Министерство образования Республики Беларусь
БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Кафедра «Системы автоматизированного проектирования»

МЕТОДЫ РАСПОЗНАВАНИЯ ОБРАЗОВ

Лабораторные работы
по дисциплине
«Алгоритмы распознавания образов: алгоритм секущих плоскостей и алгоритм оптимальной классификации»
для студентов специальности
1–40 01 02 «Информационные системы и технологии (по направлениям)»
специализации 1–40 01 02-01 «Информационные системы и технологии в проектировании и производстве»

М и н с к 2 0 0 4

УДК 681.327

Рассматриваются элементарные алгоритмы распознавания образов: алгоритм секущих плоскостей и алгоритм оптимальной классификации, используемые для создания обучаемых и самообучающихся систем распознавания.

Изложена краткая теория, взятая в основу при разработке алгоритмов, и результаты, полученные при исследовании работы алгоритмов с различным числом объектов в образах. Теоретические выкладки позволяют студентам самостоятельно разработать программное обеспечение и осуществить его тестирование на реальных предлагаемых задачах.

Составитель
И.Л. Ковалева

Рецензенты:
С.В. Абламейко, В.Б. Ковалевский

Лабораторная работа № 1

АЛГОРИТМ СЕКУЩИХ ПЛОСКОСТЕЙ

Цель: изучение алгоритма секущих плоскостей, разработка на его основе обучающейся системы распознавания изображений.

Краткие теоретические сведения

Алгоритм обучения машины «узнаванию» образов, основанный на методе секущих гиперплоскостей, заключается в аппроксимации разделяющей гиперповерхности «кусками» гиперплоскостей и состоит из следующих частей:

А. Обучение (формирование разделяющей поверхности): (1) проведение секущих плоскостей; (2) исключение лишних плоскостей; (3) исключение лишних кусков плоскостей.

Б. Распознавание новых объектов.

Геометрическая иллюстрация алгоритма

Вначале проследим построение алгоритма на условных геометрических иллюстрациях. Предположим, что нам предстоит обучить машину распознаванию трех образов, которые мы условимся называть образами A, B и C. В пространстве рецепторов этим образам соответствуют три неизвестных, но объективно существующих компактных множества точек. На рис. 1.1 эти множества изображены в виде трех областей A, B и C.

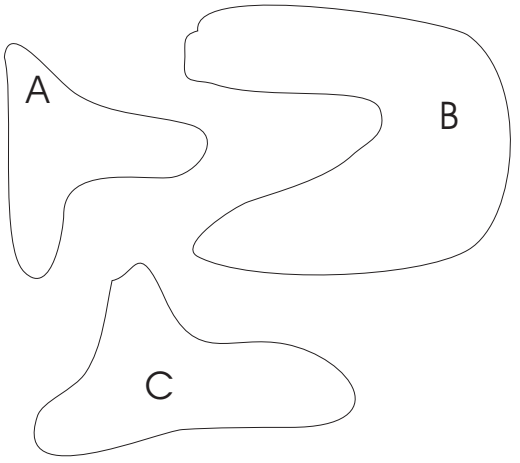

Рис.1.1. Образы A, B и C

Проведение секущих плоскостей

В машину вводятся коды двух точек, принадлежащих разным образам. Машина запоминает координаты точек в пространстве рецепторов (точки 1 и 2 на рис. 1.2) и проводит произвольную плоскость I, разделяющую эти точки. Пространство рецепторов оказывается разбитым на два полупространства, каждое из которых на данном этапе алгоритма отождествляется с одним из двух образов.

Разбиение может оказаться очень неудачным, как это показано на рис. 1.2, где большая часть области B лежит, в результате проведенной нами процедуры, в полупространстве, отнесенном к образу A.

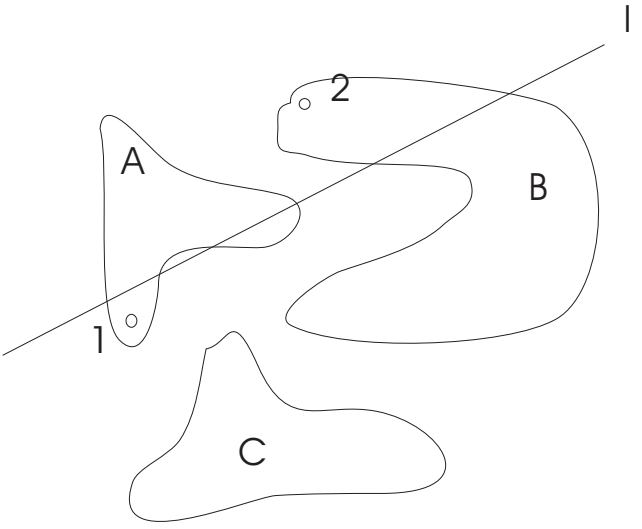

Рис. 1.2. Проведение I секущей плоскости

После проведения первой плоскости в машину вводится третий объект. Возможны два случая:

1) объект относится к образам А или В и попадает в полупространство, отнесенное к «своему» образу. При этом машина, запомнив координаты появившегося объекта, готова к восприятию следующего;

2) объект либо относится к образу С, либо, являясь объектом образов А или В, попадает не в «свое» полупространство. Тогда в одном полупространстве оказываются точки, относящиеся к разным образам. Назовем такой случай противоречием. В нашем случае точка 3, относящаяся к образу А, попадает в полупространство, отнесенное ранее к образу В, и вступает в противоречие с точкой 2 (рис.1.3).

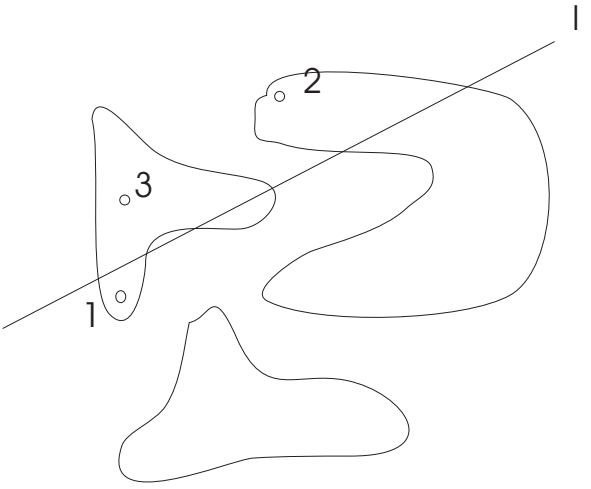

Рис.1.3. Пример противоречия

(Точки, с которыми очередной объект вступает в противоречие, будем называть оппонентами). Машина ликвидирует противоречие, проводя плоскость II, разделяющую оппонент (точку 2) и точку 3. Теперь машина относит к образу А области *DEF* и *DEG*, а к образу В – область GEH (рис. 1.4).

