Министерство образования и науки Российской Федерации
Московский физико-технический институт (государственный университет)
Факультет управления и прикладной математики
Вычислительный центр им. А. А. Дородницына РАН
Кафедра «Интеллектуальные системы»

Варфоломеева Анна Андреевна

Методы структурного обучения в задаче обобщения структур ранжирующих моделей

010656 — Математические и информационные технологии

Выпускная квалификационная работа магистра

Научный руководитель:
д. ф.-м. н. Стрижов Вадим Викторович

Москва
2015

Содержание

Аннотация

В работе решается проблема порождения глобальной ранжирующей функции в задаче информационного поиска. Требуется найти такую ранжирующую функцию, которая определяла бы релевантность различных документов поступающим запросам. Ранжирующая функция определяется как суперпозиция заданных элементарных функций. Для решения задачи поиска суперпозиции глобальной ранжирующей функции используется метод структурного обучения — метод восстановления структуры объекта по его описанию. Для обучения метода структурного обучения получена выборка "подколлекция-модель"с помощью генетического алгоритма над множеством элементарных функций. Рассматривается ряд методов прогнозирования структуры функции по матрице вероятности переходов, их работа тестируется на синтетических данных. Найденная глобальная ранжирующая функция тестируется при помощи функционала MAP на полной коллекции документов. Для проведения эксперимента используется выборка, состоящая из множеств коллекций документов конференции EURO 2010.

Ключевые слова: *структурное обучение, ранжирующая функция, нелинейные модели, индуктивное порождение, информационный поиск.*

Введение

Актуальность темы. Во многих задачах информационного поиска необходимо, получая на вход запрос, сформировать выборку документов, возвращаемых в порядке убывания релевантности данному запросу. Для формирования такой выборки необходима процедура оценки релевантности документов. Один из методов такого оценивания — использование ранжирующей функции. Ранжирующая функция определяется на признаках пары "документ-запрос" и согласно этим признакам формирует оценку релевантности запросу соответствующего документа. Таким образом, присваивая каждому документу релевантность, функция формирует выборку, соответствующую получаемому запросу.

Существующие ранжирующие функции делятся на два типа: либо они были предложены эвристически, либо были получены с помощью некоторого переборного алгоритма. Для ранжирующих функций первого типа не доказана их оптимальность. Алгоритмы поиска ранжирующих функций имеют значительную вычислительную сложность в связи с необходимостью перебора большого числа моделей. В связи с этим, такие алгоритмы имеют жетские ограничения на структурную сложность функции. Кроме того, многие алгоритмы находят лишь локально оптимальные ранжирующие функции, которые не могут быть обобщены для больших коллекций документов.

Цель работы. Предложить метод прогнозирования глобально оптимальной ранжирующей функции для коллекции документов, опираясь на известные прецеденты выбора ранжирующих функций для подколлекций документов.

Методы исследований. Для достижения поставленных целей использованы методы регрессионного анализа, структурного обучения, информационного поиска. Для программной реализации разработанного алгоритма прогнозирования глобальной ранжирующей функции использовалась среда MATLAB.

Основные положения, выносимые на защиту.

- Метод прогнозирования структуры суперпозиции функции;

- метод динамического программирования поиска оптимального дерева суперпозиции.

Научная новизна.

- Предложен алгоритм прогнозирования ранжирующей функции;
- предложен новый метод записи структуры суперпозиции функции;
- предложены методы поиска оптимального дерева суперпозиции в матрице вероятностей переходов.

Практическая ценность. Разработан программный модуль, который:

- находит ранжирующие функции для подколлекций документов, используя генетический алгоритм MVR;
- оценивает вероятности наличия в глобальной ранжирующей функции элементарных функций;
- прогнозирует структуру глобальной ранжирующей функции;
- иллюстрирует результаты.

Публикации по теме.

1. *Варфоломеева А.А.* Локальные методы прогнозирования с выбором метрики // Машинное обучение и анализ данных, 2012. Т.1, Вып. 3. Стр. 367–375.
2. *Варфоломеева А.А., Стрижов В.В.* Алгоритм разметки библиографических списков методами структурного обучения // Информационные технологии, 2014, Вып. 7. Стр. 11–15.

Обзор литературы. Наиболее известные подходы к нахождению ранжирующих функций основыаются на теоретических соображениях. Такие эвристические функции дают качественные оценки релевантности [1–6], однако было показано существование ранжирующих функций, результат которых в среднем лучше традиционно используемых функций на случайной коллекции [7].

В работе [7] использовался переборный алгоритм поиска ранжирующих функций. Они представлялись в виде суперпозиции элементов заданной грамматики — набора порождающих функций и переменных. В качестве переменных использовались признаками пары документ-запрос. Однако из-за высокой вычислительной сложности переборного алгоритма в работе [7] на сложность суперпозиции были наложены низкие ограничения: рассматривались суперпозиции, состоящие из не более чем 8 элементов грамматики.

Другим подходом к поиску функциональной зависимости по набору исходных данных является символьная регрессия [8]. Джон Коза предложил реализацию этого метода с помощью аналога генетического алгоритма [9]. Иван Зелинка предложил дальнейшее развитие этой идеи [10], получившее название аналитического программирования.

