МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)

В.В. МЯСНИКОВ

РАСПОЗНАВАНИЕ ОБРАЗОВ И МАШИННОЕ ОБУЧЕНИЕ. ОСНОВНЫЕ ПОДХОДЫ

Рекомендовано редакционно-издательским советом федерального государственного автономного образовательного учреждения высшего образования «Самарский национальный исследовательский университет имени академика С.П. Королева» в качестве учебного пособия для обучающихся по основной образовательной программе высшего образования по специальности 10.05.03 Информационная безопасность автоматизированных систем

САМАРА
Издательство Самарского университета
2023

УДК 004.93(075)
ББК 3813я7
М994

Рецензенты: д-р техн. наук, доц. С. Б. Попов,
д-р техн. наук, проф. В. А. Фурсов

Мясников, Владислав Валерьевич

М994 Распознавание образов и машинное обучение. Основные подходы: учебное пособие / *В.В. Мясников.* – Самара: Издательство Самарского университета, 2023. – 196 с. : ил.

ISBN 978-5-7883-1929-2

Пособие представляет собой краткое изложение материалов по ключевым разделам современной теории искусственного интеллекта: машинному обучению и распознаванию образов (классификации). Пособие включает, в частности, классификацию задач машинного обучения, описание основных методов ранжирования/упорядочивания объектов и методов распознавания образов (классификации): геометрических, статистических и алгебраических, а также кратких основ теории последовательного анализа и классификации, теории обучения с подкреплением и искусственных нейронных сетей.

Предназначено для студентов факультета информатики, обучающихся по направлению подготовки 01.03.02 Прикладная математика и информатика и специальности 10.05.03 Информационная безопасность автоматизированных систем.

Подготовлено на кафедре геоинформатики и информационной безопасности.

УДК 004.93(075)
ББК 3813я7

ISBN 978-5-7883-1929-2

ОГЛАВЛЕНИЕ

ВВЕДЕНИЕ

Методы машинного обучения и распознавания образов в современное время являются ядром технологий искусственного интеллекта, используемых с целью построения современной цифровой экономики.

Настоящее учебное пособие предназначено для изучения студентами ключевых разделов этих научных направлений. В нем последовательно рассматриваются:

– классификация задач машинного обучения,

– основные методы ранжирования/упорядочивания объектов,

– основные методы распознавания образов и классификации, включая геометрические, статистические и алгебраические (предложенные академиком Ю.И. Журавлевым);

– методы комбинирования решающих правил;

– элементы теории надежности обучения, предложенной в 70-х годах советскими учеными В.Н. Вапником и А.Я. Червоненкисом;

– методы последовательного анализа и классификации, включая последовательный критерий отношения вероятностей Вальда и байесовскую последовательную решающую процедуру;

– теории обучения с подкреплением;

– основы искусственных нейронных сетей.

Изложение ведется в единой терминологии и обозначениях, что упрощает процесс знакомство с материалом. Отдельные разделы пособия содержат оригинальный материал, являющийся результатом научных исследований.

1 МЕТОДЫ МАШИННОГО ОБУЧЕНИЯ

1.1 Классификация методов машинного обучения

Методы машинного обучения и *принятия решений* находят все более широкое применение в различных областях современной цифровой экономики [1–3]. Разрабатываемые научными группами для решения каких-то совершенно конкретных задач, некоторые методы и алгоритмы оказываются совершенно неожиданно востребованными в других или смежных научных направлениях, вызывая синергетический эффект повсеместно.

Для конкретизации подкласса решаемых в машинном обучении задач дадим формальное представление типовых задач машинного обучения и теории предпочтений. В задачах этого типа обычно выделяют *объекты* и *классы/метки* (англ.: *labels*), приписываемые конкретному объекту. Обычно выделяют три типа задач [4]:

– *ранжирование/упорядочивание меток* (англ.: label ranking), когда для конкретного объекта производится упорядочивание списка меток. Данная задача является обобщением классической задачи классификации или распознавания образов, когда поступивший объект необходимо отнести к конкретному классу. Описание методов этого класса представлено далее во втором разделе;

– *ранжирование/упорядочивание примеров* (англ.: instance ranking), когда множество конкретных объектов следует распределить по классам/меткам, причём сами метки/классы являются строго упорядоченными. Примером задач такого типа является распределение работ на конференцию по типовым пяти категориям: *reject, weak reject, borderline, weak accept* и *accept;*

– *ранжирование/упорядочивание объектов* (англ.: object ranking), когда множество поступивших объектов необходимо упорядочить в соответствии с некоторым представлением об их «полезности». Собственно метки в задачах этого типа не используются; Описание методов этого класса представлено далее в настоящем разделе далее.

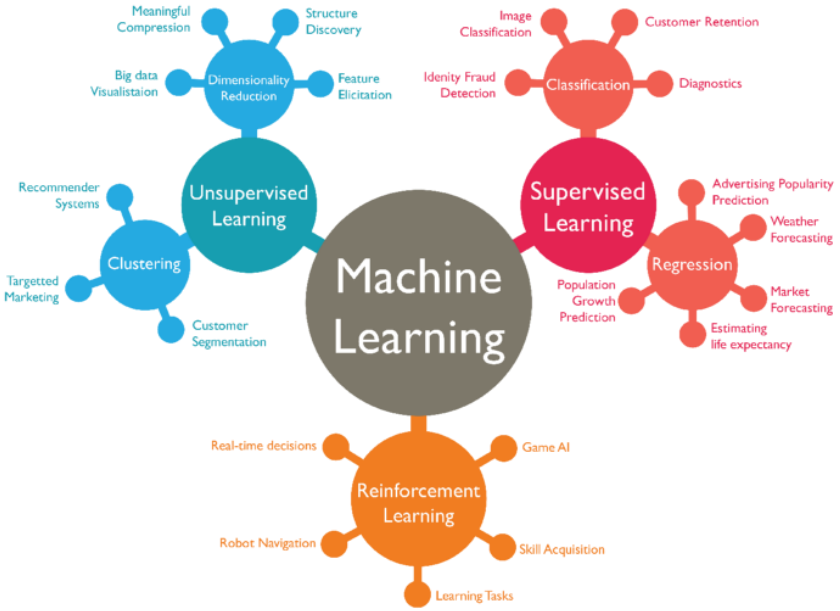


Рисунок 1.1 – Классификация методов машинного обучения и распознавания. Перевод основных терминов: machine learning – машинное обучение, supervised learning – обучение с учителем, unsupervised learning – обучение без учителя, reinforcement learning – обучение с подкреплением, classification – классификация, regression – регрессия, dimensionality reduction – снижением размерности, clustering – кластеризация. Остальные английские термины означают области применения или приложения

1.2 Упорядочивание объектов и теория предпочтений

1.2.1 Введение

Настоящий подраздел посвящен одному достаточно узкому, на первый взгляд, вопросу, связанному с реконструкцией функций по их знаковому представлению, то есть результатам сравнений их значений. В компьютерном зрении, как одном из направлений искусственного интеллекта, подобные постановки задачи, отличающейся использованием не абсолютных значений функций, а результатов парных сравнений их значений, появились в работах по цифровой обработке сигналов и изображений в 90-х и были связаны с вопросами текстурного описания изображений (работа DC. He и L. Wang [5], 1990 г.) и анализа цифровых сигналов (российские работы О.В. Цветкова [6, 7], 1991–1992 гг.). В дальнейшем это направление получило значительную практическую проработку в задачах компьютерного зрения в рамках т.н. *локальных бинарных шаблонов* (англ. LBP – Local Binary Pattern) [8–12], а теоретическую – в работах российских авторов А.Н. Каркищенко и А.В. Гончарова [13–15]. Работы российских авторов были посвящены *знаковым представлениям изображений* (термин используется по аналогии с монографией [16]), как и более поздняя публикация [17] автора настоящего пособия.

Исторически существенно раньше в экономике получила развитие т.н. *теория полезности*. У.Ст. Джевонс, К. Менгер и Л. Вальрас во второй половине XIX века предложили количественную теорию, а Ф. Эджуорт, В. Парето и И. Фишер – порядковую теорию полезности. Последняя предполагает работу только с результатами *сопоставлений полезностей* объектов/событий или с *предпочтениями* (в частном случае – при *парном сравнении* объектов/событий). К настоящему времени теория, основанная на предпочтениях, стала общепринятой и наиболее распространённой

и составляет значимую часть *теорий принятия решений и машинного обучения* [18–21].

Совершенно очевидна аналогия, возникающая при рассмотрении *знаковых представлений* цифровых сигналов и изображений и *парных сравнений* теории полезности и предпочтений: в обоих случаях делается попытка анализировать/реконструировать некоторую скрытую конструкцию по её проявлениям, представленным как результаты сравнений её реакций на входные данные. При этом принципиальным отличием от задач классификации и регрессии, составляющим классическое ядро методов машинного обучения и распознавания [21], является иная форма представления данных – *прецедент содержит информацию не о конкретном объекте, а о результате его сопоставления/сравнения с другим объектом*. Задачи такого типа в теории полезности и принятия решений получили название «*выявление предпочтений*» (англ.: preference elicitation).

1.2.2 Постановка задачи

Представим более формальную постановку рассматриваемой проблемы. С учётом превалирующего влияния в этом направлении работ по теории полезности и принятия решений [19–21] будем опираться на принятые в этих работах формальные обозначения.

Пусть на множестве $\Omega \equiv \{\omega_j\}_{j\in J}$ объектов задано отношение порядка и/или строгого частичного порядка, обозначаемое далее символами "$\le$" и/или "$\prec$". Записи $\omega_i \succ \omega_j$ и $\omega_j \prec \omega_i$ считаем эквивалентными, в ситуации $\omega_i \le \omega_j \wedge \omega_j \le \omega_i$ объекты считаем неотличимыми и формально записываем $\omega_i \sim \omega_j$. В зависимости от постановки задачи считаем, что указанные отношения могут быть определены одним из двух способов. Наиболее простой вариант

предполагает наличие некоторой (в общем случае неизвестной) числовой *функции полезности* (unity function, scoring function) $u:\Omega\to\mathbb{R}$, характеризующей *абсолютное предпочтение*. Причём выполнение $u(\omega_i)<u(\omega_j)$ эквивалентно $\omega_i\prec\omega_j$, аналогично $u(\omega_i)\le u(\omega_j)\Leftrightarrow\omega_i\le\omega_j$, а $u(\omega_i)=u(\omega_j)\Leftrightarrow\omega_i\sim\omega_j$.

Очевидно, что конкретная функция полезности порождает целый класс *эквивалентных* ей в смысле введённого отношения порядка функций полезности. Действительно, если $u(\omega)$ является функцией полезности, то для любой монотонно возрастающей функции $g:\mathbb{R}\to\mathbb{R}$ новая функция $g(u(\omega))$ порождает то же отношение порядка на Ω.

Второй вариант предполагает использование так называемой *функции предпочтения* (preference function) $p:\Omega\times\Omega\to\mathbb{R}$, характеризующей *относительное предпочтение*. Для этой функции условие $p(\omega_i,\omega_j)>0$ эквивалентно условию $\omega_i\succ\omega_j$, а $p(\omega_i,\omega_j)=0\Leftrightarrow\omega_i\sim\omega_j$. Если на функцию полезности в общем случае ограничения не накладываются, то на функцию предпочтений накладываются ограничения, естественным образом следующие из свойств отношений порядка. А именно:

– асимметричность по аргументу

$$p(\omega_j,\omega_i)>0\Leftrightarrow p(\omega_i,\omega_j)<0;$$

– транзитивность

$$p(\omega_i,\omega_j)>0\wedge p(\omega_j,\omega_t)>0\Rightarrow p(\omega_i,\omega_t)>0,$$
$$p(\omega_i,\omega_j)\ge 0\wedge p(\omega_j,\omega_t)\ge 0\Rightarrow p(\omega_i,\omega_t)\ge 0;$$

и т.п.

Достаточно очевидно, что в случае наличия любой из указанных функций альтернативная функция может быть легко получена. А именно:

– предпочтение через полезность

$$p\left(\omega_j,\omega_i\right)=u\left(\omega_j\right)-u\left(\omega_i\right);$$

– полезность через предпочтение

$$u\left(\omega_j\right)=p\left(\omega_j,\omega^*\right)\ \left(u\left(\omega^*\right)=p\left(\omega^*,\omega^*\right)=0\right).$$

Следует отметить, что для подавляющего большинства практических приложений:

– сами объекты не являются формальными математическими объектами;

– функции полезности или предпочтений часто оказываются неизвестными.

Поэтому вместо самих объектов на практике часто имеют дело с описаниями объектов в виде векторов признаков (альтернативные названия: показателей, критериев, факторов и др.) некоторого *N*-мерного пространства *X* (в некоторых работах используются более сложные описания в виде определенных математических структур [20], которые в настоящей работе не рассматриваются):

$$\mathbf{x}\equiv\mathbf{x}\left(\omega\right)\in X\ .$$

Для сокращения записи примем: $\mathbf{x}_j\equiv\mathbf{x}\left(\omega_j\right)$. При этом функции полезности и предпочтений в машинном обучении конструируются именно по описаниям, то есть формально в виде:

$$p\left(\mathbf{x},\mathbf{x}_j\right),u\left(\mathbf{x}\right). \qquad (1.1)$$

Для исключения неоднозначности форму записи $p(\omega_j,\omega_i),u(\omega_j)$ далее будем соотносить с идеальными и неизвестными функциями полезности и предпочтений, а форму (1.1) – с реконструируемыми, связанными с используемыми моделями и параметрами. Также для упрощения изложения примем следующие сокращения:

$$p_{ij} \equiv p(\omega_i,\omega_j),\ u_j \equiv u(\omega_j).$$

Далее, информация о предпочтениях при обучении может задаваться [20]:

– перечислением некоторых объектов со значениями желаемой функции полезности – *прямая информация*;

– в виде результатов *парных сравнений* p_{ij} для некоторого подмножества объектов – *косвенная информация*.

Заметим также, что указанная выше косвенная информация также может быть представлена двумя способами:

– *явно*, то есть в виде значения самой функции предпочтения $p(\omega_j,\omega_i)$ или синтезированной функции предпочтения с использованием функции полезности $p(\omega_j,\omega_i)=u(\omega_j)-u(\omega_i)$;

– неявно, в виде *знакового представления* вида:

$$z_{ij} \equiv z(\omega_i,\omega_j)=\begin{cases} 1, & p(\omega_i,\omega_j)>0; \\ 0, & p(\omega_i,\omega_j)=0; \\ -1, & p(\omega_i,\omega_j)<0. \end{cases} \qquad (1.2)$$

Примерами предоставления косвенной информации в виде парных сравнений являются: выбор конкретной позиции в выдаче/списке поисковика (выбранная позиция является предпочти-

тельной по сравнению с остальными), выбор конкретной позиции при заказе гостиницы, выбор конкретного маршрута в навигаторе и т.п.

В результате, *задача реконструкции предпочтений* формулируется при дополнительных ограничениях и вариантах постановки. Наиболее компактно они представлены на диаграмме монографии [20]. Переведённый вариант данный диаграммы представлен на рисунке 1.2. Он подразумевает выбор:

– *модели предпочтений (абсолютная, относительная);*

– *типа шкалы для модели абсолютных предпочтений (логическая, градуированная, непрерывная, конечная или бесконечная);*

– *полноты реконструируемых отношений (полное, частичное или парное упорядочивание всех объектов) для модели относительных предпочтений;*

– *способа представления обучающей информации (прямая или косвенная, обо всех объектах* или о подмножестве, косвенная информация даётся в явном виде или в неявном, то есть с использованием *знакового представления* и т.д.).

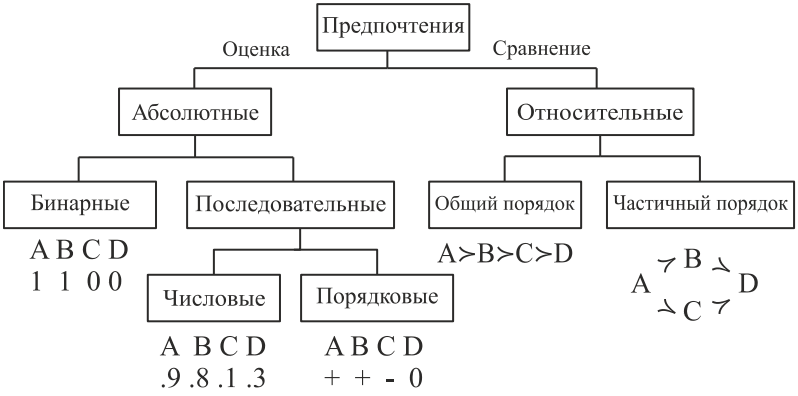


Рисунок 1.2 – Варианты итогового представления решения задачи оценки предпочтений

1.2.3 Критерии качества

Оценка качества результата решения задачи оценки полезности и предпочтений зависит от способа представления обучающей информации, выбранной модели предпочтений и полноты реконструированных отношений. Выделим два принципиально различных случая, имеющих отношение к настоящей работе.

В первом принципиально известной является исходная функция полезности $u(\omega)$ для всего множества потенциально существующих объектов. В этом случае качество реконструкции функции можно охарактеризовать погрешностью (ошибкой) следующего вида:

$$\varepsilon^s = \frac{1}{|J|}\sum_{j\in J}\left|u_j - u\left(\mathbf{x}_j\right)\right|^s, \quad (s\in\mathbb{N}). \tag{1.3}$$

Здесь $u_j, u\left(\mathbf{x}_j\right)$ – истинное и реконструированное значения функции полезности, соответственно.

Во втором случае известными являются результаты парных сравнений на некотором подмножестве исходных объектов:

$$z_{ij} = z\left(\omega_i, \omega_j\right), \quad (i,j)\in\Theta\subseteq J\times J .$$

В случае, когда множество Θ не позволяет реконструировать полный порядок через транзитивное замыкание отношения $\left(\Theta\subset J\times J\right)$ или хотя бы частичный, а допускает только анализ парных сравнений, то целесообразными вариантами анализа качества реконструкции функции полезности или предпочтений является число неверно реконструированных отношений – *расстояние Кендалла* для парных сравнений (Kendall's distance), – вычисляемое в виде

$$d = \left| \left\{ (i,j) : z\left(\omega_i, \omega_j\right) \neq z\left(\mathbf{x}\left(\omega_i\right), \mathbf{x}\left(\omega_j\right)\right),\ \ (i,j) \in \Theta \right\} \right|.$$

Эта величина может также быть использована и в «нормированном» варианте, как частота или оценка соответствующей вероятности ошибок в отношениях:

$$\tilde{d} = d \cdot |\Theta|^{-1}. \tag{1.4}$$

Более сложные способы оценки качества включают: C-индекс, статистику Jonckheere-Terpstra [20] и др.

В случае, когда множество Θ позволяет реконструировать полный порядок объектов через транзитивное замыкание отношения $\left(\Theta = J \times J\right)$, могут быть использованы более мощные способы оценки качества [16], учитывающие и абсолютное положение объектов в исходном порядке, и величину изменения ранга и т.п. В данной работе такая постановка задачи не рассматривается.

1.2.4 Метод парных сравнений (беспризнаковые методы)

Методы парных сравнений первоначально использовались для упорядочивания объектов/альтернатив, которые не имели признакового описания (например, упорядочивание типов фруктов для продаж в магазинах). Результаты таких сравнений обычно [19, 22] представлялись в виде матрицы $\left(c_{ij}\right)$ абсолютных частот предпочтения *i*-го объекта/альтернативы над *j*-ым. Для анализа таких данных в первой половине XX века были предложены две модели: Тёрстоуна и Брэдли-Терри.

Модель Тёрстоуна (Thurstone's model), предложенная в 1927 году [23], исходила из предположения, что полезность конкретного объекта $u\left(\omega\right)$ представляет собой случайную величину с нор-

мальным законом распределения. То есть, для объектов ω_0, ω_1 имеем функцию плотности вероятности:

$$f_u\left(u|\omega_j\right) \sim \mathrm{N}\left(\mu_j, \sigma_j^2\right).$$

Тогда, очевидно:

$$u(\omega_1) - u(\omega_0) \sim \mathrm{N}\left(\mu_1 - \mu_0, \sigma_{10}^2\right),$$
$$\sigma_{10}^2 \equiv \sigma_1^2 + \sigma_0^2 - 2\rho_{10}\sigma_1\sigma_0,$$

здесь ρ_{10} – коэффициент корреляции. Откуда

$$P(\omega_1 \succ \omega_0) = P\left(u(\omega_1) - u(\omega_0) > 0\right) = \Phi\left(\frac{\mu_1 - \mu_0}{\sigma_{10}}\right), \qquad (1.5)$$

здесь $\Phi(\ldots)$ – функция Лапласа. При численной оценке вероятности (1.5) как относительной частоты соответствующих предпочтений, вычисленной по матрице $\left(c_{ij}\right)$, имеем оценку величины предпочтения между альтернативами:

$$\mu_1 - \mu_0 \hat{=} \sigma_{10}\Phi^{-1}\left(\frac{c_{10}}{c_{10} + c_{01}}\right).$$

В более упрощённом варианте модели (Thurstone's Case 5 model) предполагается отсутствие корреляции и равные дисперсии у функции полезности, которые за счёт выбора масштаба можно положить равными 0.5:

$$\sigma_1^2 = \sigma_0^2 = 0.5, \;\; \rho_{10} = 0 \;\Rightarrow\; \sigma_{10}^2 \equiv 1.$$

В этом случае выражение (1.5) упрощается:

$$\mu_1 - \mu_0 \triangleq \Phi^{-1}\left(\frac{c_{10}}{c_{10}+c_{01}}\right).$$

Модель Брэдли-Терри (Bradley-Terry model), предложенная в 1952 году [19], исходила из похожих предположений. Оценивая вероятность (1.5) в виде (s – числовой неотрицательный параметр)

$$P(\omega_1 \succ \omega_0) = \frac{\pi_1}{\pi_1+\pi_0}, \quad \pi_j = \exp\left(\mu_j / s\right), \tag{1.6}$$

имеем следующую оценку величины предпочтений между альтернативами:

$$\mu_1 - \mu_0 \triangleq s\left(\ln\left(\frac{c_{10}}{c_{10}+c_{01}}\right) - \ln\left(1 - \frac{c_{10}}{c_{10}+c_{01}}\right)\right).$$

Позднее было показано, что соответствующее (1.6) распределение функции полезности является распределением Гумбеля (Gumbel distribution) [22].

1.2.5 Выявление предпочтений (preference elicitation) для объектов, описываемых признаками

Существующие подходы к реконструкции функции полезности или предпочтения для рассматриваемой проблемы мультикритериального ранжирования/упорядочивания объектов на основе парных сравнений существенно различаются в зависимости от способа/протокола получения этих данных. Ниже представлены ключевые направления и решения.

Метод анализа иерархий (МАИ, англ: Analytic Hierarchy Process) используется для мультикритериального ранжирования в ситуации, когда ограничение в получении информации отсутствует. В этом случае первоначально формируются матрицы

$\left(m_{ij}^n\right)_{i,\,j\in J}$, $n=\overline{0,N-1}$, составленные из результатов опроса пользователей относительно предпочтений *i*-го объекта/альтернативы над *j*-ым по *n*-му критерию. Правый собственный вектор $\mathbf{v}^n=\left(v_0^n,\ldots,v_{|J|-1}^n\right)^T$ указанной матрицы, характеризующий полезность объектов/альтернатив по *n*-му критерию, используется для формирования итоговой полезности объектов/альтернатив в виде скалярного произведения с весовым вектором критериев $\mathbf{w}$:

$$u\left(\omega_j\right)=\sum_{n=0}^{N-1}w_n v_j^n\,. \tag{1.7}$$

N-мерный весовой вектор критериев $\mathbf{w}$ получается независимо путем анализа матрицы $\left(\kappa_{nm}\right)_{n,m=\overline{0,N-1}}$, элементы κ_{nm} которой – суть результаты парных сравнений экспертами *n*-го и *m*-го критериев (при ограничении $\kappa_{nm}=\left(\kappa_{mn}\right)^{-1}$). В результате, элементы вектора весов $\mathbf{w}$, участвующие в вычислении функции полезности (7), задаются в виде:

$$w_n=\frac{1}{N}\sum_{m=0}^{N-1}\kappa_{nm}\left(\sum_{t=0}^{N-1}\kappa_{tm}\right)^{-1},\quad n=\overline{0,N-1}\,.$$

Основной проблемой МАИ является чрезвычайно большое число парных сопоставлений (как объектов, так и критериев), что на практике может представлять проблему [25–26]. Поэтому, оставляя в принципе неизменной линейную модель выражения (1.7), для реконструкции предпочтений часто используют иные методы и алгоритмы.

А именно, в работах [25, 28] рассматривается следующая *линейная модель*, связанная с *гипотезой «обобщённой аддитивной независимости» признаков*:

$$u\left(\mathbf{x}(\omega)\right)=\sum_{n=0}^{N-1} w_n \cdot v_n\left(x_n(\omega)\right).$$

Здесь $v_n(\ldots)$ – некоторая числовая функция, подстраиваемая для n-го критерия/признака. В работе [26] данная функция реконструировалась на дискретном множестве значений, на которые авторы квантовали значения n-го признака. В работе [27] она полагалась тождественной аргументу, а в работе [28] предполагалась нелинейной, но предопределённой. Математический аппарат во всех указанных работах предполагал байесовский принцип оценивания неизвестного линейного вектора параметров $\mathbf{w}$.

1.2.6 Знаковые представления изображений

Российские авторы А.Н. Каркищенко и А.В. Гончаров в цикле работ 2008–2011 годов [13–15] дали математическую проработку для метода описания изображений, названного ими «знаковым представлением изображения»: определены свойства знакового представления, введены меры информативности и неопределённости знаковых представлений, рассмотрены вопросы их устойчивости. Авторы различают *полное* и *оконное* знаковые представления: первое задаёт отношение на всех возможных парах отсчётов, в то время как второе – только на «близких» по расположению. Также авторы показали, что для конкретного знакового представления порождается целое множество эквивалентных изображений, приводящих к этому знаковому представлению. В итоге авторы связали конкретное знаковое представление с перестановкой отсчётов изображения, что позволило им разработать алгоритм реконструкции изображения, для которого это знаковое представление и указано. Следует отметить, что альтернативный эмпирический итеративный алгоритм ранее был указан О.В. Цветковым в работах

[6, 7] как, собственно и идея связать сигнал и его «знаковое представление». В зарубежной литературе «знаковое представление» обычно рассматривается в наиболее простом – оконном (3x3) – варианте, который получил название *локальных бинарных шаблонов* или *LBP* [12–16].

Несмотря на очевидную схожесть *знаковых представлений* компьютерного зрения и метода *парных сравнений* теории полезности и принятии решений, попыток связать и рассматривать совместно эти два направления, по информации автора, не предпринималось. В настоящей работе исследуется задача реконструкции (для задач принятия решений – функции полезности/предпочтений, для компьютерного зрения – цифрового изображения) по данным парных наблюдений.

1.3 Метод реконструкции предпочтений по знаковым представлениям

1.3.1 Общее описание метода

Предлагаемый метод базируется на ряде предположений. Во-первых, исходная внутренняя модель предпочтений нам не известна. При этом попытка реконструировать эту модель при малом количестве информации увенчаться успехом не может. Таким образом, при малом числе парных сравнений следует ориентироваться на простую модель, а по мере увеличения объёма доступной информации (числа пар сравнений) сложность модели следует увеличивать. Дополнительными аргументами в поддержку данного решения для систем оценки пользовательских предпочтений являются вопросы «холодного старта» системы и негативная оценка пользователями большого числа задаваемых им системой вопросов [25].

Во-вторых, заранее ориентироваться на линейную модель функции полезности/предпочтений оказывается чрезвычайно опасно. Это приводит нас к необходимости обеспечения возможности автоматического перехода к нелинейным моделям. Последнее, по убеждению автора работы, рационально сделать путём перехода в новое пространство признаков Y б*о*льшей размерности. В новом пространстве существенно б*о*льшей размерности в соответствие с известным утверждением [20] классы (то есть пары элементов с противоположным знаковым представлением) будут разделимы почти наверное.

В-третьих, в значительном числе задач реконструкция функции полезности не требуется, а требуется реконструкция только знакового представления (1.2). То есть задача реконструкции полезности как регрессионная задача может не решаться. Более того, как показано в работах А.Н. Каркищенко, А.В. Гончарова, О.В. Цветкова, при наличии знакового представления сама функция может быть также реконструирована в символьном или числовом виде. Таким образом, более сложная регрессионная задача может быть заменена на задачу классификации. При этом при использовании линейного классификатора оказывается возможным автоматически реконструировать и функцию полезности (см. ниже).

В результате предлагаемое решение можно представить в виде набора этапов:

1) нормализация диапазона признаков в [0,1];

2) выбор нового пространства описания (базиса анализа) Y;

3) перевод существующего описания – вектора $\mathbf{x}$ – в новое пространство признаков Y большей размерности $K=\dim(Y) \geq N$;

4) решение задачи построения линейного или нелинейного классификатора в новом пространстве Y, при этом автоматически выполняется оценка функции полезности (UF) для линейного классификатора и/или функции предпочтения (PF);

5) оценка качества построенного классификатора на тестовом множестве с использованием выражений ошибок (1.3) и (1.4);

6) если оценка удовлетворительная – выход из процедуры, в противном случае – переход к этапам 3 или 2 (если исчерпали допустимые размерности выбранного пространства).

Принципиальная схема предлагаемого подхода представлена на рисунок 1.3 и выделена пунктиром. В целом же схема, показанная на этом рисунке целиком, представляет собой принципиальную модель разработанного фреймворка (англ.: framework), который позволяет не только обеспечить исполнение указанных ранее этапов, но и провести объективную оценку предложенного подхода за счёт автоматического синтеза «неизвестных» функций полезности/предпочтений.

А именно, «идеальная» функция полезности/предпочтений задается в виде:

$$u^S(\mathbf{x}) = \sum_{k=0}^{K_S-1} w_k^S \cdot \varphi_k^S(\mathbf{x}),$$

где $\left(w_0^S, w_1^S, \ldots, w_{K_S-1}^S\right)^T$ – K_S-мерный вектор коэффициентов представления функции полезности в базисе синтеза

$$\left\{\varphi_k^S(\mathbf{x})\right\}_{k=0}^{K_S-1}.$$

И размерность K_S модели синтеза, и сам базис синтеза, и коэффициенты представления в этом базисе являются неизвестными, при этом коэффициенты выбираются случайным образом с единственным ограничением:

$$\left\|\left(w_0^S, w_1^S, \ldots, w_{K_S-1}^S\right)^T\right\| = 1.$$

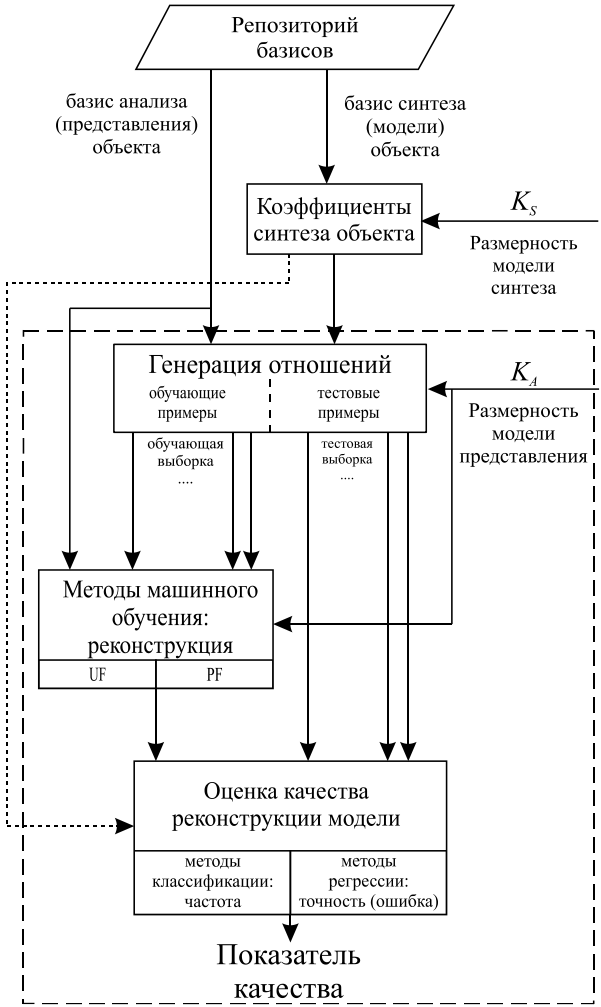


Рисунок 1.3 – Схема предлагаемого подхода

Вектор **x** в представленном выше выражении выступает в качестве первичного описания объектов/альтернатив. В случае анализа изображений вектор **x** – суть нормированные в интервал [0,1] координаты отсчетов изображения.

Искомое решение ищется в форме линейного или нелинейного классификатора над K_A значениями базисных функций анализа:

$$\left\{\varphi_k^A(\mathbf{x})\right\}_{k=0}^{K_A-1}.$$

В частности, в случае линейного классификатора имеем следующее представление реконструируемой функции полезности:

$$u(\mathbf{x})=\sum_{k=0}^{K_S-1} w_k^A\cdot\varphi_k^A(\mathbf{x}),$$

где $\left(w_0^A,w_1^A,\ldots,w_{K_A-1}^A\right)^T$ – K_A-мерный вектор коэффициентов представления функции полезности в базисе анализа, определяемый на основе обучения по результатам парных сравнений

$$u^S(\mathbf{x}_i)\underset{<}{\overset{>}{}}u^S(\mathbf{x}_j),\ \ (i,j)\in\Theta.$$

Блок синтеза предпочтений соответствует части схемы на рис. 2 с названиями «базис синтеза» и «коэффициенты синтеза». Везде ниже, в том числе в экспериментах, заглавный символ «*S*» соответствует этому блоку (блоку синтеза). Напротив, анализ и реконструкция функции полезности и функции предпочтения соответствует на схеме блоку «Методы машинного обучения», на который везде ниже (в том числе в экспериментах) ссылаемся, используя заглавный символ «*A*».

Ниже будет представлена некоторая детализация разработанного подхода в части состава репозитория базисов, классификаторов и способов обучения.

1.3.2 Репозиторий базисов

В работе используются три базиса для осуществления отображений φ^A и φ^S вида:

$$\varphi: X \to Y, \quad \mathbf{x} \mapsto \mathbf{y}(\mathbf{x}).$$

1) Базис исходного представления

$$K = \dim(Y) = \dim(X) = N;$$
$$y_n = \varphi_n(\mathbf{x}) = x_n, \quad n = \overline{0, N-1}.$$

2) Степенной (полиномиальный) базис

Многомерные базисные функции $\varphi_k(\mathbf{x})$ представляют собой произведение одномерных степенных функций координат исходного вектора (для упрощения изложения полагаем $\forall n \; K_n = K_0; \; 0 \le k_n < K_0$):

$$y_k = \varphi_k(\mathbf{x}) = \prod_{n=0}^{N-1} x_n^{k_n}, \quad k = \sum_{n=0}^{N-1} K_0^n k_n .$$

Здесь:

$$K = \prod_{n=0}^{N-1} K_n = K_0^N = \dim(Y) > \dim(X) = N.$$

2) *Фурье-базис (гармонический)*

Аналогично предшествующему случаю, многомерные базисные функции $\varphi_k(\mathbf{x})$ представляют собой произведение одномерных гармонических (косинусных) функций от координат исходного вектора (для упрощения изложения полагаем $\forall n \;\; K_n = K_0$):

$$y_k = \varphi_k(\mathbf{x}) = \prod_{n=0}^{N-1} \cos\left(\pi k_n x_n\right), \quad k = \sum_{n=0}^{N-1} K_0^n k_n .$$

$$K = \prod_{n=0}^{N-1} K_n = K_0^N = \dim(Y) > \dim(X) = N.$$

Нормировка признаков в диапазон [0,1] приводит к некоторому отличию от стандартной формы записи тригонометрического ряда Фурье, содержащей, наряду с константой, синусные $\sin(kx)$ и косинусные $\cos(kx)$ базисные функции. А именно:

а) исключение синусных функций объясняется переходом от полного [-π, π] интервала к рассмотрению только его правой части [0, π];

б) ограничение максимального значения признаков единицей влечет появление множителя «π» в аргументе тригонометрической функции.

1.3.3 Репозиторий методов классификации

В рамках предложенного подхода репозиторий методов машинного обучения (классификаторов) включает следующие:

а) логистическая регрессия (LR);

б) линейный классификатор Фишера;

в) метод опорных векторов с линейной разделяющей функций (без ядра);

г) метод опорных векторов с радиальной базисной функцией (SVM-RBF);

д) метод ближайшего соседа;

е) дерево решений;

ж) случайный лес (RF – Random Forest).

Более детальное описание указанных методов представлено пособиях [1–2], монографии [3] и последующих разделах настоящего пособия.

1.3.4 Особенности использования и обучения классификаторов в задаче реконструкции функции полезности и предпочтения

Для реализации процесса обучения и решения задачи реконструкции исходим из следующих положений:

1) поскольку первые три классификатора (а)-(в) являются линейными в новом пространстве признаков *Y*, это позволяет использовать их (то есть результат скалярного произведения вектора признаков **y** на вектор весов **w** классификатора) и для *реконструкции функции полезности* (или цифрового изображения – в задачах компьютерного зрения), то есть:

$$u(\mathbf{x})=\mathbf{w}^T\mathbf{y}(\mathbf{x}). \tag{1.8}$$

Для настройки соответствующего классификатора решается задача классификации вида ($\mathrm{I}(\ldots)$ – индикатор события):

$$\sum_{(j,i)\in\Theta}\mathrm{I}\left(\mathbf{w}^T\left(\mathbf{y}(\mathbf{x}_j)-\mathbf{y}(\mathbf{x}_i)\right)z_{ji}<0\right)\to\min_{\mathbf{w}}. \tag{1.9}$$

2) в случае использования классификаторов (г)-(ж) функция полезности напрямую реконструирована быть не может. Однако, учитывая неопределённость в факте отсутствия/присутствия линейной разделимости классов, исходная задача решается с использованием указанных классификаторов в двух постановках.

