ варьируется по пространству, позволяя решать более сложные задачи.

Однородная нейроноподобная среда подразумевает, что все нейроноподобные элементы имеют одинаковые свойства и связаны друг с другом по единой схеме. Такой подход удобен для теоретического анализа и позволяет получать универсальные решения, например, распространяющиеся волны или автоколебания, одинаково возникающие в любой части среды. Неоднородная нейроноподобная среда способна сочетать несколько видов рецептивных полей или фильтрующих операций в рамках одной модели, что хорошо согласуется с экспериментами [100, 134]. Это позволяет одновременно извлекать из изображения различные признаки – например, контуры, текстурные элементы, ключевые точки – оптимизируя систему под конкретную задачу распознавания.

1.2 Виды функций пространственной связи

В классической обработке изображений известно множество фильтров, имитирующих свойства рецептивных полей нейронов зрительной системы. Приведём несколько основных примеров таких фильтров (оконных функций), широко применяемых для выделения признаков:

1.2.1 Фильтры Габора

Фильтры Габора – функции, представляющие собой гармонические колебания (см. Рис. 1.1), взвешенные двумерной гауссовой функцией [61]. Каждый такой фильтр настроен на определённую пространственную частоту и ориентацию, благодаря чему отклик фильтра позволяет выделять на изображении локальные структуры заданной масштабно-частотной характеристики. Общая форма для симметричной и асимметричной пары может быть записана как:

$$W_{сим}^{G}=cos(k_x x+k_y y)e^{-\left\{\frac{x^2+y^2}{2\sigma^2}\right\}}$$
$$W_{асим}^{G}=sin(k_0 x+k_1 y)e^{-\left\{\frac{x^2+y^2}{2\sigma^2}\right\}} \quad (1.4)$$

, где (k_x, k_y) – определяют пространственную частоту;

σ - определяет пространственный масштаб рецептивного поля.

Фильтры Габора (1.4) хорошо описывают рецептивные поля нейронов первичной зрительной коры и применяются для распознавания текстур, отпечатков пальцев, детекции объектов определённой формы [62].

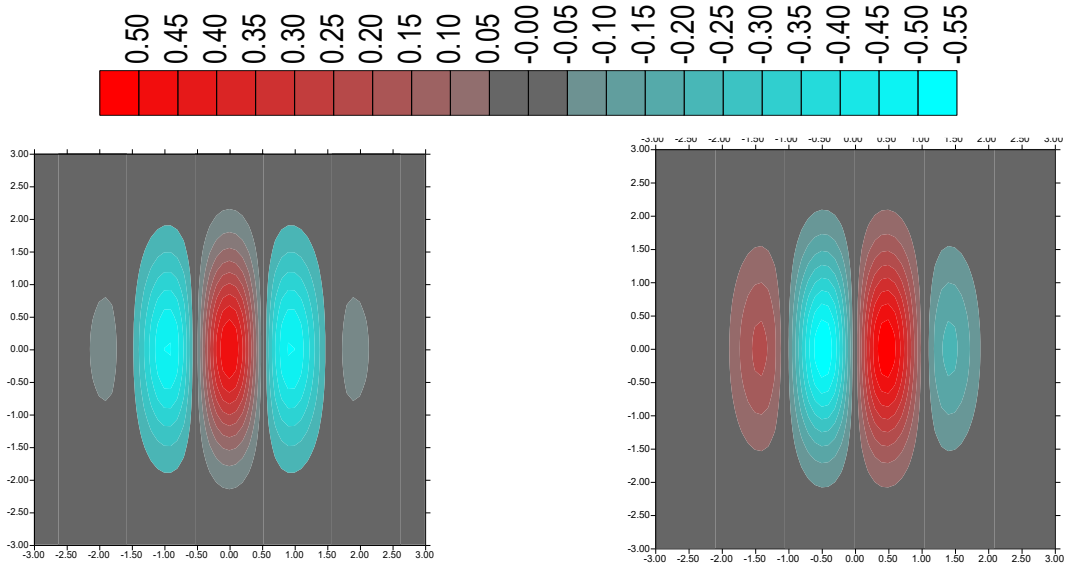


Рисунок 1.1 — Пример симметричной функции Габора (слева) и асимметричной (справа)

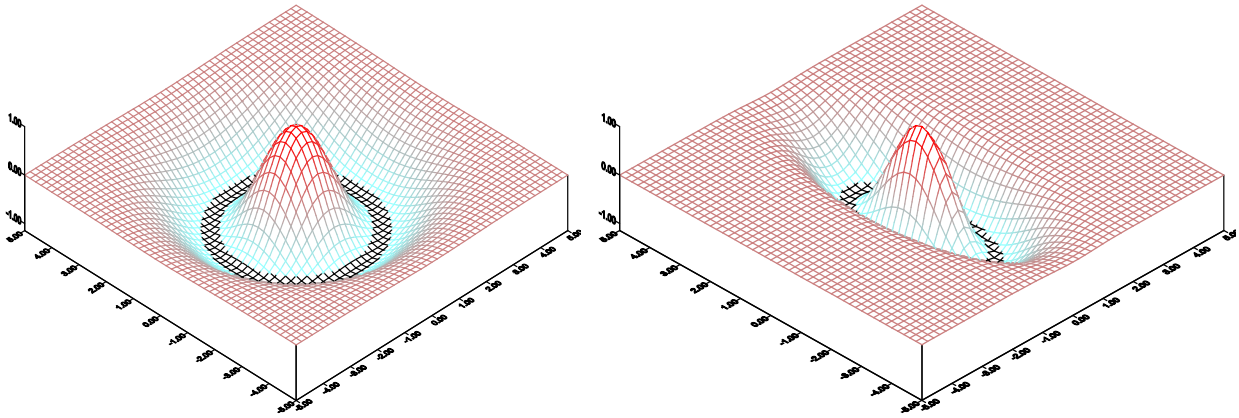
1.2.2 Функция связи типа латеральное торможение

Латеральное торможение – класс рецептивных полей с возбуждающим центром и тормозящей окрестностью (или наоборот). Эта функция порождается лапласианом гауссианы вида $e^{-\frac{r^2}{2\sigma^2}}$ и в общем случае может быть записана как:

$$W(\boldsymbol{r}) = (1 - \boldsymbol{r}B\boldsymbol{r})e^{-\boldsymbol{r}A\boldsymbol{r}} \quad (1.5)$$

, где A и B – параметры, описывающие пространственные характеристики связи.

Фактически латеральное торможение (1.5) относится к классу симметричных фильтров Габора (1.4), в котором используется приближение косинуса (Рис. 1.2).



a) Изотропная функция связи b) Анизотропная функция связи

Рисунок 1.2 — Функция связи типа латеральное торможение

Известно, что фильтр (1.5) выделяет на изображении характерные особенности – края объектов, точечные структуры – и широко используется в компьютерном зрении [123, 124]. Подобные фильтры моделируют отклик нейронов первичной зрительной коры и коленчатого тела, реагирующих на локальные контрасты [125].

1.2.3 Модификация функции связи типа латеральное торможение

Рассмотрим, как с помощью функции связи (1.5) возможно выделять объекты разного направления и размера. Для этого введем упрощенный вариант латерального торможения для двумерного случая с возможностью изменения пространственных размеров и применения преобразования поворота (см. Рис. 1.3):

$$W(x,y) = (1 - b \cdot y^2)e^{-a_x \cdot x^2 - a_y \cdot y^2} \quad (1.6)$$

, где a_x, a_y, b – параметры, определяющие размеры.

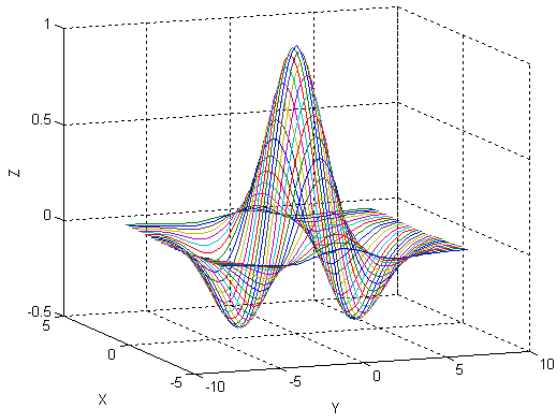


Рисунок 1.3 — Пример визуализации латеральной функции связи, используемой в дальнейшем

Известно, что в системе, описываемой уравнением (1.2) существуют распространяющиеся и неподвижные импульсы, возможно их взаимодействие, формирование стационарных структур. При использовании функции Хевисайда в качестве нелинейного преобразования ($F(Z) = \chi(Z)$), стационарные структуры образуются в том случае, если $-T + \int_D \Phi(\xi)u(r - \xi)d\xi \leq 0$. Тогда решение уравнения (1.2) известно, эквивалентно $u_0 e^{-\frac{t}{\tau_u}}$ и означает затухание активности с размерами, определяемыми характеристическими параметрами (1.6). Определим эти параметры.

Для получения связи между размерами объектов и характеристическими параметрами положим, что максимальный размер объекта есть размер

ненулевой области D функции связи. Тогда неравенство для получения стационарной структуры запишется как:

$$T_0(r) = \int_D \Phi(\xi)\overline{U}d\xi = \overline{U}\int_D \Phi(\xi)d\xi = \kappa\overline{U} \quad (1.7)$$

, где $\overline{U}$ - среднее значение уровней активности элементов в области D.

Следовательно k для функции (1.6) запишется как:

$$\kappa = \int_{-\infty}^{\infty} e^{-a_x \cdot x^2}\, dx \cdot \int_{-\infty}^{\infty}(1 - b \cdot y^2)\, e^{-a_y \cdot y^2} dy \quad (1.8)$$

Решение (1.8) известно:

$$\kappa = \frac{\pi}{\sqrt{a_x \cdot a_y}} \cdot \left(1 - \frac{b}{2 \cdot a_y}\right) \quad (1.9)$$

Без потери общности положим, что порог нелинейной функции $T = 0$, тогда из (1.9) следует, что $a_y = \frac{b}{2}$. Коэффициент b определяет минимальный захватываемый размер объекта на изображении (минимальная толщина гребня, пространственный размер объекта). Правильным выбором этого значения возможно отфильтровать размеры объектов, больших чем L_{min}, который будет связан с b соотношением $b = \frac{4}{L_{min}^2}$.

Максимальную теоретическую толщину объектов, которые останутся на изображении после обработки, можно определить из условия:

$$\left(1 - b \cdot \frac{L_{max}^2}{4}\right) e^{-a_y \cdot \frac{L_{max}^2}{4}} = h_{min} \quad (1.10)$$

, где h_{min} – пороговое значение для функции связи, для изображений с диапазоном значений [0,255] равен $\frac{1}{255}$.

Решение уравнения (1.10) можно записать, используя функцию Ламберта (F_l):

$$L_{max} = \frac{2}{\sqrt{a_y \cdot b}} \sqrt{a_y - b \cdot F_l\left(-\frac{a_y \cdot e^{\frac{a_y}{b}}}{h_{min} \cdot b}\right)} \quad (1.11)$$

Подставляя в (1.11) выражения для a_y и h_{min} получим, что $L_{max} \approx \left\lfloor 2\sqrt{17} \cdot L_{min} \right\rfloor$. Эта оценка L_{max} – теоретический верхний предел размера, достижимый при достаточном количестве итераций однородной среды. Более реалистичные оценки при малом количестве итераций можно получить, взяв за L_{max} расстояние между минимумами функции связи (1.6). Пара минимумов является решением уравнения $a_y b \cdot y^2 = b + a_y$, что приводит к оценке $L_{max} \approx \left\lfloor \sqrt{3} \cdot L_{min} \right\rfloor$. Параметр a_x в (1.6) определяет длину функции связи и в работе выбрана равной $a_x = b = \frac{4}{L_{min}^2}$.

В результате обработки на изображении будут оставаться линии и объекты с размерами, лежащими в диапазоне $[L_{min}, L_{max}]$. Параметр L_{min} в системе определяет границу между шумовыми артефактами и полезным сигналом. Так как минимальная толщина гребня/впадины не может быть меньше одного пиксела, то L_{min}>1.

1.2.4 Задачи решаемые однокомпонентной нейроноподобной средой

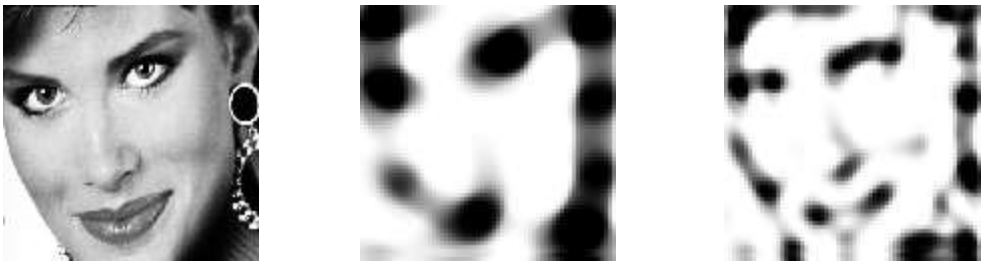
Перечислим некоторые ключевые задачи нелинейной фильтрации изображений, которые могут быть выполнены нейроноподобной средой за счёт настройки соответствующих связей:

- **Выделение контуров и границ объектов:** настройка разных масштабов изотропной функции латерального торможения позволяет обнаруживать границы между областями разной яркости, подчеркивая края объектов на изображении (Рис. 1.4).

Исходное $L_{min} = 6$ $L_{min} = 3$

Рисунок 1.4 — Пример выделения контуров и границ объектов

- **Подавление шума и сглаживание фона:** увеличение области суммирования и выбор соответствующей нелинейности приводит к усреднению мелких флуктуаций, что уменьшает аддитивный шум, сохраняя при этом общую структуру изображения (Рис. 1.5).



Исходное $L_{min} = 24$ $L_{min} = 12$

Рисунок 1.5 — Пример выделения объектов разных масштабов

- **Усиление контрастности мелких деталей:** за счёт нелинейной активационной функции можно выделять размытые детали – например, усиливать слабовыраженные элементы, делая их более заметными.
- **Выделение структурных элементов заданной ориентации или масштаба:** используя набор фильтров разных ориентаций с помощью преобразования поворота, система может извлекать из изображения линии определённого направления, текстурные паттерны с нужным периодом, и т.п. (Рис. 1.6)

0° 45° 90° 135°

Рисунок 1.6 — Фильтр латерального торможения разных направлений фильтрации

- **Обнаружение особых точек (углов, пересечений):** комбинируя несколько латеральных фильтров под разными углами, либо применяя фильтры типа «крест» или «Т-образная» маска, можно выявлять на изображении точки, где сходятся несколько границ (эти ключевые точки часто служат опорными для распознавания сложных объектов).

Существуют и другие методы традиционной обработки изображений (градиентные фильтры Собеля, операторы Кэнни, морфологические операции и т.д.), однако их можно рассматривать как частные случаи или приближенные дискретные аналоги процессов в нейроноподобной среде. Главное преимущество нейроподобного подхода – динамичность и параллелизм. Когда фильтрация реализуется через динамику распределённой сети, возможна многошаговая эволюция системы, в ходе которой выход одной итерации служит входом для следующей. Примером могут служить рекуррентные модели резервуарного вычисления, эффективно расширяющие признаковые пространства независимо от задачи и переносящие временные признаки в пространственное представление [113, 114]. В результате нейроноподобный фильтр способен выделять признаки не за один проход, а постепенно уточнять их по мере стабилизации активности.

В настоящем исследовании использована модель нейроноподобной среды для получения набора признаков изображений лиц перед распознаванием. Нейроноподобная система является однородной в том смысле, что для одного уравнения однокомпонентной среды используется одна функция связи, но выбор конечных признаков содержит набор разнородных рецептивных полей (фильтров) в нескольких масштабах и в разных ориентациях. Путём параллельного применения этих фильтров к исходному изображению формируется многомерное описание, содержащее

обогащенный набор признаков. Далее будет описано, как такие признаки интегрируются в систему распознавания с двухклассовой схемой принятия решений.

1.3 Использование нейроноподобной среды с латеральным торможением в задаче классификации лиц

Для экспериментальной проверки идеи генерации признакового пространства с помощью однокомпонентной однородной среды была создана система распознавания лиц, использующая двухклассовый подход к классификации [1, 19, 163]. В качестве исходных данных использована база изображений людей, включающая фотографии 100 различных персон. Для каждой персоны имелось несколько эталонных изображений лица – в нашем случае мы взяли по 4 примера на каждого человека для формирования обучающей базы. Все изображения были нормализованы по размеру (40x48) и приведены к одинаковым условиям освещения (нормализация средней интенсивности).

1.3.1 Формирование признаков

Каждый эталонный снимок лица преобразуется с помощью набора рецептивных полей (1.6) нейроноподобной среды (1.3) [17]. Обозначим через t число ориентаций используемых функций связи и через s – число разных масштабов. Тогда одно исходное изображение порождает $t \cdot s$ карт признаков (Рис. 1.7). Эти карты активаций можно рассматривать как обобщённые признаковые пространства, где проявляются различные детали исходного образа. Объединяя все карты, мы получаем высоко-размерный вектор признаков, описывающий изображение лица. Формально размерность такого вектора равна $N = w \cdot h \cdot t \cdot s$ (хотя на практике можно использовать более компактное представление, выбирая не все пиксели всех карт, а только статистические показатели или отклики в ключевых точках). Было сделано 5 вариаций по масштабу $L_{min} = \{2, 4, 6, 8, 10\}$ и 8 по углу, через каждые $22.5°$.

Всего получилось 40 типов рецептивных полей. С учетом того, что в работе использовано изображение размером 40x48 пикселей, размерность получившегося пространства признаков составляла 76800. Такой описательный вектор обеспечивает избыточное представление образа, необходимое для последующего повышения надёжности распознавания.

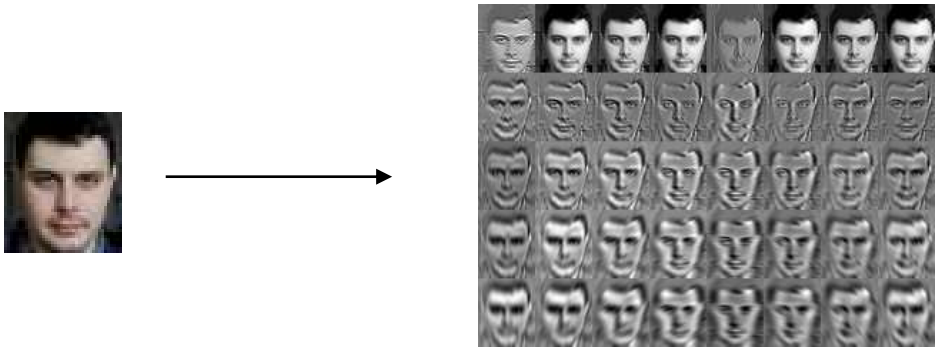


Рисунок 1.7 — Пример карты признаков, полученных с помощью разных масштабов и ориентаций

1.3.2 Внутриклассовое и межклассовое пространства (IOS и EOS)

Ключевая идея использованного алгоритма принятия решения состоит в переводе исходной многоклассовой задачи к серии двухклассовых задач вида «свой/чужой» для каждого потенциального класса. Для этого вводятся понятия *внутриклассового пространства* (Intra-Object Space, IOS) и *внеклассового пространства* (Extra-Object Space, EOS), заимствованные из подхода, предложенного ранее Могаддамом и Пентландом для распознавания лиц [157].

Алгоритм формирования IOS/EOS следующий: берем два произвольных примера i и j, если классы этих примеров совпадают, то относим их к множеству IOS, иначе эта пара попадает в множество EOS. Далее разностное изображение $u^{i,j}=u^i - u^j$ преобразовывалось в признаковое пространство с помощью однокомпонентной среды и набора функций связи. Результат отражает межклассовое различие или внутриклассовые особенности (Рис. 1.8).

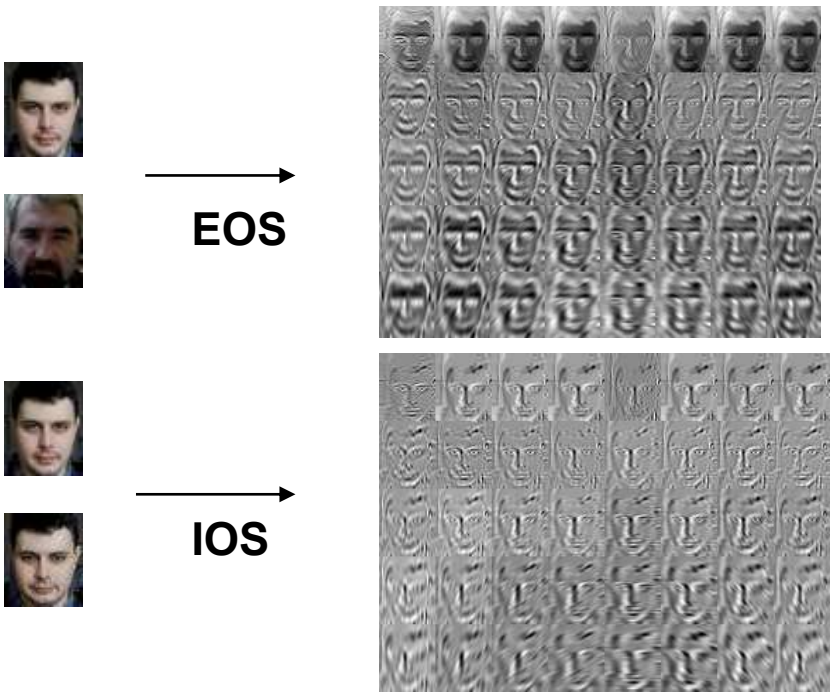


Рисунок 1.8 — Пример межклассовых и внутриклассовых признаков

Повторяя процесс для всех пар эталонных изображений, получаются множества IOS и EOS. В наших данных совокупно получилось порядка 1200 векторов в IOS и около 150000 примеров в EOS. Хотя это весьма несимметричные по объёму выборки, они хорошо характеризуют две разные природы вариаций: внутриклассовые, ограниченные изменениями объекта, и межклассовые (гораздо более разнообразные).

1.3.3 Принятие решений на основе IOS/EOS

Для автоматического принятия решения о принадлежности разностного вектора к IOS или EOS мы построили бинарный классификатор на основе алгоритма *AdaBoost* [69] (см. Приложение 1). В нашем случае в роли слабого классификатора выступает простейший пороговый признак вида:

$$h_z(u^{i,j}) = \begin{cases} 1, u_z^{i,j} > \delta_z \\ 0, otherwise \end{cases} \quad (1.12)$$

, где z – индекс слабого классификатора на общей признаковой карте, δ_z – соответствующий порог.

Правило (1.12) грубо решает, похожи или не похожи два лица по одному локальному признаку яркости для карты k в конкретной точке (m, n). Алгоритм AdaBoost позволяет итеративно подобрать множество таких признаков и объединить их взвешенные голоса в финальное решение. Была реализована каскадная схема, организованная как усечённое дерево решений (Рис. 1.9).

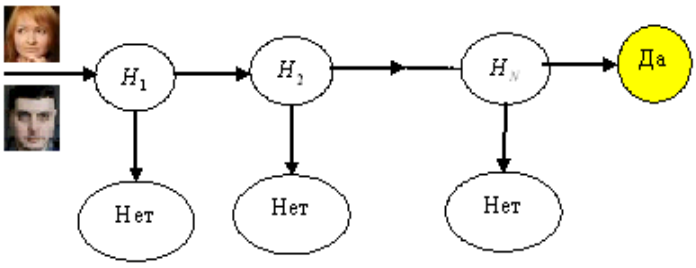


Рисунок 1.9 — Визуализация режима распознавания класса лица: ответ “Да” означает что входные изображения одного лица, в противном случае ответ – “Нет”

Обучение классификаторов проводилось на полученных множествах IOS и EOS [9]. В результате обучения мы получили композицию решающих правил, способных по входному разностному вектору u выдавать решение: входные изображения принадлежат одному человеку или разным людям.

Отметим важную особенность: благодаря использованию динамических нейроноподобных признаков, классификатор обладает определённой устойчивостью к помехам и вариациям изображений. Действительно, рецептивные поля, настроенные, например, на выделение контуров, обеспечивают инвариантность к однородным изменениям освещения и фона. Встроенная в нейроноподобную модель нелинейность служит для подавления шумовых мелких деталей, которые не повторяются на другом изображении того же лица. Все эти свойства «зашиты» в нейроподобные фильтры и не требуют обучения, в отличие от тех же сверточных сетей, которым нужно показать многие тысячи примеров для выработки подобных инвариантов.

1.3.4 Экспериментальные результаты

Работа разработанной системы распознавания лиц была протестирована на базе данных из 100 человек. Изображения были разделены на обучающую (галерея) и тестовую выборки. Галерея включала по 4 эталонных изображения каждого лица (эти снимки использовались для формирования признаков и множества IOS/EOS, а также служили эталонами для сравнения). Тестовая выборка состояла из новых фотографий тех же людей, не использованных в галерее; мы взяли по 5 тестовых изображений на каждого из 100 человек, из которых было получено 7500 пар изображений, формирующих сравнительные признаки.

Для оценки качества распознавания использовались метрики, принятые в задачах верификации (двухклассовой классификации «свой/чужой»): ошибка первого рода (FRR) – доля пропущенных собственных объектов, и ошибка второго рода (FAR) – доля ложных срабатываний. В идеале обе эти ошибки стремятся к нулю, но обычно между ними существует компромисс, зависящий от порога решающего правила. В исследуемом алгоритме таким настраиваемым параметром служит порог K на выходе последнего сильного классификатора. Пересечение этих кривых соответствует точке Equal Error Rate (EER), где ошибки первого и второго рода равны между собой. Этот показатель часто используется для сравнения алгоритмов, так как EER не зависит от произвольного выбора порога.

В первоначальной конфигурации (базовый набор рецептивных полей, без дополнительных оптимизаций) алгоритм показал значение EER на уровне ~13%. Этот результат служит отправной точкой для анализа (Рис. 1.10). Далее предпринят ряд улучшений, направленный на снижение ошибки. Во-первых, были испытаны различные параметры нейроноподобных фильтров. В частности, изменялся размер фильтров и их эксцентриситет, управляемый параметрами a и b в уравнении (1.6). Например, для латерального торможения пробовались значения эксцентриситета e = 4 и e = 8 ($b = 2$ и $b = 2.8$), что

соответствует более «растянутому» центру относительно окружения, а также дополнительные наборы масштабов L_{min}: $L^1 = \{2, 6, 10, 14, 18\}$ и $L^2 = \{2, 4, 8, 16, 32\}$. Во всех случаях перестройка фильтров требовала пересчёта признаков и повторного обучения классификатора, после чего заново измерялись FAR/FRR. Лучшие найденные параметры позволили улучшить результат распознавания до EER на уровне ~11%. Таким образом, за счёт подбора фильтров удалось ощутимо (примерно на 2 процентных пункта) повысить точность системы (Рис. 1.11). Интуитивно это объясняется тем, что более крупные рецептивные поля с большим эксцентриситетом лучше улавливают глобальные различия в структуре лица (например, форму головы или разрез глаз), тогда как мелкие фильтры дают много деталей, полезных для различения похожих лиц, но и привносят шум. Правильный баланс масштаба и формы фильтров позволил снизить как пропуски, так и ложные срабатывания.

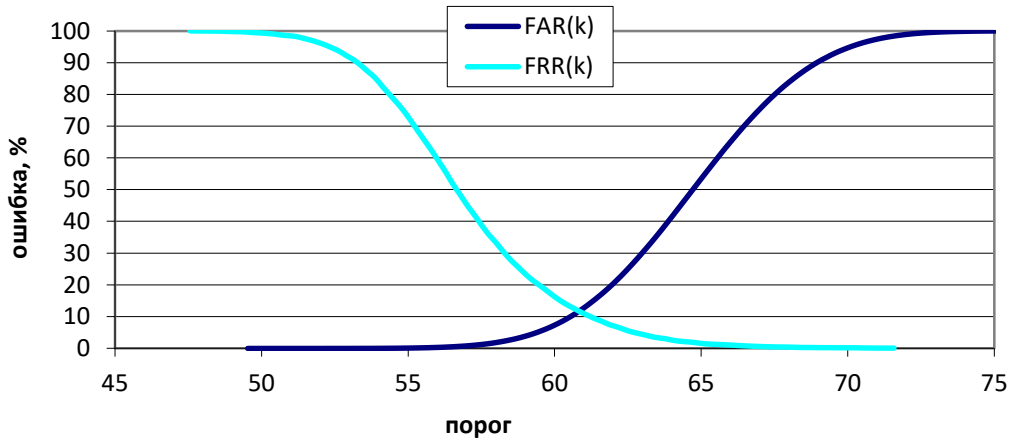


Рисунок 1.10 —Ошибка классификации с использование начального набора рецептивных полей

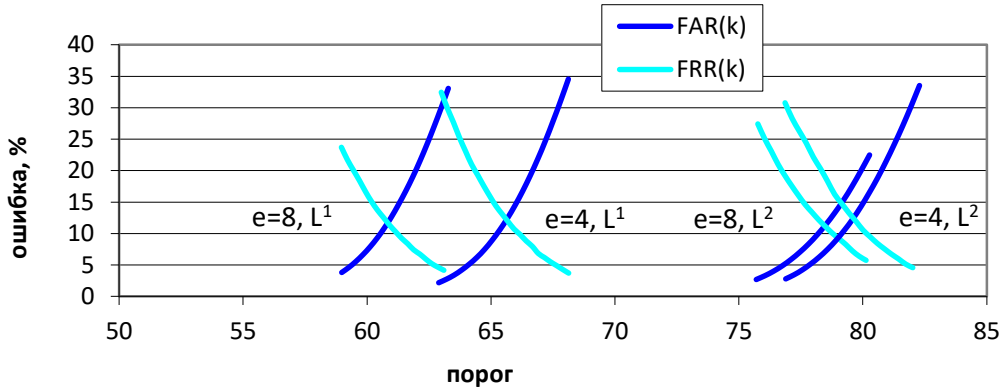
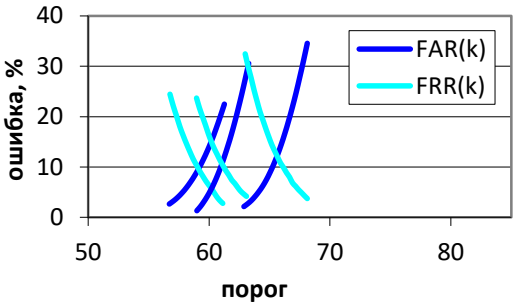
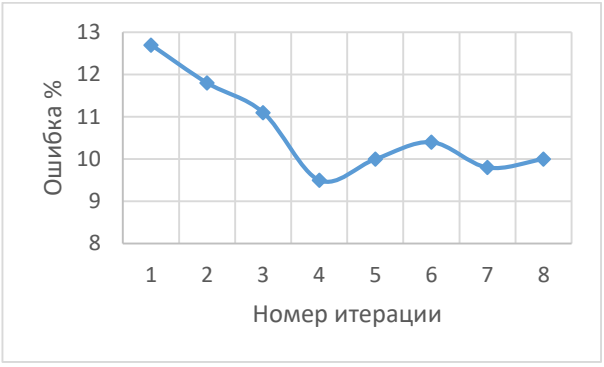


Рисунок 1.11 — Результаты оптимизации параметров рецептивных полей

Во-вторых, было рассмотрено влияние итеративной динамики нейроноподобной среды на качество признаков. Нейроноподобная модель может применяться к изображению не мгновенно, а через симуляцию нескольких шагов эволюции (итераций), в течение которых система достигает некоторого устойчивого или псевдоустойчивого состояния. В наших экспериментах разностные изображения проходили 1, 2, 3, 4 и более итераций фильтрации и каждый раз классификатор строился заново, чтобы оценить, насколько итоговая точность зависит от числа шагов. Интересно, что при увеличении числа итераций с 1 до 4 наблюдалось улучшение результатов распознавания: ошибки FAR и FRR снижались (Рис. 1.12 a), достигая минимума около 9.5% при 4 итерациях (Рис. 1.12 b). Однако при дальнейшем увеличении числа итераций (5, 6 и т.д.) качество начинало ухудшаться: ошибки вновь росли. Мы связываем это с тем, что чрезмерно долгое развитие волновых процессов в среде приводит к деградации деталей – мелкие, но значимые различия между лицами могут сглаживаться, если фильтрация идёт слишком долго и выходит на насыщение (стационарное состояние, где активность выровнена), кроме того, функция связи латерального типа обладает генеративным свойством и может добавлять и искажать элементы изображения, тем самым ухудшая качество классификатора. Таким образом, 4 шага можно считать оптимальным псевдостационарным состоянием, при котором ключевые признаки ещё присутствуют, а помехи уже существенно

подавлены для используемой экспериментальной базы. В итоге, благодаря учёту этого фактора (выбору оптимального числа итераций), ошибка распознавания была уменьшена с ~13% до ~9.5% – то есть относительное снижение ошибки составило порядка 27%.





a) b)

Рисунок 1.12 — a) Зависимости ошибки Far/Frr от порога, полученные при 4, 3 и 1 итерации нейроноподобной среды (слева направо); b) Зависимость ошибки от номера итерации

Суммарно, финальная версия системы продемонстрировала точность распознавания ~90–91% на тестовой выборке из 100 классов. Это весьма высокий показатель, учитывая простоту базовых признаков. Для сравнения, классические методы распознавания лиц на основе принципиальных компонент [176] или линейного дискриминантного анализа [40] при таком же объёме данных обычно давали более высокие ошибки, особенно на неконтролируемых изображениях. Конечно, новейшие глубокие сети сейчас достигают точности >99% на больших наборах [146, 159], однако их успех во многом обусловлен наличием огромных обучающих выборок и мощных вычислительных ресурсов. В условиях ограниченных данных и потребности в интерпретируемости решений нейроноподобный подход выглядит конкурентоспособным. Он подтверждает, что биологически вдохновленные методы могут эффективно решать задачи распознавания, а порой и дополнять современные нейросетевые технологии.

1.4 Выводы

В данной главе показано, что динамика нейроноподобных сред может быть успешно использована для фильтрации изображений и извлечения признаков, значимых в задачах классификации. Была прослежена эволюция понятия нейроноподобной распределённой системы от ранних моделей нервной ткани [32, 36, 97, 183] до современных алгоритмов обработки изображений. На примерах продемонстрировано, что, варьируя параметры такой системы, можно реализовать широкий спектр нелинейных фильтров – от выделения контуров до подавления шума и выделения ключевых точек. Это позволяет строить неоднородные нейроподобные среды, сочетающие различные типы рецептивных полей, что особенно полезно для анализа сложных визуальных сцен.

Экспериментальная часть, посвящённая распознаванию лиц, показала эффективность двухклассового подхода в комбинации с нейроноподобной фильтрацией [17, 163]. Преобразование многоклассовой задачи к серии проверок «свой или чужой» для каждого класса позволило упростить принятие решений и воспользоваться богатыми статистическими описаниями пространств IOS/EOS. Полученная система показала высокую точность на базе из 100 лиц при небольшом числе эталонов, а введение нейроподобных признаков и их многократное динамическое обновление заметно повысили надёжность распознавания. Достигнутая точность (порядка 90%) сопоставима с результатами традиционных алгоритмов машинного зрения того же класса сложности, что подтверждает жизнеспособность предложенного подхода.

Важно подчеркнуть, что методы нейроподобной фильтрации и распознавания актуальны и на современном этапе развития технологий. Несмотря на доминирование глубокого обучения, нейроморфные и биологически-инспирированные алгоритмы находят своё применение: они менее требовательны к данным, часто интерпретируемы и могут быть реализованы аппаратно для ускорения вычислений. Современные

исследования продолжают развивать идеи динамических нейронных полей – например, для когнитивной робототехники и обработки сенсорных данных в реальном времени [107]. Также развиваются аппаратные реализации нейроноподобных сетей на новых элементах, что может привести к появлению энергоэффективных устройств для мгновенной обработки.

Подводя итог, можно заключить, что сочетание нейроноподобной предобработки изображений с классическими алгоритмами классификации открывает широкие возможности в распознавании образов. Такой подход, с одной стороны, опирается на фундаментальные принципы нейрофизиологии (коллективная динамика, рецептивные поля, латеральное взаимодействие), а с другой – интегрируется с современными методами машинного обучения. В результате удаётся строить эффективные и относительно простые системы, способные решать практические задачи распознавания, подтверждая актуальность и перспективность нейроподобных методов в науке и технологиях распознавания образов.

ГЛАВА 2. ВЫБОР АЛГОРИТМОВ ПРЕОБРАЗОВАНИЯ ИЗОБРАЖЕНИЙ ПО ДИНАМИКЕ НЕВЯЗКИ НА ПРИМЕРЕ ЗАДАЧИ КЛАССИФИКАЦИИ ОТПЕЧАТКОВ ПАЛЬЦЕВ

Задача выбора подходящего алгоритма имеет ключевое значение при разработке систем оптимальной обработки данных, особенно в контексте задач классификации образов и биометрического распознавания (например, отпечатков пальцев). Известно, что не существует универсального метода, одинаково эффективного для любых данных и условий – согласно теореме «No Free Lunch», ни один алгоритм не превосходит все прочие на всех возможных задачах [186]. Поэтому оптимальный выбор алгоритма должен основываться на особенностях обрабатываемых данных, критериях качества и целевых метриках эффективности.

Традиционно оценка алгоритмов распознавания проводится на больших тестовых выборках с расчётом статистических показателей ошибок – например, false accept rate (FAR) и false reject rate (FRR), отражающих вероятности ошибок I и II рода. Однако подобная проверка не только трудоёмка и затратна по времени, но и мало помогает выявить, какие именно этапы алгоритма ответственны за снижение точности. Возникает потребность в более тонком подходе: анализировать качество и пригодность отдельных примеров в связке с конкретным алгоритмом, чтобы понимать, насколько хорошо алгоритм справляется с различными типами входных данных. Учёт этих факторов качества примеров позволяет адаптивно выбирать стратегию обработки: либо предварительно улучшать данные (фильтрация, нормализация), либо переключаться на другой алгоритм, более подходящий для данных с данным уровнем качества [10, 14, 16].

Исторически идеи адаптивной обработки информации во многом черпались из аналогий с биологическими нейронными системами мозга живых организмов. Биоморфные алгоритмы (нейроноподобные) стремятся имитировать принципы, присущие нервной системе, такие как параллелизм,

пластичность и локальное обучение. Еще в 1980-х годах К. Фукушима предложил нейросеть neocognitron [73] для распознавания образов по аналогии с иерархией зрительной коры, а его последующие работы показали возможность избирательного внимания и ассоциативной памяти в нейронных моделях [74]. Современные исследования подтверждают, что заимствование механизмов мозга может улучшить алгоритмы машинного обучения [121]. В частности, человек по-прежнему превосходит машины во многих когнитивных задачах (творчество, восприятие сложных образов и пр.), поэтому внедрение биологически правдоподобных стратегий обучения и адаптации помогает сократить разрыв между искусственными и биологическими системами. Примерами могут служить алгоритмы локального обучения без расчета градиентов (Contrastive Hebbian Learning, Equilibrium Propagation) [135, 156], вдохновленные синаптической пластичностью, нейроморфные процессы обработки сигналов, имитирующие работу сенсорных нейронов, и механизмы внимания, позволяющие выделять ключевые признаки.

Одним из способов повысить эффективность обработки данных и выделения информативных признаков является применение методов сжатия и восстановления (кодирования–декодирования) информации [23]. Примером такого подхода являются автоэнкодеры – нейронные сети, обучаемые воспроизводить свои входные данные на выходе через сжатое представление. Ещё в работах Г. Хинтона было показано, что многослойный автоэнкодер способен обучаться преобразованию высокоразмерных данных к представлению низкой размерности, сохраняя при этом существенные структуры исходных данных [96]. Последующая реконструкция (декодирование) из этого кода позволяет контролировать, какая информация потеряна при сжатии, тем самым характеризуя пригодность сжатого представления для распознавания.

Помимо автоэнкодеров, в задачах представления данных широко применяются методы *разреженного кодирования* (sparse coding) и связанные

алгоритмы обучения словарей признаков. В классической работе B. Olshausen и D. Field показано, что обучение модели представления изображений с явным стремлением к разреженности кода приводит к появлению базисных функций, схожих с рецептивными полями нейронов в первичной зрительной коре [140]. Идея разреженного кодирования состоит в том, что сложный сигнал можно выразить через линейную комбинацию небольшого числа базисных шаблонов, что обеспечивает одновременно и сжатие данных, и их интерпретируемость. При этом разреженные представления, как и автоэнкодерные коды, можно рассматривать как своеобразные признаковые пространства, более удобные для последующей классификации. Аналогичные подходы такие как разреженные автокодировщики, метод главных компонент (PCA) и др. решают общую задачу: найти такое преобразование данных, которое уменьшает размерность или избыточность, сохраняя при этом наиболее важную для распознавания информацию. В историческом контексте можно упомянуть и так называемую «машину Гельмгольца» [63], являющуюся одним из первых байесовских подходов к совместному обучению кодирующей и декодирующей моделей. Современные варианты этих идей – вариационные и разреженные модели – закладывают основу для адаптивных систем, способных самонастраиваться под структуру данных.