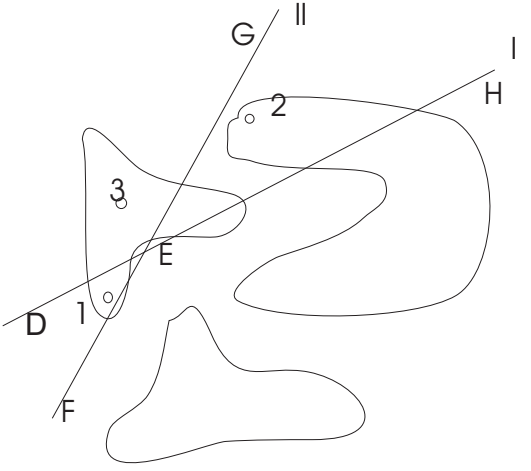

Рис. 1.4. Проведение II секущей плоскости

После проведения второй плоскости число частей, на которое разбивается пространство рецепторов, оказывается больше числа появившихся точек. При проведении последующих плоскостей число частей пространства возрастает чрезвычайно быстро (приблизительно как $2n$, где n – количество плоскостей), значительно быстрее, чем число точек. Поэтому проведение новых плоскостей, с уточнением границы областей A, B, C, одновременно сопровождается появлением большого числа «пустых» частей пространства, которые не могут быть отнесены к какому-либо образу (например, область *FEH* на рис. 1.4). Появление новой точки теперь может привести к трем ситуациям:

1) возникает противоречие;

2) противоречие не возникает, так как точка попадает в «свою» часть пространства;

3) противоречие не возникает, так как точка попадает в «пустую», не поименованную часть пространства. В этом случае машина запоминает координаты новой точки и относит область, в которую попала эта точка, к соответствующему образу.

Именно такая ситуация возникает при появлении точки 4 (рис. 1.5). Не возникает противоречия и при появлении точки 5. Машина запоминает координаты точек 4 и 5, но не проводит новых плоскостей, относя всю область *FEH* к образу C.

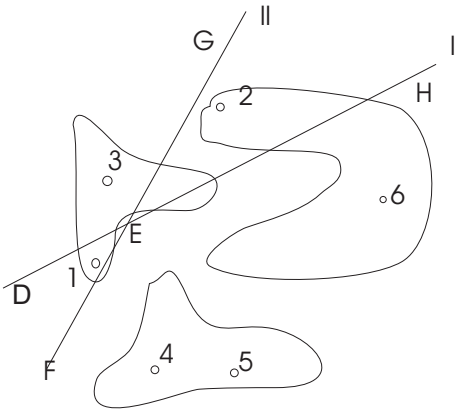

Рис. 1.5. Построение 4, 5 и 6 точек

Появление шестой точки приводит к противоречию, причем у точки 6 оказывается сразу два оппонента – точки 4 и 5. Вначале проводится плоскость III, разделяющая точки 6 и 4, затем плоскость IV, разделяющая точки 6 и 5 (рис. 1.6).

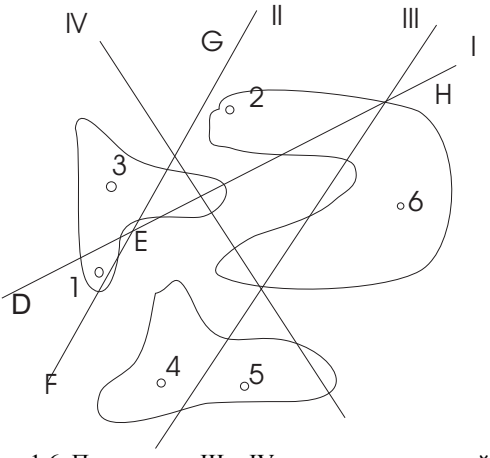

Рис.1.6. Проведение III и IV секущих плоскостей

Точка 7 (рис. 1.7) вступает в противоречие с точкой 6, что приводит к появлению плоскости V (рис. 1.8).

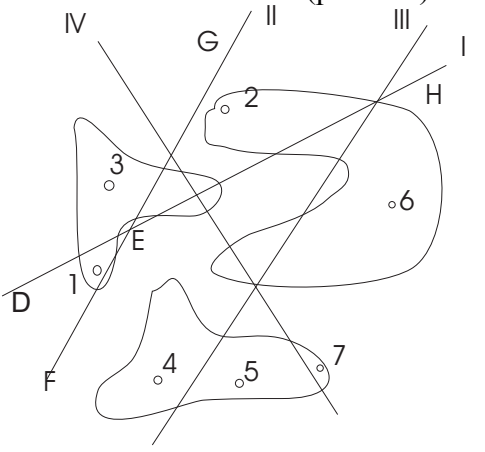

Рис. 1.7. Построение точки 7

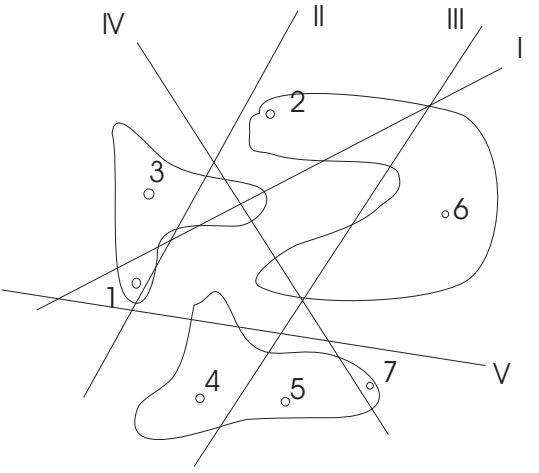

Рис.1.8. Проведение V секущей плоскости

Точка 8 попадает в «пустую» часть пространства, но затем становится оппонентом точки 9 (рис. 1.9)

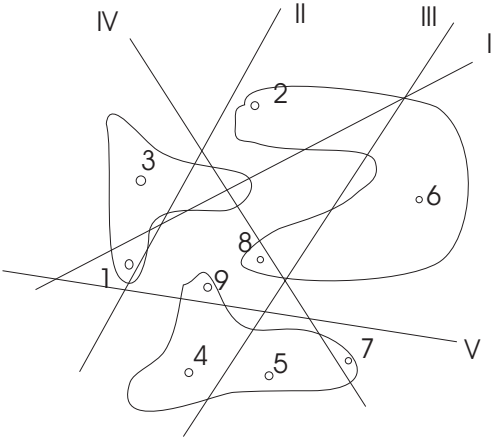

Рис. 1.9. Построение точек 8 и 9

Противоречие ликвидируется плоскостью VI (рис. 1.10). Появление десятой точки, вступающей в противоречие с точкой 8, приводит к появлению плоскости VII (рис. 1.11).