а) В качестве входного вектора выступает вектор разницы описаний

$$\mathbf{y}_{ji}\equiv\mathbf{y}(\mathbf{x}_j)-\mathbf{y}(\mathbf{x}_i)$$

и классификатор решает задачу (1.9) в виде:

$$\sum_{(j,i)\in\Theta}\mathrm{I}\left(\mathbf{w}^T\mathbf{y}_{ji}z_{ji}<0\right)\to\min_{\mathbf{w}}. \tag{1.10}$$

б) В качестве входного вектора выступает конкатенация векторов описаний, а именно:

$$\mathbf{y}_{ji} = \left(\mathbf{y}^T\left(\mathbf{x}_j\right), \mathbf{y}^T\left(\mathbf{x}_i\right)\right)^T$$

и классификатор решает задачу (1.10) с «расширенным» вектором весовых коэффициентов:

$$\dim\left(\mathbf{y}_{ji}\right) = \dim\left(\mathbf{y}\left(\mathbf{x}_j\right)\right) + \dim\left(\mathbf{y}\left(\mathbf{x}_i\right)\right).$$

3) Решения пользователей могут быть ошибочными. При этом чем меньше различие в полезностях альтернатив/объектов, тем выше вероятность ошибки. Поэтому в работе используется модель Тёрстоуна с оценкой вероятности (1.5) для внесения погрешностей в истинные предпочтения (1.2) следующим образом: при обучении верное значение знакового представления для конкретной пары $\left(\omega_j, \omega_i\right)$ используется с вероятностью (1.5), а ошибочное, то есть противоположное истинному по знаку, принимается с «противоположной» вероятностью. Указанная модификация для случая $\mu_j > \mu_i$ может быть формально представлена следующим образом (в выражении ξ – случайная величина, равномерно распределенная на отрезке $[0,1]$; z_{ji} – величина, используемая при обучении):

$$z_{ji} \Leftarrow \begin{cases} z\left(\omega_j, \omega_i\right), & \xi < P\left(u\left(\omega_j\right) - u\left(\omega_i\right) > 0\right); \\ -z\left(\omega_j, \omega_i\right), & \textit{иначе.} \end{cases}$$

Дисперсия функции полезности полагается одинаковой и равной наперёд определённой величине D_u. То есть в выражении (1.5): $\sigma_0^2 = \sigma_1^2 = D_u$, корреляция $\rho_{10} = 0$, откуда $\sigma_{10}^2 = 2D_u$.

4) Для устранения эффектов, связанных с «удачными» или, наоборот, «неудачными» распределениями обучающих и тестовых выборок, обучение и тестирование классификаторов при фиксированном наборе параметров метода производится независимо R раз. Результаты расчёта ошибок усредняются.

1.4 Экспериментальные исследования метода реконструкции предпочтений

1.4.1 Исследование эффективности реконструкции предпочтений на модельных данных

Цель экспериментов:

– подтвердить работоспособность предложенного подхода;

– определить наличие или отсутствие преимущества в использовании оценки полезности по сравнению с предпочтениями (сопоставить варианты 1 и 2);

– оценить информативность показателей качества, рассчитанных по обучающей выборке;

– выявить эффективные классификаторы, как приводящие к линейной модели, так и непараметрические;

– исследовать зависимость точности реконструкции функций полезности/предпочтений (ошибок реконструкции в форме (1.3) и (1.4)) от числа T использованных парных сравнений (показатель s в выражении ошибки (1.3) во всех экспериментах полагался $s=2$);

– исследовать влияние дисперсии функции полезности D_u на ошибки (1.3) и (1.4) реконструкции;

– сравнить эффективность различных базисов.

Для достижения указанных целей были проведены эксперименты со следующим набором параметров фреймворка (базисы и алгоритмы классификации были представлены во втором разделе):

– K_S = 3, 8, 15, 35;
– K_A = 3, 8, 15, 35, 63, 99;
– T = 500, 1000, 5000, 10000, 20000, 50000;
– D_u = 0 или 0,01;
– R = 100.

Поскольку объём настоящего пособия не позволяет представить здесь все численные результаты (они занимают несколько страниц), наиболее наглядные и содержательные фрагменты представлены на отдельных таблицах ниже. Необходимость подобного «массированного» исследования косвенно подтверждается наличием аналогичных работ [31]. Качественные выводы следующие:

– преимуществ в использовании оценки полезности по сравнению с предпочтениями (и наоборот) не выявлено;

– ошибки (1.4), рассчитанные по обучающей выборке, могут оказаться сколь угодно малыми при чрезвычайно высоких значениях ошибок на тестовой выборке;

– качественные показатели линейных классификаторов сопоставимы, однако *логистическая регрессия* имеет значительное преимущество по скорости и устойчивости (для больших величин T);

– *случайный лес* даёт лучшие качественные показатели среди нелинейных классификаторов;

– базис Фурье показал лучшие качественные результаты, чем степенной (см. таблицу 1.1);

– ошибка реконструкции функций полезности/предпочтений от числа использованных парных сравнений ожидаемо снижается и ограничена описательной способностью K_A базисных функций (см. таблицу 1.2);

– при большом T влияние дисперсии функции полезности D_u на ошибки реконструкции полезности/предпочтений в достаточно широком диапазоне D_u ограничено (см. таблицу 1.3). А

именно, при возрастании дисперсии в 100 раз (с 0,0025 до 0,25) ошибки возросли в три и семь раз, соответственно.

Таблица 1.1. **Результаты сравнения Фурье и степенного базисов (LR –логистическая регрессия, RF – случайный лес)**

K_S	K_A	T	Ошибка (1.4)			
			A: Фурье S: полином		A: полином S: Фурье	
			LR	RF	LR	RF
35	35	10000	0,2864	0,1828	0,0076	0,0145
35	35	50000	0,3341	0,1237	0,0067	0,0083
35	63	10000	0,3059	0,1945	0,0062	0,0141
35	63	50000	0,2633	0,1223	0,0049	0,0084

Таблица 1.2. **Зависимость ошибки (1.4) реконструкции функции полезности от числа использованных парных сравнений**

T	500	1000	5000	10000	20000	50000
K_S=35 K_A=15	0,012	0,013	0,0128	0,011	0,0111	0,0092
K_S=35 K_A =35	0,03	0,019	0,0048	0,0076	0,00715	0,00668
K_S=35 K_A =63	0,058	0,014	0,006	0,0062	0,00495	0,00495

Таблица 1.3. **Зависимость ошибки (4) реконструкции функции полезности от её дисперсии** D_u

D_u	Ошибка (1.4)	
	LR	RF
0	0,0043	0,012
0,0025	0,00945	0,01705
0,01	0,0144	0,0287
0,04	0,0135	0,05085
0,09	0,02195	0,06475
0,16	0,02905	0,09345
0,25	0,029	0,1228
0,36	0,03255	0,13755

1.4.2 Исследование эффективности реконструкции предпочтений на примере реконструкции цифрового изображения по его разреженному знаковому представлению

Цель экспериментов:

– подтвердить возможность реконструкции изображения по разреженному знаковому представлению;

– исследовать зависимость погрешности (1.3) реконструкции цифрового изображения от числа использованных парных сравнений (в процентном отношении от числа отношений в полном знаковом представлении, см. таблицу 1.4);

– исследовать зависимость погрешности реконструкции цифрового изображения от дисперсии D_u.

В целом, *предложенный подход продемонстрировал как работоспособность, так и эффективность* в реконструкции как функции полезности, там и функции предпочтения.

Таблица 1.4. **Зависимость ошибок (1.3) и (1.4) реконструкции цифрового изображения от числа парных сравнений (part- в процентном отношении от числа отношений в полном знаковом представлении)**

K_A	T	part, %	$\boldsymbol{\varepsilon^2}$	$\tilde{d}$ обуч.	$\tilde{d}$ тест
24	1000	0,8	**0,685**	0,242	**0,262**
24	2000	1,6	**0,663**	0,249	**0,256**
24	5000	3,9	**0,653**	0,250	**0,254**
24	10000	7,9	**0,650**	0,253	**0,256**
24	50000	39,4	**0,648**	0,253	**0,253**
99	*1000*	*0,8*	***0,177***	*0,058*	***0,112***
99	*2000*	*1,6*	***0,150***	*0,074*	***0,100***
99	*5000*	*3,9*	***0,131***	*0,083*	***0,094***
99	*10000*	*7,9*	***0,131***	*0,084*	***0,088***
99	*50000*	*39,4*	***0,125***	*0,086*	***0,087***
224	1000	0,8	**0,173**	0,002	**0,114**

Окончание табл. 1.4

224	2000	1,6	**0,126**	0,020	**0,083**
224	5000	3,9	**0,081**	0,041	**0,063**
224	10000	7,9	**0,070**	0,047	**0,059**
224	50000	39,4	**0,062**	0,052	**0,054**

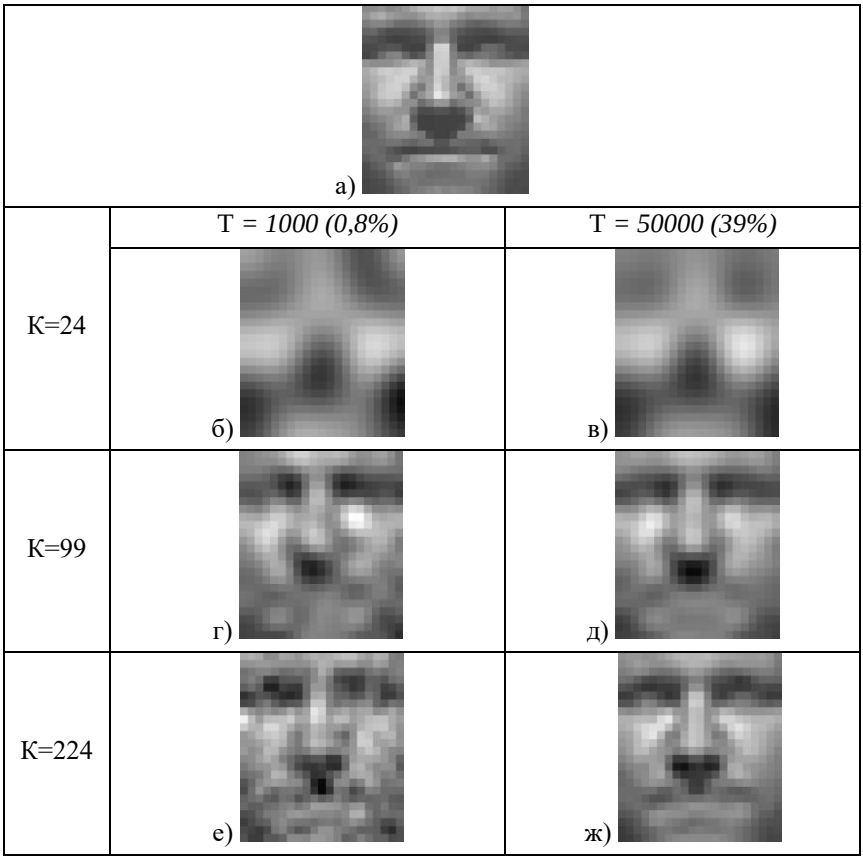


Рисунок 1.4 – Исходное (а) и реконструированные (б)-(ж) изображения, размер оригинала 24x21 отсчётов

Выводы настоящего раздела в целом повторяют выводы раздела 1.4.1. В частности, ошибка реконструкции цифрового изоб-

ражения от числа использованных парных сравнений ожидаемо снижается и ограничена описательной способностью базиса: при малом количестве базисных функций (24 и 99) снижение ошибки при росте T не происходит. На рисунке 1.4 продемонстрированы реконструированные изображения при различном числе базисных функций. В целом, *предложенный подход продемонстрировал возможность реконструкции изображения по разреженному знаковому представлению*.

1.5 Результаты первого раздела

В данном разделе представлена классификация методов машинного обучения. В соответствие с ней традиционными задачами машинного обучения являются следующие:

- упорядочивания меток,
- упорядочивания примеров и
- упорядочивание объектов.

Рассматривается решение последней из задач в постановке, когда информация об объектах представлена в виде парных сравнений. В такой постановке основной задачей является реконструкция функции предпочтения, в частном случае эта задача может быть сведена к задаче реконструкции функции полезности. Последующее изложение настоящего раздела содержит описание подхода к реконструкции функций и цифровых изображений, заданных неявно результатами парных сравнений значений функции для двух случайных аргументов. Он основан на переходе в пространство высокой размерности с последующей классификацией прецедентов-сравнений. Показано, что указанный подход позволяет эффективно решать как задачи оценки функции полезности и/или предпочтения в теории принятия решений, так и задачу реконструкции цифрового изображения по его разреженному знаковому представлению.

2 РАСПОЗНАВАНИЕ ОБРАЗОВ. МЕТОДЫ КЛАССИФИКАЦИИ

2.1 Формальная постановка задачи распознавания образов

Пусть задано некоторое множество из I подлежащих распознаванию *объектов*:

$$\Omega=\{\omega_0,\omega_1,\ldots,\omega_{I-1}\},$$

и задано его разбиение на L непересекающихся подмножеств, называемых в дальнейшем *образами* или *классами*:

$$P_\Omega=\{\Omega_0,\Omega_1,\ldots,\Omega_{L-1}\},\ \bigcup_{l=0}^{L-1}\Omega_l=\Omega.$$

Пусть каждый из объектов $\omega\in\Omega$ представляется набором числовых характеристик, называемым *вектором признаков:*

$$\overline{x}=\left(x_0,x_1,\ldots,x_{N-1}\right)^T.$$

Задача классификации заключается в отыскании решающего правила, которое по заданному вектору признаков $\overline{x}(\omega)$ указывает, какому классу Ω_l принадлежит соответствующий объект ω. Построение такого решающего правила эквивалентно разбиению *метрического пространства признаков* $D=\{\overline{x}:\overline{x}\in D\}$ на множество непересекающихся областей:

$$P_D=\{D_0,D_1,\ldots,D_{L-1}\},\ \bigcup_{l=0}^{L-1}D_l=D. \qquad (2.1)$$

При этом решение о принадлежности некоторого объекта $\omega \in \Omega$ к классу Ω_l принимается в том случае, если соответствующий объекту вектор признаков $\bar{x}(\omega) \in D$ принадлежит области D_l.

Решающее правило, предназначенное для указания, какой области D_l признакового пространства D принадлежит предъявленный вектор признаков $\bar{x}$, называется классификатором.

Таким образом, под классификатором далее мы понимают следующее отображение:

$$\Phi: D \to \{0, \ldots, L-1\}. \qquad (2.2)$$

В идеале классификатор должен быть таким, чтобы области, выделяемые в пространстве признаков, соответствовали классам, то есть в идеале для элементов множеств P_Ω и P_D должно выполняться следующее условие: объект ω принадлежит классу Ω_l тогда и только тогда, когда соответствующий объекту вектор признаков $\bar{x}(\omega)$ принадлежит области D_l:

$$\forall \omega \in \Omega: \ \omega \in \Omega_l \Leftrightarrow \bar{x}(\omega) \in D_l. \qquad (2.3)$$

Как правило, на практике данное условие не выполняется, и существует вероятность неверно проклассифицировать объект или допустить ошибку при распознавании.

2.2 Математические подходы к построению систем распознавания

Существует много способов реализовать отображение (2.2). В зависимости от того, какая математическая абстракция используется, мы получаем различные математические подход к построению систем распознавания. Заметим, что многие идентичные ре-

зультаты (алгоритмы) распознавания могут быть получены разными способами.

Геометрический подход

Подход основан на понятиях разделения. Обычно задается вид разделяющей границы и некоторым образом определяются ее параметры. Основное отличие – способы определения параметров. В одном из наиболее известных подходов, предложенных В.Н. Вапником и названным им методом опорных векторов (SVM – Support Vectors Machine), линейная разделяющая граница строится так, чтобы «зазоры» между ней и объектами обучающего множествами были максимальны [32–33]. Существуют и другие способы построения разделяющей границы [1–3].

Статистический / вероятностный

Подход основан на аппарате проверки гипотез. Фактически класс ассоциирован с гипотезой и процесс классификации – это процесс принятия или отклонения гипотезы. Упрощенно суть статистического подхода проста – мы относим объект, заданный своим вектором признаков, в тот класс, для которого получение этого вектора более вероятно (рисунок). Последовательные методы классификации здесь также присутствуют как развитие теории последовательных методов принятия решений.

Лингвистический / синтаксический / структурный

Класс ассоциируется с грамматикой (правилом вывода). Грамматика может быть детерминированная (переходы однозначны) или стохастическая (каждому переходу приписывается вероятность). Объект, составленный из непроизводных элементов, проверяется на соответствие каждой из грамматик. Та, которой он удовлетворяет (с наибольшей вероятностью – для статистических грамматик) и определяет его класс. В детерминированном случае

класс соответствует той грамматике, с помощью правил вывода которой может быть получен классифицируемый объект.

Синтаксические методы дают мощный аппарат, применимый даже для изображений. Основная их проблема – это как из реального изображения получить непроизводные элементы. На практике это очень сложно.

Логический подход

Был прототипом алгебраического подхода (см. ниже). Вроде как устарел, не развивается и в ключевых работах не рассматривается. Не рассматриваем.

Алгебраический подход

Это подход, созданный и развиваемый российской школой распознавания – школой академика Ю.И. Журавлева. В рамках алгебраического подхода сформировалось два направления.

Первое – это класс алгоритмов вычисления оценок (АВО). Суть этого подхода – это работа с некоторой моделью алгоритма, в рамки которой укладываются (точно или приближенно) большинство существующих алгоритмов. Этот класс алгоритмов определен с точностью до функциональных и числовых параметров. Фиксирую функциональные параметры мы приходим к подклассу АВО, а фиксация числовых дает нам конкретный алгоритм. При разработке «оптимального» способа нахождения числовых параметров в некотором подклассе АВО можно говорить о синтезе оптимального алгоритма в соответствующем подклассе. Это и есть одна из основных задач в рамках первого направления алгебраического подхода.

Второе направление – это алгебра алгоритмов. Суть подхода – найти способ использования произвольного множества алгоритмов (в терминах подхода – некорректных) для построения корректного алгоритма. Под корректным алгоритмом понимается алгоритм,

который работает верно на заранее заданном списке примеров (объектов). Идея построений заключается в ассоциировании каждого некорректного алгоритма с матрицей ответов этого алгоритма на примерах и построении агрегирующего функционала (линейного – линейное замыкание множества алгоритмов, полиномиальное – алгебраическое замыкание множества алгоритмов), в результате применения которого к матрицам можно получить требуемую матрицу ответов. Агрегирующий функционал с исходным множеством алгоритмов и будет новым алгоритмом, а идейный подход назван автором алгеброй над алгоритмами.

Модели биологических систем

Исторически был одним из первых подходов. Была попытка имитировать деятельность мозга, разработана для этого математическая модель – искусственная нейронная сеть (ИНС). ИНС – это суперпозиция линейных комбинаций достаточно простых элементов – нейронов. Нейрон – это функция от вектора входов. Состоит из линейной функции от вектора и нелинейного скалярного преобразования.

Нечеткие множества

Не укладываются в «четкую» постановку задачи классификации. Идейно похожи на вероятностный подход.

В настоящем разделе будут рассмотрены большинство из указанных подходов и наиболее известных алгоритмов. Более детальное их рассмотрение можно найти в пособиях [1–2] и ставших классическими монографиях [34–45].

2.3 Оценка качества классификации на практике. Кросс-валидация

2.3.1 Показатели качества классификаторов

Вопрос качества классификатора возникает вне зависимости от того, какой именно подход мы используем. В статистическом подходе, как показано в пособиях [1–2] и далее в третьем разделе, качество классификатора оценивается вероятностями ошибок. Для задачи бинарной классификации (два класса) традиционными показателями качества выступают ошибки первого и второго рода, а именно:

$$p_0 = P\left(\bar{X} \in D_1 / \Omega_0\right), \quad p_1 = P\left(\bar{X} \in D_0 / \Omega_1\right).$$

Однако в последнее время эти классические показателя достаточно часто заменяются показателями качества информационного поиска, то есть величинами *точности* (англ. *Precision*) и *полноты* (англ.: Recall). Причинами являются, в частности, сложности с определением класса «необъекта» в задачах обнаружения объектов на изображении, в задачах информационного поиска [76]. Для расшифровки указанных понятий используем иллюстрацию, представленную на рисунке 2.1.

Как показано на данном рисунке, в задаче присутствуют *примеры* двух классов: закрашенные кружки соответствуют *объектам требуемого класса*, а полые – остальным (могут представлять фон, шум или объекты других классов). Классификатор, относя примеры к требуемому классу объектов (англ.: positives) может сделать это верно (англ.: true positives – TP) или ошибиться (англ.: false positives – FP). Аналогично, классифицируя примеры как не относящиеся к объектам требуемого класса (англ.: negatives), классификатор может сделать это верно (англ.: true negatives – TN) или

ошибиться, если примеры на самом деле были объектами требуемого класса (англ.: false negatives – FN).

Используя приведенную нотацию классические вероятности ошибок первого и второго рода запишутся в виде:

$$p_0 = \frac{FP}{TN + FP}, \quad p_1 = \frac{FN}{TP + FN}.$$

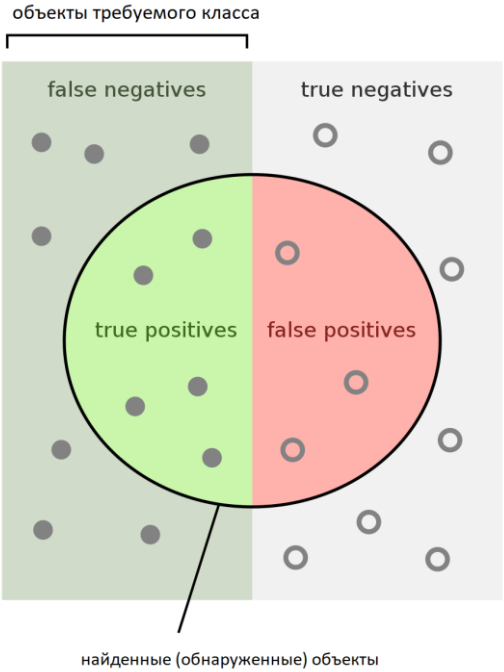


Рисунок 2.1 – Иллюстрация к понятиям точности и полноты

Как уже упоминалось, не всегда есть четкое представление о примерах, которые объектами не являются, то есть в некоторых задачах затруднительно корректно определить величину TN. Поэтому вместо указанных, классических, выражений используют

понятия точности и полноты, где «нормировка» производится по отношению к числу примеров, которые классификатор отнес к классу объектов. А именно:

$$precision = \frac{TP}{TP + FP}, \quad recall = \frac{TP}{TP + FN}.$$

Из представленных выражений очевидно, что величина полноты и ошибка второго рода (потери объекта) связаны напрямую:

$$recall = \frac{TP + FN - FN}{TP + FN} = 1 - \frac{FN}{TP + FN} = 1 - p_1 = p_{11},$$

То есть, фактически, *полнота есть вероятность верной классификации объектов искомого класса*. Для величины точности зависимость не столь очевидна, ее можно получить, записав выражение через вероятность:

$$precision = P\left(\Omega_1 / \bar{X} \in D_1\right) = \frac{P\left(\Omega_1 \cdot \left(\bar{X} \in D_1\right)\right)}{P\left(\bar{X} \in D_1\right)} =$$

$$= \frac{P(\Omega_1) P\left(\bar{X} \in D_1 / \Omega_1\right)}{P(\Omega_0) P\left(\bar{X} \in D_1 / \Omega_0\right) + P(\Omega_1) P\left(\bar{X} \in D_1 / \Omega_1\right)} =$$

$$= \frac{P\left(\bar{X} \in D_1 / \Omega_1\right)}{\frac{P(\Omega_0)}{P(\Omega_1)} P\left(\bar{X} \in D_1 / \Omega_0\right) + P\left(\bar{X} \in D_1 / \Omega_1\right)} = \frac{p_{11}}{p_{11} + p_{01} \frac{P(\Omega_0)}{P(\Omega_1)}}.$$

Окончательно:

$$precision = \frac{1 - p_1}{1 - p_1 + p_0 \frac{P(\Omega_0)}{P(\Omega_1)}}.$$

Дополнительной *характеристикой классификатора* является т.н. ROC-кривая (англ. receiver operating characteristic). ROC-

кривая – это график, отображающий изменение величины TPR (True Positive Rate) от FPR (False Positive Rate), где:

$$FPR \equiv \frac{FP}{FP+TN} = p_0, \quad TPR \equiv \frac{TP}{TP+FN} = recall = 1 - p_1 .$$

То есть фактически график показывает зависимость изменения вероятности верной классификации объектов заданного класса от вероятности ложного обнаружения. График имеет возрастающий характер от значения (0;0) до значения (1;1). Как уже отмечалось, график характеризует классификатор в целом, причем чем «выше» над диагональю проходит ROC-кривая – тем лучше классификатор. Для получения интегральной численной оценки классификатора использует площадь под ROC-кривой. Эту величину называют AUC (англ. Area Under Curve, площадь под кривой). Очевидно, что чем выше показатель AUC, тем качественнее классификатор. Значение 0,5 демонстрирует фактическую непригодность выбранного метода классификации и эквивалентно случайному угадыванию ответа. При этом, если AUC<0,5, то классификатор действует «с точностью до наоборот»: необходимо просто положительные ответы принять за отрицательные и наоборот.

2.3.2 Кросс-валидация и скользящий контроль

Хотя приведенные выше величины и могут напрямую использоваться для оценки качества классификаторов, получаемые с их помощью числовые значения могут быть достаточно далеки от того, что классификатор будет демонстрировать на практике. На первый взгляд, такое рассогласование может быть вызвано недостаточной «представительностью» обучающего множества, на котором проводится обучение и оценка соответствующих величин качества. Чтобы обойти указанную проблему, выборку

стали делить на *обучающую* и *контрольную(валидационную)*. По данным первой выборки производилась настройка классификатора, а вторая выборка использовалась для расчета итоговых значений качества полученного классификатора. Однако, достаточно быстро стало понятно, что такое деление обладает также недостатками – мы можем поделить выборку «удачно» и «неудачно», так что гарантировать похожесть результатов работы на практике на результаты работы на контрольной выборке не удается. Для преодоления этой проблемы для оценки качества классификации стали использовать процедуру *кросс-валидации* (англ.: cross-validation), *кросс-проверки* или *скользящего контроля*. Суть указанной процедуры – многократное деление всей выборки на различные пары обучающей и контрольной выборки. Качеством же классификатора принимается некоторая агрегированная характеристика (например, среднее) результатов классификации на множестве контрольных выборок. При делении выборки иногда накладывают какие-либо дополнительные требования, например, стратификации. *Стратификация* – способ деления выборки в процедуре кросс-валидации, при котором пропорция деления на обучающую и контрольную остается построенной. Данный метод позволяет снизить дисперсии получаемых оценок. В действительности, обозначенная выше проблема несоответствия результатов классификации на обучающей и контрольной выборках имеет более глубокие корни, которые в статистической теории обучения принято именовать как проблема переобучения и связывать с обобщающей способностью используемого алгоритма классификации. Данный вопрос кратко будет рассмотрен в подразделе 2.4 ниже.

Ниже представлены несколько наиболее используемых вариантов процедуры кросс-валидации. В изложении ниже N – общий объем выборки, p – число объектов в контрольной выборке.

Полный скользящий контроль (Exhaustive cross-validation)

Leave-p-out cross validation (LpO CV)

Данная процедура предполагает формирование всех возможных разбиений N объектов на выборки: контрольной длины p и обучающей длины N-p. Очевидно, что общее количество разбиений в этом случае оказывается огромным: C_N^p, поэтому на практике такой способ оценки качества классификатор встречается крайне редко. Однако, частный случай с p=1 является достаточно популярным (см. ниже).

Leave-1-out cross validation (LpO CV)

Этот способ является частным случаем предшествующего с p=1, что приводит нас к необходимости $C_N^1 = N$ разрешать проблему построения классификатора по N-1 объекту обучающей выборки, оценивая качество всего по одному контрольному объекту.

K-fold CV (контроль по Q блокам)

Выборка разбивается случайным образом на K блоков одинаковой длины, так что $N = pK$. Каждый из K блоков длиной p поочередно становится контрольным множеством, в то время как объекты оставшихся блоков в количестве

$$p(K-1)=N-p$$

формируют обучающее множество. На практике данный подход используется достаточно часто с числом K~10-20.

RxK-fold CV (контроль по RxK блокам)

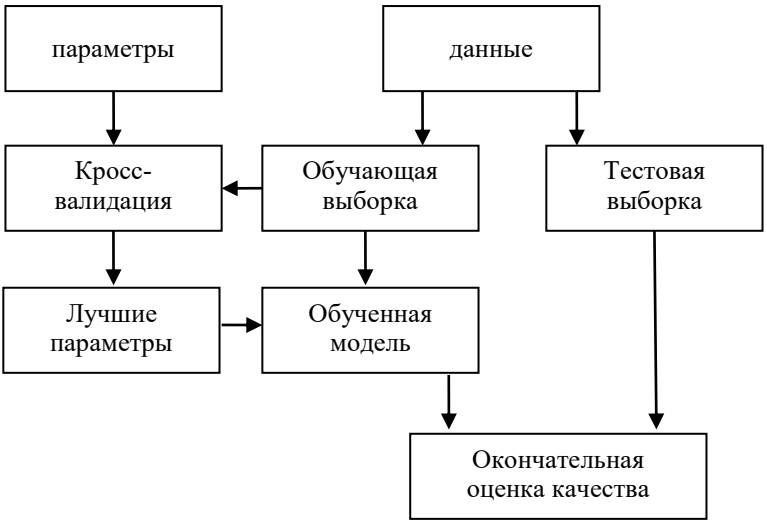
Отличается от предшествующего варианта тем, что случайное разбиение на K блоков повторяется R раз. Данный вариант со стратификацией классов часто рассматривается как де-факто стандарт.

Общая схема использования кросс-валидации при обучении классификатора

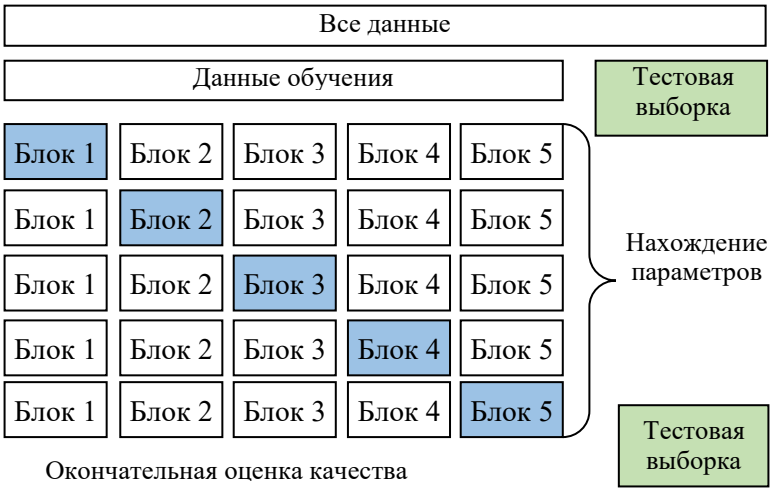
Процедура кросс-валидации может быть успешно применена для определения оптимальных *настраиваемых* параметров классификатора так, как показано на рисунках 2.2. Однако, сам факт использования кросс-валидации для настройки параметров классификатора делает оценки качества классификатора, полученные с ее помощью, *смещенными*. Для исправления указанной ситуации на практике в дополнении к обучающей(им) и контрольной(ым) (валидационной) выборкам из доступных данных заблаговременно формируют еще и тестовую выборку (англ.: test data), которая используется для получения несмещенной оценки качества классификатора (типовым является выделение 20% доступных данных на тестовое множество).

2.3.3 Выбор модели классификатора

Все указанное выше справедливо до тех пор, пока у нас фиксирована *модель классификатора* и/или его *гиперпараметры*. К ним могут относится и степень полинома дескриминантной функции, число слоев ИНС, число соседей в классификаторе по ближайшему соседу, параметры метода опорных векторов, ядра в этом методе, регуляризирующие параметры и т.п. То есть все те величины, которые при обучении классификатора на обучающем множестве не настраиваются. Для выбора оптимальных или квазиоптимальных значений таких параметров и моделей следует сопоставить результаты, полученные в прошлом разделе с помощью кросс-валидации или на тестовой выборке. То есть сравнение моделей можно производить по тем же агрегированным валидационным оценкам или на тестовой выборке. Дополнительные комментарии по способу перебора модели представлены в подразделе 2.4 в пункте «Метод упорядоченной минимизации риска».



а) общая схема метода;



б) вариант K-fold CV (K=5);

Рисунок 2.2 – Иллюстрация к использованию
процедуры кросс-валидации

Комментарий. Ряд работ в теории машинного обучения развивают идею выбора классификатора (в т.ч., модели) на основании принципа *минимальности длинны описания*. Минимальная длина описания (англ. minimum description length, MDL) – это принцип выбора модели, при котором кратчайшее описание данных является наилучшей моделью.

MDL берет свое начало в теории информации, в частности, в работах А. Колмогорова: «В алгоритмической теории информации сложность по Колмогорову объекта… представляет собой длину кратчайшей компьютерной программы (на заранее определенном языке программирования), которая выдает объект в качестве выходных данных. Это показатель вычислительных ресурсов, необходимых для определения объекта, и он также известен как алгоритмическая сложность, сложность Соломонова–Колмогорова–Чайтина, сложность размера программы, описательная сложность или алгоритмическая энтропия. Она названа в честь Андрея Колмогорова, который впервые опубликовал статью на эту тему в 1963 году [47-48] и является обобщением классической теории информации»[1]. Применительно к задачам обучения одной из первых работ в контексте MDL считается работа Йормы Риссанена [49]. В соответствие с принципом MDL выбор модели и ее настройка есть решение следующей оптимизационной задачи:

$$q^* = \arg\min_{q \in Q} \left(L(q) + L(D|q) \right),$$

здесь D – представляемые данные (выборка), q – конкретная модель/стратегия распознавания, Q – множество моделей/гипотез, $L(\ldots)$ – функция длинны описания.

2.4 Обобщающая способность методов распознавания

[1] https://en.wikipedia.org/wiki/Kolmogorov_complexity

Несмотря на кажущуюся простоту постановки задачи распознавания, ее решение на практике не является простым. Среди ученых, работающих в этом направлении, давно известна следующая байка о построении «идеального» алгоритма распознавания.

Пусть у нас есть размеченная выборка объектов/примеров (обучающая). Давайте запомним их все и при поступлении объектов будем классифицировать в соответствии с правилом: если поступивший объект совпадает с объектом из обучающей выборки, то классифицируем его в тот класс, который указан в разметке. Если же приходит объект, который в обучающей выборке не присутствовал – относим его к произвольному классу. Все претензии относительно недостаточного качества классификации (например, на контрольной или тестовой выборках) можно относить к недостаточной «репрезентативности» выборки.

Даже для неспециалиста очевидно, что следование такой стратегии решением не является. В то же время на практике достаточно часто возникают аналогичные ситуации, которые имеют ту же природу, как и описанная выше – для примера см. подраздел 3.2.3 и пункт «Древовидные решающие правила». Суть же проблемы заключается в т.н. *эффекте переобучения*, то есть в чрезмерной подстройке классификатора под элементы выборки – классификатор находит не закономерности распределений, а настраивается на конкретную выборку. В 70х годах советские ученые В.Н. Вапник и А.Я. Червоненкис [32, 33] представили стогую математическую теорию, которая устанавливает связь между основными характеристиками задачи распознавания (размером выборки N, точностью ε и надежностью η решающего правила для задачи бинарной классификации) и *емкостью множества стратегий распознавания*, названной впоследствии по фамилиям авторов как *VC-размерность*.

Понятие *емкости множества стратегий* вводится следующим образом. Пусть Q есть множество отображений (стратегий) вида

$$q:\bar{x}\to\{0,1\},$$

где «0» и «1» определяют метки соответствующих классов. Величина r является емкостью указанного множества стратегий, если:

– существует такое множество $\{\bar{x}_j\}_{j=1}^{r}$ точек объема r, что при любой возможной их классификации (очевидно, максимальное число возможных бинарных классификаций для набора из r точки равно 2^r) в Q найдется стратегия q, которая реализует (повторяет) эту классификацию;

– для любого множества $\{\bar{x}_j\}_{j=1}^{r+1}$ точек объема r+1 существует такая классификация, которая не может быть реализуема (повторена) в раках стратегий из Q.[1]

В теории Вапника-Червоненкиса понятие емкости множества стратегий используется для последующего определения границ для т.н. *функции роста*, определение которой в работе [32-33] сделано следующим образом. Обозначим величиной $\Delta^Q(\bar{x}_1,\ldots,\bar{x}_N)$ количество возможных классификаций множества точек $\{\bar{x}_j\}_{j=1}^{N}$ стратегиями из Q (число ограничено сверху значением равно 2^N поскольку не все варианты классификации/разбиения могут быть реализованы в рамках стратегий из Q). Тогда под функцией роста будем понимать следующую величину:

$$m^Q(N)=\max_{\bar{x}_1,\ldots,\bar{x}_N}\Delta^Q(\bar{x}_1,\ldots,\bar{x}_N).$$

[1] Емкость линейной стратегии классификации в n-ом пространстве равна n+1.

Тогда справедлива следующая теорема.

Теорема [32, стр. 102].

$$\eta \equiv P\left\{\sup_{q\in Q}\left|P(q)-\nu(q)\right|>\varepsilon\right\}<3m^{Q}(2N)e^{-\frac{N-1}{4}\varepsilon^{2}}. \qquad (2.4)$$

В указанном выражении $P(q)$ – истинная вероятность решения задачи бинарной классификации с использованием алгоритма q, а $\nu(q)$ – оценка этой вероятности по выборке длиной N. Величина η характеризует надежность получаемой оценки равномерной сходимости. Функцию роста в общем случае достаточно сложно оценить на практике, поэтом авторы подхода нашли решение, использовав емкость множества стратегий.

Теорема [32, стр. 102]. Если r- емкость множества стратегий Q, то

$$m^{Q}(N)\le\frac{1.5N^{r}}{r!}. \qquad (2.5)$$

Для стратегий бесконечной емкости, очевидно, $m^{Q}(N)=2^{N}$.

Подставляя (2.5) в (2.4) имеем, окончательно:

$$\eta<4.5\frac{(2N)^{r}}{r!}e^{-\frac{N-1}{4}\varepsilon^{2}}. \qquad (2.6)$$

Таким образом, последнее выражение гарантирует нам увеличение надежности классификации для случая (правая часть выражения стремится к нулю по мере роста N), когда емкость множества стратегий ограничена. В противном случае, правая часть выражения становится больше единицы и становится невозможным гарантировать сходимость эмпирической величины риска классификации к теоретической (см. пример в начале раздела).