Особое место в задачах выбора алгоритма занимает концепция *адаптивной распознающей ячейки*, предложенная в трудах В.Г. Яхно и соавторов [29]. Эта ячейка представляет собой элементарный модуль адаптивной системы распознавания – динамическую нейроподобную подсистему, интегрирующую в себе уровни кодирования, декодирования, оценивания и принятия решений. Идея заключается в том, что поток информации обрабатывается в замкнутом цикле: входные сигналы преобразуются (кодируются) в компактное описание, из которого затем восстанавливаются (декодируются) с некоторой погрешностью; параллельно вычисляются *мотивационные оценки* – показатели соответствия между оригиналом и реконструкцией (так называемые «невязки»), а также другие

статистики качества распознавания. Эти оценки выступают сигналами обратной связи, влияющими на параметры алгоритмов на лету. Если ошибки велики, ячейка может адаптироваться: скорректировать внутренние параметры или даже заменить текущий алгоритм на альтернативный для улучшения качества кодирования.

Биомеханизмы обработки информации в коре головного мозга послужили источником вдохновения для этой многоуровневой схемы. В частности, адаптивная ячейка может быть интерпретирована как аналог элементарного функционального блока мозга, где непрерывно происходит оценка рассогласований и поддерживается гомеостаз активности. Если провести параллель, то мотивирующие оценки в ячейке можно уподобить нейрофизиологическим сигналам «ошибки предсказания», которые, как известно, играют значимую роль в работе коры и таламуса, позволяя мозгу подстраивать восприятие под ожидания.

В качестве примера, иллюстрирующего поведение адаптивной ячейки, можно привести процесс распознавания визуального образа в модели «кора–таламус». При подаче постоянного входного изображения система не переходит в стационарное состояние сразу. Вместо этого она генерирует серию циклических откликов: сначала выделяются одни фрагменты или признаки изображения, затем по мере накопления оценок – другие, и так продолжается до стабилизации [32]. Каждый такой цикл – это попытка системы согласовать внутреннюю модель с входным сигналом, что схоже с тем, как мозг постепенно «встраивает» новое сенсорное воздействие в свою нейронную динамику, возможно через последовательность автоволновых вспышек активности [22]. Подобная циклическая самоорганизация улучшает надёжность распознавания: признаки, невыявленные на одном этапе, могут быть уловлены на следующем, после подстройки алгоритма. Таким образом, адаптивная нейроподобная ячейка обеспечивает не только статическое распознавание образа, но и его динамическое «прояснение» через повторяющийся процесс кодирования-декодирования и оценки качества.

Концепция, в которой эффективность работы алгоритма контролируется динамикой изменений входных данных или внутренних состояний, находит отражение и в других современных подходах к распознаванию. В частности, в области машинного обучения известна проблема *изменчивости статистики данных во времени* (concept drift) – когда распределение входных образцов со временем меняется, что может приводить к деградации качества классификатора. Для решения этой проблемы разрабатываются адаптивные алгоритмы, способные подстраиваться к постепенным или резким изменениям во входном потоке. Так, исследователи в области биометрии отмечают, что характеристики биометрических признаков пользователя (например, почерк, динамика набора текста, даже отпечатки пальцев при многократном сканировании) могут со временем изменяться, и система должна уметь обучаться на лету новым данным [151]. В работе F. Roli подробно рассмотрены *адаптивные биометрические системы, улучшающие свои показатели в процессе эксплуатации*, в частности за счёт использования неразмеченных данных пользователей для переподстройки модели классификации. Суть этих решений в том, что система постоянно отслеживает качество распознавания (например, процент успешно пройденных аутентификаций) и при накоплении достаточной статистики отклонений производит обновление своих внутренних параметров либо расширяет эталонную базу новыми образцами.

В практических биометрических системах уже применяются техники автоматической коррекции: адаптивное обновление шаблонов (эталонных образцов) отпечатков пальцев по мере поступления новых данных, регулирование порогов решения под текущие условия и т.п. Например, методика адаптивного порога для системы распознавания отпечатков может динамически менять критерий совпадения в зависимости от внешних факторов – нагрузки на систему, требуемого уровня безопасности и др. [105] Другой пример – периодическое переобучение классификатора отпечатков на новых поступивших данных пользователя, что позволяет учесть постепенное изменение состояния кожи пальца или появление новых мелких деталей на

отпечатке. Все эти подходы имеют общий принцип: отслеживать динамику ошибок (или других показателей эффективности) и на её основе вносить изменения в алгоритм. В этом плане они методологически близки к концепции адаптивной ячейке, где также заложен механизм непрерывной оценки качества (через «невязки») и соответствующей перенастройки алгоритма.

Следует отметить, что аналогичные идеи прослеживаются и в биологически вдохновленных алгоритмах обучения. Например, алгоритмы с механизмом обратной связи по ошибке – такие как адаптивный резонанс (ART) [49, 87] или разного рода самоорганизующиеся карты [115, 132] (с дополнительной фазой обучения при детекции новизны) – по сути реализуют базовый принцип: при существенной новизне входного сигнала скорректировать внутреннее представление [86]. В современных нейронных сетях с онлайн-обучением схожий подход используется для непрерывного обучения без забывания (continual learning) – модель старается сохранить уже накопленные знания, адаптируясь к новым данным посредством специальных стратегий (регуляризации, динамического изменения архитектуры и т.п.). Наконец, в задачах технического зрения внедряются схемы обратной связи, где результаты распознавания высокого уровня влияют на низкоуровневую обработку изображения (например, фокусировка на области кадра, если там предположительно находится интересующий объект). Все эти решения подчёркивают важность учета временной динамики и контекста при оценке эффективности распознавания: не достаточно разово обучить алгоритм – необходимо сопровождать его работу мониторингом и при необходимости адаптировать.

2.1 Адаптивная модель принятия решений

Характерные элементы адаптивной системы, предложенной в [26, 29] , и направления потоков информации можно представить в виде схемы Рис. 2.1.

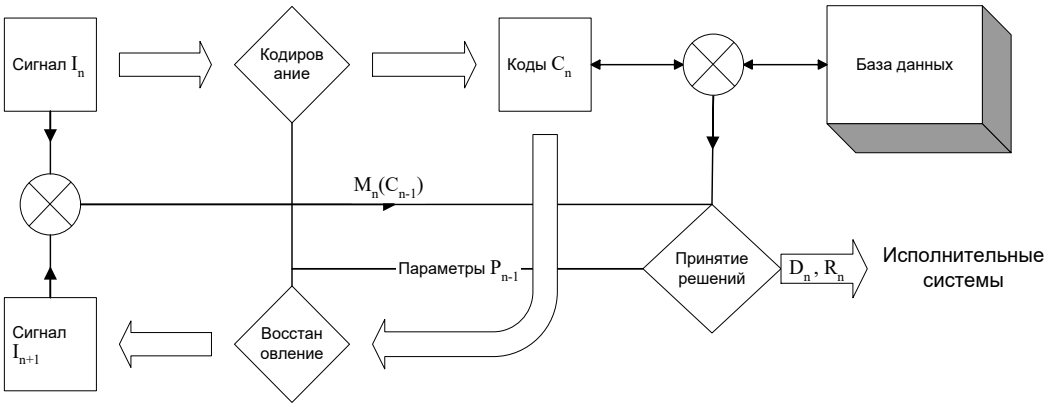


Рисунок 2.1 — Схема адаптивной модели

Схема представляет собой непрерывно анализирующую входной сигнал (I_n) систему. На каждой итерации осуществляется контроль работы алгоритмов кодирования и декодирования, их адаптация на входной сигнал при помощи изменения параметров, а также настройка элемента принятия решения через изменение порогов. База данных накапливает информацию о работе текущего алгоритма кодирования/декодирования, кодовых сигналах (C_n) и принятых решениях.

Основным элементом контроля результатов работы является определение невязок различных параметров. Невязки характеризуют степень отклонения величины от номинала (при конкретных условиях) и обычно находятся в диапазоне [0,1]. К таким величинам обычно относят время выполнения задачи, потребленную энергию, значения целевой функции, ошибки выполнения и т.д. Важное отличие рассматриваемой здесь архитектуры адаптивных систем, соответствующих «живым» прототипам, от «обычных» систем распознавания заключается в обязательном существовании в них специальных подсистем для создания имитации ожидаемого входного сенсорного сигнала и сравнения его с реально поступившим сигналом (блоки сравнения и получения ожидаемого, восстановленного сигнала) [172].

Фактически, алгоритмы блока восстановления позволяют распознающей системе создавать «виртуальную реальность», которую необходимо сравнивать с реальными входными сигналами для принятия адекватных решений. Имеется множество подтверждений существования

таких блоков создания «виртуальной реальности» в живых системах. К наиболее известным из них можно отнести процессы сна, возможных галлюцинаций, процессы идентификации и категоризации в сенсорных системах, различные режимы анализа проблемных ситуаций, когда на «внутреннем экране» могут возникать яркие образы объектов или ситуаций, с которыми живое существо играет или просто наблюдает динамику этого процесса.

Для описания возможных динамических режимов настроек функциональной модели (Рис. 2.1) используются следующие переменные:

I_n - входной сигнал, поступающий на обработку в распознающую систему;

C_n - векторы кодового описания входного сигнала;

P_n - управляющие параметры всех, участвующих в работе алгоритмов;

$M_n(C_{n-1})$ - вектор (или изображение) для величин невязок - мотиваций, полученных из сравнения наборов кодов входного сигнала и кодовых описаний для предварительно ожидаемого системой сигнала;

D_n - решения, принятые распознающей системой;

R_n - оценки уверенности, статистической достоверности принятого решения;

I_{n+1} - сигнал на выходе распознающей системы.

Использование схематического описания адаптивной распознающей системы (Рис. 2.1) позволяет ввести формальные определения для ряда динамических процессов и используемых переменных, которые обычно связаны с понятиями, вызывающими споры при обсуждении информационных систем:

- **Данными** называются наборы признаков или кодовых описаний, вычисленные с входных сигналов, или сами эти сигналы. Хранятся данные в блоке данных.

- **Знаниями** в рассматриваемой системе, как уже отмечалось, естественно назвать наборы алгоритмов (знаний), которые могут использоваться в процессах обработки и принятия решений.
- **Ценность входного информационного сигнала** может определяться по величинам невязок из блоков сравнения, вычисляемых из наборов кодовых описаний для предварительно ожидаемого системой сигнала (C_{n-1}) и реально вычисленными кодами от входного изображения (C_n). Эти же величины невязок служат стимулом для формирования **мотиваций** и принимаемых решений для дальнейшего функционирования системы. Фактически через алгоритмы принятия решений осуществляется управление состояниями всех частей распознающей системы.

Противоречия между входной информацией и представлениями о ней возможно обнаружить по невязкам алгоритмов кодирования-декодирования, полученых из сравнения с эталоном. Сигналом к смене алгоритмов, например, может служить превышение пороговых значений невязкой.

Существуют различные стратегии и критерии к замене модели знания – начиная от полного перебора всех моделей и заканчивая выбором первой, удовлетворяющей условиям оптимизации. Но наиболее важным итогом этого процесса должно стать однозначное соответствие входных сигналов кодам и кодов – восстановленным сигналам.

2.2 Оценка качества работы алгоритмов

В обработке отпечатков широко применяются фильтры для повышения контрастности папиллярных линий, удаления шумов и заполнения разрывов узора. Существует много алгоритмов улучшения, например, гистограммные методы, Габоровские фильтры, адаптивные усилители контраста и др. [98]. Однако универсальный фильтр, идеально подходящий для всех случаев, трудно подобрать. Активные (нелинейные) фильтры могут хорошо восстанавливать смазанные отпечатки, но избыточны для четких

изображений; пассивные (линейные) фильтры быстрее и проще, но не способны восстановить утраченные элементы. В работе рассматривается два подхода: (1) нейроподобный активный фильтр, способный *генерировать* недостающие фрагменты узора; (2) классический пассивный фильтр, устраняющий низкочастотный фон (размытость) и выравнивающий яркость. Задача адаптивной системы – сравнить их эффективность на данном изображении и выбрать лучший.

Основная особенность базовой модельной системы Рис. 2.1 заключается в формальном представлении необходимых и достаточных управляющих воздействий между информационными потоками и алгоритмами обработки этих потоков. В частности, вычисленные параметры изображений информационных сигналов (входные и промежуточные изображения, разнообразные кодовые описания, оценки точности соответствия, или “невязки” – мотивационные оценки) влияют на величины управляющих сигналов для каждой из операций (кодирований декодирования, вычисления невязок, принятия решений), которые в свою очередь изменяют параметры информационного потока. По виду динамики этого процесса можно сделать оценки точности для используемых алгоритмов кодирования-декодирования и сформировать мотивационные сигналы для реакции системы. По ним, например можно найти условия, при которых необходимо корректировать параметры действующих алгоритмов или проводить замену “старых” алгоритмов на “новые” алгоритмы кодирования. Основной задачей этой главы является показать возможность применения схемы Рис 2.1 для определения формализованной оценки работы алгоритмов фильтров и определить участки сигнала с сильным искажением и поэтому неприменимых в системе.

2.2.1 Описание модели сравнения алгоритмов

Как уже упоминалось, точность работы алгоритмов распознавания, даже при небольшом наборе обучающих выборок, может быть проверена по результатам циклического процесса. Скорость сходимости значений в цикле

“кодирование – восстановление – кодирование -…” и величины отличий от первоначально полученных значений характеризуют адекватность и точность выбранных алгоритмов [28]. Такой подход используется в работе для вычисления оценок качества выполненных операций алгоритмами фильтрации изображений дактоотпечатков, а полученные результаты ориентированы на повышение точности систем распознавания по биометрическим данным.

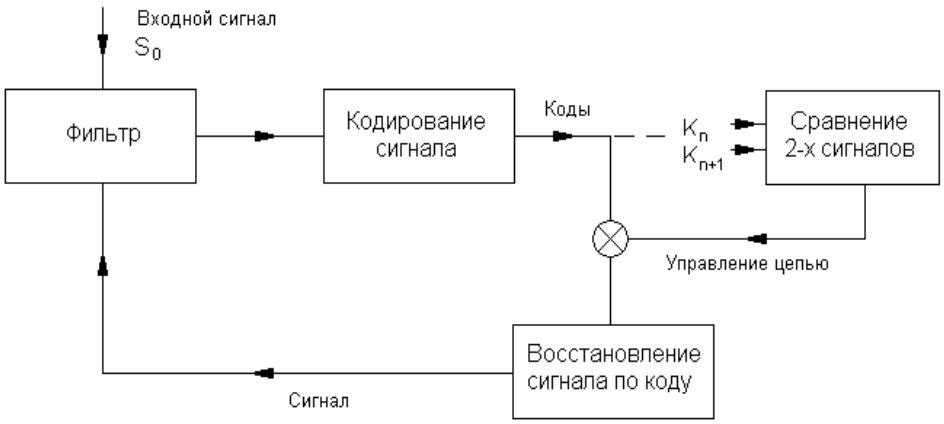


Рисунок 2.2 — Схема алгоритма анализа фильтров

В рамках модели Рис. 2.1 была построена упрощенная схема, сохраняющая при этом базовые особенности более общей, выполняющая анализ алгоритмов фильтрации (бинаризации) отпечатков пальцев (Рис 2.2). Входной сигнал поступает на один из исследуемых фильтров и проходит цепь кодирования для построения его упрощенного описания. Предполагается, что сигнал является информационно избыточными. Это условие является необходимым для правильного восстановления сигнала фильтром. Объем кодированного сигнала, у которого информационная избыточность полностью или частично устранена, много меньше, чем объем исходного сигнала. Сравнение кодов вследствие этого происходит гораздо быстрее, чем самих сигналов.

Полученные коды исходного сигнала поступают на восстанавливающую систему, и затем возвращается на вход фильтра. Сравнение происходит по двум кодовым описаниям, полученным в такой цепи с разностью в одну

итерацию. Из сравнения таких изображений формируется невязка между кодами. По изменению величин вычисляемой невязки в зависимости от номера итерации определяется «качество» используемого алгоритма.

Для исследования были использованы два различных фильтра – активный и пассивный, созданные для улучшения структуры отпечатков пальцев. На вход цепи подавалось изображение в градации серого. В качестве этапа кодирования в этой работе была применена пороговая бинаризация, преобразующая фильтрованное изображение в двоичное поле. Далее приведены описания каждого компонента системы Рис 2.2, использованного в работе.

2.2.2 Активный фильтр

Активный фильтр выполнен на базе нейроноподобной модели (1.2), с помощью неоднородной функции связи вида (1.6). Основной особенностью фильтра является использование визуальных ключей на цифровом образе для восстановления структуры отпечатка [27]. Фильтр активный – так как способен выполнять локальное усиление участков изображения (по яркости), сращивание областей разрывов, изменение расстояния между объектами.

Схема алгоритма представлена на Рис. 2.3. По входному изображению на первом этапе строятся характерные поля – толщины гребней и их направления. Эти поля используются на втором этапе обработки нейроноподобной средой (1.3).

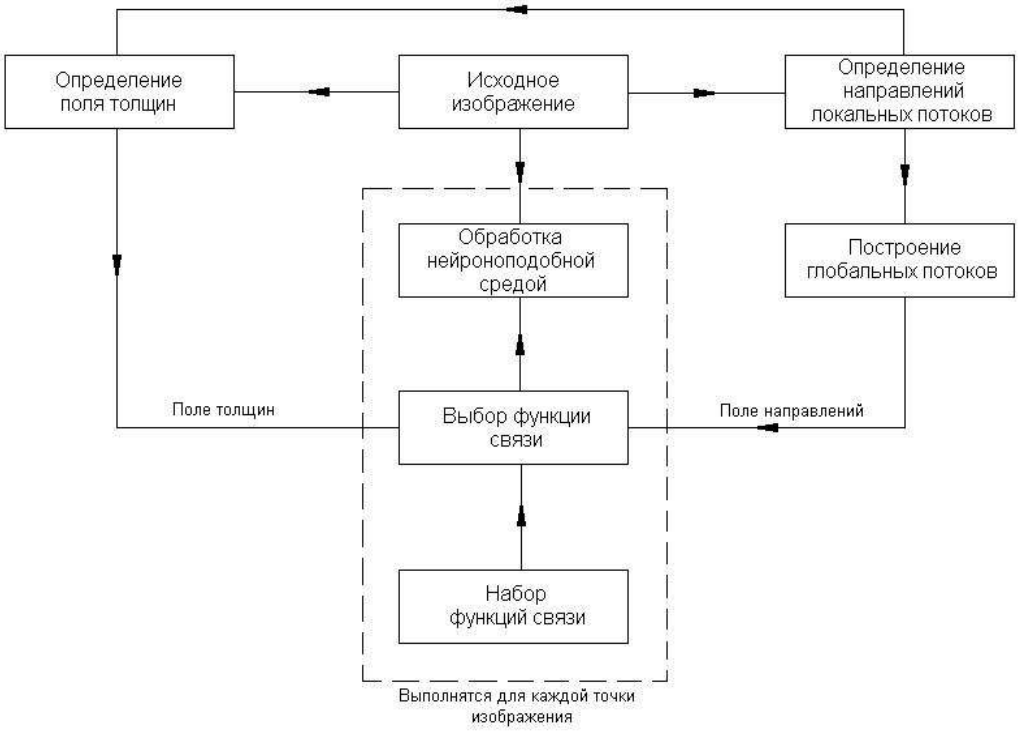


Рисунок 2.3 — Схема активного фильтра

В работе [27] были определены значения параметров, определяющие размеры $\Phi(r)$. Это $a_x = \frac{4}{L_{min}^2}$, $a_y = \frac{b}{2}$, $T_0 = 0$ для толщин линий в пределах $[L_{min}, L_{max}]$, где $L_{max} \approx \left[2 \cdot \sqrt{17} \cdot L_{min}\right]$ Используя эти данные, построен набор функций связи, перекрывающий диапазон [2,7] пикселов поперечных линейных размеров гребней на отпечатке. По данным направления и толщины находится ближайшая, удовлетворяющая требованиям, функция, которая затем используется в нейроноподобной среде.

Благодаря полям толщин и потоков направлений, характерных для каждого изображения, происходит адаптация неоднородной среды на входной отпечаток. Проще говоря, активный фильтр “реагирует” на обрывы папиллярных линий – там его нейроноподобные элементы активируются и заполняют разрыв, связываясь с соседними участками рисунка. Благодаря этому активный фильтр способен локально усиливать контраст, *сращивать разрывы* линий и даже слегка корректировать геометрию узора (например, выравнивать расстояние между линиями). На выходе активного фильтра получается улучшенное изображение отпечатка, где шумы устранены, а

разорванные линии соединены. Далее это изображение бинаризуется порогом, равным среднему значению серого, чтобы получить двоичную маску гребней, пригодную для дальнейшего сравнения.

2.2.3 Пассивный фильтр

Пассивный фильтр, напротив, не пытается дорисовать недостающие детали, а лишь вычищает шум и выравнивает яркость. В реализованном варианте это высокочастотный фильтр с адаптацией по локальному среднему: из исходного изображения вычитается его сглаженная версия (низкочастотная компонента), считая последнюю шумом или фоном. Эта составляющая формируется билинейным однопараметрическим фильтром, параметр которого определяет область частот, выделяемых из исходного изображения. В результате работы фильтра получается поле, значение каждой точки которого принимается за локальное значение среднего для некоторой области входного изображения. Используя такое пространственное распределение среднего, строится локальная дисперсия, как квадрат разности в каждой точке между интенсивностью на исходном изображении и средним. На завершающем этапе происходит формирование изображения с определёнными средним и дисперсией (в данной работе среднее и дисперсия приняты равными 0.5 при максимуме интенсивности 1.0), согласно следующей зависимости (2.1):

$$I_{filt}(i,j) = \begin{cases} 0.5 + \sqrt{\frac{0.5 \cdot \left(I(i,j) - M(i,j)\right)^2}{VAR(i,j)}}, I(i,j) \geq M(i,j) \\ 0.5 - \sqrt{\frac{0.5 \cdot \left(I(i,j) - M(i,j)\right)^2}{VAR(i,j)}}, otherwise \end{cases} \quad (2.1)$$

, где I_{filt} - реконструированное изображение

I - исходное изображение

M - распределение среднего значения по изображению

VAR - распределение дисперсии

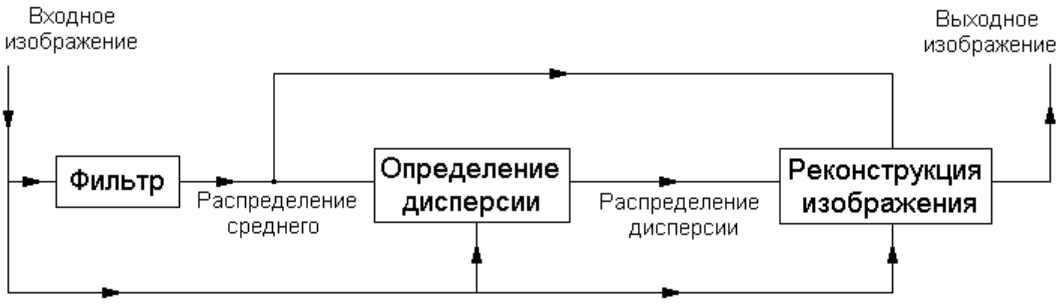


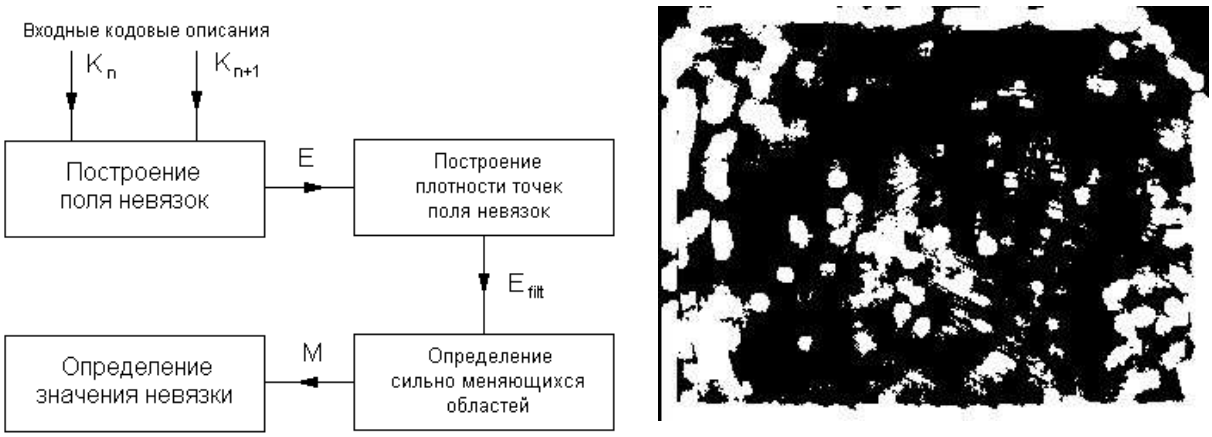
Рисунок 2.4 — Схема пассивного фильтра

Этот алгоритм (Рис 2.4) эквивалентен следующему: делаем изображение более однородным по яркости, повышая контраст в бледных местах и понижая в пересвеченных, при этом общая структура папиллярных линий сохраняется. На последнем этапе также выполняется пороговая бинаризация (порог 0.5) для получения двоичного шаблона линий. Пассивный фильтр, таким образом, улучшает соотношение сигнал/шум, удаляя фоновые искажения (например, неравномерное давление пальца на сканер, приводящее к градиенту яркости) и частично устраняет размытие, но не восстанавливает утраченные детали.

2.2.4 Алгоритм сравнения невязки

Метод вычисления невязки – является специфической частью схемы оценки качества алгоритмов. В качестве меры может быть выбрана величина структурности сигнала, невязка между отношениями сигнал\шум и т.д. В некоторых случаях сложно определить наиболее “правильный” алгоритм вычисления невязки, поэтому приходится характеризовать анализируемый алгоритм с помощью разных методам определения невязки.

В работе применен один из простейших алгоритмов вычисления невязки между изображениями, блок-схема которого изображена на Рис 2.5 b. Цель его работы – получить численное значение, характеризующее несоответствие между кодовыми описаниями.



a) b)

Рисунок 2.5 — Блок схема алгоритма сравнения (a) и пример изображения поля невязок (b)

Поле невязок E строится как модуль разности между бинарными изображениями в каждой точке: $E(i,j) = |K_{n+1}(i,j) - K_n(i,j)|$. Интегральное поле невязок E_{filt} вычисляется с помощью операций свертки с однородной круглой маской, нормированной на 1. Для определения областей с наиболее сильными изменениями выполнялась бинаризация поля невязок с порогом $T = {N}/{2\pi R^2}$, где параметр N определяет пороговое число рассогласованных пикселей. Ниже этого числа области сравниваемых изображений считаются совпадающими, выше – не совпадающими. Расчеты выполнены для маски с радиусом R=5 и N=30. Окончательная оценка величины невязки на всем поле невязок E_{filt} определялась из формулы $V = \sum_{i,j} E_{filt}(i,j)/S$, где S – площадь поля E_{filt}.

2.2.5 Алгоритм восстановления изображения

Восстановление реализовано методом наложения изотропных гауссовых ядер на позиции, подсвеченные бинарной маской. Формально такую операцию можно представить как свертку гауссового ядра с набором δ - функций: $I_{rec}(r) = \int_D M_\delta(r^|) \cdot K(r - r^|)dr^|$. Поле M_δ в таком описании получается из бинарного изображения M(r) заменой всех 1 на δ - функции.

Ядро $K(r) = A \cdot e^{-\frac{r^2}{2\sigma^2}}$ применялось с параметром $\sigma = 1.3$. Параметр $A = {}^{1}/_{4\pi\sigma^2}$ выбирался так, чтобы объем ядра был единичным. Синтезированное изображение, полученное таким методом, содержит различные градации серого и может использоваться для дальнейших преобразований.

2.2.6 Оценка качества алгоритмов бинаризации

Схема эксперимента следующая: входное изображение отпечатка пальца (в градациях серого) подается на выбранный фильтр, затем проходит цикл кодирования-восстановления, как описано ранее. Здесь этапом кодирования является простая пороговая бинаризация, а этапом восстановления – процедура восстановления изображения из бинарной маски. После восстановления сигнал (теперь снова в градациях серого) возвращается на вход того же фильтра – начинается следующий цикл. Проводится несколько итераций такого циркулирования сигнала через фильтр; на каждом шаге после бинаризации сохраняется копия кодового изображения. Затем для каждой пары последовательных итераций (например, 1-й и 2-й, 3-й и 4-й) сравниваются бинарные кодовые описания и вычисляются поля невязок и интегральные оценки, как описано выше. В итоге для каждого фильтра собираются данные о величине невязки на первых итерациях и о характере ее изменения. Эти данные позволяют судить об эффективности фильтра: интуитивно, если фильтр хорошо справляется с шумами, то уже после первой итерации разница между кодами исходного и восстановленного изображения (т.е. исходного шума и шума после фильтрации) будет небольшая. Также важно, насколько быстро уменьшаются невязки при последующих итерациях: это характеризует способность фильтра доводить сигнал до стационарного состояния (полностью очищенного изображения).

Выяснение поведения невязки в различных по качеству частях изображения проводилось по схеме обратной задачи. На базе пассивного фильтра создан алгоритм, определяющий области, где локальные дисперсия и

среднее не входят в заранее заданные диапазоны. Опытным путем установлены оптимальные, для использованной базы отпечатков, значения этих диапазонов. В результате получилась маска, которая разделяла изображение на две области, условно называемые “плохой” и “хорошей”. Подсчитывая значение невязки в каждой из областей, можно установить формализованную зависимость между невязкой и качеством изображений.

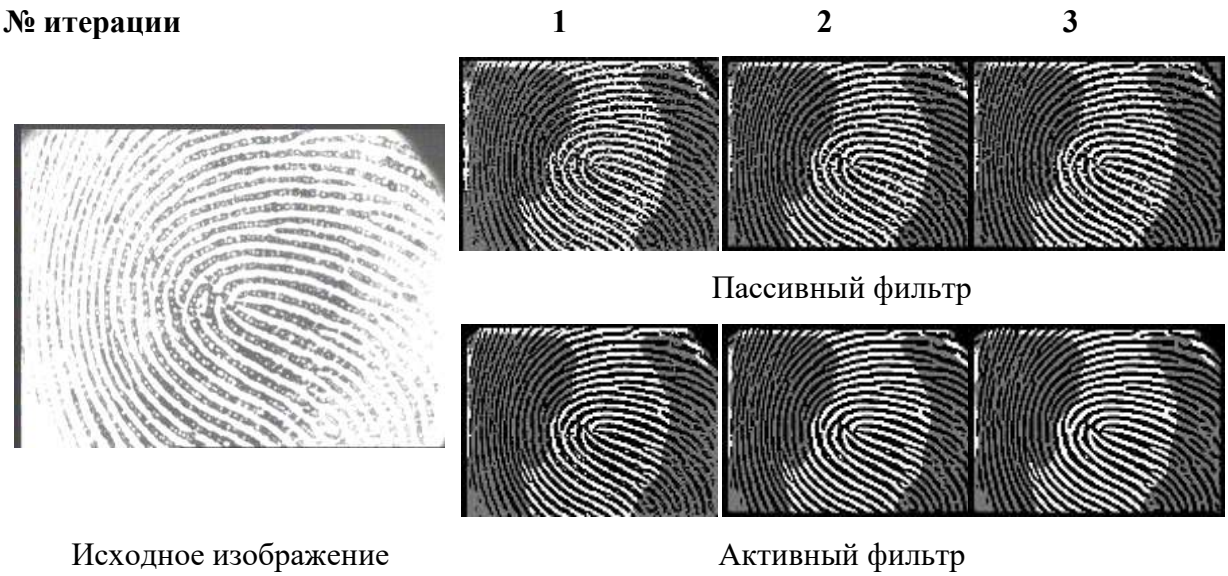


Рисунок 2.6 — Иллюстрация обработки первых трех итераций для пассивного и активного фильтров с маской областей

Рисунок 2.6 представляет собой иллюстрацию работы системы на примере первых трех итераций для двух исследуемых фильтров. С каждого этапа работы сняты сэмплы изображений. На бинарный код сигнала наложена маска различия по качеству частей (темная область – “плохая”, светлая – “хорошая”).

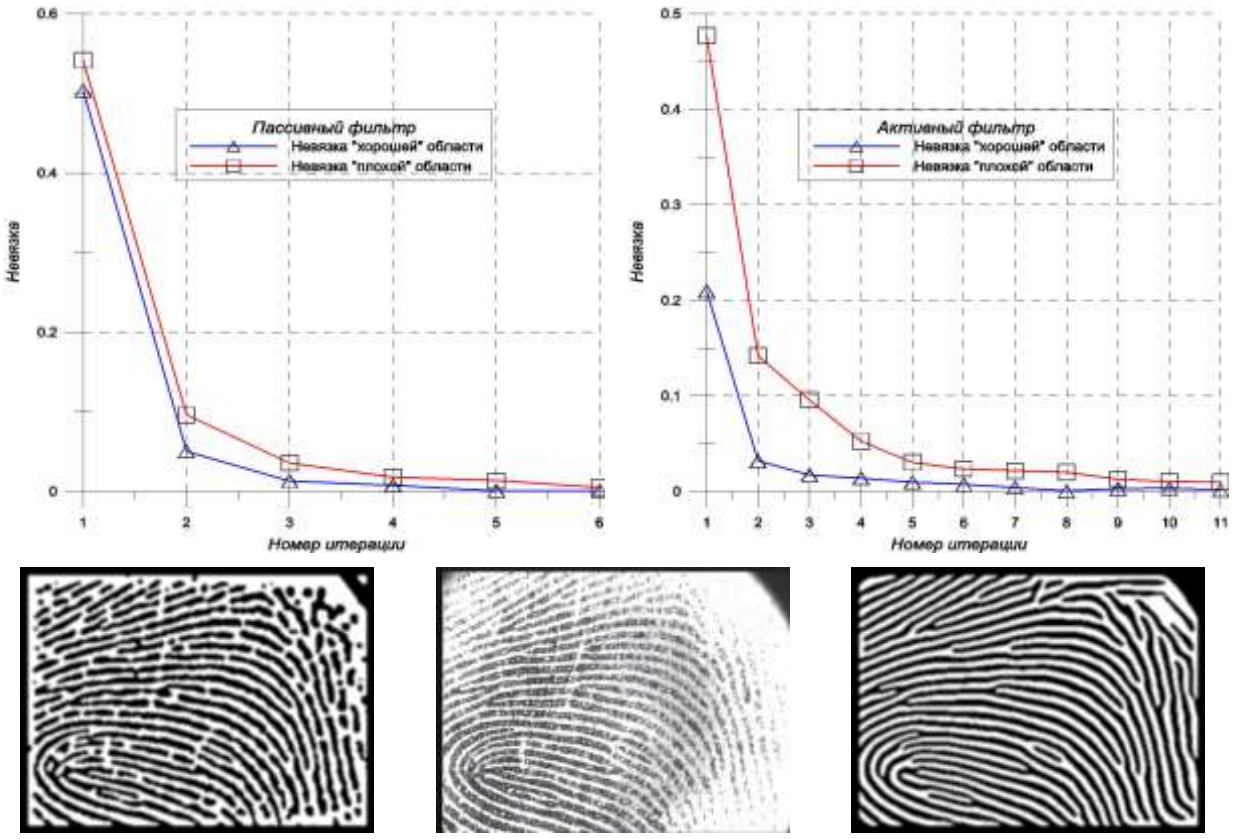


Рисунок 2.7 — Зависимость невязки от номера итерации и стационарные изображения для двух рассматриваемых фильтров

На Рисунке 2.7 представлена динамика изменения невязки в зависимости от итерации сигнала. Как видно, исходная невязка (после первой итерации) в плохих областях значительно выше, чем в хороших – что ожидаемо, поскольку шумные фрагменты изначально хуже восстанавливаются. Однако к третьей-четвертой итерации ошибки существенно падают для обоих фильтров. Интересно, что пассивный фильтр обеспечивает более резкое снижение невязки уже к 3-ей итерации, особенно в плохой зоне (с 0.55 до 0.078), тогда как у активного к 3-4 итерации в плохой зоне остается ~0.1. На первой итерации активный фильтр имел несколько меньшую невязку, чем пассивный (0.354 против 0.551), что можно интерпретировать так: активный фильтр с самого начала лучше фильтрует шум (меньше ошибка после 1 итерации, особенно заметно в хорошей зоне: 0.28 против 0.43), но из-за того, что он привносит новые элементы в изображение (дополняет линии), требуется больше итераций, чтобы

полностью стабилизировать картинку. Пассивный же фильтр только убирает часть сигнала (низкочастотную составляющую), поэтому уже после одной итерации система близка к стационарности по «хорошим» зонам (ошибка 0.4306 падает до 0.0401 – фактически изображение стало почти бинарным идеальным в этих местах). В «плохих» зонах у пассивного тоже резкое падение (0.5507 до 0.0778), то есть он очень быстро устранил большую часть помех. Но при этом пассивный фильтр не может восстановить оборванные гребни – его стационарное изображение будет просто содержать разрывы. Активный фильтр, напротив, постепенно генерирует недостающие фрагменты линий, вследствие чего полная стабилизация (невязка→0) требует больше итераций. И для активного, и для пассивного фильтра после достаточного числа итераций удается получить стационарное изображение (ошибка обнулится). Существование стационарного решения свидетельствует о корректности реализации фильтра (алгоритм не накапливает ошибку бесконечно). Это важно, например, при отладке новых фильтров – если бы адаптивная ячейка не смогла выйти на нулевую невязку, значило бы, что фильтр искажает информацию необратимо

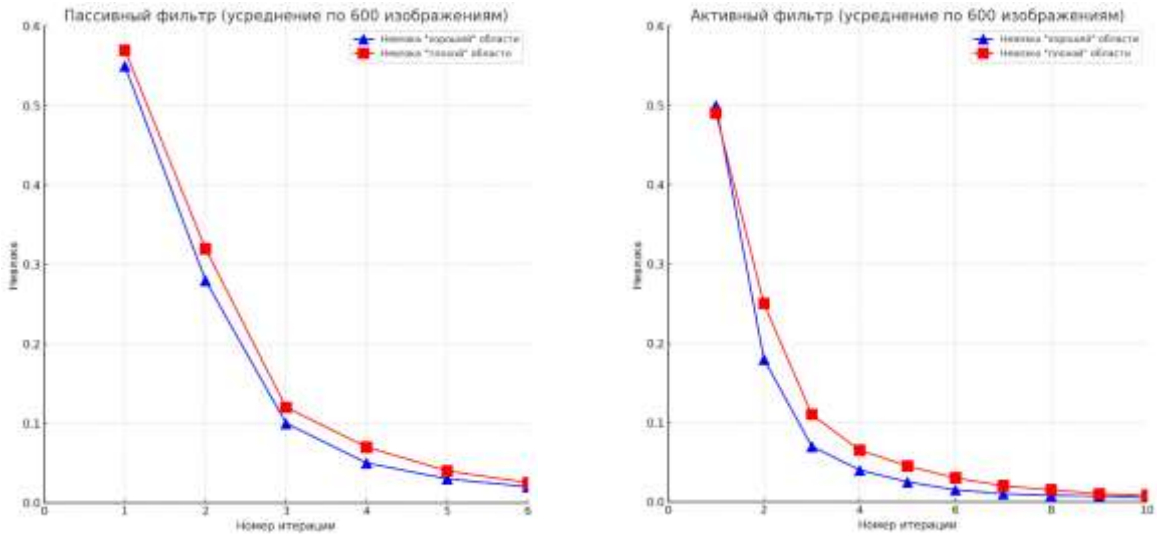


Рисунок 2.8 — Зависимость средних невязок различных по качеству областей для 600 отпечатков

Рисунок 2.8 получен усреднением невязок по 600 изображениям отпечатков, полученным от 60 различных пальцев. Здесь можно отметить закономерность: в «плохих» (низкокачественных) областях величина невязки изначально выше и убывает более быстрыми темпами, чем в «хороших. Это объясняется тем, что в шумных частях фильтр проводит больше изменений – либо чистит шум (что сразу сильно уменьшает ошибку, как у пассивного), либо дорисовывает детали (что сначала может даже увеличивать различия, но затем приводит к существенным улучшениям, как у активного). Данный эффект подтверждает возможность формализованной оценки качества изображения по динамике невязки: наблюдая, как быстро падает ошибка в разных участках, система может идентифицировать самые проблемные места (те, где ошибка особенно велика на первых итерациях) и, к примеру, сигнализировать о необходимости пересканировать палец или применить дополнительную локальную обработку.