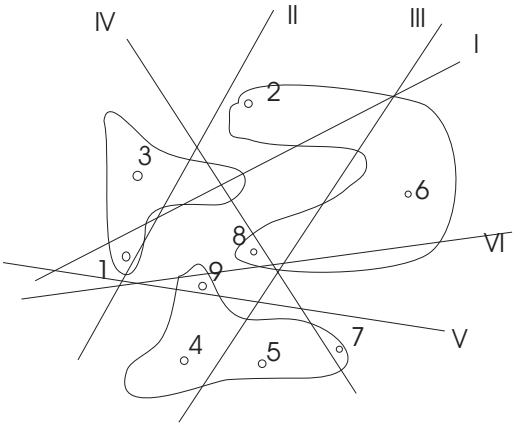

Рис. 1.10. Проведение VI секущей плоскости

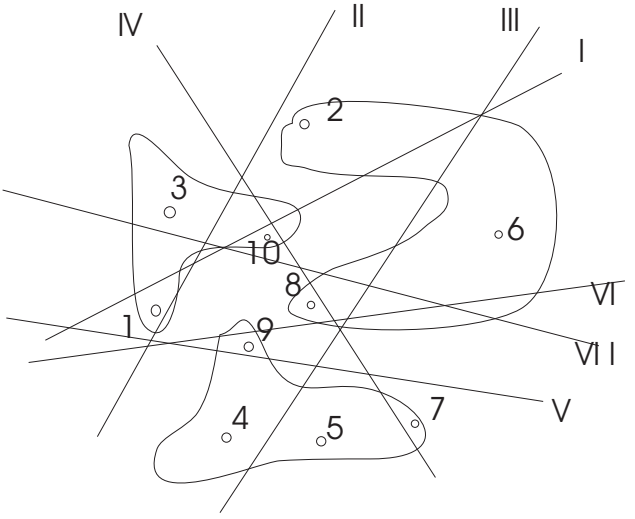

Рис. 1.11. Проведение VII секущей плоскости

Следует обратить внимание на то, что с увеличением числа введенных в машину объектов и количества секущих плоскостей противоречия будут возникать между объектами и оппонентами, лежащими все ближе и ближе к границам областей A, B, C. Область пространства, в которой возможны противоречия, все время сужается, и вероятность возникновения противоречия при появлении нового объекта становится все меньше и меньше, так как убывает число неправильно поименованных областей. Частота возникновения противоречий может, таким образом, служить критерием завершения первой части алгоритма. Закончим первую часть алгоритма проведением плоскости VII. Сложившаяся ситуация приведена на рис. 1.12.

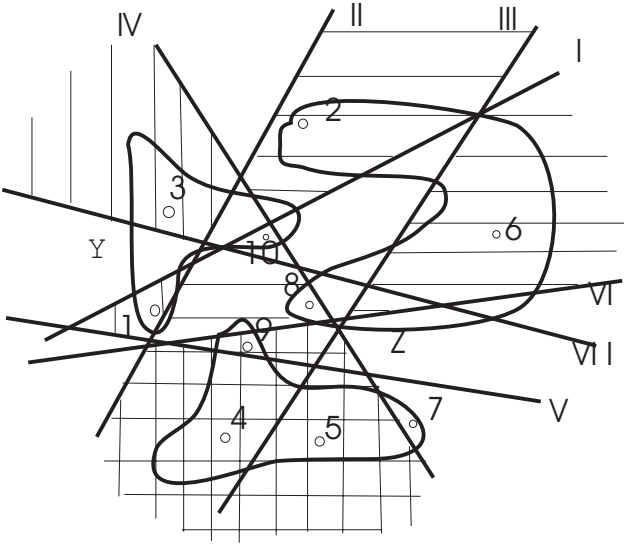

Рис. 1.12. Разбиение пространства на ряд областей

Пространство разбито на ряд областей. Некоторые из них содержат хотя бы одну точку и отнесены машиной к соответствующим образам. Эти области отмечены на рис. 1.12 различ-

ной штриховкой. Остальные области (их гораздо больше, чем первых) остались «пустыми» и ни к какому образу не отнесены.

Совершенно ясно, что на этом нельзя заканчивать процесс обучения. Подлежащая определению новая точка может попасть в «пустую» область, и ее нельзя будет отнести ни к какому образу. Необходимо каким-то способом поименовать все без исключения области пространства. Продолжение первой части алгоритма, разумеется, не может привести к этой цели, так как число «пустых» областей будет все быстрее и быстрее опережать число появившихся точек. Обратимся к основной нашей идее – предположению о компактности множеств точек, соответствующих образам. Исходя из этого предположения естественно отнести «пустые» области к тем же образам, что и какая-нибудь из соседних поименованных областей. При этом такая манипуляция с «пустыми» областями, окруженными одноименными частями пространства, будет совершенно однозначна и не приведет к ошибкам (например, пустая область у на рис. 1.12 вполне определенно относится к образу А).

Только при операциях с областями, лежащими у границ множеств, будут возможны неоднозначность и ошибки. «Пустая» область z (см. рис. 1.12), например, может быть ошибочно отнесена к образу С.

Расширение поименованных частей пространства можно представить как выбрасывание кусков плоскостей, отделяющих «пустые» области от соседних поименованных. Разумеется, как лишние, могут выбрасываться также и куски плоскостей, разделяющие одноименные области. Оставшиеся после такой операции куски плоскостей и образуют искомую разделяющую гиперповерхность.

Весьма важно выбрасывать куски плоскостей более или менее равномерно по пространству. Не соблюдая этого правила, можно прийти к заведомо неправильной разделяющей поверхности. Если, например, начав с некоторой поименованной области, присоединить к ней все соседние «пустые», к образовавшейся области – все «пустые», соседние с ней и т. д., можно почти все

пространство отнести к тому же образу, что и исходная область, ибо после окончания первой части алгоритма поименованные области, как редкие островки, «затеряны» среди областей не поименованных. На рис. 1.13 заштрихована область, которую можно отнести к образу С, если действовать таким методом, начиная с одной из поименованных областей этого образа.