Метод упорядоченной (структурной) минимизации риска

Оценки (2.6) являются чрезвычайно завышенными, чтобы быть конструктивно использованными на практике. Однако теория Вапника-Червоненкиса дает дополнительно ряд решений, которые говорят о том, как разумно может быть построен процесс обучения распознаванию. В частности, одним из частных результатов является следующий [32, стр. 207]:

$$P(q) < \nu(q) + 2\sqrt{\frac{1}{N}\left(r\left(\ln\frac{2N}{r}+1\right)-\frac{\ln\eta}{9}\right)}. \qquad (2.7)$$

Анализ правой части показывает, что величина $\nu(q)$ эмпирического риска с ростом мощности множества Q будет уменьшаться, в то время как второе слагаемое – расти. На основании этого факта В.Н. Вапником и А.Я. Червоненкисом был предложен *метод упорядоченной минимизации риска* [32, 33], который позднее был переименован в *метод структурной минимизации риска*.

Суть метода упорядоченной минимизации риска заключается в построении последовательности все более емкого множества стратегий из исходного полного их множества:

$$Q_0 \subset Q_1 \subset Q_2 \subset ... \subset Q_{R-1} = Q.$$

Примером такого построения являются полиномиальные решающие функции с увеличением степени полинома, или дерево решений с постоянно возрастающим числом терминальных вершин и т.п. Задача обучения решается последовательно, от простых стратегий к сложным. В какой-то момент правая часть в выражении (2.7) должна начать возрастать, что является сигналом к прекращению усложнения стратегии (модели классификатора). Однако, на практике выражение (2.7) не используют из-за завышенности теоретических оценок, а используют оценки кросс-

валидации для нахождения точки остановка, как указано выше в п. 2.3.3.

Дополнительные комментарии

Основные положения статистической теории обучения Вапника-Червоненкиса были изложены в 1974–1978 гг. Определенная ревизия теории надёжности обучения по прецедентам, но с комбинаторной точки зрения, была предпринята в первом десятилетии после миллениума К.В. Воронцовым [51]. Принципиальным моментом явилось дополнение правой части соответствующего неравенства (2.4) дополнительным сомножителем, названным *локальным коэффициентом разнообразия метода обучения*, который дополнительно ограничивал мощность множества стратегий классификации. Несмотря на такую корректировку, снижающую требования к объемам обучающих множеств, эти требования все также оставались завышенными (по отношению к практике).

В настоящее время, как известно, достигнуты впечатляющие результаты использования на практике искусственных нейронных сетей, содержащих миллионы и миллиарды параметров. При этом объемы обучающих данных, используемые при их настройке, чрезвычайно далеки от тех величин, которые следует использовать в соответствие с теорией Вапника-Червоненкиса. Это позволяет утверждать об *определенном разрыве в практике и теории надежности обучения по прецедентам*.

2.5 Результаты второго раздела

Во втором разделе представлена одна из возможных формальных постановок задачи распознавания образов, опирающаяся на представлении распознаваемых объектов с использованием описаний/признаков как элементов некоторого метрического пространства и последующей их классификации, заключающейся в отнесе-

нии образов объектов (т.е. векторов признаков) к той или иной области в пространстве признаков, связанной с номером класса объекта. Дальнейшее изложение направлено на конкретизацию способа решения второй задачи, именуемой в машинном обучении как задача «упорядочивания меток», и традиционно именуемой в теории распознавания образов как задача *классификации*. Представлены основные математические подходы к решению задачи классификации, которые исторически возникали при ее решении: геометрический подход, основанный на принципе разделения; статистический/вероятностный подход; лингвистический/синтаксический подход; логический; алгебраический, подход на основе моделей биологических систем; подход на основе нечетких множеств.

Рассмотрены различные способы определения качества конструируемого классификатора, а также способы оценки показателей качества по данным на практике. Обозначены основные проблемы статистического обучения, возникающие при построении классификатора на практике, связанные с проблемами переобучения. Представлены ключевые положения теории Вапника-Червоненкиса, которые позволяют связать основные характеристики задачи обучения (объем выборки, точность обучения и его надежность) с характеристиками класса стратегий (емкостью класса стратегий, известной как их VC-размерность), используемых для построения классификатора. Конструктивным выводом теории является *метод структурной минимизации риска*, в соответствие с которым все множество стратегий следует представить в виде последовательно расширяющихся подмножеств, на которых и проводить обучение с оценкой качества методом скользящего контроля. Подмножество, на котором достигается минимум соответствующей оценки и принимается как окончательное решение.

3 СТАТИСТИЧЕСКИЕ МЕТОДЫ КЛАССИФИКАЦИИ

3.1 Качество классификатора

Обозначим далее

$$p_{lj} \equiv = P\left(\bar{X} \in D_j \,/ \Omega_l\right) \quad \left(l, j = \overline{0, L-1}\right) \tag{3.1}$$

вероятность того, что классификатор принимает решение об отнесении вектора признаков некоторого объекта к области D_j, в то время как сам объект принадлежит классу Ω_l. При $l \neq j$ вероятности p_{lj} характеризуют ошибки распознавания и называются *вероятностями неверной* или *ошибочной классификации,* а вероятности p_{ll} определяют вероятности *верной (правильной) классификации* представителей соответствующего класса. Уменьшение вероятностей ошибочной классификации – это основная задача, которая возникает при построении классификатора.

Обычно классификатор задается не в виде областей признакового пространства (3.1), а в виде набора так называемых *дискриминантных* или *решающих функций* $d_l\left(\bar{x}(\omega)\right)$, $\left(l = \overline{0, L-1}\right)$. При этом процесс принятия решения осуществляется по следующему правилу: объект считается принадлежащим тому классу, дискриминантная функция которого для соответствующего вектора признаков является максимальной:

$$\forall\, j \neq l\colon \; d_l\left(\bar{x}(\omega)\right) \geq d_j\left(\bar{x}(\omega)\right) \Rightarrow \bar{x}(\omega) \in D_l\,. \tag{3.2}$$

Качество классификатора характеризуется величиной, называемой в теории статистических решений *условным средним риском.* Она задает среднюю величину потерь, связанных с принятием классификатором решения об отнесении данного вектора признаков $\bar{x}$ к классу с номером j:

$$R_j(\bar{x}) = \frac{1}{f(\bar{x})} \sum_{l=0}^{L-1} c_{lj} P(\Omega_l) f(\bar{x}/\Omega_l). \qquad (3.3)$$

В данном выражении:

– $P(\Omega_l)$ – *априорная вероятность* появления объектов из класса Ω_l;

– $f(\bar{x}/\Omega_l)$ – условная плотность вероятности случайного вектора признаков $\overline{X}$ для объектов класса Ω_l;

– $f(\bar{x})$ – безусловная плотность вероятности случайного вектора $\overline{X}$;

– элементы квадратной матрицы

$$C = \left\| c_{lj} \right\|_{l,j=0}^{L-1} \qquad (3.4)$$

характеризуют величины *штрафов* или *потерь* за ошибки классификатора. Матрица C может быть достаточно произвольной. Единственным ограничением на ее элементы является то, что штраф за ошибочное решение должен быть больше, чем штраф за решение правильное, то есть $c_{lj} > c_{ll}$.

Интегральной величиной, характеризующей качество классификатора, является *математическое ожидание потерь* или *общий риск,* который с учетом (3.3) и (3.1) имеет вид

$$R = \sum_{j=0}^{L-1} \int_{D_j} R_j(\bar{x}) f(\bar{x}) d\bar{x} = \sum_{j=0}^{L-1} \sum_{l=0}^{L-1} c_{lj} P(\Omega_l) p_{lj}. \qquad (3.5)$$

Процесс классификации аналогичен игре двух лиц с нулевой суммой, в которой одним из игроков является классификатор. В такой игре выигрыш (проигрыш) одного из участников равен проигрышу (выигрышу) другого. Выбор оптимальной стратегии в игре зависит от количества исходной информации. Могут использоваться *байесовская, минимаксная* стратегии или *стратегия Неймана-Пирсона*. В зависимости от того, какая из стратегий используется для построения классификатора, последний называют, соответственно, *байесовским классификатором, минимаксным классификатором* или *классификатором Неймана-Пирсона*.

3.2 Оптимальные стратегии классификации

3.2.1 Классификатор Байеса

Байесовская стратегия используется при наличии полной априорной информации о классах, то есть когда известны:

- функции правдоподобия для каждого из классов;
- матрица штрафов;
- априорные вероятности для каждого из классов.

Стратегия решения выбирается таким образом, чтобы обеспечить минимум общего риска (3.5). Минимальный общий риск при этом называется *байесовским риском*. В соответствии с выражениями (3.3) и (3.5) минимум общего риска R будет обеспечен, если разбиение пространства признаков D будет осуществляться по следующему правилу: вектор $\bar{x} \in D$ относится к области D_l только тогда, когда соответствующий условный средний риск $R_l(\bar{x})$ минимален:

$$\forall\, j \neq l \quad R_l(\bar{x}) < R_j(\bar{x}) \Rightarrow \bar{x} \in D_l\,. \qquad (3.6)$$

Если матрица потерь (3.4) является простейшей[1], то после подстановки в (3.6) выражения для условного среднего риска (3.3) имеем следующий явный вид байесовского классификатора (рисунок 3.1б):

$$\forall\, j \neq l \quad P(\Omega_l)\, f\left(\bar{x}/\Omega_l\right) \geq P\left(\Omega_j\right) f\left(\bar{x}/\Omega_j\right) \Rightarrow \bar{x} \in D_l\,. \qquad (3.7)$$

Из (3.7), в частности, видно, что решающими функциями байесовского классификатора являются функции

$$d_l(\bar{x}) = P(\Omega_l)\, f(\bar{x}/\Omega_l), \;\; l = \overline{0, L-1}. \qquad (3.8)$$

Часто используют также следующую форму записи байесовского классификатора:

$$\forall\, j \neq l \quad \frac{f(\bar{x}/\Omega_l)}{f(\bar{x}/\Omega_j)} \geq \frac{P(\Omega_j)}{P(\Omega_l)} \Rightarrow \bar{x} \in D_l\,. \qquad (3.9)$$

При этом функция $\Lambda_{lj}\left(\bar{x}\right) = \dfrac{f(\bar{x}/\Omega_l)}{f(\bar{x}/\Omega_j)}$ называется *отношением правдоподобия*, а величина $\lambda_{jl} = \dfrac{P(\Omega_j)}{P(\Omega_l)}$ – *пороговым значением*. Таким образом, байесовский классификатор основан на сравнении отношения правдоподобия с пороговым значением:

$$\forall\, j \neq l \quad \Lambda_{lj}(\bar{x}) \geq \lambda_{jl} \Rightarrow \bar{x} \in D_j\,,$$

и называется поэтому классификатором отношения правдоподобия.

---

[1] Матрица потерь C называется *простейшей*, если ее элементы $c_{ij} = \begin{cases} 0, & i = j \\ 1, & i \neq j \end{cases}$.

Легко показать, что при произвольном виде матрицы штрафов в случае двух классов байесовский классификатор имеет вид

$$\frac{f(\bar{x}/\Omega_1)}{f(\bar{x}/\Omega_0)} \begin{matrix} > \\ < \end{matrix} \frac{P(\Omega_0)(c_{01}-c_{00})}{P(\Omega_1)(c_{10}-c_{11})} \Rightarrow \bar{x} \in \begin{cases} D_1 \\ D_0 \end{cases}$$

с дискриминантными функциями:

$$d_j(\overline{X}) = P(\Omega_j)\left(c_{j(1-j)} - c_{jj}\right) f(\bar{x}/\Omega_j), \quad j = \overline{0,1}.$$

3.2.2 Минимаксный классификатор

Классификатор, основанный на минимаксной стратегии, используется для случая двух классов и если известны:

- функции правдоподобия для каждого из классов;
- матрица штрафов.

Минимизировать величину общего риска при отсутствии информации об априорных вероятностях классов, очевидно, невозможно. В то же время, предполагая возможность произвольного изменения значений априорных вероятностей классов, можно минимизировать максимально возможное значение риска. Действительно, общий риск (3.5) в случае двух классов может быть представлен в следующем виде:

$$\begin{aligned} R = &\left(c_{11} + p_{10}\left(c_{10} - c_{11}\right)\right) + P\left(\Omega_0\right) \cdot \\ &\cdot\left[\left(c_{00} + p_{01}\left(c_{01} - c_{00}\right)\right) - \left(c_{11} + p_{10}\left(c_{10} - c_{11}\right)\right)\right]. \end{aligned} \tag{3.10}$$

При фиксированном классификаторе изменение априорной вероятности приводит к изменению величины общего риска, причем характер зависимости в (3.10) линейный.

Поэтому поиск классификатора, минимизирующего максимально возможную величину общего риска, эквивалентен поиску такого байесовского классификатора, для которого величина (3.10)

является постоянной, не зависящей от значения априорной вероятности $P(\Omega_0)$ величиной. Таким классификатором, очевидно, является байесовский классификатор, удовлетворяющий следующему дополнительному условию:

$$\left(c_{00}+p_{01}\left(c_{01}-c_{00}\right)\right)-\left(c_{11}+p_{10}\left(c_{10}-c_{11}\right)\right)=0. \qquad (3.11)$$

Известно, что значение величины общего риска для минимаксного классификатора равно максимальному значению байесовского (минимального) риска. Пара априорных вероятностей $\left(P^*(\Omega_0),1-P^*(\Omega_0)\right)$, при которых байесовский риск принимает максимальное значение, называется *наименее благоприятным распределением априорных* вероятностей.

Таким образом, минимаксный классификатор – это байесовский классификатор, полученный для пары наименее благоприятных априорных вероятностей.

В более простой ситуации, когда элементы матрицы штрафов таковы, что

$$c_{00}=c_{11}=0,\ \ c_{10}=c_1,\ \ c_{01}=c_0,$$

условие (3.11) преобразуется в следующее:

$$p_{01}c_0=p_{10}c_1. \qquad (3.12)$$

Последнее выражение представляет собой условие выбора областей D_0, D_1 в байесовском классификаторе.

3.2.3 Классификатор Неймана-Пирсона

Классификатор, основанный на стратегии Неймана-Пирсона, используется для случая двух классов, и если известны только функции правдоподобия для каждого из классов. Суть стратегии

Неймана-Пирсона состоит в следующем: задается допустимое значение вероятности ошибки первого рода[1] p_0, а затем классификатор строится таким образом, чтобы обеспечить минимум вероятности ошибки второго рода p_1:

$$\begin{cases} p_1 \to \min\limits_{D_0, D_1} \\ p_0 = p_0^*. \end{cases}$$

Решением задачи Неймана-Пирсона является классификатор вида

$$\Lambda\left(\bar{x}\right) = \frac{f\left(\bar{x}/\Omega_1\right)}{f\left(\bar{x}/\Omega_0\right)} \begin{matrix} > \\ < \end{matrix} \lambda \;\Rightarrow\; \begin{cases} \bar{x} \in D_1 \\ \bar{x} \in D_0 \end{cases}, \qquad (3.13)$$

где значение пороговой величины λ определяется, исходя из условия: $p_0 = p_0^*$. Из выражения (3.13) следует, что *классификатор Неймана-Пирсона – это классификатор отношения правдоподобия.*

3.3 Классификаторы, основанные на параметрических и непараметрических оценках плотности вероятности

Если известны плотности вероятности (ПВ) классов (вектора признаков, соответствующие классам), то для классификации объ-

---

[1] Критерий Неймана-Пирсона в теории статистических решений традиционно используется для проверки гипотез. Поскольку в классической постановке задачи используется только две возможные гипотезы, то различают два типа ошибок:

– *ошибку первого рода* p_0 – в контексте настоящего изложения $p_0 = p_{01}$,

– *ошибку второго рода* p_1 – в контексте настоящего изложения $p_1 = p_{10}$.

Заметим, что в общем случае $p_1 + p_0 \neq 1$. В дальнейшем изложении данная терминология и приведенные обозначения также используются.

ектов можно определить поверхность, разделяющую пространство признаков на области оптимальным в том или ином смысле образом. Следующий вопрос заключается в том, как произвести классификацию объектов, если эти ПВ неизвестны. Естественный подход в этом случае состоит в нахождении оценок этих ПВ на основе имеющейся выборки объектов из каждого класса (как отмечалось, множество объектов, для каждого из которых указан его класс и представляющий его вектор признаков, называется *обучающей выборкой*). И т.о. мы сталкиваемся с проблемой обучения и построения обучающихся систем распознавания, в то время, как до этого имели дело с системами распознавания без обучения. И далее можно будет использовать уже известные классификаторы, но с заменой истинных плотностей вероятности (неизвестных) их оценками.

При оценивании плотности вероятности $f\left(\bar{x}/\Omega_l\right)$, $l=\overline{0,L-1}$ можно выделить два случая. В первом случае плотность вероятности случайного вектора признаков в каждом из классов известна априорно с точностью до некоторого количества параметров, значения которых и следует оценить по обучающим выборкам. В этом случае говорят о *параметрическом оценивании* плотности вероятности (параметрической оценке ПВ) и о *классификации, основанной на параметрическом оценивании плотности вероятности*. Во второй ситуации плотность вероятности оказывается полностью неизвестной. В этой более сложной ситуации необходимо произвести оценку на основе обучающей выборки и «вида» плотности вероятности, и ее «параметров». В этом случае говорят о *непараметрическом оценивании плотности вероятности* (непараметрической оценке плотности вероятности) и о *классификации, основанной на непараметрическом оценивании плотности вероятности*.

3.3.1 Параметрические методы оценки плотности вероятности

Параметрическая оценка плотности вероятности возможна лишь в достаточно простых ситуациях. Она может быть найдена при использовании хорошо известных статистических методов оценивания параметров, таких как *метод моментов* и *метод максимального правдоподобия*. [1]

Итак, пусть *n*-мерный вектор признаков $\bar{x}=\left(x_0,\ldots,x_{n-1}\right)^T$ имеет плотность вероятности $f\left(\bar{x},\Theta\right)$ (обозначения для класса Ω_l опустим, так как процесс построения оценки происходит одинаково для всех классов), где Θ – неизвестный параметр (скаляр) или вектор параметров; $\bar{x}_0,\ldots,\bar{x}_{N-1}$ – выборка объема *N* независимых реализаций вектора $\overline{X}$ (x_k^i – *i*-ая реализация компоненты X_k случайного вектора).

Метод моментов

В соответствии с *методом моментов* оценка параметра Θ ищется в результате решения системы уравнений, получающейся приравниванием теоретических и выборочных моментов:

$$\alpha_{k_0,\ldots,k_{n-1}}\left(\Theta\right)=E\left\{X_0^{k_0}\ldots X_{n-1}^{k_{n-1}}\right\}=\frac{1}{N}\sum_{i=0}^{N-1}\left(x_0^i\right)^{k_0}\ldots\left(x_{n-1}^i\right)^{k_{n-1}}=\alpha^*_{k_0\ldots k_{n-1}},$$

$\alpha_{k_0,\ldots,k_{n-1}}\left(\Theta\right)$ – *теоретический момент* порядка $k=k_0+\ldots+k_{n-1}$,

[1] Отличие от ситуации, изучавшейся в курсе ТВиМС состоит только в том, что выборка представляет собой результат *n* независимых наблюдений над случайным вектором, а не над случайной величиной X.

$\alpha^*_{k_0 \ldots k_{n-1}}$ – *эмпирический (выборочный) момент* того же порядка.

Различных уравнений в этой системе должно быть столько, какова размерность параметра Θ. Можно использовать и *центральные*, и *начальные* моменты.

Например, уравнениями могут быть (для случайной величины):

$$\alpha_k(\Theta) \equiv E\left\{X^k\right\} = \frac{1}{N}\sum_{i=0}^{N-1}\left(x^i\right)^k \;,\; k = \overline{0, N-1}\,,$$

$$\mu_k(\Theta) \equiv D\left\{X^k\right\} = \frac{1}{N}\sum\left(\left(x^i\right)^k - MX^k\right)^2$$

и др.

Свойства оценок, полученных по методу моментов:

- несмещенные – на всегда;
- состоятельные (при $n \to \infty$);
- эффективны – не всегда, как правило нет.

Метод максимального правдоподобия

На выборку $\bar{x}_0, \ldots, \bar{x}_{N-1}$ объема N можно смотреть и как на результат одного наблюдения над системой из N независимых и одинаково распределенных (так же как $\overline{X}$) случайных векторов $\overline{X}_0, \ldots, \overline{X}_{N-1}$. Совместная ПВ такой системы случайных векторов есть

$$\prod_{i=0}^{N-1} f\left(\bar{x}_i, \Theta\right).$$

Приведенное выражение, рассматриваемое при фиксированных выборочных значениях $\bar{x}_0, \ldots, \bar{x}_{N-1}$, как функция Θ называется *функцией правдоподобия*

$$\Theta\left(\bar{x}_0,\ldots,\bar{x}_{N-1},\Theta\right)=\prod_{i=0}^{N-1} f\left(\bar{x}_i,\Theta\right).$$

Значение параметра Θ, при котором функция правдоподобия L достигает максимума, называется *оценкой максимального правдоподобия* (ОМП) параметра Θ и обозначается $\hat{\Theta}$. Смысл: в качестве неизвестного параметра Θ принимается то его значение, при котором получение заданных выбранных значений наиболее вероятно.

ОМП обычно определяют, максимизируя выражение

$$\ln L\left(\bar{x}_0,\ldots,\bar{x}_{N-1},\Theta\right),$$

и находят как решение следующей системы уравнений:

$$\sum_{i=0}^{N-1}\frac{\partial \ln f\left(\bar{x}_i,\Theta\right)}{\partial\Theta}=0.$$

Если параметр Θ имеет размерность r: $\Theta=\left(\Theta_0,\ldots,\Theta_{r-1}\right)^T$, то эта система содержит r уравнений (размерность этой системы такова, какова размерность вектора Θ).

Свойства ОМП:

– несмещенность – не всегда;

– состоятельность – как правило;

– асимптотическая нормальность и как следствие асимптотическая эффективность.

Если при параметрической оценке вид ПВ задан предположительно (гипотетически), то оценивание ПВ не заканчивается выбором ее параметров. Необходимо убедиться, не противоречит ли гипотеза о виде ПВ эмпирической информации. Наиболее универсальным методом проверки такой гипотезы является *критерий*

χ^2 *Пирсона*. Он не зависит ни от истинной плотности вероятности, ни от ее размерности.

Окончательно, байесовский классификатор, основанный на параметрическом оценивании ПВ, записывается в виде (для $c_{ij}=1-\delta_{ij}$):

$$\forall j \neq l \quad \hat{P}(\Omega_l) f\left(\bar{x};\hat{\Theta}_l\right) \geq \hat{P}\left(\Omega_j\right) f\left(\bar{x};\hat{\Theta}_j\right) \Rightarrow x \in \Omega_l \ .$$

Величина $P(\Omega_l)$ могут быть определены на основании частот событий

$$\hat{P}(\Omega_l)=\frac{N}{N_l} \ , \ N=N_0+\ldots+N_{L-1} \ .$$

Здесь:

– N – общее количество элементов обучающей выборки во всех классах;

– N_l – количество элементов обучающей выборки в l-ом классе.

3.3.2 Непараметрические методы оценки плотности вероятности

Метод Парзена оценивания плотности вероятности

Метод подробно рассмотрен в [1-2].

Метод К ближайших соседей

Метод подробно рассмотрен в [1-2].

Гистограммные методы оценки плотности вероятности

Рассматривая k и A в оценке ПВ вида

$$\hat{f}_N\left(\bar{x}\right)=\frac{k}{N}\cdot\frac{1}{A\left(k,N,\bar{X}\right)} \qquad (3.14)$$

как свободные параметры, можно получить различные оценки ПВ, известные под названием *гистограмм*. Рассмотрим два часто встречающиеся вида гистограмм.

Метод гистограмм – ячейки одинакового размера

Разобьем признаковое неравенство на взаимно непересекающиеся ячейки $\Gamma_0,\ldots,\Gamma_{M-1}$, размеры которых одинаковы. Тогда плотность вероятности можно приближенно охарактеризовать числом объектов (векторов признаков), попавших в каждую ячейку. Пример дан на рисунке 3.1.

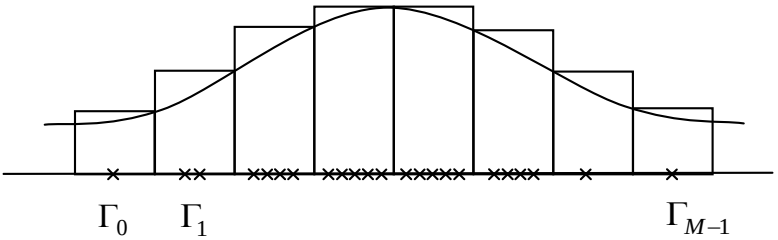


Рисунок 3.1 – Метод гистограмм с ячейками одинакового размера

Достоинства:

– простота построения;

– не требует информации о виде ПВ и,

– если использовать регулярную сетку для построения ячеек Γ_i (с равномерным шагом), то выбор нужной ячейки производится непосредственно.

Недостатки:

– неудовлетворительность оценки вблизи границ областей разбиения, где оценка претерпевает разрыв;

большим недостатком этого метода является то, что он требует слишком большой памяти: при наличии n признаков (размерность вектора $\bar{X}$) и при M градациях по каждой переменной тре-

буется M^n ячеек. Поэтому существует много модификаций этого метода, имеющих своей целью уменьшение числа ячеек. Два из них будут рассмотрены далее.

Метод гистограмм – ячейки разного размера

Число ячеек можно уменьшить, используя ячейки неодинакового размера. Пример для одномерного случая выглядит так, как показано на рисунке 3.2.

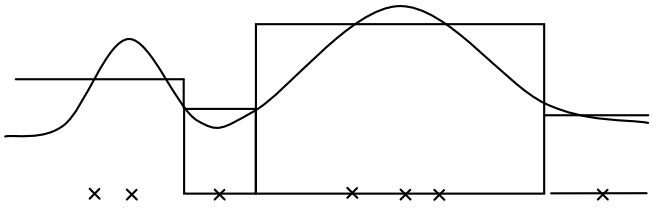


Рисунок 3.2 – Метод гистограмм с ячейками разного размера

Если известно число объектов в каждой ячейке, размер ячейки и её месторасположение, то формулу (3.14) для $\hat{f}_N(\bar{x})$ по-прежнему можно использовать в качестве оценки плотности вероятности. Для реализации этой идеи на практике нужен метод для определения числа объектов и размера каждой ячейки. Одно из возможных решений описывается в следующем алгоритме (аналог кластеризации):

Алгоритм

1) Пусть имеется m ячеек $\Gamma_0,\ldots,\Gamma_{M-1}$, каждая из которых характеризуется координатами центра M_i дисперсиями по каждой координате $\left(\sigma_{i\,0}^2,\ldots,\sigma_{i\,n-1}^2\right)$, и числом объектов k_i, попавших внутрь ячейки. При предъявлении нового объекта $\bar{x}$ вычисляется расстояние от $\bar{x}$ до центра каждой ячейки $\bar{M}_i$ по формуле:

$$d\left(\bar{x},\bar{M}_i\right)=\sum_{j=0}^{n-1}\frac{\left(x_j-M_{ij}\right)^2}{\sigma_{ij}^2},\ i=\overline{0,m-1}.$$

Находится ближайшая к $\bar{x}$ ячейка, т.е. выбирается $\bar{M}_s$ такое, что

$$s=\arg\min_{j=0,m-1}\cdot d\left(\bar{x},M_j\right).$$

Тогда объект «относится» к ячейке в соответствие со следующим правилом:

$$\begin{cases} d\left(\bar{x},M_s\right)\le\tau\Rightarrow\bar{x}\in\Gamma_s, \\ d\left(\bar{x},M_s\right)\ge\theta\cdot\tau\Rightarrow \textit{создается новая ячейка с центром}\ \bar{x}\ (\theta>1). \end{cases}$$

В остальных случаях вопрос о принадлежности $\bar{x}$ к какой-либо из ячеек остается нерешенным.

Здесь τ и θ – свободные параметры, определяющие число ячеек и точность аппроксимации.

2) Когда новый объект попадает в *i*-ю ячейку, параметры ячейки пересчитываются следующим образом:

a) Увеличивается на единицу число объектов в ячейке: $k_i=k_i+1$.

b) вычисляется новый «центр» ячейки $\bar{M}_i$ как выборочное среднее векторов, попавших в эту ячейку:

$$\bar{M}_i=\frac{1}{k_i}\sum_{\bar{x}\in\Gamma_i}\bar{x}\ .$$

c) Вычисляются новые дисперсии по формулам:

$$\sigma_{ij}^2=\max\left[\sigma_{ij}^2(0),S_{ij}^2\right],$$

где

$$S_{ij}^2 = \frac{1}{k_i} \cdot \sum_{\bar{x} \in \Gamma_i} \left(x_{ij} - M_{ij} \right)^2, \; i = \overline{0, m-1}, \; j = \overline{0, n-1}$$

– это выборочная дисперсия *j*-ой переменной в *i*-ой ячейке, подсчитанная по k_i объектам.

$\sigma_{ij}^2(0)$ – первоначально заданное (минимальное) значение дисперсии ($\sigma_{ij}^2(0)$ заменятся на σ_{ij}^2 только, если $\sigma_{ij}^2(k_i) > \sigma_{ij}^2(0)$)

3) Первый объект всегда становится центром новой ячейки. Второй объект относится к новой или существующей ячейке в соответствии с пунктом 1 и т.д. После того как все объекты распределены или среднее число объектов в ячейке достигнет некоторого заданного значения, объекты, по которым не было принято решение, распределяются по ближайшим ячейкам, и параметры этих ячеек пересчитываются в соответствии с пунктом 2.

Свободные параметры τ, θ и $\sigma_{ij}^2(0)$ можно подобрать путем повторения описанного выше процесса для одного и того же множества объектов (обучающие выборки).

Достоинства и недостатки гистограмм с ячейками разного размера.

Достоинства :

– простота построения;

– не требует информации о виде ПВ.

Недостатки:

– вычислительная сложность при получении значения плотности (долго искать требуемую ячейку);

– неопределяемость при выборе способа разбиения пространства на ячейки (и, следовательно, существенный элемент субъективности в оценке);

– неудовлетворительность оценки вблизи границ областей разбиения, где оценка притерпевает разрыв.

Как видно, недостаток «неудовлетворительность оценки вблизи границ областей разбиения, где оценка претерпевает разрыв» является общим для всех гистограмм. Отсюда возникает необходимость в процедуре сглаживания. В многомерном случае эта задача чрезвычайно сложна. Поэтому оценку ПВ в виде гистограммы можно полагать удовлетворительной только при весьма больших выборках.

Гистограммные методы оценки плотности вероятности. Дерево решений и случайный лес

Дерево решений: при построении пространство (область значений) признаков последовательно иерархически разбивается на подпространства (подобласти), которые также иерархически подвергаются разбиению до тех пор, пока (в идеале) на терминальной вершине дерева (в малой подобласти исходного пространства) не останутся только объекты одного из классов. В простейшей реализации дерева решений при попадании вновь классифицируемого вектора в терминальную вершину (в подобласть, за которую отвечает эта терминальная вершина) он относится к тому классу, представителей которого в этой вершине большинство.

Древовидное решающее правило

Пусть R_0 – риск классификации на верхнем уровне ($x-\Omega_1$, $o-\Omega_0$). Дальнейшая процедура выполняется рекурсивно: если текущий риск больше допустимого, то продолжает разбиение терминальных вершин с ошибками, как показываем на рисунке 3.3 (итог разбиения).

Риск в этом случае формируется из ошибок на терминальных вершинах и по мере их деления снижается, как показано на рисун-

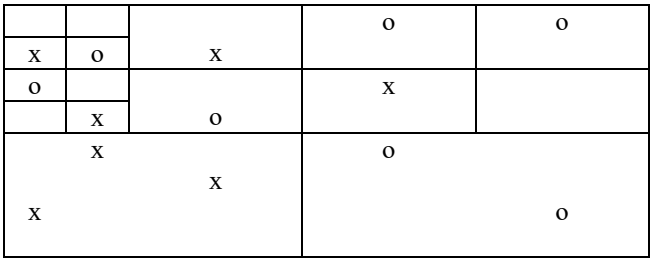


Рисунок 3.3 – Процесс разбиения пространства признаков деревом решений

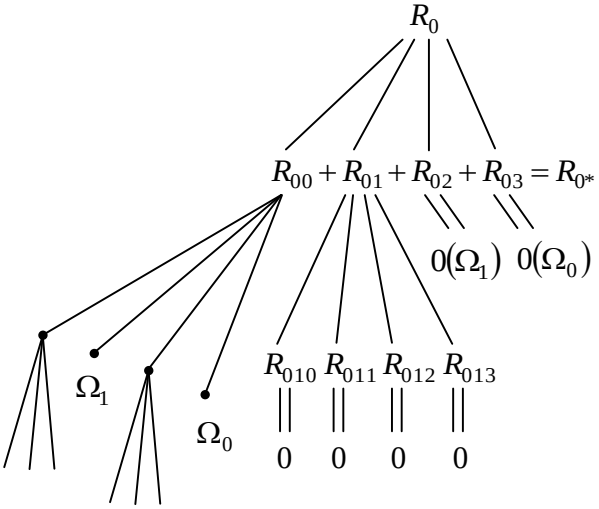


Рисунок 3.4 – Иллюстрация процесса снижения эмпирического риска для дерева решений

ке 3.4. Проблема такого решения – неясно, когда следует остановить процесс разбиения терминальных вершин. Очевидно, что эмпирический риск на выборке обучения при продолжении разбиений гарантированно сходится к «0», поскольку классификатор чрезмерно настраивается на обучающую выборку (эффект пе-

реобучения). Если остановить разбиение рано – будет завышенная ошибка классификации. Ответ на этот вопрос дает теория Вапника-Червоненкиса (см. подраздел 2.4 настоящего пособия), иллюстрация метода структурной минимизации риска показано на рисунке 3.5. Анализируется поведение кривой эмпирического риска на контрольной выборке (см. рисунок 3.5), рекомендуется использование кросс-валидации. В качестве решения выбирается то, для которого риск на контрольной выборке будет минимален.

Случайный лес

Случайный лес – это реализация метода (взвешенного) голосования (см. подразделы 6.4-6.5 настоящего пособия) нескольких классификаторов-деревьев. Случайный лес используется для устранения эффектов переобучения, которому подвержено дерево решений. Каждое из деревьев строится на случайном подмножестве обучающего множества. Дополнительно, при переходе на подуровни дерева случайным образом выбираются компоненты вектора признаков, которые могут быть использованы для определения подобластей (подходы бэггинга и RMS – см. подраздел 6.5 настоящего пособия).

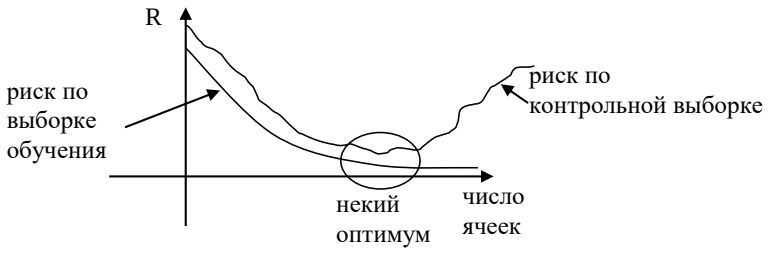


Рисунок 3.5 – Процесс изменения эмпирического риска
для дерева решений на обучающей и тестирующей выборке

3.4 Результаты третьего раздела

Третий раздел посвящен изложению основ статистической теории распознавания. Представлено формальное определение качества классификатора, принятое в статистической теории распознавания. Рассмотрены оптимальные стратегии классификации, применимые при различном количестве доступной информации о задаче: байесовский классификатор, минимаксный классификатор, классификатор Неймана-Пирсона.

Вопросы построения линейных классификаторов в данном разделе не рассматривались – читателю для изучения рекомендуется ранее изданные пособия [1-2]. В противоположность им в настоящем разделе основной упор сделан на вопросах построения классификаторов, основанных на параметрических и непараметрических оценках плотности вероятности. В том числе: оценки Парзена, метода K ближайших соседей, метода гистограмм с ячейками одинакового и адаптивного размеров, дерево решений и случайный лес деревьев.

4 ПОСЛЕДОВАТЕЛЬНЫЕ МЕТОДЫ КЛАССИФИКАЦИИ

4.1 Основные понятия последовательного анализа

В рассмотренных ранее методах классификации все *n* признаков имелись в распоряжении одновременно. Подобные процедуры называют *решающими процедурами с фиксированным объемом выборки.* Но если должна учитываться стоимость выполнения измерений признаков или признаки входных образов по своей природе возникают последовательно, необходимо применять *последовательную решающую процедуру* к этому классу задач распознавания образов.

Существую связь между количеством информации, получаемым при измерении признаков, и стоимостью выполнения этих измерений. Рациональное соотношение между качеством распознавания и числом измерений признаков можно получить, осуществляя измерение признаков последовательно и принимая решение на том шаге, когда достигнута достаточная или необходимая точность классификации. Математическим методом решения задач этого типа является *аппарат последовательной проверки гипотез* [36, 44, 68–71].

Определение: Последовательный эксперимент – это эксперимент, ход которого определенным образом зависит от получаемых результатов.

Последовательная природа такого эксперимента проявляется двояко.

Во-первых, на каждом (последовательном) шаге может быть произведен выбор того эксперимента (признака), который должен быть выполнен (измерен). В этом случае измерения, используемые на последующих шагах, зависят от ранее полученных результатов.

Во-вторых, формируется правило для завершения эксперимента. Это правило позволяет продолжать эксперимент до тех пор, пока не станет очевидным, что совокупность сделанных до данного шага наблюдений (измерений) признаков дает близкое к оптимальному решение.

Последовательный (классический) анализ обычно ограничивается экспериментами, в которых предусмотрены только правила остановки, то есть последовательность измерений (признаков) задана заранее.

Особенно следует отметить то, что последовательная процедура не уменьшает вероятность ошибки, а сокращает число признаков (измерений), требуемых для принятия решения.

4.2 Последовательный критерий отношения вероятностей Вальда

Наиболее важным результатом в последовательном анализе является последовательный критерий отношения вероятностей (п.к.о.в.) Вальда [44, 68]. Этот критерий используется для решения о выборе между двумя простыми гипотезами (классами) – Ω_0 и Ω_1, (число классов L=2).