Главный же результат экспериментов по сравнению фильтров – определение, какой алгоритм лучше для данного изображения [14]. С точки зрения критерия качества восстановления по невязке, то активный фильтр предпочтителен, если корректно подобрать его параметры. Интуитивно это объяснимо: активный фильтр дает более информативное изображение - реконструированный рисунок без шумов и разрывов - чем пассивный. Поэтому адаптивная система выберет активный фильтр как основной инструмент улучшения отпечатка. Однако замечено, что при сильных отклонениях параметров (например, слишком большая степень усиления у активного фильтра) его поведение может становиться менее стабильным, в то время как пассивный более робастен к выбору параметра – просто меняется степень фильтрации.

Адаптивная ячейка не только выбирает между разными алгоритмами, но и помогает определять качество входного сигнала. Она выполняет роль мета-алгоритма принятия решения: наблюдая за “мотивационным сигналом” – динамикой невязки [30]. В следующем разделе будет экспериментально

продемонстрировано, что использование предложенной схемы и простое сравнение изменения невязки на первых двух итерациях приводит к повышению точности распознавания и классификации отпечатков пальцев.

2.3 Классификация особых точек с помощью адаптивного выбора алгоритма бинаризации

В каждом отпечатке пальца выделяются особые точки — ключевые элементы папиллярного узора, такие как ядра (центры завитков) и дельты (точки расхождения гребней), которые служат устойчивыми опорными ориентирами при сопоставлении изображений. Классификация особых точек является важной задачей в области автоматического распознавания отпечатков пальцев, поскольку позволяет сократить пространство поиска совпадений, сгруппировать изображения по структурным признакам и повысить точность идентификации.

Традиционные алгоритмы обработки и выделения особенностей изображения чувствительны к качеству бинаризации, которая зависит от условий съёмки и качества исходного изображения. Применение одного фиксированного метода бинаризации (например, пассивного фильтра с глобальным порогом) не всегда позволяет получить стабильные результаты. В связи с этим возникает необходимость в разработке более универсального подхода [2].

Настоящий раздел посвящён исследованию эффективности применения адаптивного выбора алгоритма бинаризации, который позволяет автоматически переключаться между пассивными и активными методами фильтрации в зависимости от локальных характеристик изображения. Адаптивная система оценивания качества бинаризации на основе анализа динамики невязки обеспечивает более стабильное и информативное выделение особых точек, что, в свою очередь, приводит к повышению надёжности классификации отпечатков пальцев.

2.3.1 Извлечение особых точек из отпечатка пальца

Выделение особых точек на изображении отпечатка позволяет задать систему координат, привязанную к отпечатку, и служит основой для классификации или выравнивания отпечатков. В использованном методе извлечение особых точек выполняется на основе анализа поля направлений папиллярного узора [188]. Сначала рассчитывается *поле ориентировочных векторов* – в каждом локальном фрагменте изображения определяется среднее направление линий. Для вычисления направления используется градиентный подход: изображение сглаживается фильтром, затем для каждого пикселя вычисляются частные производные $G_x(i,j)$ и $G_y(i,j)$ - градиенты по осям x и y для пикселя (i,j). В окрестности размером $W \times W$ вокруг точки рассчитываются усреднённые величины $S_x = \sum G_x^2, S_y = \sum G_y^2, S_{xy} = \sum G_x G_y$, которые затем сглаживаются билинейным фильтром для устранения локальных шумовых отклонений. После этого направление θ доминирующей линии определяется как угол, удваивающий ориентацию градиента:

$$\theta = \frac{1}{2} \arctan \frac{2\,S_{xy}}{S_x + S_y - S_{xy}}. \qquad (2.2)$$

Детекция кандидатов в особые точки проводится путём анализа индикатора кривизны ориентации (топологического индекса) в замкнутой окрестности. Использован метод вычисления индекса Пуанкаре [43], суть которого заключается в подсчёте суммарного поворота вектора направления θ при обходе замкнутого контура вокруг предполагаемой особой точки. Формально индекс Пуанкаре можно определить как приращение угла ориентации вдоль замкнутого маршрута C вокруг точки (i,j) по полю ориентации (2.2), нормированного на 2π:

$$\Delta\Psi(i,j) = \frac{1}{2\pi} \sum_{(x,y)\in C} \Delta\left(\theta(x_{\text{next}}, y_{\text{next}}), \theta(x,y)\right) \qquad (2.3)$$

, где $\Delta(\theta_1, \theta_2)$ – разность двух направлений, нормированная в диапазоне $[-\pi, \pi]$.

На практике контур C выбирается квадратным, разделенным на блоки 3×3, и суммирование выполняется по 8 соседним ячейкам вокруг центра [189]. Если вычисленный индекс $\Delta\Psi$ по модулю близок к 0.5, то точка распознаётся как особая: положительное значение (+0.5) соответствует ядру, а отрицательное (-0.5) – дельте [189]. В случае $\Delta\Psi \approx 1$ можно говорить о наличии двойного ядра (например, отпечаток типа «двойная петля»), хотя такие случаи редки. Для повышения надёжности исключаются ложные срабатывания, возникающие из-за локальных искажений: применяется пороговая фильтрация по величине локальной дисперсии направления и морфологический анализ – удаление тесно расположенных пар ядро-дельта, не характерных для реальных отпечатков. После этого координаты оставшихся особых точек принимаются в качестве результата алгоритма обнаружения. Визуализация процесса детектирования особых точек представлена на Рис. 2.9.

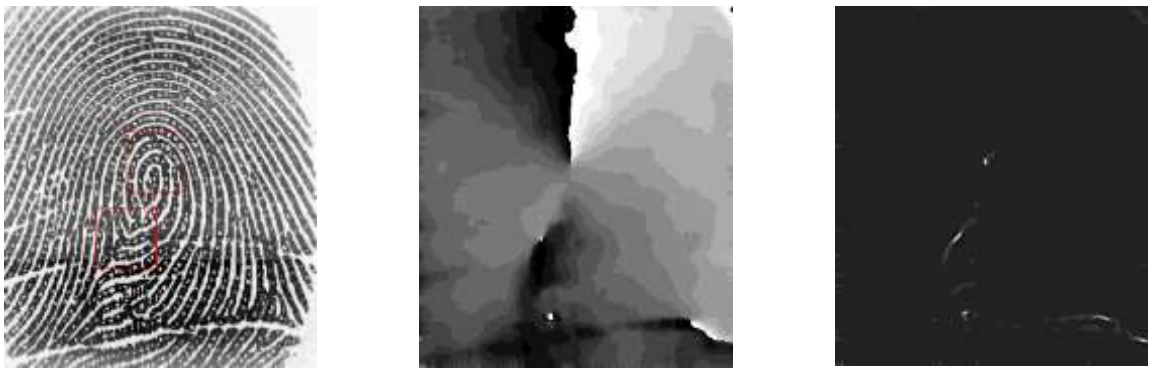


Рисунок 2.9 — (Слева направо) Исходное изображение с отфильтрованными особыми точками; поле направлений (2.2) ; поле индексов Пуанкаре (2.3)

2.3.2 Извлекаемые признаки

Каждой найденной особой точке вычисляется набор количественных признаков, описывающих окружение точки. В работе использован подход гистограмм-ориентированного окружения особой точки: вокруг каждой особой точки выделяется фиксированная окрестность радиусом R,

разделённая на M секторов, равных по угловому охвату. Для каждого сектора вычисляется среднее направление папиллярных линий внутри него с перекрытием, равным четверти сектора; таким образом, особая точка характеризуется упорядоченным набором из M средних направлений $\{\varphi_1, \varphi_2, \dots, \varphi_M\}$, пример гистограммы приведен на Рис. 2.10. Среднее направление для сектора особой точки вычисляется с помощью градиентов, рассчитанных на бинаризованных изображениях отпечатков. Для удобства дальнейших вычислений ориентации представляются в векторной форме – через значения синуса и косинуса:

$$\boldsymbol{f} = [cos\,\varphi_1, sin\,\varphi_1, cos\,\varphi_2, sin\,\varphi_2, \dots, cos\,\varphi_M, sin\,\varphi_M] \quad (2.4)$$

, где $\boldsymbol{f}$ – результирующий вектор признаков фиксированной размерности $2M$.

Такое кодирование устраняет разрывы на границе $0°/180°$ и обеспечивает инвариантность признаков к малым поворотам фрагмента отпечатка. Заметим, что выбор начального угла определяется ориентацией изображения, а не ориентацией ядра, что необходимо учитывать при сравнении признаков. Но упорядоченный набор углов φ сохраняет свои свойства при циклическом сдвиге.

Помимо направлений, вектор признаков может дополняться другими параметрами окрестности особой точки – например, распределением толщины линий, контрастностью изображения, локальной частотой папиллярного узора и др. Однако в данной работе основное внимание уделяется именно информации о направлении гребней вокруг сингулярной точки, как наиболее устойчивой и информативной для целей классификации. В дальнейшем использовалась гистограмма, построенная из 16 секторов (угол сектора 22.5^0) на площади изображения 50x50 точек (Рис 2.10).

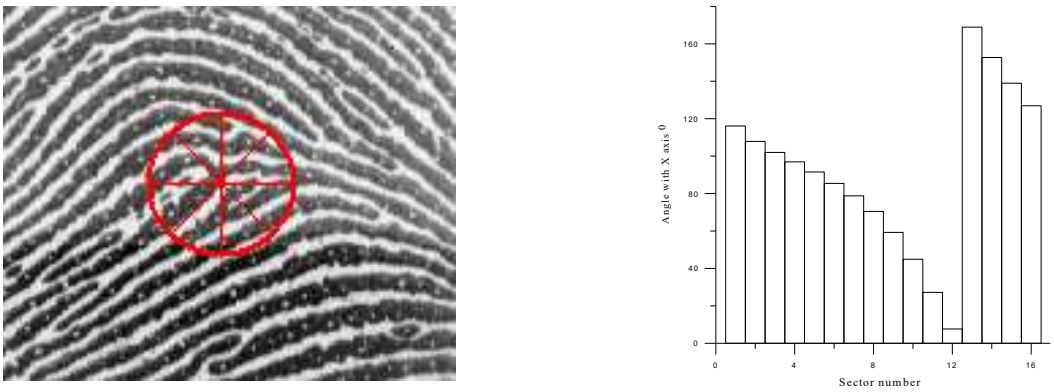


Рисунок 2.10 — Пример выбора секторов особой точки и гистограмма направлений

2.3.3 Метрика расстояния между особыми точками

После определения признакового вектора для каждой особой точки необходимо ввести меру близости (или расстояния) между двумя такими векторами. В работе в качестве меры сходства выбрана косинусная мера между двумя векторами признаков. Она удобна тем, что не зависит от масштаба векторов и фактически отражает нормированную корреляцию между признаками двух точек. Учитывая необходимость определения ориентации и свойство циклического сдвига признаков, косинусное расстояние определяется как:

$$D(\boldsymbol{f}_i, \boldsymbol{f}_j) = 1 - \max_k \frac{\langle \boldsymbol{f}_i \cdot S(\boldsymbol{f}_j, k) \rangle}{\|\boldsymbol{f}_j\| \|\boldsymbol{f}_j\|} \quad (2.5)$$

, где $S(\boldsymbol{x}, k)$ – оператор циклического сдвига на k углов вектора **x**; $\langle \boldsymbol{x}, \boldsymbol{y} \rangle$ скалярное произведение.

Косинусная мера (2.5) учитывает периодичность и сравнивает направления по существующей между ними разности углов. Кроме того, такая нормированная метрика сглаживает различия в контрасте или яркости изображений, делая систему более устойчивой к несущественным вариациям условий съёмки.

2.3.4 Методика сравнительного анализа алгоритмов бинаризации

Для оценки эффективности адаптивного выбора алгоритма бинаризации разработана методика сравнения, которая предусматривает разбиение данных на обучающую и тестовую выборки и вычисление показателей ошибок I и II рода (False Accept/Reject Rate) при классификации особых точек. Алгоритм состоит из нескольких этапов:

2.3.4.1 Извлечение особенностей

Из каждого изображения отпечатка пальца выделяются координаты особых точек по описанному выше алгоритму, и каждой особой точке присваивается метка класса (ID) – идентификатор конкретного отпечатка, которому эта точка принадлежит. Затем для обработанного одним из алгоритмов фильтрации, бинаризированного изображения отпечатка строится признаковый вектор (2.4) и таким образом, формируется набор прецедентов $\{\boldsymbol{f_n^A}, \mathrm{ID}_n\}$ – , где $\boldsymbol{f_n^A}$ вектор признаков n-й особой точки для алгоритма A, а ID_n – класс этой точки.

2.3.4.2 Формирование обучающей и тестовой выборок

Для каждого класса ID (конкретного отпечатка пальца) все его особые точки разбиваются случайным образом на две равные подвыборки: одна используется для обучения (эталонный набор точек данного отпечатка), другая – для тестирования. Такое парное разбиение выполнено для всех K классов.

2.3.4.3 Классификация точек по ближайшему соседу

Для каждой особой точки из тестовой выборки находится ближайшая особая точка из обучающей выборки с помощью выбранной метрики D (2.5). Если класс ближайшей особой точки из обучающего набора совпадает с истинным классом тестовой точки, то классификация этой точки считается успешной; в противном случае фиксируется ошибка. По результатам

сопоставления всех тестовых особых точек подсчитываются суммарные показатели по точности классификации.

2.3.4.4 Оценка FAR/FRR для выбранного алгоритма

Помимо подсчёта доли ошибочно классифицированных точек, производится более тонкий анализ – вычисляются характерные кривые FAR и FRR, отражающие зависимость ошибок I и II рода от допустимого порога сходства. Для этого для каждой тестовой точки дополнительно регистрируются две величины: D_{in} – расстояние до ближайшей особой точки из обучающей выборки того же класса, и D_{out} – расстояние до ближайшей точки другого класса из обучающего набора. Меняя порог T на максимальное допустимое расстояние, можно построить графики показателей ошибок:

$$\begin{aligned} \text{FAR}(T) &= \frac{1}{N}\sum_{i=1}^{N} I\left(D_{\text{out}}^{(i)} < T\right) \\ \text{FRR}(T) &= \frac{1}{N}\sum_{i=1}^{N} I\left(D_{\text{in}}^{(i)} > T\right) \end{aligned} \tag{2.6}$$

, где I – индикаторная функция, N – количество тестовых примеров

Изменяя параметр T, получаем семейство точек $\left(\text{FAR}(T), \text{FRR}(T)\right)$ и характерные кривые ошибок. Точка пересечения этих кривых соответствует равенству вероятностей ошибок I и II рода и называется точкой равных ошибок (Equal Error Rate, EER). Именно EER часто служит интегральным показателем качества алгоритма: чем меньше значение EER, тем лучше в целом разделяются классы отпечатков.

2.3.4.5 Сравнение трёх подходов к бинаризации

Описанные выше шаги (1–4) выполняются отдельно для трёх различных методов предварительной бинаризации отпечатков: “пассивного”, “активного” и “адаптивного”. Пассивная бинаризация представляет собой классический подход с фиксированными параметрами фильтрации и порогового преобразования изображения, описанная в разделе 2.1.3. Активная бинаризация использует более нейроноподобный фильтр, способный

восстанавливать разрывы и усиливать контраст в локальных областях изображения (Раздел 2.1.2). Наконец, адаптивный выбор алгоритма бинаризации реализуется за счёт специальной процедуры сравнения качества фильтрации: исходное изображение обрабатывается пассивным и активным алгоритмами параллельно, и на основе сравнения первых двух итераций невязки выбирается результат того алгоритма, который показал наилучшую скорость сходимости. Выбор осуществляется независимо для каждой особой точки, в ее окрестности, размером 50x50 пикселей по величине разности невязки на первой и второй итерации.

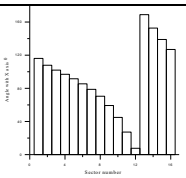
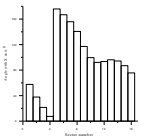
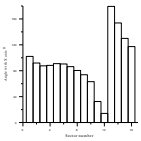
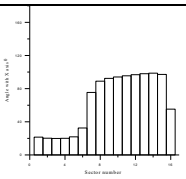
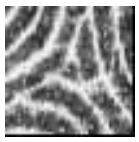
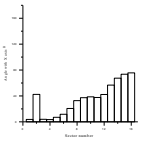
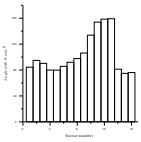
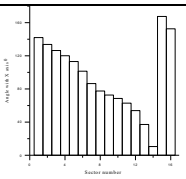
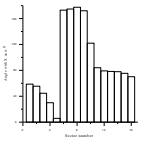
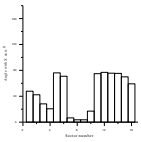
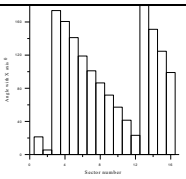
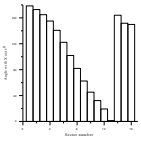
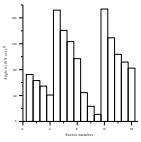
2.3.4.6 Повторная выборка и усреднение результатов

Чтобы исключить случайное влияние конкретного разбиения на обучение и тест, эксперимент с шагами (2–5) повторяется 10 раз для различных случайных разбиений исходных данных на обучающую и тестовую части. Далее результаты усредняются по всем запускам, вычисляются средние значения метрик (доля верно классифицированных особых точек, EER и т.д.) и их статистические отклонения.

2.3.5 Условия и результаты эксперимента

Для сравнительного эксперимента использована база отпечатков пальцев, включающая 600 изображений отпечатков пальцев разрешением 300×300 пикселов, полученные с помощью сенсора Veridicom. База содержит отпечатки 60 различных пальцев 6 человек; для каждого пальца выполнено по 10 снимков разного смещения и поворота. Для всех изображений проводятся единые процедуры предобработки: из каждого изображения извлекаются особые точки и применяется бинаризация с помощью трех методов, согласно методикам, описанным выше.

Таблица 2.1 — Визуализация некоторых классов особых точек в базе

Id	Гистограмма кластера	Ближайший пример	Гистограмма примера	Дальний пример	Гистограмма примера
4					
24					
16					
52					

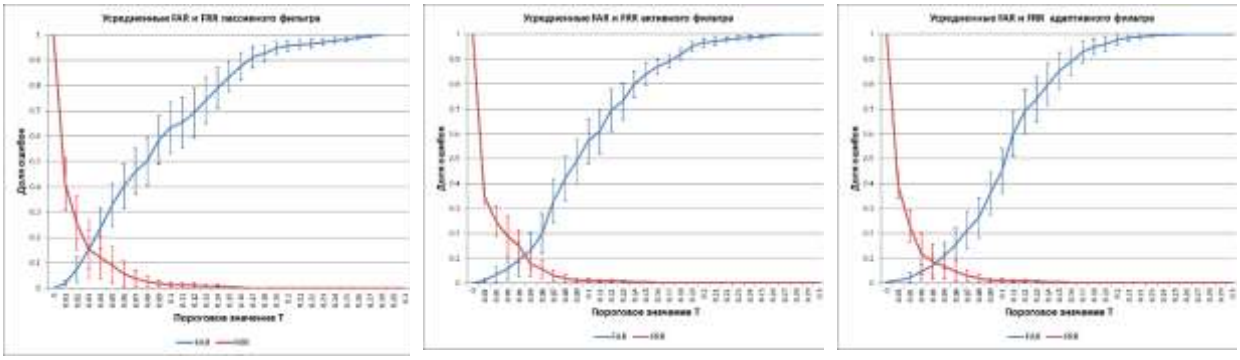
Всего из базы было получено 1411 особых точек. Каждой точке присвоен класс – от 1 до 60, соответствующий конкретному пальцу. Для визуализации, были построены гистограммы центров кластеров особых точек для каждого класса и определены ближайший и дальний пример особой точки среди всех примеров базы (см. Таблица 2.1). Затем реализована процедура, описанная в Разделе 2.3.4 и для каждого из трёх алгоритмов фильтрации повторена 10 раз (при разных разбиениях), после чего вычислены средние показатели распознавания и усредненные характеристики FAR/FRR.

Таблица 2.2 — Сравнение доли ошибок распознавания классов в зависимости от алгоритма фильтрации

Ошибка	**Пассивный**	**Активный**	**Адаптивный**
Среднее	0.134	0.097	**0.072**
Отклонение	0.031	0.02	0.023

Полученные экспериментальные результаты подтвердили преимущество адаптивного подхода к бинаризации в среднем (Таблица 2.2). Несмотря на то, что активный фильтр выбирался в почти 75% случаев,

адаптивный выбор фильтра позволял достичь улучшения не только пассивной фильтрации, но и активной. Характеристики FAR/FRR показывают похожую картину (см. Рис. 2.11). При пассивной бинаризации средняя точка равных ошибок составила EER≈15%, активная бинаризация обеспечила EER≈11%, тогда как адаптивный выбор алгоритма позволил снизить это значение до EER≈8%. Иными словами, адаптивная система почти вдвое снизила совокупную ошибку по сравнению с классическим. Также удалось уменьшить разброс ошибок между разными отпечатками: наибольшие значения FAR и FRR по классам при адаптивном подходе оказались ниже, чем при любом фиксированном способе обработки. Это указывает на определенную универсальность и устойчивость алгоритма адаптивного выбора к разнообразным искажениям отпечатков.



a) EER=0.15 b) EER=0.11 c) EER=0.08

Рисунок 2.11 — Far/Frr характеристики алгоритма классификации при использовании a) пассивного b) активного c) адаптивного алгоритмов фильтрации

2.4 Выводы

В основе рассмотренного подхода лежит концепция адаптивной распознающей ячейки [14]. Эта ячейка функционирует в цикле «кодирование–восстановление–оценка», непрерывно анализируя качество обработки. На каждом шаге вычисляется невязка (разница между оригинальным и восстановленным изображением), которая служит сигналом обратной связи. Если невязка велика, система автоматически подстраивается – корректирует

параметры или переключается на другой алгоритм. Показано, что такой биоморфный принцип позволяет динамически улучшать распознавание: изображение «проясняется» итеративно, и признаки, не выделенные сразу, могут быть обнаружены на следующих циклах адаптации.

В работе рассмотрены два метода фильтрации отпечатков пальцев: (1) *нейроподобный активный фильтр* – нелинейный алгоритм, способный восстанавливать отсутствующие фрагменты папиллярного узора и локально усиливать контраст; (2) *классический пассивный фильтр* – линейный метод, устраняющий шумы и фон (размытость), выравнивающий общую яркость. Активный фильтр эффективнее на сильно зашумленных или размытых изображениях, тогда как пассивный быстрее и достаточен для четких отпечатков.

Ключевым критерием качества служит динамика изменения невязки на первых итерациях обработки и на основе этого адаптивно выбирает оптимальный алгоритм для каждого образца. Таким образом, достигается сочетание преимуществ обоих подходов: в каждой локальной области отпечатка применяется тот метод, который там более результативен.

Предложенная адаптивная схема успешно применена к задаче классификации особых точек отпечатка пальца. Адаптивный алгоритм выбирает фильтр в окрестности каждой особой точки, анализируя невязку на первых двух итерациях кодирования-восстановления отпечатка. Экспериментально показано, что такая адаптация фильтра повышает точность: по сравнению с фиксированной обработкой качество классификации особых точек по методу ближайшего соседа заметно улучшается.

Адаптивный метод продемонстрировал существенное снижение ошибок распознавания отпечатков по сравнению с традиционными подходами. В среднем система, автоматически переключающая фильтры, смогла почти вдвое снизить совокупную ошибку классификации относительно классической пассивной схемы.

ГЛАВА 3. КОМИТЕТЫ СЛАБЫХ КЛАССИФИКАТОРОВ В ЗАДАЧАХ КЛАССИФИКАЦИИ ОБЪЕКТОВ НА ИЗОБРАЖЕНИИ

3.1 Введение: коллективное распознавание

Коллективное распознавание — это направление, изучающее способы объединения решений множества классификаторов, каждый из которых независимо оценивает один и тот же объект, ситуацию или образ. Основная задача — формирование согласованного коллективного решения на основе индивидуальных суждений классификаторов. Такой подход используется во все большем количестве прикладных и теоретических задач, включая задачи с высокой размерностью, разнообразием типов признаков (числовые, булевы, изображения и др.) и сложной структурой данных [8].

Главная идея объединения классификаторов заключается в уменьшении сложности решения задачи за счёт декомпозиции, а также в увеличении общей компетентности системы через использование специализированных классификаторов. Аналогично системе государственного управления, где различные министерства и ведомства отвечают за свои области (экономика, здравоохранение, оборона и т.д.), в ансамбле классификаторов каждый участник может быть наилучшим в своей "области ответственности" — то есть, в подмножестве признакового пространства, где он показывает наибольшую точность. Даже слабые классификаторы, ошибающиеся по-разному, могут улучшить общее качество за счёт дополнительной информации, предоставляемой ими в независимых областях признакового пространства. Это объясняется тем, что множества неверно классифицированных объектов разными алгоритмами могут не пересекаться [112].

Кризис в развитии индивидуальных методов классификации, наступивший в 1980-х годах, стал одним из толчков к развитию коллективного распознавания. При наличии множества различных подходов — статистических, синтаксических, структурных, нейросетевых и др. — стало

ясно, что не существует одного универсального метода. Это осознание, а также рост практической потребности в более точных системах, привели к идее объединения нескольких подходов в одну согласованную структуру.

Существуют различные стратегии объединения классификаторов: от простых правил голосования до вероятностных схем, нейросетевых аггрегаторов и методов мета-обучения. Наиболее распространены:

- вероятностные методы (основаны на правилах Байеса),
- методы, использующие иерархические обобщение (мета-обобщение),
- подходы с оценкой локальной компетентности,
- нейросетевые методы агрегации.

Одним из первых теоретических результатов в этой области стала теорема Кондорсе (1785 г.) [56] о присяжных, которая показала, что коллективное решение группы экспертов может быть более точным, чем мнение отдельного участника, при условии независимости и качества индивидуальных решений.

В 1952 году Фикс и Ходжес [68] представили вероятностный метод, основанный на принципе минимизации риска ошибок при классификации, позднее ставший прообразом метода ближайших соседей. Их идея заключалась в сравнении плотностей распределения вероятностей классов в окрестности исследуемой точки, с последующим отнесением этой точки к тому классу, для которого вероятность ошибки минимальна. Этот метод предполагал наличие априорной информации о распределениях, однако впоследствии послужил основой для создания эвристического метода k ближайших соседей, который заменил оценки плотности выборочным поиском ближайших обучающих примеров. Работа Фикса и Ходжеса ознаменовала начало статистически обоснованных подходов к распознаванию образов и заложила основы будущих методов, использующих локальные оценки и непараметрические процедуры.

В 1958 году Розенблат [182] разработал персептрон, одну из первых моделей нейросетевой архитектуры, способную к обучению. Персептрон представляет собой простую однослойную сеть, в которой входные данные проходят через линейное суммирование с весами, а затем подаются на функцию активации (обычно пороговую). Обучение модели происходит путём корректировки весов с использованием правила обучения на основе ошибки. Несмотря на свою простоту, персептрон положил начало целому направлению в машинном обучении — обучению с учителем для нейросетевых моделей. Его можно интерпретировать как систему, реализующую взвешенное голосование входов (или нейронов), где итоговое решение принимается на основе согласованного превышения порога, что делает его одним из прообразов коллективного принятия решений в нейронных архитектурах.

В это же время интерес исследователей привлекла работа М.М. Бонгарда [4], который предложил оригинальную схему логико-эвристического распознавания образов. Суть схемы заключалась в последовательном применении нескольких решающих признаков к объекту. Каждый этап уменьшал множество гипотез, соответствующих объекту, вплоть до выделения единственного класса. Таким образом, схема Бонгарда может рассматриваться как ранняя форма коллективного подхода, при котором каждый признак-решатель действует как локальный эксперт, а всё множество признаков последовательно уточняет обобщённое решение. Это стало прообразом каскадных и иерархических моделей, получивших развитие в более поздних ансамблевых методах.

В 1980–90-е годы были предложены более формализованные и практически применимые методы объединения решений, включая простое и взвешенное голосование [34], схемы консенсусного агрегирования, системы комитетов, динамический выбор классификаторов и др. [112, 122].

Значительный вклад внёс В. Д. Мазуров, который сформулировал метод комитетов для задач оптимизации и классификации [21]. В этом подходе поиск решения сводится к построению совокупности (комитета) решений системы

линейных неравенств, где каждый элемент комитета решает часть общей задачи. Теория комитетов Мазурова и коллег заложила основы ансамблевых методов и в последующие десятилетия была развита, в том числе в контексте нейронных сетей и нечеткой логики.

Дальнейшее развитие подходов к построению ансамблей классификаторов стало логичным этапом эволюции методов классификации. Это развитие обусловлено необходимостью преодоления ограничений индивидуальных классификаторов, особенно в условиях высокой сложности, разнообразия данных и недостаточной универсальности существующих алгоритмов. В результате были выделены парадигмы, описывающие основные направления построения коллективных систем, ориентированных либо на повышение точности отдельных участников, либо на максимальное покрытие всего пространства решений. В рамках дальнейшего развития методов объединения классификаторов сформировались две ключевые парадигмы:

1. **Оптимизация индивидуальных решений** — предполагается, что улучшение качества каждого отдельного классификатора приведёт к лучшему результату ансамбля. Однако на практике даже ансамбль хороших классификаторов может уступать ансамблю специализированных, но слабо коррелированных алгоритмов [173].
2. **Оптимизация покрытия** — предполагает, что для каждого объекта найдётся хотя бы один классификатор, который классифицирует его верно. Здесь важна диверсификация: классификаторы подбираются так, чтобы их ошибки не совпадали. Такие подходы реализуются в методах баггинга [45, 46], бустинга [69] и кросс-валидации.

Оптимизация покрытия, как методологическая парадигма, делится на две ключевые группы подходов:

- **Методы выбора компетентного классификатора**, в которых для каждого нового входного объекта динамически выбирается один наиболее подходящий классификатор. Оценка производится на основе локальной точности или другой меры компетентности.

Истоки этих подходов — в работах Растригина и Эренштейна [24], а также в более поздних исследованиях [102, 143, 174].

- **Методы согласованного объединения решений**, где все классификаторы рассматриваются как равноправные участники, а окончательное решение формируется с помощью обучаемых правил агрегирования. Эти правила обучаются на специальных выборках и могут учитывать контекст, характер задачи и статистические зависимости между классификаторами [52, 133, 148].

Компетентность классификатора — это мера его способности правильно классифицировать объекты в определённой области пространства признаков. С практической точки зрения, компетентность можно интерпретировать как апостериорную вероятность правильной классификации объекта данным классификатором в заданной локальной окрестности. Она определяется либо статистически, либо посредством оценки качества классификации на локальных выборках. Такая интерпретация позволяет динамически адаптировать выбор классификаторов в зависимости от объекта, подлежащего распознаванию.

Подходы, основанные на оценке компетентности, активно развиваются начиная с 1970-х годов [24]. Классическим вариантом является схема локального выбора наиболее компетентного классификатора. Согласно этой схеме, для каждого объекта распознавания выбирается тот классификатор, который ранее показал наилучшие результаты в близкой области признаков. Иными словами, каждый классификатор получает «зону ответственности», где он демонстрирует наибольшую эффективность.

Существенное развитие данного направления было получено в работе [39], где было экспериментально подтверждено, что использование локально компетентного классификатора повышает точность по сравнению с глобальными стратегиями объединения. Было показано, что даже слабые классификаторы при правильной локализации могут значительно повысить общую эффективность системы.

В работе [143] предложен подход на основе алгоритма «рефери» (referee), где отдельный управляющий алгоритм определяет степень доверия к каждому классификатору на основании его компетенции. Такой рефери может представлять собой отдельный классификатор, обученный на мета-признаках, или реализовываться как логическая схема голосования. Введённый механизм рефери позволяет гибко управлять структурой ансамбля и адаптироваться к изменяющимся условиям классификации.

Обобщение и развитие этой концепции приведено в [160], где предложена иерархическая архитектура с многоуровневой системой выбора компетентных классификаторов. На первом уровне выбирается подмножество потенциально компетентных моделей, а на втором — финальный классификатор. Такая архитектура позволяет сократить вычислительную нагрузку и повысить устойчивость к шуму. Позднейшие работы [174] вводят локальные вероятностные оценки компетентности, используя окна ближайших соседей и байесовские правила для оценки апостериорных доверий к классификаторам.

Таким образом, подходы на основе оценки компетенции обеспечивают высокий уровень адаптивности ансамбля и позволяют учитывать локальную структуру данных. Они особенно эффективны в ситуациях, где различные части пространства признаков имеют существенно различающиеся характеристики, что делает невозможным использование единой стратегии классификации.

Иерархическое обобщение (мета-обобщение, Stacked generalization) — это один из универсальных методов формирования коллективного решения, который позволяет эффективно учитывать ошибки базовых классификаторов. Концепция метода была предложена Вольпертом [185] и позднее развита рядом исследователей [51, 175]. В отличие от простых схем голосования, здесь решения базовых классификаторов не агрегируются напрямую, а подаются в качестве признаков на вход мета-классификатора (или классификатора верхнего уровня), который обучается на этих выходах, формируя финальное

решение [46]. Такая архитектура позволяет выявлять скрытые закономерности между выходами базовых алгоритмов и учитывать сложные взаимозависимости между их ошибками.

Важным моментом реализации мета-обобщения является процедура разбиения обучающей выборки. Обычно для каждого базового классификатора используется подмножество обучающей выборки, не пересекающееся с тем, на котором обучается мета-классификатор, чтобы избежать переобучения. Используются такие техники, как k-кратная кросс-валидация, при которой мета-классификатор обучается на предсказаниях, полученных в условиях, максимально приближенных к тестовой ситуации.

Преимущество данного метода — его универсальность: он не требует однородности в наборах признаков базовых моделей и допускает использование различных по природе классификаторов (например, деревьев решений, логистических моделей, нейросетей). Это делает мета-обобщения особенно эффективным при наличии гетерогенных источников информации или моделей различной структуры. Также был предложен ряд расширений, таких как каскадное (многоуровневое) обобщение, при котором формируются последовательные слои классификаторов, передающих друг другу результаты для уточнения решения. Это приближает архитектуру ансамбля к иерархическим нейронным сетям и позволяет повысить гибкость и адаптивность системы [76, 138].

Коллективное распознавание остаётся активно развивающейся областью, в которой открыты возможности для дальнейших исследований. Особый интерес представляют вопросы адаптивного формирования ансамблей в условиях неполной или изменяющейся информации [119], а также интеграция коллективных методов с современными технологиями глубокого обучения и обучением с ограниченным объёмом размеченных данных. Особенно перспективными являются методы, ориентированные на локальную оценку компетентности, а также иерархические и многоуровневые

архитектуры, позволяющие адаптировать ансамбль под конкретные условия применения [10].

3.2 Проблема асимметрии ошибок слабых бинарных классификаторов

Рассмотрим бинарный базовый классификатор (слабый классификатор), который определяется как (3.1):

$$h_t: X \to \{-1, +1\} \tag{3.1}$$

,где X — пространство признаков, (-1) и $(+1)$ — метки классов.

Одной из серьёзных проблем применения ансамблевых алгоритмов на основе бинарных слабых классификаторов является асимметрия ошибок первого и второго рода. На практике часто возникают ситуации, когда слабый классификатор обладает низким значением ошибки первого рода, но высоким значением ошибки второго рода, или наоборот. Например, классификатор может практически идеально отделять фоновые (отрицательные) объекты от целевых (положительных), показывая очень низкий FAR, при этом допуская относительно высокий уровень FRR (пропуская значительную долю положительных объектов). Стандартные алгоритмы накачки (boosting), такие как [70], ориентированы на минимизацию общей взвешенной ошибки классификатора, рассчитываемой как средняя сумма ошибок первого и второго рода. Поэтому они склонны отбрасывать такие "односторонне точные" слабые классификаторы, несмотря на то, что их можно эффективно использовать в отдельных зонах пространства признаков.

Графически данную проблему можно проиллюстрировать следующим образом (Рис 3.1):

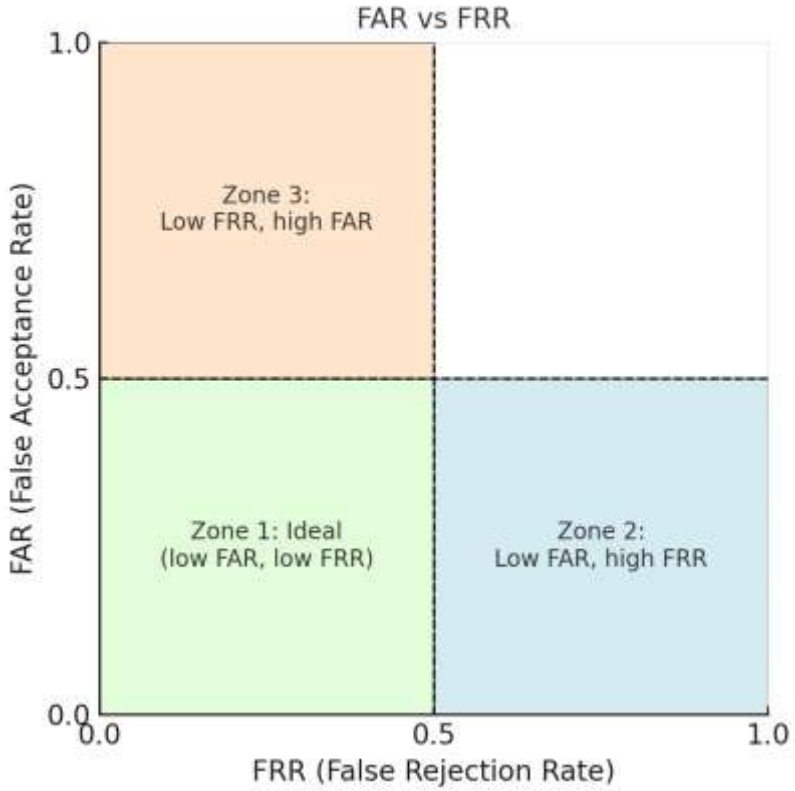


Рисунок 3.1 — Пространство показателей ошибок FAR–FRR, разбитое на три характерные области. **Зона 1** – “идеальные” классификаторы, которые охотно используются алгоритмами накачки; **Зона 2** – классификаторы, которые хорошо распознают отрицательный класс, но ошибаются в положительном; **Зона 3** – классификаторы, которые хорошо распознают положительный класс, но допускают ошибки на отрицательном

Область 1 соответствует идеальному (рабочему) случаю – классификатору с одновременно низкими значениями FAR и FRR. Область 2 (справа внизу) отражает классификатор с *низким FAR, но высоким FRR* – т.е. он почти не дает ложных срабатываний, но пропускает многие объекты положительного класса. Область 3 (слева вверху) напротив характеризует классификатор с низким FRR при высоком FAR, который редко упускает объекты положительного класса, но ценой большого числа ложных тревог. Алгоритмы накачки, такие как AdaBoost [137] фактически ориентированы только на классификаторы из области 1, отбрасывая варианты из областей 2 и 3, например алгоритм AdaBoost требует от каждого слабого классификатора

ошибки не хуже случайного угадывания – порог 50% [71]. Классификатор, сильно смещённый в одну сторону (как в Областях 2 или 3), при равных весах объектов будет ошибаться примерно в половине случаев и потому получит нулевой вес в ансамбле. Этот недостаток алгоритмов построения ансамблей по учёту асимметричных ошибок признан в литературе [137], что мотивировало появление множества модификаций бустинга для асимметричных случаев.

3.2.1 Тернарный классификатор

Для решения этой проблемы предлагается использовать тернарный классификатор, который в работах [15, 41, 120] назван «экспертом». Тернарный классификатор допускает неопределённый ответ, тем самым определяя область своей компетенции:

$$h_t: X \to \{-1, 0, +1\} \tag{3.2}$$

, где ответ 0 соответствует ситуации, когда классификатор считает объект вне зоны своей компетентности, а -1 и $+1$ — метки классов.

Такой подход позволяет использовать классификаторы, которые уверенно работают только с одним из классов, а в областях неопределённости не принимают решения. Кроме того, тернарный классификатор позволяет создавать базовые классификаторы, уверенно работающие только на подмножестве примеров, устраняя асимметрию между классами.

Тернарная схема позволяет избегать вынужденной ошибки там, где бинарный классификатор колеблется. В бинарном случае даже объекты «на границе» разделяющей поверхности будут отнесены к одному из классов, зачастую произвольно (лишь бы превышался порог), что приводит к ошибкам на пограничных случаях. Тернарный же классификатор в таких ситуациях выдаёт неопределённость, тем самым потенциально снижая количество ошибочных отнесений к классу [15]. Исключая из рассмотрения эти объекты (передавая их другому эксперту или откладывая решение), можно повысить

точность классификации на решённых случаях ценой снижения охвата (не все объекты сразу получают класс).