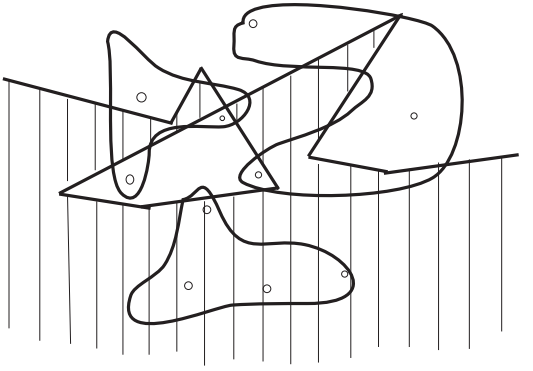

Рис. 1.13. Область, отнесенная к образу С

Для того чтобы обеспечить более или менее равномерное выбрасывание кусков плоскостей, можно прибегнуть к следующему приему. Вначале следует выяснить, нет ли плоскостей, которые целиком состоят из лишних кусков, и, если такие найдутся, выбросить эти плоскости. Затем сначала выбросить «лишние» куски одной из оставшихся плоскостей, потом другой и т.д. Можно доказать, что при этом поименованными окажутся все части пространства.

Исключение лишних плоскостей

Вернемся к рис. 1.12. Легко видеть, что плоскости I, III и V могут быть исключены целиком. Они не содержат кусков, выбрасывание которых привело бы к появлению противоречий. Исключив последовательно эти три плоскости, придем к ситуации, изображенной на рис. 1.14. Ни одна из оставшихся плос-

костей не может быть выброшена целиком. Каждая из них содержит хотя бы один кусок, исключение которого привело бы к противоречию.

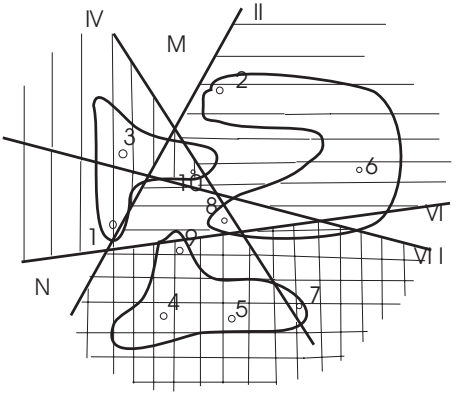

Рис. 1.14. Исключение лишних плоскостей

Исключение лишних кусков плоскостей

Проверяя поочередно все куски плоскости II, затем все куски плоскости IV и т. д. и выбрасывая те куски, исключение которых не приводит к противоречиям, придем к ситуации, при которой все пространство разделено на поименованные области (рис. 1.15).

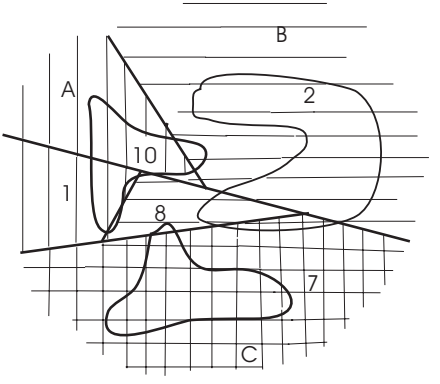

Рис. 1.15. Исключение лишних кусков плоскостей

Процесс обучения закончен. Теперь, чтобы узнать очередной объект, машине достаточно определить, в какой из областей пространства лежит соответствующая ему точка. Впрочем, машина может и ошибиться. Как видно из рис. 1.15, разбиение получилось далеко не идеальным. Значительная часть области В отнесена к образу С, а «кусочки» областей А и С попали в образ В. Это произошло потому, что в ходе обучения машины не встретились точки, расположенные в этих частях пространства. По всей видимости, если процесс обучения был бы более длительным, относительная площадь неправильно поименованных кусков (а значит, и вероятность в будущем ошибок при распознавании) была бы меньше. Однако увеличение длительности обучения приводит к расширению требуемого объема памяти машины. Есть другие способы повышения надежности распознавания. О них будет сказано далее.

Описание алгоритма

Машина, естественно, оперирует в действительности не с чертежами, а с числами. Проследим действия машины на том же примере (см. рис. 1.1).

Проведение секущих плоскостей

Запомнив координаты первых двух точек, машина выбирает случайно* n чисел λ_i (i = 1, 2, 3,..., n, где n – размерность пространства рецепторов). Затем машина определяет значение двух сумм:

$$\sigma^{(1)} = \Sigma\lambda_i \times \xi_i^{(1)} \qquad (1.1)$$

или

* Случайный выбор коэффициентов означает, что каждый очередной коэффициент может с равной вероятностью оказаться любым из некоторого конечного или бесконечного ряда чисел. Для осуществления такого выбора в машинах используют специальные датчики случайных чисел.

$$\sigma^{(2)} = \Sigma\lambda_i \text{ x } \xi_i{}^{(2)}, \qquad (1.2)$$

где $\xi_i^{(1)}$ – координаты первой, а $\xi_i^{(2)}$ – координаты второй точки. После этого машина выбирает, также случайным образом, некоторое число λ_{n+1}, большее, чем меньшая из сумм $\sigma^{(1)}$ и $\sigma^{(2)}$, но меньшее, чем большая из этих сумм. Если образовать две новые суммы

$$\sigma^{(1)} = \Sigma\lambda_i \text{ x } \xi_i{}^{(1)} - \lambda_{n+1}; \qquad (1.3)$$

$$\sigma^{(2)} = \Sigma\lambda_i \text{ x } \xi_i{}^{(2)} - \lambda_{n+1}, \qquad (1.4)$$

то, учитывая способ, которым было выбрано число λ_{n+1}, легко видеть, что одна из этих сумм обязательно будет отрицательной, а другая – положительной. Геометрически же это означает, что найденные машиной числа λ_1, λ_2 ,..., λ_{n+1} есть коэффициенты плоскости, разделяющей две наши точки. Действительно, при подстановке в левую часть уравнения этой плоскости

$$\Sigma\lambda_i \text{ x } \xi_i - \lambda_{n+1} = 0 \qquad (1.5)$$

координат одной из точек получается отрицательный результат, а при подстановке координат другой точки – положительный. Известно, что подобные результаты полу-чаются в том случае, когда точки лежат по разные стороны от плоскости.