Алгоритм построения математической модели в аналитическом программировании выглядит следующим образом: задан набор элементарных функций (например, степенная функция, $+$, sin, tan и др.), из которых можно строить различные формулы. Начальный набор формул строится либо произвольным образом, либо на базе некоторых предположений эксперта. Затем на каждом шаге производится оценка каждой из формул согласно некоторой функции качества. На базе этой оценки у части формул случайным образом заменяется одна элементарная функция на другую (например, sin на cos или $+$ на $\times$), а у некоторой другой части происходит взаимный попарный обмен подвыражениями. Данный подход может быть описан в терминах генетического алгоритма: каждый индивид является формулой, изображенной в свою очередь в виде дерева. Тогда набор формул, существующий в определенный момент, представляет собой одно поколение. При этом хромосомы представляются поддеревьями, и, в отличие от классического генетического алгоритма, могут быть различного размера (длины). Описанный выше обмен подвыражениями представляет собой в этом случае генетическое скрещивание, замена одной элементарной функции у некоторых деревьев — мутацию. При этом возникает ряд сложностей, связанных с областями определения и арностями элементарных функций, записанных в узлах дерева. Данный метод фактически является ненаправленным поиском и перебирает большое количество неподходящих деревьев до того момента, как приблизится к

оптимуму.

Модификация этого метода предложена в работах [11, 12], где с помощью проверки на наличие тождественных деревьев значительно сокращался объем перебора.

Генетические алгоритмы также получили распространение в задачах информационного поиска. Ранее было предложено использовать этот подход к поиску оптимального описания документов при индексировании [13, 14], при решении задачи кластеризации документов, ко-релевантных для множества запросов [15, 16]. Также генетические алгоритмы использовались для настройки параметров запросов [17, 18], для нахождения множества ключевых слов для бинарного векторного описания документа [19], для нахождения оптимальных коэффициентов для линейной суперпозиции ранжирующих функций [20, 21].

В задаче поиска ранжирующей функции генетические алгоритмы также были использованы в работах [22, 23], однако в них не накладывались регуляризаторы на функционалы качества модели, в результате чего получаемые модели имели излишнюю сложность и были неинтерпретируемыми. Кроме того, было показано, что генетический алгоритм порождения моделей имеет свойство застревать в локальных минимумах — с некоторого момента все модели популяции очень "похожи"друг на друга, и алгоритм не может сгенерировать новые.

Альтернативой аналитическому программированию можно считать подход обучения в глубину (Deep Learning) [24, 25]. Этот подход заключается в иерархическом представлении данных, в котором на нижнем уровне находятся сам набор данных, а на каждом уровне выше — более абстрактное его представление, которое представляет собой некую скрытую комбинацию из данных, найденных раннее. Так, например, метод получил широкое распространение в обработке и классификации изображений. Набором данных является матрица яркости пикселей некоторого изображения, на следующем уровне — данные о выраженных геометрических закономерностях на изображении (отрезки, кривые, окружности), на более высоких уровнях иерархии — более сложные и абстрактные выявленные закономерности. В основном, для поиска таких представлений используется нейронная сеть с большим количеством скрытых слоев [26]. В случае изображений [27], когда большое значение имеют локальные признаки, сеть содержит много сверточных слоев, осуществляющих поиск именно таких

признаков. При использовании методов обучения в глубину для обработки изображений, нейронная сеть обучается, получая на вход и на выход одинаковый набор данных, после чего каждый из уровней сети представляется как информация о данных на определенном уровне абстракции. Также значения на некотором слое сети могут использоваться как набор признаков данного изображения для последующей классификации. Однако в задачах поиска функциональной зависимости этот метод пока слабо распространен в силу того, что сложно учитывать и описать ограничения на построение дерева суперпозиции во внутренних слоях сети.

В данной работе для поиска ранжирующей функции предлагается рассмотреть метод, основанный на прогнозировании структуры функциональной зависимости. Предполагается, что функциональная зависимость существенно нелинейна и, аналогично описанному выше, является суперпозицией элементарных функций. При этом делается ограничение на максимальную сложность модели. Дерево суперпозиции представляется в виде матрицы. В таком виде задача сводится к задаче структурного обучения, описанной, например, в [28–30]. Методы структурного обучения решают задачу нахождения структуры или зависимости, имеющейся внутри исходных данных. Например, метод широко применим для синтаксического разбора предложений [31]: на вход алгоритма подается предложение, на выход алгоритм должен предоставить дерево синтаксического разбора. Также методы структурного обучения используются в области компьютерного зрения: в работе [32] решается задача определения глубины нахождения объекта на изображении, т.е. алгоритм строит граф, первый слой которого отвечает за объекты, наиболее близкие к объективу, второй слой — за объекты, находящиеся за первыми, и так далее. В работе [33] методы прогнозирования структуры были использованы для разметки библиографических списков. Как видно, именно методам структурного обучения свойственно работать с данными, имеющими ограничения сложного вида на выходе. При этом стоит отметить, что указанное выше ограничение на максимальную сложность модели не является таким строгим, как в случае полного перебора, так как методы структурного обучения не являются переборными, за счет чего имеют намного меньшую сложность.

1 Постановка задачи

Постановка задачи разбивается на две части — получение выборки для алгоритма структурного обучения и поиск глобальной ранжирующей функции с помощью найденной выборки.

1.1 Получение выборки

Задана выборка документов $= \{d_i\}_{i=1}^{N}$, где N — общее число документов. Задано множество запросов к этой коллекции $Q = \{q_i\}_{i=1}^{|Q|}$. Также задано разбиение коллекции C на подколлекции $C_k \subseteq C$. Для каждого документа d_j и запроса q_i экспертно заданы ранги релевантности документов h_{ij}:

$$h_{ij} : q_i \times d_j \to \mathbb{Y} = \{0, 1, \emptyset\},$$

где 1 соответствует релевантному документу для данного запроса, 0 — нерелевантному, а $\emptyset$ означает, что экспертная оценка не проставлена.