Введём некоторые определения:

- **Экспертная область** – это подмножество пространства объектов, в котором “эксперт” принимает уверенное решение о классе (т.е. где $(h_t(x) \neq 0)$). Соответственно, область неопределённости $(h_t(x) = 0)$ является вне компетенции данного комитета.
- Показатель **эффективности эксперта** R_3 – доля объектов, попавших в экспертную область, иными словами, доля тех объектов, для которых “эксперт” все же принял решение. Можно рассматривать эффективность как $(R_3 = 1 - \frac{N_{\text{неопред.}}}{N_{\text{общий}}})$.
- **Ошибка тернарного классификатора (эксперта)** E_3 – доля объектов, попавших в экспертную область, но, тем не менее, не верно классифицирована.
- **Ошибка бинарного классификатора** E_2 – не верно классифицированная доля объектов.
- **Экспертным комитетом** называется ансамбль классификаторов (не обязательно тернарных), реализующий правило тернарной логики.

Отметим, что введенные характеристики связаны соотношением $E_2 \leq 1 - R_3(1 - E_3)$, которое следует из того факта, что оптимальная разделяющая гиперплоскость проходит между гиперплоскостями тернарного классификатора и не все примеры, не попавшие в экспертную область, будут не верно классифицированы.

3.2.2 Экспертный комитет

Еще одним из способов задействовать «отброшенные» односторонние классификаторы является использование “зазора” классификации. Этот зазор служит формализацией области неопределённости классификатора: объекты,

попавшие в зазор, считаются *неразрешимыми* (неуверенными) и остаются без решения. Таким образом, выход такого классификатора становится тернарным – помимо классов +1 и –1 вводится третий исход 0, означающий *отказ от классификации* для случаев, в которых модель недостаточно уверена.

Вместо стандартного «сильного классификатора» предлагается использовать «экспертный комитет» – взвешенную сумму классификаторов, снабжённую двумя порогами T_1и T_2, определяющими зоны уверенных решений и зону неопределённости:

$$H(x) = \begin{cases} +1, \sum_{k=1}^{K} \alpha_k h_k(x) \geq T_1 \\ -1, \sum_{k=1}^{K} \alpha_k h_k(x) \leq T_2 \\ 0, \ otherwise \end{cases} \quad (3.3)$$

, где α_k – вес k -го эксперта, $h_k(x)$ – ответ k -го эксперта на объекте x, T_1, T_2 – пороги принятия решения.

Таким образом, экспертный комитет может откладывать принятие решения в случаях недостаточной уверенности. Отметим, что формулировка (3.3) может использовать решения не только бинарных, но и тернарных классификаторов, что освобождает ансамбль от предположения о равномерности и однородности распределения ошибок отдельных классификаторов. Для ансамбля важно, чтобы слабые классификаторы были *диверсифицированы* – имели хотя бы частично независимые ошибки и разнородные критерии принятия решений.

Рассматриваемый ансамбль (3.3) получает возможность воздержаться от решения, если суммарный «голос» недостаточно определён. Практически это эквивалентно построению *каскада классификаторов* [120]– цепочки из нескольких комитетов: если первый комитет (первый эксперт) не вынес решения (ответ 0), объект передаётся следующему и так далее, пока какой-то эксперт не классифицирует объект уверенно либо не исчерпаются все (Рис 3.2). Такая каскадная архитектура рассматривается в следующих разделах применительно к задаче детекции [11].

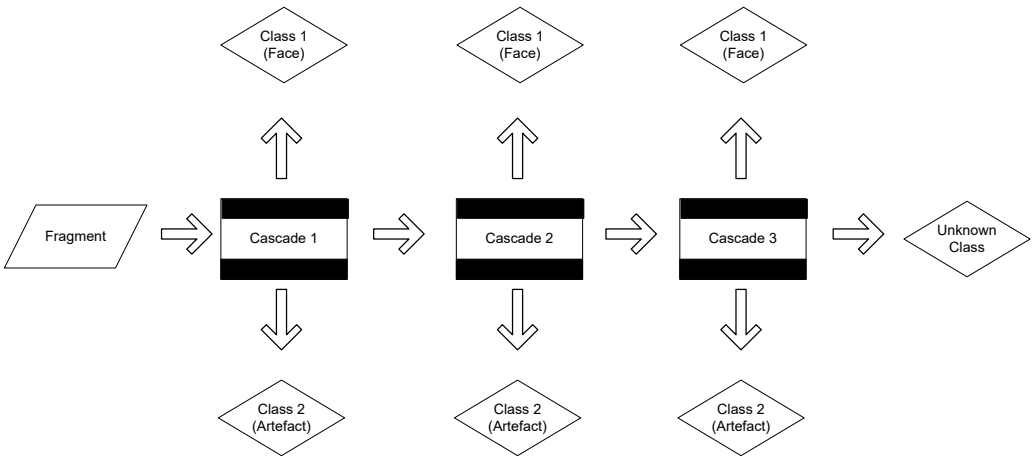


Рисунок 3.2 — Принятие решения на основе каскадов тернарной логики

3.2.3 Модифицированный для тернарных классификаторов adaboost

Алгоритм AdaBoost [69] был предложен Йоавом Фрейндом и Робертом Шапиром в 1997 году как метод, позволяющий создавать сильные классификаторы путём объединения множества простых, или слабых, классификаторов. Основной целью AdaBoost являлось повышение общей точности классификации за счёт последовательного обучения множества классификаторов, каждый из которых компенсирует ошибки предыдущих. Алгоритм оказался особенно полезным в задачах, где индивидуальные классификаторы дают лишь немного лучше случайного угадывания, и существенно повысил эффективность решения задач классификации и распознавания образов. Статья авторов также предлагает способ оценки общей ошибки ансамбля, которая оценивается как:

$$\epsilon_{final} \leq 2^K \prod_{k=1}^{K} \sqrt{\epsilon_k(1-\epsilon_k)} \tag{3.4}$$

, где ϵ_k – ошибка выбранного классификатора на k итерации, а K – количество итераций.

Это означает, что при условии, что каждый слабый классификатор имеет ошибку менее 0.5, ошибка итогового классификатора экспоненциально

уменьшается с увеличением числа итераций. Полный алгоритм Adaboost приведен в Приложении А.

Рассмотрим ошибку, которую формирует тернарный слабый классификатор, учитывая только определенные ответы. Пусть D_k – распределение весов примеров, полученной на k итерации. Тогда ошибку тернарного классификатора можно записать как:

$$\epsilon_k = \frac{\sum_{i:h_k(x_i)\neq 0} D_k(i)[y_i \neq h_k(x_i)]}{\sum_{i:h_k(x_i)\neq 0} D_k(i)} = \frac{E_k}{W_k} \tag{3.5}$$

, где E_k – суммарный вес ошибок на уверенно классифицированных объектах, а W_k – суммарный вес уверенно классифицированных объектов, y_i – метка класса для i – го примера.

Используя такое определение ошибки (3.5) возможно изменить стандартный алгоритм Adaboost (Таблица 1), который бы учитывал экспертные области отдельного тернарного классификатора [3].

Таблица 3.1 — Модифицированный алгоритм Adaboost для построения ансамбля тернарных классификаторов

1. Инициализация весов $D_1(i) = \sigma_i$.
2. Повторить для $k = 1, \dots, K$:
   a. Нормализовать веса $D_k(i)$, чтобы они образовывали распределения вероятности.
   b. Обучить эксперта $h_k(x) \to \{-1,0,+1\}$ с учетом весов каждого примера $D_k(i)$.
   c. Вычислить ошибку каждого классификатора: $\epsilon_k = \frac{\sum_{i:h_k(x_i)\neq 0} D_k(i)[y_i \neq h_k(x_i)]}{\sum_{i:h_k(x_i)\neq 0} D_k(i)}$.
   d. Выбрать классификатор h_k с минимальной ошибкой ϵ_k^{min}
   e. Рассчитать вес: $\alpha_k = \frac{1}{2}\ln\left(\frac{1-\epsilon_k^{min}}{\epsilon_k^{min}}\right)$.
   f. Обновить веса:

$$D_{k+1}(i) = \begin{cases} D_k(i), h_k(x_i) = 0 \\ D_k(i)\, exp\left(-\alpha_k y_i h_k(x_i)\right), h_k(x_i) \neq 0 \end{cases}$$

3. Рассчитать T_1, T_2 исходя из правила:

$$T_1 = \min_{i,\, y_i=1} \sum \alpha_k h_k(x_i), \;\; T_2 = \max_{i,\, y_i=-1} \sum \alpha_k h_k(x_i).$$

4. Итоговый экспертный комитет:

$$H(x) = \frac{1}{2} sign\left(\sum \alpha_k h_k(x_i) - T_1\right) + \frac{1}{2} sign\left(\sum \alpha_k h_k(x_i) - T_2\right)$$

Как видно из представленного алгоритма (Таблица 3.1) изменения касаются способа подсчета ошибки и способа обновления весов примеров, необходимых для отбора следующего лучшего классификатора. Для представленного алгоритма возможно сформулировать теорему о *верхней границе ошибки ансамбля тернарных классификаторов*:

Теорема (о верхней границе ошибки ансамбля тернарных слабых классификаторов):

Пусть $\epsilon_k < 0.5$ *для всех* k, *тогда ошибка ансамбля тернарных классификаторов ограничена сверху величиной:*

$$\epsilon_{final} \leq \prod_{k=1}^{K}\left[2\sqrt{\epsilon_k(1-\epsilon_k)}W_k + (1-W_k)\right] \quad (3.6)$$

Доказательство:

Разделим множество объектов на уверенно классифицированные (где $h_k(x_i) \in \{-1, +1\}$) и неопределённые ($h_k(x_i) = 0$). На объектах с неопределённым ответом веса D_k не обновляются. Тогда нормировочный коэффициент Z_k на шаге k можно определить как:

$$Z_k = \sum_{i:h_k(x_i)\neq 0} D_k(i)e^{-\alpha_k y_i h_k(x_i)} + \sum_{i:h_k(x_i)=0} D_k(i) \quad (3.7)$$

, где $\alpha_k = \frac{1}{2}\ln\left(\frac{1-\epsilon_k}{\epsilon_k}\right)$,

Обозначим $W_k = \sum_{i:h_k(x_i)\neq 0} D_k(i)$ — суммарный вес уверенных классификаций. Тогда, подставляя это выражение в (3.7) получим:

$$Z_k = W_k[(1-\epsilon_k)e^{-\alpha_k} + \epsilon_k e^{\alpha_k}] + (1-W_k) \quad (3.8)$$

Подставляя $\alpha_k = \frac{1}{2}\ln\left(\frac{1-\epsilon_k}{\epsilon_k}\right)$ в (3.8), получаем $(1-\epsilon_k)e^{-\alpha_k} + \epsilon_k e^{\alpha_k} = 2\sqrt{\epsilon_k(1-\epsilon_k)}$ и следовательно:

$$Z_k = 2W_k\sqrt{\epsilon_k(1-\epsilon_k)} + (1-W_k) \quad (3.9)$$

Тогда, используя (3.9) ошибка итогового ансамбля ограничена сверху как:

$$\epsilon_{final} \leq \prod_{k=1}^{K} Z_k = \prod_{k=1}^{K}\left[2W_k\sqrt{\epsilon_k(1-\epsilon_k)} + (1-W_k)\right] \quad (3.10)$$

Таким образом, использование тернарных классификаторов позволяет эффективно управлять ошибками первого и второго рода, вводя область неопределённости и повышая устойчивость итогового решения с помощью порогов T_1, T_2.

3.2.4 Применение тернарных классификаторов в задаче детектирования объектов

Рассмотрим, как описанные идеи используются в задаче детектирования объектов на изображениях. Работа [15] описывает задачу детектирования

фасадов домов с использованием каскадного детектора на основе экспертных комитетов. Система устроена как последовательность ансамблей тернарных классификаторов (комитетов), каждый из которых отвечает за обнаружение части образцов, общая структура которой изображена на Рис 3.2. Входной фрагмент изображения (например, оконное квадратное окно, проходящее по снимку) последовательно проверяется комитетами: если текущий комитет достаточно уверен, что фрагмент принадлежит классу «фасад» или «не фасад», он выдаёт соответствующее решение (объект обнаружен или отброшен). Если же комитет не набрал нужного большинства голосов (тернарный ответ 0 – «неуверен»), фрагмент передаётся следующему классификатору каскада. Такая организация гибче классического каскада Виолы–Джонса, поскольку позволяет наращивать структуру новыми экспертами для новых видов объектов без полной переработки всей цепочки, а изменения одного звена не требуют переобучения всего детектора. Кроме того, система может кодировать дополнительную информацию – например, индекс сработавшего экспертного комитета может указывать на тип фасада или условия, при которых он обнаружен.

В детекторе фасадов в качестве признакового пространства было выбрано модифицированное преобразование Цензуса (MCT), подробное описание которого представлено в **Приложении Б**. В работе в общей сложности сгенерировано 730 различных преобразования в окне размером 32x32 пикселей, которые включают различные вариации соотношения сторон, положения и масштаба MCT. Тернарный слабый классификатор построен на основе статистики по обучающей выборке для каждого из таких признаков и представляет собой вектор тернарных ответов на каждый из 512 вариантов. Ответ принимается в случае, если частота появления варианта для одного из классов превышает 0.9 и неопределенный ответ (0) в иных случаях.

Обучение детектора происходит итеративно, аналогично классическому подходу [180] с каскадом бинарных классификаторов, но с учётом тернарной логики. Для сравнения также был обучен и классический детектор на

бинарных слабых классификаторах. Алгоритм построения детектора можно описать так:

1. **Формирование обучающей выборки**. Исходные изображения размечаются: выделяются области, содержащие фасады зданий (положительные примеры, класс +1), и фоновые области без фасадов (отрицательные примеры, класс -1). Эти изображения фрагментируются на квадратные окна разных масштабов, генерируя множество кандидатов двух классов. Поскольку отрицательных фрагментов значительно больше, осуществляется случайное прореживание фона для балансировки данных. В эксперименте было выбрано 1102 фрагмента фасадов и 5000 фоновых фрагментов в обучающей базе.

2. **Обучение экспертных комитетов**. Из пространства признаков с помощью процедуры модифицированного Adaboost (Таблица 3.1) отбирается набор информативных слабых классификаторов и вычисляются их веса. Пороги экспертного комитета выбираются так, чтобы на обучающей базе не было ошибок I и II рода (FAR и FRR равны нулю). Критерием остановки отбора слабых классификаторов служит достижение минимально требуемой эффективности эксперта при отсутствии ошибок на обучающей выборке. В отличие от классического алгоритма, в котором критерием останова считается максимально допустимая ошибка I рода (FAR), при отсутствии ошибок II рода (FRR).

3. **Тестирование и итеративная накачка.** Построенный сильный классификатор применяется ко всем обучающим изображениям. Выявляются фрагменты, которые были пропущены предыдущими классификаторами. Эти фрагменты добавляются в обучающую выборку, после этого обучение следующего комитета происходит на новой выборке. Таким образом, каждый новый экспертный комитет фокусируется на тех объектах, которые не смогли пройти предыдущие стадии.

4. **Формирование детектора**. Шаг 2–3 повторяется: последовательно обучаются второй, третий и т.д. комитеты, каждый на оставшихся

нераспознанными объектах. Процесс продолжается, пока в выборке остаются не классифицируемые ни одним экспертом объекты или пока не будут удовлетворены требования по качеству системы (минимальный порог отказов).

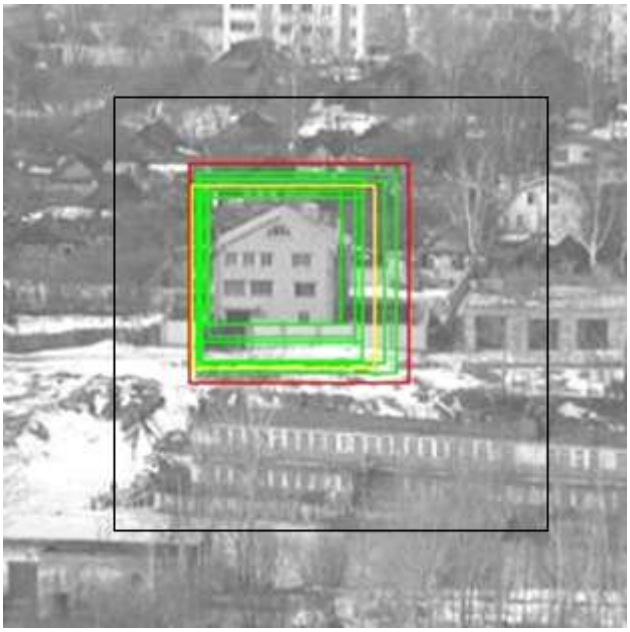


Рисунок 3.3 — Пример работы детектора (по цвету прямоугольника): Красный – размеченная цель, Желтый – лучший результат поиска, Зеленый – другие близкие результаты поиска, Черный – ошибочные результаты поиска

В результате получается каскад из нескольких экспертных комитетов, каждый специализирован на своём подмножестве пространства фасадов. Первый комитет, как правило, отлавливает большинство легко распознаваемых фасадов; последующие берут на себя более сложные случаи. В конечной системе при прогоне нового изображения каскад работает так: окно изображения проходит через первый тернарный классификатор. Если он выдаёт +1 («фасад»), окно считается обнаруженным фасадом; если -1, окно отбрасывается как фон. Если же результат 0 (неопределённость), окно передаётся второму комитету, и т.д., аналогично схеме, показанной на Рис 3.2. Эта каскадная детекция эффективно отсекает фон на ранних стадиях, избегая затрат на проверку всеми классификаторами каждого окна, и концентрирует

усилия на трудных случаях на более глубоких стадиях (пример работы Рис 3.3).

Построенная детектирующая систему была протестирована на ~200 контрольных изображениях городской застройки. Уже первый экспертный комитет показал высокое качество: доля пропущенных фасадов (FRR) составила 13.7%, а ложных срабатываний (FAR) –1.67%. Для сравнения, аналогичный детектор, построенный по бинарной схеме каскадов, имел существенно худшие показатели на тех же данных: FRR ≈ 30,3%, FAR ≈ 5.3%. Иными словами, тернарный подход почти вдвое уменьшил пропуск целей и в несколько раз сократил ложные тревоги по сравнению с классическим бинарным комитетом. Итоговые результаты тестирования на контрольных изображениях детекторов представлены в Таблице 3.2.

Таблица 3.2 — Сравнение результатов тестирования бинарного и тернарного детекторов

	Бинарный детектор	**Тернарный детектор**
FAR	0.071	**0.041**
FRR	0.52	**0.28**
Эффективность (R_3)	**1.0**	0.94

Отметим, что высокая ошибка II рода (FRR) в Таблице 3.2 не означает пропуска всего объекта. Для успешной детекции достаточно определить хотя бы одного кандидата в окрестности объекта. На контрольной выборке изображений оба варианта детектора смогли определить как минимум один размеченный кандидат на каждый объект. Но детектор, основанный на тернарных правилах, определяет больше верных кандидатов, что позволяет более точно локализовать объект. Кроме того, первый же каскадный элемент смог более строго отсечь фоновые объекты, при этом пропустив меньше кандидатов, чем бинарный классификатор.

3.3 Оптимальная разделяющая гиперплоскость

Одним из ключевых принципов классификации является требование **максимальной уверенности** при принятии решения. Интуитивно это можно объяснить с позиции коллектива классификаторов («комитета экспертов»): если большинство независимых классификаторов дают один и тот же ответ для некоторого образца, такой ответ с высокой вероятностью правильный и будет устойчив при появлении новых данных [92]. Геометрически это соответствует тому, что образец лежит далеко от границы разделяющей гиперплоскости, в глубине области своего класса. В противном случае, если образец расположен близко к границе разделения, разные классификаторы (с незначительно различающимися параметрами) могли бы классифицировать его по-разному. Следовательно, **максимизация зазора** между классами – то есть выбор разделяющей гиперплоскости, отстоящей как можно дальше от ближайших обучающих точек каждого класса – приводит к более надёжной, обобщающей классификации [142]. В литературе давно отмечено, что чем больше зазор классификатора, тем ниже ожидаемая ошибка обобщения модели [92] – модель менее подвержена переобучению. Таким образом, стремление «согласовать мнение большинства экспертов» сводится к задаче построения классификатора с максимально возможным зазором между классами.

Принцип максимального зазора лежит в основе метода опорных векторов (Support Vector Machine, SVM), впервые предложенного В. Н. Вапником и соавторами. Ещё в 1963 году Вапник с коллегами разработали алгоритм построения оптимальной разделяющей гиперплоскости для линейно разделимых данных (известный как метод *обобщённых портретов*) [6, 7]. Идея заключалась в выборе из множества разделяющих гиперплоскостей той, которая обеспечивает наибольший зазор между классами. Спустя несколько десятилетий эта теория была развита в полноценный алгоритм машинного обучения. Классический SVM был

сформулирован Вапником совместно с К. Кортес: ими предложен критерий максимизации зазора с допущением ошибок на обучении для случая неполной разделимости классов [57]. В этой работе была обоснована высокая обобщающая способность классификатора, достигаемая за счёт введения специального регуляризационного члена, ослабляющего строгость зазора.

Важно отметить, что реализация SVM приводит к решению задач квадратичной оптимизации с линейными ограничениями, в результате которых оптимальная гиперплоскость определяется лишь частью обучающих образцов, называемых *опорными векторами*. Именно они лежат на границе зазора и ограничивают его ширину; все остальные образцы можно отодвинуть дальше от границы без ухудшения качества разделения. За счёт этого классический SVM обладает свойством разреженности решения: как правило, доля опорных векторов невелика по сравнению с размером выборки [158], что упрощает решающее правило. Тем не менее, при неблагоприятных данных число опорных векторов может быть значительным, вплоть до включения значительной части выборки.

За время, прошедшее с появления SVM, было предложено множество его модификаций и улучшений. Одним из наиболее известных расширений является **ν-SVM** – вариант метода с параметром ν, предложенный для более удобного управления сложностью модели. В ν-SVM параметр $\nu \in (0, 1]$ позволяет напрямую контролировать долю опорных векторов и ошибок классификации, по сути, заменяя собой коэффициент регуляризации C в классической постановке [158]. Такое перепараметризование упрощает настройку модели: выбор ν задаёт компромисс между шириной зазора и количеством допущенных ошибок на обучении. Авторами ν-SVM теоретически и экспериментально показано, что данный параметр имеет ясный смысл и позволяет добиться той же эффективности, что и оптимизация с подбором C. Помимо ν-SVM, развитие метода опорных векторов включало обобщения на случаи одномерной классификации (one-class SVM для

выявления выбросов), поддержка регрессии (SVR, ввод ε-нечувствительной зоны ошибки) и множество прикладных усовершенствований.

Современные исследования продолжают развивать метод максимизации зазора. Предлагаются новые алгоритмы обучения SVM, повышающие его эффективность и пригодность для больших данных. В частности, были разработаны онлайн-алгоритмы обучения SVM, позволяющие обновлять модель последовательно, поступление за поступлением новых данных [50]. Также появились методы отбора признаков на основе SVM – хотя сам по себе классический алгоритм не выполняет это автоматически, можно использовать его веса для ранжирования важности признаков. К примеру, метод Recursive Feature Elimination (RFE) последовательно удаляет наименее важные признаки, основываясь на весовом коэффициенте вектора **w** SVM, что позволило успешно отбирать информативные подмножества признаков в задачах биоинформатики [90]. Эти и другие расширения сохраняют основной принцип максимизации зазора, но делают его применение более гибким и интерпретируемым.

Несмотря на выдающуюся эффективностью и строгие теоретические основы, классический алгоритм SVM обладает рядом существенных недостатков в практическом отношении:

- **Отсутствие дообучения**. Стандартный SVM – это *batch*-алгоритм: он обучается на фиксированной выборке и не предусматривает простого механизма дозагрузки новых данных. Добавление даже нескольких образцов требует перезапуска обучения на всём множестве, что вычислительно затратно. Это ограничивает применение SVM в условиях постоянно обновляющихся данных (онлайн-обучение).
- **Нет встроенного отбора признаков**. В процессе обучения SVM не происходит отсеивания нерелевантных признаков – модель учитывает *все* поданные на вход признаки (если пользователь явно

не осуществил предварительный отбор). Для задач с очень высокой размерностью признакового пространства это может приводить к переобучению или затруднять интерпретацию модели.

- **Трудности с интерпретацией значимости признаков**. Связанный с предыдущим пунктом недостаток – SVM не предоставляет явного рейтинга важности отдельных признаков. В линейном SVM относительную важность можно оценивать по абсолютным значениям компонент весового вектора **w**, однако при использовании нелинейных ядер влияние каждого исходного признака скрыто в пространстве высокой размерности. В отличие от, скажем, решающих деревьев, где вклад признаков ясен, в SVM требуется дополнительный анализ, чтобы понять, какие признаки наиболее влияют на решение.
- **Большое число опорных векторов**. Для задания разделяющей гиперплоскости SVM необходимо хранить все опорные векторы, каждый из которых представляет некоторый обучающий пример. Решающая функция SVM в разреженном случае имеет вид: $f(x) = \sum_{i \in SV} \alpha_i K(x_i, x) + b$, где суммирование идёт по опорным векторам. Поэтому память, необходимая для хранения модели, линейно растёт с числом опорных объектов. В худшем случае, особенно на сложных данных, опорными могут стать большинство обучающих примеров – тогда модель практически запоминает выборку и теряет преимущества обобщения. На больших выборках это приводит к высоким затратам по памяти и времени обучения [136]. Таким образом, классический SVM плохо масштабируется по количеству данных: сложность обучения растёт не менее квадратично от размера выборки, что затрудняет его применение в больших базах данных без специальных оптимизаций.

Перечисленные ограничения стимулировали исследования по усовершенствованию метода опорных векторов. В следующих разделах будет описан последовательный онлайн-алгоритм “OnSVM” [12, 13, 117], предназначенный для преодоления некоторых недостатков классического SVM. *OnSVM* поддерживает пошаговое дообучение модели на новых данных, включает механизмы отбора наиболее информативных объектов и признаков, позволяет обучать экспертные комитеты (3.3), а по качеству классификации не уступает классическому SVM. Такой подход объединяет преимущества максимизации зазора с эффективностью и адаптивностью, необходимыми для применения в современных условиях потоковых данных и высокомерных признаковых пространств.

3.3.1 Математическая формулировка оптимизационной задачи

Исходно имеется набор объектов обучающей выборки $S = \{x_1, \dots, x_N\}$ и соответствующий им набор меток $L = \{y_1, \dots, y_N\}$, где $x_i \in X$ - обучающий прецедент пространства X в котором задано скалярное произведение, а $y \in -1,1$ метка класса. Уравнение гиперплоскости в пространстве X имеет вид:

$$\left\{x \in X \middle| H_{(w,b)}(x) = \langle w, x\rangle - b = 0\right\} \tag{3.11}$$

,где $w \in X, \|w\| = 1, b \in \Re$, а пара $(w, b) \in X \times \Re$ задает гиперплоскость в пространстве X и вектор w является нормалью к гиперплоскости. Заданному вектору w соответствует множество гиперплоскостей H_w, среди которых найдется пара $H_{(w,b^+)}, H_{(w,b^-)} \in H_w$, таких что:

$$\begin{matrix}(w,b^+){:}H_{w,b^+}(x_i)\geq 0, \forall i \in S^+ \\ (w,b^-){:}H_{w,b^-}(x_i)< 0, \forall i \in S^-\end{matrix} \tag{3.12}$$

,где $S^+ = \{i \in Z_+ | y_i = 1\}$, $S^- = \left\{j \in Z_+ \middle| y_j = -1\right\}$ - индексы объектов разных классов. Величину $\rho_{(n,b^+,b^-)} = b^+ - b^-$ называют “зазором” гиперплоскостей, в зависимости от набора обучающих данных эта величина может быть как

положительной, так и отрицательной. Условия (3.12) определяют параметры b^+и b^- однозначно при условии максимальной величины зазора.

В случае положительного зазора множества объектов, образуемые индексами S^+ и S^-, разделены в пространстве *X*. Именно такого положения желательно достигнуть на первом этапе обработки данных за счет оптимального выбора признакового пространства, однако достичь этого удается далеко не всегда.

Более важным для решения практических задач и более важным является случай отрицательной величины зазора. При этом гиперплоскости (3.12) разбивают признаковое пространство на три области: область H^+ - содержит только примеры с меткой 1, область H^- - содержит только примеры с меткой -1, а область между гиперплоскостями H^0содержит объекты обоих классов; Рис. 3.4 качественно поясняет такую ситуацию на примере двумерного пространства. В этом случае максимальная величина зазора (минимальная по абсолютной величине) соответствует минимизации долей данных из обучающей выборки в области между гиперплоскостями [5, 158, 178].

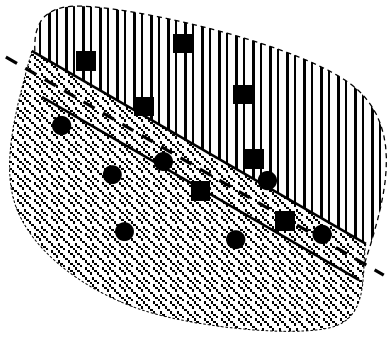


Рисунок 3.4 — Геометрическая интерпретация на плоскости классифицирующей гиперплоскости и зазора

Поиск параметров таких гиперплоскостей может быть сформулирован в виде оптимизационной задачи:

$$\max_{w,b^+,b^-}(b^+-b^-)$$
$$s.t.\begin{cases}\langle w,x\rangle-b^+\geq 0,\forall i:y_i=1\\ \langle w,x\rangle-b^-<0,\forall i:y_i=-1\\ \|w\|^2=1\end{cases} \quad (3.13)$$

Обучающая выборка всегда представляет собой лишь часть общего набора возможных ситуаций, поэтому проведение этих гиперплоскостей не может гарантировать от ошибок. Поэтому обычно некоторую долю ошибок считают допустимой и проводят разграничивающие гиперплоскости исходя из этой величины. Для описания этого в задачу (3.13) вводят корректирующие коэффициенты $\xi_i \geq 0$ и параметры регуляризации $c_i \geq 0, \eta \geq 0$ [5]. С учетом этого оптимизационная задача (3.13) формулируется следующим образом:

$$\max_{w,b^+,b^-}(b^+-b^--\sum_{i=1\ldots N} c_i\xi_i)$$
$$s.t.\begin{cases}y_i\left(\langle w,x\rangle-\frac{1+y_i}{2}b^++\frac{y_i-1}{2}b^-\right)\geq-\eta\xi_i\\ \xi_i\geq 0\\ \|w\|^2=1\end{cases} \quad (3.14)$$

Параметры c_i, η увеличивают величину зазора (уменьшают его по модулю) и будут увеличивать ошибку, но при этом они повышают обобщающую возможность классификатора. Здесь они вводятся как свободные параметры задачи, и их оптимальные значения будут обсуждаться ниже.

3.3.2 Формулировка двойственной задачи

Задача (3.14) является задачей нелинейного программирования, поскольку ограничивающие условия целевой функции имеют квадратичную функцию. Поиск решения задачи в прямой постановке представляет известную сложность, поэтому перейдем к двойственной оптимизационной задаче, эквивалентной исходной. Введем функцию Лагранжа оптимизационного функционала для задачи (3.14):

$$L(w,b^+,b^-,\xi,\alpha,\lambda,\mu)=\sum_{i=1\dots N} c_i\xi_i-(b^+-b^-)+\frac{\lambda}{2}(\|w\|^2-1)$$
$$\sum_{i=1\dots N}\alpha_i\left(y_i\left(\langle w,x\rangle-\frac{1+y_i}{2}b^+ +\frac{y_i-1}{2}b^-\right)+\eta\xi_i\right)-\sum_{i=1\dots N}\mu_i\xi_i \quad (3.15)$$

,где $\lambda\neq 0, \alpha_i\geq 0, \mu_i\geq 0$ - коэффициенты Лагранжа. Эту функцию необходимо минимизировать по прямым переменным (w, b^+, b^-, ξ) и максимизировать по двойственным переменным (α, λ, μ). Эти условия приводят к следующим выражениям:

$$\begin{gathered}\frac{\partial L}{\partial w}=\lambda w-\sum_{i=1\dots N}\alpha_i y_i x_i=0\Rightarrow w=\frac{1}{\lambda}\sum_{i=1\dots N}\alpha_i y_i x_i\\ \frac{\partial L}{\partial b^+}=-1+\sum_{i=1\dots N}\alpha_i y_i\frac{1+y_i}{2}=0\Rightarrow\sum_{i=1\dots N}\alpha_i=1,\forall i{:}y_i=1\\ \frac{\partial L}{\partial b^-}=1-\sum_{i=1\dots N}\alpha_i y_i\frac{y_i-1}{2}=0\Rightarrow\sum_{i=1\dots N}\alpha_i=1,\forall i{:}y_i=-1\\ \frac{\partial L}{\partial \xi_i}=c_i-\eta\alpha_i-\mu_i=0\Rightarrow\alpha_i\leq\frac{c_i}{\eta}\\ \lambda^2=\sum_i\sum_j\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle\end{gathered} \quad (3.16)$$

С учетом всех подстановок (3.16) функция Лагранжа (3.15) принимает вид $L=-\sqrt{\sum_i\sum_j\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle}$, и двойственная задача оптимизации формулируется как:

$$\begin{gathered}min\left(W(\alpha)=\frac{1}{2}\sum_i\sum_j\alpha_i\alpha_j y_i y_j\langle x_i,x_j\rangle\right)\\ s.t.\begin{cases}\sum_{i=1\dots N}\alpha_i=1,\forall i{:}y_i=1\\ \sum_{i=1\dots N}\alpha_i=1,\forall i{:}y_i=-1\\ 0\leq\alpha_i\leq\frac{c_i}{\eta}\end{cases}\end{gathered} \quad (3.17)$$

Такая формулировка оптимизационной задачи позволяет применить модифицированный вариант метода градиентного спуска для построения алгоритма ее решения. Подробно этот метод - метод локальной оптимизации (СМО), будет рассмотрен в следующих разделах.

3.3.3 Решающая функция

В простейшем, бинарном, варианте решающая функция зависит от координат в N-мерном пространстве и может принимать либо значение +1, которое соответствует принадлежности данного объекта к первому классу, либо значение -1, которое соответствует принадлежности объекта второму классу. Обычно $f_2: X\to\{-1,1\}$ использует проведение одной гиперплоскости в X - пространстве и имеет вид:

$$f_2(x) = sgn\left(\frac{1}{\lambda}\sum_{1\dots N}\alpha_i y_i\langle x_i, x\rangle - \frac{b^+ + b^-}{2}\right) \quad (3.18)$$

В (3.18) разделяющая граница проведена посередине между гиперплоскостями, определяемыми параметрами b^+, b^-, которые задаются условиями дополняющей нежесткости Каруша-Куна-Таккера (*ККТ*) [48]: $b^+ = \max\limits_{i\in S^+:\alpha_i<\frac{c_i}{\eta}}(\langle w, x_i\rangle)$,$b^- = \min\limits_{i\in S^-:\alpha_i<\frac{c_i}{\eta}}(\langle w, x_i\rangle)$.

В случае частичной вложенности множеств объектов разных классов, которому соответствует отрицательная величина зазора между гиперплоскостями, возможно использовать тернарные функции принятия решения вида $f_3: X \to \{-1,0,1\}$, аналогично (3.3):

$$f_3(x) = \begin{cases} 1, \frac{1}{\lambda}\sum_{1\dots N}\alpha_i y_i\langle x_i, x\rangle - max(b^+, b^-) \geq 0 \\ -1, \frac{1}{\lambda}\sum_{1\dots N}\alpha_i y_i\langle x_i, x\rangle - min(b^+, b^-) < 0 \\ 0, otherwise \end{cases} \quad (3.19)$$

В этом случае для контроля качества классификатора (3.19) распространяются все определения, используемые для экспертного комитета, такие как “ошибка” и “эффективность” тернарного классификатора. Также можно ввести дополнительный параметр – *ошибка зазора* – который определяется долей объектов, попавших в зазор: $E_m = \frac{1}{N}|\{i|f_3(x_i) = 0\}|$. Тогда «эффективность классификатора» будет вычисляться ка: $R_3 = 1 - E_m$.

3.3.4 Обобщение на случай нелинейной гиперповерхности

На практике часто линейной границы оказывается недостаточно для получения оптимальных результатов классификации, поскольку исходные данные могут обладать более сложной, нелинейной структурой классов. В таких случаях используют некоторое нелинейное преобразование исходного пространства в признаковое $\Phi: X \to F, F \in \Re^g$, выбираемое обычно априорно [147].

Заметим, что двойственная задача оптимизации (3.17)

зависит от объектов обучения только в виде попарных скалярных произведений $\langle x_i, x_j \rangle$, как и решающие правила (3.18), (3.19), также зависящие только от скалярных произведений между распознаваемым объектом и объектами обучения. Это обстоятельство позволяет применить так называемый *ядровой переход*. Предположим далее, что скалярное произведение в пространстве F известно как функция K в исходном пространстве X:$\langle \Phi(x), \Phi(y) \rangle = K(x, y)$. Оптимизационная задача (3.17) с учетом ядровой функции может быть записана как:

$$min\left(W(\alpha)=\frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j K_{i,j}\right) \quad s.t. \begin{cases} \sum_{\forall i: y_i=1} \alpha_i = 1 \\ \sum_{\forall i: y_i=-1} \alpha_i = 1 \\ 0 \le \alpha_i \le \frac{c_i}{\eta} \end{cases} \tag{3.20}$$

3.3.5 Выбор параметров регуляризации

Классическая формулировка задачи *SVM* содержит свободный параметр, выбор которого в определенной степени затруднен [58]. Попытка систематизации этого параметра привела к появлению модификации ν -*SVM*, которая впервые изложена в работе [158]. Сформулируем здесь следствия предложенного алгоритма (3.17) и покажем, что при определенных условиях его параметры c_i, η эквивалентны параметрам известного ранее алгоритма ν -*SVM*.

Для определения связи между параметрами введем понятие ошибок обучения:

Определение (доля ошибок обучения**)**: Доля *ошибок обучения* для решающей функции $f_3(x)$ (без учета ошибки зазора) определяется как доля объектов обучения, для которых $\xi_i > 0$, $R_e = \frac{1}{N}|\{i|\xi_i > 0\}|$. Тогда справедлива следующая теорема:

Теорема (Свойства параметра η для классификации): предположим, что параметры функции принятия решения $f_3(x)$ соответствуют решению оптимизационной задачи (3.20). Тогда:

1. η является верхней оценкой на долю ошибок обучения;
2. η является нижней оценкой на долю опорных векторов;

при нормировочных условиях $\begin{cases} \sum_{\forall i: y_i=1} c_i = 1 \\ \sum_{\forall i: y_i=-1} c_i = 1 \end{cases}$

Доказательство: 1) Из условий *KKT*, если $\xi_i > 0$, то $\mu_i = 0$ и $\alpha_i = \frac{c_i}{\eta}$. В худшем случае только доля $\eta = \sum_{D^+ \in S^+} c_i = \sum_{D^- \in S^-} c_i$ объектов обучения будет удовлетворять условиям (3.19) на вес α_i целевой функции, с учетом нормировки. При этом оставшиеся примеры имеют нулевые веса α_i, выпадая из множества опорных векторов.

2) Согласно п. 1. минимальное количество опорных векторов определяется максимальной оценкой на долю ошибочных объектов, т.к. веса остальных векторов нулевые.

3.3.6 О связи с методом ν -SVM

Рассмотрим теперь двойственную оптимизационную задачу в постановке ν *-SVM [158]*:

$$\begin{gathered} min\left(W(\alpha)=\frac{1}{2}\sum_i \sum_j \alpha_i \alpha_j y_i y_j K_{i,j}\right) \\ s.t.\begin{cases} \sum_{i=1}^N y_i \alpha_i = 0 \\ \sum_{i=1}^N \alpha_i \geq \nu \\ 0 \leq \alpha_i \leq c_i \end{cases} \end{gathered} \tag{3.21}$$

Соответствующие условия Лагранжа и условия дополняющей нежесткости *ККТ* для (3.21) могут быть записаны как:

$$\begin{gathered} \alpha_i + \mu_i = c_i, \sum_{i=1}^N y_i \alpha_i = 0, \sum_{i=1}^N \alpha_i - \delta = \nu \\ \delta\rho = 0, \mu_i \zeta_i = 0, \sum_{i=1\ldots N} \alpha_i (y_i(\langle w, x\rangle + b) - \rho + \xi_i) = 0 \end{gathered} \tag{3.22}$$

, где δ, μ_i, α_i - коэффициенты Лагранжа. Между задачами (3.20) и (3.21) существует связь, которую можно сформулировать следующим образом:

Теорема (об эквивалентности). Если метод ν *-SVM* приводит к такому решению, что $\rho > 0$, тогда задачи (3.20) и (3.21) эквивалентны и их решения совпадают при условии, что параметр η равен ν.

