1961

Том VI Б И О Ф И З И К А Вып. 2

Главный редактор А. М. КУЗИН
Зам. главного редактора Г. М. ФРАНК

Редакционная коллегия:

Н. А. АНДРЕЕВ, Х. С. КОШТОЯНЦ, А. В. ЛЕБЕДИНСКИЙ,
Н. Д. НЮБЕРГ, Б. Н. ТАРУСОВ

Ответственный секретарь журнала Л. П. КАЮШИН

Журнал выходит 6 раз в год

СОДЕРЖАНИЕ

Том VI, вып. 2 1961

МОДЕЛИРОВАНИЕ ПРОЦЕССА УЗНАВАНИЯ НА ЦИФРОВОЙ СЧЕТНОЙ МАШИНЕ

М. М. БОНГАРД

Институт биологической физики АН СССР, Москва

Универсальные числовые машины хорошо решают задачи с известным алгоритмом решения. Однако в процессе решения многих задач человеком большой процент работы приходится на действия, описать алгоритм которых он не может. Человек пользуется «интуицией», «рассуждениями по аналогии, «сходством», «выдвигает гипотезу» (формально мало обоснованную) и т. д. Программирование подобных действий встречает затруднение вследствие того, что часть операций, происходящих в мозгу при их выполнении, ускользает от контроля сознания. Настоящая работа посвящена попытке запрограммировать для счетной машины выполнение одной из таких «подсознательных» операций — узнавание.

Постановка вопроса

Пусть мы задумали число 73. Если теперь предложить человеку: «Отгадайте задуманное мною целое положительное число, меньше ста», то лишь в одном проценте случаев задуманное число будет случайно правильно угадано. В то же время, если прибавить: «Задуманное число является элементом множества — 23, 3, 13, 43,...», то процент правильных ответов резко возрастет, хотя, строго говоря, имеется сколько угодно множеств с этими четырьмя элементами, и не содержащих числа 73. В зависимости от своего математического опыта, один увидит здесь ряд чисел оканчивающихся на 3, другой — члены арифметической прогрессии, третий — простые числа. Ни один из них не сможет доказать, что его гипотеза истинна, но всем трем наличие такой гипотезы резко *сократит область поисков*. Каждый из этих людей увидел в совокупности чисел 23, 3, 13, 43 что-то знакомое, «узнал» объект, с которым встречался ранее, и это «узнавание» помогло ему применить накопленный раньше математический опыт [1]. Заметим, что этот опыт не мог быть применен в виде строгого дедуктивного вывода, ибо невозможно доказать принадлежность числа 73 к множеству 23, 3, 13, 43... Нет и не может быть алгоритма, который регулярным образом приводил бы к открытию закона, по которому подобраны элементы множества, и в то же время сам факт решения человеком многих задач показывает, что область поисков (область случайного или систематического перебора) была сильно сокращена. Без такого сокращения сложные задачи вообще не могли бы быть решены в обозримое время.

Ни одну большую и сложную задачу человек не решает как совершенно новую. Он всегда пользуется «аналогиями» с уже решенными задачами, узнает части, «похожие» на уже решенные задачи. При этом очень часто человек не может сказать, почему эта часть показалась

[1] Само собой разумеется, что «узнавание» может пойти и по ложному пути. Например, кто-то может решить, что задуманное число — меньше 50.

ему более похожей на задачу α, чем на задачу β, и все же подмеченное им сходство с задачей α помогло ему решить большую задачу.

Закономерности отождествления «по сходству» минуют наше сознание и в более простых случаях. Вероятно, большинство людей (особенно не художников) не сможет сказать, в чем сходство всех мужских лиц, отличающее их от женских. Несмотря на это, мы прекрасно отличаем мужчин от женщин.

Каким образом мы узнаем наших знакомых на фотографии? Ясно, что это не может осуществляться путем сравнения с хранимыми в памяти эталонами. Во-первых, этих эталонов пришлось бы иметь слишком много. Малейшее изменение ракурса, углового размера, выражения лица, прически, условий освещения и т. д. приводит к *другому* распределению возбуждений на нашей сетчатке. Для узнавания путем сравнения на полное тождество мы должны были бы хранить в памяти виды нашего знакомого при всех вариантах изменения условий наблюдения. Помимо очень нерациональной загрузки памяти, это привело бы также к большому времени узнавания. Ведь нам приходилось бы каждый раз сравнивать с лежащей перед нами фотографией все эталоны из памяти (если заранее неизвестно, что́ будет на фотографии, то нельзя сказать, в каком порядке выгоднее вести сравнение).

Во-вторых, подобная система принципиально не позволяла бы узнавать человека в ракурсе (или освещении и т. д.), в котором мы его раньше не видели.

Для узнавания предмета в новом, не встречавшемся ранее виде, мы должны иметь абстрактное понятие о нем. Абстрактное понятие появилось, конечно, в результате наблюдения предмета в разных видах, но оно охватывает более широкий класс возможных видов, чем тот, на основании которого абстракция возникла. Только при помощи признаков, *не изменяющихся* при изменении условий наблюдения предмета, можно узнать предмет в новом виде. Отсюда следует, что разным возможным видам предмета признаки должны ставить в соответствие одну и ту же характеристику. Другими словами, признак должен осуществлять вырожденное отображение вида объекта. Всюду в дальнейшем мы будем называть *признаком* алгоритм, осуществляющий вырожденное преобразование вида объекта. Результат преобразования мы будем называть *характеристикой объекта по данному признаку.*

Приведем примеры, поясняющие определения:

Проверка материала, из которого сделан предмет,— есть признак. Высказывание: «кастрюля медная» есть характеристика кастрюли по этому признаку.