Введем переменную x — нормализованную частоту встречаемости слова w в документе d, и переменную y — нормализованную частоту встречаемости слова w в коллекции:

$$x_w^d = t_d^w \log\big(1 + c \cdot \frac{l_{avg}}{l_d}\big),$$

$$y_w = \frac{N_w}{N},$$

где w — одно из слов запроса q, N_w — количество документов в коллекции, содержащих w, N — общее число документов в коллекции, t_d^w — частота встречаемости слова w в документе d, l_d — длина документа в коллекции, l_{avg} — средняя длина документа в коллекции, c — некоторая константа.

Задано множество $\mathcal{G}$ порождающих функций. Для каждой функции $g : \mathbb{R} \times \ldots \times \mathbb{R} \to \mathbb{R}$ из набора $\mathcal{G}$ определены её арность $v = v(g)$, области определения и значений: $\mathrm{dom}(g), \mathrm{cod}(g)$. Известно множество $\mathcal{F}$ суперпозиций порождающих функций, при этом заданы правила индуктивного порождения функции $f \in \mathcal{F}$:

$$\mathcal{F} = \{f_s \mid \mathbf{f}_s : (\hat{\mathbf{w}}_k, \mathbf{X}) \mapsto \mathbf{y}, s \in \mathbb{N}\}.$$

Будем искать модели, наилучшим образом аппроксимирующие функции h_{ij} среди множества суперпозиций $\mathcal{F}$.

Релевантность данного документа определяется как сумма значений функции для каждого слова из запроса.

Нужно найти модель $f_j \in \mathcal{F}$, оптимальным образом аппроксимирующую экспертно заданные ранги для отдельной подколлекции документов C_k. Проверка качества производится при помощи функционала MAP:

$$m(f, C, Q) = \sum_{i=1}^{|Q|} \frac{P(q_i)}{|Q|}, \tag{1}$$

где $P(q_i)$ — отношение числа релевантных для запроса q_i согласно ранжирующей функции f документов к экспертному числу релевантных документов для этого же запроса. Таким образом ставится задача максимизации для каждой подколлекции документов в отдельности:

$$f_j = \arg\max_{f \in \mathcal{F}}(m(f, C_j, Q)), \tag{2}$$

В результате получена выборка $Q \times C_j \to f_j$, где f_j — оптимальная ранжирующая функция для множества запросов Q к кластеру C_j. Обозначим множество функций f_j через H.

1.2 Структурное обучение

На данном этапе дана выборка $Q \times C_j \to f_j$. Ставится задача отыскания оптимальной глобальной для всей коллекции документов $C = \bigcup_{j=1,\dots,l} C_j$ и запросов Q ранжирующей функции f из порождаемого множества моделей $\mathcal{F} = \{f_s\}$, где $\mathbb{W}$ — пространство параметров, доставляющую минимум заданной функции ошибки, определяемой ниже.

Необходимо найти такой вектор $\hat{\mathbf{w}}$ параметров метода $a : Q \times C \to f$, который доставлял бы минимум ошибки S:

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{w} \in \mathbb{W}} S(\mathbf{w} \mid H, Q \times C). \tag{3}$$

В качестве функции ошибки S используется сумма квадратов регрессионных остатков:

$$S(\mathbf{w} \mid H, Q \times C) = \sum_{i=1}^{l} \parallel a(\mathbf{w} \mid Q \times C_i) - f_i \parallel_2, \tag{4}$$

где l — общее число кластеров. Качество полученной функции для всей коллекции также тестируется при помощи функционала MAP.

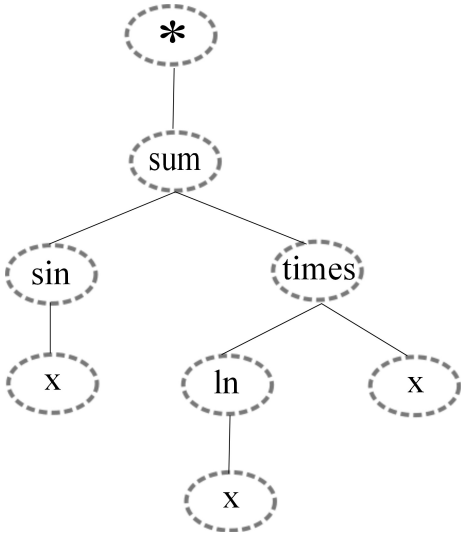

Рис. 1: Пример дерева суперпозиции $f = \sin(x) + (\ln x)x$.

1.3 Способ задания структуры суперпозиции

Каждой суперпозиции f ставится в соответствие дерево Γ_f вида (рис.1), строящееся по следующим правилам:

- корнем дерева является специальный символ “ * ”, имеющий одну дочернюю вершину;
- в остальных вершинах V_i дерева Γ_f находятся соответствующие порождающие функции g_i из набора $\mathcal{G}$;
- число дочерних вершин V_j у некоторой вершины V_i равно арности соответствующей функции g_i: $v = v(g_i)$;
- область определения порождающей функции дочерней вершины V_j содержит область значений функции родительской вершины V_i: $\mathrm{dom}(g_i) \supset \mathrm{cod}(g_j)$;
- в листьях дерева Γ_f находятся свободные переменные x_i.