Доказательство. Заметим, что при $\rho > 0$ из условий *ККТ* следует равенство $\sum_{i=1}^{N} \alpha_i = \nu$. Нетрудно убедиться, что это равенство справедливо и для (3.17) заменой $\alpha_i = \frac{\beta_i}{2\eta}$ и сложением ограничивающих равенств. Разность этих равенств дает эквивалентное условие $\sum_{i=1}^{N} y_i \beta_i = 0$ на веса разных классов. Задачи (3.20) и (3.21) имеют совпадающие условия, при равных параметрах $\eta = \nu$ их решения совпадают.

Параметры гиперплоскости (3.18) могут быть выражены через параметры ν -*SVM*: $b^+ = \rho - b$, $b^- = -\rho - b$, при соблюдении условий теоремы. Решающая функция запишется как $f_2(x) = sgn(\sum_{1\dots N} \alpha_i y_i K(x_i, x) + b)$, с величиной зазора гиперплоскости $\rho = \frac{(b^+ - b^-)}{2}$.

3.3.7 Метод последовательной оптимизации

Наиболее известными методами решения проблемы *SVM*, проблемы квадратичного программирования, является метод сопряженных градиентов [178] и метод последовательной оптимизации (*SMO*) [147]. Эти методы основаны на поиске оптимального сдвига вдоль заранее выбранного направления. Каждый сдвиг происходит относительно начального вектора α вдоль направления h так, чтобы удовлетворять ограничениям задачи. Новый вектор запишется как $\alpha^* = \alpha + \lambda^* h$, где:

$$\begin{matrix} \lambda^* = argmax\, W(\alpha + \lambda h) \\ s.t. 0 \le \lambda \le \varphi(\alpha, h) \end{matrix} \qquad (3.23)$$

Верхняя граница $\varphi(\alpha, h)$ определяется условиями оптимизации квадратичной задачи. Платт [147] отметил, что расчеты величины сдвига становятся намного быстрее, если направление поиска в основном содержит нулевые коэффициенты. Он предложил оставить лишь пару ненулевых коэффициентов, таких, чтобы $\sum_k h_k = 0$. Алгоритм *SMO* использует направление поиска h, в котором отличны от нулевых лишь пара коэффициентов: +1 и -1. Отметим, что такой выбор автоматически

удовлетворяет условию ограничения на сумму весовых коэффициентов α, а ограничение на величину α можно получить выбором сдвига λ.

В рассматриваемой задаче (3.23) расчеты сдвига можно производить сразу для 4-х коэффициентов направления без существенной потери скорости. Ограничения здесь разделены для примеров разных классов и поэтому точное решение может быть найдено для совместных пар из разных классов. Новый вектор весов можно записать как:

$$\alpha^* = \alpha + \vartheta^* u + \phi^* v \tag{3.24}$$

,где $u = Z_{i,j}$ и $u = Z_{t,p}$ $i,j \in S^+$ $t,p \in S^-$. Здесь $Z_{x,y}$ - разреженный вектор коэффициентов +1, -1 с индексами x, y соответственно.

В этом случае, задача квадратичной оптимизации запишется в следующей форме:

$$\begin{gathered}(\vartheta,\phi)=argmax\, W(\alpha+\vartheta u+\phi v) \\ 0\le\vartheta\le min\left(\begin{matrix}\frac{c_i}{\eta}-\alpha_i \\ \alpha_j\end{matrix}\right)=\vartheta_{max} \\ s.t. \\ 0\le\phi\le min\left(\begin{matrix}\frac{c_t}{\eta}-\alpha_t \\ \alpha_p\end{matrix}\right)=\phi_{max}\end{gathered} \tag{3.25}$$

Вычисляя экстремум целевой функции $W(\alpha + \vartheta u + \phi v)$ по параметрам (ϑ, ϕ) получим:

$$\begin{cases}\hat{\vartheta} = \frac{-\langle vg\rangle T+\langle ug\rangle V}{2(UV-TT)} \\ \hat{\phi} = \frac{-\langle ug\rangle T+\langle vg\rangle U'}{2(UV-TT)}\end{cases} \begin{gathered}U = \sum_i \sum_j u_i u_j y_i y_j K_{i,j} \\ V = \sum_i \sum_j v_i v_j y_i y_j K_{i,j} \\ T = \sum_i \sum_j u_i v_j y_i y_j K_{i,j}\end{gathered} \tag{3.26}$$

,где $g_k = \frac{\partial W(\alpha)}{\partial \alpha_k} = \sum_i \alpha_i y_k y_i K_{k,i}$ - градиент целевой функции.

Применяя разреженную форму для направлений u и v, оптимальные значения величины шага можно записать как:

$$\begin{cases} \hat{\vartheta} = \frac{-T(g_t - g_p) + V(g_i - g_j)}{2(UV - TT)} \\ \hat{\phi} = \frac{-T(g_i - g_j) + U(g_t - g_p)}{2(UV - TT)} \end{cases}, \quad \begin{matrix} U = K_{i,i} + K_{j,j} - 2K_{i,j} \\ V = K_{t,t} + K_{p,p} - 2K_{t,p} \\ T = -K_{i,t} + K_{i,p} + K_{j,t} - K_{j,p} \end{matrix} \tag{3.27}$$

С учетом ограничений, оптимальные значения шага достигаются при значениях $\begin{matrix} \vartheta^* = \min(\vartheta_{max}, \hat{\vartheta}) \\ \phi^* = \min(\phi_{max}, \hat{\phi}) \end{matrix}$. На практике алгоритм *SMO* обычно используют с небольшим допуском $\tau > 0$ [53, 55]. Выбираются только такие направления поиска, чтобы $\langle h, g \rangle \geq \tau$ и $\vartheta_{max} > 0$, $\phi_{max} > 0$. При таком выборе появляется возможность увеличения целевой функции W вдоль направления h. Поэтому для определения направления поиска, будем выбирать такие пары, что:

$$\begin{matrix} i, j \in S^+, u = Z_{i,j} \\ t, p \in S^-, v = Z_{t,p} \end{matrix} \Leftrightarrow \begin{cases} \alpha_i < \frac{c_i}{\eta}, \alpha_j > 0 \\ \alpha_t < \frac{c_t}{\eta}, \alpha_p > 0 \\ g_i + g_t - g_j - g_p \geq \tau \end{cases} \tag{3.28}$$

Среди возможных квартетов индексов объектов (i, j, t, p), удовлетворяющих условиям (3.28) наиболее эффективен квартет с максимальным направленным градиентом $\langle h, g \rangle$. Этот выбор описан в контексте понятия оптимальной гиперплоскости в работах [5, 178].

3.3.8 Алгоритм последовательного обучения OnSVM

Алгоритм основан на последовательной оптимизации параметров задачи (3.20) для каждого вновь добавляемого объекта x_k. Выделим три основных шага алгоритма:

- **Insert***:* добавление объекта во множество опорных векторов $S' = \{S, x_k\}$;
- **Solve***:* поиск τ -четверки и формирование для неё оптимального решения;
- **Remove***:* удаление объекта из множества S при достижении условий исключения.

3.3.8.1 Алгоритм *Insert(k)*

В отличие от постановки задачи классической *SVM*, потоковое добавление объектов во множество *S* приводит к изменению ограничивающих условий на коэффициент α (3.20). В самом деле, поскольку нормировочный коэффициент c зависит от общего числа объектов, то добавление нового объекта снижает верхнюю границу $\alpha \leq \alpha_{bound} = \frac{c}{\eta}$. Первый этап, *Insert(k),* состоит из операций ограничения параметров α, присвоении новому объекту начального значения и пересчета градиентов g.

Алгоритм 3.3 — Добавление объекта во множество опорных векторов

Insert(k)

if $y_k = +1$ then

$N^+ \leftarrow N^+ + 1, \alpha^+_{bound} \leftarrow \frac{1}{\eta N^+}$

$\alpha_s \leftarrow min(\alpha_s, \alpha^+_{bound}), \alpha_k \leftarrow 1 - \sum_{i \in S^+} \alpha_i$

$S^+ \leftarrow S^+ \cup \{k\}$

else

$N^- \leftarrow N^- + 1, \alpha^-_{bound} \leftarrow \frac{1}{\eta N^-}$

$\alpha_s \leftarrow min(\alpha_s, \alpha^-_{bound}), \alpha_k \leftarrow 1 - \sum_{i \in S^-} \alpha_i$

$S^- \leftarrow S^- \cup \{k\}$

endif

$g_s \leftarrow -\sum_{i \in S} y_i y_s \alpha_i K_{i,s}, \forall s \in S,$

,здесь N^+, N^-- количество объектов с положительными и отрицательными метками соответственно, $\alpha^+_{bound}, \alpha^-_{bound}$ - верхние границы.

Начальное значение α выбрано таким, чтобы удовлетворять ограничительным равенствам. Этот выбор может не соответствовать оптимальному решению. Сдвиг верхней границы также может повлиять на отклонение от оптимума. Поэтому, следующий шаг – это поиск оптимального совместного решения для измененного множества опорных векторов *S*.

3.3.8.2 Алгоритм *Solve()*

На шаге *Solve()* формируется τ -четверка опорных векторов из S с максимальным направленным градиентом. В случае, если τ -четверка не может быть найдена, этот шаг должен быть пропущен. Оптимальное решение для τ -четверки можно получить используя (3.28). Также на этом этапе определяются необходимые параметры смещения гиперплоскостей b.

Алгоритм 3.4 — Поиск совместного решения для τ -четверки

Solve()

$i \leftarrow \underset{s \in S^+}{argmax}(g_s)$ with $\alpha_i < \alpha_{bound}^+, j \leftarrow \underset{s \in S^+}{argmin}(g_s)$ with $\alpha_j > 0$

$t \leftarrow \underset{s \in S^-}{argmax}(g_s)$ with $\alpha_t < \alpha_{bound}^-, p \leftarrow \underset{s \in S^-}{argmin}(g_s)$ with $\alpha_p > 0$

$\delta \leftarrow g_i - g_j + g_t - g_p$

if $\delta > \tau$ then

$\vartheta \leftarrow min\left(\hat{\vartheta}, \alpha_{bound}^+ - \alpha_i, \alpha_j\right)$,

$\phi \leftarrow min\left(\hat{\phi}, \alpha_{bound}^- - \alpha_t, \alpha_p\right)$, $\{\hat{\vartheta}, \hat{\phi}\}$-as defined in (3.27)

$\alpha_i \leftarrow \alpha_i + \vartheta, \alpha_j \leftarrow \alpha_j - \vartheta, \alpha_t \leftarrow \alpha_t + \phi, \alpha_p = \alpha_p - \phi$

$b^+ \leftarrow -\underset{s \in S^+}{max}(g_s)$ with $\alpha_s < \alpha_{bound}^+$

$b^- \leftarrow \underset{s \in S^-}{max}(g_s)$ with $\alpha_s < \alpha_{bound}^-$

endif

Для τ -четверки отбираются те вектора i, которые не достигли ограничений на α_i в соответствующих направлениях поиска решения.

3.3.8.3 Алгоритм *Remove()*

Заключительный шаг *Remove()* необходим для освобождения множества S опорных векторов от векторов, не участвующих в формировании параметров разделяющей гиперплоскости j: $\alpha_j = 0, g_j < 0$.

Алгоритм 3.5 — Исключение опорного вектора

Remove()

$i \leftarrow \underset{s \in S^+}{argmin}(g_s)$, with $\alpha_s = 0$

if $g_i < 0$ then $\quad S^+ = S^+ - i$

$j \leftarrow \underset{s \in S^-}{argmin}(g_s)$, with $\alpha_s = 0$

if $g_j < 0$ then $\quad S^- = S^- - j$

Алгоритм *Remove()* исключает из каждого подмножества S^+, S^- не более одного опорного вектора, удовлетворяющего условиям исключения. Решение принимается по минимальному значению градиента g.

3.3.8.4 Алгоритм OnSVM

Алгоритм обучения *OnSVM* состоит из процедур: *Initialization()* – сбор минимального числа необходимых опорных векторов; *Process()* - потоковая обработка примеров объектов, включая поиск оптимального решения и удаления опорных векторов; *Finalize()*- поиск окончательного решения.

Алгоритм 3.6 — Алгоритм OnSvm

Initialize()

Repeat at least 4 times:

-Fetch example k;

-Call ***Insert***(k);

Process()

Repeat a predefined number of times:

-Fetch an example k;

-Run ***Insert***(k);

-Run ***Solve***();

-Run ***Remove***();

Finalize()

Repeat while $\delta > \tau$

Run ***Solve***()

Run ***Remove***()

Алгоритм OnSVM может быть использован как алгоритм последовательного обучения. В этом случае модель границы готова к распознаванию объектов сразу после процедуры *Initialize*(). Обучение последующих примеров проводится процедурой *Process*(). Шаг *Finalize*() введен для окончательной оптимизации опорных векторов, нарушающих τ - условие.

На практике последовательные алгоритмы обучаются на основе эпох. Каждая эпоха состоит из случайно отобранных примеров объектов. После окончания заданного количества итераций по эпохам выполняется процедура *Finalize*(). Эмпирические данные свидетельствуют, что обученный на одной эпохе классификатор почти не отличается от решения SVM.

Рассмотренный алгоритм основан на простом и эффективном методе SMO. Вопрос сходимости этого метода в его обобщенной формулировке (GSMO) подробно рассмотрен в работе [109]. Легко убедиться, что алгоритм OnSVM является алгоритмом GSMO и удовлетворяет условиям теоремы о сходимости. Поэтому, согласно выводам сделанным в работе [109], для заданного положительного $\tau > 0$, (Алг. 3.6) сходится за конечное число шагов k.

3.3.9 Увеличение производительности

Существенный недостаток алгоритма (Алг. 3.6) заключается в необходимости пересчета градиентов g при добавлении объекта. Пересчет градиентов существенно увеличивает количество вычислений процедуры обучения, замедляя работу алгоритма *Insert()*. Причина состоит в зависимости верхней границы α_{bound} от количества обучаемых векторов.

Заметим, что сдвиг верхней границы влияет на те опорные вектора, величины весов α которых достигли границы, а также на веса достаточно близких к границе. Таким образом, опорные вектора из множества S можно разделить на 3 подмножества: с не модифицируемыми весами S_0; вектора с граничными весами – подмножество S_1; подмножество S_2 с весами не дальше

от границы α'_{bound} чем на величину сдвига $\Delta\alpha = \alpha^{'}_{bound} - \alpha^{''}_{bound}$. Здесь $\alpha^{'}_{bound}$ и $\alpha^{''}_{bound}$ - верхние границы до добавления вектора и после соответственно. Очевидно, что граничные опорные вектора имеют одинаковую величину сдвига, чтобы удовлетворить граничным условиям, после добавления вектора. Это свойство можно использовать для уменьшения количества вычислений ядровой функции.

Введем дополнительную переменную $\Delta g_i = \sum_{j\in S_1} y_i y_j K_{i,j}$ как сумму ядровых функций по отношению к опорному вектору i. Тогда градиент этого вектора запишется как:

$$g^{''}_i = g^{'}_i + \Delta\alpha\Delta g + \sum_{j\in S_2}\left(\alpha^{'}_j - \alpha^{''}_{bound}\right) y_i y_j K_{i,j}\,, \forall i \in S \tag{3.29}$$

Вычисление ядровых функций здесь необходимо выполнить лишь для опорных векторов из множества S_2, а величины Δg_i могут быть вычислены заранее. Заметим также, что подмножество S_2 - переходное: $S_0 \to S_2 \to S_1$. Такой переход требует изменения переменной Δg_i:

$$\Delta g^{''}_i = \Delta g^{'}_i + y_i y_j K_{i,j}, \forall i \in S_1, j \in S_2 \tag{3.30}$$

Кроме прямого перехода опорного вектора в подмножество S_1, возможен и обратный переход. Такая ситуация возникает во время поиска оптимального совместного решения для τ -четверки в алгоритме *Solve*. При обратном переходе, $S_1 \to S_0$:

$$\Delta g^{''}_i = \Delta g^{'}_i - y_i y_j K_{i,j}, \forall i \in S_1, j \notin S_1 \tag{3.31}$$

Алгоритм *Solve* также изменяет веса α в случае существования τ -четверки. Для сохранения действительных значений градиента, в алгоритм *Solve* вводим дополнительный пересчет градиента:

$$g^{''}_s = g^{'}_s - \vartheta y_i y_s K_{i,s} + \vartheta y_j y_s K_{j,s} - \phi y_p y_s K_{p,s} + \phi y_t y_s K_{t,s}, \forall s \in S \tag{3.32}$$
$$i,j,p,t \in \tau - quad$$

Предлагаемые изменения существенно уменьшают количество вычислений ядровых функций. Так для Алг. 3.6 оценка на количество вызовов

ядровой функции составляет $C(K)\sim N^2$. С учетом ускоряющих изменений количество вызовов может быть уменьшено до линейной зависимости и в общем случае составляет $C^{OnSVM}(K)\sim N + o(N^2) \ll N^2$. Здесь N – количество опорных векторов во множестве *S*.

3.3.10 Вычислительный эксперимент

Апробация предложенного алгоритма проводилась на данных *MINIST* ручного начертания цифр [35, 108]. По этим данным была сформирована задача классификации начертания каждой цифры от всех остальных цифр в наборе 0-9, данные разделялись на тренировочную и тестовую группы в соотношении $2:1$. В экспериментах использовались данные Optdigits [108] и Pendigits [35], заметно различающиеся по сложности разделения данных на классы. На этих данных предложенный здесь алгоритм $OnSVM$ сравнивался с известной реализацией v *-SVM* из открытой библиотеки машинного обучения *LIBSVM* [53] при ядровой функции гауссова типа $K_{RBF}(x,y) = exp(-\gamma\|x - y\|^2)$, использовались значения параметров $\eta = 0.01, \gamma = 0.0005, \tau = 0.0001$.

В Таблице 3.7 приведены результаты сравнения алгоритмов $OnSVM$ и $v - SVM$ на базе *Optdigits*, в Таблице 3.8 – на данных *Pendigits*, здесь Num – класс распознавания; nSV – количество опорных векторов, nSV Max – пиковое количество опорных векторов на эпохе обучения; E_2 – ошибка классификации на тестовых примерах в режиме бинарного классификатора; E_3 – ошибка тернарного классификатора; R_3 – эффективность тернарного классификатора.

Таблица 3.7 — Сравнительные результаты работы алгоритма для набора Optdigits

	OnSVM					**LIBSVM**	
Num	**nSV**	**nSV Max**	E_2 **(%)**	E_3 **(%)**	R_3 **(%)**	**nSV**	E_2 **(%)**
0	58	60	0.0	0.00	96.17	80	0.05
1	104	109	0.89	0.05	93.88	149	0.67
2	99	100	0.11	0.00	96.00	126	0.17
3	119	126	0.72	0.00	93.9	158	0.67
4	100	101	0.33	0.00	95.33	133	0.17
5	107	121	0.45	0.11	94.6	139	0.45
6	63	66	0.22	0.00	95.66	84	0.17
7	83	88	0.72	0.00	95.83	105	0.72
8	164	171	0.95	0.00	90.49	203	0.95
9	175	183	1.28	0.06	93.77	230	0.95
Total	107.2	112.5	0.57	0.02	94.6	140.7	0.5

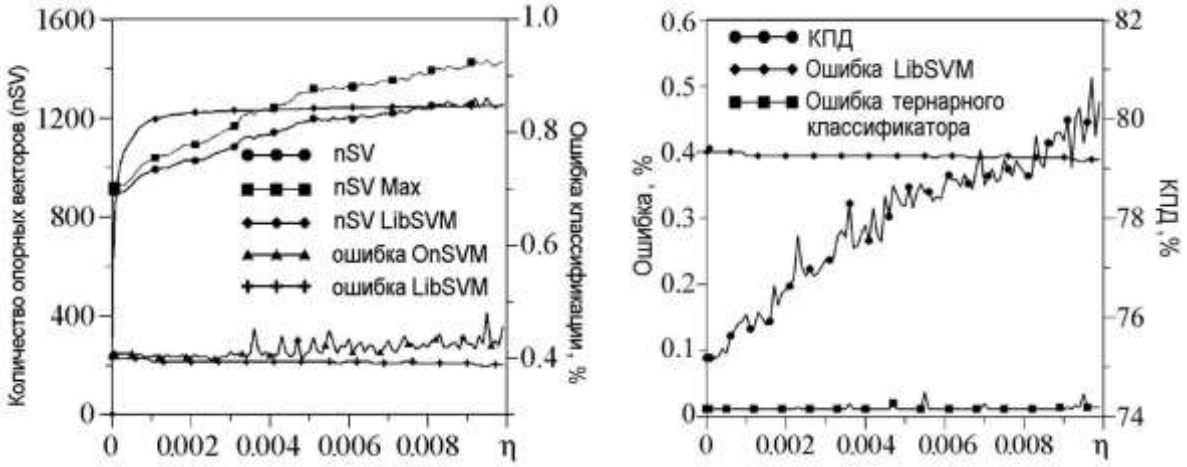
Таблица 3.8 — Сравнительные результаты работы алгоритма для набора Pendigits

	OnSVM					**LIBSVM**	
Num	**nSV**	**nSV Max**	E_2 **(%)**	E_3 **(%)**	R_3 **(%)**	**nSV**	E_2 **(%)**
0	1036	1046	0,60	0,00	74,80	1267	0,60
1	963	991	0,40	0,00	71,40	1267	0,34
2	1168	1285	0,11	0,00	79,54	1143	0,11
3	827	959	0,28	0,03	86,00	872	0,28
4	957	979	0,34	0,00	75,85	1218	0,37
5	884	906	0,48	0,00	74,70	1157	0,43
6	858	880	0,17	0,00	75,73	1186	0,14
7	934	966	1,03	0,00	73,10	1186	1,03
8	1117	1126	0,11	0,00	73,79	1354	0,11
9	1032	1046	0,49	0,08	72,73	1291	0,51
Total	978.1	1018.4	0.4	0.01	75.76	1149.1	0.39

В Таблице 3.7 и Таблице 3.8 приведены также результаты для ошибок распознавания в режиме тернарного классификатора. Возрастание обобщающей способности алгоритма в этом режиме работы классификатора может значительно повышаться за счет ухудшения эффективности классификатора – уменьшения количества примеров, по которым классификатор принимает решение о присвоении метки класса.

Параметр nSV Max позволяет оценить максимальный объем памяти, необходимый для кеширования ядровой матрицы. Кеширование ядровых функций значительно ускоряет процесс обучения, но размер памяти, требующийся для кеша, пропорционален квадрату количества обучаемых векторов. В алгоритме *OnSVM* необходимый размер памяти пропорционален параметру nSV Max. В проведенных экспериментах этот параметр не отличался от общего количества опорных векторов более чем на 5%.

На Рис. 3.5 показаны результаты усреднения измерений параметров 10-ти классификаторов на наборе данных Pendigits для каждой из цифр. Рис. 3.5a показывает зависимость количества опорных векторов от параметра η в сравнении с реализацией *LIBSVM* и зависимость ошибки E_2 от η рассматриваемых алгоритмов. Рис. 3.5b показывает зависимости ошибки E_3 и эффективности R_3 от параметра алгоритма η.



a) b)

Рисунок 3.5 — Сравнение качества алгоритмов OnSVM и LIBSVM в зависимости от параметра η: a) количества векторов и ошибки бинарного классификатора b) эффективности и уровня ошибок тернарного классификатора

Таблица 3.9 - сводная таблица сравнения рассматриваемых методов на базах данных Optdigits и Pendigits, а также на искусственно синтезированной базе данных Waveform [47], которая была получена в результате 100 экспериментов, в каждом из которых использовались обучающая и тестовая базы из 4000 и 1000 примеров соответственно при значениях параметров алгоритма $\eta = 0.001, \gamma = 0.005, \tau = 0.001$. Здесь же приведено количество примеров для каждой из трех рассмотренных групп.

Таблица 3.9 — Сводная сравнительная таблица различных наборов данных

	OnSVM		**LIBSVM**		**Datasize**	
Dataset	**nSV**	E_2 **(%)**	**nSV**	E_2 **(%)**	**Train**	**Test**
Pendigits	978.1	0.4	1149.1	0.39	7494	3823
Optdigits	107.2	0.57	140.7	0.5	3823	1797
Waveform	377	12.5	444	12.9	4000	1000

3.4 Применение алгоритмов построения экспертных комитетов в задаче определения атрибутов лица на изображении

В рамках исследования была проведена серия вычислительных экспериментов с целью проверки эффективности предложенных методов построения ансамблей экспертов для распознавания атрибутов лица.

Распознавание лицевых атрибутов представляет собой сложную задачу, обусловленную широкими вариациями внешнего вида лиц зависящего от возраста, позы, освещения, выражения лица, а также разнообразием этнических характеристик. Системы классификации пола, возраста и этнической принадлежности традиционно требуют наличия сложных признаков, больших обучающих выборок и значительных вычислительных ресурсов. Кроме того, во многих приложениях, таких как системы реального времени, встраиваемые устройства и мобильные приложения, особенно важно снижать вычислительные затраты без существенного ухудшения качества распознавания.

Распознавание лицевых атрибутов включает определение характеристик человека по его изображению, таких как пол, возраст и этническая принадлежность. На сегодняшний день в этой области было предложено множество методов, использующих различные подходы к извлечению признаков и обучению классификаторов.

Распознавание пола является одной из наиболее исследованных задач в области обработки изображений лиц. Ранние методы полагались на глобальные признаки, такие как гистограммы интенсивностей или преобразования Фурье. Однако эти признаки оказались недостаточно устойчивыми к изменениям освещения и ракурса. Более современные подходы используют локальные дескрипторы, такие как локальные бинарные паттерны (LBP) [33] и гистограммы ориентированных градиентов (HOG) [60, 88], которые доказали свою эффективность при распознавании пола. С появлением методов глубокого обучения, сверточные нейронные сети (CNN)

также стали активно применяться для этой задачи [126, 154], демонстрируя высокую точность, но требуя значительных вычислительных ресурсов.

Определение возраста человека по изображению лица также является сложной задачей, поскольку лицо меняется непрерывно в процессе старения. Методы возрастной классификации можно условно разделить на две категории:

- Классификация на основе заранее заданных возрастных групп [89],
- Регрессионные методы для предсказания точного возраста [78].

Традиционные признаки включают текстурные характеристики кожи, геометрические пропорции лица и комбинацию локальных дескрипторов. В последние годы также применяются методы глубокой регрессии с использованием сверточных сетей [152].

Определение этнической принадлежности по изображению лица основывается на выявлении текстурных и морфологических особенностей, характерных для разных этнических групп. В работах [129, 131] было показано, что методы, основанные на локальных признаках, таких как LBP и MCT, хорошо подходят для выделения этнических особенностей, поскольку они улавливают микро-структуры кожи, форму глаз, губ и носа. Использование ансамблей классификаторов также демонстрирует высокую эффективность при решении этой задачи [144].

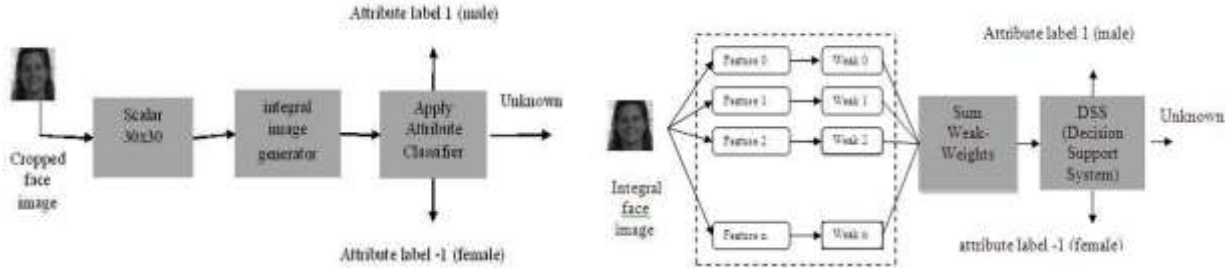
3.4.1 Распознавания атрибутов лица с помощью ансамбля экспертов

В работе [41] представлена система для обнаружения человеческого лица и распознавания его атрибутов, таких как пол, возраст и этническая принадлежность, с использованием классификатора на основе экспертного комитета. Центральный вклад данной работы заключается в разработке функции классификации атрибутов изображения, которая принимает произвольный фрагмент изображения и отображает его в пространство состояний $\{-1, 0, 1\}$. Полный список исследуемых атрибутов изображения лица представлены в Таблице 3.10.

Таблица 3.10 — Типы атрибутов для классификации по лицу

Атрибут	**Метка 1**	**Метка -1**
Пол	Мужчина	Женщина
Возраст	Взрослый	Ребенок
Этнос	Европеец	Азиат

Функция распознавания атрибутов представляет собой один экспертный комитет, работающий с данными, полученными из простых признаков MCT исходного изображения.



a) b)

Рисунок 3.6 — Блок-схема работы системы распознавания атрибутов лица (a) и . Структура сильного классификатора (b)

Процесс работы распознавания изображён на блок-схеме (см. Рис 3.6 a):

1. Сначала размер найденного фрагмента лица нормализуется до 30×30 пикселей,

2. Затем изображение преобразуется в интегральную форму для ускорения вычислений,

3. Интегральное изображение подаётся на вход сильного классификатора, который принимает одно из трёх решений.

Финальное решение формируется на основе структуры комитета, как показано на Рис 3.6 b.

Слабый классификатор рассматривается как функция, которая принимает код C, полученный с помощью MCT (см. Приложение Б) и отображает его в пространство решений. Функция принятия решения слабого классификатора представляет собой простую таблицу соответствия индексов

и ответов. Для конкретного преобразования значение таблицы с индексом m заполняется по следующим правилам:

1. Если частота появления любого из классов меньше заранее заданного порога L, то выбирается ответ 0. Этот порог необходим, чтобы отсечь редкие случаи появления шаблона. В экспериментах выбрано значение $L = 0.1$.

2. Если доля представленности одного из классов превышает порог P=0.9, то выбирается наиболее вероятный класс, иначе – неопределенный вариант ответа 0.

Принятие решение по определению атрибута лица осуществляется с помощью экспертного комитета в виде (3.3). Для каждого типа атрибутов построен соответствующий классификатор, базирующийся на общем наборе из ~700 преобразований МСТ (Приложение Б). Этот набор получен различными смешениями, масштабами и соотношениями сторон, производящимися в окне 30x30 пикселей.

3.4.2 Методика экспериментов

Для построения классификаторов и оценки качества классификаторов была собрана совокупность изображений лиц человека из различных источников: сети Интернет, базы данных FERET [104] и FG-NET [145]. Характеристики используемых баз данных приведены в Таблице 3.11:

Таблица 3.11 — Используемые базы данных для построения классификаторов атрибутов

База данных	**Атрибут метки 1**	**Ко-во изображений**	**Атрибут метки −1**	**Кол-во изображений**
FERET	Мужчины	785	Женщины	461
FG-NET+	Взрослые	722	Дети	905
RUS-IND	Европейцы	910	Азиаты	646

На каждом изображении осуществлялось обнаружение лица с помощью детектора лиц и последующей корректировкой результата детекции вручную.

Вырезанные и масштабированные изображения лица затем использовались для построения соответствующих классификаторов. Все классификаторы были обучены по следующей схеме: 70% выборки использовались для обучения, 30% — для тестирования.

Эксперимент состоит из следующих шагов, выполняемых для каждого детектора атрибута:

1. Проводится S циклов случайного разбиения датасета на тестовую и тренировочную выборки.

2. На каждой тренировочной выборке s обучается классификатор с помощью метода Алгоритм 3.1. для максимального числа слабых тернарных классификаторов K.

3. Строится зависимость средней ошибки (доли не верных ответов для оптимального порога $\langle E_2 \rangle$) от числа классификаторов на тренировочной и тестовой базе по всем разбиениям S.

4. Для K' слабых тернарных классификаторов, полученных на шаге 2 на выборке s, обучается линейный классификатор в соответствии с Алгоритм 3.6, при этом сохраняются два среза результатов обучения: OnSVM-0.5 на первой половине тренировочной базы; OnSVM-1.0 на полной тренировочной базе.

5. Качество классификаторов OnSVM-1.0, OnSVM-0.5 проверяются на соответствующих тестовых частях s и усредняются по всем выборкам S.

Для определения качества классификаторов введены следующие метрики: точность бинарного классификатора при оптимальном пороге $(1 - \min_S E_2)$; средняя точность тернарного классификатора $(1 - \langle E_3 \rangle)$; средняя эффективность тернарного классификатора $\langle R_3 \rangle$. Шаги 4-5 введены для демонстрации возможности построения классификатора на части обучающей базы и возможности дообучения.

3.4.3 Результаты экспериментов

3.4.3.1 Классификатор пола

Классификатор пола был обучен и протестирован на базе изображений лиц FERET. Проведено $S = 58$ циклов обучения на различных случайно выбранных подвыборках из исходных данных. На каждом цикле обучен сильный классификатор с помощью алгоритма M-Adaboost с количеством итераций обучения $K = 300$ (Рис. 3.7).

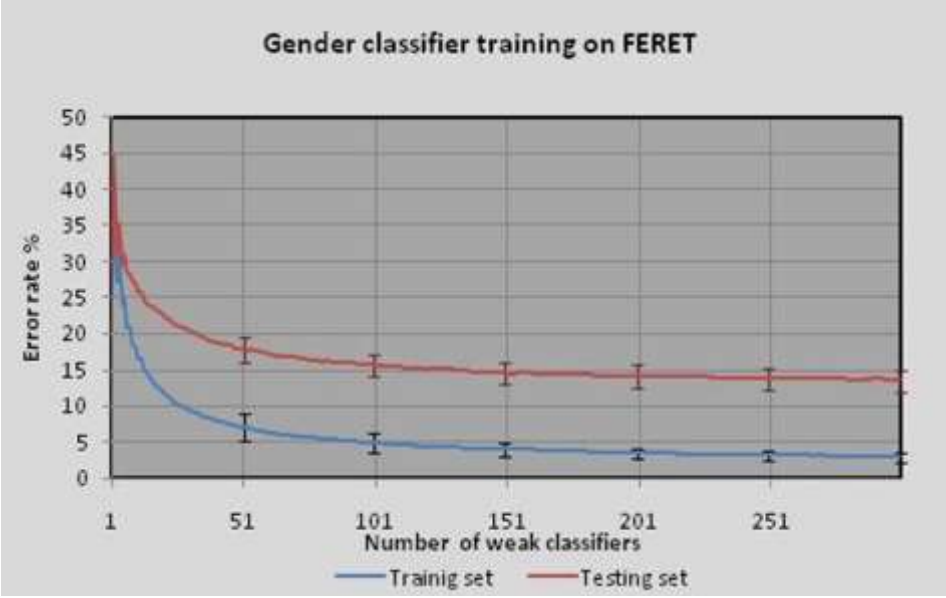


Рисунок 3.7 — Изменение ошибки $\langle E_2 \rangle$ в зависимости от количества слабых тернарных классификаторов на базе FERET

После использования приблизительно 250 слабых признаков ошибка на тестовой выборке переставала снижаться (Рис. 3.7). При этом ошибка на обучающей выборке продолжала уменьшаться по мере добавления новых признаков. Для построения характеристик алгоритма OnSVM выбран параметр $K' = 250$. Усредненные результирующие характеристики классификаторов (Таблица 3.12):

Таблица 3.12 — Качество алгоритмов в задаче классификации пола на тестовой части базы FERET

Алгоритм	**Бин. точность %**	**Тер. точность %**	**Эффективность %**
M-Adaboost	89	96	92
OnSVM-1.0	**92**	97	94
OnSVM-0.5	80	91	85

Пример работы классификатора пола на реальных изображениях показан на Рис. 3.8.

Рисунок 3.8 — Пример работы классификатора пола: синим цветом отмечены фрагменты, классифицированные как мужчины; красным цветом — женщины

3.4.3.2 Классификатор возраста

Классификатор возраста был обучен и протестирован на объединённой базе изображений, состоящей из FG-NET и собственной базы изображений. Процедура разбиения данных на обучающую и тестовую выборки также повторялась $S = 58$. В процессе обучения классификатор M-Adaboost обновлялся в течение $K = 500$ итераций.

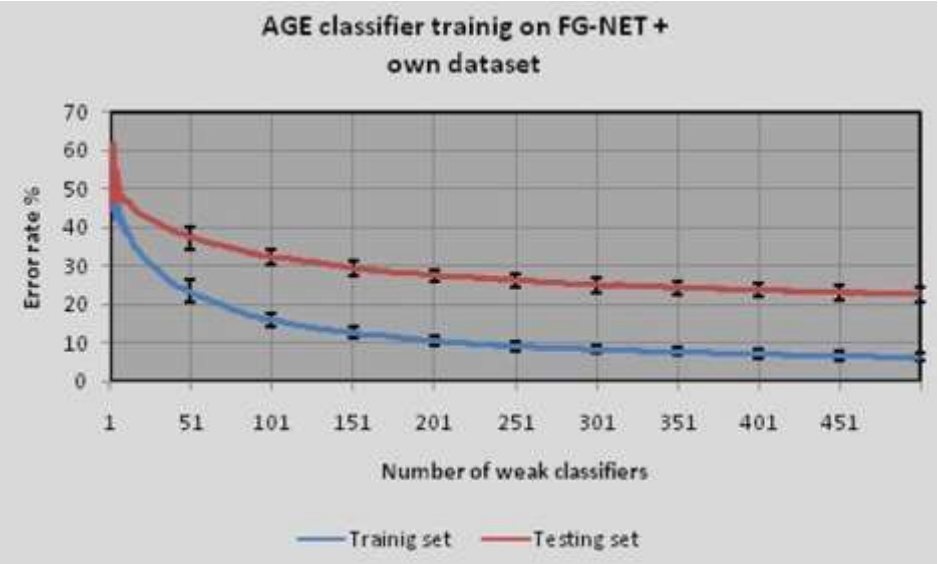


Рисунок 3.9 — Изменение ошибки $\langle E_2 \rangle$ в зависимости от количества слабых тернарных классификаторов на базе FG-NET+

Ошибка на тестовой выборке снижалась медленнее, чем в случае классификации пола (Рис. 3.9). Чётко выраженного плато (стабилизации ошибки) после 500 итераций достигнуто не было. На дальнейших шагах использовался параметр $K' = 400$, результирующие (Таблица 3.13).

Таблица 3.13 — Качество алгоритмов в задаче классификации возраста на тестовой части базы FG-NET+

Алгоритм	**Бин. точность %**	**Тер. точность %**	**Эффективность %**
M-Adaboost	80	94	83
OnSVM-1.0	**81**	94	86
OnSVM-0.5	74	88	80

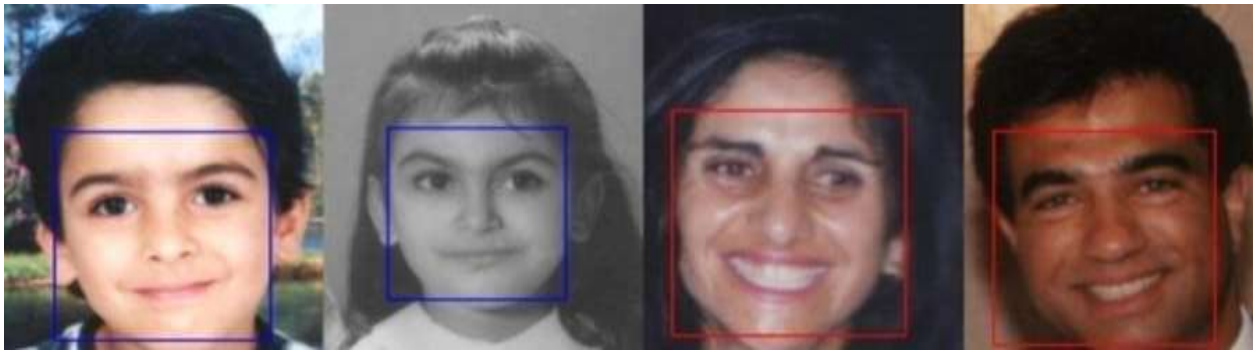


Рисунок 3.10 — Пример результата работы классификатора возраста: синим цветом выделены дети; красным цветом — взрослые

3.4.3.3 Классификатор этнической принадлежности

Для задачи классификации этнической принадлежности использовались изображения из базы Indian Face Database и собственной базы данных. Процесс разбиения повторялся $S = 20$ раз. Классификатор обучался с использованием $K = 200$ итераций отбора слабых признаков. Ошибка тестирования быстро стабилизировалась при сравнительно небольшом количестве признаков, поэтому на дальнейших шагах использовался параметр $K' = 150$ (Рис. 3.11).