Итак, пусть случайный вектор $\overline{X} = \left(X_0, \ldots, X_{n-1}, \ldots\right)^T$ имеет функцию плотности $f\left(\overline{x}/\Omega_l\right)$ $\left(l \in \overline{0, L-1}\right)$ и наблюдения $x_0, \ldots, x_{n-1}, \ldots$ компонент вектора производится последовательно. Задача состоит в нахождении решающего правила (критерия), ко-

торое принимает решение в класс Ω_0 или Ω_1 на основании текущего множества из n наблюдений или откладывает решение на последующие шаги.

Пусть также мы хотим получить решение задачи классификации с качеством: вероятностями верной классификации $\left(1-p_0^*\right)$ для Ω_0, $\left(1-p_1^*\right)$ для Ω_1. Здесь p_0^* – задаваемая вероятность ложного обнаружения (тревоги), p_1^* – задаваемая вероятность пропуска объекта.

П.к.о.в. Вальда: На n-том шаге последовательной процедуры вычисляется отношение правдоподобия (ОП) для n признаков:

$$L\left(x_0,\ldots,x_{n-1}\right)=\frac{f\left(x_0,\ldots,x_{n-1}|\Omega_1\right)}{f\left(x_0,\ldots,x_{n-1}|\Omega_0\right)}\equiv\frac{f_n\left(\bar{x}|\Omega_1\right)}{f_n\left(\bar{x}|\Omega_0\right)}\equiv L_n\left(\bar{x}\right). \qquad (4.1)$$

Полученная величина сравнивается с двумя порогами A и B, называемыми *останавливающими порогами* или *останавливающими границами.* Решение о классификации или продолжении измерений принимается следующим образом:

если $L_n\left(\bar{x}\right)\geq A$, то $\bar{x}\in\Omega_1$,

если $L_n\left(\bar{x}\right)\leq B$, то $\bar{x}\in\Omega_0$,

если $B<L_n\left(\bar{x}\right)<A$, то должны быть произведены дополнительные измерения и процесс продолжается до $(n+1)$-го шага.

Постоянные A и B называются соответственно *верхним и нижним порогами.* Они могут быть выбраны таким образом, чтобы получить заданные значения вероятностей ошибок p_0^* и p_1^*.

Утверждение [44, 68]:

$$A \le \frac{1-p_1^*}{p_0^*}\ ,\ \ B \ge \frac{p_1^*}{1-p_0^*}\ . \qquad (4.2)$$

Замечания:

1. В п.к.о.в. не требуется независимости и равенства распределений случайных величин $X_0, X_1, \ldots, X_n$.

2. П.к.о.в. завершается (сходится) с вероятностью «1» как для класса Ω_0 , так и для класса Ω_1 .

3. П.к.о.в. минимизирует среднее количество наблюдений $E\{n/\Omega_0\}$ и $E\{n/\Omega_1\}$, требуемое для достижения заданных значений вероятностей ошибок p_0^* и p_1^* .

4. П.к.о.в. не зависит от априорных вероятностей $P(\Omega_0)$ и $P(\Omega_1)$.

5. Если вместо ОП в выражении (1) используется логарифм ОП (ЛОП), то пороговые значения выбираются из условий:

$$a = \ln A \le \ln \frac{1-p_1^*}{p_0^*}\ ,\ \ b = \ln B \ge \ln \frac{p_1^*}{1-p_0^*}\ .$$

6. Если приращения в п.к.о.в. (1) невелики и неравенства (4.2) приближенно можно считать равенствами (только для непрерывных случайных векторов), тогда вероятности ошибок классификации можно рассчитать через пороговые значения *A* и *B:*

$$p_1 \approx \frac{B(A-1)}{A-B}\ ,\ \ p_0 \approx \frac{1-B}{A-B}\ .$$

Среднее число наблюдений в п.к.о.в.

Если компоненты вектора $\overline{X}$ – независимые и одинаково распределенные случайные величины $\left(X_i \cong X\right)$, тогда выполняются равенства:

$$E\left\{n|\Omega_0\right\}=\frac{\ln A\cdot p_0+\ln B\cdot\left(1-p_0\right)}{E\left\{Z|\Omega_0\right\}},\ E\left\{n|\Omega_1\right\}=\frac{\ln A\cdot\left(1-p_1\right)+\ln B\cdot p_1}{E\left\{Z|\Omega_1\right\}},$$

$$\text{где } Z=\ln\frac{f\left(X|\Omega_1\right)}{f\left(X|\Omega_0\right)}. \qquad (4.3)$$

Примечание: В ситуации, когда независимые компоненты вектора признаков $\overline{X}$ имеют нормальный закон распределения с параметрами $N\left(m_l,\sigma_l^2\right)$ и $\sigma_0=\sigma_1$, тогда

$$E\left\{Z|\Omega_1\right\}=\frac{\left(m_1-m_0\right)^2}{2\sigma^2},\ E\left\{Z|\Omega_0\right\}=-\frac{\left(m_1-m_0\right)^2}{2\sigma^2}.$$

4.3 Модифицированный последовательный критерий отношения вероятностей

Как было показано ранее, вероятности ошибок могут быть предварительно заданы при применении п.к.о.в. Однако в этом случае число измерений признаков, необходимое для окончательного решения, является случайной величиной, которая в общем случае зависит от заданных вероятностей ошибок, и с положительной вероятностью может быть больше любой постоянной. Так как практически нецелесообразно допускать произвольно большое число измерений признаков для завершения последовательного процесса, мы часто бываем заинтересованы в установлении верх-

ней границы для числа измерений признаков, в пределах которой классификатор образов должен получить окончательное решение.

Определение. Алгоритм п.к.о.в., в котором при последовательной классификации пороги претерпевают изменения для усечения процесса классификации, называют *модифицированным п.к.о.в.*

4.3.1 Постановка задачи модифицированного п.к.о.в.

Модифицированный п.к.о.в. формулируется следующим образом [44,68]. Пусть $E\left(n|\Omega_l\right)$ - среднее число измерений признаков в классах Ω_l $\left(l=0,1\right)$, то есть соответствующее принятию окончательного решения. В соответствии с требованием, чтобы при отнесении $\overline{X}$ к классу вероятности ложного распознавания не превышали p_0^* и p_1^*, задача заключается в построении процедуры с изменяющимися во времени порогами для решения $\overline{X}\in\Omega_1$, или $\overline{X}\in\Omega_0$, при котором $E\left(n|\Omega_l\right)$ минимально.

Процедура модифицированного п.к.о.в. заключается в следующем:

пусть $g_1(n)$ и $g_2(n)$ являются постоянными или монотонно невозрастающей и неубывающей функциями *n.* Классификатор непрерывно производит измерения до тех пор, пока последовательное отношение вероятностей $L_n\left(\overline{x}\right)$ лежит между $e^{g_1(n)}$ и $e^{g_2(n)}$, то есть $e^{g_2(n)}<L_n\left(\overline{x}\right)<e^{g_1(n)}$ $n=0,1,2...$ В противоположной ситуации:

если $L_n\left(\overline{x}\right)\geq e^{g_1(n)}$, то принимается решение $\overline{x}\in\Omega_1$,

если $L_n\left(\overline{x}\right)\leq e^{g_2(n)}$, то принимается решение $\overline{x}\in\Omega_0$.

4.3.2 Важный частный случай

Рассмотрим модифицированный п.к.о.в., задав границы $g_1(n)$ и $g_2(n)$ следующим образом:

$$g_1(n)=a^*\left(1-\frac{n}{N}\right)^{r_1},\ g_2(n)=b^*\left(1-\frac{n}{N}\right)^{r_2} \qquad (4.4)$$

при $r_1>0$, $r_2\le 1$, $a^*>0$, $b^*<0$. Здесь N – предварительно заданное число измерений признаков, проводимых до того, как произойдет усечение и классифи-катор будет вынужден дать окончательное решение. Используем далее обозначение

$$\tilde{L}_n=\ln L_n=\ln\frac{f\left(\bar{x}|\Omega_1\right)}{f\left(\bar{x}|\Omega_0\right)}.$$

Тогда модифицированный п.к.о.в. определяется следующими неравенствами [44,68]:

$$b^*\left(1-\frac{n}{N}\right)^{r_2}<\tilde{L}_n\left(\bar{x}\right)<a^*\left(1-\frac{n}{N}\right)^{r_1}.$$

Нарушение любого из этих неравенств соответствует классификации $\bar{x}\in\Omega_1$ или $\bar{x}\in\Omega_0$.

4.3.3 Среднее число наблюдений и вероятности ошибочной классификации в модифицированном п.к.о.в.

Для модифицированного п.к.о.в. с границами $g_1(n)$ и $g_2(n)$ в виде (4.4), а также при условии, что компоненты вектора $\overline{X}$ – независимые и одинаково распределенные случайные величины $\left(X_i\cong X\right)$, средняя длительность наблюдений составляет [44, 68]:

$$E\{n|\Omega_1\}=\frac{a^*}{E\{Z|\Omega_1\}+r_1\dfrac{a^*}{N}},\ E\{n|\Omega_0\}=\frac{b^*}{E\{Z|\Omega_0\}+r_0\dfrac{b^*}{N}}$$

где

$$Z=\ln\frac{f(X|\Omega_1)}{f(X|\Omega_0)}.$$

Выражения для вероятностей ошибок имеет вид:

$$p_0=e^{-a^*}\left(1+\frac{r_1\left(a^*\right)^2}{NE\{Z|\Omega_1\}+r_1a^*}\right),\ p_1=e^{b^*}\left(1-\frac{r_0\left(b^*\right)^2}{NE\{Z|\Omega_0\}+r_0b^*}\right).$$

4.3.4 Связь п.к.о.в. и модифицированного п.к.о.в.

Если компоненты вектора $\overline{X}$ – независимые и одинаково распределенные случайные величины, а границы начинаются в одинаковых точках $\left(a^*=a,\ b^*=b\right)$, тогда средние длительности наблюдений и вероятности ошибок для п.к.о.в. Вальда и модифицированного п.к.о.в. связаны следующими соотношениями [44, 68]:

$$\frac{E^{Вальда}\{n|\Omega_l\}}{1+r_l}<E\{n|\Omega_l\}\le E^{Вальда}\{n|\Omega_l\},\ p_l\ge p_l^{Вальда},\ l=\overline{0,1}.$$

4.3.5 Выводы для п.к.о.в.

1. Процесс классификации всегда заканчивается при заранее определенном максимальном числе измерений признаков.

2. Среднее число измерений признаков можно изменять: обычно оно меньше требуемого для п.к.о.в. Вальда с фиксированными параллельными границами.

3. Путем выбора точек границ можно получить такие же малые вероятности ошибок, как в п.к.о.в. Вальда.

4.4 Обобщенные п.к.о.в. для случая более двух классов

4.4.1 Обобщенный п.к.о.в.

На *n*-том шаге вычисляется обобщенные последовательные отношения вероятностей (ПОВ) для каждого класса [44, 68]:

$$U_n\left(\bar{x}|\Omega_l\right)= f_n\left(\bar{x}|\Omega_l\right)\left[\prod_{j=0,\, j\neq l}^{L-1} f_n\left(\bar{x}|\Omega_j\right)\right]^{-\frac{1}{L-1}},\ l=\overline{0,L-1}.$$

Далее величина $U_n\left(\bar{x}|\Omega_l\right)$ сравнивается с останавливающей границей для l -го класса образов A_l : если существует номер класса l ($l=\overline{0,L-1}$), для которого выполняется неравенство $U_n\left(\bar{x}|\Omega_l\right)< A_l$, тогда соответствующий класс Ω_l исключается из дальнейшего рассмотрения, т.е. считается, что текущий объект этому классу не принадлежит.

После исключения на текущем шаге всех классов Ω_l , для которых выполнялось условие $U_n\left(\bar{x}|\Omega_l\right)< A_l$, общее количество классов уменьшается, а из оставшихся составляется новое множество обобщенных ПОВ. Исключения, производимые на каждом шаге алгоритма, продолжаются до тех пор, пока не останется только один класс. Этот класс и принимается в качестве результата распознавания.

Останавливающая граница A_l для класса Ω_l задается следующим образом:

$$A_l = (1 - p_{ll}) \left[\prod_{j=0,\, j \neq l}^{L-1} (1 - p_{jl}) \right]^{-\frac{1}{L-1}}, \quad l = \overline{0, L-1}.$$

4.4.2 Обобщенный модифицированный п.к.о.в.

На *n*-том шаге вычисляется обобщенные ПОВ для каждого класса:

$$U_n\left(\bar{x} | \Omega_l\right) = f_n\left(\bar{x} | \Omega_l\right) \left[\prod_{j=0,\, j \neq l}^{L-1} f_n\left(\bar{x} | \Omega_j\right) \right]^{-\frac{1}{L-1}}, \quad l = \overline{0, L-1}.$$

Далее величина $U_n\left(\bar{x} | \Omega_l\right)$ сравнивается с останавливающей границей для l -го класса образов $g_l(n)$: если существует номер класса l, для которого $U_n\left(\bar{x} | \Omega_l\right) < g_l(n)$, тогда этот класс исключается из дальнейшего рассмотрения. После исключения на текущем шаге всех классов Ω_l , для которых выполнялось это условие, общее количество классов уменьшается, а из оставшихся составляется новое множество обобщенных ПОВ. Исключения, производимые на каждом шаге алгоритма, продолжаются до тех пор, пока не останется только один класс. Этот класс и принимается в качестве результата распознавания.

Границы $g_l(n)$, $n = \overline{0, N}$ в общем случае являются функциями n и не обязательно одинаковыми для всех классов. По аналогии с модифицированным п.к.о.в. можно предложить границы в виде:

$$g_l(n) = A_l^* \left(1 - \frac{n}{N}\right)^{r_l}, \quad n = \overline{0, N}.$$

4.5 Байесовская последовательная решающая процедура

4.5.1 Постановка задачи

Напомним некоторые сведения о байесовском классификаторе, использующем фиксированный (по длине) вектор признаков. А именно, байесовский классификатор – это классификатор, который обеспечивает минимум общего риска:

$$R=\sum_{l=0}^{L-1}\sum_{j=0}^{L-1}c_{lj}P(\Omega_l)\,p_{lj}\ ,$$

где

$$p_{lj}=\int_{D_j} f\left(\bar{x}/\Omega_l\right)d\bar{x}\ .$$

Эти два выражения можно переписать по-другому:

$$R=\sum_{l=0}^{L-1}P(\Omega_l)\sum_{j=0}^{L-1}\int_{D_j} c_{lj}\,f\left(\bar{x}/\Omega_l\right)dx\ .$$

Введем обозначение: $L\left(\Omega_l,d\left(\bar{x}\right)\right)=c_{lj}$ при $\bar{x}\in D_j$. Эта функция $L(\ldots)$ характеризует средние потери при классификации из класса Ω_l в класс Ω_j . Тогда последнее выражение можно записать так:

$$R=\sum_{l=0}^{L-1}P(\Omega_l)\int_{D} L\left(\Omega_l,d\left(\bar{x}\right)\right)f\left(\bar{x}/\Omega_l\right)d\bar{x}\ .$$

Теперь обобщим это выражение на случай последовательной классификации. Очевидна одна модификация – необходим учет стоимости измерений признаков. Для этого введем величину

$S_n^*\left(\bar{x}\right)$ – стоимость (суммарная) наблюдения признаков с нулевого по *n*-ый: $x_0,\ldots,x_n$. Также введем понятие решающей функции на *n*-ом шаге. А именно, пусть $d^n\left(\bar{x}\right)$ – дискретная (решающая) функция, основанная на замерах $x_0,\ldots,x_n$ и принимающая по ним решения о классе объекта. Тогда общий риск классификации при использовании последовательной стратегии запишется в виде [44, 68]:

$$R=\sum_{l=0}^{L-1}P(\Omega_l)\sum_{n=0}^{N-1}\int_{D^n}\left[S_n^*\left(\bar{x}\right)+L\left(\Omega_l,d^n\left(\bar{x}\right)\right)\right]f_n\left(\bar{x}/\Omega_l\right)dx_0\ldots dx_n\ .$$

Здесь D^n – область признакового пространства *D*, которая используется для принятия окончательного решения в некоторый класс на *n*-ом испытании. Очевидно $D^n\bigcap D^j=\varnothing$. *N* – общее (максимальное) количество испытаний/признаков.

Чтобы было понятие, рассмотрим это выражение для *N*=1 и 2.

Для *N*=1

$$R=\sum_{l=0}^{L-1}P(\Omega_l)\int_{D^0=D}\left[S_0^*(x)+L\left(\Omega_l,d^0(x)\right)\right]f\left(x/\Omega_l\right)dx\ .$$

Как очевидно, это выражение – обычное выражение для величины риска классификации, записанное с учетом стоимости измерения.

Для *N*=2

$$R=\sum_{l=0}^{L-1}P(\Omega_l)\int_{D^0\subseteq D}\left[S_0^*(x_0)+L\left(\Omega_l,d^0(x_0)\right)\right]f\left(x_0/\Omega_l\right)dx_0+$$

$$+\sum_{l=0}^{L-1}P(\Omega_l)\int_{D^1=D\setminus D_0}\left[S_1^*(x_0,x_1)+L\left(\Omega_l,d^1(x_0,x_1)\right)\right]f\left(x_0,x_1/\Omega_l\right)dx_0dx_1.$$

Здесь также все понятно. В общем случае мы можем остановить процесс решения как на одном признаке (x_0), так и после просмотра 2-х признаков: (x_0 и x_1). Поэтому в общем риске эти две ситуации (они несовместны) учитываются отдельно и соответствующие им величины рисков складываются. Заметим, что каждое из таких слагаемых (по $n=\overline{0,\ldots,N-1}$) – фактически обычный общий риск для вектора $\left(x_0,\ldots,x_n\right)^T$, рассчитанный на той области D^n , на которой принимается окончательное решение .

Задача построения байесовской последовательности процедуры заключается в нахождении множества решающих правил $d^n\left(x_0,\ldots,x_n\right)$, $n=\overline{0,\ldots,N-1}$, которые минимизируют общий риск *R* для заданного множества величин стоимостей наблюдений (или симметричного нахождения множества областей $\left\{D_l^n\right\}_{\substack{n=\overline{0,N-1}\\ l=\overline{0,L-1}}}$, минимизирующих риск *R*).

Такая постановка задачи, как очевидно, не учитывает возможности изменения порядка замеров/измерений признаков – то есть является классической. Но далее мы коснемся и возможности изменения порядка.

Итак, сформированный критерий дает возможность сформировать и оптимальную (байесовскую) стратегию поведения при классификации. Действительно, на каждом шаге можно вычислить средний риск (не общий, т.к. вектор задан) продолжения последовательного процесса (т.е. выполнения дополнительного наблюдения), и средний риск остановки процесса и принятия окончательного решения (на текущем шаге). Байесовское решение оптимально в смысле минимизации среднего риска. Поэтому решение, продолжать последовательный процесс или остановить его и принять окончательное решение, может быть получено сравне-

ние следующих средних рисков на каждом шаге. **Смысл**: *дополнительные измерения признаков следует продолжать до тех пор, пока это приводит к понижению риска*.

Однако в этой простой стратегии есть очевидные сложности. Действительно, легко определить средний риск, связанный с решением об окончании процедуры и принятии решения (можно посчитать ошибку классификации из заданных выборок). Но не легко определить средний риск, связанный с выполнением дополнительного наблюдения. Для случая выполнения одного дополнительного наблюдения средний риск относится к выполнению следующего шага и лучшего из возможных продолжений. Соответственно, чтобы определить лучшее решение на данном шаге (т.е. продолжать или не продолжать процесс), необходимо знать лучшее решение в будущем.

Другими словами, поскольку речь идет о поиске оптимальной решающей процедуры, порядок действия в натуральном времени от настоящего к будущему оказывается малоэффективным, т.к. текущий оптимум включает в себя будущий оптимум. Поэтому единственной альтернативой является действие обратно во времени, т.е. по оптимальному будущему поведению определять оптимальное поведение в настоящем.

Именно поэтому процедуру последовательной классификации для байесовского критерия принято называть обратной. Далее будет более понятно, почему все это именно так.

4.5.2 Обратная процедура конечного последовательного распознавания

Рассмотрим поочередные наблюдения или замеры признаков $x_0,\ldots,x_n$, $n=\overline{0,N-1}$ с известной функцией распределения $F\left(x_{n+1}/x_0,\ldots,x_n\right)$ случайных величин x_{n+1} при заданной после-

довательности $x_0,\ldots,x_n$. После наблюдения очередного замера признака, решение, принимаемое классификатором, сводится либо к выбору усечения последовательности измерения признаков и принятию окончательного решения (выбору класса), либо к наблюдению следующего замера перед принятием окончательного решения. Обозначим $\rho_n(x_0,\ldots,x_n)$ – минимальный средний риск последовательного решающего процесса $x_0 \to x_n$. Пусть S_{n+1} – стоимость выполнения одного дополнительного измерения x_{n+1} (т.е. это стоимость получения еще одного признака, а не стоимость замеров всех $x_0,\ldots,x_{n+1}=S^*_{n+1}(\Omega)$).

Пусть $R(x_0,\ldots,x_n;d_l)$ – риск принятия окончательного решения о том, что следующий объект был из класса Ω_l на основании замеров $x_0,\ldots,x_n$.

Если классификатор останавливает процесс, то при оптимальном решающем правиле средний риск составит:

$$\min_l R(x_0,\ldots,x_n;d_l).$$

Если классификатор продолжить процесс, то необходимо сделать еще один замер x_{n+1} и тогда средний риск будет иметь *две составляющие*: *собственно плату за выполнение замера* S_{n+1} и *величину, характеризующую риск принятия решения* на $(n+1)$-ом шаге. Т.к. значение x_{n+1} нам не известно, то мы можем опираться только на среднее значение этого риска. А именно, на величину

$$\int_{X_{n+1}} \rho_{n+1}(x_0,\ldots,x_n,x_{n+1})\,dF\left(x_{n+1}/x_0,\ldots,x_n\right).$$

Таким образом, средний риск продолжить процесс:

$$S_{n+1}+\int\limits_{X_{n+1}}\rho_{n+1}(x_0,\ldots,x_n,x_{n+1})dF\left(x_{n+1}/x_0,\ldots,x_n\right).$$

Интегрирование производится по области возможных значений x_{n+1}.

Окончательно, основное функциональное (рекуррентное) уравнение, определяющее последовательность средних рисков $\rho_n(\cdot)$ имеет вид [44,68]:

$$\rho_n(x_0,\ldots,x_n)=\min\begin{cases}\text{продолжить: } S_{n+1}+\\ \quad+\int\limits_{X_{n+1}}\rho_{n+1}(x_0,\ldots,x_n,x_{n+1})dF\left(x_{n+1}/x_0,\ldots,x_n\right),\\ \text{остановить: } \min\limits_l R(x_0,\ldots,x_n,d_l).\end{cases}$$

В случае конечного процесса ($n=\overline{0,N-1}$), оптимальное решающее правило может быть определено в обратном порядке, исходя из заданной функции риска (или вероятностей ошибок) на последнем шаге. Действительно, на последнем шаге

$$\rho_{N-1}(x_0,\ldots,x_{N-1})=\min_l R(x_0,\ldots,x_{N-1};d_l),$$

так как никаких замеров мы сделать больше не можем. Тогда для предыдущего шага:

$$\rho_{N-2}(x_0,\ldots,x_{N-2})=\begin{cases}\text{продолжить: } S_{N-1}+\\ \quad+\int\limits_{X_{N-1}}\rho_{N-1}(x_0,\ldots,x_{N-1})dF\left(x_{N-1}/x_0,\ldots,x_{N-2}\right),\\ \text{остановить: } \min\limits_l R(x_0,\ldots,x_{N-2},d_l),\end{cases} \quad (4.5)$$

где $\rho_{N-1}(\cdot)$ получено ранее. Продолжая так до самого первого (x_0) шага, получим выражение для среднего риска для x_0 :

$$\rho_0(x_0)=\min\begin{cases}\text{продолжить: } S_1+\int\limits_{X_1}\rho_1(x_0,x_1)dF\left(x_1/x_0\right),\\ \text{остановить: } \min\limits_l R(x_0,d_l).\end{cases} \quad (4.6)$$

Здесь ρ_1 – получена на предыдущем шаге.

Но, вообще говоря, принять решение можно, даже не измеряя первый признак (например, по априорным вероятностям)!! Поэтому надо учесть и эту ситуацию. Это делается еще одним шагом:

$$\rho=\min\begin{cases}\text{продолжить: } S_0+\int\limits_{X_0}\rho_0(x_0)dF(x_0),\\ \text{остановить: } \min\limits_l R(d_l)\equiv 1-\max\limits_l P(\Omega_l).\end{cases}$$

То есть возможно, что процесс классификации «закончится не начавшись», за счет выбора номера класса по априорным вероятностям этих классов. А если они равны – подбрасывания монетки.

То есть все зависит от стоимости признаков и их «разделяющей» способности.

Очевидны *вычислительные сложности*, которые возникают при решении такой задачи:

– оценивание условных вероятностей высокого порядка (память, вычислительные ресурсы – время),

– последовательные численные вычисления интегралов.

Для упрощения этой задачи используют ряд подходов (мы их не рассматриваем):

– используют дискретные распределения вероятностей (замена интегралов суммы);

– задание аналитических моделей распределений (чтобы аналитически/таблично рассчитывать интегралы);

– использование предположения о марковой зависимости замеров.

Замечание.

Если стоимости наблюдений равны, x_i – независимые и одинаково распределенные случайные величины и $N=\infty$ (не ограничено) тогда п.к.о.в. Вальда и последовательная байесовская процедура классификации совпадают.

4.6 Результаты четвертого раздела

Четвертый раздел посвящен изложению основ последовательных методов классификации. Данная группа методов в значительной степени базируется на *аппарате последовательной проверки гипотез*, развиваемый в классической статистике. В рамках данного раздела рассмотрены два наиболее известных решения:

– последовательный критерий отношения вероятностей Вальда (п.к.о.в.) и

– байесовская последовательная решающая процедура.

Критерий п.к.о.в. рассмотрен в нескольких вариантах:

– классическом (два класса и неизменные пороги),

– модифицированном (два класса и изменяемые пороги) и

– обобщенном (более двух классов).

Представлены основные положения и теоремы, показывающие взаимосвязь этих вариантов, характеризующих среднюю длительность наблюдений и качество получаемого решения.

5 АЛГЕБРАИЧЕСКИЕ МЕТОДЫ РАСПОЗНАВАНИЯ

Наряду со статистическим/вероятностным подходом, который очень часто используется в задачах распознавания, существуют и другие, конкурирующие с ним, подходы. Среди них выделяется алгебраический подход, предложенный в 70-х годах академиком Ю.И.Журавлевым и развиваемый его коллегами. Среди наиболее известных результатов школы академика Ю.И.Журавлева можно выделить следующие:

– *алгоритм вычисления оценок* (далее – АВО) – Ю.И. Журавлев, И.Б. Гуревич [45];

– алгебра алгоритмов распознавания (Ю.И. Журавлева) [45];

– теория локальных и универсальных ограничений (К.В. Рудаков [46]);

– комбинаторная теория надежности алгоритмов (2010, К.В. Воронцов [50-51]).

В рамках настоящего раздела кратко представлены первые два результата.

5.1 Класс алгоритмов вычисления оценок

Несмотря на название, под АВО понимается не столько алгоритм, сколько «модель» алгоритма, который задает целый класс алгоритмов распознавания. Конкретный алгоритм получается, когда определен каждый элемент этой модели, являющийся функциональным или числовым параметром. Конкретизация функциональных параметров определяет *подкласс АВО*, а число-

вых – конкретный *алгоритм в рамках этого подкласса*. Одной из причин появления АВО было желание авторов упорядочить существовавшее в то время множество алгоритмов распознавания, введя некий «метаалгоритм» распознавания, с использованием которого можно «точно» или «приближенно» представить все существующие.

В основе алгоритмов АВО лежит естественный эвристический принцип, которым пользуется человек – это *принцип прецендентности* или *частичной прецендентности*, в соответствии с которым в похожих ситуациях следует действовать аналогичным образом. Принцип действия АВО состоит в вычислении нескольких *оценок сходства*, которые характеризуют близость распознаваемого и эталонного объектов по некоторым подмножествам признаков. Система подмножеств признаков формируются из всего множества признаков, доступных наблюдению. При этом подмножества не обязаны быть непересекающимися.

Исходная (априорная) информация, доступная на этапе обучения, в АВО представляется в виде таблицы следующего вида

Таблица 5.1. **Представление информации в АВО**

Классы	Объекты	Признаки						
		x_0	x_1	x_2	x_3	x_4	x_5	x_6
Ω_0	ω_0							
	ω_1							
Ω_{L-1}	ω_2							
	ω_3 … ω_{N-1}							

Каждый из признаков может принимать значения из различных множеств, например:

$\{0,1\}$, 0 – признак не выражен, 1 – признак выражен;

$\{0,1,e\}$, где e – информация о признаке отсутствует;

$\{0,1,2,\ldots,d\}$ – степень выраженности признака имеет различные градации;

$\{a,b\}$ – признак принимает значения из заданного диапазона.

$\{S_j\}_{j=0}^{M-1}$ – это система опорных множеств (подмножеств) признаков, которые в совокупности могут характеризовать ситуацию (состояние) объекта. Наиболее общей ситуацией является случай, когда рассматривается множество всех подмножеств признаков. В этом случае $M = C_n^1 + C_n^2 + \ldots + C_n^n \le 2^n$, где n – размерность пространства признаков (их количество).

Подобное выделение системы подмножеств признаков не случайно – оно обусловлено тем, что основная «различающая» информация может лежать не в каком-то одном признаке, а в их совокупности. Необходимость использование системы опорных множеств признаков можно проиллюстрировать следующим примером: судить о том, имеет ли человек избыточный вес невозможно только по его весу – необходимо знать также и его рост. Эти два признака могут определять одно из опорных подмножеств в системе медицинской диагностики.

Далее, АВО использует эти оценки сходства для формирования единой оценки, которая характеризует «близость» искомого объекта к тому или иному классу.

На последнем шаге оценки «близости» для разных классов сравниваются и принимается окончательное решение.

Графически схема принятия решения в АВО можно (условно) представить так, как показано на рисунке 5.1.

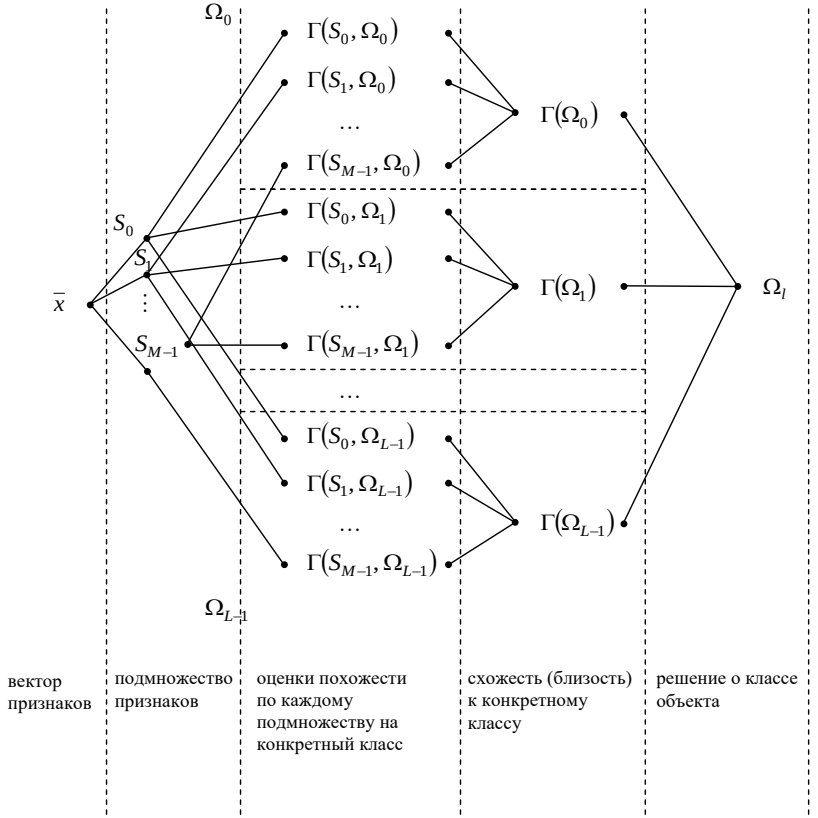


Рисунок 5.1 – Схема принятия решения в АВО

Конкретизация модели АВО заключается в конкретизации каждого из этапов на схеме. А именно:

1. Выделение системы подмножеств признаков, которые называют *системой опорных множеств признаков.*

2. Введение функции близости/схожести d_{ij} по i-му признаку на подмножества S_j (системы опорных множеств), вычисляемого между эталонным и искомым объектом.

3. Задание правил:

a) вычисления предикатов f_j «похож» – «не похож» между эталонным и искомым объектом по значениям функции близости d_{ij};

b) вычисления оценки $\Gamma\left(S_j, \Omega_l\right)$ схожести искомого объекта с классом по опорному подмножеству на основании предикатов f_i «похож» – «не похож»;

c) формирования итоговой оценки схожести $\Gamma\left(\Omega_l\right)$ между объектом и классом Ω_l по оценкам $\Gamma\left(S_j, \Omega_l\right)$;

d) принятия решения отнесения объекта к классу на основании множества оценок $\Gamma\left(\Omega_l\right)$ ($l=\overline{0, L-1}$) – решающего правила.

Фиксация способа выбора системы опорных множеств, типа функции близости, правил вычисления оценок и решающего правила определяет выбор *подкласса АВО*, а задание значений соответствующих параметров – *конкретный алгоритм АВО*. Очевидно, что имеет место взаимнооднозначное соответствие между конкретным алгоритмом АВО из конкретного подкласса и набором его числовых параметров. Поэтому, если мы фиксируем подкласс алгоритмов АВО, то конкретному значению параметров алгоритма можно сопоставить значения некоторого функционала качества распознавания. Значит можно менять параметры алгоритма таким образом, чтобы значение функционала качества изменялось в нужном направлении. И в тех случаях, когда наилучший набор параметров может быть найден, имеется гарантия что на данном обучающем множестве множества объектов в данном подклассе алгоритмов типа АВО для заданного контрольного материала не существует лучшего алгоритма распознавания, чем найденный. Факт связи (взаимнооднозначной) между алгоритмом и его пара-

метрами позволил авторам данного алгебраического подхода (Журавлев Ю.И. и Гуревич И.Б., 70-е гг.) говорить о *синтезе оптимального алгоритма в классе алгоритмов вычисления оценок*.

В рамках настоящего пособия мы не будем касаться вопросов синтеза оптимального алгоритма, поскольку в общем случае это сложная вычислительная задача. Вместо этого рассмотрим примеры конкретных алгоритмов АВО и механизм формирования в них решения.

Пример.

Пусть информация о задаче распознавания задана в виде таблицы 5.2.

Пусть необходимо проклассифицировать вектор $\bar{x}$ для объекта ω.

Таблица 5.2. **Пример информации о задаче распознавания**

Классы	Объекты	x_0	x_1	x_2	x_3	x_4	x_5	x_6
Ω_0	ω_0	0	0	1	0	0	1	
	ω_1	1	0	1	1	0	0	
	ω_2	1	0	0	0	1	0	
Ω_1	ω_3	0	1	1	1	0	0	
	ω_4	0	1	0	1	0	0	
	ω_5	1	0	1	1	1	0	
Тестовый объект	ω	1	0	1	1	0	0	

Поскольку АВО – это модель, то первое что необходимо сделать, это *указать подкласс АВО* в котором мы будем далее работать. Для этого необходимо указать функциональные параметры

модели АВО. Например, это можно сделать так, как показано в следующей таблице 5.3.

Таблица 5.3. **Конкретизация подкласса АВО**

Элемент модели	Конкретизация
Система опорных множеств	$S_0=\{x_0,x_1\}$, $S_1=\{x_2,x_3\}$, $S_2=\{x_3,x_5\}$
Тип функции близости $d_{ij}(\bar{x},\bar{a})$ на подмножестве признаков S_j	$d_{ij}(\bar{x},\bar{a})=\lvert x_i-a_i\rvert$, где a_i – значение *i*-го признака из множества S_j у эталонного объекта, x_i – значение того же признака у искомого объекта
Правило вычисления предиката $f_j(\bar{x},\bar{a})$ «похож» – «не похож» по системе S_j	$f_j(\bar{x},\bar{a})=\begin{cases}1, \ \forall\, x_i\in S_j \ \ d_{ij}(\bar{x},\bar{a})\le\varepsilon_{ij}, \ ; \\ 0, \ \textit{иначе.}\end{cases}$ ε_{ij} – числовые параметры алгоритма
Вычисление оценки схожести $\Gamma(S_j,\Omega_l)$ на основании признаков «похож» – «не похож»	$\Gamma(S_j,\Omega_l)=\sum\limits_{\bar{a}\in\Omega_l} f_j(\bar{x},\bar{a})$ (число похожих эталонов)
Формирование оценки $\Gamma(\Omega_l)$ из оценок $\Gamma(S_j,\Omega_l)$	$\Gamma(\Omega_l)=\sum\limits_{j=0}^{\Im-1}\alpha_{jl}\Gamma(S_j,\Omega_l)$, α_{jl} – числовые параметры алгоритма
Решающее правило	Номер класса (правило максимального голосования): $l^*=\arg\max\limits_{l=0,L-1}\Gamma(\Omega_l)$

Таким образом, подкласс алгоритмов АВО задан. Теперь осталось указать конкретные числовые параметры для того, чтобы определить алгоритм распознавания в этом подклассе окончатель-

но. В качестве параметров здесь выступают величины ε_{ij} и α_j ($j=\overline{0,\Im-1}$). Зададим их следующими:

$$\varepsilon_{ij}=0\,,\ \alpha_j=1\,,$$

т.е. все опорные множества имеют одинаковый вес, а сравнение признаков ведется по совпадению.

Тогда получаем следующее решение задачи распознавания:

$$\begin{array}{ccc}
\Gamma(S_0,\Omega_0)=2 & & \Gamma(S_0,\Omega_1)=1 \\
+ & & + \\
\Gamma(S_1,\Omega_0)=1 & & \Gamma(S_1,\Omega_1)=2 \\
+ & & + \\
\underline{\Gamma(S_2,\Omega_0)=1} & & \underline{\Gamma(S_2,\Omega_1)=3} \\
\Gamma(\Omega_0)=4 & & \Gamma(\Omega_1)=6
\end{array}$$

$$\Downarrow$$

$$\omega\in\Omega_1!$$

Если мы по-другому зададим числовые или функциональные параметры, то можем получить совершенно другое решение задачи распознавания! Например, определим функциональные элементы модели АВО, как показано в таблице 5.4.

