Создание ранжирующих моделей для систем информационного поиска.*

Лепехин М. Н., Кулунчаков А. С., Стрижов В. В.
lepehin.mn@phystech.edu

В данной работе исследуются различные методы построения нелинейных моделей для задач ранжирования. В качестве возможных моделей рассматривается множество суперпозиций простых функций, предлагаемых экспертами. Структура суперпозиции функций определяется с помощью обхода в глубину её синтаксического дерева. Предлагается метод прогнозирования такой структуры. Подход к порождению структуры оптимальной модели проверяется как на синтетических данных, так и на основе данных из текстовой коллекции TREC. На этих данных качество построенных суперпозиций алгебраических функций сравнивается с результатами уже известных моделей.

Ключевые слова: *ранжирующие системы, информационный поиск, временные ряды, структурное обучение, суперпозиция моделей.*

1 Введение

В данной работе решается задача ранжирования текстов по поисковым запросам. Эта задача актуальна для современных поисковых систем. Для того, чтобы пользователь получал на свои запросы ссылки на релевантные им документы, производится ранжирующая процедура. Она для каждой пары *документ-запрос* выдаёт релевантность документа данному запросу.

Несмотря на то, что уже существует большая коллекция алгоритмов для решения этой задачи, часто качество их работы оказывается недостаточно высоким. Например, в работах [1, 2] приведены примеры решений задачи ранжирования, столкнувшиеся с проблемой переобучения.

Существуют алгоритмы построения моделей высокого качества, позволяющие относительно эффективно бороться с переобучением [3]. Они строят ранжирующую модель как суперпозицию элементарных функций. Под элементарными функциями в данном случае подразумеваются показательная, логарифмическая и степенная функции. Лучшие по качеству композиции алгебраических функций из [3] превосходят на коллекциях TREC модели из [1, 2]. Но проблема такого метода построения моделей в том, что он использует полный перебор. Из-за этого такие алгоритмы имеют высокую вычислительную сложность и, следовательно, продолжительное время работы даже при построении относительно простых формул.

Кроме того, существуют решения задачи ранжирования с использованием генетического алгоритма. Генетические алгоритмы применяются для решения широкого класса задач. Например, в [6] они применяются для поиска и классификации изображений. Его основное преимущество - гибкость порождаемых им суперпозиций. В работах [4, 5] это позволило существенно расширить класс рассматриваемых ранжирующих функций по сравнению с работой [3]. Однако после 30-40 итераций генетического алгоритма сложность порождаемых суперпозиций примитивов резко возрастает, не давая при этом существенного повышения качества. Причина такой работы заключается в том, что предложенные методы не решали проблему эволюционной стагнации при прекращении изменения популяции.

*Работа выполнена при финансовой поддержке РФФИ, проект №00-00-00000. Научный руководитель: Стрижов В. В. Задачу поставил: Эксперт Кулунчаков А. С. Консультант: Консультант Кулунчаков А. С.

Как правило, проблема эволюционной стагнации возникает, когда значительная часть построенных функций похожи по структуре и дают результаты высокого качества. В таком случае модели, получаемые при операции кроссовера, мало отличаются по структуре от уже сохранённых. Однако из-за высокого качества сохранённых ранжирующих суперпозиций вероятность того, что операция мутации позволит позволит получить из более качественную модель, мала. Из-за этого порождённая ранжирующая функция скорее всего будет удалена из множества.

В работе [7] предлагается учесть проблемы, возникающие при решении задачи ранжирования текстов при помощи описанных выше алгоритмов. Для этого предлагается изменить реализацию генетического алгоритма так, чтобы он был устойчив к эволюционной стагнации. На каждом шаге алгоритма при помощи структурных метрик индицируется эволюционная стагнация. Если эволюционная стагнация произошла, производится преобразование популяции ранжирующих моделей путём замены наименее качественных из них на случайные. Такой подход позволяет внести разнообразие в популяцию и сделать метод порождения моделей более устойчивым к стагнации. Помимо этого, для ограничения структурной сложности предлагается использовать регуляризаторы, штрафующие за сложность порождаемых функций. Такое видоизменение алгоритма позволяет ограничивать структурную сложность генерируемых суперпозиций и, как следствие, бороться с переобучением .

При исследовании текстов в задачах ранжирования будут использованы основные характеристики слов текста - tf (частоты слов в документе) и idf (числа документов, в которых слово встречается).