Проверка четности числа — признак. Высказывание: «это число четное» — характеристика по признаку.

Далеко не все признаки оказываются полезными при узнавании. Если нам нужно узнать знакомого человека среди других людей, то мы никогда не пользуемся признаком «проверка наличия головы». Все люди имеют по этому признаку одну и ту же характеристику — «голова есть». Характеристика по этому признаку не дает сечения на множестве (не отделяет одних людей от других) и поэтому не помогает нам узнать данного человека.

Есть и другая причина, делающая некоторые признаки бесполезными для узнавания данных объектов. Иногда на афишах печатают разные части текста разным цветом. При этом характеристика каждой буквы алфавита по признаку цвета не остается постоянной и вследствие этого не помогает узнавать буквы. Мы будем называть *полезными* признаки, удовлетворяющие следующим двум требованиям: 1) они должны давать сечение на множестве объектов, подлежащих различению (должны существовать, по крайней мере, два объекта, имеющие разные характеристики по этому признаку); 2) характеристика каждого объекта должна быть постоянной, не зависящей от «условий наблюдения». Для простоты мы не останавливаемся на количественных градациях степени полезности признаков. Несколько слов об этом будет сказано при описании той части программы, которая ведет отбор полезных признаков.

* * *

Общая схема работы программы узнавания получается следующей: программа наблюдает множество подлежащих в дальнейшем узнаванию объектов. Каждый из них показан в нескольких «ракурсах». Наблюдение-«обучение» заключается в отборе (из некоторого множества) полезных признаков и запоминании характеристик объектов по этим признакам. При узнавании программе показывают предмет в новом (не фигурировавшем при обучении) виде. Программа находит характеристики этого предмета по всем отобранным признакам. После этого происходит сравнение полученных характеристик с наборами характеристик, хранимыми в памяти. Сходство устанавливается по максимальному числу совпадающих характеристик.

Формальное описание программы

А. Объекты узнавания. Рассмотрим числа, представленные в табл. 1. Нетрудно убедиться, что числа, стоящие в каждой строке табл. 1, *а*, удовлетворяют формуле $C=AB(A-B)$, а табл. 1,*б* — формуле $C=(A+B)\cdot(A-B)$. Мы будем рассматривать подобные таблицы как некоторые «фотографии» того закона, по которому построено число C из чисел A и B. Легко видеть, что каждый закон может быть представлен бесконечным количеством таблиц («фотографий») и что, строго говоря, по таблице невозможно однозначно восстановить закон, если о нем нам заранее ничего не известно.

Таблица 1

а			*б*		
A	B	C	A	B	C
5	3	30	5	3	16
2	4	—16	2	4	—12
3	8	—120	3	8	—55
—4	3	84	—4	3	7

Таблица 2

а			*б*		
A	B	C	A	B	C
3	5	—30	3	5	—16
3	2	6	3	2	5
4	8	—128	4	8	—48
—5	2	70	—5	2	21

Для обучения в машину вводится по две таблицы, представляющих каждый закон. Программе сообщается, что эти две таблицы относятся к закону номер один, следующие — к закону номер два и т. д. Сами законы в машину *не вводятся.* Каждая таблица состоит из строчек по три числа. Количество строчек может быть от одной до восьми включительно. При обучении может применяться от 2 до 24 законов.

Для узнавания вводятся другие таблицы (например 2,*а* и 2,*б*), являющиеся представителями тех же или других законов. Число строк в каждой таблице и число законов при узнавании может быть не таким, как при обучении, но заключено в тех же пределах (за исключением того, что при узнавании можно дать только одну таблицу). Задача программы заключается в том, чтобы относительно каждой таблицы, введенной для узнавания, определить: «более всего это похоже на закон номер *Z*».

Б. Построение признаков. Напомним, что признаком называется алгоритм, осуществляющий вырожденное преобразование объекта. В разбираемой программе характеристика таблицы по любому признаку может иметь два значения: 0 и 1.

Признаки строятся следующим способом: берется простая арифметическая функция от A, B и C — f_i (A, B, C), например, A+C или C:B и т. д. Затем полученное число подставляется в один из трех логических операторов. Логический оператор превращает это число в логическое переменное, принимающее два значения: 0 и 1.

Всего в описываемой программе применяется три логических оператора: L_1 — проверка на целочисленность, L_2 — проверка знака, L_3 — сравнение с единицей по модулю.