Пронумеруем вершины такого дерева по порядку, соответствующему порядку обхода в ширину, начиная с корня дерева.

Каждому дереву Γ_f ставится в соответствие бинарная трехиндексная матрица Z размера $s \times s \times (l+n)$, где s — максимально допустимая сложность суперпозиции, l – число элементарных функций набора $\mathcal{G}$, $n = 2$ — число свободных переменных

функции. Объединим набор элементарных функций G и набор свободных переменных $x_i, i = 1, \ldots, n$ в одно множество $G^* = G \cup \{x_i, i = 1, \ldots, n\}$, и пронумеруем элементы этого нового множества.

Элементы матрицы Z отвечают за наличие ребра между двумя вершинами в дереве. Первый индекс матрицы отвечает за номера вершин, являющихся началом ребра, т.е. относящимся к родительской вершине. Второй индекс матрицы отвечает за номера вершин, являющихся концом ребра в дереве. По третьему индексу матрицы определяется та элементарная функция g_s или свободная переменная x_i, которая находится на конце данного ребра: по индексу выбирается элемент из пронумерованного множества G^*. Например, если $Z(i, j, k) = 1$, это значит что в дереве существует ребро (i, j), и j-ая вершина отвечает за k-ый элемент из множества G^*.

На матрицу Z по построению накладываются следующие ограничения:

- для фиксированного первого индекса i в срезе матрицы $Z_i = Z(i, j, k), j = 1, \ldots s, k = 1, \ldots l + n$ содержится либо количество единиц, равное арности $v = v(g_r)$ элементарной функции g_r, записанной в вершине i, либо ноль;

- для фиксированного второго индекса j в срезе матрицы $Z_j = Z(i, j, k), i = 1, \ldots s, k = 1, \ldots l + n$ может содержаться только одна единица (у вершины в дереве только один родитель);

- матрица Z — верхне-треугольная с нулевой диагональню по первым двум индексам, т.е. $\forall i = 1, \ldots, s, \forall j \leqslant i, \forall k = 1, \ldots l + n : Z(i, j, k) = 0$.

Обозначим для удобства множество матриц, удовлетворяющих данным условиям как $\mathcal{M}$.

1.4 Уточнение постановки задачи структурного обучения

Поскольку по матрице из множества $\mathcal{M}$ можно однозначно восстановить суперпозицию функции, задача прогнозирования суперпозиции f сводится к поиску матрицы $\mathbf{Z}_f$ из множества $\mathcal{M}$, максимизирующей вероятность переходов в дереве суперпози-

ции:

$$\mathbf{Z}_f = \arg\max_{\mathbf{Z}\in\mathcal{M}} \sum_{i,j,k} P_{ijk} \times Z_{ijk}, \tag{5}$$

где матрица вероятностей переходов $\mathbf{P}$ определяется с помощью векторной логистической регрессии с функцией ошибки, соответствующей гипотезе порождения данных биномиальным распределением.

2 Генетический алгоритм порождения моделей

Для поиска ранжирующих функций, оптимальных для каждой подколлекции в отдельности, будем использовать генетический алгоритм. Для каждой подколлекции C_i и множества запросов Q имеем набор значений переменных x и y, определяющих пару "документ — запрос". Фиксируем грамматику G, состоящую из переменных двух типов и порождающих функций — унарных и бинарных. Функция f, аппроксимирующая искомую ранжирующую функцию h_i, ищется в виде суперпозиции элементов грамматики.

Используется генетический алгоритм порождения моделей, являющийся обобщением алгоритма из [22] на случай параметрических моделей. На первой итерации создаётся некоторое начальное множество случайных моделей и некоторых моделей из множеств $\mathcal{F}$ и $\mathcal{T}$. Опишем итерацию алгоритма:

1. Выбирается некоторое подмножество моделей, лучших в смысле функционала качества MAP.

2. Если лучшая модель среди выбранных удовлетворяет требуемой точности, то алгоритм заканчивается на этом шаге.

3. Используем алгоритм cross-mutation для порождения новых моделей из множества имеющихся. Формируем новое множество моделей. Повторяем итерацию.

3 Решение задачи структурного обучения

Пусть с помощью векторной логистической регрессии найдена матрица вероятностей переходов P_f. Ставится задача отыскания матрицы $\mathbf{Z}_f$ из допустимого множества

матриц $\mathcal{M}$, удовлетворяющей условию 5. Для этого разобьем матрицу $\mathbf{P}_f$ на два блока. Блок $P'_{s\times s\times l}$:

$$P'_{ijk} = p(v_i \to v_j, v_j = g_k)$$

отвечает за вероятности переходов между порождающими функциями. Блок $P''_{s\times s\times n}$:

$$P''_{ijk} = p(v_i \to v_j, v_j = x_k)$$

содержит значения вероятностей перехода от порождающих функций к независимым переменным. Введем понятия открытой вершины. Назовем вершину дерева v_i — *открытой*, если она относится к порождающей функции, и существует вершина, являющаяся для вершины v_i родительской, но у вершины v_i нет дочерних вершин:

$$(i \le l)\&(\exists j : (j,i) = 1)\&(\nexists k : (i,k) = 1).$$

3.1 Жадный алгоритм прогнозирования структуры модели

Зададим значение K — максимально допустимую сложность суперпозиции. Опишем "жадную"процедуру построения оптимального дерева $\hat{\Gamma}_f$.