В данной работе предлагается развить идею предсказания промежуточной мета-модели. Для этого рассматривается кластеризация документов по значениям tf-idf, посчитанным по корпусу текста. При этом предполагается, что текст разбивается на кластеры таким образом, что внутри каждого кластера документы будут близки друг к другу при ранжировании. Мета-модель предлагается строить как линейную комбинацию моделей, построенных для каждого кластера.

2 Постановка задачи

Имеется коллекция текстовых документов C, состоящая из документов $\{d_i\}_{i=1}^{|C|}$ и множество поисковых запросов $Q = \{q_j\}_{j=1}^{|Q|}$. Для каждого из запросов $q \in Q$ есть непустое множество документов $C_q \subseteq C$, оценённых экспертами. Все оценки r - бинарные:

$r(d, q) \to \{0, 1\}$, где оценка 1 ставится в случае релевантности документа d запросу q.

Для каждого слова при зафиксированной коллекции документов C введём обозначение $\text{count}(w, C)$ - количество документов $d \in C$, в которые входит слово w.

Кроме того, для каждой пары *слово-документ* определим $\text{freq}(w, d)$ - количество вхождений слова w в документ d.

Введём характеристику коллекции size_{avg} - среднее количество слов в документах коллекции. Для каждого документа $d \in C$ определим $\text{size}(d)$ - количество слов в документе d.

Будем рассматривать две характеристики пары *документ-слово*: $(d, w, C) \to (\text{tf}, \text{idf})$, где

$$\text{idf}(w, C) := \frac{\text{count}(w, C)}{|C|}$$

$$\mathrm{tf}(w,d,C) := \mathrm{freq}(w,d) * log\left(1 + \frac{\mathrm{size}_{avg}}{\mathrm{size}(d)}\right)$$

В дальнейшем для удобства будем опускать параметр C, считая его заданным изначально.

Для аппроксимации функции r будем генерировать суперпозицию грамматических элементов. Значение функции f на паре (d,q) определяется как сумма её значений на парах (d,w), где $w \in q$ - слово из запроса.

$$f(d,q) := \sum_{w \in q} f(\mathrm{tf}(w,d), \mathrm{idf}(w))$$

Генерируемая суперпозиция оценивает релевантность документа для каждого запроса. Для оценивания качества модели на выборке, содержащей множество запросов Q, нужно усреднить оценки по все запросам. Для этого будем использовать метрику качества MAP (mean average precision).

$$\mathrm{MAP}(f,C,Q) = \frac{1}{|Q|}\sum_{q \in Q} \mathrm{AvgP}(f,q),$$

где

$$\mathrm{AvgP}(f,q) = \frac{\sum\limits_{k=1}^{|C_q|} \mathrm{Prec}(k) \times r(q,k)}{\sum\limits_{k=1}^{|C_q|} r(q,k)}, \mathrm{Prec}(k) = \frac{\sum\limits_{s=1}^{k} r(q,s)}{k}$$

В качестве математических примитивов $h(x,y)$ будем использовать функции $\sqrt{x}, x + + y, x - y, x * y, x/y, log\ x, e^x$.

Пространство всех суперпозиций указанных выше примитивов обозначим $\mathcal{F}$.

Основная цель данной статьи - нахождение функции f, которая максимизирует следующую метрику качества:

$$f^* = \arg\max_{f \in \mathcal{F}} S(f,C,Q), \quad S(f,C,Q) = \mathrm{MAP}(f,C,Q) - R(f),$$

где R - регуляризатор, штрафующий за структурную сложность порождаемой суперпозиции.

3 Постановка задачи на кластерах документов

Обозначим L множество всех рассматриваемых слов в документах, $|L| = n$.

Определим tf - idf для всей коллекции документов. Рассмотрим отображение $V : C \to \mathbb{R}^n$, в котором каждому документу сопоставляется вектор tf - idf представления всех слов в нем. Для кластеризации документов будем использовать их представление в пространстве $\mathbb{R}^n$. Расстояние между документами считаем при помощи стандартной эвклидовой метрики.

Получаем множество кластеров $D, |D| = m$. Разбиение на кластеры выполним с использованием метода k ближайших соседей. Для каждого кластера при помощи генетического алгоритма построим семейство ранжирующих функций $F^*_{d_i} = \{f^1_i, \ldots, f^n_i\}$. В каждом семействе i выделим наилучшую по описанной выше метрике ранжирующую функцию $f^*_i \in F_{d_i}$.