Условимся обозначать положение данной точки относительно плоскости значком «1», если при подстановке координат этой точки в левую часть уравнения плоскости получается положительное число, и значком «0» в противном случае. Будем говорить, что точка имеет знак «1» или знак «0» относительно данной плоскости. Представим часть памяти машины после проведения первой разделяющей плоскости (см. рис. 1.2) в виде табл. 1.1, которую будем в дальнейшем называть таблицей знаков.

Таблица 1.1

Таблица знаков			
Номер точки	Образ	Номер плоскости	
		I	
		Знак точки	
1 2	*A* *B*	0 1	

При появлении следующей (третьей) точки (см. рис. 1.3) машина определяет ее знак относительно первой плоскости и заполняет третью строку таблицы знаков (табл. 1.2).

Таблица 1.2

Таблица знаков				
Номер точки	Образ	Номер плоскости		
		I		
		Знак точки		
1 2 3	*A* *B* *A*	0 1 1		

Затем машина производит поиск оппонента, т. е. поочередно сравнивает последнюю строку таблицы с каждой из предыдущих строк. Противоречием считается совпадение строк, относящихся к разным образам. В данном случае противоречие налицо: точка 3, относящаяся к образу А, попала по ту же сторону от плоскости I, что и относящаяся к образу В точка 2 .

Машина проводит новую плоскость II, разделяющую точки 2 и 3 (см. рис. 1.4) и вычисляет знаки всех занесенных в таблицу точек относительно этой плоскости.

Таблица знаков принимает вид табл. 1.3.

Т а б л и ц а 1.3

Таблица знаков				
Номер точки	Образ	Номер плоскости		
		I	II	
		Знак точки		
1	*A*	0	1	
2	*B*	1	0	
3	*A*	1	1	

Противоречие ликвидировано.

Появляются точки 4, 5 и 6 (см. рис. 1.5). После вычисления их знаков приходим к табл. 1.4.

Т а б л и ц а 1.4

Таблица знаков				
Номер точки	Образ	Номер плоскости		
		I	II	
		Знак точки		
1	*A*	0	1	
2	*B*	1	0	
3	*A*	1	1	
4	*C*	0	0	
5	*C*	0	0	
6	*B*	0	0	

Поиск оппонентов, произведенный после появления точки 4, а затем после появления точки 5, дает отрицательный результат. После появления шестой точки обнаруживается противоречие:

точка 6 (образ B) лежит по ту же сторону от плоскостей I и II, что и точки 4 и 5 (образ C). Проводится плоскость III, разделяющая точки 4 и 6, а затем, после нового поиска оппонента, – плоскость IV, разделяющая точки 6 и 5 (см. рис. 1.6). Знаки всех точек относительно новой плоскости заносятся в табл. 1.5.

Таблица 1.5

Таблица знаков					
Номер точки	Образ	Номер плоскости			
		I	II	III	IV
		Знак точки			
1	*A*	0	1	1	1
2	*B*	1	0	1	0
3	*A*	1	1	1	1
4	*C*	0	0	1	1
5	*C*	0	0	0	1
6	*B*	0	0	0	0

Противоречия вновь ликвидированы.

С появлением новых точек машина продолжает заполнять таблицу знаков. С каждой новой точкой в таблице появляется новая строка, с каждой плоскостью – новый столбец. После завершения первой части алгоритма таблица знаков имеет вид табл. 1.6.

Таблица 1.6

Таблица знаков								
Номер точки	Образ	Номер плоскости						
		I	II	III	IV	V	VI	VII
		Знак точки						
1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
1	*A*	0	1	1	1	0	1	1
2	*B*	1	0	1	0	0	1	0

Окончание табл. 1.6

1	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*
3	*A*	1	1	1	1	0	1	0
4	*C*	0	0	1	1	1	0	1
5	*C*	0	0	0	1	1	0	1
6	*B*	0	0	0	0	0	1	0
7	*C*	0	0	0	0	1	0	1
8	*B*	0	0	1	1	0	1	1
9	*C*	0	0	1	1	0	0	1
10	*A*	0	0	1	1	0	1	0

По своему содержанию табл. 1.6 эквивалентна рис. 1.12. Каждая строка таблицы соответствует одной из заштрихованных на рис. 1.12 частей пространства и представляет собой своеобразный код такой области – выпуклого *n*-мерного многогранника, образованного секущими плоскостями*. Этот код указывает, по какую сторону от каждой из секущих плоскостей лежит многогранник, содержащий данную точку.

Исключение лишних плоскостей

После заполнения таблицы знаков машина переходит ко второй части алгоритма. В начале из таблицы знаков «выбрасывается» столбец, соответствующий плоскости I. Затем производится поиск противоречий, т.е. поочередное сравнение каждой строки со всеми остальными. «Выбрасывание» столбца означает, что его цифры при этом не учитываются. Если противоречий не обнаруживается, т. е. не находится одинаковых строк, относящихся к разным образам, столбец исключается из памяти машины. В противном случае столбец «восстанавливается», т. е. его цифры учитываются в последующих операциях. Машина

* Мы называем многогранниками как замкнутые, так и незамкнутые области пространства, границы которых состоят из кусков плоскостей.

переходит к столбцу II, затем к III и далее до последнего столбца. В нашем случае после выбрасывания лишних столбцов I, III и V таблица знаков принимает вид табл. 1.7.

Таблица 1.7

Таблица знаков					
Номер точки	Образ	Номер плоскости			
		II	IV	VI	VII
		Знак точки			
1	*A*	1	1	1	1
2	*B*	0	0	1	0
3	*A*	1	1	1	0
4	*C*	0	1	0	1
5	*C*	0	1	0	1
6	*B*	0	0	1	0
7	*C*	0	0	0	1
8	*B*	0	1	1	1
9	*C*	0	1	0	1
10	*A*	0	1	1	0

В этой таблице есть полностью совпадающие строки. Это означает, что в пространстве рецепторов есть многогранники, содержащие более чем одну точку (см. рис. 1.14). Такие многогранники могли появиться уже в первой части алгоритма, а после выбрасывания плоскостей число их могло еще более возрасти. Между тем для обозначения каждого многогранника вполне достаточно одной точки. «Лишние» точки (т. е. «лишние» строки таблицы знаков) можно исключить из памяти машины. Поэтому после проверки возможности исключить последний столбец машина переходит к исключению лишних строк. Для этого все строки таблиц, начиная со второй, сравниваются с первой строкой, и совпадающие с ней строки исключаются из таблицы. Затем производится сравнение со следующей строкой и так до

тех пор, пока в таблице не останутся только несовпадающие строки. После выбрасывания лишних строк приходим к табл. 1.8. На рис. 1.14 точки, оставшиеся после исключения лишних строк, зачернены.

