УДК 004.8

С.И. Николенко, А.А. Фишков

ОБЗОР МОДЕЛЕЙ ПОВЕДЕНИЯ ПОЛЬЗОВАТЕЛЕЙ ДЛЯ ЗАДАЧИ РАНЖИРОВАНИЯ РЕЗУЛЬТАТОВ ПОИСКА

Николенко С.И., Фишков А.А. **Обзор моделей поведения пользователей для задачи ранжирования результатов поиска.**
Аннотация. Модели поведения пользователей – одно из основных направлений исследований в области улучшения интернет-поиска; это обычно вероятностные модели, обучающиеся из данных о пользовательских действиях (click logs). Мы представляем обзор современных моделей поведения пользователей, а также рассказываем о том, как модели поведения комбинируются с другими признаками в функции ранжирования.
Ключевые слова: модели поведения пользователя, вероятностные графические модели, усиление.

Nikolenko S.I., Fishkov A.A. **A survey of user behaviour models for search results ranking.**
Abstract. User behaviour models represent important features used to enhance web search results; they are usually probabilistic models learned from logs of user actions (click logs). We present a survey of modern user behaviour models and review how behaviour models are combined with other features in the ranking function.
Keywords: user behaviour models, probabilistic graphical models, boosting.

1. Введение. С каждым годом всё более привычным способом доступа к различной информации становится сеть Интернет. Поисковые системы – важнейшая часть современного интернета и ставший уже неотъемлемым признак современного информационного общества [36].

Взаимодействие с такой системой происходит следующим образом: пользователь вводит запрос, после чего поисковая система представляет ему список документов, которые наилучшим образом соответствуют его запросу. Средняя длина современного поискового запроса составляет всего 2,5 слова (по данным исследования [47]), и, конечно, пользователь не сможет просмотреть все документы из базы поисковой системы, т.е. принципиальное значение приобретает задача ранжирования документов по степени их соответствия запросу – *релевантности*.

Для решения задачи ранжирования, а также для оценки качества поиска используются экспертные оценки релевантности документа запросу. На основе информации о содержимом документа,

тексте запроса и других признаках этой пары ранжирующая функция строится с помощью методов машинного обучения [4–6, 9, 13]. Получение экспертных оценок – процесс трудоёмкий и дорогостоящий. Этот подход не лишён недостатков: эксперты могут расходиться во мнениях, да и сами мнения экспертов могут меняться со временем в результате появлением новых страниц или других событий, происходящих в мире или в сети.

Известно, что современные поисковики ведут историю посещений, запросов и переходов для всех пользователей. Интерес представляет использование этой информации для ранжирования и улучшения поиска. Эта информация доступна в больших объёмах, не требует больших затрат на получение и содержит оперативную информацию для огромного числа документов. Впервые эта идея была предложена в работе [20]. Авторами были проведены эксперименты с участием людей и сформулированы основные положения об их поведении при взаимодействии с поисковой системой. Среди последующих работ наиболее сильные результаты были получены в области построения вероятностных моделей поведения пользователя [7, 10, 11, 16, 35].

В этой работе мы предлагаем обзор современных моделей поведения пользователя, а также приводим описание того, как же объединить большое число имеющиеся признаков в единую функцию ранжирования. Работа организована следующим образом: раздел 2 вводит основные принципы информационного поиска и место вероятностных моделей в информационном поиске. В разделе 3 рассматриваются основные гипотезы о поведении пользователя, из которых исходят создатели моделей, и относительно простые вероятностные модели, иллюстрирующие эти гипотезы. Раздел 4 посвящён более сложным вероятностным моделям поведения пользователя, представленным в виде вероятностных графических моделей. В разделе 5 мы описываем, как именно эти и другие модели можно внедрить в реальную поисковую систему, т.е. как из множества признаков (в том числе результатов различных моделей поведения пользователя) можно обучить единую ранжирующую функцию. Раздел 6 завершает работу.

2. Принципы информационного поиска и вероятностные модели.

2.1. Информационный поиск. В наше время под поиском информации обычно подразумевают поиск в интернете, однако

термин «информационный поиск» возник гораздо раньше. Согласно монографии [25], *информационный поиск* – это процесс поиска в большой коллекции (хранящейся, как правило, в памяти компьютеров) некоего неструктурированного материала (обычно – документа), удовлетворяющего информационные потребности.

