Применение методов искусственного интеллекта для планирования персональной стратегии обучения

Бекташев Р.А.[1]

Аннотация

В статье поставлена задача планирования персональной стратегии обучения и приведены результаты применения различных подходов к ее решению на примере курса обучения программированию.

Ключевые слова: компьютерная обучающая система, предсказание элементов последовательности, нейронная сеть, рекуррентная нейронная сеть, компактное дерево предсказаний

1. Введение

Анализ данных в образовании (Educational Data Mining, EDM) становится эффективным инструментом в руках исследователей, который позволяет совершенствовать процесс обучения как со стороны студента, так и со стороны преподавателя. Решая широкий спектр задач, современные алгоритмы анализа данных позволят сгладить возможные различия в подготовке студентов, а также повысить ее средний уровень, что, несомненно, полезно для университетов. Новая университетская образовательная модель [1] ставит задачу построения такой информационно-образовательной среды, которая сможет служить платформой для применения предложенных методов анализа данных, способствуя более эффективному использованию ресурсов университета, студента и преподавателя. Подход с использованием вычислительных технологий успешно

[1] *Бекташев Руслан Анварович* — аспирант каф. математической теории интеллектуальных систем мех.-мат. ф-та МГУ, e-mail: ruslanbektashev@gmail.com.

Bektashev Ruslan Anvarovich — graduate student, Lomonosov Moscow State University, Faculty of Mechanics and Mathematics, Chair of Mathematical Theory of Intellectual Systems.

применяется во многих бизнес-моделях (рекомендательных системах, голосовых помощниках и т. п.), поэтому, даже учитывая специфичность образовательной модели, и здесь можно рассчитывать на аналогичные результаты.

В университетах программы курсов обычно разделены на темы, излагаемые во время лекций. Каждая тема закрепляется решением набора задач на семинарах. Для проверки решений этих задач преподаватели могут использовать автоматическую систему в которую студенты будут отправлять свои решения. Система может вести учет различных данных в процессе обучения (например, время решения задачи, количество неудачных попыток, качество решения и т. д.). Собранные данные можно использовать для выявления применяемых шаблонов, типичных ошибок и соответствующей корректировки плана обучения. Однако, не вся собранная информация сразу готова к обработке — ввиду того, что решением задачи является текст на некотором языке, оценка качества решения представляет собой отдельную непростую задачу. Такие свойства решения, как: уникальность, краткость, соответствие ограничениям и др. обычно выявляются преподавателем вручную, но этот процесс может быть частично автоматизирован. Для этого необходимо формальное определение понятия *оценка свойства решения*, после чего станет возможным построение соответствующих алгоритмов.

2. Постановка задачи

2.1. Основные понятия

Множество всех задач некоторого раздела обозначим P.

Определение 1. *Конечная последовательность слов в некотором конечном алфавите символов является решением задачи.*

Множество всех решений некоторой задачи $p \in P$ обозначим A_p.

Определение 2. *Множество* $S = \{s \mid 0 \leqslant s \leqslant 100,\ s \in \mathbb{R}\}$ *будем называть шкалой оценивания.*

Введем полный порядок на множестве решений относительно некоторого свойства f:

Определение 3. $\forall\, a, b \in A_p$ *верно одно из утверждений:*

- $a \leqslant_f b$, *если* b *в большей степени обладает свойством* f, *чем* a

- $a \succcurlyeq_f b$, *иначе*

Определение 4. *Монотонное отображение* $F\colon A_p \to S$ *является оценкой свойства решения.*

Выбранная шкала оценивания может обеспечить инъективность оценки свойства решения, хотя это, вообще говоря, необязательно. Для некоторых свойств решения имеет смысл ввести упрощенную шкалу оценивания. К примеру, свойство решения быть верным (т. е. решать поставленную задачу) может принимать два значения: «да» (1) или «нет» (0).