В этом случае результирующий алгоритм типа АВО фактически является аналогом («аппроксимацией») метода ближайшего соседа, причем параметр Δ определяет «разрешающую» способность этого алгоритма.

Продемонстрированная гибкость АВО, заключающаяся в возможности представлять или «аппроксимировать» работу других алгоритмов распознавания, оказывается и его силой, и его слабостью: заранее часто не известно, как строить подкласс АВО для конкретной задачи.

Таблица 5.4. **Конкретизация подкласса АВО. Вариант 2**

Элемент модели	Конкретизация
Система опорных множеств	$S_0=\{x_0,\ldots,x_{n-1}\}$, $S_1=S_0$, $S_2=S_0$, …
Тип функции близости $d_{ij}(\bar{x},\bar{a})$ на подмножестве признаков S_j	$d_{ij}(\bar{x},\bar{a})=(x_i-a_i)^2$, где a_i – значение i-го признака из множества S_j у эталонного объекта, x_i – значение того же признака у искомого объекта
Правило вычисления предиката $f_j(\bar{x},\bar{a})$ «похож» – «не похож» по системе S_j	$f_j(\bar{x},\bar{a})=\begin{cases}\sqrt{\sum_{i=0}^{n-1} d_{ij}(\bar{x},\bar{a})}\le j\Delta; \\ 0, \textit{ иначе.}\end{cases}$ Δ – параметр
Вычисление оценки схожести $\Gamma(S_j,\Omega_l)$ на основании признаков «похож» – «не похож»	$\Gamma(S_j,\Omega_l)=\sum_{\bar{a}\in\Omega_l} f_j(\bar{x},\bar{a})$ (число похожих эталонов)
Формирование оценки $\Gamma(\Omega_l)$ из оценок $\Gamma(S_j,\Omega_l)$	$\Gamma(\Omega_l)=\arg\min_{j=0,1\ldots}\{\Gamma(S_j,\Omega_l)\neq 0\}$ Т.е. $\Gamma(\Omega_l)$ содержит номер j, для которого «шар» соответствующего радиуса не пуст
Решающее правило	Номер класса (правило минимального голосования): $l^*=\arg\min_{l=0,L-1}\Gamma(\Omega_l)$

Поэтому наряду с очевидными достоинствами АВО, к которым относятся:

– гибкость;

– возможность описания или «аппроксимации» других алгоритмов распознавания как подкласса АВО;

у АВО есть и очевидные недостатки:

– не известно как выбирать систему опорных множеств, системы сравнения, предикаты и т.п., то есть как для конкретной практической задачи конкретизировать подкасс АВО;

– вычислительная сложность параметрической оптимизации АВО в выбранном подклассе алгоритмов. Как указывали авторы АВО, некоторые вопросы вычислительной сложности могут быть решены за счет выбора системы опорных множеств [45].

5.2 Алгебраическая теория алгоритмов распознавания

Появление АВО характеризует определенный этап развития алгебраической теории распознавания для советской школы. На этом этапе (70-е г.) производилась попытка строить единообразное описание для достаточно богатого множества эвристических («некорректных» – неформализованных) алгоритмов распознавания, которые уже существовали.

Как отметил Ю.И. Журавлев в своей монографии, «Каждое такое описание ограничивало множество алгоритмов некоторыми рамками и по сути задавало *модель распознающего алгоритма*. А введение понятия модели распознавания алгоритма позволило изучать множество «некорректных» (эвристических) процедур распознавания с помощью строгих математических методов. В частности это позволило, как видно на примере АВО, ставить и решать в рамках модели оптимизационную задачу выбора алгоритма.

Следующим этапом развития алгебраической теории распознавания советской школы является выход за рамки моделей алго-

ритмов. А именно, анализ множества некорректных алгоритмов распознавания позволил описывать не только отдельные частные алгоритмы, но и принципы их формирования. Эти принципы, действующие уже над множествами алгоритмов, как оказывается, могут быть реализованы в виде точных математических описаний. В результате оказалось возможным сформулировать общее определение алгоритма распознавания и изучить свойства множеств таких алгоритмов. Оказалось, что *множество алгоритмов распознавания является алгеброй*, причем операции этой алгебры обладают свойствами, позволяющими детально изучать множества этих алгоритмов» [45]. Ниже даны основные определения и положения предложенной Ю.И. Журавлевым алгебраической теории распознавания, по возможности сохранены оригинальные обозначения.

5.2.1 Постановка задачи и основные определения

Пусть дано множество Ω объектов, подлежащих распознаванию и называемых в дальнейшем *допустимыми объектами*. Считаем, что множество допустимых объектов покрыто конечным числом подмножеств $\Omega_0,\ldots,\Omega_{L-1}$ и $\bigcup_{l=0}^{L-1}\Omega_l=\Omega$. Подмножества Ω_l ($l=\overline{0,L-1}$), называемые *классами*, могут пересекаться:

$$\Omega_l\bigcap\Omega_j\neq 0 \quad l\neq j\ ,$$

но не допускается «совпадающих» классов:

$$l\neq j\Rightarrow\Omega_l\neq\Omega_j\ .$$

Далее, каждый допустимый объект $\omega\in\Omega$ определен значениями своих *характеристик (признаков)* $\bar{x}(\omega)$, обозначаемые в оригинальных работах как $I(\omega)=\bar{x}(\omega)$.

Определение 5.1 [50]. Алгоритм *A* называется *распознающим*, если он переводит информацию об объектах $I_N \equiv \left\{ I(\omega_i) \right\}_{i=0}^{N-1}$ в матрицу $\left\| \alpha_{il}^A \right\|_{N \times L}$, называемую *информационной и* составленную из элементов $\{0, 1, \Delta\}$, которые интерпретируются следующим образом:

если $\alpha_{il} = 1$, то считается, что $\omega_i \in \Omega_l$;

$\alpha_{il} = 0$, то считается что $\omega_i \notin \Omega_l$;

$\alpha_{il} = \Delta$, то считается, что алгоритм не может определить класс объекта.

Обозначим далее $\left\| \alpha_{il}^0 \right\|_{N \times L}$ – требуемую классификацию множества допустимых объектов.

Определение 5.2 [45]. Алгоритм *A* называется *корректным*, если выполняется равенство

$$\left\| \alpha_{il}^A \right\|_{N \times L} = \left\| \alpha_{il}^0 \right\|_{N \times L}$$

Алгоритм, не являющийся корректным, называется *некорректным*. Таким образом, под корректным алгоритмом понимается алгоритм, безошибочно распознающий объекты выборки.

Конечной целью алгебраической теории распознавания Ю.И. Журавлева являлось построение корректного алгоритма распознавания над множеством некорректных (эвристических) алгоритмов. Важным этапом построения является следующая теорема, которая приводится с доказательством, так как в ней указан конструктивный способ построения оператора *C* (см. доказательство ниже).

Теорема 5.1 [45]. Каждый распознающий алгоритм представим как последовательное выполнение операторов *B* и *C*: $A = B \cdot C$, где

$$B(I_N)=\left\|a_{il}^{A}\right\|_{N\times L},\quad a_{il}^{A}\in\mathbf{R},$$

$$C\left(\left\|a_{il}^{A}\right\|_{N\times L}\right)=\left\|\alpha_{il}^{A}\right\|_{N\times L},\quad \alpha_{il}^{A}\in\{0,1,\Delta\}.$$

Доказательство [45]: Пусть D – алгоритм перехода матрицы $\|\alpha_{il}\|\xrightarrow{k}\|a_{il}\|$. В качестве D можно рассмотреть, например следующий алгоритм:

$$a_{il}=\alpha_{il}\ ,\ \text{если}\ \ \alpha_{il}\in\{0,1\},$$

$$a_{il}=1/2\ ,\ \text{если}\ \ \alpha_{il}=\Delta\ .$$

Обозначим через D^{-1} обратный алгоритм перехода от $\|a_{il}\|\to\|\alpha_{il}\|$. Положим

$$B=A\cdot D\ \ (\text{действует } A, \text{ затем } D),$$

$$C=D^{-1}\ .$$

Тогда:

$$A=B\cdot C=(A\cdot D)\cdot D^{-1}=A\,.$$

Из теоремы следует, что множество алгоритмов распознавания **A** порождает два множества: **B** и **C**. Элементы множества **B** называются *алгоритмическими операторами* или просто *операторами*, а множество **C** -- *решающими правилами*. Таким образом, *алгоритмическим (или распознающим) оператором* называется оператор *B*, если он преобразует информацию о допустимых объектов в числовую матрицу $\{a_{il}\}_{N\times L}$. А *решающим правилом* является оператор, который преобразует эту числовую матрицу

в информационную матрицу, т.е. выполняет преобразование $\|a_{il}\| \to \|\alpha_{il}\|$.

Для разных типов алгоритмов распознавания элементы матрицы $\{a_{il}\}_{N\times L}$ имеют разный смысл. В частности:

– в статистической теории распознавания это матрица апостериорных вероятностей,

– в геометрической теории это матрица расстояний до (центра) класса,

– в АВО это матрица оценок принадлежности классам,

– в теории нечетких множеств это матрица со значениями функции принадлежности и т.п.

В качестве распознающего оператора может выступать любой оператор, который попадает под определение, данное выше. Однако, решающие правила не могут быть произвольными. Из всего возможного множества решающих правил в алгебраической теории рассматриваются только корректные решающие правила, определение которым дается ниже.

Определение 5.3 [45]. Решающее правило C называется *корректным* на Ω, если для каждого конечного набора допустимых объектов $\omega_0, \ldots, \omega_{N-1}$ из Ω существует хотя бы одна числовая матрица $\|a_{il}\|_{N\times L}$ такая, что $C\left(\|a_{il}\|_{N\times L}\right) = \|\alpha_{il}\|_{N\times L}$, где $\|\alpha_{il}\|_{N\times L}$ – заданная информационная матрица множества объектов $\{\omega_i\}_{L=\overline{0,\,N-1}}$.

Пример. Корректным является следующее решающее правило C^*:

$$\alpha_{il} = \begin{cases} 1, & \text{если} \quad a_{il} > a^{\max}, \\ 0, & \text{если} \quad a_{il} < a^{\min}, \\ \Delta, & \text{если} \quad a^{\min} \le a_{il} < a^{\max}. \end{cases}$$

Подведем некий итог тому, что мы на данный момент имеем. Имеем мы формализм, в соответствии с которым каждый алгоритм распознавания A для задачи с N допустимыми объектами ассоциирован со своей информационной матрицей $\left\|\alpha_{il}^{A}\right\|_{N\times L}$. Это первый важный факт.

Второй важный факт –каждый алгоритм A ассоциирован также с матрицей числовой $\left\|a_{il}^{A}\right\|_{N\times L}$, которую порождает распознающий оператор B алгоритма A, а также неким корректным решающим правилом C^{*}.

Другими словами, каждый алгоритм может нами быть ассоциирован с точкой в некотором пространстве, координаты которой определяются одной из матриц. Вид пространства зависит от того, какой именно формализм описания алгоритма распознавания мы выбираем. Если мы выбираем формализм «алгоритм» = «информационная матрица», то тогда алгоритм A ассоциирован с точкой в пространстве $\mathbf{B}_{\Delta}^{N\times L}$ $\left(\mathbf{B}_{\Delta}=\{0,1,\Delta\}\right)$. Если же мы принимаем формализм «алгоритм» = «распознающий оператор»·+ «решающее правило», то алгоритм оказывается ассоциирован с точкой в пространстве $\mathbf{R}^{N\times L}$.

Идея алгебраического подхода Ю.И. Журавлева состоит в том, чтобы либо в рамках первого, либо второго формализма определить операции над множеством алгоритмов $\mathbf{A}$, т.е. над множеством точек/векторов в соответствующих пространствах, которые позволили бы *представить корректный алгоритм в виде функции от некорректных*, т.е. построить

$$A^{*}=F(\mathbf{A}),$$

где $\mathbf{A}$ – заданное нам множество алгоритмов, *базис*. Другими словами, необходимо получить заданную точку/вектор в пространстве (числовая/информационная матрица корректного алгоритма) с помощью множества известных точек/векторов (числовые/информационные матрицы некорректных алгоритмов из базиса). Операция $F(\ )$, определенная на множестве алгоритмов из $\mathbf{A}$, называется *корректирующей операцией*, а множество алгоритмов, которое можно получить из $\mathbf{A}$ путем применения операции $F(\ )$ – *замыканием множества* $\mathbf{A}$ и обозначается $[\mathbf{A}]$.

В зависимости от того, какого вида (линейного, полиномиального, произвольного) используется корректирующая операция *F*, получаем различные типы замыканий $[\mathbf{A}]$, если *F:*

- линейная, то получаем *линейное замыкание*;
- полиномиальная, то получаем *алгебраическое замыкание*;
- произвольная, то получаем *функциональное замыкание*.

Способ замыкания зависит от выбранного формализма и свойств пространства, в котором строим $[\mathbf{A}]$. Так, в силу специфики элементов множества $\mathbf{B}_\Delta$ на них плохо вводить попарные операции («+», «*»). Поэтому линейное и алгебраическое замыкания строятся на множестве вещественных матриц $\left\|a_{il}\right\|_{N\times L}$ распознающих операторов. Функциональное замыкание, учитывающее и порядок операторов при вызове $F(A_0,\ldots,A_{M-1})$ может производиться и для формализма «Алгоритм»-»Матрица Ответов».

Ниже кратко описан первый подход, использующий формализм с представлением алгоритма распознавания $A=B\cdot C$, где Журавлевым Ю.И. и его коллегами в рамках алгебраического подхода были получены основные теоретические результаты. Иллюстрация идеи

построения замыкания и корректного алгоритма над множеством алгоритмов распознавания представлена на рисунке 5.2.

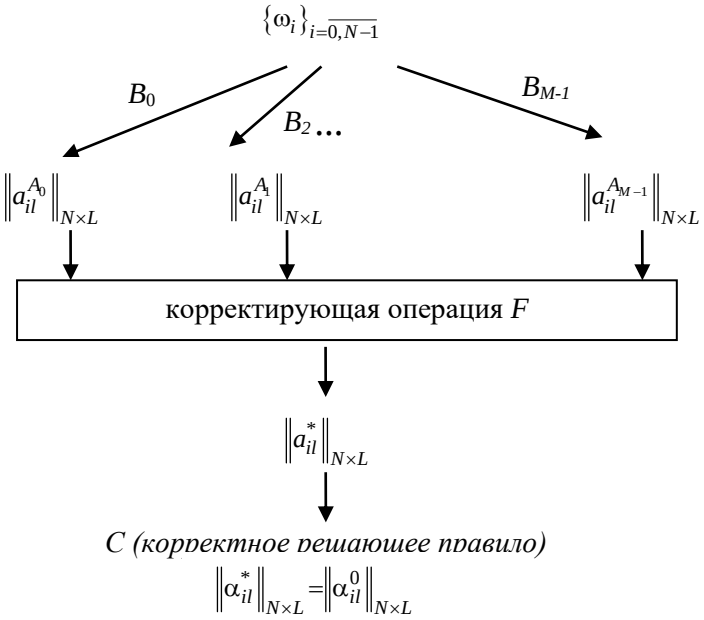


Рисунок 5.2 – Иллюстрация к построению корректного алгоритма над множеством некорректных (эвристических) алгоритмов распознавания

5.2.2. Алгебра над множеством распознающих операторов и некорректных алгоритмов

На множестве распознающих операторов $\mathbf{B}$, которое порождается множеством алгоритмов $\mathbf{A}$, можно легко ввести операции сложения, умножения и умножения на скаляр. А именно, пусть $B_1, B_2 \in \mathbf{B}$ и

$$B_1(I_N)=\left\|a_{il}^1\right\|_{N\times L},\ B_2(I_N)=\left\|a_{il}^2\right\|_{N\times L}.$$

Тогда операторы «+», «*» и «*c» определяются следующим образом:

$$(c\cdot B_1)(I_N)=\left\|c\cdot a_{il}^1\right\|_{N\times L}, \tag{5.1}$$

$$(B_1+B_2)(I_N)=\left\|a_{il}^1+a_{il}^2\right\|_{N\times L}, \tag{5.2}$$

$$(B_1\cdot B_2)(I_N)=\left\|a_{il}^1\cdot a_{il}^2\right\|_{N\times L}. \tag{5.3}$$

Вначале рассмотрим замыкание $[\mathbf{B}]$ относительно операций (5.1) и (5.2). Элементами этого замыкания являются распознающие операторы вида

$$\sum_j c_j B_j \text{ , где } B_j\in\mathbf{B}. \tag{5.4}$$

Утверждение [45]. По операциям сложения (5.2) и умножения на скаляр (5.1) множество распознающих операторов образует *линейное векторное пространство*.

Замыкание вида (5.4) называется *линейным замыканием* (линейной оболочкой) множества **B** распознающих операторов (в общем случае – произвольного множества векторов).

Замыкание множества **B** относительно всех трех операций (5.1)- (5.3) является ассоциативной линейной алгеброй с коммутативным умножением. Элементами этого замыкания являются распознающие операторы вида (*операторные многочлены*)

$$\sum_{k=0}^{K}\ \sum_{\bar{k}\in\left\{(k_0,k_1,\ldots,k_{M-1}):\ \sum_{j=0}^{M-1}k_j=k\right\}} d_{k\bar{k}}\prod_{m=0}^{M-1}(B_m)^{k_m}, \tag{5.5}$$

здесь K – степень замыкания или *степень операторного многочлена*. Замыкание вида (5.5) называют алгебраическим замыканием множества $\mathbf{B}$ распознающих операторов

Ключевым результатом построенной академиком Ю.И. Журавлевым алгебры над множеством распознающих операторов и некорректных алгоритмов является формализованный пусть построения корректного алгоритма, позволяющий свести задачу распознавания к задаче, фактически, решения систем линейных алгебраических уравнений с возможными дополнительными ограничениями. Основные шаги решения следующие:

1. Формируется множество распознающих операторов $\mathbf{B}$, порождаемых множеством некорректных алгоритмов распознавания $\mathbf{A}$, то есть, по сути, матриц $\left\{a_{il}^{A}\right\}_{N\times L}$ $\left(A\in\mathbf{A}\right)$;

2. Строим линейный функционал (5.4) или полином (5.5) над этими матрицами, как над переменными;

3. Присоединяем произвольное корректное решающее правило C^{*}.

Вопрос нахождения числовых параметров линейного или алгебраического замыкания, как и вопрос о структуре полинома, в рамках настоящего пособия не рассматривается. Читателю рекомендуется ознакомиться с оригинальными работами [45-46].

5.3 Результаты пятого раздела

Данные раздел содержит краткое описание двух подходов к построению алгоритмов распознавания, разработанных российской школой распознавания под руководством *академика Ю.И. Журавлева*.

Первое решение – это класс алгоритмов вычисления оценок (АВО). Суть этого подхода – это работа с некоторой моделью алгоритма, в рамки которой укладываются (точно или приближенно) большинство существующих алгоритмов. Этот класс алгоритмов определен с точностью до функциональных и числовых параметров. Фиксирую функциональные параметры мы приходим к подклассу АВО, а фиксация числовых дает нам конкретный алгоритм. При разработке «оптимального» способа нахождения числовых параметров в некотором подклассе АВО можно говорить о синтезе оптимального алгоритма в соответствующем подклассе.

Второе рассмотренное решение – это алгебра алгоритмов распознавания. Суть предложенного Ю.И. Журавлевым решения – это найти способ использования произвольного множества алгоритмов (в терминах подхода – некорректных) для построения *корректного алгоритма*. Под корректным алгоритмом понимается алгоритм, который безошибочно классифицирует обучающую/контрольную выборки. Идея построений заключается в ассоциировании каждого некорректного алгоритма с матрицей ответов этого алгоритма на примерах и построении агрегирующего функционала (линейного – линейное замыкание множества алгоритмов, полиномиальное – алгебраическое замыкание множества алгоритмов), в результате применения которого к матрицам можно получить требуемую матрицу ответов. Агрегирующий функционал с исходным множеством алгоритмов и будет новым алгоритмом, а идейный подход назван Ю.И. Журавлевым *алгеброй над алгоритмами*.

6 МЕТОДЫ СОВМЕСТНОЙ КЛАССИФИКАЦИИ

6.1 Совместная классификация и основные стратегии

Основной целью проектирования любой системы классификации является построение такого решающего правила, которое обеспечивало бы максимально возможное качество распознавания. Именно поэтому интенсивно развиваться математические методы и алгоритмы построения решающих правил с повышенной структурной сложностью, то есть методов *совместной классификации*, называемых также *мультиклассификацией* [59–66].

Идея совместной классификации достаточна проста – использовать результаты распознавания сразу нескольких классификаторов для улучшения качественных показателей всей *композиции классификаторов* в целом [45]. Получаемое совместное решение должно оказаться не хуже индивидуальных решений, которые принимаются каждым из используемых классификаторов, называемых в ряде работ *экспертами* (experts) [67]. Последние в некоторых задачах (например, бустинге – см. подраздел 6.4) называю также «слабыми» классификаторами.

Поиски путей нахождения наилучшей схемы совместной классификации привели к появлению целого ряда различных способов ее построения. В то же время стратегий совместной классификации, как показывает анализ этих решений, относительно немного. Выделим их.

С точки зрения *реализации процесса совместной классификации* стратегии совместной классификации можно разделить на *последовательные* и *параллельные*.

В рамках *последовательной стратегии совместной классификации* вектор(а) признаков подвергаются последовательному анализу каждым из входящих в композицию экспертов. В случае, если данный эксперт достаточно «эрудирован», ответ о принадлежности к конкретному классу выносится немедленно. В противном случае анализ данных осуществляется следующим экспертом. Задачей настройки последовательной схемы классификации является нахождение параметров всех классификаторов и правила их взаимодействия. Число экспертов в такой схеме принятия решения может быть различным: от двух классификаторов до неограниченного их количества.

Надо отметить, что появление и развитие последовательной стратегии классификации во многом было обусловлено существованием хорошо развитой *статистической теорией последовательного анализа гипотез* [44, 68–71], которая рассмотрена, в частности в четвертом разделе настоящего пособия.

В рамках *параллельной стратегии совместной классификации* вектор(а) признаков подвергаются одновременному анализу несколькими экспертами. Каждый их них выносит свое решение о принадлежности анализируемых данных к тому или иному классу. В этом случае задачей настройки параллельной схемы классификации является задача поиска функции «агрегирования» решений различных экспертов, а также поиск оптимальных параметров классификации каждого эксперта в составе общей схемы. Иллюстрация параллельной стратегии совместной классификации приведена на рисунке 6.1.

Заметим, что при параллельной стратегии возможно различное представление *информации о решениях экспертов*. Наиболее типичными вариантами решений экспертов являются:

– *номер класса*, к которому принадлежат классифицируемые данные по мнению эксперта. Такой вариант будем рассматривать

как *минимальную информацию о решении*. Одним из наиболее известных современных решений построения композиции алгоритмов с минимальной информацией о решениях экспертов является *метод бустинга*, представленный в подразделе 6.4 настоящего пособия;

– *числовые величины*, которые характеризуют оценки (например, апостериорные вероятности) принадлежности классифицируемых данных к каждому классу (максимальный объем информации). Такой вариант будем рассматривать как *максимальную информацию о решении.* В России теория построения композиции алгоритмов с максимальной информацией о решениях экспертов была развита академиком Ю.И. Журавлевым в 70-х года и получила название *алгебры алгоритмов распознавания*. Краткое описание этой теории дано в пятом разделе настоящего пособия.

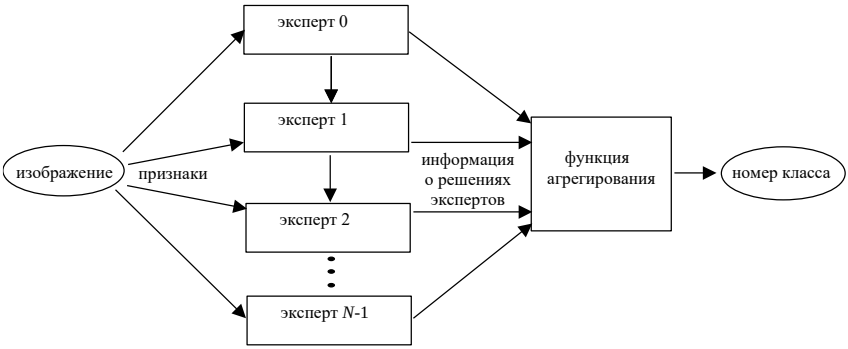


Рисунок 6.1 – Схема параллельной стратегии
совместной классификации

Появление параллельных стратегий классификации вначале было обусловлено желанием разработчиков повысить качество систем распознавания. В последнее время параллельная стратегия

классификации бурным развитием обязана в первую очередь возросшим потенциалом вычислительной техники, в особенности компьютерам и специализированным вычислительным средствам с параллельной архитектурой. Использование многопроцессорных вычислительных систем для решения задачи распознавания позволяет естественным образом разделить процедуру классификации на ряд параллельно выполняемых процессов, каждый из которых реализует процесс принятия решения некоторого конкретного эксперта. Подобная технология в настоящие время эффективно используется, например, в нейронных сетях [61–62].

Заметим также, что поиск функции агрегирования в первое время проводился эвристическими методами, часто использовались аналогии с коллективными методами принятия решений, которые характерны для людей. Наиболее ярким примером такой функции агрегирования является классификация по правилу «максимальное голосования». В соответствии с этим правилом решение о принадлежности к некоторому классу принималось, если за этот класс «проголосовало» наибольшее число экспертов. В случае учета «веса» экспертов указанное правило носит названия «взвешенного максимального голосования». Оба эти правила для задачи бинарной (двуклассовой) классификации можно представить следующим образом:

– *правило максимального голосования*:

$$C(\bar{x}) \equiv \sum_{m=0}^{M-1} d_m(\bar{x}), \tag{6.1}$$

– *правило взвешенного максимального голосования:*

$$C(\bar{x}) \equiv \sum_{m=0}^{M-1} \alpha_m d_m(\bar{x}). \tag{6.2}$$

В представленных выражениях $d_m(\bar{x}) \in \{-1,1\}$ – решающее правило m-го эксперта, $C(\bar{x})$ – правило мультиклассификатора, $\alpha_m \in \mathbf{R}$ $\left(m = \overline{0,M-1}\right)$ – коэффициенты композиции.

В рамках данного раздела представлены некоторые методы и алгоритмы совместной классификации. В частности, в подразделе 6.2 показано, как использовать байесовский классификатор для дискретных векторов признаков для настройки композиции с минимальной информацией о решениях экспертов. В подразделе 6.3 рассмотрена последовательная стратегия классификации, выполняющая предварительное обнаружение и последующее распознавание искомого объекта. Представленное в этом подразделе решение может легко быть обобщено на параллельную стратегию, выступив как альтернатива методу бустинга, который рассмотрен в подразделе 6.4. Наконец, в подразделе 6.5 представлены наиболее известные способы обучения отдельных экспертов для использования их в правилах голосования (6.1)–(6.2), позволяющие избежать типичных для композиции классификаторов проблем.

6.2. Параллельная схема совместной классификации с минимальной информацией о решениях экспертов

Рассмотрим ситуацию, когда информация от эксперта поступает в виде номера класса. В этом случае не требуется никаких дополнительных предположений о способе представления образа и виде решающей функции. Вначале дадим формальную постановку задачи совместной классификации, а затем рассмотрим возможные методы ее решения.

6.2.1 Формальная постановка задачи

Пусть все множество объектов разделено на L классов Ω_l, $l=\overline{0,L-1}$. Пусть для классификации использовать N классификаторов-экспертов, каждый из которых выносит свое решение Q_n $\left(n=\overline{0,N-1}\right)$ относительно принадлежности конкретного объекта ω некоторому классу. Для определенности будем считать, что n-ый классификатор относит анализируемый объект к l-ому классу, если $Q_n=l$. Таким образом множество возможных решений каждого из классификаторов $\left\{l:l=\overline{0,L-1}\right\}$, и на решение Q_n можно смотреть как на дискретную случайную величину с некоторым законом распределения. В этом случае на вход мультиклассификатора (функции агрегирования) подается случайный вектор $\mathbf{Q}=\left(Q_0,\ldots,Q_{N-1}\right)^T$, в качестве компонент содержащий мнения экспертов. Тогда решением задачи настройки мультиклассификатора является нахождение функции или отображения, которое по заданному конкретному вектору решений $\mathbf{q}=\left(q_0,\ldots,q_{N-1}\right)^T$ определяет, к какому именно классу Ω_l принадлежит соответствующий этому вектору объект ω.

Очевидно, постановка задачи таким образом приводит нас к стандартной байесовской процедуре классификации дискретного случайного вектора (см. пособие [1-3]).

6.2.2 Байесовская процедура совместной классификации

Пусть на всем множестве возможных векторов (возможных решений экспертов) $q=\left(q_0,\ldots,q_{N-1}\right)^T$ для каждого класса Ω_l ($l=\overline{0,L-1}$) определена его функция правдоподобия $p\left(\mathbf{Q}=\mathbf{q}/\Omega_l\right)$.

Тогда оптимальным решением задачи классификации, как известно, является правило максимума апостериорной вероятности[1]:

объект относят к тому классу Ω_l , для которого

$$P(\Omega_l/\mathbf{Q}=\mathbf{q})=\max_{j=0,L-1} P(\Omega_j/\mathbf{Q}=\mathbf{q}). \qquad (6.3)$$

Здесь апостериорные вероятности $P(\Omega_j/\mathbf{Q}=\mathbf{q})$ определяется по формуле Байеса:

$$P(\Omega_j/\mathbf{Q}=\mathbf{q})=\frac{P(\Omega_j)P(\mathbf{Q}=\mathbf{q}/\Omega_j)}{\sum_{i=0}^{L-1} P(\Omega_i)P(\mathbf{Q}=\mathbf{q}/\Omega_i)}.$$

На практике процедура построения байесовского агрегирующего решающего правила сводится к получению для каждого класса Ω_l $\left(l=\overline{0,L-1}\right)$ таблицы размера L^N, которая задает распределение вероятностей случайного вектора $\mathbf{Q}$: $P(\mathbf{Q}=\mathbf{q}/\Omega_l)$. Таким образом необходимо произвести оценку L^{N+1} вероятностей $P(\mathbf{Q}=\mathbf{q}/\Omega_l)$ $\left(l=\overline{0,L-1}\right)$. При реализации процедуры распознавания достаточно одной таблицы размером L^N, которая содержит для каждого возможного вектора $\mathbf{q}$ соответствующий номер класса, получаемый заранее в соответствии с формулой (6.3). В таблице 6.1 приведены ее размеры.

Для реализации байесовской процедуры совместной классификации в режиме реального времени (например, при автоматическом чтении текста), занимаемый соответствующей таблицей объем данных должен помещаться в оперативную память компьютера,

[1] Здесь и далее в разделе матрица штрафов предполагается простейшей.

что означает практическую невозможность использования этого решения для тех параметров (N, L) таблицы, которые выделены серым цветом.

Таблица 6.1. **Размер таблицы байесовской процедуры совместной классификации**

		число классов L							
		2	4	8	16	32	64	128	256
число экспертов N	2	2^2	2^4	2^6	2^8	2^{10}	2^{12}	2^{14}	2^{16}
	4	2^4	2^8	2^{12}	2^{16}	2^{20}	2^{24}	2^{28}	2^{32}
	8	2^8	2^{16}	2^{24}	2^{32}	2^{40}	2^{48}	2^{56}	2^{64}
	16	2^{16}	2^{32}	2^{48}	2^{64}	…	…	…	…
	32	2^{32}	2^{64}	…	…	…	…	…	…

Кроме того, при настройке такого классификатора возникают дополнительные ограничения, связанные с устойчивостью получаемых оценок вероятностей соответствующих законов распределения. Действительно, получение «хорошей» в статистическом смысле оценки одного значения вероятности требует десятков, а иногда сотен и тысяч экспериментов. Учитывая при этом, что число оцениваемых значений вероятностей составляет L^{N+1}, то объем выборочных данных становится просто катастрофически большим.

Таким образом, объективно существуют причины, которые в ряде случаев делают невозможным использование байесовской процедуры совместной классификации. Это привело к появлению различных алгоритмов, основанных на определенных упрощениях. Одним из типичных упрощений является допущение независимости решений классификаторов-экспертов и/или их признаков. Ниже приведен способ настройки совместного решающего правила для такой ситуации. Однако надо заметить, что в ситуации не-

большого числа классов и малого количества классификаторов-экспертов целесообразно использовать приведенную выше байесовскую процедуру в силу ее оптимальности.

6.2.3 Совместная классификация при независимых экспертах

Предположим, что решения Q_n, выносимые различными экспертами, являются независимыми в каждом из классов. С вероятностной точки зрения это предположение означает выполнение равенств:

$$P(\mathbf{Q}=\mathbf{q}/\Omega_l)=\prod_{n=0}^{N-1}P(Q_n=q_n/\Omega_l),\quad l=\overline{0,L-1}.$$

Тогда байесовская дискриминантная функция l-го класса

$$d_l(\mathbf{q})=P(\Omega_l)P(\mathbf{Q}=\mathbf{q}/\Omega_l),\quad ,\quad l=\overline{0,L-1}$$

представима в виде

$$d_l(\mathbf{q})=P(\Omega_l)\prod_{n=0}^{N-1}P(Q_n=q_n/\Omega_l),\quad l=\overline{0,L-1}.$$

Введем в рассмотрение функцию:

$$G_s(q)=\begin{cases}1, & \text{если } q=s,\\ 0, & \text{иначе.}\end{cases}$$

Тогда значение вероятности $P(Q_n=q_n/\Omega_l)$ может быть представлено следующим образом:

$$P(Q_n=q_n/\Omega_l)=\sum_{s=0}^{L-1}P(Q_n=s/\Omega_l)G_s(q_n)\ .$$

В этом случае выражение для дискриминантной функции совместного решающего правила перепишется в виде:

$$d_l(\mathbf{q}) = P(\Omega_l)\prod_{n=0}^{N-1}\sum_{s=0}^{L-1} P(Q_n = s/\Omega_l)G_r(q_n),\ \ l=\overline{0,L-1}. \quad (6.4)$$

Применяя к полученному выражению логарифмическое преобразование, а также учитывая, что для конкретного n функция $G_s(q_n)=0$ при всех $s \neq q_n$, получим

$$d_l(q) = \ln P(\Omega_l) + \sum_{n=0}^{N-1}\sum_{s=0}^{L-1} G_s(q_n)\cdot \ln P(Q_n = s/\Omega_l),\ \ l=\overline{0,L-1}. \quad (6.5)$$

Обозначим:

$$z_j = G_s(q_n)\text{, где } \ j = L\cdot n + s\ ,\ \ j=\overline{0,L\cdot N-1}\ ,\ \ s=\overline{0,L-1}\ ,$$

$$z = (z_0,\ldots,z_{L\times N-1})\ . \quad (6.6)$$

Введем $z=(z_0,\ldots,z_{LN-1})^T$ – бинарный вектор длины $L\cdot N$. Введем коэффициенты

$$W_j^l = \ln P(Q_n = s/\Omega_l)\ ,\ \ j = Ln+s\ ,\ \ n=\overline{0,N-1}\ ,\ \ s=\overline{0,L-1}\ . \quad (6.7)$$

Тогда окончательным решением задачи является классификатор с дискриминантной функцией вида:

$$d_l(\mathbf{y}) = \left[\mathbf{W}^l\right]^T \mathbf{z} + \ln P(\Omega_l)\ ,$$

здесь $\mathbf{W}^l$ – вектор-столбец, составленный из коэффициентов W_j^l ($j=\overline{0,L\cdot N-1}$).Таким образом *оптимальная решающая функция композиции оказалась линейной относительно введенного бинар-*

ного вектора признаков (6.6) и преобразованного признакового пространства.

Замечание 1

В качестве функции $G_s(q)$ может выступать, например, полином (L-1)-го порядка:

$$G_s(q)=\frac{1}{\prod_{l=0}^{L-1}(s-l)}\prod_{l=0}^{L-1}(q-l). \qquad (6.8)$$

Из выражений (6.5) и (6.8) видно, что в признаковом пространстве решений экспертов классификатор композиции является *полиномиальным*, и степень полинома, описывающего его решающую функцию, равна (L-1). То есть при построении совместного решения правила распознавания для двух классов мнения экспертов комбинируются в линейной форме, для трех классов они учитываются в полиноме второй степени и т.д.

В дополнение к изложенному общему алгоритму совместной классификации рассмотрим одни важный частный случай, когда решается задача бинарного (двухальтернативного) распознавания (обнаружения) и в композиции, в соответствии с первым замечанием, комбинируются мнения экспертов линейным образом. Данная задача известна как задача распознавания бинарных векторов признаков и ее решение дано, например, в монографии [3]. В данном случае ее решение вытекает как частный случай (6.4) и (6.8).

Итак, пусть L=2. Тогда функция $G_s(q)$ определяется следующим образом:

$$G_0(q)=1-q\ ,\ G_1(q)=q\ .$$

В результате дискриминантная функция (6.4) может быть переписана следующим образом:

$$d_l(\mathbf{q}) = P(\Omega_l)\prod_{n=0}^{N-1}\Big(\big(1-P(Q_n=1/\Omega_l)\big)(1-q_n)+P(Q_n=1/\Omega_l)q_n\Big),\ \ l=\overline{0,L-1}.$$

Окончательным решением задачи является мультиклассификатор с дискриминантной функцией вида:

$$d_l(\mathbf{q}) = \left[\ln\big(P(\Omega_l)\big)+\sum_{n=0}^{N-1}\ln\big(1-P(Q_n=1/\Omega_l)\big)\right] + \sum_{n=0}^{N-1} q_n\cdot\ln\left(\frac{P(Q_n=1/\Omega_l)}{1-P(Q_n=1/\Omega_l)}\right),$$

который, как ожидалось, является линейным относительно решений экспертов.

Аналогичным образом можно получить выражение для мультиклассификатора в терминах отношения правдоподобия:

$$\forall\ j\neq l\quad \tilde{\Lambda}_{lj}(\overline{q})\geq\tilde{\lambda}_{jl}\Rightarrow\mathbf{q}\in\Omega_l,$$

где

$$\tilde{\Lambda}_{lj}(\mathbf{q}) = \sum_{n=0}^{N-1} W_{lj}^n\cdot q_i,\qquad W_{lj}^n = \ln\left(\frac{P(Q_n=1/\Omega_l)}{1-P(Q_n=1/\Omega_l)}\cdot\frac{1-P(Q_n=1/\Omega_j)}{P(Q_n=1/\Omega_j)}\right),$$

$$\tilde{\lambda}_{jl} = \ln\left(\frac{P(\Omega_j)}{P(\Omega_l)}\right)+\sum_{n=0}^{N-1}\ln\left(\frac{1-P(Q_n=1/\Omega_j)}{1-P(Q_n=1/\Omega_l)}\right).$$

Очевидно, отношение правдоподобия также является линейной функцией решений экспертов.