Таким образом:

$$L_1\mathrm{N}=1, \text{ если N — целое;}$$
$$L_1\mathrm{N}=0, \text{ если N — дробное;}$$
$$L_2\mathrm{N}=1, \text{ если } \mathrm{N}\geqslant 0;$$
$$L_2\mathrm{N}=0, \text{ если } \mathrm{N}<0;$$
$$L_3\mathrm{N}=1, \text{ если } |\mathrm{N}|\geqslant 1;$$
$$L_3\mathrm{N}=0, \text{ если } |\mathrm{N}|<1.$$

Приведем пример. Пусть f_i(A, B, C) есть B:A; тогда

$$L_1 f_i(7;\ 9;\ 3)=0,$$
$$L_2 f_i(7;\ 9;\ 3)=1,$$
$$L_3 f_i(7;\ 9;\ 3)=1.$$

Обозначим $L_j f_i$ через l_{ji} ($l_{ji}=l_{ji}$ (A, B, C), тогда признаком для одной строки таблицы является любая логическая функция $F(l_{j_1 i_1};\ l_{j_2 i_2})$ от двух логических переменных $l_{j_1 i_1}$ и $l_{i_2 i_2}$. Напомним, что имеется всего 16 логических функций двух переменных: a и b. Из них две (тождественные 0 и 1) вообще не интересны. Четыре являются функциями только одного переменного (a; $\bar{a}$; b; $\bar{b}$). Таким образом, остается 10 функций двух переменных, это:

$$\overline{ab};\quad \overline{\bar{a}b};\quad \overline{a\bar{b}};\quad \overline{\bar{a}\,\bar{b}};$$
$$ab;\quad \bar{a}b;\quad a\bar{b};\quad \bar{a}\,\bar{b};$$
$$ab\vee\bar{a}\bar{b};\quad \bar{a}b\vee a\bar{b}.$$

Программа рассматривает все эти функции. Признаком для строки является алгоритм вида $F_m[L_{j_1} f_{i_1}$ (A, B, C); $L_{j_2} f_{i_2}$ (A, B, C)]. Признаком для всей таблицы является логическое произведение признаков для отдельных строк. Таким образом, признак для таблицы имеет вид:

$$F_m[L_{j_1} f_{i_1}(\mathrm{A}_1, \mathrm{B}_1, \mathrm{C}_1);\quad L_{j_2} f_{i_2}(\mathrm{A}_1, \mathrm{B}_1, \mathrm{C}_1)]\wedge$$
$$\wedge F_m[L_{j_1} f_{i_1}(\mathrm{A}_2, \mathrm{B}_2, \mathrm{C}_2);\quad L_{j_2} f_{i_2}(\mathrm{A}_2, \mathrm{B}_2, \mathrm{C}_2)]\wedge\ldots$$
$$\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots$$
$$\ldots\wedge F_m[L_{j_1} f_{i_1}(\mathrm{A}_n, \mathrm{B}_n, C_n);\quad L_{j_2} f_{i_2}(\mathrm{A}_n, \mathrm{B}_n, \mathrm{C}_n)].$$

Здесь индексы при A, B и C соответствуют номеру строки таблицы. Разные признаки отличаются друг от друга значением i_1, i_2, j_1, j_2, m, что соответствует разным выборам арифметических функций, логических операторов и логических функций. Всего программа имеет возможность выбирать из 5436 признаков.

В. Отбор признаков. Само собой разумеется, что имеет смысл запоминать не все признаки, а только *полезные*. Первое требование, предъявляемое к полезному признаку: он должен давать сечение на множестве таблиц, по которым ведется обучение. Это значит, что признак не должен давать *всем* таблицам одну и ту же характеристику (безразлично — 0 или 1), так как характеристика по такому признаку не несет

никакой информации о номере закона таблицы. Ясно, что наибольшую информацию дает признак, который половину таблиц характеризует нулем, а половину — единицей. Кроме того, мало полезны признаки, дающие сечение, совпадающее с сечением, уже произведенным одним из предыдущих признаков.

Второе требование к полезному признаку: он должен давать двум таблицам, представляющим один и тот же закон, одинаковую характеристику.

В соответствии с этими замечаниями подпрограмма отбора признаков работает так: таблицы, на которых идет обучение, разбиты на две группы. Первая — рабочая, вторая — контрольная. Каждая группа содержит по одной таблице, представляющей каждый закон. Программа берет признак, подлежащий отбору, и присваивает по нему характеристики всем таблицам рабочей группы. Затем проверяется, сколько таблиц имеют характеристику 1. Пусть используется S таблиц. Если число единиц не равно S/2 при четном S или S/2 ± 1/2 при нечетном S, то признак бракуется. Если он проходит эту проверку, то проверяется, не дает ли он уже встречавшееся ранее сечение. В случае, если признак выдержан и это испытание, программа присваивает характеристики по этому признаку таблицам контрольной группы. Затем проверяется совпадение характеристик таблиц рабочей и контрольной групп. В случае совпадения их для всех законов признак запоминается« в случае несовпадения — бракуется. После исчерпания запаса всех 5346 возможных признаков программа уменьшает строгость первого этапа отбора и снова просматривает первоначально отвергнутые признаки. Теперь пропускаются признаки, дающие S/2±1 единиц при четном S и S/2±1½ единиц при нечетном S *. Затем снова уменьшается строгость отбора и т. д. Процесс заканчивается, когда или отобраны 30 полезных признаков или дальнейшее уменьшение строгости отбора привело бы к запоминанию бесполезных (не дающих сечения на множестве S таблиц) признаков.

Легко доказать, что при любом способе разбиения двойного набора таблиц на рабочую и контрольную группы будут отобраны те же самые признаки.