Определение 5. *Пусть* $M \subset \mathbb{Z}_+$. *Монотонное отображение* $I\colon S \to M$ *назовем интерпретацией оценки свойства решения.*

Для вычисления значения F на практике необходимо построить соответствующий алгоритм, который принимает на вход решение, а на выходе выдает оценку свойства решения. Однако, определенная ранее шкала оценивания непрерывна, поэтому следует дискретизировать образ оценки свойства решения:

Определение 6. *Отображение* $\bar{F} = F \circ I$ *назовем дискретной оценкой свойства решения.*

2.2. Построение дискретного пространства признаков

Любое отображение F кодирует некоторое свойство f решения задачи $p \in P$. Большинство таких свойств бессмысленны или вырождены. Однако, некоторыми из этих свойств руководствуются преподаватели при оценивании решения $a \in A_p$ вручную. К примеру: решение проходит 7 из 10 тестов, подготовленных преподавателем для данной задачи; имеется 80-процентная схожесть с решениями данной задачи у другого студента.

Определение 7. *Множество* $\mathbb{F} = \{f_1, f_2, \ldots, f_n\}$ *свойств решения порождает пространство признаков*

$$T_a = S_1 \times S_2 \times \cdots \times S_n \simeq S^n \subset \mathbb{R}^n$$

решения a *задачи* p. *Компонента* t_i *точки* $t = (t_1, t_2, \ldots, t_n)$ *характеризует степень обладания свойством* f_i *решения* a.

Пространство признаков можно дискретизировать, если вместо F использовать $\bar{F}$ с интерпретацией $\bar{S} = \{0, 1, \ldots, 100\}$.

Определение 8. *Пространство признаков решения a, порожденное свойствами $\bar{\mathbb{F}}$, назовем дискретным пространством признаков и обозначим $\bar{T}_a$.*

Для построения необходимого нам дискретного пространства признаков следует определить множества осмысленных свойств решения каждой задачи из некоторого раздела, которые можно эффективно оценить.

Определение 9. *Объединение свойств решения всех задач из раздела порождает универсальное пространство признаков для раздела, обозначим его T_P .*

Основную сложность здесь представляет автоматическая оценка некоторого свойства.

Задача 1. *Построить алгоритм $V_{\bar{F}}$ оценивающий заданное свойство f со сложностью не более $O(l)$, где l - количество символов в решении.*

При вычислении оценки некоторых свойств решения, описываемых регулярными выражениями, можно построить конечный автомат, распознающий слова допускаемые этими выражениями. Примером такого свойства может служить «наличие или отсутствие некоторого множества слов». Для свойства «компилируется» алгоритмом является процесс компиляции [9], который не может быть реализован классическим конечным автоматом. Существуют также свойства, задача оценки которых алгоритмически неразрешима, например - «корректно решает поставленную задачу» [10]. Для решений, имеющих вид текстов программ на некотором языке программирования, подобные свойства рассматриваются в направлении верификации программ [11]. В нашем случае формальное доказательство корректности программы не обязательно, достаточно лишь доказательства корректности на некотором нетривиальном наборе входных данных - ограниченного тестирования черного ящика.

2.3. Планирование персональной стратегии обучения

Элемент $t \in \bar{T}_a$, отражающий совокупные свойства решения, обозначает степень освоения навыков, тренируемых или тестируемых задачами P. Последовательность

$$T_k = (t_0, t_1, \ldots, t_k),\ t_i \in \bar{T}_P,\ i = \overline{0,k}$$

отражает процесс обучения в виде результатов решения k последовательных задач из раздела.

Степень освоения студентом навыков или тем в данный момент времени можно также представить в виде точки в конечномерном дискретном пространстве $\bar{S}_1 \times \bar{S}_2 \times \cdots \times \bar{S}_m \simeq \bar{S}^m$, где каждой оси соответствует один навык или тема. Последовательность

$$L_q = (l_0, l_1, \ldots, l_q),\ l_i \in \bar{S}^m,\ i = \overline{0, q}$$

такая, что

$$l_i \preccurlyeq l_{i+1},\ i = \overline{0, q-1}$$

является кривой обучения [4] и описывает процесс обучения студента во времени с 1-го до q-го тактов измерения. Для данного типа кривых разработан алгоритм поиска ближайшего соседа [4], который можно использовать для планирования персональной стратегии обучения. Однако, условие монотонности в представлении данных имеет существенный недостаток — в таком случае не учитывается склонность студента к забыванию части информации при длительном обучении без закрепления.