- На нулевом шаге процедуры объявляем корень дерева открытым: $i = 1$.
- Пока количество вершин дерева не превышает K, повторяем:
  1. выбираем максимальные вероятности переходов $c^i_{jk} = \max\limits_{j=1,\dots,s,k=1,\dots,l+n} P_{ijk}$ для всех открытых вершин i;
  2. достраиваем матрицу из условия максимизации вероятности перехода: $(j^*,k^*) = \arg\max\limits_{(j,k)} c^i_{jk}, \quad Z(i,j^*,k^*) = 1$;
  3. добавляем j^* к списку открытых вершин, если $(i,j^*,k^*) \in P'$;
- если количество единиц превышает K, ставим в соответствие всем открытым вершинам независимые переменные: $(j^*,k^*) = \arg\max\limits_{(j,k)} P''_{ijk}, \quad Z(i,j^*,k^*) = 1$ для всех i-открытых.

Процедура может быть прервана, если множество открытых вершин пусто, но сложность суперпозиции еще не превысила заданную максимальную сложность K. В таком случае построенная оптимальная суперпозиция будет иметь меньшую сложность.

3.2 Модификация жадного алгоритма прогнозирования структуры модели

В отличие от предыдущего метода, который на каждой итерации сохраняет только один лучший вариант построения дерева, в данном методе будем хранить на каждом шаге некоторое множество лучших деревьев-кандидатов. На каждой итерации алгоритма рассматриваются способы достроить каждое из деревьев-кандидатов, после чего сохраняются и переходят на следующую итерацию только лучшие из достроенных деревьев.

Опишем алгоритм более формально. Зададим значение K — максимально допустимую сложность суперпозиции. Также требуется задать R — поддерживаемое число оптимальных суперпозиций. Будем поддерживать R лучших деревьев-кандидатов $Z_r, r = 1, ..., R$.

- Объявляем корень дерева открытым: $i = 1$.
- Пока количество вершин дерева не превышает K, повторяем:
    1. для каждого дерева-кандидата Z_r для всех открытых вершин i сортируем по убыванию вероятности переходов из них $P^r_{ijk}, j = 1, \ldots, s, k = 1, \ldots, l+n$, запишем R лучших значений как $T_{i^r} = \{P^r_{i^r jk}, j = 1, ..., s, k = 1, \ldots, l + n\}$, $|T_{i^r}| = R$ и вершины, соответствующие этим значениям;
    2. соединяем все отобранные лучшие вероятности переходов T_{i^r}, выбираем из них только R лучших значений и соответствующие им новые вершины деревьев-кандидатов $(j_r, k_r), r = 1, \ldots, R$;
    3. достраиваем деревья-кандидаты, используя найденные лучшие варианты: для каждого нового перехода T_{i^r} берем отвечающее ему дерево-кандидат Z^r и достраиваем его: $Z^r(i^r, j_r, k_r) = 1$;
    4. добавляем j_r к списку открытых вершин для матрицы Z^r, если $(i^r, j_r, k_r) \in P'$;
- если количество единиц превышает K, для новых переходов рассматриваем только независимые переменные, т.е. только блок матрицы P''.

Также, как и в предыдущем случае, процедура может быть прервана заранее, если множество открытых вершин пусто, и не достигнута максимально допустимая сложность суперпозиции K.

3.3 Метод динамического программирования для прогнозирования структуры модели

Назовем дерево *недостроенным*, если его множество открытых вершин не пусто. Пусть есть некоторое недостроенное дерево суперпозиции Г, с ненулевым числом открытых вершин. Мы хотим достроить дерево оптимальным с точки зрения динамического программирования способом. Для этого,фиксируя одну из этих открытых вершин с номером j^* и хранящейся в ней функции k^*, будем искать оптимальные деревья, для которых j — корень.

Опишем рекурсивную процедуру поиска оптимальной структуры поддерева с вершиной в точке j $FindOptTree(j,k)$. Процедура $FindOptTree(j,k)$ просматривает все возможные переходы из заданной вершины j. Для каждого перехода в вершину j_1 процедура оценивает вероятность поддерева, начинающегося в j, содержащего данный переход (j, j_1) и продолжающегося оптимальным поддеревом из вершины j_1, т.е.рекурсивно вызывая себя из конечной вершины перехода.

Более формально, если есть некоторое недостроенное дерево Г, ему соответствует матрица Z, и процедура вызывается в одной из его открытых вершин $FindOptTree(j^*,k^*)$;

- если k^* — независимая переменная, то процедура завершена, возвращаем $(p_{j^*}=1, Z=Z)$;
- если k^* — элементарная функция, то:
- для всех возможных переходов из j^* $(j^*,j,k), j=j^*+1,\ldots,s, k=1,\ldots,l+n$
  1. считаем $(p_j, Z_j) = FindOptTree(j,k)$;
  2. возвращаем новую вероятность поддерева

$$p_{j^*} = \max_{j=j^*+1,\ldots,s, k=1,\ldots,l+n} P_(j^*,j,k) * p_j$$

3. достраиваем возвращаемую матрицу $Z = Z_j, \quad Z(j^*, j', k') = 1$, где

$$(j', k') = \underset{j=j^*+1,\ldots,s,k=1,\ldots,l+n}{\arg\max} P_(j^*, j, k) * p_j.$$

Заметим, что в данном методе нет контроля за максимально допустимой сложностью суперпозиции.

Для того, чтобы с помощью предлагаемого метода найти дерево суперпозиции полностью, требуется зафиксировать начальную нулевую матрицу переходов, объявить корень дерева $i = 1$ открытой вершиной, и вызвать процедуру из этой вершины.