В результате обучения в памяти остаются программы полезных признаков (30 или меньше) и характеристики всех S пар таблиц по всем признакам (внутри пары таблицы всегда имеют одинаковые характеристики).

Г. Узнавание. Подпрограмма узнавания присваивает неизвестным таблицам характеристики по всем отобранным признакам. Затем полученные характеристики сравниваются с характеристиками таблиц, использованных при обучении. Сравнение заключается в подсчете числа признаков, по которым характеристики не совпадают. В результате получаются, например, такие сведения: характеристики этой неизвестной таблицы расходятся с характеристиками таблиц первого типа по 17 признакам, второго типа — по 3 признакам, третьего типа — по 14 признакам и т. д. Далее программа выбирает три наименьших числа и печатает номера относящихся к ним типов в порядке возрастания количества расходящихся характеристик.

* * *

Помимо окончательного результата — суждения о том, на какие знакомые таблицы больше всего похожи таблицы, данные для узнавания, программа печатает следующие промежуточные сведения: запоминаемые

* В программе предусмотрена возможность начинать с менее строгого, чем «S/2» отбора.

полезные признаки; сечения, производимые этими признаками, на множестве таблиц, данных для обучения; сигналы о переходе к менее строгому отбору признаков; сигнал об окончании обучения; характеристики таблиц, применявшихся при обучении, по всем отобранным признакам; характеристики таблиц, данных для узнавания, по всем признакам.

Реальная работа программы

Все опыты прозводились на машине М-2 ИНЭУМ АН СССР, имеющей среднюю скорость работы — 2000 трехадресных операций в секунду.

А. Узнавание арифметических законов, участвовавших в процессе отбора признаков. В опытах этой серии в машину для обучения вводились таблицы, содержащие целые A и B (например, табл. 1). Все строчки каждой таблицы были построены по одному и тому же арифметическому закону. Для узнавания вводились таблицы, построенные по тем же законам, что и использованные при обучении, но содержащие другие числа; A и B — тоже целые.

Проводились опыты с обучением по 6, 8, 9 и 24 законам.

Обучение по шести законам при трех строчках в каждой таблице занимало ~6 мин., по 24 законам при восьми строчках в каждой таблице ~35 мин.

Узнавание каждого закона (включая время печати ответа) занимало 7—10 сек.

Применялись два набора по 24 закона, приведенные в табл. 3. При обучении по 6, 8 и 9 законам использовались первые законы из набора 1.

Для краткости мы не останавливаемся на разных вариантах опытов с обучением по 6, 8 и 9 законам (отбирались признаки с различными критериями полезности), так как все они дали полное узнавание таблиц. Это значит, что номер соответствующего закона всегда печатался на первом месте.

При узнавании 24 законов не все законы были узнаны правильно. Некоторые номера попали на второе или даже более далекое место. Для оценки и сравнения качества работы программы в разных случаях полезно определять среднее место M_c, на которое приходится номер соответствующего закона. Очевидно, что программа, «гадающая наудачу», дала бы при выборе из S законов (при большом числе проб) среднее место $M_г = \frac{S+1}{2}$. Такой же результат покажет программа, которой обучение «не пошло в прок». Идеально узнающая программа даст среднее место, равное $M_0 = 1$. Соотношение «гадания» и «знания» в работе программы мы будем характеризовать величиной:

$$\eta = \frac{M_c - M_0}{M_г - M_0} = \frac{M_c - 1}{M_г - 1}.$$

При идеальной работе $\eta = 0$, при случайном гадании $\eta = 1$.

При узнавании первого набора из 24 законов (табл. 3) $\eta = 0{,}025$; при работе с набором 2 (табл. 3) $\eta = 0{,}036$. Таким образом, при узнавании 24 законов элемент случайного гадания составлял около 3% *. Такой результат кажется нам вполне удовлетворительным.

Б. Узнавание арифметических законов, не участвовавших в процессе отбора признаков. Обучение в этой серии опытов производилось так

* Можно также считать, что величина η характеризует степень незнания, остающуюся после работы программы.