Алгоритм 1 Создание ранжирующией функции для коллекции документов

Вход. N_{epoch} - количество итераций генетического алгоритма, $\mathbf{C}$.
Выход. f^* - наилучшая модель в итоговой популяции.
Действия.
Сгенерировать начальную случайную популяцию $\mathcal{T}_0$
epoch := 0
пока $epoch < N_{epoch}$
повторять
Выполнить кроссовер для случайной пары суперпозиций из $\mathcal{T}_{epoch}$
Мутировать случайную суперпозицию из $\mathcal{T}_{epoch}$
Отранжировать популяцию $\mathcal{T}_{epoch}$ согласно метрике MAP, учитывая регуляризацию по числу вершин.
Выбрать наилучшие суперпозиции, составить из них популяцию $\mathcal{T}_{epoch+1}$
epoch := epoch + 1

Определим метрику качества на кластерах:

$$f^* = \underset{W \in \mathbb{R}^m}{\arg\max} \left(\big(MAP\big(\sum_{i=1}^{m} W_i f_i^*, C, Q\big)\big) - \sum_{i=1}^{m} R(f_i^*) \right)$$

Значения весов будем подбирать при помощи линейной регрессии.

4 Описание базового эксперимента

Случайная суперпозиция создаётся как случайное дерево малой глубины. В каждом узле этого дерева случайно выбирается одна из базовых операций (см. их список в разделе "Постановка задачи"). После выполнения данной процедуры получаются деревья из 15-30 узлов. При выполнении операции скрещивания на двух объектах популяции используется обмен случайных двух узлов в деревьях суперпозиций. В качестве операции мутации используется замена случайного поддерева суперпозиции на вновь сгенерированое дерево небольшой глубины. Кроме того, используется метод преодоления стагнации из [7]. Как только происходит стагнация.

5 Основные определения

Определение 1. *Скрещивание функций* f_i и f_j, представленных раскрашенными деревьями T_i, T_j - процесс получения новых моделей $\widetilde{f}_i, \widetilde{f}_j$ посредством обмена деревьев T_i, T_j двумя случайными и равновероятными поддеревьями.

Пример.

$$f(x,y) = \sqrt{x} + sin(xy), g(x,y) = e^y + x^2$$

$$\downarrow$$

$$\widetilde{f}(x,y) = \sqrt{x} + x^2, \widetilde{g}(x,y) = e^y + sin(xy)$$

Определение 2. *Мутация функции* f_i, представленной раскрашенным деревом T_i - процесс получения новой модели $\widetilde{f}_i$ посредством замены случайного равновероятного поддерева T_i сгенерированным случайным деревом ограниченной структурной сложности.

Пример.

$$f(x,y) = (x+y)(7x^3 + x^2 - 29) \Rightarrow (x+y)e^x$$

Определение 3. *Популяция* $\mathcal{T}$ - некоторое подмножество пространства допустимых суперпозиций $\mathcal{F}$.

Опишем формально шаги алгоритма.

Используемый метод - итерационный. На i-й итерации происходит порождение популяции $\mathcal{T}_i \subset \mathcal{F}$ путём объединения следующих множеств:

1) популяция предыдущей итерации: $\mathcal{T}_{i-1}$

2) набор моделей, получившихся при скрещивании моделей популяции $\mathcal{T}_{i-1}$

3) набор моделей, получившихся после мутации моделей популяции $\mathcal{T}_{i-1}$

4) множество случайных моделей $\mathcal{R}_i$, которое генерируется в случае, когда обнаруживается стагнации генетического алгоритма.

Для того, чтобы ограничить мощность популяции, порождённой на итерации i, в множестве $\mathcal{T}_i$ остаются только $|\mathcal{T}_{i-1}|$ наилучших моделей согласно критерию качества $\mathcal{MAP}$[7]. Таким образом, мощность популяции не меняется от итерации к итерации.

Эта мощность задаётся на 0-й итерации при генерации множества случайных функций $\mathcal{T}_0 = \mathcal{R}_0$.