Алгоритм поиска ближайшего соседа заключается в разбиении пространства $\bar{S}^m$ на такие сектора, что при попадании объекта-запроса в один из секторов, поиск сводится к перебору объектов внутри данного сектора. Фактически задается такое отображение точек

$$h\colon \bar{S}^m \to \{1, \ldots, k\},$$

что в соответствии с характеристиками точек суммарная характеристика сектора одинакова для всех секторов. Каждой точке s пространства $\bar{S}^m$ ставится в соответствие тройка (v, r, k), где: $v = \|s\|_\Sigma$, $r = \|s - c\|_\Sigma$, $k = \arg\min \{\|s - (v, \ldots, 0)\|_\Sigma, \ldots, \|s - (0, \ldots, v)\|_\Sigma\}$.

Номер сектора, которому принадлежит точка $s \in \bar{S}^m$ по заданной тройке вычисляется по формуле

$$h(s) := m(100m + 1) \cdot l(v, r) + m \cdot g(r) + k,$$

где

$$l(v, r) = \left\lfloor \frac{v}{g(r) + 1} \right\rfloor,\ g(r) = \begin{cases} r, & r \leqslant 1 \\ r - 1, & r > 1 \end{cases}$$

В случае, если сектор с номером $h(s)$ не содержит объектов, находятся соседние сектора, в которых поиск продолжается. Соседние сектора определяются как объединение трех множеств:

$$\{h(s_1) \mid s_1 \to (l(v, r), g(r), i),\ 0 \leqslant i \leqslant m,\ i \neq k\},$$

$$\{h(s_2) \mid s_2 \to \begin{cases} (l(v,r)-1, g(r), k), & l(v,r) > 0 \\ (l(v,r)+1, g(r), k), & (l(v,r)+1)(g(r)+1) \leqslant 100m \end{cases}\},$$

$$\{h(s_3) \mid s_3 \to \begin{cases} (l(v,r), g(r)-1, k), & g(r) > 0 \\ (l(v,r), g(r)+1, k), & r \leqslant 100m \end{cases}\}.$$

Доказательство корректности алгоритма и оценка его сложности приведены в [4].

Заметим, что переход от последовательности типа T к последовательности типа L, без условия монотонности элементов последней, прост. В самом деле, достаточно покомпонентно перебирать элементы T и заполнять компоненты элементов L, оставляя наибольшую из компонент предыдущего элемента L и текущего элемента T.

Сформулируем задачу планирования стратегии обучения используя кривую обучения. Каждый студент находится в некотором начальном состоянии l_0, которому соответствует кривая $L_0 = (l_0)$. Цель обучения — через q тактов измерения оказаться в состоянии $l_{max} = (100, \ldots, 100)$, то есть освоить все навыки и темы некоторого раздела. Кривая, соответствующая успешному обучению:

$$L_q = \underbrace{(l_0, \ldots, l_{max})}_{q+1}$$

Планирование стратегии обучения заключается в предсказании d будущих состояний студента для некоторой начальной кривой L_i, чтобы назначать ему задачи, тренирующие соответствующие навыки, и закреплять освоенные темы.

Зафиксируем целевую функцию для оптимизации - по результатам решения планируемой задачи студент должен наиболее далеко продвинуться к цели l_{max}. Введем для задачи следующие параметры:

- c - сложность
- r - степень освоения тем, необходимая для решения
- a - максимальная степень освоения тем, возникающая в результате решения

Каждой задаче $p \in P$ поставим в соответствие тройку (c, r, a)

$$e\colon P \to \mathbb{N} \times \bar{S}^m \times \bar{S}^m = P^e.$$

Определим отображение:

$$\varphi\colon \bar{S}^m \times P^e \to \bar{S}^m$$

$$\varphi(l, e(p)) = \frac{1}{c} \cdot (r - l) + \|(r - l)\| \cdot a$$

Заметим, что максимум выбранной функции существует при любом нормировании образа.