Таблица 3

	Набор 1				Набор 2		
1	$(A-B)AB$	II	I	1	$A-A=O$	III	IV
2	$\frac{A}{B}+A+B$	VI	I	2	$\frac{A}{B}-\frac{1}{A}$	III	IV
3	$\frac{B-A}{A}-A$	IV	II	3	AB^2-A	I	VI
4	$(A+B)(A-B)$	II	I	4	B^2-B^4	II	I
5	$\frac{A-B}{A+B}$	III	IV	5	$\frac{A+B}{A-B}$	III	IV
6	$\frac{AB}{A+B}$	III	V	6	$(B-A)^2-B$	V	I
7	$(A+B)AB$	I	VI	7	$(B-A)^2+B$	VI	I
8	$\frac{A+B}{A}+B$	III	IV	8	$\frac{B}{(A-B)^2}$	III	IV
9	$\frac{(A-B)\cdot A}{B}$	II	III	9	$\left(\frac{B}{A-B}\right)^2$	III	IV
10	$\frac{A}{B}\left(A-\frac{A}{B}\right)$	VI	I	10	$\left(\frac{A}{B}\right)^2-\frac{A}{B}$	III	IV
11	$\frac{A+B}{AB}$	III	IV	11	$\left(\frac{A}{B}\right)^2+\frac{A}{B}$	III	VI
12	$\frac{A-B}{AB}$	IV	III	12	B^3-B^2	I	VI
13	$\frac{B-2A}{B}$	IV	II	13	$\frac{A}{B^4}$	III	IV
14	$\frac{2A^2}{B}$	I	III	14	$\frac{B}{B^4}$	III	IV
15	$AB-B+A$	I	VI	15	A^3B^2	V	I
16	$(A-AB)\cdot B$	II	I	16	$4B$	I	VI
17	$(A-2B)\cdot A$	II	I	17	$6B$	I	VI
18	$\frac{AB}{B^2+A}$	III	IV	18	$4A^2B^2$	V	VI
19	$\frac{A+B}{B^2}$	III	IV	19	$5B$	I	VI
20	$\frac{B}{A}\left(A+\frac{B}{A}\right)$	III	IV	20	$\frac{(B-A)^2}{A}$	III	I
21	$\frac{B}{B+2A}$	III	V	21	$\frac{B}{A}-\frac{A}{B}$	III	IV
22	$(A+B)B$	I	VI	22	$\frac{A}{B-B^2}$	IV	III
23	$4A^2B^2$	V	VI	23	$\frac{A}{3B}$	IV	III
24	$\frac{A^4}{B}$	V	I	24	$\frac{A}{4B}$	IV	III

же, как и в предыдущей. Затем в машину вводились таблицы, построенные *не по тем* законам, по которым проводилось обучение. Программа присваивала этим новым законам характеристики по признакам, отобранным во время обучения, и характеристики запоминались. После этого для узнавания вводились *другие* таблицы, построенные по тем же новым законам. Им снова присваивались характеристики и они сравнивались с характеристиками, находящимися в памяти. Таким образом, отличие серии опытов «*Б*» от предыдущей заключалось в том, что в серии «*А*» узнавание производилось при помощи признаков, *специально подобранных к данной группе арифметических законов,* а в серии «*Б*» использовались признаки, подобранные *не к той группе законов*, которая узнавалась. Таким образом, в серии «*Б*» проверялась *степень универсальности* отобранных признаков.

После обучения на втором наборе из 24 арифметических законов было проведено узнавание 24 законов 1 набора. Было получено $\eta=0{,}029$. Сравнение с результатами серии «*А*» и ($\eta=0{,}025$ и $\eta=0{,}036$) показывает, что узнавание при помощи «чужих» признаков происходило в этом опыте не хуже, чем при помощи «своих».

Был прозведен опыт с обучением по шести арифметическим законам (первые 6 из набора 1). Затем по отобранным признакам проводилось узнавание 24 законов. Узнавание 1 набора дало $\eta=0{,}007$, 2 набора — $\eta=0{,}025$.

Любопытно отметить, что в этом опыте узнавание по «чужим» признакам дало результат, даже несколько лучший, чем при пользовании признаками, специально подобранными для узнаваемых законов. Причина этого вовсе не в том, что признаки, отобранные при обучении на 6 законах, работали более надежно, чем отобранные при обучении на 24 законах. Наоборот, просмотр результатов показывает, что средняя вероятность ошибки в каждой характеристике по признакам, полученным при обучении по 6 законам, равна 4,2%, а для признаков, полученных при обучении по 24 законам, только 1,4%. Чем же объясняется, что более надежно работающие признаки дают иногда худшее узнавание? Дело в том, что качество узнавания определяется не только надежностью признаков, но и тем, *по скольким признакам* отличаются друг от друга законы. При применяющемся в данной программе алгоритме отбора признаков при обучении по 24 законам, характеристики некоторых законов по всем признакам могут получиться очень похожими (а иногда даже одинаковыми). Ясно, что при этих условиях ошибки в малом числе признаков могут привести к ошибочному узнаванию. Признаки же, отобранные при обучении по 6 законам, хотя и работали менее надежно, но зато давали для разных законов более сильно различающиеся наборы характеристик. В данном случае это привело к некоторому улучшению узнавания.

Таблица 4

I	II	III
(A+B) AB AB+A+B	(B—A) AB A—AB—B	$\frac{A+B}{AB}$ $\frac{A}{(A+B)\,B}$
IV	V	VI
$\frac{A-B}{AB}$ $\frac{B-2A}{A}$	A^2B^3 A^3B^2	2A+3B 3A+2B

Во всяком случае, признаки, отобранные при обучении по 6 законам, оказались пригодными для вполне удовлетворительного узнавания, по крайней мере, 48 законов (а в действительности, вероятно, и гораздо большего числа).

В. «Классификация» законов. В этом опыте в машину для обучения были введены *6 групп* законов (табл. 4). Каждая группа состояла из двух законов, похожих один на другой в отношении применяющихся в них арифметических действий. Таблицы, построенные по законам каждой группы, занимали места, предназначенные для рабочей и контрольной таблиц одного закона в предыдущих сериях опытов. Поэтому программа при обучении отбирала признаки, которые давали для законов одной группы одинаковые характеристики. Затем вводили 24 закона и программа «решала», на какую группу данный закон более всего похож. Два первых места для каждого закона приведены в табл. 3 (римские цифры). Читатель может сопоставить «мнение» программы с тем, как бы он сам разбил приведенные арифметические законы по группам табл. 4. В этом опыте программа произвела классификацию по принципу, заданному не строгим алгоритмом, а примерами.