Таблица 1.8

Таблица знаков					
Номер точки	Образ	Номер плоскости			
		II	IV	VI	VII
		Знак точки			
1	*A*	1	1	1	1
2	*B*	0	0	1	0
3	*A*	1	1	1	0
4	*C*	0	1	0	1
7	*C*	0	0	0	1
8	*B*	0	1	1	1
10	*A*	0	1	1	0

Исключение лишних кусков плоскостей

Ранее мы условились сначала выбрасывать лишние куски одной плоскости, затем другой и т. д. Рассмотрим плоскость II (см. рис. 1.11). Лишними ее кусками являются границы между многогранниками 2 и M^*, 3 и 10, 4 и N. Если выбросить эти границы, одноименные многогранники 3 и 10 будут объединены в одну область пространства, многогранник M подсоединен к многограннику 2, а многограник N – к многограннику 4.

Мы уже упоминали, что строки таблицы знаков являются кодами именованных многогранников. Цифры этих кодов указывают, по какую сторону от каждой плоскости лежит

* Мы обозначаем именованные многогранники по номеру одной из содержащихся в них точек, а «пустые» – буквами.

многогранник и все заключенные в нем точки пространства. Многогранники, разделенные только одним куском одной плоскости, имеют коды, отличающиеся только одним разрядом, причем этот разряд соответствует разделяющей их плоскости. Это легко проследить по рис. 1.14 и табл. 1.8 на примере многогранников 3 и 10, 10 и 8, 8 и 4, 4 и 7.

Забежим вперед и рассмотрим работу машины при распознавании новых объектов. При появлении нового объекта машина, очевидно, должна вычислить его знаки относительно всех плоскостей и сравнить полученный код со всеми строками таблицы знаков. Если точка, соответствующая новому объекту, попадет, например, в многогранник 2, ее код совпадает со второй строкой таблицы знаков и объект будет отнесен к образу *B*. Объект, точка которого попадет в присоединенный к образу *B* многогранник *M*, т. е. будет иметь код, отличающийся от кода многогранника 2 первым (соответствующим плоскости II) разрядом, также должен быть отнесен к образу *B*. Иными словами, выбрасывание куска плоскости II между многогранниками 2 и *M* эквивалентно утверждению, что первый разряд кода второго многогранника не является существенным и может не учитываться при распознавании новых объектов. Достаточно совпадения остальных разрядов кода, чтобы объект был отнесен к образу *B*.

Выбрасывание всех указанных выше кусков плоскости II означает, что во втором столбце таблицы знаков оказываются несущественными цифры, лежащие в 2, 3, 4, 7 и 10-й строках. Цифры 1-й и 8-й строк – существенны, так как соответствующие им первый и восьмой многогранники относятся к разным образам и разделяющий их кусок плоскости II (см. рис. 14) выбросить нельзя. Невозможность исключения плоскости II видна и из табл. 1.8. Если исключить из нее второй столбец, то относящиеся к разным образам многогранники 1 и 8 будут иметь одинаковые коды, что, безусловно, недопустимо.

Из сказанного ясно, что выбрасывание лишних кусков плоскостей сводится к составлению таблицы существенных и несущественных разрядов для всех строк таблицы знаков.

Условимся в этой новой таблице (назовем ее таблицей разрядов) ставить единицу (1) на месте несущественного разряда таблицы знаков и нуль (0) – на месте существенного. Составление таблицы разрядов происходит следующим образом. В таблицу разрядов заносится единица в первую строку первого столбца. Это эквивалентно выбрасыванию куска плоскости II, ограничивающего первый многогранник, т. е. объединению его с многогранником 8. Затем производится проверка законности такого объединения. Проверка заключается в поиске противоречия в таблице знаков, т. е. в сравнении остальных разрядов первой строки с соответствующими разрядами других строк. Если противоречие, т. е. совпадение строк, относящихся к разным образам, не найдено, машина переходит ко второй строке первого столбца таблицы разрядов и заносит в нее единицу. Если противоречие существует, перед переходом ко второй строке единица в первой строке заменяется нулем. Затем машина возобновляет поиск противоречий в таблице знаков и расстановку нулей и единиц в таблице разрядов.

После перехода ко второму и последующим столбцам каждая из двух сравниваемых в данный момент строк может иметь несущественные (т. е. уже отмеченные единицами в таблице разрядов) разряды. Поэтому сравнение ведется только по разрядам, одновременно существенным для обеих сравниваемых строк.

После заполнения последней строки в последнем столбце таблица разрядов примет вид табл. 1.9.

Таблица 1.9

Таблица разрядов					
Номер точки	Образ	Номер плоскости			
		II	IV	VI	VII
		Разряд точки			
1	*2*	*3*	*4*	*5*	*6*
18	*A*	0	1	0	0

Окончание табл. 1.19

1	*2*	*3*	*4*	*5*	*6*
2	*B*	1	0	1	0
3	*A*	1	0	1	0
4	*C*	1	1	0	0
7	*C*	1	1	0	0
8	*B*	0	1	0	0
10	*A*	1	0	1	0

После заполнения таблицы разрядов в таблице знаков могут оказаться строки, отличающиеся только несущественными разрядами (т. е. совпадающие по существенным разрядам). Таким строкам будут соответствовать полностью одинаковые строки таблицы разрядов. В нашем случае это строки 3 и 10, а также 4 и 7. Очевидно, из каждой группы сходных строк могут быть исключены все строки, кроме одной, что и производится после заполнения таблицы разрядов, но поиск совпадающих строк здесь производится только по существенным разрядам. В нашем случае из таблиц исключаются 3-я и 4-я строки. Обе таблицы в окончательном варианте принимают вид табл. 1.10 и 1.11. Процесс обучения закончен.