Задача 2. *Построить такое продолжение L_{i+d} заданной последовательности L_i, что:*

$$\varphi(l_i, e(p)) \to \max_{p \in P},$$

где образ отображения нормирован некоторой метрикой.

Выбор метрики определяет способ выбора оптимальной задачи. В случае, например, манхэттенской метрики $\| \ \|_1$, оптимальной задачей является та, которая закрепляет много различных еще не освоенных тем.

Стоит отметить, что предположение о том, что студент решит назначенную задачу - вероятностное, а значит последующие предположения должны, вообще говоря, опираться на условную вероятность предшествующих событий.

3. Применение методов искусственного интеллекта для планирования персональной стратегии обучения

3.1. Построение дискретного пространства признаков

В рамках данного курса разработано пространство признаков с общими и тематическими навыками.

Общие навыки:

Свойство	*Интерпретация*
Компилируется	$\{0, 1\}$
Протестировано	$\{0, \ldots, 100\}$
Уникально	$\{0, \ldots, 100\}$
Сдано вовремя	$\{0, \ldots, 100\}$
Соответствует ограничениям	$\{0, 1\}$
Оформлено в едином стиле	$\{0, 1\}$

Оценка признаков уникальности, соответствия ограничениям и оформления в едином стиле представляют собой отдельные нетривиальные задачи.

Тематические навыки:

Свойство	*Интерпретация*
Битовые операции	$\{0, 1\}$
Сортировки	$\{0, 1\}$
Алгебра логики	$\{0, 1\}$
k-значная логика	$\{0, 1\}$
Структуры и классы	$\{0, 1\}$

Оценка тематических свойств в некоторых случаях также представляет собой непростую задачу. Для некоторых свойств были созданы модули автоматического их обнаружения в решениях.

3.2. Планирование персональной стратегии обучения

В задачах продолжения последовательностей хорошо себя зарекомендовали рекуррентные нейронные сети, а именно: LSTM (Long-Short Term Memory) [5], CW-RNN (Clockwork Recurrent Neural Network) [6], GRU (Gated Recurrent Unit) [7] и др. Данные сети были использованы в экспериментах с искусственными данными процессов обучения 4 групп студентов по 50 человек на курсе «Практикум на ЭВМ» Ташкентского филиала МГУ им. М. В. Ломоносова.

Каждому студенту были назначены 3 случайных задачи разной сложности, остальные задачи назначались на основании принадлежности студента к одному из выделенных классов:

- отличники - решают задачи любой сложности
- хорошисты - в сложных задачах иногда испытывают трудности
- троечники - затрудняются решить задачи любой сложности в некоторых разделах
- двоечники - с трудом решают самые простые задачи

В процессе имитации обучения некоторые студенты переходили из одного класса в другой в результате ускоренного роста генерируемых результатов начиная с некоторого момента. Для такого класса студентов

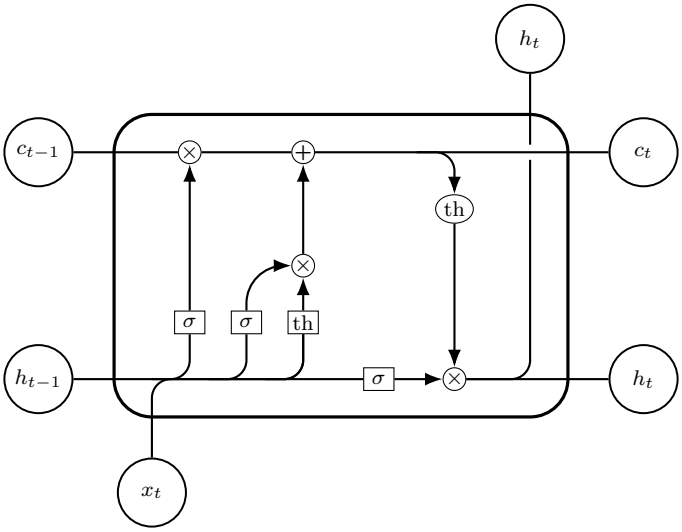


Рис. 1. Ячейка LSTM

необходимо учитывать *давние* результаты и не повышать сложность назначаемых задач слишком быстро, давая им возможность закрепить результат.