Г. Логические законы. В этом опыте обучение производилось на таблицах, в которых число С являлось не арифметической, а логической функцией А и В. Например, если А и В положительны, то С=1, а если

нет, то С=0; или: если $A\geqslant B$, то С=1, а если нет, то С=0 и т. д. Обучение производилось на четырех законах, затем для узнавания были введены другие четыре таблицы, построенные по тем же законам. Все четыре закона были узнаны правильно.

Д. Смешанные законы. Смешанными законами мы называем построение разных строчек таблицы по разным арифметическим законам, в зависимости от дополнительных логических условий. Например: С=А+В, если $A\geqslant 0$ и $B\geqslant 0$; С=А—В, если $A\geqslant 0$ и $B<0$; С=А:В, если $A<0$ и $B\geqslant 0$; С=АВ, если $A<0$ и $B<0$.

Обучение проводилось на шести законах подобного типа. При узнавании были введены другие таблицы, построенные по тем же законам. Все шесть были правильно узнаны. Тот же результат дал опыт с 10 смешанными законами.

Е. Узнавание законов типа, не участвовавшего в процессе отбора признаков.

В этой серии опытов обучение производилось на законах одного типа (например, арифметических), а узнавать давали законы другого типа (например, логические). При этом программа могла получить задание двух типов: 1) отличать законы нового типа друг от друга; 2) отличать их, кроме того, и от законов, на которых производилось обучение.

После обучения на 9 арифметических законах узнавание 4 логических дало следующие результаты: узнавание среди всех законов (и арифметических и логических) $\eta=0{,}25$; узнавание среди только логических законов — $\eta=1{,}00$. Это значит, что хотя программа совершенно не различала между собой логические законы, она могла отличать их от арифметических.

Для узнавания 10 смешанных законов признаки, отобранные при обучении на 9 арифметических, дали: узнавание среди всех законов $\eta=0{,}10$; узнавание среди только смешанных — $\eta=0{,}20$.

После обучения на 10 смешанных законах узнавание 4 логических дало: узнавание среди всех (смешанных и логических) — $\eta=0{,}19$; узнавание среди только логических — $\eta=0{,}83$. Узнавание 9 арифметических законов дало: узнавание среди всех законов (смешанных и арифметических) — $\eta=0{,}025$; узнавание среди только арифметических — $\eta=0{,}055$.

Таким образом, признаки, отобранные при обучении на арифметических законах, дают возможность удовлетворительно узнавать смешанные, и наоборот. Эти признаки не позволяют различить между собой применявшиеся в наших опытах логические законы, все же отличая их (весьма надежно) от тех законов, на которых производилось обучение (арифметических или логических).

Обсуждение результатов

А. Нужно ли программе самой учиться? Нельзя ли заложить в программу узнавания хорошо подобранные человеком готовые признаки? Тогда программе не нужно будет тратить время на обучение. Она сразу, так сказать от рождения, будет готова узнавать. Этот вопрос напрашивается после сравнения времени, уходящего на обучение (минуты), и на узнавание (секунды). Ответ на вопрос не так прост; он зависит от того, *знает ли заранее человек, какие объекты придется узнавать программе.* Если это заранее точно известно (например, сказано, что программа должна различать такие-то десять арифметических законов), то программист может подобрать признаки, заложить их в программу и обучение станет излишним. Такая программа будет очень хорошо (вероятно, лучше, чем обучающаяся) и быстро работать, но лишь до тех пор, пока не изменятся объекты узнавания. Как мы уже знаем, для других законов оказываются полезными другие признаки. Поэтому, если человек заранее не знает, с какими законами придется иметь дело про-

грамме («в каком мире она будет жить») или если разнообразие законов достаточно велико, то выгоднее написать обучающуюся программу, предоставив ей самой приспосабливаться к «миру», в который она попадет.

Любопытно отметить, что оба эти пути нашли отражение в эволюции животных. Это — организация программ, управляющих поведением насекомых и позвоночных. Насекомое почти все умеет от рождения. Удельный вес обучения весьма мал. Многие же позвоночные, наоборот, рождаются, почти ничего не умея. Значительную часть приспособленности своего поведения к среде они получают не по наследству, а путем обучения. Обучение занимает время, зато животное может научиться тому, чего не умели его родители.

Итак, обучающиеся программы нужны тогда, когда заранее неизвестно, в каком «мире» программе придется «жить», или мир этот широк, или программист сам не знает, какие признаки окажутся полезными. Во всех этих случаях недостаток знаний или умения программиста может быть восполнен обучением программы.

Б. Что такое «признак»? До сих пор ограничивались чисто формальным подходом к признакам. Рассмотрим теперь, каким элементам человеческого мышления аналогичны возникающие в программе «признаки».