Замечание 2

В отличие от общей ситуации, количество требуемых для оценки параметров при независимых решениях экспертов в соответствии с выражением (6.7) составляет $L\times N$ для каждого класса. В этом случае объем данных, используемых при классифика-

ции равен $N \times L^2$, что существенно меньше, чем приведенные в табл. 6.2 значения. Это позволяет использовать представленную здесь процедуру совместной классификации для бо́льшого числа классов и классификаторов-экспертов.

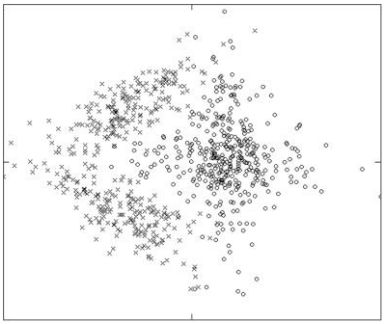
Вычисление вероятностей ошибочной классификации при совместной классификации рассмотрено в пособии [2].

Эффективность правил совместной классификации

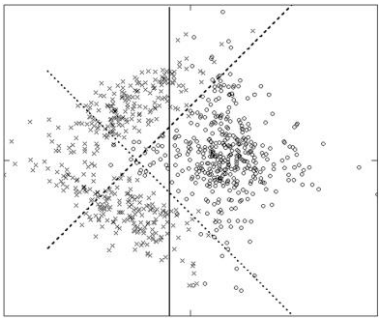
Для оценки эффективности использования совместного решающего правила был проведен ряд экспериментальных исследований по классификации. Данные были представлены в виде векторов смеси нормальных законов распределений, как показано на рисунке 6.2,а. В качестве экспертов выступали линейные классификаторы, параметры которых оказывались локально оптимальными. Результаты экспериментальных исследований представлены в таблице 6.2 и на рисунках 6.2,б–6.2,в. Формируемая разделяющая граница мультиклассификатора, базируясь на линейных разделяющих границах экспертов, показанных на рисунке 6.2,б, оказывается кусочно-линейной (см. рисунок 6.2,в). Как следствие такой «адаптации» к типу распределения данных, качество работы мультиклассификатора оказалось лучше качества любого из экспертов и для вероятностей ошибок p_0 и p_1, и для величины риска в целом (см. таблицу 6.2).

Таблица 6.2. **Эффективность совместной классификации с минимальной информацией о решениях (независимых) экспертов**

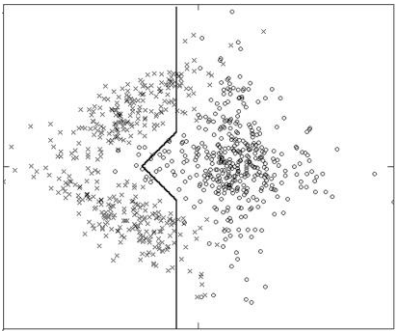
Классификатор	p_0	p_1	Риск
эксперт 1	0,785	0,085	0,435
эксперт 2	0,775	0,08	0.425
эксперт 3	0,23	0,12	0,175
Совместная классификация	0,23	0,055	0,143



а) исходные данные;



б) разделяющие границы экспертов;



в) совместная классификация
с независимыми решениями экспертов;

“……” – эксперт 1,
“- – - – -” – эксперт 2,

“ ” – эксперт 3,

“x” – объекты класса Ω_0,
“o” – объекты класса Ω_1,

Рисунок 6.2 – Вектора признаков и результаты классификации
с минимальной информацией о решениях экспертов

6.3 Двухэтапная последовательная процедура совместной классификации

6.3.1 Структура двухэтапной последовательной процедуры совместной классификации

Примем модель данных, в соответствии с которой большая часть анализируемых (классифицированных) данных относится к не интересующему нас «фоновому» классу, а интересующие нас

данные, интерпретируемые как «объекты», встречаются достаточно редко. Подобная модель характерна для многих приложений, в частности для задач поиска и распознавания локальных объектов на изображении [72–75].

Из принятой модели следует, что прямое использование расширенного набора признаков и качественного решающего правила для большей части анализируемых фрагментов данных неэффективно и вычислительно избыточно, так как их занимает не интересующая нас фоновая составляющая. Поэтому представляется целесообразным решать задачу в два этапа:

– до процесса распознавания провести предварительный анализ данных с целью выделения из них фрагментов, которые потенциально содержат интересующую информацию – объекты. Объем «полезных» данных по сравнению с первоначальным оказывается сравнительно небольшим;

– по выделенным данным производить по возможности полный анализ с целью достаточно качественной классификации. Большая вычислительная сложность производимого анализа компенсируется малым числом анализируемых фрагментов.

Легко заметить, что первый этап по своему содержанию близок к обычной схеме обнаружения. Однако, к нему предъявляются дополнительные требования: низкой вычислительной сложности и малой вероятности пропуска объекта при одновременно небольшом числе выделяемых фрагментов.

Основное преимущество такой *двухэтапной процедуры последовательного обнаружения и распознавания* состоит в возможности, с одной стороны, снижения вычислительной сложности (первый этап) и, с другой стороны, использования расширенного набора средств формирования признаков и классификации (второй этап).

Далее примем следующие обозначения:

$\bar{x}$ – вектор признаков объекта;

$\{\Omega_l\}_{l=0}^{L-1}$ – классы, среди которых Ω_0 – класс фона, а остальные классы интерпретируются как классы объекта;

$Q \equiv Q(\bar{x})$ – предикат детектирования объекта в данных экспертом первого этапа. Событие Q интерпретируется как решение об отнесении вектора признаков к классу объектов, а $\bar{Q}$ – к классу фона;

$Q_l \equiv Q_l(\bar{x})$ – предикат классификации поступивших данных экспертом второго этапа к классу Ω_l.

Заметим, что в общем случае события Q и Q_0 не являются несовместными (может выполняться неравенство $P(QQ_0) \neq 0$), то есть заведомо допускается попадание "фоновых" данных на второй этап процедуры совместной классификации.

В рамках вышеуказанных соглашений вероятности классификации при обнаружении и распознавании в двухэтапной процедуре задаются следующим образом:

$$\begin{aligned} p_{ji} &= P(QQ_i / \Omega_j), \qquad i = \overline{1, L-1},\ j = \overline{0, L-1} \\ p_{j0} &= P(\bar{Q} / \Omega_j) + P(QQ_0 / \Omega_j), \end{aligned} \quad . \qquad (6.9)$$

Здесь произведение событий QQ_i означает, что анализируемый объект эксперт первого этапа отнес к классу объектов, и одновременно с этими классификатор второго этапа определил его к i-ому классу. При отнесении случайного фрагмента к классу фона возможно, с одной стороны, сделать это на этапе предварительного обнаружения (событие $\bar{Q}$), с другой стороны, проклассифицировать его в нулевой класс на втором этапе (событие QQ_0). По-

скольку события QQ_0 и $\bar{Q}$ несовместны, общая вероятность отнесения случайного фрагмента к классу «фон» может быть записана как сумма вероятностей первого и второго случаев. Обозначение класса Ω_j при записи вероятностей означает принадлежность объекта соответствующему классу. Поэтому запить $P\left(\bar{Q}/\Omega_j\right)$ обозначает вероятность классификации как «необъект» вектора признаков, который принадлежал объекту класса Ω_j. Событие $P\left(Q\Omega_i\right)$ интерпретируется как одновременная вероятность того, что эксперт первого этапа проинтерпретировал вектор признаков как объект и этот же вектор признаков принадлежал объекту класса Ω_i, и т.д.

В дополнении к качественным показателям процесса распознавания введем в рассмотрение характеристику, характеризующую вычислительную сложность процедуры совместной классификации. Заметим, что традиционно оценка сложности алгоритмических и программных средств производится по формуле:

$$U_a + \eta U_m \ , \tag{6.10}$$

где U_a, U_m – количество, соответственно, сложений и умножений, используемых для обработки или классификации одного фрагмента изображения, то есть сложность на отсчет изображения; η – коэффициент относительной сложности операции умножения по отношению к операции сложения, типичное значение которого для наиболее массовых компьютеров лежит в диапазоне (1...3).

В единицах (6.10) средняя вычислительная сложность одного акта обнаружения и распознавания составляет:

$$U = U_I + P(Q)U_{II} \ , \tag{6.11}$$

где U_I и U_{II}, соответственно, сложность первого и второго экспертов.

Разумный компромисс между качеством решения задачи и затраченными вычислительными ресурсами достигается, если рассматривать величины сложности и качества либо как составляющие некоторой целевой функции, которая отражает приемлемость выбора каждой конкретной пары, либо как критерии оптимальности и ограничения в задаче оптимизации технологии. Наиболее часто встречающейся задачей *параметрической оптимизации*, используемой при построении алгоритма распознавания, является задача улучшения его качества при одновременном ограничении на сложность процесса принятия решения [72–3]:

$$\begin{cases} R(\mathbf{a}) \to \min\limits_{\mathbf{a}}, \\ U(\mathbf{a}) \le U_{\lim}. \end{cases} \tag{6.12}$$

Здесь $U_{\lim}$ – верхняя граница сложности, $\mathbf{a}$ – вектор параметров алгоритма, по которому производится оптимизация.

6.3.2 Параметрическая оптимизация двухэтапной последовательной процедуры совместной классификации

Пусть решается задача параметрической оптимизации двухэтапной последовательной процедуры совместной классификации в виде (6.12) по вектору параметров

$$\mathbf{a} = (\mathbf{a}_I, \mathbf{a}_{II}),$$

где $\mathbf{a}_I$ и $\mathbf{a}_{II}$ задают, соответственно, параметры экспертов первого и второго этапов. Тогда, учитывая выражения (6.9), (6.11) и (6.12) задача оптимизации перепишется в виде:

$$\begin{cases} \sum\limits_{j=0}^{L-1} r_{j0} + \sum\limits_{i,j=0}^{L-1} \left(r_{ji} - r_{j0}\right) P\left(\Omega_j\right) P\left(Q(\mathbf{a}_I) Q_i(\mathbf{a}_{II}) / \Omega_j\right) \to \min\limits_{(\mathbf{a}_I, \mathbf{a}_{II})}, \\ P\left(Q(\mathbf{a}_I)\right) \le \dfrac{U_{\lim} - U_I}{U_{II}}. \end{cases}$$

Очевидно, что показатель сложности двухэтапной процедуры не содержит зависимости от совместного вектора параметров $\mathbf{a}$. В то же время показатель качества функционально с ним связан, а, следовательно, не может выступать в качестве критерия настройки ни одного из двух экспертов. Получаем противоречие – мы не можем настроить совместную процедуру без настройки каждого из ее экспертов, и не можем настроить ни одного из экспертов по критерию качества процедуры в целом. Для того чтобы изменить эту ситуацию, воспользуемся возможностью представления вероятностей классификации из класса в класс в виде двух альтернативных выражений:

$$\begin{aligned} P\left(QQ_i / \Omega_j\right) &= P\left(Q / \Omega_j\right) P\left(Q_i / Q\Omega_j\right), \\ P\left(QQ_i / \Omega_j\right) &= P\left(Q_i / \Omega_j\right) P\left(Q / Q_i\Omega_j\right), \end{aligned} \quad i, j = \overline{0, L-1}. \qquad (6.13)$$

Пусть теперь в процессе оптимизации получено некоторое решение $\left(\mathbf{a}_I^{fix}, \ \mathbf{a}_{II}^{fix}\right)$. Зафиксируем одного из экспертов, например второго, и будем изменять параметры другого. Тогда получим:

$$P\left(Q(\mathbf{a}_I) Q_i\left(\mathbf{a}_{II}^{fix}\right) / \Omega_j\right) = P\left(Q_i\left(\mathbf{a}_{II}^{fix}\right) / \Omega_j\right) P\left(Q(\mathbf{a}_I) / Q_i\left(\mathbf{a}_{II}^{fix}\right) \Omega_j\right), \ i, j = \overline{0, L-1}.$$

То есть изменение вероятности классификации из класса в класс пропорционально изменению вероятности $P\left(Q(\mathbf{a}_I) / Q_i\left(\mathbf{a}_{II}^{fix}\right) \Omega_j\right)$, причем событие, стоящее как условие, является фиксированным в процессе изменения параметров перво-

го этапа. Аналогично, фиксированным оказывается и коэффициент пропорциональности изменения $P\left(Q_i\left(\mathbf{a}_{II}^{fix}\right)/\Omega_j\right)$. Точно также, в соответствии с первым выражением в (6.13), можно получить величину изменения искомой вероятности при фиксированном первом эксперте.

Таким образом, можно, фиксируя параметры то первого, то второго из них, находить вектор оптимальных изменений и производить соответствующую коррекцию, что приводит к решению задачи настройки последовательной процедуры совместной классификации. Заметим, что подобный подход требует итерационного представления алгоритма оптимизации, аналогичного известному методу покоординатного спуска [77]. При этом указанные возможности позволяют построить итерационные алгоритмы различных типов. В настоящем разделе рассматриваются два из них.

Итерационный алгоритм параметрической оптимизации на основе градиентного метода

Для построения итерационного алгоритма оптимизации двухэтапной композиции экспертов на основе градиентного метода необходимым является наличие возможности вычисления действительного значения вектора изменения параметров. Для этого требуется вычисление производных вида:

$$\frac{\partial P\left(Q(\mathbf{a}_I)Q_i\left(\mathbf{a}_{II}\right)/\Omega_j\right)}{\partial \mathbf{a}}.$$

Учитывая выражение (6.13), получаем следующий способ их вычисления в некоторой точке $\left(\mathbf{a}_I^{fix},\ \mathbf{a}_{II}^{fix}\right)$:

$$\left.\frac{\partial P\left(Q(\mathbf{a}_I)Q_i\left(\mathbf{a}_{II}\right)/\Omega_j\right)}{\partial \mathbf{a}_I}\right|_{(\mathbf{a}_I,\mathbf{a}_{II})=\left(\mathbf{a}_I^{fix},\mathbf{a}_{II}^{fix}\right)} = P\left(Q_i\left(\mathbf{a}_{II}\right)/\Omega_j\right)\left.\frac{\partial P\left(Q(\mathbf{a}_I)/Q_i\left(\mathbf{a}_{II}\right)\Omega_j\right)}{\partial \mathbf{a}_I}\right|_{\mathbf{a}_I=\mathbf{a}_I^{fix}},$$

$$\left.\frac{\partial P\left(Q(\mathbf{a}_I)Q_i\left(\mathbf{a}_{II}\right)/\Omega_j\right)}{\partial \mathbf{a}_{II}}\right|_{(\mathbf{a}_I,\mathbf{a}_{II})=\left(\mathbf{a}_I^{fix},\mathbf{a}_{II}^{fix}\right)} = P\left(Q(\mathbf{a}_I)/\Omega_j\right)\left.\frac{\partial P\left(Q_i\left(\mathbf{a}_{II}\right)/Q(\mathbf{a}_I)\Omega_j\right)}{\partial \mathbf{a}_{II}}\right|_{\mathbf{a}_{II}=\mathbf{a}_{II}^{fix}}.$$

Зная теперь значения производных можно воспользоваться градиентным или любым другим методом оптимизации первого порядка [77].

К сожалению, воспользоваться алгоритмом оптимизации на основе градиентного метода достаточно сложно. Основной проблемой является то, что на практике вид зависимости вероятностей классификации от вектора параметров алгоритма классификации не может быть представлен в аналитическом виде, а значит и вычисление указанных производных невозможно. В тех редких случаях, когда получение градиента все же допускается, существуют дополнительные трудности применения градиентного алгоритма: учет ограничений в задаче оптимизации; выбор шага при коррекции параметров, высокая вычислительная сложность такого алгоритма применительно к задаче оптимизации процедуры распознавания. Все это делает использование итерационного алгоритма оптимизации на основе градиентного подхода малоэффективным или невозможным в большинстве реальных задач. Поэтому необходима его модификация. Эта модификация, исходя из перечисленных недостатков градиентного алгоритма, должна касаться процесса модификации параметров. А именно, если на очередной итерации происходит полная коррекция вектора параметров каждого из этапов, то можно без каких бы то ни было затруднений использовать известные алгоритмы[ограничений по сложности в таком случае также не является проблематичным.

Таким образом, итерационный алгоритм параметрической оптимизации двухэтапной последовательной процедуры классификации может быть представлен следующим образом.

Итерационный алгоритм поэтапной параметрической оптимизации

1) Решается задача оптимизации алгоритма распознавания второго этапа (второго эксперта) в виде:

$$\sum_{j=0}^{L-1} r_{j0} + \sum_{i,j=0}^{L-1} \left(r_{ji} - r_{j0}\right) P\left(\Omega_j\right) P\left(Q(\mathbf{a}_I)/\Omega_j\right) P\left(Q_i(\mathbf{a}_{II})/Q(\mathbf{a}_I)\Omega_j\right) \to \min_{\mathbf{a}_{II}}. \quad (6.14)$$

2) Решается задача оптимизации алгоритма распознавания первого этапа (первого эксперта) в виде:

$$\begin{cases} \sum_{j=0}^{L-1} r_{j0} + \sum_{i,j=0}^{L-1} \left(r_{ji} - r_{j0}\right) P\left(\Omega_j\right) P\left(Q_i(\mathbf{a}_{II})/\Omega_j\right) P\left(Q(\mathbf{a}_I)/Q_i(\mathbf{a}_{II})\Omega_j\right) \to \min_{\mathbf{a}_I}, \\ P\left(Q(\mathbf{a}_I)\right) \le \dfrac{U_{\lim} - U_I}{U_{II}}. \end{cases} \quad (6.15)$$

3) Если точность решения задачи не удовлетворяет, то возвращаемся к решению задачи оптимизации второго этапа.

Для определенности будем считать, что алгоритм прекращает свою работу в ситуации, если новые изменения величины риска R и сложности $P(Q)$ не превышают некоторых изначально заданных величин. В качестве начального состояния в данном алгоритме выбирается такое состояние классификатора первого этапа, когда $Q=\Omega_0$, то есть на первой итерации в качестве условий будут выступать только события принадлежности объектов Ω_j.

Сделаем некоторые замечания относительно задач (6.14) и (6.15). При решении задачи распознавания число сформированных классов не превышает их количество в первоначальной задаче. В задаче же

обнаружения (6.15) число всех условных событий составляет L^2 , то есть квадратично зависит от первоначального числа классов. Такое увеличение их числа может привести к неэффективному решению задачи – известным является факт повышения значения среднего риска при увеличении числа классов и неизменном пространстве признаков [1,36]. Поэтому при решении каждой конкретной задачи необходимо, учитывая конкретные значения штрафов r_{ij} , производить модификацию выражения (6.15) для снижения числа условных событий. Это может быть эффективно сделано, если матрица штрафов является простейшей, как в следующем подразделе.

Итерационный алгоритм поэтапной параметрической оптимизации в случае простейшей матрицы штрафов. Итерационные схемы алгоритма

Пусть матрица штрафов является простейшей. В этом случае задачи оптимизации (6.14) и (6.15) меняются. А именно, задача (6.14) может быть записана в виде:

$$P(\Omega_0)P\left(Q(\mathbf{a}_I)/\Omega_0\right)+ \\ +\sum_{j=0}^{K-1}P\left(\Omega_j\right)P\left(Q(\mathbf{a}_I)/\Omega_j\right)P\left(Q_i(\mathbf{a}_{II})/Q(\mathbf{a}_I)\Omega_j\right)\to\min_{\mathbf{a}_{II}}, \tag{6.16}$$

Для задачи (6.15) показатель качества можно переписать следующим образом:

$$R=\sum_{j=1}^{L-1}P\left(\Omega_j\right)P\left(\bar{Q}/\Omega_j\right)+\sum_{j=0}^{L-1}\sum_{\substack{i=0,\\ i\neq j}}^{L-1}P\left(\Omega_j\right)P\left(QQ_i/\Omega_j\right)= \\ =P\left(\bar{Q}\cdot\bigcup_{j=1}^{L-1}\Omega_j\right)+\sum_{j=0}^{L-1}P\left(Q\Omega_j\bigcup_{\substack{i=1,\\ i\neq j}}^{L-1}Q_i\right)=$$

$$= P\left(\bar{Q}\cdot\bigcup_{j=1}^{L-1}\Omega_j\right)+\sum_{j=0}^{L-1}P\left(Q\Omega_j\bar{Q}_j\right)=P\left(\bar{Q}\cdot\bigcup_{j=1}^{L-1}\Omega_j\right)+P\left(Q\bigcup_{j=0}^{L-1}\Omega_j\bar{Q}_j\right).$$

Окончательно:

$$\begin{cases} P\left(\bar{Q}(\mathbf{a}_I)\cdot\bigcup_{j=1}^{L-1}\Omega_j\right)+P\left(Q(\mathbf{a}_I)\bigcup_{j=0}^{L-1}\Omega_j\bar{Q}_j(\mathbf{a}_{II})\right)\to\min\limits_{\mathbf{a}_I}, \\ P\left(Q(\mathbf{a}_I)\right)\le\dfrac{U_{\lim}-U_I}{U_{II}}. \end{cases} \qquad (6.17)$$

Очевидно, что показатель качества этой задачи может быть представлен в виде:

$$P\left(\bar{Q}\cdot V_1\right)+P\left(Q\cdot V_0\right),$$

где

$$V_1=\bigcup_{j=1}^{L-1}\Omega_j,\quad V_0=\bigcup_{j=0}^{L-1}\Omega_j\bar{Q}_j\ . \qquad (6.18)$$

В то же время, используя другие преобразования, возможно получение и других форм представления критерия (6.17). Некоторые из вариантов приведены в таблице 6.3.

Аналогичным образом может быть изменено и выражение (6.16). Подобная неоднозначность представления задачи оптимизации в общем случае может привести к множественности результатов ее решения. Однако, если на каждой итерации выносится оптимальное или *байесовское* решение, то результат оптимизации не будет зависеть от выбора *итерационной схемы*. Покажем это на примере второй и третьей схем. Для первой из них запишем байесовский классификатор. Как известно, его разделяющая граница определяется из неравенства [1-3,36]:

$$P\left(\Omega_0\bigcup_{l=1}^{L-1}Q_l\right)p\left(\mathbf{y}\Big/\Omega_0\bigcup_{l=1}^{L-1}Q_l\right)\gtrless P\left(\bigcup_{l=1}^{L-1}\Omega_l Q_l\right)p\left(\mathbf{y}\Big/\bigcup_{l=1}^{L-1}\Omega_l Q_l\right).$$

Таблица 6.3. **Формы представления составляющих критерия качества**

итерационная схема	V_0	V_1
1	Ω_0	$\bigcup_{j=0}^{L-1}\Omega_j Q_j$
2	$\Omega_0\bar{Q}_0$	$\bigcup_{j=1}^{L-1}\Omega_j Q_j$
3	$\bigcup_{j=0}^{L-1}\Omega_j\bar{Q}_j$	$\bigcup_{j=1}^{L-1}\Omega_j$
4	$\bigcup_{j=0}^{L-1}\overline{\Omega_j}Q_j$	$\bigcup_{j=1}^{L-1}\Omega_j$

Граница не изменится, если к правой и левой частям неравенства добавить одну и ту же величину $P\left(\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)p\left(\mathbf{y}\Big/\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)$.

Учитывая, что присутствующее в этом выражении событие несовместно ни с одним из событий третьей схемы, получим:

$$P\left(\left(\Omega_0\bigcup_{l=1}^{L-1}Q_l\right)\bigcup\left(\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)\right)f\left(\mathbf{y}\Big/\left(\Omega_0\bigcup_{l=1}^{L-1}Q_l\right)\bigcup\left(\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)\right)\gtrless$$

$$\gtrless P\left(\left(\bigcup_{l=1}^{L-1}\Omega_l Q_l\right)\bigcup\left(\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)\right)p\left(\mathbf{y}\Big/\left(\bigcup_{l=1}^{L-1}\Omega_l Q_l\right)\bigcup\left(\bigcup_{l=1}^{L-1}\Omega_l\bar{Q}_l\right)\right).$$

Из последнего неравенства следует:

$$P\left(\bigcup_{l=0}^{L-1}\Omega_l\bar{Q}_l\right)p\left(\mathbf{y}\Big/\bigcup_{l=0}^{L-1}\Omega_l\bar{Q}_l\right)\underset{<}{>}P\left(\bigcup_{l=1}^{L-1}\Omega_l\right)p\left(\mathbf{y}\Big/\bigcup_{l=1}^{L-1}\Omega_l\right),$$

что задает выражение разделяющей границы байесовского классификатора для третьей итерационной схемы. Аналогичным образом можно показать эквивалентность, с точки зрения байесовского решающего правила, остальных итерационных схем. В то же время, при использовании классификатора, отличного от байесовского, результаты оптимизации будут зависеть от вида задачи, и, следовательно, от итерационной схемы. Значит для нахождения наилучшего решения необходимо также исследовать результаты оптимизации для каждой из схем и выбрать наилучшую для решения задачи.

Вернемся к вопросу размерности задач распознавания и обнаружения – числу классов в каждой из них. Очевидно, что поскольку алгоритм обнаружения может быть сведен к виду (6.18), где V_0 и V_1 выполняют роль событий принадлежности к соответствующему классу, то и число классов в задаче обнаружения снижено до двух. Число классов в алгоритме распознавания (6.16), как и в (6.14), осталось идентичным числу классов в первоначальной задаче.

6.3.3 Результаты экспериментальных исследований

Для демонстрации эффективности итерационного алгоритма оптимизации двухэтапной последовательной процедуры совместной классификации рассмотрим задачу поиска и распознавания локальных объектов на изображении.

Характер изображений в этой задаче и специфика процесса их анализа позволяют говорить об удовлетворении принятой ранее

модели данных. А именно, для задачи поиска и распознавания локальных объектов на изображении характерно, что [1, 72–76]:

– подавляющую часть изображения занимает не интересующая нас «фоновая» составляющая;

– интересующие нас на изображении локальные области – объекты – малы по сравнению с размерами самого изображения;

– расположение объектов в плоскости изображения таково, что для каждого из них можно указать такое положение некоторой выпуклой наперед заданной области, что никакой другой объект в ней не будет присутствовать;

– количество содержащихся на изображении объектов мало по сравнению с общим количеством отсчетов изображения.

В рамках данной модели одним из наиболее удобных способов анализа данных является их локальная обработка [1, 72–76]. В задачах обнаружения и распознавания локальных объектов на изображении при каждом положении окна обработки анализируемый фрагмент рассматривается как отдельное изображение (объект), для которого и производится классификация.

Для экспериментальных исследований были выбраны изображения, содержащие объекты двух классов. Изображение, приведенное на рисунке рис. 6.2,а использовалось для обучения и оценки результатов классификации. В качестве признаков в работе были выбраны следующие:

– на первом этапе – два локальных средних с различными размерами окна усреднения (15*15, 25*25), вычислительная сложность их расчета составляет 8 операций на отсчет;

– на втором этапе – пять моментных инвариантов (см. п.9.2.3), вычисленных по градиенту выделенного фрагмента (25*25) изображения с использованием алгоритма прямой свертки; вычислительная сложность расчета инвариантов составляет приблизительно 6300 операций на отсчет изображения.

В качестве классификаторов в работе были использованы:

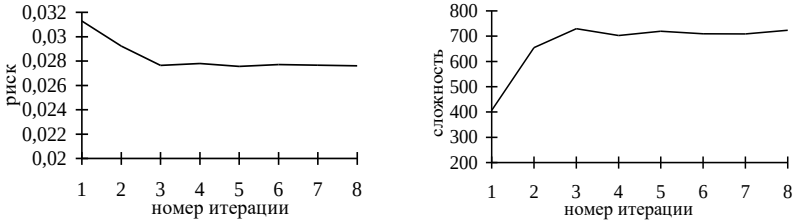
– на первом этапе – линейный классификатор, рассчитываемый с использованием процедуры Петерсона-Матсона; вычислительная сложность его использования для классификации составляет 4 операции на отсчет [1–3];

– на втором этапе – квадратичный классификатор, вычисляемый в предположении нормальности закона распределения вектора признаков на втором этапе, вычислительная сложность его использования для классификации изображения при двух классах объектов и одном классе фона составляет приблизительно 200 операций на отсчет [1–3].

Таким образом, в проведенных исследованиях:

$$U_I = 12, \qquad U_{II} = 6500 .$$

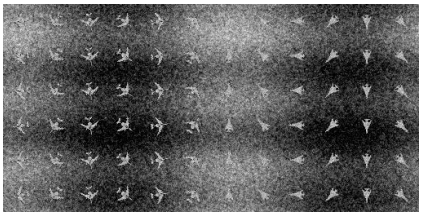
Сравнение качества работы оптимизированной двухэтапной процедуры производилось с наилучшим достижимым качеством классификатора-эксперта второго этапа. Ограничения на сложность процедуры не задавались. Изменение решения в процессе оптимизации и ее результат продемонстрированы на рисунках 6.3–6.4.



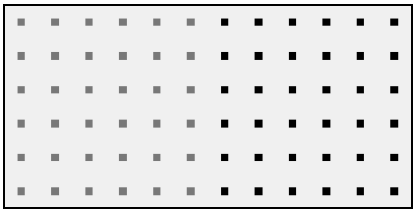
а) величина среднего риска при распознавании

б) вычислительная сложность при распознавании

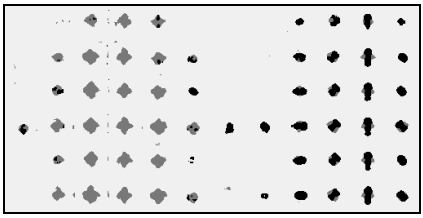
Рисунок 6.3 – Изменение показателей эффективности в процессе оптимизации двухэтапной последовательной процедуры совместной классификации с использованием итерационного алгоритма



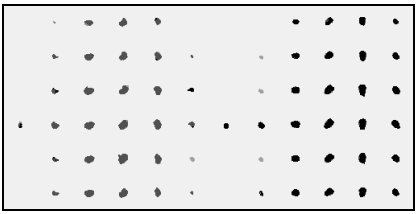
а) входное изображение;



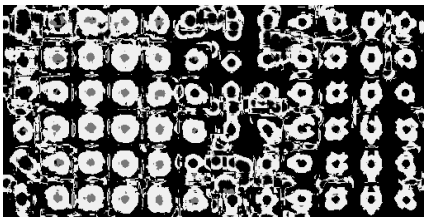
б) требуемая классификация;



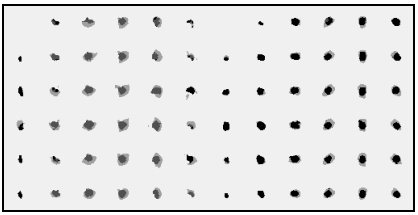
в) наилучший результат распознавания классификатора второго этапа (риск равен 0,05634) и его состояние на первой итерации алгоритма оптимизации;



г) результаты работы двухэтапной процедуры классификации на первой итерации алгоритма оптимизации;



д) результат работы классификатора второго этапа на последней итерации алгоритма оптимизации



е) результаты работы двухэтапной процедуры классификации на последней итерации алгоритма оптимизации;

Рисунок 6.4 – Экспериментальные результаты оптимизации двухэтапной последовательной процедуры совместной классификации

На рисунках 6.3 представлено изменение показателей эффективности процедуры совместной классификации, а на рисунках 6.4 – изменение меток классов (полей классификации). Результаты

позволяют утверждать о приблизительно двукратном выигрыше в качестве распознавания при использовании настроенной двухэтапной последовательной процедуры совместного распознавания с двумя экспертами. При этом до 15% выигрыша достигалось за счет использования итерационного алгоритма оптимизации композиции. Заметим также, что дополнительно произошло снижение сложности процесса принятия решения: по сравнению со сложностью эксперта второго этапа значение вычислительной сложности снизилось приблизительно в 10 раз (см. рисунок 6.3б).

6.4 Бустинг как метод построения алгоритма совместной классификации

Бустинг (англ.: boosting) – процедура последовательного построения композиции алгоритмов машинного обучения, при которой каждый следующий алгоритм стремится компенсировать недостатки композиции всех предыдущих алгоритмов [55–57]. Бустинг на протяжении последнего десятилетия является одним из наиболее эффективных методов машинного обучения алгоритмов композиции и построения самой композиции. Наиболее известными решениями, полученными с использованием бустинга являются детектор Виола-Джонса [58] и случайный лес (см. п. 3.3.2 настоящего пособия).

6.4.1 Постановка задачи

Рассмотрим задачу бинарной (двуклассовой) классификации. Пусть задано множество из N объектов, для каждого из которых известен вектор признаков и метка принадлежности к классу:

$$\left\{\left(\overline{x}_j, r_j\right)\right\}_{j=0}^{N-1}, \quad \forall j\, r_j \in \{-1,1\}.$$

По аналогии с методом опорных векторов и логистической регрессии, положительное значение метки означает принадлежность объекта к классу Ω_1, отрицательное – Ω_0. Пусть, далее, множество алгоритмов бинарного (двуклассового) распознавания задано своими решающими функциями $\{d_m(\bar{x})\}_{m=0}^{M-1}$, причем $d_m(\bar{x}) \in \{-1,1\}$. Напомним, что *отступ* (англ.: margin) на *j*-ом объекте от границы между классами для задачи бинарной классификации определяется следующим образом:

$$M_j \equiv r_j d_m(\bar{x}_j).$$

Задача состоит в построении линейной композиции алгоритмов вида:

$$C_{M-1}(\bar{x}) \equiv \sum_{m=0}^{M-1} \alpha_m d_m(\bar{x}), \qquad (6.19)$$

то есть в определении коэффициентов композиции (6.19) и параметров участвующих в ней алгоритмов в некотором смысле оптимальным образом.

6.4.2 Алгоритм AdaBoost

Алгоритм был описан Yoav Freund и Robert Schapire в 1995 году, а в 2003 они за свою работу [55] получили Gödel Prize. Участвующие в композиции (6.19) алгоритмы авторы интерпретировали как «слабые», то есть такие, чье качество двуклассового распознавания оказывалось немного выше 0.5, то есть чуть лучше случайного угадывания. Даже в такой ситуации алгоритм бустинга позволял получить «сильный» алгоритм распознавания по мере роста числа «слабых».

Для получения обоснования алгоритма рассмотрим частичную сумму (6.19):

$$C_m(\bar{x}) \equiv C_{m-1}(\bar{x}) + \alpha_m d_m(\bar{x}). \qquad (6.20)$$

Процесс настройки композиции будем производить, последовательно присоединяя к существующей композиции следующий классификатор. То есть на текущем шаге потребуется определять коэффициент α_m и параметры для следующей решающей функции $d_m(\bar{x})$ в соответствие с заданным критерием качества, что характерно при реализации «жадной» стратегии построения композиции.

В алгоритме AdaBoost критерий качества композиции $C_m(\bar{x})$ определяется с использованием отступов на всем множестве объектов обучения:

$$\Im(m) \equiv \sum_{j=0}^{N-1} e^{-r_j C_m(\bar{x}_j)} \quad \left(= \sum_{j=0}^{N-1} e^{-M_j} \right). \qquad (6.21)$$

Критерий, очевидно, требует минимизации. Подставляя (6.21) в (6.20), получаем:

$$\Im(m) \equiv \sum_{j=0}^{N-1} e^{-r_j\left(C_{m-1}(\bar{x}_j) + \alpha_m d_m(\bar{x}_j)\right)} = \sum_{j=0}^{N-1} w_j^m e^{-r_j \alpha_m d_m(\bar{x}_j)}, \quad (6.22)$$

где

$$w_j^m = e^{-r_j C_{m-1}(\bar{x}_j)}. \qquad (6.23)$$

Множество объектов для последней суммы разобьем на два подмножества: в первом новый классификатор не делает ошибок, а на объектах второго ошибается. Для сокращения формальной записи обозначим множества соответствующих индексов следующим образом:

$$j{:}d_m\left(\overline{x}_j\right)=r_j;\quad j{:}d_m\left(\overline{x}_j\right)\neq r_j.$$

Поскольку для объектов из указанных множеств справедливо:

$$r_m d_m\left(\overline{x}_j\right)=\begin{cases}1,\ j{:}d_m\left(\overline{x}_j\right)=r_j,\\ -1,\ j{:}d_m\left(\overline{x}_j\right)\neq r_j;\end{cases}$$

выражение (6.22) можно переписать следующим образом:

$$\begin{aligned}\Im(m)\equiv&\sum_{j:d_m\left(\overline{x}_j\right)=r_j}w_j^m e^{-\alpha_m}+\sum_{j:d_m\left(\overline{x}_j\right)\neq r_j}w_j^m e^{\alpha_m}=\\ =&e^{-\alpha_m}\sum_{j=0}^{N-1}w_j^m+\left(e^{\alpha_m}-e^{-\alpha_m}\right)\sum_{j:d_m\left(\overline{x}_j\right)\neq r_j}w_j^m.\end{aligned}\qquad(6.24)$$

В указанном выражении новое решающее правило $d_m\left(\overline{x}\right)$ влияет только на второе слагаемое, что позволяет настроить его в результате решения следующей оптимизационной задачи:

$$d_m\left(\overline{x}\right){:}\sum_{j:d_m\left(\overline{x}_j\right)\neq r_j}w_j^m\to\min_{d_m}\qquad(6.25)$$

Настроив очередное решающее правило $d_m(\cdots)$, весовой коэффициент при нем в композиции (6.20) определим из условия минимума критерия (6.24). Необходимым условием минимума, как известно, является равенство нулю производной:

$$\frac{\partial\Im}{\partial\alpha_m}\equiv e^{\alpha_m}\sum_{j:d_m\left(\overline{x}_j\right)\neq r_j}w_j^m-e^{-\alpha_m}\sum_{j:d_m\left(\overline{x}_j\right)=r_j}w_j^m=0.$$

Отсюда

$$\alpha_m = \frac{1}{2}\ln\left(\frac{\sum\limits_{j:d_m(\bar{x}_j)=r_j} w_j^m}{\sum\limits_{j:d_m(\bar{x}_j)\neq r_j} w_j^m}\right).$$

Введя величину, характеризующую взвешенную ошибку классификации в виде:

$$\varepsilon_m = \frac{\sum\limits_{j:d_m(\bar{x}_j)\neq r_j} w_j^m}{\sum\limits_{j=0}^{N-1} w_j^m},$$

выражение для коэффициента в композиции (6.20) можно записать более компактно:

$$\alpha_m = \frac{1}{2}\ln\left(\frac{1-\varepsilon_m}{\varepsilon_m}\right). \qquad (6.26)$$

В результате алгоритм AdaBoost в виде псевдокода принимает следующий вид:

For m=0 to M-1 **do**

If (m == 0) then

$w_j^{-1}=1, \quad j=\overline{0,N-1};$

Else

Рассчитать w_j^m по (6.23)

End if

Найти $d_m(\bar{x})$ как решение задачи (6.25);

Расчет веса α_m по (6.26);

End For.