4 Вычислительный эксперимент

4.1 Вычислительный эксперимент на синтетических данных

Для тестирования методов прогнозирования структуры модели была сгенерирована выборка синтетических данных следующим образом. Экспертно задан набор порождающих функций $\mathcal{G}$, для каждой из которых известны арность функции $v = v(g)$, области определения и значений: $dom(g), cod(g)$. По набору $\mathcal{G}$ было построено конечное множество суперпозиций $\mathcal{F}$ — библиотека функций. Экспертно заданы значения независимых переменных $\mathbf{X}$. Значения зависимых переменных заданы как

$$\mathbf{y}_s = f_s(\mathbf{w}_s, \mathbf{X}) + \tau_f,$$

где $\mathbf{w}_s$ — вектор параметров модели, и τ_f — шумовая добавка.

Для обучения алгоритма векторной логистической регрессии была использована стандартная нейронная сеть в среде Matlab с двумя скрытыми слоями, имеющая на выходном слое сигмоидную функцию активации. На вход такой нейросети подается сгенерированная регрессионная выборка, выходом алгоритма является матрица вероятностей $\mathbf{P}$. Полученная таким образом матрица вероятностей поступает на вход указанным выше алгоритмам построения оптимального дерева $\hat{\Gamma}_f$.

Для тестирования качества алгоритмов использовался метод LOO(Leave-One-Out), по которому множество регрессионных выборок разбивается таким образом, что в обучении алгоритма использовались все выборки, за исключением одной. Кон-

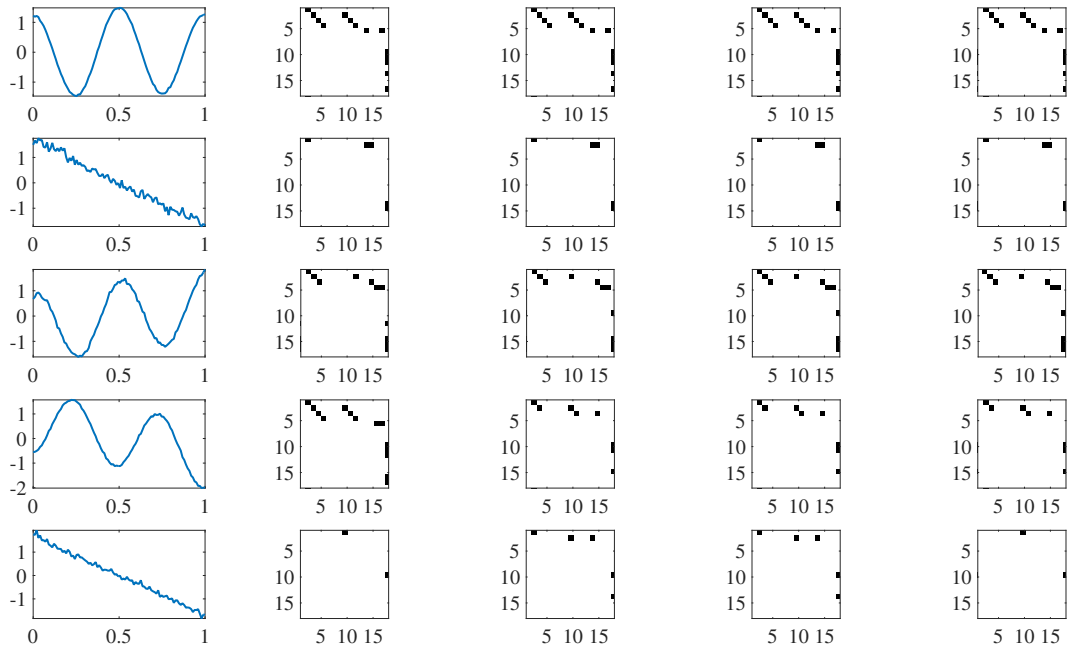

Рис. 2: Тестирование методов прогнозирования.

троль проводился на одной оставшейся выборке, для которой по полученному алгоритму вычислялось оптимальное дерево суперпозиции $\hat{\Gamma}_k$ и строилась модель $\hat{f}_k$.

На рис.2 изображены результаты тестирования предлагаемых методов. Для наглядности, получаемая трехмерная матрица связи переведена в двумерную матрицу смежности графа — дерева функции. В левой колонке изображен вид исходных моделей, структуру которых требуется прогнозировать. Вторая колонка — истинные структуры моделей. Три следующие колонки содержат результаты прогнозирования предлагаемых методов: жадного алгоритма, модификации жадного алгоритма с сохранением нескольких лучших моделей, и метода динамического программирования.

Из рис.2 видно, что жадный алгоритм склонен усложнять структуру модели, в то время как другие методы справляются с моделями малой структурной сложности лучше.

4.2 Вычислительный эксперимент на коллекции аннотаций EURO 2010

Для проверки работы метода прогнозирования структуры ранжирующей функции была использована коллекция аннотаций к докладам на конференции EURO 2010. Коллекция содержит 1663 документа, которые распределены по 24 областям (areas) и 178 направлениям (streams). В качестве документов использовались тексты аннотаций, в качестве запросов к ним — названия докладов. Текст считался релевантен запросу, если данный документ и документ, откуда взято название, являющееся запросом, пинадлежат одному направлению. Таким образом была составлена выборка документов $= \{d_i\}_{i=1}^{N}$, множество запросов к этой коллекции $Q = \{q_i\}_{i=1}^{|Q|}$, и для каждого документа d_j и запроса q_i экспертно заданы ранги релевантности документов h_{ij}:

$$h_{ij} : q_i \times d_j \to \mathbb{Y} = \{0, 1, \emptyset\}.$$

В качестве разбиения коллекции C на подколлекции $C_k \subseteq C$ использовалась принадлежность документа некоторой области. Таким образом, было получено 24 подколлекции документов, 1663 запроса к ним и экспертно заданные ранги релевантности.