Признак осуществляет вырожденное преобразование таблицы. Многим таблицам соответствует одинаковая характеристика по данному признаку, и даже совокупность характеристик по всем признакам имеет разнообразие, во много раз меньшее, чем разнообразие таблиц. Таким образом, характеристика таблицы по всем признакам несет значительно меньше информации, чем сама таблица. Зачем же нужно такое перекодирование, явно сопряженное с потерей информации? Ответ заключается в рассмотрении того, *какая* информация теряется. Вспомним, что *разные таблицы,* построенные *по одному и тому же закону*, имеют одинаковый (или почти одинаковый) набор характеристик по признакам. Значит, при рассматриваемом перекодировании теряется информация об индивидуальных особенностях таблицы, построенной по данному закону. Если закон один, а числа в таблицах разные, то по набору характеристик мы не можем восстановить конкретные числа. Информация о них утеряна. Зато сохранилась информация о законе. Действительно, *разным* законам соответствуют *разные* наборы характеристик. Поэтому сохранилась именно интересовавшая нас информация. Трудность узнавания новой таблицы заключается в том, что таблица имеет новые (не встречавшиеся раньше) числа. Таблица несет информацию не только о законе, по которому она построена, но и о конкретных числах, входящих в нее. Эта вторая (не нужная нам) часть информации много больше первой. От нее необходимо избавиться (окончательный результат должен быть независимым от конкретных чисел), что и достигается присвоением характеристик по признакам.

Этот процесс аналогичен созданию абстрактных понятий. Когда мы говорим про животное: «кошка», то эта характеристика (в действительности — набор характеристик по многим признакам) не несет информации о цвете, кличке, возрасте, характере и других *индивидуальных* особенностях данной кошки, зато обозначает особенности, отделяющие *всех* кошек от собак, слонов и других животных. По каким признакам нужно брать характеристики для получения полезной абстракции — зависит от того, от каких животных надо отличать кошку. Если в это множество входят корова и олень, то полезной будет характеристика по признаку «наличие рогов», а если рогатых животных нет в рассматриваемом множестве, то «наличие рогов» будет бесполезным признаком (не дает сечения на множестве). «Убирающиеся когти» будут отделять кошку от собаки, но окажутся бесполезными для отделения кошки от

тигра. Таким образом, абстрактные понятия человека — набор характеристик по полезным признакам. Поэтому присвоение данной таблице набора характеристик по признакам формально и, по существу, неотличимо от создания в машине абстрактного представления о *законе,* по которому построена таблица (именно о законе, а не о данной конкретной таблице).

Можно возразить, что абстракции, выработанные программой, работают только на весьма ограниченном множестве законов (тех, на которых происходило обучение), а для новых законов могут оказаться бесполезными. Это возражение справедливо, но в равной мере оно относится и к абстракциям, возникающим в человеческой голове. Трех-четырехлетний ребенок, впервые попавший в зоопарк, увидев тигра, говорит: «кис-кис». Его понятие «кошка» еще не включает признаков, отличающих ее от тигра (и не будет включать до тех пор, пока в процессе обучения он по той или иной причине не встретится с потребностью различать их). Да и взрослый европеец, впервые столкнувшийся с необходимостью различать в лицо несколько японцев, делает это с большим трудом. Признаки, весьма полезные для различения лиц европейского склада, не дают возможности хорошо различать между собой японцев, хотя и очень надежно отличают их от всех лиц европейцев. В этом отношении человек ведет себя так же, как программа, которая обучалась на арифметических законах, а должна узнавать логические правила.

Итак, наборы характеристик по признакам — эквиваленты абстрактных понятий о законах построения таблиц.

Напомним, что признаки, отобранные при обучении на нескольких таблицах, оказались полезными для узнавания гораздо большего числа таблиц похожего типа. Это позволяет надеяться, что обучающейся программе можно будет показывать лишь часть (достаточно разнообразную) того мира, в котором ей придется действовать. Таким способом удастся сильно сократить сроки обучения и необходимый объем памяти. Сам же по себе этот факт подтверждает наш взгляд на набор признаков как на абстракцию, применимую к более широкому классу законов, чем тот, на котором происходило обучение.

В. Что делает программа и что сделал программист? При оценке работы различных «обучающихся», «самоорганизующихся» и т. п. программ многие впадают в крайности двух противоположных типов. Одни говорят: «программу написал человек, он тем самым задал правила ее работы во всех случаях жизни. Конечно, он не знает заранее, что́ она будет делать, но лишь потому, что не обладает достаточным временем для выяснения этого. Человек всему научил программу, а значит, сам умеет то, что умеет она, только машина работает быстрее». Другая крайность заключается в том, что обращают внимание только на работу уже организованной программы и говорят: «программа выбирает, программа принимает решение, программа приходит к выводу...», забывая, что в этих действиях программа опирается и на запас знаний *программиста.*

Мы считаем, что в действительности положение здесь такое же, как в случае с профессором, давшим трудную задачу студенту. Студент использует идею, а иногда и метод, сообщенные ему учителем. Но если сам профессор не занимается работой над задачей, то через некоторое время студент (пользуясь этим методом) накопит столько знаний, что сможет решить, например, разновидность первой задачи, совершенно непосильную профессору (если профессор сам не пройдет сходный путь обучения или не воспользуется результатами студента).

Программа способна превзойти создавшего ее человека в том же смысле, в котором человек может превзойти своего учителя. При этом не следует забывать, что для продуктивной работы необходим достаточно высокий начальный уровень организации. В человека или в про-

грамму должно быть заложено (учителем, программистом, предшественником-человеком или машиной) достаточно сведений для сокращения области поисков, иначе работа потребует так много времени, что станет практически невыполнимой.