Используя построенную в итоге историю обучения применим нейронные сети для предсказания успеха в решении назначенных задач начиная с 4-ой. Опустим вероятностную природу процесса планирования и будем предполагать, что каждое новое предположение производится на основании уже произошедших событий. Для каждого метода измерим точность предсказания, как отношение верных предсказаний к общему их количеству.

Функционально ячейка LSTM задается следующими уравнениями:

$$h_t = o_t \cdot \mathrm{th}(\bar{c}_t)$$
$$c_t = f_t \cdot c_{t-1} + i_t \cdot \bar{c}_t$$

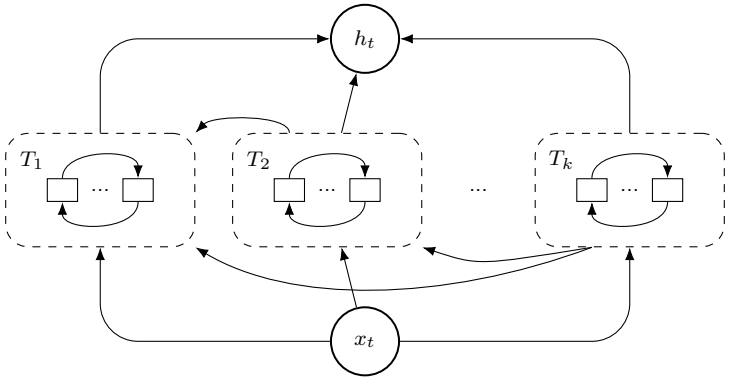


Рис. 2. Архитектура CW-RNN: нейроны скрытого слоя разделены на k групп. Нейроны i-ой группы подключены друг к другу и имеют период T_i. Группа i подключена к группе j, если $T_i < T_j$.

где

$$\begin{aligned} \bar{c}_t &= \mathrm{th}(W_c \cdot [h_{t-1}, x_t] + b_c) \\ f_t &= \sigma(W_f \cdot [h_{t-1}, x_t] + b_f) \\ i_t &= \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \\ o_t &= \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \end{aligned}$$

Уравнения ячейки GRU принимают следующий вид:

$$h_t = (1 - z_t) \cdot h_{t-1} + z_t \cdot \hat{h}_t$$

где

$$\begin{aligned} z_t &= \sigma(W_z \cdot [h_{t-1}, x_t]) \\ r_t &= \sigma(W_r \cdot [h_{t-1}, x_t]) \\ \hat{h}_t &= \mathrm{th}(W \cdot [r_t \cdot h_{t-1}, x_t]) \end{aligned}$$

Результат применения нейронных сетей представлен в таблице 1.

Для конечномерного дискретного пространства признаков можно применить алгоритм предсказания следующего элемента в последовательности, основанный на модели *компактное дерево предсказаний*