Для обнаружения стагнации метода, нам потребуется ввести метрику $\mu : \mathcal{F}^2 \to \mathbb{R}$ для определения того, насколько модели структурно близки друг к другу. Вводимые метрики должны удовлетворять стандартным метрическим аксиомам:

1) Неотрицательность:

$$\mu(f_1, f_2) \geqslant 0 \ \forall f_1, f_2 \in \mathcal{F}$$

$$\text{При этом } \mu(f_1, f_2) = 0 \Leftrightarrow f_1 = f_2.$$

2) Симметричность:

$$\mu(f_1, f_2) = \mu(f_2, f_1) \ \forall f_1, f_2 \in \mathcal{F}.$$

3) Неравенство треугольника:

$$\mu(f_1, f_2) \leqslant \mu(f_1, f_3) + \mu(f_3, f_2) \ \forall f_1, f_2, f_3.$$

Далее будем использовать метрики и формулы из [7].

Определение 4. *Окрестностью функции* $f \in \mathcal{F}$ *радиуса* r назовём множество $U_r(f) = \{f' \in \mathcal{F} : \mu(f, f') < r\}$. Т.е. окрестность функции f - множество моделей из $\mathcal{F}$, структурное расстояние до которых меньше r.

Для отбора суперпозиций необходимо не только проверять структурную похожесть моделей, но и смотреть, насколько близкие значения функции принимают на одинаковых наборах аргументов. Введём η - метрику между результатами ранжирования моделей:

$$\eta(f, f') = \frac{1}{|C|(|C|-1)} \sum_{d_j, d_k \in C} [f(d_j) < f(d_k)][f'(d_j) > f'(d_k)].$$

$\eta(f_i, f_j)$ - количество инверсий, которые необходимо провести для того, чтобы поисковую выдачу f_i преобразовать в поисковую выдачу f_j.

Для обнаружения стагнации запишем требование к моделям:

$$\nu\Big([\mu(f_i,f_j)\leqslant\alpha_1]\Rightarrow[\eta(f_i,f_j)\leqslant\alpha_2]\Big)\geqslant\alpha_3(\alpha_1,\alpha_2),$$

где $\alpha_1,\alpha_2,\alpha_3$ - некоторые постоянные, $\frac{\partial\alpha_3}{\partial\alpha_1}\geqslant 0$, $\mu(A)$ - частота появления события A в серии испытаний.

Определение 5. *Стагнация порождения* - ситуация, когда генетический алгоритм не порождает новые модели из-за малого диаметра популяции.

При возникновении стагнации операция скрещивания конструирует лишь те модели, которые уже есть в популяции, а при мутации получаются суперпозиции более низкого качества, которые будут удалены перед следующей итерацией.

Определение 6. *Сложность модели* f - число элементов грамматики $\mathcal{G}$, из которых она составлена.

Определение 7. *Диаметр* $\mu(\mathcal{T})$ *популяции* $\mathcal{T}$ - среднее расстояние между моделями популяции $\mathcal{T}$.

$$d(\mathcal{T})=\frac{\sum\limits_{j,k=1,\ldots,|\mathcal{T}|}\mu(f_i,f_j)}{\mathrm{AvLen}(\mathcal{T})},\ \mathrm{AvLen}(\mathcal{T})=\frac{1}{\mathcal{T}}\sum_{j=1}^{|\mathcal{T}|}|f_j|$$

6 Используемые метрики

6.1 Схожесть согласно изоморфизму

Определение 8. Два графа G_1 и G_2 называются *изоморфными друг другу*, если существует взаимно-однозначное отображение $\varphi: V(G_1)\to V(G_2)$ такое, что $v,u\in G_1$ смежны $\Leftrightarrow$ смежны $\varphi(v)$ и $\varphi(u)$ графа G_2.

Определение 9. Два дерева T_i,T_j *имеют общий подграф*, если $\exists\widetilde{T}_i$ - поддерево T_i, $\widetilde{T}_j$ - поддерево T_j, что $\widetilde{T}_i$ и $\widetilde{T}_j$ изоморфны друг другу. Данный подграф называется *общим подграфом* деревьев T_i и T_j.

Определение 10. Общий подграф деревьев T_i,T_j, имеющий наибольшее число вершин, называется *наибольшим общим подграфом*. Обозначение: T_{ij}.

Число вершин в дереве T_i будем называть *мощностью дерева* T_i и будем обозначать его $|T_i|$.

Введём расстояние между двумя покрашенными деревьями $\mu_1(T_i,T_j)=|T_i|+|T_j|-$ $-2|T_{ij}|$.