Результатом бустинга является линейная композиция алгоритмов в виде (6.19), известное *как правило взвешенного максимального голосования*.

6.5 Способы обучения алгоритмов композиции

Одним из условий построения хорошей по качеству распознавания композиции алгоритмов является присутствие достаточных различий в их работе (результатах классификации). Действительно, если алгоритмы композиции работают хорошо, но практически идентично классифицируют элементы выборки, то объединение их в композицию дает незначительный эффект. Ситуация может дополнительно усугубиться тем, что для построения базовых «слабых» алгоритмов используются одни и те же методы или модели.

Для преодоления указанного недостатка существуют технические приемы, позволяющие получать действительно различные решения-алгоритмы даже в тех ситуациях, когда используется одна и та же модель и метод обучения. Ниже дано их краткое описание.

Бэггинг (англ.: bagging, bootstrap aggregation)

Метод был предложен в 1996 году L.Breiman [78]. Суть метода состоит в обучении каждого из классификаторов в композиции на различных выборках той же длинны, что и исходная. Получение различных выборок обеспечивается выбором с возвращениями, то есть отдельные объекты могут попадать в новую выборку несколько раз. Поскольку получаемые выборки различны, различными оказываются и получаемые для них решающие правила, которые можно комбинировать, например, с помощью простого голосования. Такой метод не только позволяет получить взаимную компенсацию ошибок алгоритмов композиции, но и

добиться в некотором смысле «устойчивости» к ошибкам в ней, поскольку в формируемые подвыборки ошибочные объекты могут не попасть. Аналогичным образом работает известный в компьютерном зрении алгоритм RANSAC – RAndom Sample Consensus – метод согласованной оценки параметров на основе случайных выборок [52].

Метод случайных подпространств

Метод случайных подпространств (англ.: RSM – random subspace method) был предложен Tin Kam Ho в 1998 году [53]. Если в бэггинге подвыборки формируются по объектам, то в RSM – по признакам, которые также отбираются случайным образом (без повторения). Отбор подмножеств признаков неявно также выполняет процедуру фильтрации, позволяя получить поднаборы, свободные от «мешающих» признаков. Метод оказывается особенно удобным в случае большого числа признаков, что типично для некоторых приложений (например, задач компьютерного зрения).

Комбинированный метод

Указанные подходы – бэггинг и RMS могут быть использованы совместно для построения композиций.

При этом, в отличие от бустинга, построение алгоритмов композиции можно производить независимо, то есть с использованием *параллельных вычислительных устройств*.

6.6 Результаты шестого раздела

В шестом разделе представлены методы совместной классификации, решение в которых принимается на основании композиции решений нескольких других классификаторов, часто именуемых *экспертами*. Выделены две основные *стратегии совместной классификации*: *последовательная и параллельная*.

С *последовательной стратегий* принятия решений мы знакомились в четвертом разделе настоящего пособия, когда рассматривали последовательные методы классификации. В рамках шестого раздела последовательная стратегия рассматривается в частном случае – в *двухэтапной последовательной процедуре совместной классификации* (обнаружения и распознавания). В рамках этой процедуры структура композиции и, как следствие, процесс принятия решения оказываются фиксированы, однако для достижения требуемой эффективности могут настраиваться сами эксперты. Представлен оригинальный последовательный метод настройки такой композиции, который гарантирует сходимость процесса настройки. Результаты представленных экспериментов показывают существенное повышение качества классификации при возможных ограничениях на вычислительную сложность самого процесса классификации.

С параллельной стратегия классификации мы также знакомились – к ней можно отнести алгебру алгоритмов Ю.И. Журавлева и построение корректирующей операции (замыкание) над некорректными эвристическими алгоритмами распознавания. Напомним, что построение замыкания алгоритмов распознавания требовало представления алгоритма распознавания в виде композиции двух операторов: алгоритмического оператора и решающего правила (как правило, порогового). Первый оператор давал полную информацию о решении алгоритма (например, апостериорные вероятности, расстояние до разделяющей границы, значение оценки принадлежности и т.п.), что позволяет отнести этот подход совместной классификации к категории «с полной информацией о решении эксперта».

В рамках настоящего, шестого, раздела параллельная стратегия классификации рассмотрена более подробно для ситуаций, когда информация о решениях экспертов минимальна, то есть при

построении композиции классификаторов используется только номер класса. Однако, даже для этой ситуации рассматриваются различные постановки задачи и их решения:

– эксперты зафиксированы, композиция строится как байесовский классификатор над решениями экспертов. Рассматриваются частные случаи с независимыми экспертами и переходом в пространство повышенной размерности, где классификатор оказывается линейным;

– эксперты не зафиксированы, но фиксирована модель композиции – линейная (правило взвешенного голосования). Для этого случая рассмотрен алгоритм бустинга как пример «жадного» алгоритма настройки алгоритмов композиции и ее коэффициентов.

Дополнительно в настоящем разделе рассмотрены способы построения различных экспертов с использованием одной выборки: бэггинг и метод случайных подпространств. Решения, получаемые с использованием этих подходов и/или их комбинации могут быть использованы в тех методах совместной классификации, где допустимо использовать фиксированных (настроенных) экспертов (см. примеры выше).

7 ОБУЧЕНИЕ С ПОДКРЕПЛЕНИЕМ

Методы обучения с подкреплением – методы машинного обучения, в которых каждое принятое агентом (системой) решение характеризуется наградой, косвенно оценивающей его правильность. При этом само правильное действие агенту не указывается.

7.1 Постановка задачи

Пусть $\mathbf{S}$ – множество состояний (англ.: *state*) некоторого *агента* (англ.: *agent*), который находится в некоторой *среде* (англ.: *environment*) и взаимодействует с ней с помощью множества возможных действий $\mathbf{A}$ (англ.: *action*). Упрощённо, задача *обучения с подкреплением* (англ.: Reinforcement Learning – RL) состоит в построении такого отображения $\mathbf{S} \to \mathbf{A}$, которое для каждого конкретного состояния $s \in \mathbf{S}$ указывало бы то действие $a \in \mathbf{A}$, которое необходимо предпринять в указанном состоянии в некотором смысле оптимальным образом. Построение указанного отображения производится путём анализа последовательности взаимодействия агента со средой посредством действий, последовательно переводящих агента из начального состояния s_{start} в конечное $s_{terminal}$:

$$s_{start} = s_0, a_0, s_1, a_1, \ldots s_{N-2}, a_{N-2}, s_{N-1} = s_{terminal}. \qquad (7.1)$$

Такую последовательность обозначают τ и называют *траекторией* (англ.: *trajectory*). Процесс взаимодействия схематически изображен на рисунке 7.1.

Учитывая возможную статистическую природу перехода состояний при конкретных действиях агента, для решения указанной проблемы обычно используют аппарат Марковских процессов принятия решений:

$$<S, A, P, R, \gamma>. \tag{7.2}$$

Здесь P определяет вероятности перехода из состояния в состояние при конкретных действиях в некоторый момент времени t, характеризующие среду в которой находится и действует агент, т.е.:

$$P_a\left(s_{perv}, s_{next}\right) = P_a\left(s_{t+1} = s_{next} \middle| s_t = s_{perv}, a_t = a\right).$$

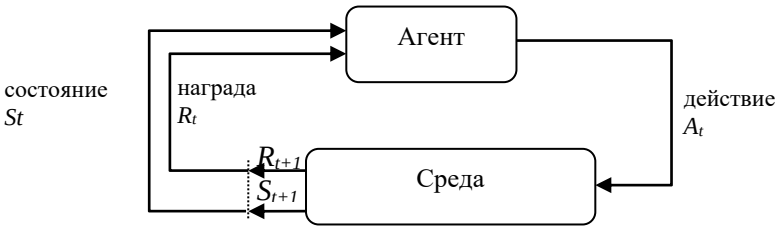


Рисунок 7.1 – Иллюстрация взаимодействия агента со средой

Величины $r_t \equiv R_a\left(s_t, s_{t+1}\right)$ характеризует «*награду*» (англ.: *reward*) за переход из состояния s_t в состояние s_{t+1} под действием a_t. Они используются для формальной записи целевой функции RL-задачи, обозначающей *доходность/полезность* и определяемой как сумма *будущих дисконтированных вознаграждений*:

$$R(\tau) = \sum_{t=0}^{T} r_t \gamma^t . \tag{7.3}$$

Здесь величина γ – *коэффициент дисконта* (англ.: *discount factor*), суть которого – придать большие веса тем наградам (и, как следствие, действиям), которые были получены (выполнены) ранее. Формулу (7.3) можно рассматривать как с конечным ($T<\infty$), так и бесконечным числом наград (и как следствие, состояний агента): $T=\infty$. Как правило, в первом случае полагают $\gamma\equiv 1$, на английском ситуацию и награду именуют *finite-horison undiscounted return*), во втором $\gamma\in\mathbf{R}[0,1)$, и величину (7.3) называют на английском *infinite-horison discounted return.*

Учитывая сделанные обозначения и пояснения, траекторию (7.1) можно переписать следующим образом, включив награды агенту в момент перехода в очередное состояние:

$$\tau:\ s_{start}=s_0,\underset{r_0}{a_0},s_1,\underset{r_1}{a_1},s_2\ldots s_{N-2},\underset{r_{N-1}}{a_{N-2}},\underset{r_{N-2}}{s_{N-1}}=s_{terminal}. \qquad (7.4)$$

В рамках представленного формализма *RL-задача* заключается в нахождении т.н. *политики (*англ.: *policy)* или *стратегии* $\pi:\mathbf{A}\times\mathbf{S}\to\mathbf{R}[0,1]$, характеризующей вероятность (предпочтительность) действия a_t в каждый момент времени и состояния:

$$s_t:\ \pi(a,s)=\Pr(a_t=a|s_t=a). \qquad (7.5)$$

Более «прямолинейным» решением является построение детерминированной стратегии $\pi:\mathbf{S}\to\mathbf{A}$, определяющей оптимальное действие в конкретном состоянии: $\pi(s)=a$.

Построение оптимальное стратегии обычно производится с использованием *функций полезности*. *Функция полезности состояния* (англ.: *state-value function*) $V_\pi:\mathbf{S}\to\mathbf{R}$ в RL-задачах определяется как *ожидаемый доход* (7.3), получаемый в результате следования стратегии π, начиная с*j* стартового состояния *s*. Фактически, данная функция указывает полезность соответствующего состояния:

$$V_\pi(s)=E\{R|s_0=s\}=E\left\{\sum_{t=0}^{\infty} r_t\gamma^t|s_0=s\right\}. \quad (7.6)$$

Здесь $E\{\ldots\}$ – оператор математического ожидания, связанный с распределением траекторий со стартовым состоянием s.

Один из возможных путей решения RL-задачи заключается в построении т.н. *функции полезности действия* (англ.: *action-value function*) или *Q-функции*, которая задаётся как ожидаемое значение целевой функции (7.3) для конкретной пары состояние-действие и фиксированной стратегии π:

$$Q^\pi(s,a)=E\{R|s_t=s,a_t=a,\pi\} \quad (7.7)$$

и удовлетворяет уравнению Беллмана. Отношение между этими двумя функциями полезности достаточно очевидны:

$$V^\pi(s)=\max_{a\in\mathbf{A}} Q^\pi(s,a) \quad \forall s\in\mathbf{S}.$$

Также достаточно очевидно, что в случае известной оптимальной функции полезности действия $Q^*(s,a)$, *оптимальная стратегия* (англ.: optimal policy), являющаяся решением RL-задачи, определяется следующим образом:

$$\pi^*(s)=\arg\max_{a\in\mathbf{A}} Q^*(s,a). \quad (7.8)$$

Классификация RL-методов

Итерация стратегий и итерация (функций) полезности

При *итерации стратегий* первоначально выбирается случайная стратегия и, следуя ей, производится оценка функции полезности. Затем определяется новая стратегия, которая соответствует построенной на предыдущем шаге функции полезности.

Процесс продолжается до нахождения оптимальной стратегии. То есть в данном типе методов стратегия модифицируется/корректируется напрямую.

$$\pi_0 \xrightarrow{E} V^{\pi_0} \xrightarrow{I} \pi_1 \xrightarrow{E} V^{\pi_1} \xrightarrow{I} \pi_2 \xrightarrow{E} \ldots \xrightarrow{I} \pi^* \xrightarrow{E} V^*.$$

При *итерации полезности* первоначально выбирается случайная функция полезности, затем ее значения обновляются. Процесс продолжается до нахождения оптимальной функции полезности. Суть подхода в том, что стратегия определяется неявно, следуя оптимальной функции полезности.

Примеры RL-методов

Наиболее известными RL-методами (исторически) являются:

– метод Q-learning [79], разработанный в 1989 году и используемый в случае небольшого числа состояний и действий и заключающийся в итерационном формировании оценок $Q(s,a)$ величин в соответствующей таблице;

– метод REINFORCE, предложенный в 1992 году и реализующий градиентный метод над параметрически заданной политикой π [80] (метод относится к классу методов TD – temporal difference);

– алгоритм Actor-Critic, предложенный в 1999 [81] и использующий независимо функцию политики (π-функцию) и функцию оценок (Q-функцию). Функция политики именуется как *актёр*, потому что она используется для выбора действий, а Q-функция известна как *критик*, потому что она «критикует» (оценивает) действия, совершённые актёром. Такое разбиение позволило улучшить получаемые решения (метод также относится к классу методов TD);

– Double Q-learning [82], предложенный в 2011 году и являющаяся модификацией базового алгоритма: он использует две Q-функции, одна из которых определяет действие, а вторая – оценку его «полезности»;

– метод Deep Q-learning или DQN, который был предложен в 2014 году фирмой DeepMind (дочерняя фирма корпорации Google) и использующий в качестве отображения Q глубокую сверточную нейронную сеть;

– развитие метода Actor-Critic: алгоритмы A2C (англ. A2C – Advantage Actor Critic) и его асинхронная версия A3C [83], предложенные в 2016 году. Одном из последних и наиболее совершённых на текущий момент считается метод Soft Actor-Critic, разработанный в 2018 году [84].

Некоторые из указанных алгоритмов рассмотрены далее.

7.2 Метод Q-learning и его модификации

7.2.1 Метод Q-learning

Q-learning – RL-алгоритм, определяющий требуемое действие *a* в соответствии с выражением (7.6), где вместо оптимальной Q-функции Q^* используется формируемая им оценка. Алгоритм был предложен Крисом Уоткинсом в 1989 году [79], а доказательство его сходимости было дано Уоткинсом и Питером Даяном в 1992 году. Алгоритм не использует модель среды, что относит его к без-модельным (англ.: off-model), и предполагает *конечное* (*дискретное*) множество возможных состояний и действий агента. Это позволяет продемонстрировать принцип работы алгоритма с использованием таблицы, приведенной на рисунке 7.2, которая и является оценкой Q-функции.

Алгоритм Q-learning

Перед началом обучения все ячейки Q-таблицы инициализируется произвольным фиксированным значением (выбираемым с учетом априорных данных о задаче, при их отсутствии – нулем). Затем в каждый момент времени *t* агент, находясь в состоянии s_t,

выбирает действие a_t и переходит в новое состояние s_{t+1} (которое зависит как от предыдущего состояния s_t, так и от выбранного действия a_t и реакции на это действие среды) и получает вознаграждение $r_t \equiv R_a(s_t, s_{t+1})$.

Initialized

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	327	0	0	0	0	0	0
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	499	0	0	0	0	0	0

Training

Q-Table		Actions					
		South (0)	North (1)	East (2)	West (3)	Pickup (4)	Dropoff (5)
States	0	0	0	0	0	0	0
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	328	-2.30108105	-1.97092096	-2.30357004	-2.20591839	-10.3607344	-8.5583017
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	499	9.96984239	4.02706992	12.96022777	29	3.32877873	3.38230603

Рисунок 7.2 – Иллюстрация к алгоритму Q-learning.
Перевод терминов: initialized – инициализированная, training – обучение, states – состояния, actions – действия, Q-table – Q-таблица

Ядром Q-алгоритма является уравнение Беллмана, используемого для простого обновления Q-функции с использованием средневзвешенной текущей оценки суммарного дохода и новой награды:

$$Q(s_t,a_t)\leftarrow Q(s_t,a_t)+\alpha\left(r_t+\gamma\max_{a\in\mathbf{A}}Q(s_{t+1},a)-Q(s_t,a_t)\right)= \\ =(1-\alpha)Q(s_t,a_t)+\alpha\left(r_t+\gamma\max_{a\in\mathbf{A}}Q(s_{t+1},a)\right). \tag{7.9}$$

Здесь величина $\alpha\in\mathbf{R}(0,1]$ имеет смысл *коэффициента скорости обучения* (англ.: *learning rate*) и определяет, насколько быстро новая информация будет менять текущие полученные оценки Q-функции. Как видно из приведенного выражения, новое значения $Q(s_t,a_t)$ есть взвешенная с учетом этого коэффициента сумма двух величин:

– полученной ранее оценки суммарной награды (дохода) $Q(s_t,a_t)$ и

– текущей награды, определяемой как сумма награды r_t, полученной только что, и дисконтированной на величину γ ожидаемой награды на следующем шаге s_{t+1} (состояние, в котором мы уже оказались), выполнив действие a_t: с учетом стратегии максимизации ожидаемой награды ее величина может быть оценена как $\max_{a\in\mathbf{A}}Q(s_{t+1},a)$.

Алгоритм Q-learning завершает свою работу, когда мы достигаем некоторого предопределенного терминального состояния $s_{terminal}$ (7.8). Однако алгоритм может быть использован и для случая, когда конечное состояние может отсутствовать, то есть в режиме бесконечного обучения. Для гарантии сходимости алго-

ритма достаточно лишь требования $\gamma<1$, что обеспечивает конечность суммарной награды, дохода (как следствие конечности суммы геометрической прогрессии).

Сходимость алгоритма и скорость обучения

Скорость обучения $\alpha\in\mathbf{R}(0,1]$ определяет, в какой степени вновь полученная информация о награде переопределяет старую оценку дохода. Значение коэффициента $\alpha=0$ соответствует ситуации, когда агент ничему не учиться (исключительно используя предварительные знания о Q-функции), в то время как $\alpha=1$ заставляет агента учитывать только самую последнюю информацию, игнорируя всю предшествующую информацию об оценке. В полностью детерминированных средах скорость обучения $\alpha=1$ является оптимальной. Когда же задача является стохастической, алгоритм сходится при некоторых дополнительных условиях на скорость обучения: необходимо, чтобы она уменьшилась до нуля (см. методы стохастической аппроксимации [2-3]). На практике часто используется постоянная скорость обучения, например, $\alpha=0.1$.

Коэффициент дисконта

Коэффициент дисконта определяет важность будущих вознаграждений. Значение $\gamma=0$ сделает агента «близоруким», то есть учитывающим только текущие вознаграждения. Напротив, значение $\gamma\approx1$ заставит алгоритм стремиться к длительному обучению и высокому вознаграждению. Если коэффициент дисконта $\gamma\geq1$, алгоритм может вообще не достигнуть конечного состояния, поскольку последовательность оценок полезности с аддитивными, недисконтированными вознаграждениями становится бесконечной. Даже при коэффициенте дисконтирования лишь немного меньшем, чем «1», обучение Q-функции приводит к нестабильно-

стям при обучениям с использованием искусственной нейронной сетью (см. раздел 7 настоящего пособия).

Начальные условия

Поскольку Q-learning является итеративным алгоритмом, он неявно предполагает наличие начальной оценки. Есть разные стратегии инициализации.

а) Высокие начальные значения в Q-таблице, также известные как «оптимистичные начальные условия», могут стимулировать исследование: независимо от того, какое действие выбрано, правило обновления приведет к тому, что получаемая награда будет иметь более низкие значения, чем существующая альтернатива из начальных условий, тем самым увеличивая вероятность их выбора.

б) Первая награда может быть использована для сброса и переинициализации начальных приближений. Согласно этой идее, при первом выполнении действия вознаграждение используется для определения значения ячейки таблицы. Это позволяет обучаться немедленно в случае фиксированных детерминированных вознаграждений. Также ожидается, что модель, которая включает *сброс начальных условий* (англ.: RIC – reset of initial conditions), будет лучше предсказывать поведение участников, чем модель, которая предполагает любое произвольное начальное условие (англ.: AIC – arbitrary initial conditions).

7.2.2 Модификация метода: DQN

Deep Q-learning – реализация метода Q-learning с использованием глубоких сверточных нейронных сетей, примененная компанией DeepMind (в 2014 году приобретена Google). Q-таблица представляет собой матрицу, для работы с которыми удобна сверточная нейронная сеть. Основной проблемой при такой реализации является нестабильность обучения.

7.2.3 Двойное Q-обучение

В выражении (7.9) будущее максимальное приблизительное значение действия в Q-learning оценивается с использованием той же Q-функции, что и в текущей стратегии выбора действий, соответственно: $\max_{a \in \mathbf{A}} Q(s_{t+1}, a)$ и $Q(s_t, a_t)$.

В «шумной»/сложной среде такая реализация Q-learning иногда может переоценивать значения действий, замедляя обучение. Для исправления этого был предложен вариант алгоритма, называемый двойным Q-обучением (англ.: *Double Q-learning)*. В этой реализации алгоритма для оценки доходности используется другая стратегия, чем та, которая используется для выбора следующего действия.

На практике две отдельные Q-функции – Q^A и Q^B - обучаются взаимно симметричным образом с использованием отдельных опытов/примеров. Шаг обновления двойного Q-обучения выглядит следующим образом:

$$Q_{t+1}^A(s_t, a_t) = Q_t^A(s_t, a_t) + \\ + \alpha_t(s_t, a_t)\left(r_t + \gamma Q_t^B\left(s_{t+1}, \arg\max_{a \in \mathbf{A}} Q_t^A(s_{t+1}, a)\right) - Q_t^A(s_t, a_t)\right),$$

$$Q_{t+1}^B(s_t, a_t) = Q_t^B(s_t, a_t) + \\ + \alpha_t(s_t, a_t)\left(r_t + \gamma Q_t^A\left(s_{t+1}, \arg\max_{a \in \mathbf{A}} Q_t^B(s_{t+1}, a)\right) - Q_t^B(s_t, a_t)\right).$$

Теперь оценочная стоимость дисконтированного будущего оценивается с использованием другой стратегии, что решает проблему переоценки.

Этот алгоритм был также объединен с глубоким обучением (см. п.7.2.2), что привело к алгоритму Double DQN, который превосходит оригинальный алгоритм DQN.

7.2.4 TD-обучение

Метод временных различий (англ.: **T**emporal Difference learning) относится к классу методов без использования моделей среды и применяется для *оценки функции полезности состояний* для процесса с конкретной стратегией.

Перепишем выражение (7.6) следующим образом:

$$V_\pi(s)=E_\pi\left\{\sum_{t=0}^{\infty} r_t\gamma^t \mid s_0=s\right\}=E_\pi\left\{r_0+\sum_{t=1}^{\infty} r_t\gamma^t \mid s_0=s\right\}=r_0+\gamma V_\pi(s_1).$$

Таким образом $r_0+\gamma V_\pi(s_1)$ является ожидаемой оценкой величины $V_\pi(s)$ для конкретной стратегии π с учетом принятой марковской модели. Тогда текущую оценку функции полезности можно скорректировать, «добавив» расхождение оценок на соседних шагах, а именно:

$$V_\pi(s)\leftarrow V_\pi(s)+\beta\left(\left(r_0+\gamma V_\pi(s_1)\right)-V_\pi(s)\right), \qquad (7.10)$$

где $\beta\in\mathbf{R}(0,1)$ – коэффициент скорости обучения. Величина $\left(r_0+\gamma V_\pi(s_1)\right)-V_\pi(s)$, стоящая в правой скобке, и есть временная разность оценок, положенная в название алгоритма.

Переписав алгоритм в другом виде:

$$V_\pi(s)\leftarrow(1-\beta)V_\pi(s)+\beta\left(r_0+\gamma V_\pi(s_1)\right), \qquad (7.11)$$

можно видеть, что по сути алгоритм производит усреднение текущих и предсказываемых оценок функции полезности состояний.

7.3 Алгоритм REINFORCE

Алгоритм REINFORCE относится к градиентным алгоритмам, использующим метод Монте-Карло при оценке величины градиента целевой функции. Учитывая градиентную природу алгоритма, стратегия задается параметризованной функцией $\pi_\theta(a|s)$ с параметрами θ. В качестве целевой функции выступает ожидаемая величина дохода (7.3) для конечной суммы с единичным коэффициентом дисконта:

$$\Im(\theta)=E_{\pi_\theta}\left\{\sum_{t=0}^{T-1}R_{a_t}(s_t,s_{t+1})\right\}=E_{\pi_\theta}\left\{R(\tau)\right\}.$$

Математическое ожидание здесь берется по распределению траекторий, получаемых с использованием стратегии π_θ с заданными параметрами θ (альтернативная форма записи $E_{\tau\sim p_\theta(\tau)}\{\ldots\}$).

Мы можем максимизировать целевую функцию, используя *градиентный алгоритм (градиентный подъем)*, а именно:

$$\theta\leftarrow\theta+\rho\cdot\nabla_\theta\Im(\theta)=\theta+\rho\nabla_\theta E_{\pi_\theta}\left\{R(\tau)\right\}, \qquad (7.12)$$

здесь $\rho>0$ – шаг градиентного алгоритма. Для применения этого алгоритма необходимо получить явное выражение градиента целевой функции. Как известно из курса теории вероятностей, математическое ожидание произвольной дискретной случайной величины X определено как:

$$E\{X\}=\sum_i x_i P(X=x_i)=\sum_i x_i P(x_i),$$

Перепишем выражение градиента критерия, как показано ниже (рассмотрение непрерывного случая производится полностью аналогично, то есть формальной заменой суммы на интеграл):

$$\nabla_\theta \Im(\theta) = \nabla_\theta E_{\pi_\theta}\{R(\tau)\} = \nabla_\theta \sum_\tau R(\tau) P(\tau|\theta).$$

Последнее выражение можно преобразовать следующим образом:

$$\nabla_\theta \sum_\tau R(\tau) P(\tau|\theta) = \sum_\tau R(\tau) \nabla_\theta P(\tau|\theta) = \sum_\tau R(\tau) P(\tau|\theta) \frac{\nabla_\theta P(\tau|\theta)}{P(\tau|\theta)} = ...$$

Для дальнейших преобразований используем то известное свойство производной логарифма, что:

$$\frac{\nabla_\theta z(\theta)}{z(\theta)} = \nabla_\theta \log(z(\theta)).$$

Тогда:

$$... = \sum_\tau P(\tau|\theta) R(\tau) \nabla_\theta \log(P(\tau|\theta)) = E_{\pi_\theta}\{R(\tau) \nabla_\theta \log(P(\tau|\theta))\}. \quad (7.13)$$

Распределение вероятности $P(\tau|\theta)$ траектории для конкретного θ может быть представлена в виде (учитываем марковость рассматриваемого процесса):

$$P(\tau|\theta) = p(s_0) \prod_{t=0}^{T-1} P\left(s_{t+1}^\tau | s_t^\tau, a_t\right) \pi_\theta\left(s_t^\tau, a_t\right).$$

Здесь $p\left(s_0^\tau\right)$ – распределение вероятностей начального состояния, а $P\left(s_{t+1}^\tau | s_t^\tau, a_t\right)$ – вероятность перехода в новое состояние s_{t+1}^τ путем выполнения действия a_t из состояния s_t^τ, характеризуется средой.

Следуя соотношение (7.16), рассмотрим далее логарифм представленного выражения:

$$\log P(\tau|\theta)=\log p\left(s_0^\tau\right)+\sum_{t=0}^{T-1}\left(\log P\left(s_{t+1}^\tau|s_t^\tau,a_t\right)+\log\pi_\theta\left(s_t^\tau,a_t\right)\right)=$$

$$=\left[\log p\left(s_0^\tau\right)+\sum_{t=0}^{T-1}\log P\left(s_{t+1}^\tau|s_t^\tau,a_t\right)\right]+\sum_{t=0}^{T-1}\log\pi_\theta\left(s_t^\tau,a_t\right).$$

Если взять градиент от правой и левой части, получим:

$$\nabla_\theta\log P(\tau|\theta)=\nabla_\theta\left[\log p\left(s_0^\tau\right)+\sum_{t=0}^{T-1}\log P\left(s_{t+1}^\tau|s_t^\tau,a_t\right)\right]+\sum_{t=0}^{T-1}\nabla_\theta\log\pi_\theta\left(s_t^\tau,a_t\right).$$

Учитывая, что первое слагаемое, приведенное в квадратных скобках и включающее априорные вероятности начальных состояний $p\left(s_0^\tau\right)$ и вероятности перехода между состояниями $P\left(s_{t+1}^\tau|s_t^\tau,a_t\right)$, определяется моделью среды и не зависит от параметров θ стратегии, его градиент оказывается нулевым:

$$\nabla_\theta\left[\log p\left(s_0^\tau\right)+\sum_{t=0}^{T-1}\log P\left(s_{t+1}^\tau|s_t^\tau,a_t\right)\right]=0.$$

В результате:

$$\nabla_\theta\log P(\tau|\theta)=\sum_{t=0}^{T-1}\nabla_\theta\log\pi_\theta\left(s_t^\tau,a_t\right). \qquad (7.14)$$

Подставляя полученное выражение в формулу (7.13) для градиента целевой функции, получаем:

$$\nabla_\theta\Im(\theta)=E_{\pi_\theta}\left\{R(\tau)\nabla_\theta\log\left(P(\tau|\theta)\right)\right\}=E_{\pi_\theta}\left\{R(\tau)\sum_{t=0}^{T-1}\nabla_\theta\log\pi_\theta\left(s_t^\tau,a_t\right)\right\}. \qquad (7.15)$$

Вычислить теоретическое значение градиента, используя полученное выражение, затруднительно. Однако, возможно оценить значение градиента численно, используя метод Монте-Карло: то есть заменить оператор математического ожидания на усреднение значений по множеству траекторий Ξ:

$$\nabla_\theta \Im(\theta) \approx \frac{1}{|\Xi|} \sum_{\tau \in \Xi} \left(R(\tau) \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta \left(s_t^\tau, a_t^\tau \right) \right), \qquad (7.16)$$

здесь $|\Xi|$ – мощность множества траекторий (их количество).

Псевдокод алгоритма

0. Задаем начальное значение параметров θ стратегии.

1. Для текущей стратегии π_θ формируем множество Ξ_N, применяя стратегию с произвольных начальных состояний.

2. Вычисляем оценку градиента целевой функции по соотношению (7.16).

3. Обновляем параметры стратегии по соотношению (7.12).

4. Повторяйте 1-3, пока не получим оптимальную стратегию π_θ (значение дохода и вектор параметров перестанут меняться).

Достоинства и недостатки алгоритма

Достоинства

а) алгоритм реализуем как для дискретного, так и для непрерывного множества действий,

б) как любой градиентный алгоритм, имеет гарантию сходимости (при верно подобранном шаге алгоритма).

Недостатки

а) ресурсозатратность, объясняемая сразу рядом обстоятельств:

– как любой градиентный алгоритм – долгое время работы (шаг мал для гарантии сходимости);

– большое число взаимодействия со средой: для одного шага обновления параметров (7.12) необходимо взаимодействовать со средой $|\Xi| T$, где $|\Xi|$ - количество траекторий, T – длина траектории;

– большая дисперсия слагаемых в (7.16), что требует увеличения числа траекторий $|\Xi|$;

– построенные траектории повторно не используются.

б) Как для любого градиентного алгоритма, сходимость гарантирована к локальному минимуму (для алгоритма Q-learning – к глобальному).

7.4 Метод Actor-Critic

Алгоритм является развитием алгоритма REINFORCE и схематично может быть изображен так, как показано на рисунке 7.3.

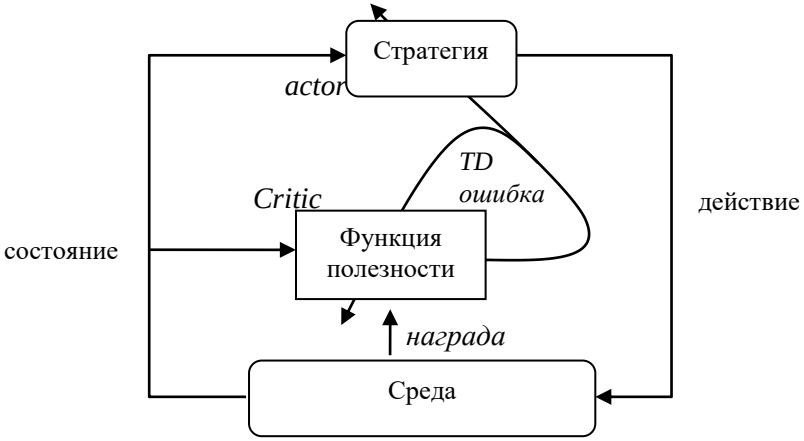


Рисунок 7.3 – Иллюстрация к алгоритму Actor-Critic

7.4.1 Обоснование использования опорного значения

Как было указано выше, метод Reinforce имеет существенные недостатки, к которым относятся высокая зашумленность градиента и его большая дисперсия. Рассмотрим метод Actor-Critic, индуцирующий ряд алгоритмов типа Actor-Critic, который в значительной степени от этих недостатков избавлен. Указанный метод использует т.н. *опорное значение* (англ.: *baseline*) полезности для снижения дисперсии. Объясним на примере.

Пусть в выражении (7.16) величины $R(\tau)$ и $\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta \left(s_t^\tau, a_t \right)$ для трех траекторий имеют, соответственно, значения:

$R(\tau)$	0.5	0.2	0.3
$\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta \left(s_t^\tau, a_t \right)$	1000	1001	1002

Тогда дисперсия соответствующих величин 5000, 2002, 3006 при среднем 3336 получается равной 23286.8. В то же время, если скорректировать значения наград, вычтя из них среднее значение 1001, то дисперсия получаемых величин 0.5, 0 и 0.3 будет всего 0.1633! Указанный пример показывает целесообразность коррекции значений наград с использованием некоторого опорного значения b:

$$R(\tau) \leftarrow R(\tau) - b. \tag{7.17}$$

Однако, использование этого приема будет обосновано, если удастся доказать, что такая коррекция наград не приведет к смещению получаемого решения. Это можно сделать, показав, что величина градиента (7.15) в результате модификации (7.17) не претерпит изменений, то есть доказать равенство:

$$E_{\pi_\theta}\left\{R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} = E_{\pi_\theta}\left\{\left(R(\tau)-b\right)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\}. \quad (7.18)$$

Правая часть этого выражения равна:

$$E_{\pi_\theta}\left\{R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} - b\cdot E_{\pi_\theta}\left\{\nabla_\theta \log\left(P(\tau|\theta)\right)\right\}$$

и для доказательства (7.19) достаточно показать, что

$$E_{\pi_\theta}\left\{\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} = 0 .$$

Имеем

$$E_{\pi_\theta}\left\{\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} = \sum_\tau P(\tau|\theta)\nabla_\theta \log\left(P(\tau|\theta)\right) =$$

$$= \sum_\tau \nabla_\theta P(\tau|\theta) = \nabla_\theta \sum_\tau P(\tau|\theta) = \nabla_\theta 1 = 0.$$

Что доказывает равенство (7.18). При этом дисперсия градиента целевой функции в результате модификации (7.17) становится равной (используем известное соотношение для дисперсии $Var\{X\} = Var\left\{X^2\right\} - \left(Var\{X\}\right)^2$, здесь $Var\{\ldots\}$ – оператор дисперсии):

$$Var_{\pi_\theta}\left\{\left(R(\tau)-b\right)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} =$$

$$= E_{\pi_\theta}\left\{\left(\left(R(\tau)-b\right)\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} - \left(E_{\pi_\theta}\left\{\left(R(\tau)-b\right)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\}\right)^2 =$$

$$= E_{\pi_\theta}\left\{\left(\left(R(\tau)-b\right)\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} - \left(E_{\pi_\theta}\left\{R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\}\right)^2 =$$

$$= E_{\pi_\theta}\left\{\left(R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} - \left(E_{\pi_\theta}\left\{R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\}\right)^2 -$$

$$-2bE_{\pi_\theta}\left\{R(\tau)\left(\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} + b^2 E_{\pi_\theta}\left\{\left(\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} =$$

$$= Var_{\pi_\theta}\left\{R(\tau)\nabla_\theta \log\left(P(\tau|\theta)\right)\right\} -$$

$$-2bE_{\pi_\theta}\left\{R(\tau)\left(\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\} + b^2 E_{\pi_\theta}\left\{\left(\nabla_\theta \log\left(P(\tau|\theta)\right)\right)^2\right\}$$

и будет зависеть от величины b. Таким образом, можно получить меньшую дисперсию и, как следствие, лучшую оценку приближения с использованием метода Монте-Карло.

Выбор же конкретной величины опорного значения приводит к различным алгоритмам типа Actor-Critic. В частности:

Таблица 7.1. **Варианты RL-алгоритмов типа Actor-Critic**

Величина награды $\left(R(\tau)-b\right)$	Алгоритм типа Actor-Critic
$R(\tau)$	REINFORCE
$Q_{\pi}(s,a)$	Q-Actor-Critic
$A_{\pi}(s,a)$	Advantage Actor-Critic (A2C)
δ	TD Actor-Critic

Скорректированная (с учетом опорного значения) величина награды по сути оценивает текущую настроенную стратегию, что и привело к именованию алгоритма как Actor-Critic:

– критик оценивает функцию полезности. При этом может использоваться как функция полезности состояния, так и функция полезности действия;

– актор преобразует стратегию таким образом, чтобы «угодить» критику.

Рассмотрим указанные в таблице 7.1 алгоритмы.