В чем же превосходит своего учителя описываемая программа? В нее заложен принцип узнавания, множество, из которого можно отбирать признаки, критерии отбора признаков. До всего этого она *не сама* додумалась. Сама же программа отбирает признаки, полезные для распознавания заданного множества законов. К концу обучения она вырабатывает в себе абстрактные представления о законах, достаточные для правильного узнавания *других* таблиц, построенных по тем же законам. В этот момент программа уже умеет делать то, чего не умеет программист. Заметим, что в период обучения программа вела экспериментальную работу. Поэтому часть сведений, запасенных в ней к концу обучения, поступили в нее не от программиста, а непосредственно из природы (хотя заложил в нее способность к экспериментированию программист).

Самостоятельная экспериментальная работа программы может восполнить некоторые просчеты программиста. Так например, нет никакой гарантии, что для описываемой программы выбраны наилучшие логические операторы или наилучший алгоритм отбора признаков. Но если алгоритм отбора изменить или оставить только два логических оператора вместо трех и т. п. (такие опыты делались), то программа приспосабливается к новым условиям, отбирает *другие* полезные признаки и все равно успешно добивается конечной цели — правильно узнает таблицы.

Г. Связь с физиологией. Мы далеки от мысли, что настоящая работа доказала сходство операций, производимых программой и мозгом при решении задачи узнавания. Имеет ли эта работа какое-нибудь отношение к физиологии? В настоящее время не ново и даже стало «модным» агитировать за применение моделирования различных функций живого организма. Поэтому мы рассмотрим только специфику моделирования достаточно сложных функций мозга.

Можно ли было бы изучать физиологию скелетных мышц, не зная их основной функции — совершать механическую работу; нервного волокна — передавать информацию; сердца — гнать кровь? Конечно, такие исследования были бы очень мало продуктивными. По отношению к сердцу, нервным волокнам и скелетной мускулатуре подобный вопрос является в настоящее время абстрактным. В то же время о работе мозга мы знаем еще очень мало. Теперь уже многим ясно, что под влиянием внешних воздействий в мозгу не просто замыкаются новые пути, а возникают новые *программы анализа* воздействий и *программы реакций*. Какие задачи решают эти программы? Какими алгоритмами осуществляются? Какие логические операции производит мозг для вырабатывания этих новых программ? Обо всем этом мы почти ничего не знаем. Таким образом, мы мало знаем не только о механизме работы мозга (как он действует), но и об основных функциях (что он делает!). Какое отношение имеет ко всему этому моделирование процесса узнавания? Вряд ли будет преувеличением сказать, что узнавание объектов является одной из основных функций целых отделов мозга, например зрительных областей. Какие операции нужно произвести над сигналами, посылаемыми палочками и колбочками сетчатки для того, чтобы животное имело возможность реагировать *на предмет*, то есть *одинаково* реагировать на различные варианты сигналов, вызываемые этим предметом? Моделируя узнавание пытаются нащупать подход к этому вопросу, определить алгоритмы, которые должны осуществляться в системе, способной узнавать. Появление даже начальных знаний в этой области сделает исследование физиологии крупных нервных структур значительно более целеустремленным.

* * *

1. Осуществлена программа, обучающаяся узнаванию. Объектами узнавания служат таблицы чисел, построенные по разным законам.

2. В процессе обучения программа отбирает «признаки» — алгоритмы, дающие вырожденные отображения таблиц. Отбираются «полезные» признаки — признаки, дающие одинаковые характеристики таблицам, построенным по одному и тому же закону, и разные характеристики таблицам, построенным по разным законам.

3. Узнавание заключается в присвоении характеристик по признакам неизвестной таблице, и в сравнении полученного набора характеристик с наборами, хранимыми в памяти.

4. Описано узнавание программой разных типов законов при разных вариантах обучения.

5. В обсуждении рассматривается целесообразность обучения, аналогия между «признаками» и абстрактными понятиями, степень самостоятельности обученной программы и возможная польза настоящей работы для изучения механизма работы мозга.

* * *

Автор приносит искреннюю благодарность А. С. Кронроду и А. Л. Лунцу, обучившим его программированию, и А. Л. Брудно за предоставление возможности работать на машине.

Поступила в редакцию
12.VII.1960

MODELLING OF RECOGNITION PROCESS IN COUNTING MACHINE

M. M. BONGARD

1. A program is worked out that learns itself to recognize. The recognition objects are the tables of numbers constructed according to different laws.

2. In the process of learning the program selects the «indications» — algorythms that give the singular transformations of tables. The «useful» indications are selected, that give similar characteristics to the tables that are constructed according to one and the same law, and different characteristics to the tables, constructed by different laws.

3. The recognition consists in giving the characteristics by the indications to the unknown table and in the comparison of the received set of characteristics with the sets that are in the memory.

4. The recognition by the program of different types of laws on different variants of learning are described.

5. In the discussion we consider the usefulness of learning, analogy between the «indications» and abstract conceptions, the degree of independence of the trained program and possible rôle of this investigation for the brain physiology.

Received: 12.VII.1960