Т а б л и ц а 1.10

Таблица знаков					
Номер точки	Образ	Номер плоскости			
		II	IV	VI	VII
		Знак точки			
1	*A*	1	1	1	1
2	*B*	0	0	1	0
7	*C*	0	0	0	1
8	*B*	0	1	1	1
10	*A*	0	1	1	0

Таблица 1.11

Таблица разрядов					
Номер точки	Образ	Номер плоскости			
		II	IV	VI	VII
		Разряд точки			
1	*A*	0	1	0	0
2	*B*	1	0	1	0
7	*C*	1	1	0	0
8	*B*	0	1	0	0
10	*A*	1	0	1	0

Все пространство разбито на три области, соответствующие образам A, B и C (см. рис. 1.15). В том, что разбиением охвачено действительно все пространство, и в нем не осталось «пустых» не поименованных областей, можно убедиться следующим образом. Перебрав все возможные четырехзначные коды от 0000 до 1111, можно удостовериться в том, что любой из этих кодов по существенным разрядам совпадет с одной из строк таблицы знаков (см. табл. 1.10). А это и означает, что любая точка в пространстве рецепторов попадет в одну из трех поименованных областей, т. е. что в рассматриваемом случае непоименованные области отсутствуют.

Распознавание новых объектов

При предъявлении нового объекта машина вычисляет его знаки относительно всех плоскостей и полученный код поочередно сравнивает по существенным разрядам со всеми строками таблицы знаков (табл. 1.10). При совпадении строк машина относит новый объект к соответствующему образу.

Секущие плоскости проводятся случайно и независимо друг от друга. Поэтому, если провести обучение несколько раз на одном и том же материале (т. е. осуществить несколько вариантов обучения), будет весьма маловероятным, что ошибки в разбиении пространства рецепторов окажутся одинаковыми во всех вариантах. Следует ожидать, что в каждом из вариантов машина будет ошибаться по-разному. Это дает основание применить метод параллельных вариантов. При использовании этого метода одновременно и независимо друг от друга на одном и том же материале обучаются несколько машин. При узнавании новых объектов каждая машина будет относить эти объекты к какому-то образу, может быть, не к одному и тому же. Окончательное решение принимается «голосованием» машин – объект относится к тому образу, к которому его отнесло большее число машин. Эксперименты показывают, что метод параллельных вариантов весьма эффективен.

Другой способ повышения надежности распознавания состоит в некотором улучшении метода проведения секущих плоскостей. Можно предположить, что если проводить секущие плоскости близко к плоскости, проходящей через середину прямой, соединяющей объект и оппонент и перпендикулярной к этой прямой*, то результирующая разделяющая поверхность будет ближе к истинной границе между образами. Эксперименты подтверждают это предположение.

В экспериментах с «улучшенным» алгоритмом проведение секущих плоскостей происходило следующим образом.

---

* Если в качестве секущих брать плоскости, перпендикулярные к прямой «объект – оппонент» и проходящие через ее середину, то для каждого набора фигур обучение будет происходить строго определенным образом, все варианты будут идентичными и метод параллельных вариантов ничего не даст. Поэтому, даже упорядочивая проведение секущих плоскостей, нужно все же оставить элемент случайности при их выборе.

Выбиралось некоторое число k (его величина уточнялась экспериментом), и после случайного выбора величин λ_i, и вычисления $\sigma^{(1)}$ и $\sigma^{(2)}$ (см. первую часть алгоритма) модуль разности $\sigma^{(1)}$ и $\sigma^{(2)}$ сравнивался с k. Если $|\sigma^{(1)} - \sigma^{(2)}| > k$, выбранные λ_i считались пригодными и вводились в память машины; если $|\sigma^{(1)} - \sigma^{(2)}| \le k$, то λ_i выбирались вновь до тех пор, пока модуль разности $\sigma^{(1)}$ и $\sigma^{(2)}$ не превзойдет k. Кроме того, в качестве свободного члена выбиралось число

$$\lambda_{n+1} = \frac{\sigma^{(1)} + \sigma^{(2)}}{2}. \qquad (1.6)$$

Геометрически эти условия означают, что секущая плоскость SS проходит через середину прямой, соединяющей объект и оппонент, и располагается внутри некоторого угла AOB вблизи перпендикуляра OC к этой прямой (рис. 1.16), причем этот угол тем меньше, чем больше величина k.

В экспериментах с «улучшенным» алгоритмом при $k = 2$ и при $k = 5$ средний процент правильно узнанных фигур повысился почти до 80 % при $k = 2$ и более чем до 85% при $k = 5$. Один из вариантов при $k = 5$ дал надежность распознавания почти 90 %. Такое резкое повышение надежности распознавания за счет более организованного проведения секущих плоскостей говорит о том, что гипотеза компактности справедлива, по крайней мере, в отношении применявшихся образов. В противном случае разные методы проведения плоскостей давали бы примерно одинаковый эффект.

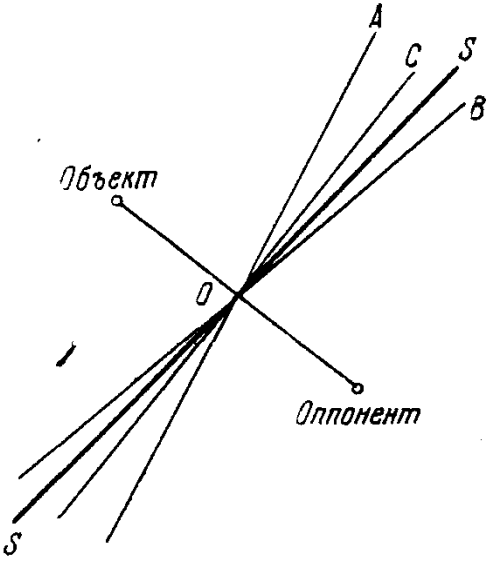

Рис. 1.16. Проведение секущих плоскостей

Метод параллельных вариантов позволяет еще больше повысить надежность распознавания. Применение этого метода к вариантам первоначального алгоритма позволило машине правильно узнать 88,5 % фигур, а когда метод был применен к вариантам улучшенного алгоритма, надежность узнавания возросла до 98,5 %, т. е. машина ошибалась только в трех случаях из двухсот.

Таким образом, эксперименты показали, что метод секущих плоскостей действительно позволяет обучить машину распознаванию сложных фигур. А так как никакие сведения о свойствах фигур машине не сообщались, этот же алгоритм в принципе дает машине возможность обучиться распознаванию широкого класса иных образов, аналогичных по сложности арабским цифрам.