С помощью генетического алгоритма порождения моделей был получен ряд моделей (табл.1). Эти модели использовались для обучения алгоритма структурного обучения, после чего алгоритму подавалась на вход случайная подвыборка пар “документ-запрос”, и прогнозируемая алгоритмом функция оценивалась с помощью функционала MAP (1). Ряд полученных функций и соответствующих значений функционала содержится в табл.2. Для прогнозирования структуры суперпозиции использовался метод динамического программирования.

Заключение

В работе ставится и решается задача поиска ранжирующей функции, определяющей релевантность документов поступающему запросу. Поставленная задача решается в два этапа: на первом этапе с помощью генетического алгоритма для подколлекций документов находятся локально оптимальные ранжирующие функции, после чего на втором этапе алгоритма эти функции используются для обучения метода струк-

Таблица 1: Порожденные функции

Номер	Символьная запись
1	$\log \frac{x}{y} + \sqrt{y}$
2	$\sqrt{x} - x$
3	$\sqrt{x} - \exp x$
4	$\log \frac{x}{y*y}$
5	$\log \frac{x}{y} + \sqrt{x}$
6	$\log \sqrt{\sqrt{\frac{x}{y}}}$
7	$x + y - x^2$
8	$\log \frac{x}{y} + y$

Таблица 2: Спрогнозированные функции

Номер	Символьная запись	MAP
1	$\frac{\log x + \log y}{\exp x}$	0.321
2	$\sqrt{x} - x$	0.308
3	$\sqrt{x} - \exp x + \sqrt{y} - \log(x)$	0.306
4	$\log \frac{x}{y}$	0.303
5	$\log \frac{x}{y*y}$	0.302
6	$x + y - x^2$	0.297
7	$\log \frac{x}{y}$	0.291

турного обучения, прогнозирующего глобально оптимальную ранжирующую функцию. Предложен ряд методов поиска оптимальной структуры модели по матрице вероятностей перехода. Качество предлагаемых методов прогнозирования структуры проверено на выборке синтетических данных. Работа метода поиска ранжирующей функции проиллюстрирована на коллекции аннотаций к докладам на конференции EURO 2010.

Список литературы

[1] *Salton Gerard, McGill Michael J.* Introduction to Modern Information Retrieval. — New York, NY, USA: McGraw-Hill, Inc., 1986.

[2] *Porter M. F.* Readings in Information Retrieval / Ed. by Karen Sparck Jones, Peter Willett. — San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., 1997. — Pp. 313–316. http://dl.acm.org/citation.cfm?id=275537.275705.

[3] *Shamir Ron, Tsur Dekel.* Faster Subtree Isomorphism. // *J. Algorithms.* — 1999. — Vol. 33, no. 2. — Pp. 267–280.

[4] *Metzler Donald, Croft W. Bruce.* A Markov Random Field Model for Term Dependencies // Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval. — ACM, 2005. — Pp. 472–479. http://doi.acm.org/10.1145/1076034.1076115.

[5] *Amati Gianni, Van Rijsbergen Cornelis Joost.* Probabilistic Models of Information Retrieval Based on Measuring the Divergence from Randomness // *ACM Trans. Inf. Syst.* — 2002. — Vol. 20, no. 4. — Pp. 357–389. http://doi.acm.org/10.1145/582415.582416.

[6] *Clinchant Stéphane, Gaussier Eric.* Information-based Models for Ad Hoc IR // Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval. — SIGIR '10. — New York, NY, USA: ACM, 2010. — Pp. 234–241. http://doi.acm.org/10.1145/1835449.1835490.

[7] *et al. Parantapa Goswam.* Exploring the Space of IR Functions. — 2014. — Pp. 372–384.

[8] *Riccardo Poli William B. Langdon Nicholas F. McPhee.* A Field Guide to Genetic Programming. — 2008.

[9] *Koza John R.* Genetic Programming: On the Programming of Computers by Means of Natural Selection. — The MIT Press, 1992.

[10] *Ivan Zelinka Zuzana Oplatkova, Nolle Lars.* Analytic programming Symbolic Regression by means of arbitrary Evolutionary Algorithms // *International Journal of Simulation Systems, Science & Technology.* — 2005. — Vol. 6, no. 9. — Pp. 44–56.

[11] *Г.И. Рудой В.В. Стрижов.* Алгоритмы индуктивного порождения суперпозиций для аппроксимации измеряемых данных // *Информатика и её применения.* — 2013. — Vol. 1.

[12] *Г.И. Рудой В.В. Стрижов.* Упрощение суперпозиций элементарных функций при помощи преобразований графов по правилам // Интеллектуализация обработки информации. Доклады 9-й международной конференции. — 2012. — Pp. 140–143.

[13] *Gordon M.* Probabilistic and Genetic Algorithms in Document Retrieval // *ACM.* — 1988. — Vol. 31, no. 10. — Pp. 1208–1218. http://doi.acm.org/10.1145/63039.63044.