Метрические свойства функции μ_1 доказаны в работе [8].

6.2 Схожесть согласно редакторским расстояниям

Снова будем рассматривать представление функции f покрашенным деревом T_f. При этом дереву T_f сопоставим строку - последовательность меток вершин, строящаяся при обходе дерева в глубину.

На полученных строках зададим метрику μ_2.

Определение 11. *Расстоянием Левенштейна* между строками s_1 и s_2 называется минимальное число операций вставки, удаления и замены одного символа на другой, которые необходимо произвести для преобразования s_1 в s_2.

В работе [7] доказаны метрические свойства μ_2.

Перенесём понятие редакторского расстояния между строками на покрашенные деревья.

Определение 12. *Расстоянием Левенштейна* между деревьями T_i и T_j называется минимальное количество операций добавления, удаления и перекрашивания вершин, необходимых для того, чтобы деревья были изоморфными.

Разобъём множество всех покрашенных деревьев на классы эвивалентности так, чтобы любые два дерева T, T' лежат в одном классе эквивалентности $\Leftrightarrow T$ и T' изоморфны.

Тогда, пользуясь определением 12, введём метрику μ_3 на множестве классов эквивалентности помеченных деревьев.

Все описанные выше метрики используются для определения стагнации. Алгоритм обнаружает стагнацию, если выполняется условие $d(\mathcal{T}) < t$, где пороговое значение t находится эмпирически.

7 Регуляризаторы

Ещё одна проблема, которая может происходить при применении генетического алгоритма - построение моделей сложной структуры, из-за чего происходит переобучение. Для того, чтобы предотвращать переусложнение суперпозиций, можно использовать регуляризацию.

Для построения регуляризатора будем рассматривать следующие параметры дерева:

1) количество вершин в дереве,

2) количество листьев в дереве,

3) глубина дерева.

При ограничении этих параметров структурная сложность порождаемых суперпозиций не будет слишком большой.

В данной работе будет рассмотрен следующий регуляризатор:

$$R(f) = p * \mathrm{MAP}(f)|f|^* log(|f| + 1),$$

где $|f|^*$ - количество переменных в модели.

Его выбор обоснован тем, что его использование в [7] дало наилучшие результаты по сравнению с другими рассматриваемыми регуляризаторами.

8 Разбиение на кластеры

9 Данные эксперимента

Для обучения и тестирования модели используется коллекция текстовых документов TREC [13]. В частности коллекции Trec-5 – Trec-8. Для каждой коллекции представлены набор запросов, ранжирование документов, проведенное экспертами экспертами, и сами документы, на которых решается задача информационного поиска. Основой каждой коллекции является набор документов, предоставляемых NIST, являющимся спонсором конференции TREC. В каждой коллекции представлено в среднем 500 000 документов, 100 запросов и порядка 2000 ответов для каждого запроса. Номер каждой коллекции непосредственно связан с номером конференции, на которой рассматривалась данная коллекция.

Литература

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

[2] *Ponte, Jay M. and Croft, W. Bruce* A Language Modeling Approach to Information Retrieval // In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 275–281. ACM

[3] *P. Goswami, S. Moura, E. Gaussier, M.-R. Amini, F. Maes* Exploring the space of ir functions // ECIR'14, 2014, pp. 372–384.

[4] *Fan, Weiguo and Gordon, Michael D. and Pathak, Praveen* Personalization of Search Engine Services for Effective Retrieval and Knowledge Management // In Proceedings of the twenty first international conference on Information systems (ICIS '00). Association for Information Systems Atlanta, GA, USA, 20-34.

[5] *Fan, Weiguo and Gordon, Michael D. and Pathak, Praveen* A Generic Ranking Function Discovery Framework by Genetic Programming for Information Retrieval // Inf. Process. Manage. 40, 4 (May 2004), 587-602.

[6] *C.-H. Lina, H.-Y. Chenb, Y.-S. Wua* Study of image retrieval and classification based on adaptive features using genetic algorithm feature selection, Expert Systems with Applications38241 (2014) 66116621.

[7] *A.S. Kulunchakov, V.V. Strijov* Study of image retrieval and classification based on adaptive features using genetic algorithm feature selection, Expert Systems with Applications: An International Journal (2017).

[8] *Макаров* Метрические свойства функций расстояний между молекулярными графами, Журнал структурной химии (2007) - Vol.2, 223-229.