Задание

1. Разработать систему распознавания объектов, реализующую алгоритм секущих плоскостей.
2. Выполнить обучение системы на примере объектов, предложенных преподавателем.
3. Для защиты работы предъявить результаты распознавания системой новых объектов.

Лабораторная работа № 2

АЛГОРИТМ ОПТИМАЛЬНОЙ ОБЪЕКТИВНОЙ КЛАССИФИКАЦИИ

Цель: изучение алгоритма объективной классификации, выполнение с его помощью классификации объектов на незаданное число классов.

Краткие теоретические сведения

Существует достаточно большое количество практических задач, в которых число классов неизвестно и должно быть определено в ходе классификации. В этом случае возникает проблема выбора наилучшей в каком-то смысле классификации. В такой постановке задача классификации может быть отнесена к группе оптимизационных задач. Поэтому для ее решения необходимо определить критерии оптимизации и параметры, варьируя которые можно будет из множества возможных классификаций выбрать одну, наилучшую с точки зрения критерия.

Основываясь на гипотезе компактности образов, кажется естественным считать, что при прочих равных условиях классификация тем лучше, чем ближе друг к другу точки внутри каждого класса. Вместе с тем классификация тем лучше, чем дальше друг от друга классы, получившиеся в результате

классификации. С точки зрения геометрического подхода, принятого в теории распознавания образов, для определения близости точек можно использовать расстояния между ними или их потенциал. Поэтому в качестве меры близости точек внутри класса можно принять «собственный потенциал» класса

$$\Phi(A, A) = \frac{2}{N_A(N_A - 1)} \sum_{j,p;p>j} \Phi(x_j, x_p), \qquad (2.1)$$

где N_A – число объектов в классе, суммирование происходит так, что j принимает все значения от 1 до n, а p – для каждого j все значения, большие j, $N_A(N_A - 1)/2$ – (число сочетаний из N_A по 2) – количество членов суммы. Иными словами, $\Phi(A, A)$ пропорционален сумме потенциалов, взаимно создаваемых друг на друге всеми точками класса. Очевидно, что $\Phi(A, A)$ тем больше, чем теснее между собой расположены точки внутри класса, чем компактнее класс.

Введем величину «среднего собственного потенциала» данной классификации

$$F_1 = \frac{1}{k} \sum_{i=1}^{k} \Phi(A_i, A_i), \qquad (2.2)$$

где k – число классов в классификации. По-видимому, если сравнивать между собой несколько разных классификаций, то величина F_1 будет максимальной у той из них, в которой точки каждого класса в среднем лежат наиболее тесно. Величину F_1 можно принять за первый критерий оптимизации, а ее максимальное значение будет соответствовать классификации, оптимальной с точки зрения этого критерия.

Вторым критерием оптимизации должна быть величина, характеризующая близость классов между собой. В качестве этой величины примем

$$F_2 = \frac{2}{k(k-1)} \sum_{i,j;j>i}^{k} \Phi(A_i, A_j), \qquad (2.3)$$

где k – число классов в классификации, а Φ (A_i, A_j) – мера близости между классами A_i и A_j.Суммирование производится так, что i принимает все значения от 1 до k, а значения j выбираются для каждого i выбираются так, чтобы они были больше i. Если сравнивать между собой несколько разных классификаций, то величина F_2 , очевидно, будет минимальна у той из них, где классы в среднем расположены дальше друг от друга. Величину F_2 можно принять за второй критерий оптимизации, а ее минимальное значение будет соответствовать классификации, оптимальной с точки зрения этого критерия.

При отыскании наилучшей классификации на основе анализа множества альтернативных вариантов необходимо принять решение по выбору оптимального варианта, т. е. решить задачу выработки предпочтения среди некоторого множества альтернативных классификаций. Вариант классификации, выбранный по одному критерию, может оказаться неудовлетворительным с точки зрения другого, не менее важного критерия. Для достижения нужного эффекта принятие окончательного решения предлагается осуществить на основе сравнения классификаций по значениям их обоих критериев.

Такая постановка задачи поиска оптимальной классификации позволяет отнести эту задачу к разряду многокритериальных.

Почти все математические методы оптимизации предназначены для отыскания оптимального решения одной функции – одного критерия. Поэтому одним из вариантов решения многокритериальных задач является ее приведение к однокритериальной с одним обобщенным критерием.

При отыскании наилучшей классификации искомый обобщенный критерий должен быть таким, чтобы наилучшая классификация соответствовала относительно большому F_1

и, одновременно относительно малому F_2 . В качестве такого критерия может быть принят, например,

$$F = F_1 - F_2 . \qquad (2.4)$$

Та классификация из нескольких альтернативных классификаций одной и той же совокупности объектов, для которой критерий F принимает максимальное значение, очевидно, является в определенном смысле объективно оптимальной классификацией.

Задание

1. Разработать программу классификации объектов, реализующую алгоритм оптимальной классификации.
2. Определить оптимальное число классов для объектов, предложенных преподавателем.
3. Для защиты работы предъявить программу и полученные результаты классификации объектов.

Литература

1. Zadeh L.A. Fuzzy Sets. Information and Control. – Vol. 8. – 1965.
2. Аркадьев А.Г., Браверман Э.М. Обучение машины классификации объектов. – М.: Наука, 1981.
3. Айзерман М.А., Браверман Э.М., Розоноэр Л.И. Метод потенциальных функций. – М.: Наука, 1987.

С о д е р ж а н и е

Учебное издание

МЕТОДЫ РАСПОЗНАВАНИЯ ОБРАЗОВ

Лабораторные работы
по дисциплине
«Алгоритмы распознавания образов: алгоритм секущих плоскостей и алгоритм оптимальной классификации»
для студентов специальности
1–40 01 02 «Информационные системы и технологии (по направлениям)»
специализации 1–40 01 02-01 «Информационные системы и технологии в проектировании и производстве»

Составитель КОВАЛЕВА Ирина Львовна

Редактор А.М. Кондратович. Корректор М.П. Антонова
Компьютерная верстка А.Г. Гармазы
Подписано в печать 15.03.2004.
Формат 60 x 84 1/16. Бумага типографская № 2.
Печать офсетная. Гарнитура Таймс.
Усл.печ.л. 2,0. Уч.-изд.л. 1,5. Тираж 100. Заказ 478.
Издатель и полиграфическое исполнение:
Белорусский национальный технический университет.
Лицензия ЛВ № 155 от 30.01.2003. 220013, Минск, проспект Ф.Скорины, 65.