7.4.2 Алгоритм Q-Actor-Critic

Вернемся еще раз к выражению (7.15):

$$\nabla_\theta \Im(\theta) = E_{\pi_\theta}\left\{R(\tau)\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t\right)\right\} =$$

$$= E_{\pi_\theta}\left\{\sum_{t=0}^{T-1} R_{a_t}\left(s_t^\tau, s_{t+1}^\tau\right)\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t\right)\right\} =$$

$$= E_{\pi_\theta}\left\{\sum_{t=0}^{T-1}\left[\nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t\right)\cdot\sum_{\tilde{t}=0}^{T-1} R_{a_{\tilde{t}}}\left(s_{\tilde{t}}^\tau, s_{\tilde{t}+1}^\tau\right)\right]\right\}.$$

Так как в некоторый момент времени t от действия a_t зависят только награды $R_{a_{\tilde{t}}}\left(s_{\tilde{t}}^\tau, s_{\tilde{t}+1}^\tau\right)$ для $\tilde{t} \geq t$, то мы можем отбросить постоянные слагаемые наград до этого момента в силу доказанного соотношения (7.18). Имеем:

$$\nabla_\theta \Im(\theta) = E_{\pi_\theta}\left\{\sum_{t=0}^{T-1}\left[\nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t\right)\cdot\sum_{\tilde{t}=t}^{T-1} R_{a_{\tilde{t}}}\left(s_{\tilde{t}}^\tau, s_{\tilde{t}+1}^\tau\right)\right]\right\}. \quad (7.19)$$

Величина $\sum_{\tilde{t}=t}^{T-1} R_{a_{\tilde{t}}}\left(s_{\tilde{t}}^\tau, s_{\tilde{t}+1}^\tau\right)$, по сути, есть суммарный выигрыш (доход), полученный для пары состояние-действия $\left(s_t^\tau, a_t\right)$, что совпадает со значением функции полезности действия (7.7). Тогда вид (7.19) будет:

$$\nabla_\theta \Im(\theta) = E_{\pi_\theta}\left\{\sum_{t=0}^{T-1}\nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t\right) Q_{\pi_\theta}\left(s_t^\tau, a_t\right)\right\}. \quad (7.20)$$

7.4.3 Алгоритм Advantage Actor-Critic (A2C)

Алгоритм Advantage Actor-Critic (A2C) или алгоритм актора-критика с преимуществом использует несколько иное базовое зна-

чение, уменьшив потенциальный выигрыш $Q_{\pi_\theta}\left(s_t^\tau, a_t\right)$ на функцию полезности состояния $V_{\pi_\theta}\left(s_t^\tau\right)$:

$$A_{\pi_\theta}\left(s_t^\tau, a_t\right) = Q_{\pi_\theta}\left(s_t^\tau, a_t\right) - V_{\pi_\theta}\left(s_t^\tau\right). \tag{7.21}$$

Эту величину авторы алгоритма назвали «*преимуществом*» (англ.: advantage). При этом оценка функции полезности состояния $V_{\pi_\theta}\left(s_t^\tau\right)$ в этом алгоритме выполняется с использованием выражения (7.10) TD-метода:

$$V_\pi(s) \leftarrow V_\pi(s) + \beta\left(\left(r_0 + \gamma V_\pi(s_1)\right) - V_\pi(s)\right), \tag{7.22}$$

Оценка величины преимущества:

$$A_{\pi_\theta}\left(s_t^\tau, a_t\right) = R_{a_t}\left(s_t^\tau, s_{t+1}^\tau\right) + V_{\pi_\theta}\left(s_{t+1}^\tau\right) - V_{\pi_\theta}\left(s_t^\tau\right). \tag{7.23}$$

Подставим (7.21) в оценку градиента (7.16), оставив одну траекторию τ:

$$\nabla_\theta \Im(\theta) \approx \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta\left(s_t^\tau, a_t^\tau\right) A_\pi\left(s_t^\tau, a_t^\tau\right). \tag{7.24}$$

Окончательно, алгоритм A2C может быть представлен следующим образом.

Псевдокод алгоритма

0. Задаем начальное значение параметров θ стратегии.

1. При нахождении в текущем состоянии s_t выполняем действие a_t в соответствие с текущей стратегией π_θ, переходим в состояние s_{t+1} и получаем вознаграждение $R_{a_t}\left(s_t, s_{t+1}\right)$.

2. Выполняем коррекцию функции полезности состояния по соотношению (7.22).

3. Обновляем оценка величины преимущества по соотношению (7.23).

4. Рассчитываем значение градиента после выполнения 1 и 2 для каждого состояния текущей траектории τ .

5. Обновляем параметры стратегии по соотношению (7.12).

6. Повторяйте 1-5 для новой траектории τ пока не получим оптимальную стратегию π_θ (значение дохода и вектор параметров перестанут меняться)..

7.4.4 Алгоритм Asynchronous Advantage Actor-Critic (A3C)

Данный алгоритм не предполагает математических изменений алгоритма A2C. Суть алгоритма A3C – асинхронный запуск различных экземпляров алгоритма A2C из различных начальных состояния, но с одинаковыми параметрами стратегии. Каждый отработавший на очередной траектории алгоритм производит коррекцию параметров стратегии, которую в дальнейшем и используют остальные экземпляры. Алгоритм позволяет в условиях имитационной среды на высокопроизводительных параллельных вычислительных системах ускорить процесс получения решения. Для реальных сред, где существенным моментом является процесс взаимодействия со средой и физическим актором(ами), преимущество данного подхода не столь однозначно.

7.5 Результаты седьмого раздела

В рамках данного раздела рассмотрены методы обучения с подкреплением (RL-методов) – такие методы машинного обучения, в которых каждое принятое агентом (системой) решение ха-

рактеризуется наградой, косвенно оценивающей его правильность. Нами рассмотрены несколько наиболее известных методов:

– метод Q-learning и его модификации (DQN и двойного Q-обучения). Метод Q-learning рассчитан на дискретный конечный набор состояний и действий, что позволяет корректировать таблицу Q-функции;

– алгоритм REINFORCE, который является прямой реализацией градиентного алгоритма применительно к проблеме RL-обучения. Хотя алгоритм REINFORCE имеет все недостатки, присущие градиентным алгоритмам (в частности, медленная скорость работы и, как следствие, большое число взаимодействий со средой), на основе этого алгоритма были построены более совершенные и быстрые алгоритмы, рассмотренные ниже;

– метод Actor-Critic (AC), который развивает идею алгоритма REINFORCE, но находит способ получить оценку производной при каждом наблюдении. Реализация AC-метода включает сразу несколько алгоритмов, отличающихся способом «аппроксимации» производной: алгоритм Q-Actior-Critic, алгоритм Advantage Actior-Critic (A2C), алгоритм Asynchronous Advantage Actior-Critic (A3C).

Указаны достоинства и недостатки приведенным методов/алгоритмов решения проблемы RL-обучения.

8 ИСКУССТВЕННЫЕ НЕЙРОННЫЕ СЕТИ. КРАТКОЕ ВВЕДЕНИЕ

Искусственная нейронная сеть (ИНС) – это параметрически заданная функция, реализующая отображение множества входных сигналов во множество выходных с использованием модели, составленной из множества связанных между собой элементов – *стандартных формальных нейронов или искусственных нейронов.*

8.1 Стандартный формальный нейрон

Искусственный нейрон имитирует в первом приближении работу биологического нейрона. На вход искусственного нейрона поступает некоторое множество сигналов x_1, x_2,…, x_n,, обозначаемое вектором **x**, каждый из которых является выходом другого нейрона. Каждый сигнал умножается на соответствующий вес w_1, w_2,…, w_3, аналогичный «синаптической силе» биологической связи, и поступает на суммирующий блок, обозначенный **Σ** на рисунках 8.1а и 8.1б. Суммирующий блок складывает взвешенные входы алгебраически, создавая выход, который обозначается «s». В векторных обозначениях преобразование может быть записано следующим образом:

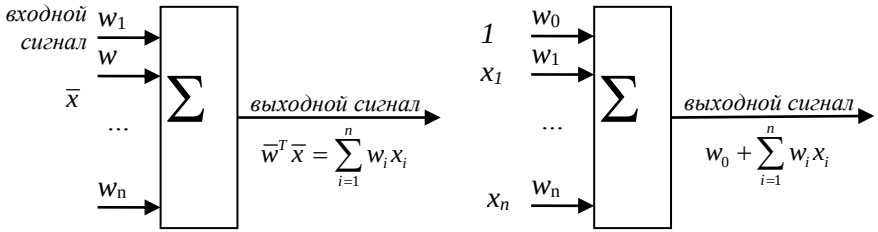
$$s = \overline{w}^T \overline{x} \ .$$

Этот элемент искусственного нейрона носит название адаптивный сумматор**.** *Адаптивный сумматор* вычисляет скалярное произведение вектора входного сигнала *x* на вектор параметров. Адаптивным его называют из-за наличия вектора настраиваемых

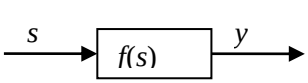
параметров. Последнем преобразованием в искусственном нейроне является нелинейное преобразование выхода сумматора, которое выполняется *активационной функцией* нейрона (рисунок 8.1,в) $\bar{w}\ \bar{w}$. Окончательно, работа формального нейрона описывается следующим выражением:

$$y = f(s) = f\left(\bar{w}^T \bar{x}\right).$$

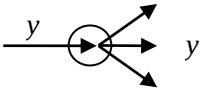
Точка ветвления, показанная на рисунке 8.1,г служит для рассылки выходного сигнала одного нейрона по другим нейронам ИНС. Она получает скалярный входной сигнал y и передает его всем своим выходам. Таким образом, *стандартный формальный нейрон* составлен из входного сумматора, нелинейного преобразователя и точки ветвления на выходе, как показано на рисунке 8.1,д.



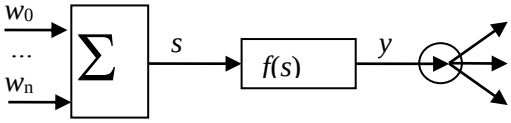
а) адаптивный сумматор; б) неоднородный адаптивный сумматор;



в) активационная функция;



г) точка ветвления;



д) формальный нейрон;

Рисунок 8.1 – К определению формального нейрона как элемента ИНС

Важным моментом при построении ИНС является нелинейность функции активации нейрона. К ней обычно предъявляют следующие требования:

– являются монотонно возрастающими;

– первая производная существует почти всюду;

– производная легко вычисляются через значения самой функции.

Наиболее типичными активационными функциям являются:

– рациональная сигмоида;

$$f(S)=\frac{S}{|S|+a} \text{ или } f(S)=a_1\frac{S}{|S|+a_2}+a_3 .$$

– логистическая функция (или функция Ферми);

$$f(S)=\frac{1}{1+e^{-S}} \text{ или } f(S)=\frac{1}{1+e^{-2aS}}$$

– ReLU:

$$f(s)=\begin{cases} s, & s\geq 0; \\ 0, & s<0. \end{cases}$$

– Leaky ReLU:

$$f(s)=\begin{cases} s, & s\geq 0; \\ 0.01s, & s<0. \end{cases}$$

8.2 Архитектуры ИНС

Несмотря на то, что вариантов построения ИНС из указанных элементов оказывается множество, их все обычно можно поделить на две базовые группы: *слоистые и полносвязные сети.* При этом, если функция активации во всех нейронах ИНС одинакова, то та-

кую сеть называют *однородной (гомогенной)*. Если же f зависит еще от одного или нескольких параметров, значения которых меняются от нейрона к нейрону, то есть называют *неоднородной (гетерогенной)*.

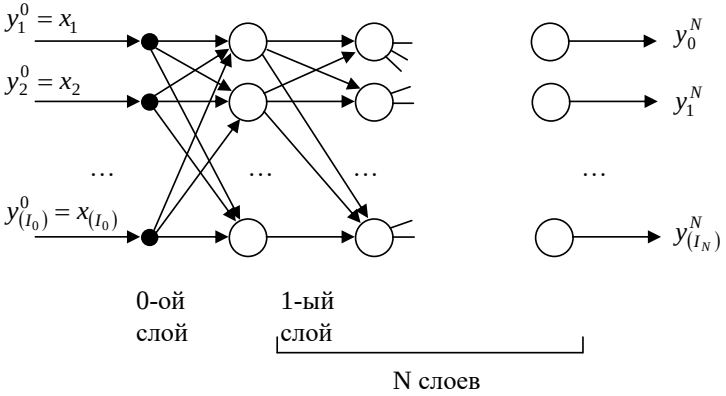


Рисунок 8.2 – Многослойная ИНС или многослойный персептрон

В *слоистых или многослойных сетях* нейроны расположены в несколько слоев, как показано на рисунке 8.2. Нейроны первого слоя получают входные сигналы, преобразуют их и через точки ветвления передают нейронам второго слоя. Далее срабатывает второй слой и т.д. до *k*-го слоя, который выдает выходные сигналы для интерпретатора и пользователя. Если не оговорено противное, то каждый выходной сигнал *i*-го слоя подается на вход всех нейронов $i+1$-го. Число нейронов в каждом слое может быть любым и никак заранее не связано с количеством нейронов в других слоях. Стандартный способ подачи входных сигналов: все нейроны первого слоя получают каждый входной сигнал. Формальное выражение прохождения сигнала для всех нейронов, начиная с первого слоя до последнего $n=1,2,3,\ldots,N$ имеет вид:

$$S_j^{(n)} = \sum_{i \in I_{(n-1)}} w_{ij}^{(n)} y_i^{(n-1)}\ ,\ y_j^{(n)} = f\left(S_j^{(n)}\right), \qquad (8.1)$$

Здесь N – число слоев НС, I_n – множество нейронов в n-ом слое ($n=\overline{1,N}$), w_{ij}^n – синаптический вес j-го нейрона слоя n от i-ого нейрона слоя n-1, $\left(i \in I_{n-1}, j \in I_n\right)$; $y_j^{(n)}$ – выход активационной функции j-го нейрона n-го слоя, $\left(j \in I_n\right)$. $S_j^{(n)}$ – выход сумматора j-го нейрона n-го слоя, $\left(j \in I_n\right)$. Дополнительно, $y_j^{(0)} = x_j$ – входные сигналы, поступающие на вход нейронов первого слоя ($j \in I_0$), $y_j^{(N)}$ – выходные значения сети, ($j \in I_N$),

Особое распространение получили трехслойные сети, в которых каждый слой имеет свое наименование: первый – входной, второй – скрытый, третий – входной.

В *полносвязных сетях* каждый нейрон передает свой выходной сигнал остальным нейронам, включая самого себя (рисунок 8.3). Выходными сигналами сети могут быть все или некоторые выходные сигналы нейронов после нескольких тактов функционирования сети. Все входные сигналы подаются всем нейронам. Формальное выражение перерасчета сигналов нейронов от такта (t-1) к такту t имеет вид:

$$y_j(t) = f\left(\sum_{i \in I} w_{ij} y_i\ (t-1)\right),\ \ j \in I\ .$$

Если полносвязная сеть функционирует до получения ответа заданное число тактов k, то ее можно представить как частный случай k – слойной ИНС, все слои которой одинаковы и каждый из них соответствует такту функционирования полносвязной сети.

Существенное различие между полносвязной и слоистой сетями возникает тогда, когда число тактов функционирования заранее не ограничено – слоистая сеть такая работать не может.

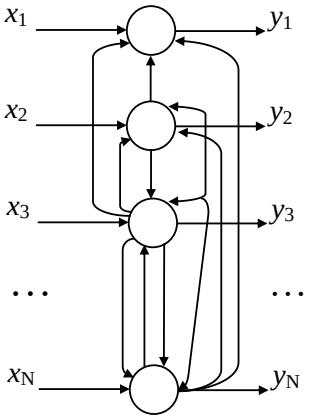


Рисунок 8.3 – Полносвязная ИНС

Наряду с указанными группами сетей еще рассматривают смешанный вариант, так называемые рекуррентные сети. В общем случае, *рекуррентной сеть* – это любая ИНС, где есть обратные связи. То есть любая ИНС, не являющаяся слоистой, может быть определена как рекуррентной. Однако на практике часто под рекуррентной понимают слоистую сеть, где между некоторыми слоями существуют обратные связи.

8.3 Обучения ИНС. Алгоритм обратного распространения ошибки

Одном из наиболее известных алгоритмов обучения ИНС является алгоритм обратного распространения ошибки, относящийся

к классу алгоритмов стохастического градиента, который для слоистой однородной ИНС с логистической активационной функцией представим в виде.

Алгоритм обучения

1. Задать значения весов ИНС случайным образом в диапазоне [-1,1].

2. Подать на вход сети один из *K*-ый образец и рассчитать значения выходов нейронов начиная с 1-го слоя до последнего по формуле (8.1). На основании полученных значений выходов ($y_j^{(n)}$) выполняется шаг 2.

3. Рассчитать значения ошибки $g_j^{(n)}$ (при обработке *k*-го образца) по формуле (начиная с $n=N$ и до 1):

* для $n=N$

$$g_j^N = y_j^N \left(1-y_j^N\right)\left(d_j - y_j^N\right), \tag{8.2}$$

* для $n<N$

$$g_j^n = \underbrace{y_j^n \left(1-y_j^n\right)}_{(n)} \cdot \sum_{i\in I_{n+1}} g_i^{(n+1)} w_{ji}^{(n+1)} \ . \tag{8.3}$$

4. Скорректировать веса $w_{ij}^{(n)}$ (по *k*-ому образцу):

$$w_{ij}^{(n)} = w_{ij}^{(n)}(k-1) - \rho g_i^n \cdot y_i^{(n-1)} \ .$$

5. Протестировать работу сети – это может выполняться периодически через некоторое количество шагов. Если ошибка существенна, то go to 1. В противном случае конец.

Примечание. Сети попеременно предъявляются все тренировочные образцы в случайном порядке, чтобы сеть не «забывала» образцы других классов.

Современные *сверточные* (англ.: convolutional neural net) и *GAN-сети* (англ.: Generative Adversarial Network – генеративно-состязательные сети) являются модификациями представленных классических архитектур ИНС.

8.4 Результаты восьмого раздела

В представленном разделе кратко представлены основы искусственных нейронных сетей. Под искусственной нейронной сетью (ИНС) мы понимаем параметрически заданную функцию, реализующая отображение множества входных сигналов во множество выходных с использованием модели, составленной из множества связанных между собой элементов – *стандартных формальных нейронов или искусственных нейронов.* Представлены основные архитектуры ИНС: слоистые, полносвязные и рекуррентные. Для слоистых сетей детально представлен алгоритм обратного распространения ошибки (без доказательства). Представленный материал достаточен для первичного знакомства с понятиями ИНС и может использоваться как краткое введение в тему для тех, кто намерен начать изучать данный предмет. Расширенное изложение вопросов ИНС представлено в многочисленных монографиях, изданных в России за последнее время, в частности [85–88].

ЗАКЛЮЧЕНИЕ

Настоящее учебное пособие представляет собой краткое изложение материалов по ключевым разделам современной теории искусственного интеллекта: машинному обучению и распознаванию образов (классификации).

Оно включает, в частности, классификацию задач машинного обучения, описание основных методов ранжирования/упорядочивания объектов и методов распознавания образов (классификации): геометрических, статистических и алгебраических, а также кратких основ теории последовательного анализа и классификации, теории обучения с подкреплением и искусственных нейронных сетей.

Важно отметить, что данное учебное пособие может быть использовано совместно с практикумом «Основы статистической теории распознавания образов и машинного обучения. Лабораторный практикум на PYTHON» того же автора, являясь его дополнением. Однако некоторые вопросы современного машинного обучения остались этими двумя изданиями не охваченными. В частности, глубокие нейронные сети и глубокое обучение остались за рамками материала обоих изданий.

БИБЛИОГРАФИЧЕСКИЙ СПИСОК

1. Мясников, В.В. Основы статистической теории распознавания образов. Лабораторный практикум: учебное пособие / В.В. Мясников. – Самара: Изд-во Самар. гос. аэрокосм, ун-та, 2007. – 80 с.
2. Мясников, В.В. Основы статистической теории распознавания образов и машинного обучения. Лабораторный практикум на PYTHON: учебное пособие / В.В. Мясников. – Самара: Издательство Самарского университета, 2023. – 131 с.
3. Методы компьютерной обработки изображений / М.В. Гашников, Н.И. Глумов, Н.Ю. Ильясова [и др.]; под ред. В.А. Сойфера. – Москва: Физматлит, 2001. – 780 с.
4. Preference Learning. Eds. J. Fürnkranz, E. Hüllermeier. – Berlin Heidelberg: Springer-Verlag, 2011.
5. He, DC. Texture Unit, Texture Spectrum, And Texture Analysis / DC. He, L.Wang // IEEE Transactions on Geoscience and Remote Sensing. – 1990. – Vol.28. – P.509 – 512.
6. Цветков, О.В. Вычисление оценки энтропии биосигнала, инвариантной к изменению его амплитуды, с использованием рангового ядра / О.В. Цветков // Изв. вузов. Радиоэлектроника. – 1991. – Т. 34. – № 8. –С. 108–110.
7. Цветков, О.В. Оценка близости числовых последовательностей на основе сопоставления их ранговых ядер / О.В. Цветков // Изв. вузов. Радиоэлектроника. – 1992. – № 8. – С. 28–33.
8. Ojala, T. Performance evaluation of texture measures with classification based on Kullback discrimination of distributions / T. Ojala, M. Pietikäinen, D. Harwood // Proceedings of the 12th IAPR

International Conference on Pattern Recognition (ICPR 1994). – 1994. – Vol.1. – P. 582 – 585.
9. Ojala, T. A Comparative Study of Texture Measures with Classification Based on Feature Distributions / T. Ojala, M. Pietikinen // Pattern Recognition. – 1996. – Vol. 29. – P. 51–59.
10. Pietikäinen, M. Computer Vision Using Local Binary Patterns / M. Pietikäinen, A. Hadid, G. Zhao, T. Ahonen // Computational Imaging and Vision, Series. Springer-Verlag London – 2011. – Vol. 40, P. 212.
11. Brahnam, S. Local Binary Patterns: New Variants and Applications / S. Brahnam, C. Lakhmi, L. Nanni, A. Lumini // Studies in Computational Intelligence. – 2014. – Springer-Verlag Berlin Heidelberg. – Vol. 506.
12. Ojala, T. Multiresolution grayscale and rotation invariant texture classification with local binary patterns / T. Ojala, M. Pietikinen, T. Menp // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 2002 – Vol. 24(7). – P. 971–987.
13. Гончаров, А.В. Исследование свойств знакового представления изображений в задачах распознавания образов / А.В. Гончаров // Известия ЮФУ. Технические науки. – 2009. – Тематический выпуск – С. 178–188.
14. Каркищенко, А.Н. Исследование устойчивости знакового представления изображений / А.Н. Каркищенко // Автоматика и телемеханика. – 2010. – Т. 9. – С. 57–69.
15. Броневич, А.Г. Анализ неопределенности выделения информативных признаков и представлений изображений / А.Г. Броневич, А.Н. Каркищенко, А.Е. Лепский. – Москва: ФИЗМАТЛИТ, 2013. – 320 с.
16. Болдин, М.В. Знаковый статистический анализ линейных моделей Под ред. Е.Ю. Ходан / М.В. Болдин, Г.И. Симонова, Ю.Н. Тюрин. – Москва: Наука. Физмалит, 1997. – 288 с.

17. Мясников, В.В. Локальное порядковое преобразование цифровых изображений / В.В. Мясников // Компьютерная оптика. – 2015. – Т. 39, № 3. – С. 397–405.
18. Bradley, R.A. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons / R.A. Bradley, M.E. Terry // Biometrika. – 1952. – Vol. 39, No 3/4. – P. 324–345.
19. Фишберн, П. Теория полезности для принятия решений. Пер. с англ. / П. Фишберн. – Москва: Наука, 1978. – 352 с.
20. Fürnkranz, J. Pairwise Preference Learning and Ranking / J. Fürnkranz, E. Hüllermeier // In: Lavrač, N., Gamberger, D., Blockeel, H., Todorovski, L. (eds) Machine Learning: ECML 2003. ECML 2003. Springer, Berlin, Heidelberg. Lecture Notes in Computer Science. – 2003. – Vol. 2837. – pp. 8-15.
21. Murphy, K.P. Machine Learning: A Probabilistic Perspective / K.P. Murphy. – MIT Press, 2012. – 1098 p.
22. Tsukida, K. How to Analyze Paired Comparison Data / K.Tsukida and Maya R. Gupta // UWEE Technical Report Number UWEETR-2011-0004, Seattle, Washington 2011. – 27 p.
23. Thurstone, L.L. A law of comparative judgment / L.L. Thurstone // Psychological Review. – 1927. – Vol. 34. – No 4. – P. 273–286.
24. Bradley, R.A. Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons / R.A. Bradley, M.E. Terry // Biometrika. – 1952. – Vol. 39.– No 3/4. – P. 324–345.
25. Saaty, T.L. Relative measurement and its generalization in decision making why pairwise comparisons are central in mathematics for the measurement of intangible factors the analytic hierarchy/network process / T.L. Saaty // RACSAM – Revista de la Real Academia de Ciencias Exactas, Fisicas y Naturales. Serie A. Matematicas. – 2008. – Vol. 102. – No 2. – P. 251–318.
26. Viappiani, P. Preference modeling and preference elicitation: An overview / P. Viappiani // CEUR Workshop Proceedings. – 2014. – Vol. 1278. – P. 19–24.

27. Guo, S. Real-time multiattribute Bayesian preference elicitation with pairwise comparison queries / S. Guo, S. Sanner // Journal of Machine Learning Research. – 2010. – Vol. 9. – P. 289–296.
28. Arentze, T.A. Adaptive personalized travel information systems: A bayesian method to learn users' personal preferences in multimodal transport networks / T.A. Arentze // IEEE Transactions on Intelligent Transportation Systems. – 2013. – Vol. 14. – Adaptive personalized travel information systems. – No 4. – P. 1957–1966.
29. Campigotto, P. Personalized and Situation-Aware Multimodal Route Recommendations: The FAVOUR Algorithm / P. Campigotto, C. Rudloff, M. Leodolter, D. Bauer // IEEE Transactions on Intelligent Transportation Systems. – 2017. – Vol. 18. No 1. – P. 92–102.
30. Zhang, S. Deep Learning based Recommender System: A Survey and New Perspectives / S. Zhang, L. Yao, A. Sun, Y. Tay // ACM Computing Surveys. – 2019. – Vol. 52. No 1. – P. 1-38.
31. Melnikov, V. Pairwise versus pointwise ranking: A case study / V. Melnikov, P. Gupta, B. Frick, D. Kaimann, E. Hüllermeier // Schedae Informaticae. – 2016. – Vol. 25. – P. 73–83.
32. Вапник, В.Н. Теория распознавания образов: Стат. проблемы обучения / В.Н. Вапник, А.Я. Червоненкис. – Москва: Наука, 1974. – 415 с.
33. Вапник, В.Н. Восстановление зависимостей по эмпирическим данным / В.Н. Вапник. – Москва: Наука, 1979. – 448 с.
34. Дуда, Р. Распознавание образов и анализ сцен / Р. Дуда, П. Харт; пер. с англ – М.: Мир, 1976. – 512 с.
35. Ту, Дж. Принципы распознавания образов / Дж. Ту, Р. Гонсалес; пер. с англ. – Москва: Мир, 1978. –412с.
36. Фукунага К. Введение в статистическую теорию распознавания образов / Пер. с англ. – Москва: Наука, 1979. – 368 с.
37. Аркадьев, А.Г. Обучение машины распознаванию образов / А.Г. Аркадьев, Э. М. Браверман. – Москва: Наука, 1964. – 112 с.

38. Барабаш, Ю.Л. Вопросы статистической теории распознавания / Ю.Л. Барабаш, Б.В. Варский, В.Т. Зиновьев. – Москва: Советское радио, 1967. – 399 с.
39. Бонгард, М.М. Проблема узнавания / М.М. Бонгард. – Москва: Физматгиз, 1967.
40. Аркадьев, А.Г. Обучение машины классификации объектов / А.Г. Аркадьев, Э. М. Браверман. – Москва: Наука, 1971. – 192 с.
41. Горелик, А.Л. Методы распознавания/ А.Л. Горелик, В.А. Скрипкин. – 4-е изд. – Москва: Высшая школа, 1984, 2004. – 262 с.
42. Васильев, В.И. Распознающие системы. Справочник/ В.И. Васильев. – 2-е изд. – Киев: Наукова думка, 1983. – 424 с.
43. Фомин, Я.А. Распознавание образов: теория и применения / Я.А. Фомин. – 2-е изд. – Москва: ФАЗИС, 2012. – 429 с.
44. Фу, К. Последовательные методы в распознавании образов и обучении машин / К. Фу; перевод с англ. Э.Ф. Зайцева ; под ред. Л.А. Мееровича и Я.З. Цыпкина. – Москва: Наука, 1971. – 255 с.
45. Журавлев, Ю.И. Избранные научные труды / Ю.И. Журавлев / – Фу, К. – Москва: Магистр, 1998. – 416 с.
46. Рудаков, К.В. Алгебраическая теория универсальных и локальных ограничений для алгоритмов распознавания: дис. док. физ.-мат. наук: 05-13-17 / К.В. Рудаков. – Москва: Вычислительный центр АН СССР, 1992. – 274 с.
47. Kolmogorov, A. On Tables of Random Numbers / A. Kolmogorov // Sankhyā Ser. A. – 1963.- Vol.25. – pp. 369–375.
48. Kolmogorov, A. On Tables of Random Numbers / A. Kolmogorov // Theoretical Computer Science. – 1998. – Vol.207(2). – pp. 387–395.
49. Rissanen, J. Modeling by shortest data description / J. Rissanen // Automatica. – 1978. – Т. 14, вып. 5.
50. Воронцов, К.В. Теория надёжности обучения по прецедентам. Курс лекций [сайт]. – 2011. – http://www.machinelearning.ru

/wiki/index.php?title=Теория_надёжности_обучения_по_прецедентам_(курс_лекций,_К.В.Воронцов) (дата обращения 01.03.2021).
51. Воронцов, К.В. Комбинаторная теория надёжности обучения по прецедентам: дис. … д-а физ.-мат. наук по специальности 05.13.17 / К.В. Воронцов. – Теоретические основы информатики. – Москва, 2010.
52. Cantzler, H. Random sample consensus (ransac) / H. Cantzler // Institute for Perception, Action and Behaviour, Division of Informatics, University of Edinburgh. – 1981.
53. Ho, Tin Kam. The Random Subspace Method for Constructing Decision Forests / Tin Kam Ho // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1998.- Vol.20 (8) – pp. 832–844.
54. Schapire, R.E. The Strength of Weak Learnability / R.E. Schapire // Machine Learning. – 1990. – Vol. 5 (2). – pp. 197–227.
55. Freund, Y. A desicion-theoretic generalization of on-line learning and an application to boosting / Y. Freund, R.E. Schapire // Lecture Notes in Computer Science, Berlin, Heidelberg: Springer Berlin Heidelberg, 1995. – pp. 23–37.
56. Freund, Y. Decision-Theoretic Generalization of On-line Learning and an Application to Boosting / Y. Freund, R.E. Schapire // Journal of Computer and System Sciences. – 1997. – Vol. 55(1). – pp. 119–139.
57. Schapire, R.E. Improved Boosting Algorithms Using Confidence-Rated Predictors / R.E. Schapire, Y.Singer // Machine Learning, 1999. – Vol. 37(3). – pp. 297–336.
58. Viola, P. Rapid object detection using a boosted cascade of simple features / P. Viola, M. Jones // Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001. IEEE Comput. Soc. 1. – 2001.

59. Воронцов, К.В. Лекции по алгоритмическим композициям. Курс лекций [сайт] / К.В. Воронцов. – 2007. http://www.ccas.ru/voron/download/ Composition.pdf (дата обращения 01.03.2021).
60. Franke, J. Comparison of Two Approaches for Combining the Votes of Cooperating Classifiers / J. Franke, E.A. Mandler // Proceedings 11th IAPR International Conference on Pattern Recognition, Conference B: Pattern Recognition Methodology and Systems II. – 1992. – pp. 611–614.
61. Hansen, L.K. Neural network ensembles / L.K. Hansen, P. Salamon // IEEE Transactions on Pattern Analyis and Machine Intelligence. – 1990. – Vol.12(10). – pp. 993–1001.
62. Hashem, L.K. Improving model accuracy using optimal linear combinations of trained neural networks / L.K. Hashem, B. Schmeiser // IEEE Transactions on Neural Networks. – 1995. – Vol.6(3). – pp. 792–794.
63. Ho, J. Decision combination in multiple classifier systems / J. Ho, J. Hull, S.N. Srihari // IEEE Transaction Pattern Analysis and Machine Intelligence. – 1994.- Vol. 16(l). – pp. 66–75.
64. Kittler, J. On combining classifiers / J. Kittler, M. Hatef, R.P.W. Duin, J. Matas // IEEE Transactions on Pattern Analysis and Machine Intelligence. – 1998. – Vol.20(3). – pp. 226–239.
65. Kittler, J. Combining Classifiers / J. Kittler, M. Hater, R. Duin // Proceedings of 13th International Conference on Pattern recognition. Track B: Pattern Recognition and Signal Analysis II. Vienna, Austria. – 1996. – pp. 897–901.
66. Xu, A. Methods of combining multiple classifiers and their applications to handwriting recognition / A. Xu, C. Krzyzak, Y. Suen // IEEE Trans. SMC. – 1992. – Vol. 22(3). – pp. 418–435.
67. Nair, D. Hierarhical, Modular Architectures for Object Recognition by Parts / D. Nair, J.K. Aggarwal // Proc. 13th Int. Confer. On Pattern Recognition 1. Vienna, Austria. 1996. – pp. 601–606.

68. Вальд, А. Последовательный анализ / А. Вальд. – Москва: Физматгиз, 1960.
69. Леман, Э. Проверка статистических гипотез / Э. Леман. – Москва: Наука, 1979.
70. Синдлер, Ю.Б. Метод двухступенчатого статистического анализа и его приложения в технике / Ю.Б. Синдлер. – Москва: Наука, 1973.
71. Башаринов, А.Е. Методы статистического последовательного анализа и их радиотехнические приложения / А.Е. Башаринов, Б.С. Флейшман. – Москва: Сов. Радио, 1962.
72. Абчук, В.А. Поиск объектов / В.А. Абчук, В.Г. Суздаль. – Москва: Сов, радио. 1977.
73. Анисимов, Б.В. Распознавание и цифровая обработка изображений / Б.В. Анисимов, В.Д. Курганов, В.К. Злобин. – Москва: Высшая школа, 1983.
74. Глумов, Н.И. Информационная технология обнаружения объектов на изображении в режиме скользящего окна / Н.И. Глумов, Э.И. Коломиец, В.В. Сергеев // Научное приборостроение. – 1993. – Том 1. – С. 72–88.
75. Casasent, D. New Techniques for Object Detection and Recognition / D. Casasent // Proceedings of The 10th Scandinavian Conference on Image Analysis II. Lappeenranta, Finland. – 1997. – pp. 597–604.
76. Myasnikov, V.V. On the modified quality criterion for a procedure to detect objects in spatially-extended fields / V.V. Myasnikov // Proceedings of the 10th Scandinavian Conference on Image Analysis SCIA'96 1. Lappeenranta, Finland. – 1997. – pp. 405–410.
77. Самарский, А.А. Численные методы / А.А. Самарский, А.В. Гулин. – Москва «Наука», 1989.
78. Breiman, L. Bagging Predictors / L. Breiman // Machine Learning. – 1996. – Vol.24. – pp. 123–140.

79. Watkins, C.J. Learning from Delayed Rewards / C.J. Watkins [Электронный ресурс]. – 1989. – URL: http://www.cs.rhul.ac.uk/~chrisw/thesis.html (дата обращения 20.08.2021).
80. Ronald, W.J. Simple statistical gradient-following algorithms for connectionist reinforcement learning / W.J. Ronald // Machine Learning. –1992. – Vol. 8. – P. 229–256.
81. Sutton, R.S. Policy gradient methods for reinforcement learning with function approximation / R.S. Sutton, D. McAllester, Y. Mansour // Proceedings of the 12th International Conference on Neural Information Processing Systems. –1999. – P. 1057–1063.
82. Hado, H. Double Q-learning / H. Hado // Advances in Neural Information Processing Systems. –2011. – Vol. 23. – P. 26132622.
83. Mnih, V. Asynchronous methods for deep reinforcement learning / V. Mnih, A.P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, K. Kavukcuoglu // Proceedings of The 33rd International Conference on Machine Learning. –2016. – Vol. 48. – P. 1928–1937.
84. Haarnoja, T. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor / T. Haarnoja, A.Zhou, P. Abbeel, S. Levine // Proceedings of the 35th International Conference on Machine Learning. –2018. – Vol. 80. – P. 1861–1870.
85. Розенблатт, Ф. Принципы нейродинамики: Перцептроны и теория механизмов мозга / Перевод с англ. В. Я. Алтаева [и др.]; под ред. [и с предисл.] д-ра физ.-мат. наук С.М. Осовца / Ф. Розенблатт. – Москва: Мир, 1965. – 480 с.
86. Хайкин, С. Нейронные сети : полный курс / С. Хайкин; [пер. с англ. Н.Н. Куссуль, А.Ю. Шелестова]. – Изд. 2-е, испр. – Москва [и др.]: Вильямс, 2008. – 1103 с.
87. Гудфеллоу, Ян. Глубокое обучение / Ян Гудфеллоу, Иошуа Бенджио, Аарон Курвилль; пер. с англ. А.А. Слинкина. – 2-е цв. изд., испр. – Москва: ДМК Пресс, 2018. – 651 с.

88. Николенко, С. Глубокое обучение. Погружение в мир нейронных сетей: 16+ / С. Николенко, А. Кадурин, Е. Архангельская. – Санкт-Петербург: Питер, 2021. – 476 с.

Учебное издание

Мясников Владислав Валерьевич

РАСПОЗНАВАНИЕ ОБРАЗОВ И МАШИННОЕ ОБУЧЕНИЕ. ОСНОВНЫЕ ПОДХОДЫ

Учебное пособие

Редакционно-издательская обработка Л.Р. Дмитриенко

Подписано в печать 22.06.2023. Формат 60x84 1/16.
Бумага офсетная. Печ. л. 12,25.
Тираж 27 экз. Заказ № .

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«САМАРСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ
УНИВЕРСИТЕТ ИМЕНИ АКАДЕМИКА С.П. КОРОЛЕВА»
(САМАРСКИЙ УНИВЕРСИТЕТ)
443086, Самара, Московское шоссе, 34.

Издательство Самарского университета.
443086, Самара, Московское шоссе, 34.