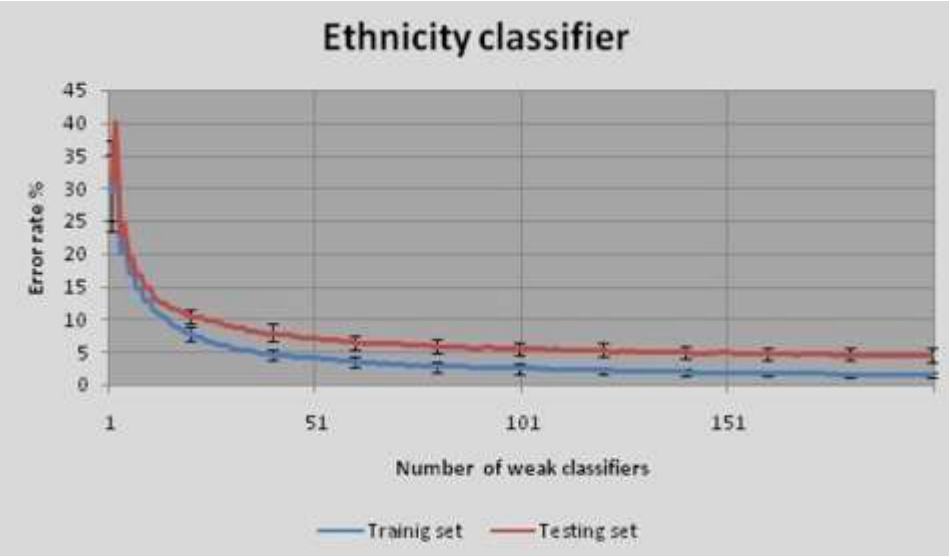


Рисунок 3.11 — Изменение ошибки $\langle E_2 \rangle$ в зависимости от количества слабых тернарных классификаторов на базе RUS-IND

Таблица 3.14 — Качество алгоритмов в задаче классификации этнической принадлежности на тестовой части базы RUS-IND

Алгоритм	**Бин. точность %**	**Тер. точность %**	**Эффективность %**
M-Adaboost	**96**	98	97
OnSVM-1.0	**96**	98	97
OnSVM-0.5	94	96	97

Качество распознавания этнических признаков (Таблица 3.14) оказалось лучше, чем в задачах классификации пола и возраста.

3.5 Выводы

Проведённое исследование алгоритмов построения экспертных комитетов на основе тернарных классификаторов и метода последовательного обучения OnSVM позволило сформулировать следующие обобщающие выводы:

1. Предложенный последовательный алгоритм OnSVM продемонстрировал хорошую сходимость к решению классического ν-SVM, обеспечивая достижение приемлемого качества за один проход по обучающим примерам. Важным преимуществом данного подхода является компактность итоговых моделей: число опорных векторов в среднем оказывается на 20-30% ниже по сравнению с традиционными SVM-алгоритмами, при сопоставимом уровне точности (Таблица 3.7 , Таблица 3.8). Это обстоятельство существенно снижает вычислительные затраты и требования к памяти.

2. Использование тернарных классификаторов в рамках алгоритма OnSVM позволило дополнительно уменьшить ошибки распознавания за счёт введения промежуточного («неопределённого») ответа. Несмотря на некоторое снижение общей вычислительной эффективности, такой подход оправдан повышением точности распознавания (Рис. 3.5).

3. Онлайновая природа OnSVM означает, что система готова к распознаванию новых объектов уже после первоначальной инициализации – дальнейшее обучение может происходить параллельно с процессом распознавания, адаптируя модель по мере поступления новых данных. Это свойство было продемонстрировано на базах классификации атрибутов лица (Таблица 3.12, Таблица 3.13, Таблица 3.14).

4. Применение алгоритма M-AdaBoost для построения сильных классификаторов из ансамбля слабых тернарных признаков позволило выявить минимально необходимый набор признаков, обеспечивающий оптимальную надёжность распознавания. Данный подход характеризуется

низкой вычислительной сложностью на этапе эксплуатации классификатора и доказанной устойчивостью к переобучению (Рис. 3.7, Рис. 3.9, Рис. 3.11).

5. Было установлено, что введение тернарного классификатора в сильный ансамбль позволяет повысить итоговое качество классификации на 2–6%, главным образом благодаря корректной обработке «неопределённых» ситуаций (Таблица 3.12, Таблица 3.13, Таблица 3.14).

Таким образом экспериментально подтверждены преимущества предложенных подходов на основе последовательного алгоритма OnSVM и ансамблей тернарных слабых классификаторов с использованием метода M-AdaBoost – высокая точность, экономичность реализации и устойчивость к переобучению. Тернарные классификаторы, позволяют повысить надёжность за счёт введения нейтрального исхода. Алгоритм OnSVM демонстрирует, как идеи ансамблей могут быть перенесены на уровень опорных векторов и онлайн-обучения, обеспечивая быструю адаптацию модели. Это открывает возможности применять OnSVM в задачах реального времени и на больших потоках данных, а тернарная схема решения – адаптировать модель под неоднородные данные, возлагая тяжёлые случаи на последующие ступени обработки. В задаче детекции объектов комбинация этих идей приводит к системе, которая точнее и гибче классических детекторов вроде каскада Виолы–Джонса. Такой каскад может со временем расширяться новыми «экспертами», справляясь с всё более сложными случаями, не теряя при этом в производительности за счёт многоступенчатой фильтрации простых негативных примеров на ранних этапах, что необходимо при дрейфе характеристик распознаваемого объекта со временем или корректировки системы распознавания на новые объекты.

ГЛАВА 4. КОРРЕКЦИЯ ОШИБОК В ПРИНЯТИИ РЕШЕНИЙ

Современные системы искусственного интеллекта (ИИ) всё чаще внедряются в различные сферы человеческой жизни — от медицины и автономного транспорта до финансов и промышленности. Несмотря на многочисленные успехи в области машинного обучения и глубокого обучения, все системы ИИ неизбежно совершают ошибки. Эти ошибки представляют серьёзную угрозу для безопасного и эффективного применения ИИ в реальных сценариях, где точность и надёжность являются критически важными. Причины ошибок многообразны:

Сложность реального мира: В практике машинного обучения часто наблюдается отклонение от предположения о независимости и одинаковом распределении выборок (H.O.P.), лежащего в основе большинства теоретических моделей [59, 93]. Согласно этому предположению, существует фиксированное вероятностное распределение на пространстве данных, из которого наблюдения извлекаются независимо друг от друга. При этом обучающая выборка в задаче с учителем включает как входные данные, так и соответствующие целевые значения, а вероятность задаётся на объединённом пространстве входов и выходов.

Следует подчеркнуть, что гипотеза существования стационарного распределения вероятностей в условиях реальной среды является крайне сильным допущением. Причины этому: устаревание данных во времени, изменение условий окружающей среды, появление новых типов объектов, отличающихся от обучающих примеров, или систематические отличия в выборке. В связи с этим были предложены различные вспомогательные концепции, такие как дрейф распределения данных, позволяющие ослабить данное требование [116]. Несмотря на это, предпосылка H.O.P. по-прежнему занимает ключевое место в статистической теории обучения, в рамках которой выборка рассматривается как независимая и одинаково распределённая реализация из фиксированного распределения.

Неопределённость данных: обучающие наборы данных не охватывают всех возможных ситуаций, с которыми система может столкнуться в реальности. Так, датасеты могут содержать случайные ошибки измерений или маркировки (алеврическая неопределённость), а модель может быть не способна адекватно обобщить вне обучающей выборки (эпистемическая неопределённость). Также выделяют эффект смещение выборки (selection bias), который приводит к тому, что обучающие данные не представляют корректно целевую среду [127]. Ещё одна известная причина – эффект “ускоренного обучения” (shortcut learning) - модель находит в данных простые корреляции, позволяющие успешно проходить тесты, но эти корреляции не связаны с глубинной сущностью задачи [141]. В результате система демонстрирует высокие результаты на привычных бенчмарках, однако почти не умеет обобщать решение при малейшем изменении условий задачи.

Необъяснимость ошибок: выявление источника и причины ошибки часто затруднено, что усложняет их устранение. Такие ошибки могут быть как программными так и появляться в результате неожиданного человеческого поведения и непреднамеренного использования, а также возможны многие другие возможные причины. Например – ошибки, вызванные специально подобранных, противостоящих (adversarial) примеров. Было показано, что добавление к исходным данным небольших, на первый взгляд незначительных, возмущений может привести к сбою в работе даже высокоточных моделей [190].

Многие подходы к машинному обучению основаны на исправлении ошибок. Хорошо известным примером является обратное распространение ошибок, начиная с классических алгоритмов персептрона [182] до современных методов глубокого обучения [79]. Необходимость исправления ошибок ИИ обсуждается в литературе по обучению с подкреплением. В области обучения с подкреплением, основанного на моделях, мотивация заключается в неизбежных расхождениях между моделями окружающей

среды, используемыми для обучения агента, и реальностью, в которой действует этот агент.

Существует несколько подходов для борьбы с ошибками в системах ИИ. Один из наиболее очевидных методов заключается в систематическом повторном обучении системы ИИ с учётом выявленных ошибок. Однако этот метод имеет ряд существенных недостатков:

1. Для сохранения существующих навыков система должна заново обучаться на полном наборе исходных данных.
2. Повторное обучение требует значительных вычислительных ресурсов и времени.
3. В процессе переобучения могут появиться новые ошибки.
4. Сохранение текущих навыков не гарантировано.

Эти ограничения делают метод переобучения неэффективным для оперативного исправления ошибок в реальных условиях. Альтернативой полному переобучению является подход с использованием корректоров [82, 84] . Корректоры представляют собой внешние устройства или алгоритмы, которые дополняют существующие системы ИИ, диагностируют ошибки и предоставляют исправленные выходные данные. Преимущество корректоров заключается в их гибкости и обратимости: оригинальная система ИИ остаётся неизменной и может быть легко восстановлена при необходимости.

4.1 Метод коррекции алгоритмов распознавания

Большинство методов исправления ошибок полагаются на итеративное обучение на больших выборках, чтобы предотвратить появление новых ошибок. В отличие от них, метод корректоров ориентирован на неитеративную коррекцию на основе одного или нескольких размеченных примеров — особенно в условиях высокой размерности.

Создание корректора ИИ требует набора размеченных ошибок, с помощью которого обучается бинарный классификатор, различающий нормальное поведение и ситуации с высокой вероятностью ошибки. Такая

задача сводится к обучению с одним или несколькими примерами, когда один из классов (ошибки) представлен ограниченным числом меток.

Проблема обучения по малому числу примеров является одной из ключевых в машинном обучении [179]. Обучение по одному примеру (one-shot learning) формирует описание класса на основе одного размеченного примера, тогда как обучение по нескольким примерам (few-shot learning) подразумевает, что классификатор обобщает новые классы используя только небольшое количество примеров [161].

Современные подходы к реализации обучения на ограниченных примерах часто требуют предварительной тренировки на задачах, схожих, но не идентичных новым задачам [150, 181]. После такой подготовки система приобретает мета-навыки: она может обучаться решению новых задач, которые существенно не отличаются от предыдущих, без необходимости в больших наборах данных и длительном обучении [161, 168].

Эффективность таких подходов и методов часто достигается благодаря либо **снижению размерности** данных, либо эффекту **«благословения размерности»** [83, 177].

- При снижении размерности отбирается ограниченное число признаков, достаточных для принятия решений по одному или нескольким примерам.
- Эффект благословения размерности [38, 106] означает, что при соблюдении условий регулярности распределений [81, 82] могут быть применены классические методы принятия решений (например, метод опорных векторов, линейный дискриминант Фишера и т.д.).

Используя описанные предпосылки для создания эффективных критериев построения классификаторов по малым данным, в работе [80] предложена концепция корректоров. Корректор включает в себя бинарный классификатор, который разделяет ситуации с высоким риском ошибки и нормальное функционирование системы. Для этого используется обучающий набор, включающий:

- **Класс «норма»** (ситуации без ошибок).
- **Класс «ошибки»** (ситуации с размеченными ошибками).

Таким образом, корректоры позволяют осуществлять локальное исправление ошибок без необходимости полного переобучения системы.

Общая схема алгоритма коррекции приводится на Рис 4.1. Корректор, используя входные сигналы и сигналы о внутреннем состоянии детектора предсказывает область подлежащую корректировке и ожидаемый результат корректировки.

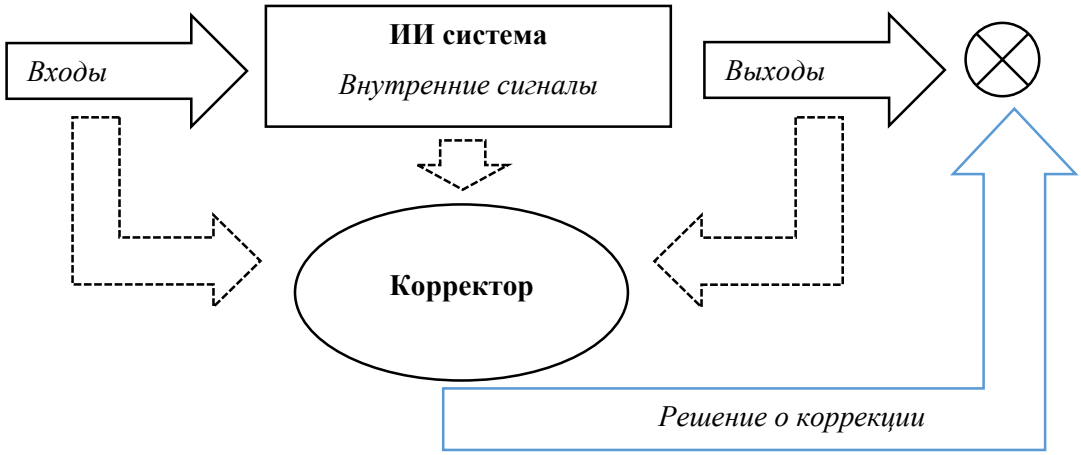


Рисунок 4.1 — Схема встраивания корректора в существующий детектор

Корректор получает на вход сигналы от системы ИИ (входные данные, внутренние сигналы и выходы) и принимает решение о необходимости коррекции.

Алгоритм коррекции:

1. **Получение данных от базового классификатора**: Корректор получает входные данные, внутренние сигналы и выходы системы.
2. **Классификация ошибок**: Корректор использует стохастическое разделение для классификации ситуаций с высоким риском ошибки. Применяется классификатор для разделения ситуаций на нормальные и потенциально ошибочные.

3. **Принятие решения о коррекции**: Если ситуация определяется как потенциально ошибочная, применяется классификатор или регрессор для корректировки выходного результата.
4. **Использование корректоров для новых классов**: Корректор обучается на новых данных, использует одноразовое обучение для выявления и корректировки новых ошибок.
5. **Многокорректорная система**: Для сложных задач используется многокорректорная система, где несколько корректоров специализируются на разных типах ошибок, а диспетчер распределяет задачи между ними.

В работе [80] предложено отойти от классической гипотезы регулярности распределения данных и предположить, что данные обладают мелкозернистой структурой с множеством кластеров и соответствующих пиков плотности вероятности. В работах [80, 81] были разработаны теоремы стохастического разделения [82] для данных с мелкозернистой кластерной структурой, позволяющие заменить вероятностный подход на функциональный анализ с предельным переходом к бесконечным размерностям. Эти теоремы обосновывают возможность применения линейных методов, таких как дискриминант Фишера, для разделения данных в высоких размерностях представленных в виде дискретных, сферических или эллиптических гранул при выполнении условий на плотности распределений и размеры множеств.

4.2 Коррекция детекторов для задачи детектирования объектов на изображении

Применяя идеи и принципы, изложенные в предыдущих разделах, разработана биоморфная система для семантического анализа [118, 165], которая использует индивидуальные каскадные детекторы для каждого концепта из своего словаря, применяя сильные классификаторы [180], основанные на нелокальных бинарных шаблонах [72]. Детекторы

организованы последовательно, используя многоступенчатую технику обнаружения, при которой каждый детектор использует каскад связанных сильных классификаторов.

4.2.1 Детектирование объектов с помощью каскадного детектора

Алгоритм каскадного детектора [180] зарекомендовал себя как один из эффективных методов детектирования объектов (например, лиц) на изображениях благодаря использованию каскадов слабых классификаторов, обученных на наборе простых признаков. Алгоритм реализует идею скользящего окна, при этом каждый фрагмент изображения классифицируется как содержащий объект (1) или нет (0). Каскады признаков организованы в виде усеченного бинарного дерева, при этом ответ (0) любого сильного классификатора интерпретируется как ‘не объект’, и ‘объект’ в остальных случаях. Основные этапы работы алгоритма детектора можно описать следующим образом:

4.2.1.1 Признаковое пространство детектора

В классической версии детектора [180] используются признаки Хаара, которые вычисляются на основе разностей сумм интенсивностей в прямоугольных областях. В разработанном алгоритме используются признаки модифицированного Цензуса, принцип формирования и кодирования которых описан в Приложении Б. На основе такого кодирования получен слабый классификатор, для этого каждому коду бинарного паттерна ставится в соответствие ответ слабого классификатора {0, 1}, определяемый из сравнения откликов распределений на обучающей базе данных (см. Рис. 4.2).

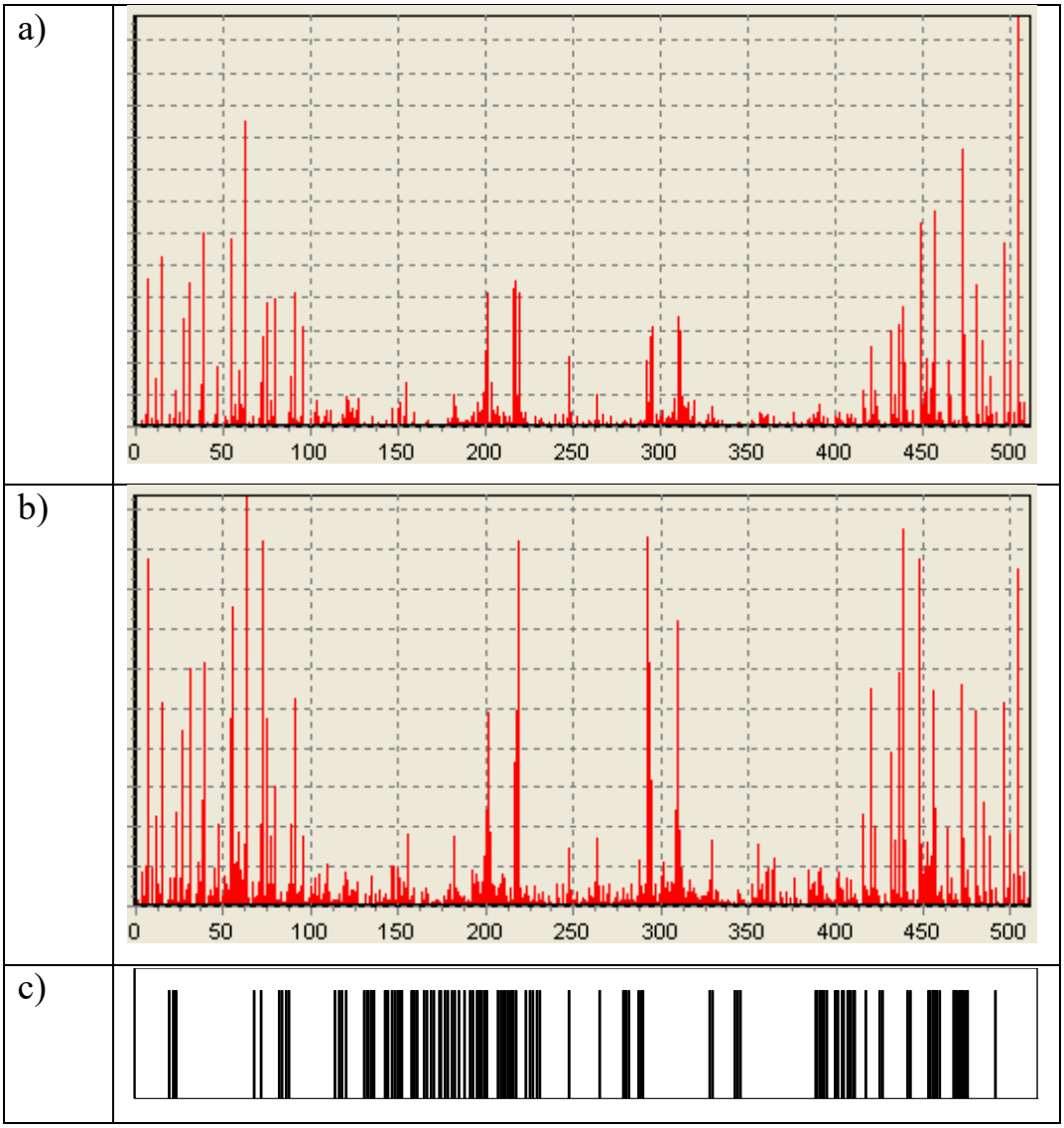


Рисунок 4.2 —Гистограммы распределения откликов преобразования Цензуса на a) фоновых фрагментах; b) искомых объектах и принятые решения c) на базе из 10000 фрагментов

4.2.1.2 Каскадная структура детектора

Каскад состоит из последовательности этапов, где каждый этап быстро отбрасывает большую часть негативов, оставляя для дальнейшей обработки лишь те фрагменты, которые могут содержать искомый объект. Если хотя бы один этап отклоняет фрагмент, дальнейшие вычисления для него не проводятся. «Сильный классификатор» создаётся с использованием алгоритма AdaBoost [70], объединяющего признаки и слабые классификаторы. Его

бинарная функция 0,1 определяется в процессе обучения, минимизируя ошибки распознавания в обучающей базе данных.

4.2.1.3 Сканирование изображения

Детектор использует метод скользящего окна, которое перемещается по изображению с заданными шагами s_x и s_y. Пусть изображение имеет размеры $W \times H$, а окно – размер $w \times h$, тогда координаты левого верхнего угла окна задаются как:

$$x_i = i \cdot s_x, y_j = j \cdot s_y, i = 0,1, \dots, \left\lfloor \frac{W-w}{s_x} \right\rfloor, j = 0,1, \dots, \left\lfloor \frac{H-h}{s_y} \right\rfloor \quad (4.1)$$

Для каждого окна R_{ij} вычисляются признаки, и с помощью каскадной архитектуры сильных классификаторов выполняется бинарная классификация $D(R_{ij}) \rightarrow \{0,1\}$,где “1” – означает обнаружение объекта, а “0” – отсутствие объекта. Эта процедура позволяет разбить изображение на множество перекрывающихся или неперекрывающихся фрагментов, в каждом из которых выполняется классификация

4.2.1.4 Масштабирование и соотношение сторон

Для учёта объектов разных размеров детектор применяется на множестве масштабов, а соотношение сторон окна может быть произвольным, что позволяет адаптироваться к различным типам объектов и условиям съемки. Использованная схема (Рис 4.3) обеспечивает высокую скорость работы, так как большинство окон быстро отбрасываются на ранних этапах каскада, а детальная проверка проводится лишь для потенциально релевантных фрагментов.

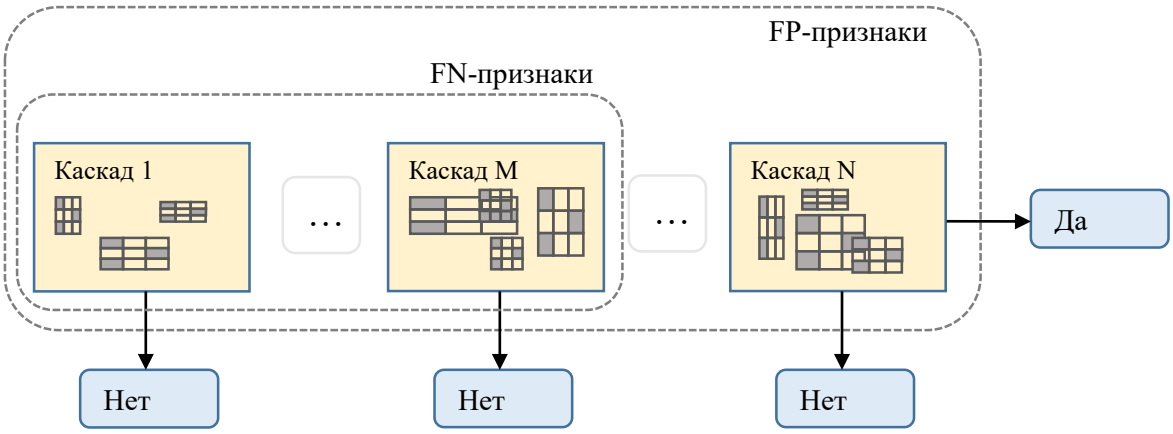


Рисунок 4.3 — Схема каскадного детектора с модифицированным преобразованием Цензуса

4.2.2 Постановка задачи корректировки детекторов

Детектор $D(R)$ может ошибаться, порождая два типа ошибок:

- Ложный пропуск (False Negative, FN): объект присутствует, но детектор выдал 0.
- Ложное срабатывание (False Positive, FP): объект отсутствует, но детектор выдал 1.

Для повышения точности детекции вводятся два типа корректоров:

- **Корректор FN:** обучается на ошибках, когда детектор не срабатывает (FN). При этом в качестве признаков используются данные, полученные только с K каскадов детектора, а также значения средней интенсивности внутри фрагмента, рассчитанные по модифицированному преобразованию Цензуса.
- **Корректор FP:** обучается на ложных срабатываниях (FP) и корректно определенных положительных примерах (TP). Для него используются признаки, аналогичные признакам для FN корректора, но учитываются все каскады детектора.

Для создания и применения корректоров во время работы детектора разработана схема обучения (Рис 4.4) [118]:

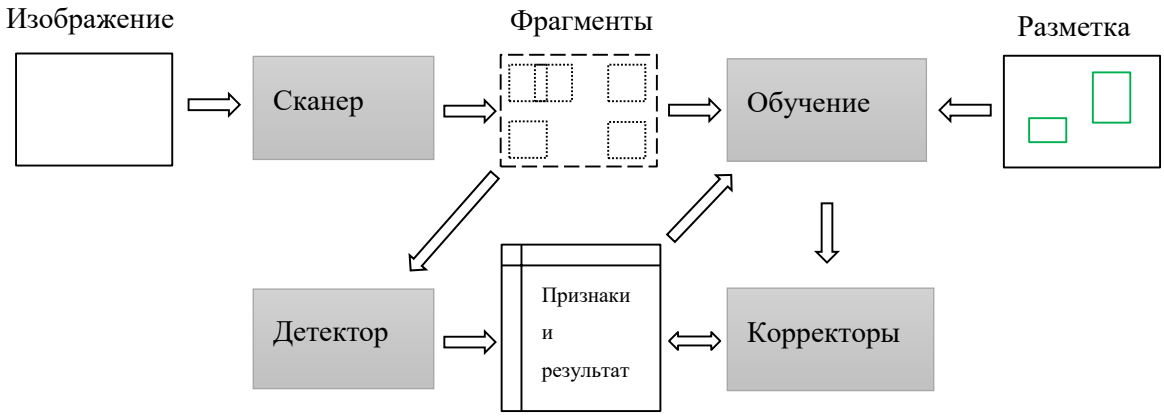


Рисунок 4.4 — Общая схема построения корректоров во время работы детектора

Схема на Рис. 4.4 позволяет получать корректоры в случае неполной разметки на входных изображениях. Фрагменты, используемые для обучения, накапливаются во внутренней базе процедуры обучения и используются в момент достижения количества примеров необходимого порога.

4.2.3 Отбор фрагментов для построения корректоров

Для обучения корректоров отбираются фрагменты на основе степени совпадения с разметкой (G). В качестве меры совпадения используется метрика относительного пересечения (IoU):

$$IoU(R, GT) = \frac{|R \cap G|}{|R \cup G|} \tag{4.2}$$

,где R – прямоугольный фрагмент, а G – соответствующая разметка (Рис. 4.5).

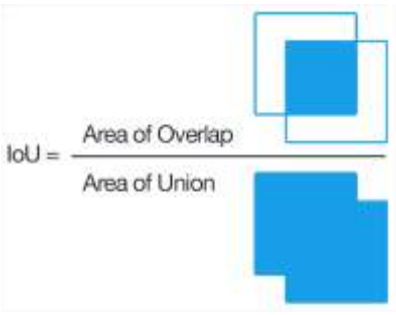


Рисунок 4.5 — Визуализация подсчета метрики пересечения фрагментов

4.2.3.1 Отбор примеров для FN корректора

Фрагмент R считается ошибкой детектора и отбирается для обучения корректора FN по следующему правилу:

$$M_{fn}^{i} = \begin{cases} \emptyset, \text{ если } \exists R: [IoU(R, G_i) > T_{hi} \wedge D(R) = 1] \\ \{ R \mid IoU(R, G_i) > T_{hi} \wedge D(R) = 0 \}, \text{ иначе} \end{cases} \quad (4.3)$$

Фрагменты, не содержащие ошибки (TN), выбираются с заранее заданной вероятностью p_{tn} при условии, что для любого G_i выполняется условие:

$$M_{tn}^{i} = \{ R \mid IoU(R, G_i) < T_{lo} \wedge D(R) = 0 \} \quad (4.4)$$

Параметр p_{tn} необходим для сокращения количества фрагментов tn, в случаях, когда размеченных фрагментов намного меньше фоновых и балансировки базы данных для обучения корректора. (4.3) (4.4)

4.2.3.2 Отбор примеров для FP корректора

Для построения FP корректора используются два типа фрагментов. Первый тип – это ложные срабатывания, которые определяются с помощью порога пересечения с разметкой:

$$M_{fp}^{i} = \{ R \mid IoU(R, G_i) < T_{lo} \wedge D(R) = 1 \} \quad (4.5)$$

Второй тип фрагмента – истинно положительные, используемые в качестве позитивных примеров для корректора.

$$M_{tp}^{i} = \{ R \mid IoU(R, G_i) > T_{hi} \wedge D(R) = 1 \} \quad (4.6)$$

В обучающий набор они также включаются с вероятностью p_{tp}. Визуализация отбора фрагментов для построения корректоров по правилам (4.3), (4.4), (4.5) и (4.6) изображена на Рис 4.6.

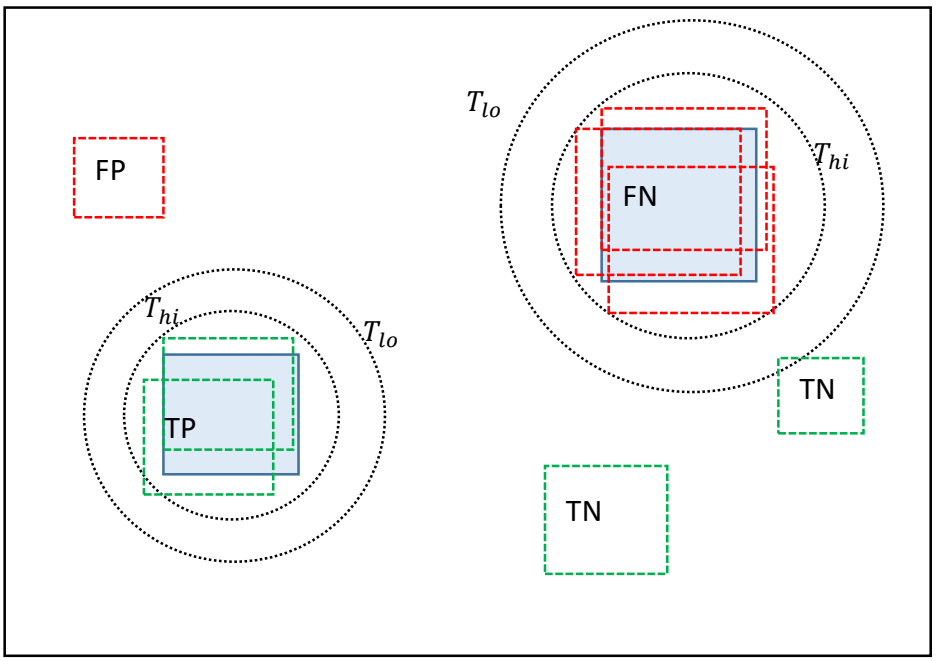


Рисунок 4.6 — Визуализация процесса сбора фрагментов для построения корректоров

Обучающие выборки для корректоров собираются до тех пор, пока их размер не достигнет заранее заданного порога N_{fn} N_{fp} для корректоров FN и FP соответственно.

4.2.4 Построение корректоров

Корректоры FN и FP принимают на вход объединенный вектор признаков $\boldsymbol{x}$ для каждого фрагмента R и формируют бинарный ответ о принадлежности фрагмента к корректируемым множествам. Для этого применяется алгоритм, разработанный для пространств высокой размерности, который на основе анализа различимости классов (ошибок и корректных решений) строит разделяющую гиперплоскость [167].

Используя теоретическое обоснование возможности применения критерия Фишера в задаче построения корректора базового классификатора, приведен общий алгоритм получения корректоров (Алгоритм 4.1), который строит разделяющую гиперплоскость между множеством ошибок X и множеством верных ответов Y.

Алгоритм 4.1: Алгоритм построения корректора

1. Заданы множества X, Y; количество кластеров k; количество главных компонент m; порог θ (или вектор порогов $(\theta_1, \dots, \theta_k)$, дискриминанты Фишера $(w_1, \dots, w_k)$, центроиды матриц H и W.
2. Вычислить центр тяжести x_c множества X Сформировать централизованные множества X_c и Y^*, полученные путем вычитания центра тяжести из каждого элемента X, Y.
3. Извлечь главные компоненты из X_c.
4. Выбрать m компонентов из n, соответствующих наибольшим собственным значениям $(\lambda_1 \geq \cdots \geq \lambda_m > 0)$ ковариационной матрицы X_c, и спроецировать X_c и Y^* в пространство главных компонентов (получив X_r и Y_r^* соответственно), и построить матрицу перехода H.
5. Построить матрицу W:

   $$W = \mathrm{diag}(\lambda_1, \dots, \lambda_m)$$

   соответствующую операции отбеливания для множества X_r, и применить эту трансформацию к X_r и Y_r^* (получив X_w и Y_w^*).
6. Разбить X_w на k кластеров с помощью алгоритма k-средних. Центроиды соответствующих кластеров — это $c_1, \dots, c_k$.
7. Для каждой пары $\left(X_w^i, Y_w^*\right)$, где i — номер кластера, вычислить дискриминант Фишера $(w_1, \dots, w_k)$ и соответствующие пороги $(\theta_1, \dots, \theta_k)$.

Алгоритм 4.1 преобразует данные в многокластерное признаковое пространство, где задача состоит в минимизации ошибок в каждом из кластеров. Для оценки полученного пространства признаков строились распределения измерений в изучаемом кластере Рис 4.7.

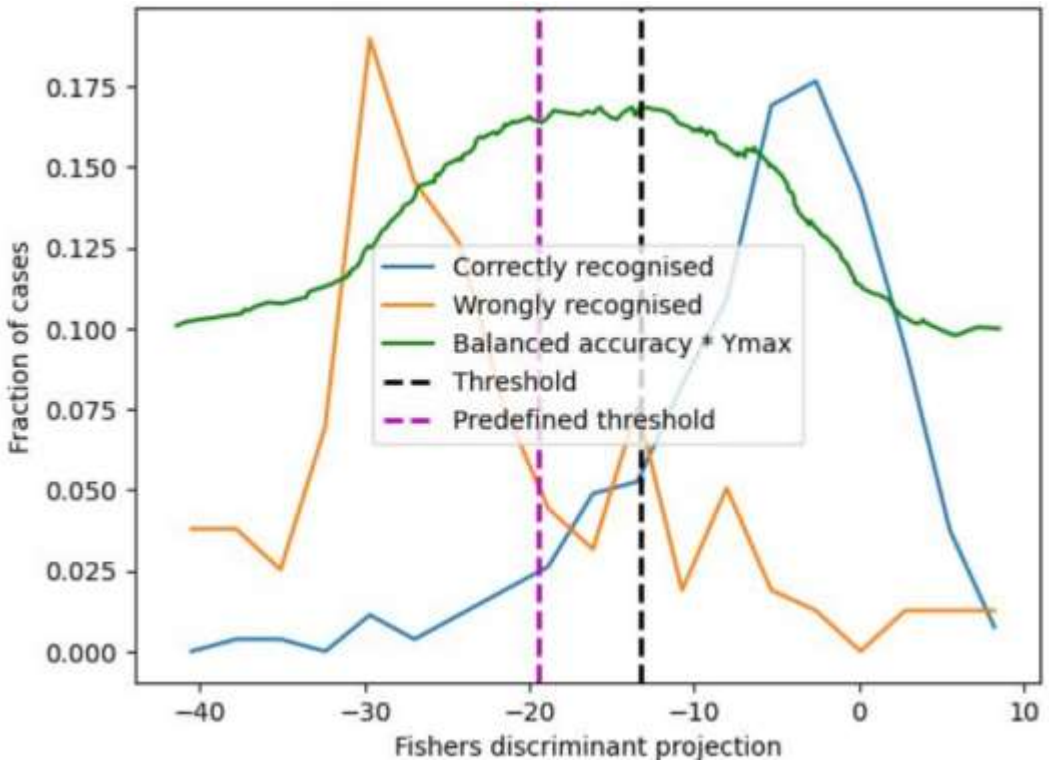


Рисунок 4.7 — Пример распределений семплов TP и FP после применения корректора. (Синяя и оранжевая кривые соответственно)

Как только корректор построен на обучающих данных, алгоритм его применения следующий:

Алгоритм 4.2: Алгоритм коррекции

1. Заданы входное измерение z; центр тяжести x_c; центроиды $c_1, \dots, c_k$; вектор порогов $(\theta_1, \dots, \theta_k)$; дискриминанты Фишера $(w_1, \dots, w_k)$; матрицы перехода H и W.
2. Вычесть центр тяжести x_c и применить преобразования перехода H и W.
3. Определить с помощью Евклидового расстояния ближайший t кластер из набора $c_1, \dots, c_k$.
4. Определить, принадлежит ли измерение к корректируемому множеству, если $(w_t, z) < \theta_t$.

Для построения корректоров используются признаки, извлекаемые из фрагментов изображения с помощью детектора. В качестве признаков берутся значения средней интенсивности сегментов, рассчитанные по модифицированному преобразованию Цензуса. Размер входного вектора для корректора определяется количеством признаков в каждом каскаде и количеством каскадов:

$$x(R) = [x_R^1, x_R^2, \dots, x_R^N],\ x_R^m = \left[\mu_R^{1,m}, \mu_R^{2,m}, \dots, \mu_R^{Z^m,m}\right] \quad (4.7)$$

, где $\mu_R^{k,m}$ –средняя интенсивность 9 сегментов слабого классификатора k на m-ом каскаде во фрагменте R.

Для FN корректора используются данные, полученные с первых K каскадов детектора. Для FP корректора используются признаки, полученные со всех каскадов, где $M \geq K$ – общее число каскадов детектора.

Обучающие выборки для корректоров собираются до достижения минимальных размеров N_{fn}и N_{fp}. Таким образом, корректоры обучаются на достаточно большом количестве примеров, что позволяет минимизировать риск переобучения и обеспечить высокую обобщающую способность.

Для повышения точности работы корректоров FN и FP предлагается дополнительно разбивать ошибки на группы (кластеры), поскольку ошибки, возникающие при работе детектора, часто имеют различную природу и отличаются по характеру признакового пространства. Кластеризация позволяет учитывать специфические особенности ошибок и строить отдельные гиперплоскости для каждого типа ошибок. Предположим, что имеется набор ошибок детектора (FN или FP), представленных в виде множества признаковых векторов:

$$X_{neg} = \{x_1, x_2, \dots, x_M\} \quad (4.8)$$

,где каждый признак x_i — это вектор значений признаков, полученных из фрагмента, соответствующего ошибке детекции.

Целью предварительной обработки является разбиение множества X_{neg} на заранее заданное число кластеров K:

$$X_{\text{neg}} = \bigcup_{j=1}^{K} X_j \, , X_i \cap X_j = \emptyset, \; i \neq j \tag{4.9}$$

Это разбиение (4.9) осуществляется с помощью алгоритма K-means [170], минимизирующего внутрикластерную дисперсию.

После разбиения ошибок на кластеры для каждого полученного кластера X_j строится отдельная разделяющая гиперплоскость, которая отделяет ошибки данного кластера от множества всех положительных (корректных) примеров X_{pos}.

Для каждого кластера решается отдельная задача дискриминации с использованием критерия Фишера.