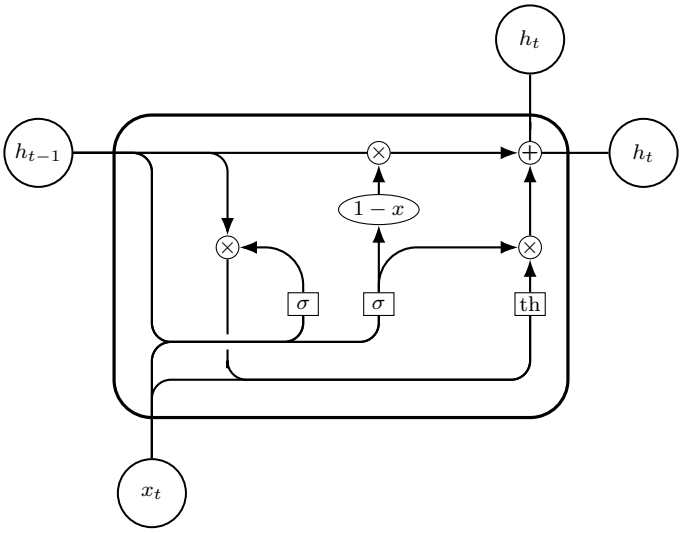


Рис. 3. Ячейка GRU

Нейронная сеть	*Искусственные данные*	*Реальные данные*
LSTM	84.12	56.87
CW-RNN	85.77	54.90
GRU	89.43	58.01

Таблица 1. Точность предсказания, %

(Compact Prediction Tree, CPT, [8]). CPT состоит из трех структур данных: *дерева предсказаний* (PT), *обратного индекса* (II), *массива указателей* (LT).

PT. Пустое дерево предсказаний состоит из одного узла - корня. В процессе обучения модели дерево заполняется по следующему алгоритму: для очередной последовательности проверить наличие ее первого элемента среди потомков корня; если таковой потомок имеется, то перейти к поиску следующего элемента последовательности в потомке; иначе, вставить оставшиеся элементы последовательности ветвью, начиная с последнего узла, для которого нашлись совпадения или корня, если совпадений не было вообще.

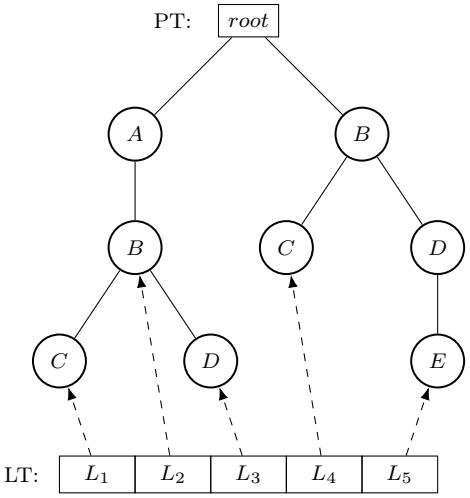
II. Обратный индекс является двумерной двоичной матрицей, строки которой соответствуют уникальным элементам последовательностей, содержащихся в PT. Если элемент x содержится в последовательности y, то в соответствующей ячейке матрицы $\mathrm{II}_{x,y}$ находится 1, иначе - 0.

LT. Массив указателей содержит указатели на последние элементы последовательностей в PT доступные по номерам последовательностей.

Алгоритм: Заполнение CPT

дано: CPT, кривая обучения L
текущий узел ← корень PT
цикл $l \in L$
  если множество потомков текущего узла $\neq \varnothing$ и содержит l
    текущий узел ← потомок, равный l
  иначе
    добавить l в множество потомков текущего узла
    текущий узел ← новый потомок, равный l
  конец если
конец цикла
обновить II и LT

Для заданной последовательности $X = (x_1, x_2, \ldots, x_n)$ процесс предсказания следующего элемента заключается в следующем: для некоторого m, $1 < m < n$, выберем все последовательности, содержащие последние m элементов $\{x_{n-m+1}, x_{n-m+2}, \ldots, x_n\} \subset X$ в любом порядке и в любой позиции. Такие схожие с X последовательности, обозначим их $\mathbb{Y}$, находятся битовым перемножением всех строк II, соответствующих выбранным элементам. Для каждой последовательности $Y \in \mathbb{Y}$ выделим подпоследовательность, начинающуюся с элемента, следующего за



II:

	L_1	L_2	L_3	L_4	L_5
A	1	1	1	0	0
B	1	1	1	1	1
C	1	0	0	1	0
D	0	0	1	0	1
E	0	0	0	0	1

Рис. 4. Пример структуры CPT

Модель	*Искусственные данные*	*Реальные данные*
CPT	94.36	85.55

Таблица 2. Точность предсказания, %

последним элементом схожим с X. Каждому элементу y таких подпоследовательностей поставим в соответствие два числа:

- s - число вхождений элемента y в последовательностях $\mathbb{Y}$
- $c = \frac{s}{r}$, где r - число последовательностей в PT, содержащих y (т.е. сумма строки матрицы II, соответствующей элементу y)

Элементы, соответствующие минимальному s - кандидаты. Искомым элементом является кандидат с наименьшим c.