[14] Learning to Rank by Optimizing NDCG Measure / Hamed Valizadegan, Rong Jin, Ruofei Zhang, Jianchang Mao // Advances in Neural Information Processing Systems 22 / Ed. by Y. Bengio, D. Schuurmans, J.D. Lafferty et al. — Curran Associates, Inc., 2009. — Pp. 1883–1891. http://papers.nips.cc/paper/3758-learning-to-rank-by-optimizing-ndcg-measure.pdf.

[15] *Gordon Michael D.* User-based document clustering by redescribing subject descriptions with a genetic algorithm // *JASIS.* — 1991. — Vol. 42, no. 5. — Pp. 311–322. http://dx.doi.org/10.1002/(SICI)1097-4571(199106)42:5<311::AID-ASI1>3.0.CO;2-J.

[16] *Raghavan Vijay, Agarwal Brijesh.* Optimal Determination of User-oriented Clusters: An Application for the Reproductive Plan // Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application. — Hillsdale, NJ, USA: L. Erlbaum Associates Inc., 1987. — Pp. 241–246. http://dl.acm.org/citation.cfm?id=42512.42545.

[17] *Yang Jing-Jye, Korfhage Robert, Rasmussen Edie M.* Query Improvement in Information Retrieval Using Genetic Algorithms - A Report on the Experiments of the TREC Project // TREC. — 1992. — Pp. 31–58. http://trec.nist.gov/pubs/trec1/papers/03.txt.

[18] The Use of Genetic Programming to Build Queries for Information Retrieval. / Frederick E. Petry, Bill P. Buckles, Thyagarajan Sadasivan, Donald H. Kraft // International Conference on Evolutionary Computation. — 1994. — Pp. 468–473. http://dblp.uni-trier.de/db/conf/icec/icec1994-1.html.

[19] *Chen Hsinchun.* Machine Learning for Information Retrieval: Neural Networks, Symbolic Learning, and Genetic Algorithms // *Journal of the American Society of Information Science.* — 1995. — Vol. 46, no. 3. — Pp. 194–216. http://ai.bpa.arizona.edu/papers/PS/mlir93.ps.

[20] *Billhardt Holger, Borrajo Daniel, Maojo Victor.* Using genetic algorithms to find suboptimal retrieval expert combinations. // SAC. — ACM, 2002. — Pp. 657–662. http://dblp.uni-trier.de/db/conf/sac/sac2002.html.

[21] *Pathak Praveen, Gordon Michael, Fan Weiguo.* Effective information retrieval using genetic algorithms based matching function adaptation // in Proceedings of the 33rd Hawaii International Conference on System Science (HICSS. — 2000.

[22] *Fan Weiguo, Gordon Michael D., Pathak Praveen.* A Generic Ranking Function Discovery Framework by Genetic Programming for Information Retrieval // *Inf. Process. Manage.* — 2004. — Vol. 40, no. 4. — Pp. 587–602. http://dx.doi.org/10.1016/j.ipm.2003.08.001.

[23] *Fan Weiguo, Gordon Michael D., Pathak Praveen.* Personalization of Search Engine Services for Effective Retrieval and Knowledge Management // Proceedings of the

Twenty First International Conference on Information Systems. — ICIS '00. — Atlanta, GA, USA: Association for Information Systems, 2000. — Pp. 20–34. http://dl.acm.org/citation.cfm?id=359640.359720.

[24] *Bengio Yoshua.* Learning Deep Architectures for AI // *Foundations and Trends in Machine Learning.* — 2009. — Vol. 2, no. 1. — Pp. 110–127.

[25] *Itamar Arel Derek C. Rose Thomas P. Karnowski.* Deep Machine Learning—A New Frontier in Artificial Intelligence Research // *IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE.* — 2010. — November. — Vol. 13-19.

[26] *Yoshua Bengio Aaron Courville Pascal Vincent.* Representation Learning: A Review and New Perspectives // *IEEE Transactions on Pattern Analysis and Machine Intelligence.* — 2013. — Vol. 35, no. 8. — Pp. 1798–1828.

[27] *Li HuiMing.* Deep Learning for Image Denoising // *International Journal of Signal Processing, Image Processing and Pattern Recognition.* — 2014. — Vol. 7, no. 3. — Pp. 171–180.

[28] *Kwok T.-Y. Yeung D.-Y.* Constructive Algorithms for Structure Learning in Feedforward Neural Networks for Regression Problems // *IEEE Transactions on Neural Networks for Regression Problems.* — 1997. — Vol. 8. — Pp. 630–645.

[29] *Martins A. F. T.* The Geometry of Constrained Structured Prediction: Applications to Inference and Learning of Natural Language Syntax. — Carnegie Mellon University, 2012.

[30] *Jaakola T. Sontag D.* Learning Bayesian Network Structure using LP Relaxations // *Proceedings of the 13th International Conference on Artifcial Intelligence and Statistics (AISTATS).* — 2010. — Vol. 9, no. 1. — Pp. 358–365.

[31] *Jaakkola T.* Scaled structured prediction. — http://video.yandex.ru/users/yaevents/view/486/user-tag/научный2012.

[32] *H. Lampert C.* Maximum Margin Multi-Label Structured Prediction // *NIPS.* — 2011. — Vol. 11. — Pp. 289–297.

[33] *Варфоломеева А.А. Стрижов В.В.* Алгоритм разметки библиографических списков методами структурного обучения // *Информационные технологии.* — 2014. — Vol. 7. — Pp. 11–15.