4.2.5 Правило Фишера для принятия решения

В основе корректирующего алгоритма лежит правило Фишера для линейной дискриминации, используя основания [85]. Пусть имеется два класса: класс ошибок (например, FN или FP) с распределениями признаков $N(\mu_1, \sigma_1^2)$ и класс корректных фрагментов с распределениями $N(\mu_2, \sigma_2^2)$. Оптимальный вектор весов w для линейного дискриминанта определяется максимизацией критерия Фишера:

$$J(w) = \frac{(\mu_1 - \mu_2)^2}{\sigma_1^2 + \sigma_2^2} \tag{4.10}$$

Линейное решение принимается по следующей функции:

$$f(\boldsymbol{x}) = w^T \boldsymbol{x} + b \tag{4.11}$$

, где b – смещение. Класс определяется по правилу:

$$C(\boldsymbol{x}) = \begin{cases} 1, \; f(\boldsymbol{x}) \geq \theta \\ 0, \; otherwise \end{cases} \tag{4.12}$$

,где θ – порог.

При построении корректоров для работы с детектором в условиях несбалансированных классов ключевым требованием является достижение нулевого уровня ошибок принятия чужого (FAR) на выбранных данных, так как фрагментов, на которое делится изображение значительно больше фрагментов, содержащих искомый объект. Для этого порог θ подбирается таким образом, чтобы ни один пример из класса, не являющегося ошибкой, не был ошибочно классифицирован как ошибка для представленного набора данных. Формально, пусть $\mathcal{X}_1$ – множество ошибочных фрагментов (fn для FN корректора и fp для FP корректора). Тогда условие выбора порога записывается как:

$$\theta = \min\{ f(\boldsymbol{x}) \mid \boldsymbol{x} \in \mathcal{X}_1 \} \tag{4.13}$$

Если условие (4.13) не выполняется, то производится дополнительная настройка весов или выбирается более строгий порог, позволяющий обеспечить требуемый уровень $FAR = 0$.

4.2.6 Экспериментальные результаты

Для оценки эффективности предложенного подхода были проведены эксперименты на двух наборах данных (базах):

Первая база содержит изображения железнодорожных цистерн, на которых детектируются области, содержащие цифровые идентификаторы цистерн. База состоит из 1153 изображения вагонов с размеченными идентификаторами (Рис 4.8).

Рисунок 4.8 — Пример разметки базы идентификаторов ж/д цистерн

Вторая база содержит изображения цифр на ж/д цистернах. База состоит из 1067 изображений идентификаторов с размеченными цифрами [0..9] на каждом (Рис 4.9).

Рисунок 4.9 — Пример разметки цифр идентификатора

Каждая база данных случайно разделяется на три части в соотношении 5/4/1 для соответствующих выборок:

- **Обучающая выборка** используется для построения каскадного детектора.
- **Корректирующая выборка** используется для получения обучающих наборов и построения FN и FP-корректоров.
- **Тестовая выборка** используется исключительно для оценки качества работы алгоритма (без корректоров и после применения корректоров).

При этом эксперимент по разделению повторяется 10 раз для уменьшения зависимости от неравномерности распределения тестовых и обучающих данных.

В экспериментах используются стандартные метрики, характеризующие качество работы детектора объектов: точность (precision) и полнота (recall):

$$Pr = \frac{tp}{tp + fp}$$
$$Re = \frac{tp}{tp + fn} \quad (4.14)$$

, где:

tp — число правильно детектированных объектов;

fp — число ложных срабатываний;

fn — число пропущенных объектов.

При определении того, является ли обнаружение истинным или ложным, использовался порог по IoU равный 0.5, а результаты по каждому разделению усреднены.

Обученный детектор на первой части базы сначала применялся на тестовой выборке без корректоров и вычислялись метрики. Затем на основе ошибок (FN и FP), полученных на корректирующей выборке, были обучены соответствующие корректоры. После этого проводилось повторное тестирование на той же тестовой выборке с применением обученных корректоров, чтобы оценить эффективность корректоров. Для детектирования цифр было построено 10 различных детекторов, локализующих соответствующую цифру на изображении. В итоговых результатах представлено усредненное значение метрик всех детекторов цифр (Таблица 4.1).

Таблица 4.1 — Результаты тестирования детекторов

База данных	**Конфигурация**	**Precision**	**Recall**
Цифры	Без корректоров	**0.92**	0.94
Цифры	FN-корректоры (1)	0.91	**0.98**
Идентификаторы	Без корректоров	0.36	**0.98**
Идентификаторы	FP-корректоры (5)	**0.65**	0.94

Для определения устойчивости характеристик качества корректора от величин порогов пересечения фрагментов с разметкой были построены зависимости $Precision(T)/Recall(T)$ для двух исследуемых типов корректоров (Рис. 4.10). Заметим, что при высоких значениях порога корректору не хватает примеров для построения разделяющей поверхности, что связано с выбираемым шагом скользящего окна фрагментов. Фрагменты становятся маловариативны и это ухудшает обобщение корректора на другие изображения.

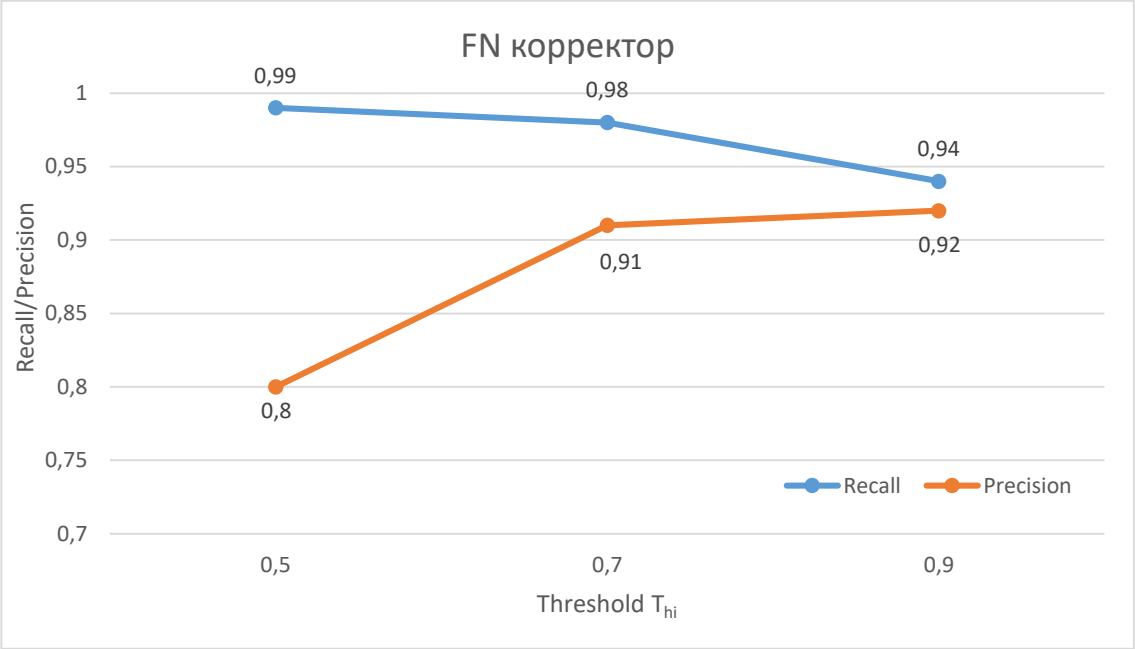


Рисунок 4.10 — Зависимость качества FN корректора от верхнего порога пересечения фрагментов с разметкой на базе ‘Цифры’ при фиксированном T_{lo} (0.15)

При увеличении порога T_{lo} страдает Recall характеристика FP-корректора, в то время как Precision характеристика возрастает (Рис. 4.11).

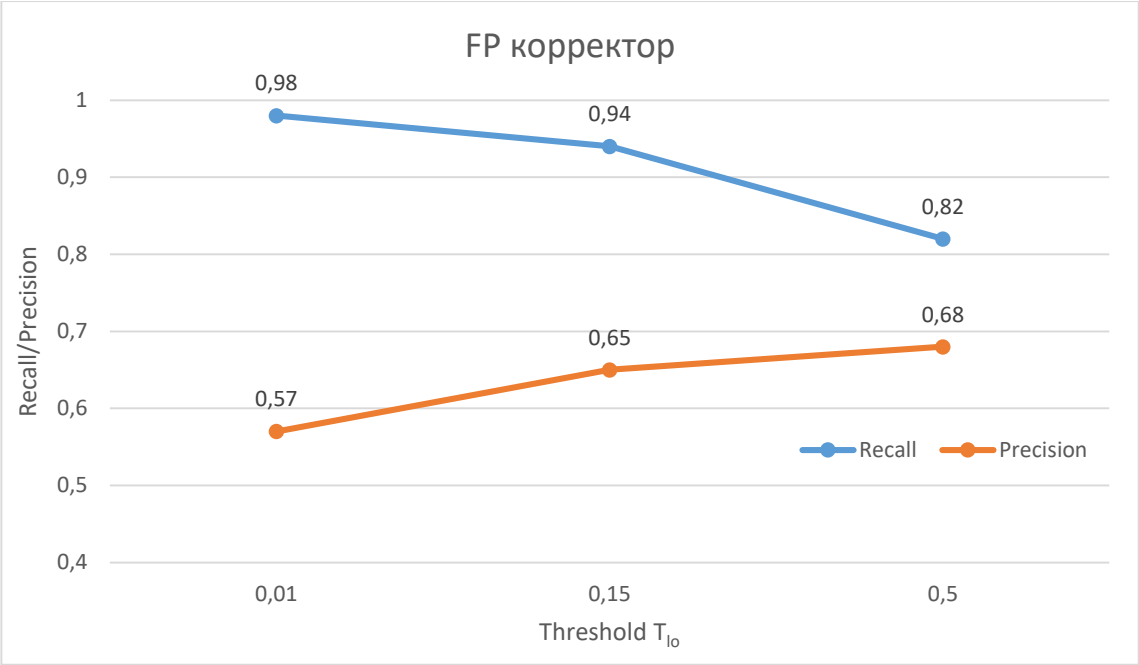


Рисунок 4.11 — Зависимость качества FP корректора от верхнего порога пересечения фрагментов с разметкой на базе ‘Идентификаторы’ при фиксированном T_{hi} (0.75)

Результаты экспериментов (Таблица 4.1, Рис. 4.10, Рис. 4.11) на выбранных базах демонстрируют преимущество применения корректоров для улучшения характеристик детекторов. Использование FN-корректора позволяет существенно повысить значение метрики полноты (Recall), что соответствует снижению количества пропущенных объектов. FP-корректор эффективно подавляет ложные срабатывания, увеличивая точность (Precision).

Полученные результаты показывают, что предложенная методика построения корректоров с использованием разделения классическими методами в пространствах высокой размерности и модифицированного преобразования Цензуса является эффективным инструментом улучшения качества работы детекторов объектов, особенно в ситуациях ограниченного количества обучающих данных [118]. Описанный подход демонстрирует значительный потенциал для улучшения точности и стабильности детекторов, построенных на основе метода [179] и модифицированного преобразования Цунзуса.

4.3 Модель коррекции с помощью динамических сверток

Использование внутренних сигналов каскадных детекторов является не единственным способом реализации модулей коррекции. В случае относительно небольшого количества ошибок, более универсальным решением проблемы коррекции может стать подход сравнения накопленных ошибочных примеров. Эта категория подходов нацелена на изучение набора функций проекции, которые преобразуют запрос и примеры изображений во внутреннее представление и классифицируют их в режиме прямого распространения [67, 162, 168, 179]. Мета-обучение здесь состоит в обучении вспомогательной сети параметризации, которая учится задавать параметры классификации прямого распространения для задачи с небольшим числом примеров. Наша архитектура схожа с методами, которые изучают семантическое соответствие пары изображений и категорий. Большинство этих методов используют фиксированную, заранее определенную метрику схожести или линейный классификатор после объединения изображения и встраивания категории. В отличие от них, мы используем преимущество от более глубокой архитектуры, включающей обучаемую нелинейную метрику в виде параметризованной сверточной сети.

В данном разделе работы предлагается использовать алгоритм динамических свёрток в качестве one-shot классификатора ошибок детектора [164]. Динамическая сверточная нейросеть генерирует свои фильтры (ядра свёртки) адаптивно на основе входных данных. Изначально подобная архитектура была предложена для задач распознавания изображений и видео и позволила улучшить точность и скорость обучения современных свёрточных сетей по сравнению с базовыми моделями с фиксированными ядрами [44]. Мы расширяем применение динамических свёрток на задачу корректировки ошибок: алгоритм в реальном времени получает новый пример ошибки от детектора (например, неверно классифицированное изображение) и сразу формирует на его основе эталонное представление – эмбеддинг, то есть

компактный вектор признаков данного образца. Затем входной сигнал текущего кадра сравнивается с эталонными примерами через их эмбеддинги: сеть динамически генерирует свёрточные фильтры на основе эмбеддинга ошибки и применяет их к текущим данным, выдавая результат сравнения в виде коррекции. Если классификатор распознаёт ситуацию как потенциальную ошибку, то сеть может выдать скорректированное решение вместо исходного вывода детектора. Таким образом реализуется one-shot классификация новых ошибок: корректирующая модель на лету подстраивается под единственный образец ошибки, не требуя длительного переобучения. Важным условием эффективности такого подхода является качество пространства эмбеддингов: представления должны быть близки для похожих (однотипных) ошибок и существенно различаться для разных типов ошибок, обеспечивая надёжное распознавание целевого сбоя среди потоков данных.

4.3.1 Концепция динамических свёрток

Принцип динамической свёртки состоит в том, что параметры свёрточного фильтра не фиксированы после обучения, а *генерируются на основе входного образца*. Иными словами, сеть сама производит ядро свёртки в зависимости от характеристик поступающих данных. В рассматриваемой архитектуре корректоров это достигается с помощью двухветвевой нейросети: одна ветвь (условно назовём её *ветвью* Q) извлекает эмбеддинг из опорного образца (например, изображения, на котором детектор ошибся), а другая ветвь (*ветвь* G) осуществляет свёртки над текущим входным изображением, причем её свёрточные слои используют ядра, сформированные на основе эмбеддинга из ветви Q. Такая схема позволяет «на лету» адаптировать фильтры под конкретный образец ошибки. В результате вместо универсальных признаков, настроенных средним образом для всего набора данных, сеть фокусируется на отличительных чертах именно того класса ситуаций, который представлен недавно полученным примером ошибки. Это повышает точность обнаружения

подобных ситуаций по сравнению с традиционными свёрточными сетями [164].

Ранее идея генерировать фильтры из данных уже демонстрировала улучшение качества моделей: так, замена статических свёрток на динамические в классификаторах изображений ShuffleNet и MobileNet дала прирост производительности [187]. Динамические свёртки успешно применялись не только в задачах классификации изображений, но и, например, в аудиоаналитике — для повышения устойчивости к шумам в распознавании речи и звуков [110]. Эти результаты подтверждают, что динамическая параметризация свёрток позволяет моделям выделять ключевые особенности сигналов и тем самым улучшать точность распознавания.

В контексте классификации ошибок детектора динамические свёртки дают важное преимущество: корректирующая модель способна мгновенно обучиться на единичном новом случае ошибки. Эмбеддинг, вычисленный для образца-ошибки, сразу используется для настройки фильтров, поэтому классификатор ошибок не требует долгого градиентного обучения – фактически обучение происходит за один шаг путем прогонки примера через сеть. Динамические свёртки используют этот эмбеддинг и адаптируют решающий слой (ядро свёртки) под каждый новый класс ошибки. При этом, в отличие от простого линейного классификатора, свёрточная сеть может моделировать более сложные разделяющие поверхности в пространстве признаков, что особенно важно, когда различие между ошибочной и корректной классификацией незначительное [166]. Таким образом, концепция динамических свёрток объединяет идеи мета-обучения и преимуществ высокоразмерных представлений для обеспечения высокой точности распознавания ошибок по одному примеру.

4.3.2 Реализация базовой и динамической модели

Для проверки эффективности подхода рассмотрим две архитектуры свёрточного корректора: базовую (статическую) и динамическую. *Базовый*

сверточный корректор реализует традиционный подход: он принимает на вход одновременно текущее изображение g и опорный пример ошибки q (например, конкатенирует их по каналам) и пропускает их через стандартные сверточные слои Рис 4.12 (a). На выходе базовая модель выдаёт бинарное решение y – является ли входная ситуация ошибочной (похожа на пример q) или нет. В такой архитектуре параметры всех свёрток фиксированы после обучения на наборе данных и не изменяются при поступлении новых ошибок. Это означает, что для добавления нового типа ошибки потребовался бы этап дообучения модели на новых данных, что нежелательно в реальном времени.

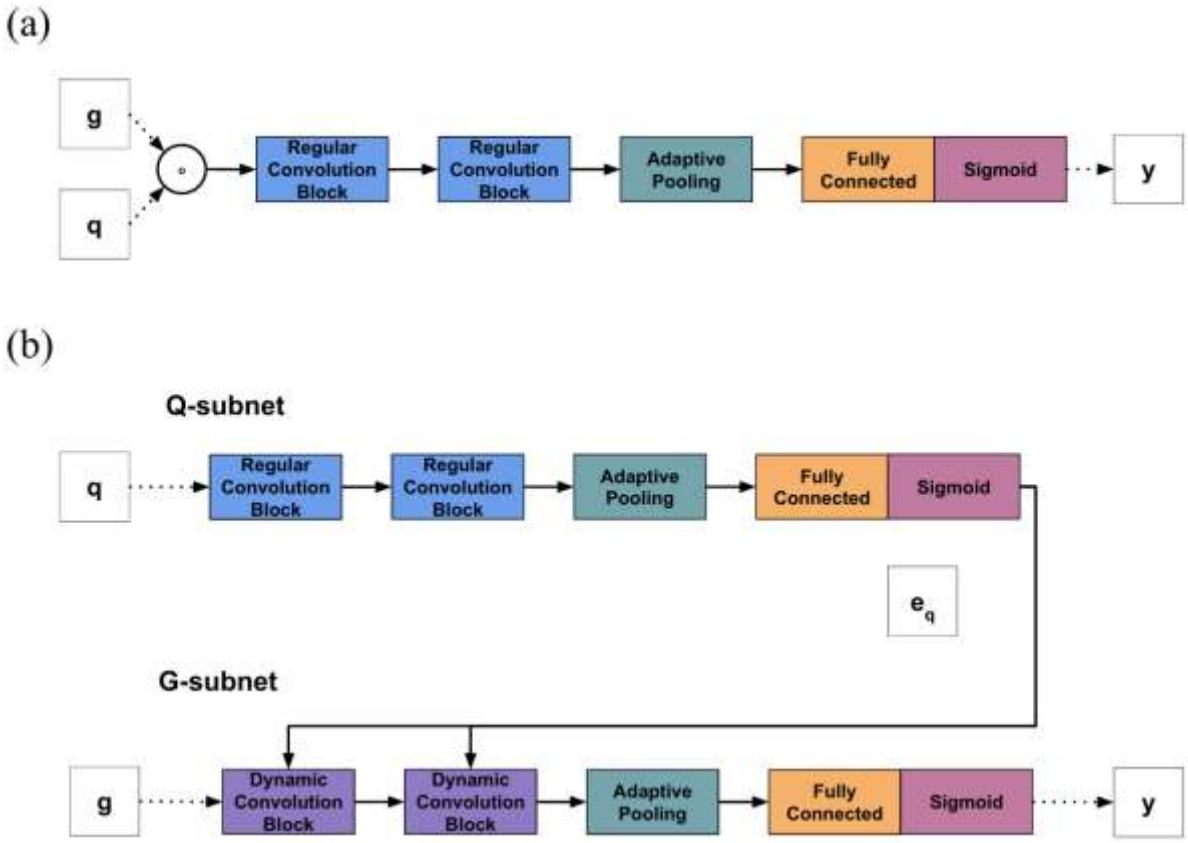


Рисунок 4.12 — a) Структура базовой модели; b) Структура динамической модели

В *динамической модели*, напротив, архитектура разбивается на две части (Рис 4.12 b): первая подсеть Q получает на вход только образец q и вычисляет для него эмбеддинг e_q. Вторая подсеть G обрабатывает текущее изображение g, но её сверточные слои построены особым образом: вместо обычных свёрток используются *динамические сверточные блоки*, которые генерируют свои

ядра на основе эмбеддинга e_q (Рис 4.12 b). Структурно ветвь G повторяет базовую модель за тем исключением, что каждый сверточный блок включает механизм динамической генерации фильтров. Формально, динамическая модель может быть представлена как:

$$y=G(g,e_q,\theta_g)$$
$$e_q=Q(q,\theta_q) \quad (4.15)$$

, где θ_g и θ_q – обучаемые параметры сетей.

Реализация динамического сверточного блока содержит свертку 3×3 (Dynamic Depth-wise Conv2d), параметры которой вычисляются в параллельной ветви блока на лету (Рис 4.13). Эта параллельная ветвь представляет собой небольшой слой универсального аппроксиматора [99], который, принимая на вход эмбеддинг e_q, определяет значения весов фильтра. Как правило, он состоит из одного или двух полносвязных слоёв с нелинейностью (ReLU) между ними [94].

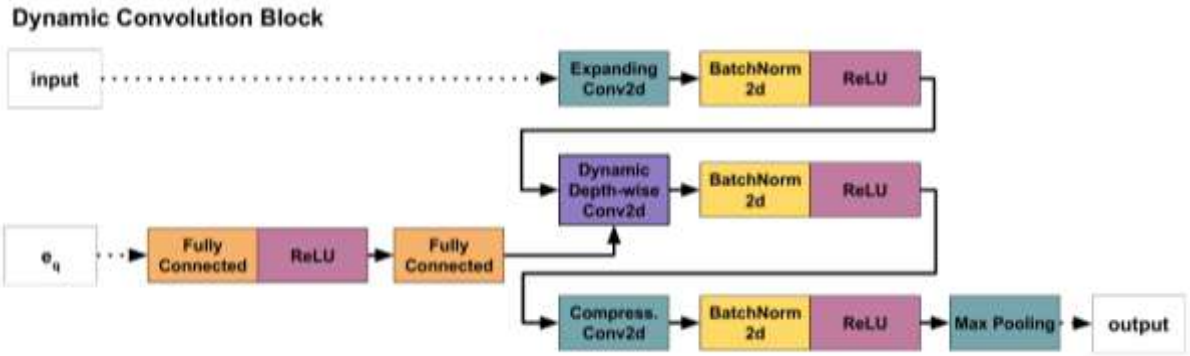


Рисунок 4.13 — a) Структура блока регулярной свертки. b) Структура блока динамической свертки

Размер эмбеддинга e_q и архитектура генерирующей подсети являются гиперпараметрами, определяющими количество генерируемых весов и, следовательно, сложность модели. В «лёгких» вариантах модели можно использовать меньший эмбеддинг и упрощённый генератор ядра (например, без скрытого слоя), чтобы сократить число параметров. «Тяжёлые» варианты,

напротив, задействуют более крупный эмбеддинг и дополнительный слой в генераторе для повышения выразительности модели.

Обучение динамической модели осуществляется классическим методом, используя алгоритм оптимизации Адам [111], на парах случайно выбранных изображений (q, g). После обучения сеть способна генерировать корректоры для новых ошибок без дополнительной оптимизации весов – достаточно подать новый пример q в ветвь Q. Для сравнения, базовая модель в аналогичной ситуации потребовала бы дообучения на новом примере или предварительного включения этого типа ошибки в тренировочный набор.

4.3.3 Эксперименты и результаты

Для оценки эффективности описанной архитектуры были проведены эксперименты, сравнивающие динамический корректор с базовым. Модель протестирована на задаче сопоставления изображений (matching) на примере датасета MNIST. Каждое испытание представляло собой определение, принадлежит ли пара изображений (образец q и текущий g) одному классу (в нашем контексте это аналогично проверке, повторяется ли ошибка q на новом входе g). Качество классификации оценивалось с помощью метрики F_1, вычисляемой как:

$$F_1 = \frac{2*TP}{2*TP+FP+FN} \tag{4.16}$$

, где TP – количество верно обнаруженных случаев ошибки, FP – ложные тревоги, а FN – пропущенные ошибки. Оба варианта модели (базовый и динамический) настраивались так, чтобы иметь близкое число обучаемых параметров для корректного сравнения.

Результаты показали определенное преимущество динамических свёрток перед базовой версией. Динамическая модель обучалась заметно быстрее и достигала более высокого качества распознавания при сопоставимой сложности сети (Рис 4.14).

Модель	Парам	F1-score
“Тяжелая” дин.	19946	0.96
“Тяжелая” трад.	22626	0.93
“Легкая” дин.	9874	0.94
“Легкая” трад.	9376	0.91

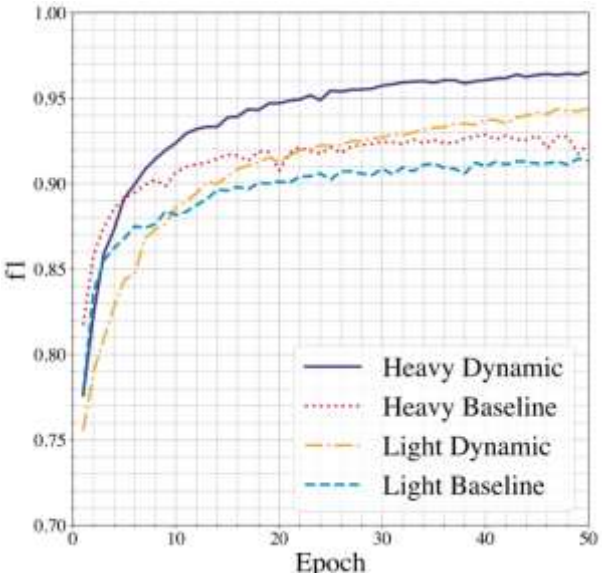


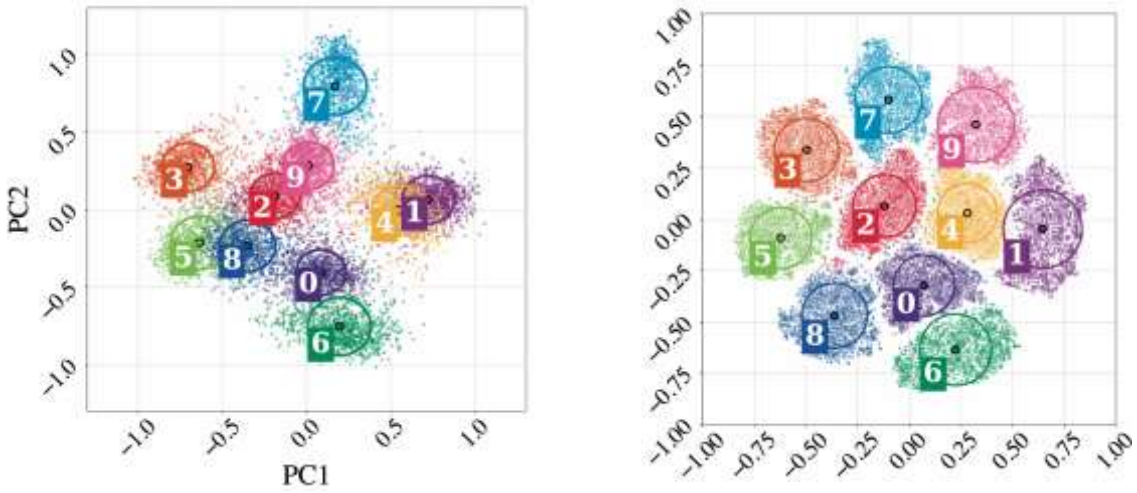
Рисунок 4.14 —Сравнение моделей с использованием динамической свертки и традиционного варианта (дин. и трад. соответсвенно) на валидационной части базы MNIST

Например, «тяжёлая» версия динамического корректора (≈20 тыс. параметров) на валидации достигла среднего $F_1 \approx 0.96$, тогда как базовая свёрточная модель с таким же порядком параметров — лишь около 0.93. Аналогично, в облегчённой конфигурации (≈10 тыс. параметров) динамический подход дал $F_1 \approx 0.94$ против ~0.91 у базового.

Помимо более высокого итогового качества, динамическая сеть демонстрирует более лучшую сходимость: на графиках обучения видно, что её функция потерь убывает быстрее, а F_1 на проверочном наборе растёт опережающими темпами по сравнению с базовой моделью (Рис 4.14). Это указывает на то, что генерация частей фильтров из эмбеддинга облегчает модели поиск разделяющей поверхности, фактически предоставляя ей удачную инициализацию слоёв для конкретной задачи сопоставления образцов.

Отметим, что эмбеддинги, обученные в динамической модели, действительно удовлетворяют требуемым свойствам кластеризации по классам. В работе [164] показано, что в пространстве эмбеддингов изображения одного класса образуют компактные кластеры, хорошо

отделимые от кластеров других классов (Рис 4.15). На двумерных проекциях (PCA и t-SNE) точки, соответствующие изображениям разных цифр MNIST, сгруппированы по цветам, и облака разных цифр практически не перекрываются (кругом обозначено среднее отклонение от центра кластера).



a) b)

Рисунок 4.15 — 2D-проекции пространства эмбеддингов e_q с использованием (a) Первых компонент преобразования PCA и (b) двумерного преобразования tSNE. Каждая точка соответствует одному изображению из набора данных, а ее цвет соответствует классу изображений, указанному на рисунке

Эта разделимость подтверждает применимость подхода редукции размерности к задаче коррекции: даже с одним прототипом класса сеть может уверенно отличать его от других объектов. Более того, динамический подход позволяет использовать заранее рассчитанные усреднённые эмбеддинги классов: отметим, что можно вычислить «прототипический» эмбеддинг класса (например, усреднив эмбеддинги нескольких известных изображений этого класса) и использовать его вместо ветви Q для ускорения работы. Это эквивалентно тому, чтобы для заранее известных типов ошибок хранить усреднённый эмбеддинг и тем самым мгновенно переключать корректирующую сеть на обнаружение этой ошибки без прогонки отдельного примера через ветвь Q. Такой приём экономит вычислительные ресурсы и

упрощает масштабирование системы при росте числа известных видов ошибок.

Наш эксперимент с динамическими свёртками на более простой задаче подтверждает, что даже единичный корректирующий модуль, обученный на одном примере, может существенно повысить точность системы. При переходе к реальным сложным данным ожидается, что комбинация динамических свёрток с подходами позволит масштабировать решение на множество типов ошибок.

4.4 Выводы

В работе реализованы два наиболее разработанных подхода к реализации корректирующих алгоритмов: используя эффект “благословения” в высокоразмерных пространствах и подход на основе снижения размерности исходного пространства. Оба эти подхода позволяют получать качественные решения и улучшать характеристики исходных детекторов. Архитектура корректоров обладает рядом преимуществ, которые делают её привлекательной для различных применений. Во-первых, она обеспечивает освоение нового класса ошибок без перестройки основной модели – это особенно важно в автономных системах, где оперативность коррекции критична. Во-вторых, при использовании корректоров на базе динамических сверток за счёт адаптивных фильтров достигается высокая точность распознавания аномалий: сеть фокусируется на специфических паттернах ошибки, игнорируя посторонние вариации данных, что повышает чувствительность корректоров. В-третьих, предлагаемые варианты корректоров относительно экономичны – после первоначального обучения корректоры добавляют не так много вычислительных затрат, так как спроецировать измерения в определенный кластер или сгенерировать эмбеддинг для нового примера и применить несколько специализированных свёрток обычно быстрее, чем переобучать весь базовый алгоритм.

Перспективные области применения такой архитектуры включают любые ситуации, где основная модель должна адаптироваться к новым или редким событиям. Например, в системах компьютерного зрения корректоры могут обучаться обнаруживать новые виды объектов или дефектов на производстве сразу после первого случая брака. В кибербезопасности аналогичный подход поможет по одному экземпляру выявлять новые атаки или аномальное поведение, генерируя фильтры для характерного сигнала угрозы. В обработке языковых данных корректоры могут по одному примеру нового сленгового или ошибочного слова корректировать работу языковой модели.

Отдельно стоит упомянуть возможность интеграции нескольких корректоров в мульти-корректорную систему для одновременной работы с множеством различных типов ошибок. В таком случае входящие данные сначала анализируются диспетчером, который определяет, к какому кластеру (типу) потенциальной ошибки они ближе, и перенаправляет их в соответствующий элементарный корректор. Каждый корректирующий модуль настроен под свой класс ошибок и работает независимо. Этот подход масштабируется с ростом числа ошибок: добавление нового типа сводится к обучению ещё одного модуля на единственном примере и не влечёт изменения остальных.

Возможные улучшения архитектуры связаны главным образом с поиском высокоразмерного пространства признаков, пригодного для применения стохастических теорем и классических методов построения разделяющей гиперплоскости. Ещё одно направление развития – применение корректоров в сочетании с методами выявления новизны: например, сначала определять, что текущий вход явно выбивается из распределения обучающих данных детектора (новый класс или аномалия), и только затем запускать механизм корректировки. Это могло бы снизить число излишних срабатываний корректоров.

ВЫВОДЫ

1. Показано, что расширение признакового пространства с помощью серии однородных однокомпонентных нейроноподобных сред с различающимися параметрами функции связи типа латеральное торможение, а также использование временной динамики позволяют улучшить качество базового распознающего алгоритма в задаче классификации лиц.

2. Разработан алгоритм фильтрации для бинаризации изображений отпечатков пальцев на основе неоднородной однокомпонентной нейроноподобной среды и на собственной базе изображений показано, что качество его изображений выше и его выбор более предпочтителен, чем классический алгоритм нормализации яркости.

3. На примере задачи классификации отпечатков пальцев показано, что адаптивный выбор алгоритма бинаризации на основе анализа динамики невязки изображения отпечатка в процессе “кодирования-восстановления” позволяет улучшать качество распознавания конечного алгоритма.

4. Предложена концепция тернарного классификатора, называемого экспертом, который использует дополнительный ответ для обозначения границы области своей компетенции и введены параметры для оценки качества такого классификатора.

5. Предложен, теоретически и экспериментально обоснован алгоритм M-Adaboost для построения ансамблей тернарных классификаторов методом накачки, который позволяет уменьшить общую ошибку на ~40% на примере детектирования фасадов домов на изображении

6. Разработан алгоритм OnSVM, основанный на принципе максимизации зазора между ответами ансамбля классификаторов, для построения и дообучения экспертного комитета. Эффективность алгоритма продемонстрирована в задачах классификации атрибутов лица человека и на них показано преимущество решения OnSVM перед алгоритмами накачки на 2-6%.

7. Реализованы два подхода к коррекции алгоритмов распознавания - с использованием эффекта “благословения” размерности и с помощью редукции исходного пространства - на примере детектора объектов на изображении и сравнении двух изображений.

8. Разработан механизм “внимания” на основе динамических сверток для задач технического зрения, позволяющий выделять признаки на изображения в зависимости от внешних управляющих сигналов, и показана его эффективность в задаче сравнения двух изображений.

ЗАКЛЮЧЕНИЕ

В диссертации представлено комплексное исследование нейроморфных алгоритмов принятия решений в задачах распознавания объектов на изображении, охватывающее четыре взаимосвязанных уровня обработки визуальной информации: динамическую фильтрацию, адаптивный выбор алгоритмов, построение коллективных моделей и разработку корректоров и механизмов внимания.

В первой главе исследована динамика преобразования изображений в нейроноподобных средах. Сформулированы и проанализированы уравнения возбуждающе-тормозной динамики, рассмотрены однородные и неоднородные функции пространственной связи, включая модифицированные латеральные фильтры, позволяющие выделять объекты различных масштабов и ориентаций. Разработана процедура формирования признаков на основе банка нейроноподобных сред; экспериментально показано, что временная динамика улучшает разделимость классов в задаче классификации лиц.

Во второй главе предложена адаптивная модель принятия решений, основанная на цикле «кодирование – восстановление», и введено формализованное понятие динамики невязки как индикатора качества изображения. Разработана методика сравнения алгоритмов бинаризации отпечатков пальцев, включающая активный (нейроноподобный) и пассивный фильтры. На собственной базе изображений показано, что анализ изменения невязки на первых итерациях позволяет автоматически выбирать алгоритм, наиболее подходящий к локальной структуре данных, что ведёт к повышению качества классификации отпечатков пальцев.

В третьей главе выполнен анализ комитетных методов классификации. Формализовано понятие тернарного классификатора-эксперта как модели с областью компетентности и описаны параметры его качества. Разработан алгоритм M-AdaBoost для построения ансамблей экспертов; теоретически обоснована его работа в условиях асимметричных ошибок. Экспериментально показано, что M-AdaBoost обеспечивает уменьшение ошибки ансамбля по

сравнению с классическими методами бустинга, включая задачу детектирования фасадов зданий.

Разработан алгоритм OnSVM, реализующий принцип максимизации зазора между ответами членов экспертного комитета и обеспечивающий вставку и удаление опорных векторов в потоковом режиме. Показано, что предложенный метод повышает точность классификации атрибутов лица на 2–6% по сравнению с бустинговыми ансамблями.

В четвертой главе исследованы два подхода к построению корректоров: на основе эффекта стохастического разделения в высоких размерностях и на основе редукции пространства признаков. Дополнительно предложен механизм внимания на базе динамических свёрток, позволяющий выбирать релевантные признаки в задачах сравнения изображений.

Перспективным направлением развития является создание альтернативных общепринятым архитектур анализа данных, в которых ядром служит не статическая конвейерная обработка признаков, а *опережающее предсказание входного сигнала* и его согласование с внутренними моделями. В таких системах распределённая нейроноподобная среда должна формировать *многомерное внутреннее представление образа объекта* (информационный образ), которое не просто реагирует на вход, а задаёт ожидаемую структуру сигнала и управляет динамикой его обработки.

Особый интерес представляет развитие архитектур с *постоянными обратными связями*, преднастраивающими входные фильтры на основе текущего внутреннего состояния системы и накопленных представлений. В этом случае блоки «внимания», корректоры и ансамбли экспертов могут рассматриваться как элементы единого предсказательно-корректирующего контура: внутренний образ объекта задаёт ожидания, через обратные связи модулирует параметры фильтрации и распознавания, а невязка между предсказанным и реальным сигналом служит мотивационным сигналом для адаптации.

Дальнейшие исследования могут быть направлены на интеграцию разработанных в диссертации механизмов (нейроноподобной динамики, адаптивного выбора алгоритмов, корректоров и экспертных комитетов) в такие предсказательные архитектуры, а также на их применение в реальных биометрических системах и автономных технических системах зрения, работающих в условиях неполных, зашумлённых и изменчивых данных.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Беллюстин Н. С., Калафати Ю. Д., Ковальчук А. В., Тельных А. А., Шемагина О. В., Яхно В. Г. Нейроноподобный детектор лица. Технические особенности реализации и обучения. // Нейроинформатика – 2008. Сборник научных трудов. Ч. 2. М.:МИФИ. 2008. С. 123–132.

2. Беллюстин Н. С., Ковальчук А. В. Разработка адаптивного базового классификатора для когнитивной системы // Труды 3-й Всероссийской конференции «Нелинейная динамика в когнитивных исследованиях – 2013». 2013. С. 13–15.

3. Беллюстин Н. С., Ковальчук А. В., Соколов М. Е., Тельных А. А., Яхно В. Г. Система детектирования фасадов зданий на изображениях // Материалы XV международной научно-технической конференции «Информационные системы и технологии (ИСТ-2009)». 2009. С. 219–220.

4. Бонгард М. М. Проблема узнавания / Бонгард М. М., Москва: Наука, 1967. 320 с.

5. Вапник В. Н. Алгоритмы и программы восстановления зависимостей / В. Н. Вапник, Москва: ФИЗМАТЛИТ, 1984.

6. Вапник В. Н., Лернер А. Я. Узнавание образов при помощи обобщенных портретов // Автомат. и телемех. 1963. № 6 (24). С. 774–780.

7. Вапник В. Н., Червоненкис А. Я. Об одном классе персептронов // Автомат. и телемех. 1964. № 1 (25). С. 112–120.

8. Городецкий В. И., Серебряков С. В. Методы и алгоритмы коллективного распознавания: обзор // Труды СПИИРАН, СПб, Наука. 2006. № 3 (1). С. 139–171.

9. Ковальчук А. В. Об автоматизации построения детекторов объектов на примере детектора изображения лиц // 14 научная школа «Нелинейные волны - 2008». Тезисы докладов. 2008. С. 79.

10. Ковальчук А. В. Нейроноподобный алгоритм с функциями обучения, хранения условных вероятностей и идентификации на основе конкуренции

последовательностей // Нелинейная динамика в когнитивных исследованиях. 2009. С. 70.

11. Ковальчук А. В., Беллюстин Н. С. Модель узнавания объекта по изображению на основе дерева принятия решений // 5-я Международная конференция по Когнитивной науке. 2012. (1). С. 415–417.

12. Ковальчук А. В., Беллюстин Н. С. Последовательный алгоритм обучения ядровых классификаторов на опорных векторах // Нейроинформатика – 2013. Сборник научных трудов Ч.3. М.:МИФИ. 2013. С. 240–248.

13. Ковальчук А. В., Беллюстин Н. С. Алгоритм классификации потоковых сигналов на основе последовательной машины опорных векторов // ПНД. Известия «ВУЗ». 2015. № 5 (23). С. 62–79.