Результат применения CPT с использованием приведенного выше пространства признаков представлен в таблице 2.

4. Заключение

Описанные выше методы анализа данных применяются в системе поддержки преподавания курса «Практикум на ЭВМ» в Ташкентском филиале МГУ им. М. В. Ломоносова. Курс ставит перед студентами цель

получить навыки разработки на языке C/C++ самодостаточных программных продуктов, решающих поставленные преподавателем задачи. Посредством лекций, семинаров и практических занятий материал курса должен быть постепенно усвоен студентами. Этот курс обширный и покрывает 4 года обучения:

	1 семестр	2 семестр
1 год	Простые алгоритмы	Дискретная математика
2 год	Операционные системы	Основы ООП
3 год	Дискретная оптимизация	Численные методы, часть 1
4 год	Численные методы, часть 2	-

Каждый раздел содержит набор практических задач разной сложности и тематики, а практическим занятиям предшествуют соответствующие лекции или курсы. Согласно плану обучения, студент должен решить некоторое случайное подмножество задач из раздела для перехода к следующему. В противном случае преподаватель может назначить студенту дополнительные задания, обычно заключающиеся в решении нового подмножества задач. В построении этого подмножества преподаватели опираются на рекомендации представленных в статье методов.

Из проведенных экспериментов следует, что применение CPT для решения задачи планирования персональной стратегии обучения выдает неплохой результат, когда подход с использованием нейронных сетей указывает на необходимость уточнения пространства признаков или корректировки архитектуры применяемых нейронных сетей. В этом направлении будут проводиться дальнейшие исследования.

Список литературы

[1] Алисейчик П. А., Вашик К., Ж. Кнап, Кудрявцев В. Б., Шеховцов С. Г., Строгалов А. С., "Моделирование процесса обучения", *Интеллектуальные системы*, **10** (2006), 85.

[2] Алисейчик П. А., Вашик К., Кудрявцев В. Б., Строгалов А. С., "Компьютерные обучающие системы", *Интеллектуальные системы*, **8** (2004), 20.

[3] Алисейчик П. А., Кудрявцев В. Б., Строгалов А. С., "Автоматная модель обучения", *Материалы VIII Международного семинара «Дискретная математика и и ее приложения», изд мехмат ф-т МГУ*, 2004, 7.

[4] Аширматов Б. Д., "Многомерный поиск в обучающих системах", *Интеллектуальные системы*, **15** (2011), 14.

[5] Sepp Hochreiter, Jürgen Schmidhuber, "Long Short-Term Memory", *Neural Computation*, 1997, № 9(8), 1735-1780.

[6] Jan Koutník, Klaus Greff, Faustino Gomez, Jürgen Schmidhuber, “A Clockwork RNN”, 2014, arXiv: https://arxiv.org/abs/1402.3511.

[7] Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, “Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation”, 2014, arXiv: https://arxiv.org/abs/1406.1078.

[8] Ted Gueniche, Philippe Fournier-Viger, Vincent S. Tseng., “Compact Prediction Tree: A Lossless Model for Accurate Sequence Prediction”, Advanced Data Mining and Applications: 9th International Conference, 2013, № II.

[9] Ахо А., Ульман Дж., *Теория синтаксического анализа, перевода и компиляции*, **1**, Мир, 1978.

[10] Котов В. Е., Сабельфельд В. К., *Теория схем программ*, Наука, 1991.

[11] Камкин А. С., *Введение в формальные методы верификации программ*, Москва, 2018.

Using AI approach to plan personal learning strategy

Bektashev R.A.

Abstract

In this article the problem of planning personal learning strategy is set and results of applying various approaches to its solution are presented.

Keywords: computer education system, sequence elements prediction, neural network, recurrent neural network, compact prediction tree