14. Ковальчук А. В., Иванов А. Е., Яхно В. Г. Оценка качества работы алгоритмов бинаризации по динамике процессов кодирования - восстановления // Нейроинформатика – 2005. Сборник научных трудов Ч.1. М.:МИФИ. 2005. С. 274–282.

15. Ковальчук А. В., Соколов М. Е. Экспертные комитеты: принципы построения и применения // Нейроинформатика – 2009. Сборник научных трудов Ч.1. М.:МИФИ. 2009. С. 117–123.

16. Ковальчук А. В., Соколов М. Е., Тельных А. А. Сопоставление нейроноподобных и классических методов в задаче уточнения координат глаз // Нейроинформатика – 2008. Сборник научных трудов. Ч. 2. М.:МИФИ. 2008. С. 52.

17. Ковальчук А. В., Соколов М. Е., Тельных А. А. Нейроноподобная фильтрация в системе распознавания, использующей двухклассовый подход // Нейроинформатика - 2009. Сборник научных трудов Ч.2. М.:МИФИ. 2009. С. 262–270.

18. Ковальчук А. В., Тельных А. А., Шемагина О. В., Яхно В. Г. Программное обеспечение для изучения динамики преобразования сенсорных

сигналов в нейронно-глиальных сетях мозга // Нейроинформатика – 2017. Сборник научных трудов Ч.2. М.:МИФИ. 2017. С. 130–139.

19. Ковальчук А.В., Беллюстин Н. С., Тельных А. А., Яхно В. Г. О методе промежуточного контроля в сложной системе обнаружения и распознавания лиц // Математические методы распознавания образов, ММРО-13. М.:МАКС Пресс. 2007. С. 478–481.

20. Кудряшов А.В., Яхно В.Г. Распространение областей повышенной импульсной активности в нейронной сети. // В сб. Динамика биологических систем. 1978. (Вып.2). С. 45–59.

21. Мазуров В. Д. Метод комитетов в задачах оптимизации и классификации / В. Д. Мазуров, Москва: Наука, 1990. 248 с.

22. Мастеров А.В., Рабинович М.И., Толков В.Н., Яхно В.Г. Исследование режимов взаимодействия автоволн и автоструктур в нейроподобных средах. // В сб. Коллективная динамика возбуждений и структурообразование в биологических тканях. 1988. С. 89–104.

23. Парин С. Б., Ковальчук А. В., Чернова М. А., Полевая С. А. Маловостребованный ресурс нейроноподобного моделирования: адаптивное управление сигналами о рассогласовании в биологических системах // Нейроинформатика – 2012. Сборник научных трудов Ч. 2. М.:МИФИ. 2012. С. 187–195.

24. Растригин Л. А., Эренштейн Р. Х. Метод коллективного распознавания / Л. А. Растригин, Р. Х. Эренштейн, Москва: Библиотека по автоматике. Вып. 615. Энергия, 1981. 78 с.

25. Сбитнев В. И., Драбкин Г. М. Перенос спайков в статистических нейронных ансамблях. I. Концепция фазовых переходов // Биофизика. 1975. (Т.20). С. 669–702.

26. Тельных А. А., Яхно В. Г. Нейроноподобные модели второго и третьего уровней - адаптивные распознающие системы // Материалы XII Международной конференции по нейрокибернетики, Издательство Северокавказского научного центра высшей школы. 1999. С. 164–168.

27. Чайкин А. В., Ковальчук А. В. Использование нейроноподобных алгоритмов для обработки изображений в однородных сетях и их кодирования на примере биометрической системы по дактоотпечаткам // Нейроинформатика – 2004. Сборник научных трудов. Ч. 2. М.:МИФИ. 2004. С. 231–238.

28. Яхно В. Г. Процессы самоорганизации в распределенных нейроноподобных системах. Примеры возможных применений. // Лекции по Нейроинформатике - 2001. М.:МИФИ. 2001. С. 103–141.

29. Яхно В. Г., Нуйдель И. В., Иванов А. Е., Беллюстин Н. С., Будников Д. Н., Тельных А. А., Еремин Е. В., Коган А. Н., Костин М. А., Перминов А. О., Раджабова Ю. Х., Сорокин М. А., Тихомиров Д. А., Шилин С. Г., Шилин А. В. Исследование динамических режимов нейроноподобных систем. Примеры приложений // Информационные технологии и вычислительные системы. 2004. № 1. С. 126–148.

30. Яхно В. Г., Нуйдель И. В., Тельных А. А., Бондаренко Б. Н., Сборщиков И. Ф., Хилько А. И. Метод адаптивного распознавания информационных образов и система для его осуществления // 2000.

31. Яхно В. Г., Тельных А. А., Нуйдель И. В., Полевая С. А., Парин С. Б., Беллюстин Н. С., Еремин Е. В., Разумов В. А., Иванов А. Е., Чайкин А. В., Шемагина О. В., Спицын И. Г., Краева Т. А. Программные модели обработки зрительных сигналов // Альманах клинической медицины. 2006. (Том XII). С. 69.

32. Яхно В.Г., Беллюстин Н.С., Красильникова И.Г., Кузнецов С.О., Нуйдель И.В., Панфилов А. И., Перминов А.О., Шадрин А.В., Шевырев А.А. Исследовательская система принятия решений по фрагментам сложного изображения, использующая нейроноподобные алгоритмы // Изв. Вузов. Радиофизика. 1994. (Т. 37. N8). С. 961–986.

33. Ahonen T., Hadid A., Pietikäinen M. Face Recognition with Local Binary Patterns // Lecture Notes in Computer Science (including subseries Lecture Notes in

Artificial Intelligence and Lecture Notes in Bioinformatics). 2004. (3021). С. 469–481.

34. Ali K. M., Pazzani M. J. Error reduction through learning multiple descriptions // Machine Learning 1996 24:3. 1996. № 3 (24). С. 173–202.

35. Alpaydin E., Alimoglu Fevzi. Pen-Based Recognition of Handwritten Digits // 1996.

36. Amari S. Dynamics of pattern formation in lateral-inhibition type neural fields // Biological Cybernetics. 1977. № 2 (27). С. 77–87.

37. Amodei D., Olah C., Brain G., Steinhardt J., Christiano P., Schulman J., Dan O., Google Brain M. Concrete Problems in AI Safety 2016.

38. Anderson J., Belkin M., Goyal N., Rademacher L., Voss J. The more, the merrier: the blessing of dimensionality for learning large Gaussian mixtures // JMLR: Workshop and Conference Proceedings. Conference on Learning Theory. 2014. (35). С. 1–30.

39. Bay S. D., Pazzani M. J. Characterizing Model Errors and Differences 2002.

40. Belhumeur P. N., Hespanha J. P., Kriegman D. J. Eigenfaces vs.Fisherfaces: Recognition using class specific linear projection // IEEE Trans. Pattern Anal. Machine Intell. 1997. (19). С. 711–720.

41. Bellustin N., Kalafati Y., Kovalchuk A., Telnykh A., Shemagina O., Yakhno V., Vaish A., Sharma P. Instant Human Face Attributes Recognition System // International Journal of Advanced Computer Science and Applications, Special Issue on Artificial Intelligence. 2011. № 3 (1). С. 112–120.

42. Bengio Y., Lee D.-H., Bornschein J., Mesnard T., Lin Z. Towards Biologically Plausible Deep Learning 2015.

43. BoJin, Hua P., Ming L. Fingerprint singular point detection algorithm by Poincaré index // WSEAS TRANSACTIONS on SYSTEMS. 2008.

44. Brabandere B. De, Jia X., Tuytelaars T., Gool L. Van Dynamic Filter Networks // Advances in Neural Information Processing Systems. 2016. С. 667–675.

45. Breiman L. Bagging predictors // Machine Learning. 1996. № 2 (24). C. 123–140.

46. Breiman L. Random forests // Machine Learning. 2001. № 1 (45). C. 5–32.

47. Breiman L., Stone C. J. Waveform Database Generator (Version 1) // 1984.

48. Brezhneva O. A., Tret'Yakov A. A., Wright S. E. A simple and elementary proof of the Karush-Kuhn-Tucker theorem for inequality-constrained optimization // Optimization Letters. 2009. № 1 (3). C. 7–10.

49. Carpenter G., vision S. G.-C., graphics undefined, image and, 1987 undefined A massively parallel architecture for a self-organizing neural pattern recognition machine // Elsevier.

50. Cauwenberghs G., Poggio T. Incremental and decremental support vector machine learning 2000.C. 409–415.

51. Chan P. K., Stolfo S. J. A Comparative Evaluation of Voting and Meta-learning on Partitioned Data // Proceedings of the 12th International Conference on Machine Learning, ICML 1995. 1995. C. 90–98.

52. Chan P. K., Stolfo S. J. On the Accuracy of Meta-learning for Scalable Data Mining // Journal of Intelligent Information Systems. 1997. № 1 (8). C. 5–28.

53. Chang C.-C., Lin C.-J. LIBSVM : a library for support vector machines. 2001.

54. Chua L. O., Yang L. Cellular Neural Networks: Theory // IEEE Transactions on Circuits and Systems. 1988. № 10 (35). C. 1257–1272.

55. Collobert R., Williamson R. C. SVMTorch: support vector machines for large-scale regression problems // The Journal of Machine Learning Research. 2001. (1). C. 143–160.

56. Condorcet J. Essai sur l'application de l'analyse à la probabilité des décisions rendues à la pluralité des voix ([Reprod.]) / par M. le marquis de Condorcet,... 1785.

57. Cortes C., Vapnik V., Saitta L. Support-vector networks // Machine Learning 1995 20:3. 1995. № 3 (20). С. 273–297.

58. Cristianini N., Shawe-Taylor J. An Introduction to Support Vector Machines and Other Kernel-based Learning Methods / N. Cristianini, J. Shawe-Taylor, Cambridge University Press, 2000. 189 с.

59. Cucker F., Smale S. On the Mathematical Foundations of Learning // Bulletin of the American Mathematical Society. 2001. № 1 (39).

60. Dalal N., Triggs B. Histograms of oriented gradients for human detection // Proceedings - 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, CVPR 2005. 2005. (I). С. 886–893.

61. Daugman J. G. Uncertainty relation for resolution in space, spatial frequency, and orientation optimized by two-dimensional visual cortical filters // Journal of the Optical Society of America. 1985. № 7 (2). С. 1160–1169.

62. Daugman J. G. Complete Discrete 2-D Gabor Transforms by Neural Networks for Image Analysis and Compression // IEEE Transactions on Acoustic, Speech and Signal Processing. 1988. № 36 (V). С. 1169–1179.

63. Dayan P., Hinton G. E., Neal R. M., Zemel R. S. The Helmholtz machine. // Neural computation. 1995. № 5 (7). С. 889–904.

64. Deng L. The MNIST database of handwritten digit images for machine learning research // IEEE Signal Processing Magazine. 2012. № 6 (29). С. 141–142.

65. Eckhorn R., Reitboeck H. J., Arndt M., Dicke P. Feature Linking via Synchronization among Distributed Assemblies: Simulations of Results from Cat Visual Cortex // Neural Computation. 1990. № 3 (2). С. 293–307.

66. Elitalib E. H., Bahar A. A. A. Neuromorphic Computing Architectures for Real-time Image Processing and Pattern Recognition // Algorithm Asynchronous. 2023. № 1 (1). С. 24–32.

67. Fei-Fei L., Fergus R., Perona P. One-shot learning of object categories // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2006. № 4 (28). С. 594–611.

68. Fix E., Hodges J. L. Discriminatory Analysis. Nonparametric Discrimination: Consistency Properties // International Statistical Review / Revue Internationale de Statistique. 1989. № 3 (57). С. 238.

69. Freud Y., Schapire R. E. Experiments with a new boosting algorithm // Machine Learning Proceedings of the Thirteen International Conference. 1996. С. 148–156.

70. Freund Y., Schapire R. E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting // Journal of Computer and System Sciences. 1997. № 1 (55). С. 119–139.

71. Freund Y., Shapire R. E. A short introduction to boosting // Journal of Japanese Society or Artificial Intelligence. 1999. С. 771–780.

72. Fröba B., Ernst A. Face detection with the modified census transform // Proceedings - Sixth IEEE International Conference on Automatic Face and Gesture Recognition. 2004. С. 91–96.

73. Fukushima K. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position // Biological Cybernetics. 1980. № 4 (36). С. 193–202.

74. Fukushima K. Neocognitron: A hierarchical neural network capable of visual pattern recognition // Neural Networks. 1988. № 2 (1). С. 119–130.

75. Gallego G., Delbruck T., Orchard G., Bartolozzi C., Taba B., Censi A., Leutenegger S., Davison A., Conradt J., Daniilidis K., Scaramuzza D. Event-based Vision: A Survey // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2020. № 1 (44). С. 154–180.

76. Gama J., Brazdil P. Cascade Generalization // Machine Learning. 2000. № 3 (41). С. 315–343.

77. Geirhos R., Jacobsen J.-H., Michaelis C., Zemel R., Brendel W., Bethge M., Wichmann F. A. Shortcut Learning in Deep Neural Networks // Nature Machine Intelligence. 2023. № 11 (2). С. 665–673.

78. Geng X., Zhou Z. H., Smith-Miles K. Automatic age estimation based on facial aging patterns // IEEE transactions on pattern analysis and machine intelligence. 2007. № 12 (29). С. 2234–2240.

79. Goodfellow I., Bengio Y., Courville A., Bengio Y. Deep learning / I. Goodfellow, Y. Bengio, A. Courville, Y. Bengio, 2016.

80. Gorban A. N., Grechuk B., Mirkes E. M., Stasenko S. V., Tyukin I. Y. High-Dimensional Separability for One- and Few-Shot Learning // Entropy 2021, Vol. 23, Page 1090. 2021. № 8 (23). С. 1090.

81. Gorban A. N., Makarov V. A., Tyukin I. Y. The unreasonable effectiveness of small neural ensembles in high-dimensional brain // Physics of Life Reviews. 2019. (29). С. 55–88.

82. Gorban A. N., Tyukin I. Y. Stochastic Separation Theorems // Neural Networks. 2017. (94). С. 255–259.

83. Gorban A. N., Tyukin I. Y. Blessing of dimensionality: Mathematical foundations of the statistical physics of data // Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences. 2018. № 2118 (376).

84. Gorban A. N., Tyukin I. Y., Romanenko I. The Blessing of Dimensionality: Separation Theorems in the Thermodynamic Limit // IFAC-PapersOnLine. 2016. № 24 (49). С. 64–69.

85. Grechuk B., Gorban A. N., Tyukin I. Y. General stochastic separation theorems with optimal bounds // Neural Networks. 2021. (138). С. 33–56.

86. Grossberg S. The complementary brain: unifying brain dynamics and modularity // Trends in Cognitive Sciences. 2000. № 6 (4). С. 233–246.

87. Grossberg S. Adaptive Resonance Theory: How a brain learns to consciously attend, learn, and recognize a changing world // Neural Networks. 2013. (37). С. 1–47.

88. Guo G., Dyer C. R., Fu Y., Huang T. S. Is gender recognition affected by age? // 2009 IEEE 12th International Conference on Computer Vision Workshops, ICCV Workshops. 2009. С. 2032–2039.

89. Guo G., Guowang Mu, Fu Y., Huang T. S. Human age estimation using bio-inspired features // IEEE Conference on Computer Vision and Pattern Recognition. 2009. С. 112–119.

90. Guyon I., Weston J., Barnhill S., Vapnik V. Gene selection for cancer classification using support vector machines // Machine Learning. 2002. № 1–3 (46). С. 389–422.

91. Hadsell R., Rao D., Rusu A. A., Pascanu R. Embracing Change: Continual Learning in Deep Neural Networks // Trends in Cognitive Sciences. 2020. № 12 (24). С. 1028–1040.

92. Hastie T., Friedman J., Tibshirani R. The Elements of Statistical Learning / T. Hastie, J. Friedman, R. Tibshirani, New York, NY: Springer New York, 2001.

93. Hastie T., Tibshirani R., Friedman J., Friedman J. The elements of statistical learning: data mining, inference, and prediction / T. Hastie, R. Tibshirani, J. Friedman, J. Friedman, 2009.

94. Heinecke A., Ho J., Hwang W. L. Refinement and Universal Approximation via Sparsely Connected ReLU Convolution Nets // IEEE Signal Processing Letters. 2020. (27). С. 1175–1179.

95. Hendrycks D., Dietterich T. Benchmarking Neural Network Robustness to Common Corruptions and Perturbations // 7th International Conference on Learning Representations, ICLR 2019. 2019.

96. Hinton G. E., Salakhutdinov R. R. Reducing the dimensionality of data with neural networks // Science. 2006. № 5786 (313). С. 504–507.

97. Hodgkin A. L., Huxley A. F. A quantitative description of membrane current and its application to conduction and excitation in nerve // The Journal of Physiology. 1952. № 4 (117). С. 500.

98. Hong L., Wan Y., Jain A. Fingerprint image enhancement: Algorithm and performance evaluation // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998. № 8 (20). С. 777–789.

99. Hornik K., Stinchcombe M., White H. Multilayer feedforward networks are universal approximators // Neural Networks. 1989. № 5 (2). С. 359–366.

100. Hubel D. H., Wiesel T. N. Receptive fields of single neurones in the cat's striate cortex // The Journal of Physiology. 1959. № 3 (148). C. 574–591.

101. Izhikevich E. M. . Dynamical Systems in Neuroscience : the Geometry of Excitability and Bursting / E. M. . Izhikevich, MIT Press, 2014. 522 c.

102. Jacobs R. A., Jordan M. I., Nowlan S. J., Hinton G. E. Adaptive Mixtures of Local Experts // Neural Computation. 1991. № 1 (3). C. 79–87.

103. Johnson J. L. Pulse-coupled neural nets: translation, rotation, scale, distortion, and intensity signal invariance for images // Applied Optics. 1994. № 26 (33). C. 6239.

104. Jonathon Phillips P., Moon H., Rizvi S. A., Rauss P. J. The FERET evaluation methodology for face-recognition algorithms // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2000. № 10 (22). C. 1090–1104.

105. Joshi V. B., Raval M. S. Adaptive Threshold for Fingerprint Recognition System Based on Threat Level and System Load // Procedia Computer Science. 2020. (171). C. 498–507.

106. Kainen P. C. Utilizing Geometric Anomalies of High Dimension: When Complexity Makes Computation Easier // Computer Intensive Methods in Control and Signal Processing. 1997. C. 283–294.

107. Kamkar S., Abrishami Moghaddam H., Lashgari R., Erlhagen W. Brain-inspired multiple-target tracking using Dynamic Neural Fields // Neural Networks. 2022. (151). C. 121–131.

108. Kaynak C. Methods of Combining Multiple Classifiers and Their Applications to Handwritten Digit Recognition 1998.

109. Keerthi S. S., Gilbert E. G. Convergence of a generalized SMO algorithm for SVM classifier design // Machine Learning. 2002. № 1–3 (46). C. 351–360.

110. Kim S. H., Nam H., Park Y. H. Temporal Dynamic Convolutional Neural Network for Text-Independent Speaker Verification and Phonemic Analysis // IEEE International Conference on Acoustics, Speech, and Signal Processing. 2021. (2022-May). C. 6742–6746.

111. Kingma D. P., Ba J. L. Adam: A Method for Stochastic Optimization // 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings. 2014.

112. Kittler J., Hatef M., Duin R. P. W., Matas J. On combining classifiers // IEEE Transactions on Pattern Analysis and Machine Intelligence. 1998. № 3 (20). С. 226–239.

113. Klinshov V. V., Kovalchuk A. V., Franović I., Perc M., Svetec M. Rate chaos and memory lifetime in spiking neural networks // Chaos, Solitons & Fractals. 2022. (158). С. 112011.

114. Klinshov V. V., Kovalchuk A. V., Soloviev I. A., Maslennikov O. V., Franović I., Perc M. Extending dynamic memory of spiking neuron networks // Chaos, Solitons & Fractals. 2024. (182). С. 114850.

115. Kohonen T. Self-organized formation of topologically correct feature maps // Biological Cybernetics. 1982. № 1 (43). С. 59–69.

116. Kompa B., Snoek J., Beam A. L. Second opinion needed: communicating uncertainty in medical machine learning // npj Digital Medicine. 2021. № 1 (4). С. 1–6.

117. Kovalchuk A. V., Bellyustin N. S. Online learning algorithm of kernel-based ternary classifiers using support vectors // Optical Memory and Neural Networks (Information Optics). 2013. № 3 (22). С. 193–205.

118. Kovalchuk A. V., Lebedev A. A., Shemagina O. V., Nuidel I. V., Yakhno V. G., Stasenko S. V. Enhancing Cascade Object Detection Accuracy Using Correctors Based on High-Dimensional Feature Separation // Technologies 2025, Vol. 13, Page 593. 2025. № 12 (13). С. 593.

119. Kovalchuk A. V., Telnykh A. A. Object detection system using expert committeets // 9-th International Conference on Pattern Recognition and Image Analysis: New Information Technologies, Conference Proceedings. 2008. (2). С. 212–214.

120. Kovalchuk A. V., Telnykh A. A. Object detection system using expert committees // Pattern Recognition and Image Analysis. 2008. (1). С. 343–346.

121. Kubo Y., Chalmers E., Luczak A. Biologically-inspired neuronal adaptation improves learning in neural networks // Communicative & Integrative Biology. 2023. № 1 (16). С. 2163131.

122. Kuncheva L. I., Bezdek J. C., Duin R. P. W. Decision templates for multiple classifier fusion: an experimental comparison // Pattern Recognition. 2001. № 2 (34). С. 299–314.

123. Kuzhentsov S. O., Nuidel I. V., Panfilov A. I., Yakhno V. G. Image processing by neuron-like algorithms // The Proc. SPIE “Optical Information Science and Technology” – in Optical Memory and Neural Networks. 1998. (3402). С. 479–485.

124. Kuzhetsov S. O., Nuidel I. V., Yakhno V. G. Segmentation and Pattern Recognition of a Composite Image Product by System of Elements with Neural-network Architecture под ред. E. A. Holden, A. Kryuk, Manchester University Press, 1991.С. 591–596.

125. Kuznetsova G. D., Nuidel I. V., Khurlapov P. G., Yakhno V. G. Modeling of normal and pathological sensor activity image transformation versions in an animal’s cortex // The Proc. SPIE “Optical Information Science and Technology” – in Optical Memory and Neural Networks. 1998. (3402). С. 486–499.

126. Levi G., Hassncer T. Age and gender classification using convolutional neural networks // IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops. 2015. (2015-October). С. 34–42.

127. Liu C., Tang K., Qin Y., Lei Q. Bridging Distribution Shift and AI Safety: Conceptual and Methodological Synergies 2025.

128. Liu H., Liu M., Li D., Zheng W., Yin L., Wang R. Recent Advances in Pulse-Coupled Neural Networks with Applications in Image Processing // Electronics 2022, Vol. 11, Page 3264. 2022. № 20 (11). С. 3264.

129. Lu X., Jain A. K. Ethnicity identification from face images // SPIE. 2004. (5404). С. 114–123.

130. Lukoševičius M., Jaeger H. Reservoir computing approaches to recurrent neural network training // Computer Science Review. 2009. № 3 (3). С. 127–149.

131. Makinen E., Raisamo R. Evaluation of Gender Classification Methods with Automatically Detected and Aligned Faces // IEEE Transactions on Pattern Analysis and Machine Intelligence. 2008. № 03 (30). С. 541–547.

132. Malsburg C. Self-organization of orientation sensitive cells in the striate cortex // Kybernetik. 1973. № 2 (14). С. 85–100.

133. Merz C. J. Combining Classifiers Using Correspondence Analysis // Advances in Neural Information Processing Systems. 1997. (10).

134. MOUNTCASTLE V. B. Modality and topographic properties of single neurons of cat's somatic sensory cortex // Journal of neurophysiology. 1957. № 4 (20). С. 408–434.

135. Movellan J. R. Contrastive Hebbian Learning in the Continuous Hopfield Model // Connectionist Models. 1991. С. 10–17.

136. Nalepa J., Kawulok M. Selecting training sets for support vector machines: a review // Artificial Intelligence Review 2018 52:2. 2018. № 2 (52). С. 857–900.

137. Nikolaou N., Brown G. Calibrating AdaBoost for Asymmetric Learning // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2015. (9132). С. 112–124.

138. Nilnumpetch C., Amornsamankul S., Kraipeerapun P. Cascade Generalization and Complementary Neural Networks for Multiclass Classification // International Conference on Electrical, Computer, and Energy Technologies, ICECET 2022. 2022.

139. Nuidel I. V., Kuznetsov S. O. Use of homogeneous neuron-like media for image processing // Radiophysics and Quantum Electronics. 1994. № 8 (37). С. 680–685.

140. Olshausen B. A., Field D. J. Sparse coding with an overcomplete basis set: A strategy employed by V1? // Vision Research. 1997. № 23 (37). С. 3311–3325.

141. Ong Ly C., Unnikrishnan B., Tadic T., Patel T., Duhamel J., Kandel S., Moayedi Y., Brudno M., Hope A., Ross H., McIntosh C. Shortcut learning in medical AI hinders generalization: method for estimating AI model generalization without external data // npj Digital Medicine. 2024. № 1 (7). С. 1–10.

142. Opper M., Kinzel W., Kleinz J., Nehl R. On the ability of the optimal perceptron to generalise // Journal of Physics A: Mathematical and General. 1990. № 11 (23). С. L581.

143. Ortega J., Koppel M., Argamon S. Arbitrating Among Competing Classifiers Using Learned Referees // Knowledge and Information Systems 2001 3:4. 2001. № 4 (3). С. 470–490.

144. O'toole A. J., Deffenbacher K. A., Valentin D., Abdi H. Structural aspects of face recognition and the other-race effect // Memory & Cognition. 1994. № 2 (22). С. 208–224.

145. Panis G., Lanitis A. An Overview of Research Activities in Facial Age Estimation Using the FG-NET Aging Database // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2015. (8926). С. 737–750.

146. Parkhi O. M., Vedaldi A., Zisserman A. Deep Face Recognition // British Machine Vision Conference. 2015. С. 41.1-41.12.

147. Platt J. Fast training of support vector machines using sequential minimal optimization под ред. B. Scholkopf, C. J. C. Burges, A. J. Smola, Cambridge, MA: MIT Press, 1999.С. 185–208.

148. Prodromidis A. L., Chan P. K., Stolfo S. J. Meta-learning in distributed data mining systems: Issues and approaches // Advances in Distributed and Parallel Knowledge Discovery. 1999. (3).

149. Ranganath H. S., Kuntimad G., Johnson J. L. Pulse coupled neural networks for image processing // Conference Proceedings - IEEE SOUTHEASTCON. 1995. С. 37–43.

150. Ravi S., Larochelle H. Optimization as a Model for Few-Shot Learning // International Conference on Learning Representations. 2016.

151. Roli F., Didaci L., Marcialis G. L. Adaptive Biometric Systems That Can Improve with Use // Advances in Biometrics: Sensors, Algorithms and Systems. 2008. С. 447–471.

152. Rothe R., Timofte R., Gool L. Van Deep Expectation of Real and Apparent Age from a Single Image Without Facial Landmarks // International Journal of Computer Vision. 2018. № 2–4 (126). С. 144–157.

153. Rudin C. Stop Explaining Black Box Machine Learning Models for High Stakes Decisions and Use Interpretable Models Instead // Nature Machine Intelligence. 2018. № 5 (1). С. 206–215.

154. Rwigema J., Mfitumukiza J., Tae-Yong K. A hybrid approach of neural networks for age and gender classification through decision fusion // Biomedical Signal Processing and Control. 2021. (66). С. 102459.

155. Samek W., Müller K.-R. Towards Explainable Artificial Intelligence // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2019. (11700 LNCS). С. 5–22.

156. Scellier B., Bengio Y. Equilibrium propagation: Bridging the gap between energy-based models and backpropagation // Frontiers in Computational Neuroscience. 2017. (11). С. 246298.

157. Schneiderman H., Kanade T. Probabilistic modeling of local appearance and spatial relationships for object recognition // Proceedings of IEEE Conference on Computer Vision snd Pattern Recognition. 1998. С. 45–51.

158. Schölkopf B., Smola A. J., Williamson R. C., Bartlett P. L. New Support Vector Algorithms // Neural Computation. 2000. № 5 (12). С. 1207–1245.

159. Schroff F., Kalenichenko D., Philbin J. FaceNet: A Unified Embedding for Face Recognition and Clustering // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2015. (07-12-June-2015). С. 815–823.

160. Seewald A. K., Fürnkranz J. An evaluation of grading classifiers // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2001. (2189). С. 115–124.

161. Snell J., Swersky K., Zemel R. Prototypical Networks for Few-shot Learning // Advances in Neural Information Processing Systems. 2017. (2017-December). С. 4078–4088.

162. Snell J., Swersky K., Zemel R. Prototypical Networks for Few-shot Learning // Advances in Neural Information Processing Systems. 2017. (2017-December). С. 4078–4088.

163. Sokolov M. E., Tel'nykh A. A., Kovalchuk A. V., Bellyustin N. S., Nuidel I. V., Yakhno V. G. Face recognition using «lateral inhibition» function features // Optical Memory and Neural Networks (Information Optics). 2009. № 1 (18). С. 1–5.

164. Soloviev I., Kovalchuk A., Klinshov V. Dynamic convolution for image matching // The European Physical Journal Special Topics. 2024. С. 1–9.

165. Stasenko S., Telnykh A., Shemagina O., Nuidel I., Kovalchuk A., Yakhno V. Biomorphic artificial intelligence system for pattern recognition problems with adaptive error correction // Proceedings 4th International Conference «Neurotechnologies and Neurointerfaces», CNN 2022. 2022. С. 185–189.

166. Stasenko S. V., Kovalchuk A. V., Eremin E. V., Drugova O. V., Zarechnova N. V., Tsirkova M. M., Permyakov S. A., Parin S. B., Polevaya S. A. Using Machine Learning Algorithms to Determine the Post-COVID State of a Person by Their Rhythmogram // Sensors 2023, Vol. 23, Page 5272. 2023. № 11 (23). С. 5272.

167. Stasenko S. V., Lebedev A. A., Shemagina O. V., Nuidel I. A., Kovalchuk A. V., Yakhno V. G. Adaptive correction of the multi-cascade detector of biomorphic artificial intelligence system for pattern recognition problems // Proceedings 6th International Conference «Neurotechnologies and Neurointerfaces», CNN 2024. 2024. С. 221–225.

168. Sung F., Yang Y., Zhang L., Xiang T., Torr P. H. S., Hospedales T. M. Learning to Compare: Relation Network for Few-Shot Learning // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2017. С. 1199–1208.

169. Tanaka G., Yamane T., Héroux J. B., Nakane R., Kanazawa N., Takeda S., Numata H., Nakano D., Hirose A. Recent advances in physical reservoir computing: A review // Neural Networks. 2019. (115). С. 100–123.

170. Tao C. W. Unsupervised fuzzy clustering with multi-center clusters // Fuzzy Sets and Systems. 2002. № 3 (128). С. 305–322.

171. Taori R., Dave A., Shankar V., Carlini N., Recht B., Schmidt L. Measuring Robustness to Natural Distribution Shifts in Image Classification // Advances in Neural Information Processing Systems. 2020. (2020-December).

172. Thompson M., Duda R. O., Hart P. E. Pattern classification and scene analysis / M. Thompson, R. O. Duda, P. E. Hart, A Wiley-Interscience publication, 1974. 370 с.

173. Tin Kam H. Multiple classifier combination: Lessons and next steps // Hybrid Methods in Pattern Recognition. 2002. С. 171–198.

174. Ting K. Decision Combination Based on the Characterisation of Predictive Accuracy // Intelligent Data Analysis. 1997.

175. Ting K. M., Witten I. H. Issues in Stacked Generalization // Journal Of Artificial Intelligence Research. 2011. (10). С. 271–289.

176. Turk M., Pentland A. Eigenfaces for Recognition // Journal of Cognitive Neuroscience. 1991. № 1 (3). С. 71–86.

177. Tyukin I. Y., Gorban A. N., Alkhudaydi M. H., Zhou Q. Demystification of few-shot and one-shot learning // International joint conference on neural networks (IJCNN). 2021.

178. Vapnik V. Estimation of Dependences Based on Empirical Data: Springer Series in Statistics (Springer Series in Statistics) / V. Vapnik, Berlin, Heidelberg: Springer-Verlag, 1982.

179. Vinyals O., Blundell C., Lillicrap T., Kavukcuoglu K., Wierstra D. Matching Networks for One Shot Learning // Advances in Neural Information Processing Systems. 2016. С. 3637–3645.

180. Viola P., Jones M. Rapid object detection using a boosted cascade of simple features // Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition. 2001. (1).

181. Wang Y., Yao Q., Kwok J. T., Ni L. M. Generalizing from a Few Examples // ACM Computing Surveys (CSUR). 2020. № 3 (53).

182. White B. W., Rosenblatt F. Principles of Neurodynamics: Perceptrons and the Theory of Brain Mechanisms // The American Journal of Psychology. 1963. № 4 (76). C. 705.

183. Wilson H. R., Cowan J. D. Excitatory and inhibitory interactions in localized populations of model neurons // Biophysical journal. 1972. № 1 (12). C. 1–24.

184. Wilson H. R., Cowan J. D. A mathematical theory of the functional dynamics of cortical and thalamic nervous tissue // Kybernetik. 1973. № 2 (13). C. 55–80.

185. Wolpert D. H. Stacked generalization // Neural Networks. 1992. № 2 (5). C. 241–259.

186. Yang X. S., He X. S., Fan Q. W. Mathematical framework for algorithm analysis // Nature-Inspired Computation and Swarm Intelligence: Algorithms, Theory and Applications. 2020. C. 89–108.

187. Zhang K., Zhang J., Wang Q., Zhong Z. DyNet: Dynamic Convolution for Accelerating Convolutional Neural Networks // Proceedings of ICLR 2020 Conference. 2020.

188. Zhang W., Wang Y. Singular Point Detection in Fingerprint Image // The 5th Asian Conference on Computer Vision. 2002.

189. Zhou J., Gu J., Zhang D. Singular Points Analysis in Fingerprints Based on Topological Structure and Orientation Field // Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics). 2007. (4642 LNCS). C. 261–270.

190. Zhou S., Liu C., Ye D., Zhu T., Zhou W., Yu P. S. Adversarial Attacks and Defenses in Deep Learning: From a Perspective of Cybersecurity // ACM Computing Surveys. 2022. № 8 (55).

ПРИЛОЖЕНИЯ

Приложение А: Алгоритм Adaboost

Алгоритм AdaBoost [70] является одним из способов реализации некоторого класса ассоциативных машин. Эксперты сети, работающие на основе метода усиления, обучаются на примерах из различных распределений. AdaBoost отличается адаптивной настройкой ошибок слабых гипотез, что является причиной его названия. Производительность алгоритма ограничена только теми распределениями, которые формируются в процессе обучения.

Важным теоретическим свойством AdaBoost является следующая теорема [69]:

Теорема о верхней границе Adaboost:

Пусть при вызове алгоритма AdaBoost слабая модель обучения генерирует гипотезы с ошибками , определяемыми как:

$$\epsilon_k = \sum_i D_k(i)[y_i \neq h_k(x_i)]$$

,где $D_k(i)$ — веса объектов на k-ой итерации. Пусть также $\epsilon_k < 0.5$, тогда ошибка окончательной гипотезы ограничена сверху величиной:

$$\epsilon_{final} \leq 2^K \prod_{k=1}^{K} \sqrt{\epsilon_k(1-\epsilon_k)}$$

Согласно этой теореме, если ошибка слабой гипотезы лишь немного меньше 0.5, то ошибка обучения окончательной гипотезы экспоненциально стремится к нулю. Это, однако, не гарантирует малую ошибку на тестовых данных.

Пусть дано множество маркированных примеров $\{(x_i, y_i)\}$, σ_i – начальный вес каждого примера, число итераций K.

Таблица 1 — Алгоритм обучения Adaboost

5. Инициализация весов $D_1(i) = \sigma_i$.
6. Повторить для $k = 1, \dots, K$:

a. Нормализовать веса $D_k(i)$, чтобы они образовывали распределения вероятности.
b. Обучить бинарный классификатор $h_k(x) \to \{-1, +1\}$ с учетом весов каждого примера $D_k(i)$.
c. Вычислить ошибку каждого классификатора: $\epsilon_j = \sum_i D_k(i)\left[y_i \neq h_j(x_i)\right]$.
d. Выбрать классификатор h_k с минимальной ошибкой ϵ_k^{min}
e. Рассчитать вес: $\alpha_k = \frac{1}{2}\ln\left(\frac{1-\epsilon_k^{min}}{\epsilon_k^{min}}\right)$.
f. Обновить веса $D_{k+1}(i) = D_k(i)\exp\left(-\alpha_k y_i h_k(x_i)\right)$

7. Итоговый классификатор: $H(x) = \text{sign}(\sum \alpha_k h_k(x))$.

Алгоритм останавливается, если значение общей ошибки распознавания удовлетворяет неравенству:

$$\epsilon_{final} \leq \epsilon_{threshold}$$

,где $\epsilon_{threshold}$ выбирается эмпирически по результатам вычислительных экспериментов.

Приложение Б: Модифицированное преобразование Цензуса (MCT)

Модифицированное преобразование Цензуса [72] преобразует изображение в признаковое пространство, которое используется для описания локальных структур, таких как границы, сегменты, седловые точки и т.д. Пусть S — фрагмент изображения (окно), к которому применяется модифицированное преобразование. Этот фрагмент разбивается на 9 равных сегментов, которые обозначаются как $C_1, C_2, \ldots, C_9$ расположены в виде сетки 3×3 (Рис. 1).

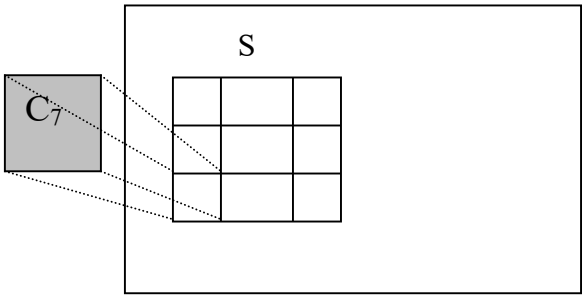


Рисунок 1 — Разбиение фрагмента на 9 областей.

Для каждого сегмента C_i вычисляется средняя интенсивность:

$$\mu_{C_i} = \frac{1}{|C_i|}\sum_{(u,v)\in C_i} I(u,v) \quad (1)$$

, где $I(u,v)$ — интенсивность пикселя с координатами (u,v) в сегменте C_i, а $|C_i|$ — количество пикселей в сегменте.

Средняя интенсивность всего окна S вычисляется по формуле:

$$\mu_R = \frac{1}{|S|}\sum_{(u,v)\in S} I(u,v) \quad (2)$$

, где $|S|$ — общее количество пикселей в окне *S*.

После вычисления средних значений для каждого сегмента и всего окна, для каждого сегмента C_i определяется бинарный признак $T(C_i)$ следующим образом:

$$T(C_i) = \begin{cases} 1, \text{если } \mu_{C_i} \geq \mu_R \\ 0, \text{иначе} \end{cases} \quad (3)$$

Бинарный паттерн, сформированный при помощи преобразования Цензуса, кодируется единственным целым числом, индексом паттерна. Каждое окно *S*, разбитое на 9 сегментов $C_1, C_2, \ldots, C_9$, имеет 9-ти битный индекс паттерна, который может быть записан в десятичной системе от 0 до 511 следующим образом:

$$H(S) = \sum_{i=1}^{9} T(C_i) \cdot 2^{i-1} \quad (4)$$

,где $T(C_i)$ — бинарный признак сегмента C_i.

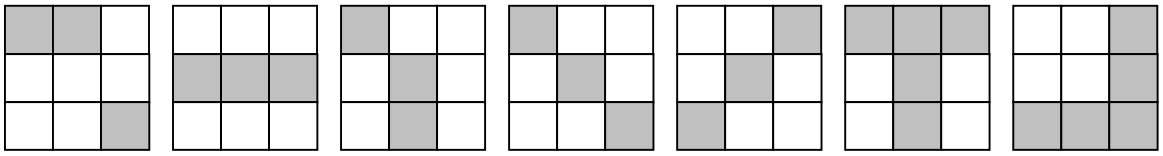


Рисунок 2 — Примеры кодов и соответствующих активных фрагментов

Основные особенности модифицированного подхода:

- **Масштабируемость.** Преобразование применяется к окнам любого масштаба, что позволяет адаптироваться к объектам различных размеров.
- **Гибкое соотношение сторон.** Поскольку окно S может иметь произвольное соотношение сторон, метод универсален для различных задач детекции.
- **Адаптивность к неоднородному освещению.** Использование среднего значения помогает снизить влияние локальных вариаций яркости.