Сначала информационный поиск касался научных учреждений, и многие университеты и библиотеки стали использовать информационно-поисковые системы – программно-аппаратные комплексы, предоставляющие возможности поиска информации. Вскоре этот подход распространился на медицинские, государственные и банковские структуры. Сейчас, конечно, подобные системы внедрены повсеместно. Для взаимодействия с поисковой системой пользователь делает запрос – формулирует свою информационную потребность на языке, понятном системе. В ответ на запрос система выдаёт пользователю упорядоченный список документов. Для определения соответствия документов запросам в теории информационного поиска вводится следующее понятие: *релевантность* – это соответствие документа информационному запросу. По методу определения обычно различают формальную и содержательную релевантности. Формальная релевантность определяется с помощью некоторого алгоритма, реализованного в информационно-поисковой системе. Содержательная релевантность – это соответствие документа запросу пользователя, определяемое неформальным путём, по семантике документа.

На первый взгляд, цель информационного поиска можно сформулировать следующим образом: найти все релевантные документы. Но при работе с большими коллекциями документов итоговое количество документов, соответствующих запросу, может быть настолько большим, что человек просто не сможет просмотреть их все. Таким образом, одной из важных задач поисковой системы является *ранжирование* документов по степени их соответствия запросу. Функцию, сопоставляющую каждому документу число – вычисленную релевантность документа данному запросу – называют *ранжирующей функцией* (ranking function). Эта функции учитывает различные признаки документа, запроса, а также всей коллекции документов в поисковой системе. Сейчас большое распространение получили ранжирующие функции, полученные с помощью алгоритмов комбинации моделей; мы рассмотрим некоторые из них в разделе 5. Для обучения комбинации моделей требуется

наличие некоторого числа документов с известными истинными значениями релевантности. Их получают от специальных экспертов – асессоров. Методику получения этих оценок можно кратко описать следующим образом: сначала для оцениваемого запроса формируются условия релевантности – предположения о наиболее вероятной интерпретации запроса, о возможной информационной потребности пользователя, сделавшего данный запрос; среди всех возможных интерпретаций следует выбирать ту, которой придерживается подавляющее большинство пользователей.

При взаимодействии с поисковой системой пользователь не всегда может адекватно выразить свою информационную потребность с помощью запроса. Например, по запросу «цезарь» поисковая система может выдать пользователю биографию Гая Юлия Цезаря, а может – рецепт салата «Цезарь». Хотя и тот и другой документы несомненно являются релевантными запросу пользователя, однозначно определить, какой именно из них отвечает информационным потребностям пользователя, система не может.

Пертинентность – это соответствие найденных информационно-поисковой системой документов информационным потребностям пользователя, независимо от того, как полно и как точно эта информационная потребность выражена в тексте информационного запроса. Информацию о пертинентности может предоставить лишь сам пользователь поисковой системы. С точки зрения пользователя, цель информационного поиска – найти все пертинентные и только пертинентные документы; как говорил один из создателей Google Ларри Пейдж, «идеальная поисковая система понимает, что вы имеете в виду, и возвращает в точности то, что вы хотели». В общем случае эта задача неразрешима, так как, как мы только что видели, один и тот же запрос может соответствовать многим разным намерениям пользователя.

Современные поисковые системы ранжируют результаты поиска на основе упомянутой выше формальной (ещё её называют вычисленной) релевантности. Для увеличения степени пертинентности этих результатов для конкретного пользователя используются технологии «персонализированного поиска»: поисковая система хранит информацию о пользователе, его предпочтениях и по каждому запросу формирует выдачи в соответствии с ними. На сегодняшний день функция персонализированного поиска доступна в поисковой системе Google. Для её работы пользователю предлага-

ется самостоятельно «удалять» из выдачи не релевантные данному запросу документы, эти документы в дальнейшем не будут ему показаны по схожим запросам. Эти технологии только начинают развиваться и внедряться, и в настоящей работе мы не будем их рассматривать.

2.2. Вероятностная модель поиска. Как уже упоминалось выше, пользователь неточно формулирует свою информационную потребность в виде запроса. Имея только запрос, система не может точно определить релевантность того или иного документа. Для принятия решений в условиях неопределённости необходим математический аппарат теории вероятностей.

Рассмотрим поиск с ранжированием, в котором пользователь формулирует запрос и в ответ на него получает от поисковой системы упорядоченный список документов. Предположим, что оценки релевантности бинарные: документ может быть либо релевантным данному запросу, либо не релевантным. Таким образом, для каждого документа d и запроса q вводится случайная величина $R(d,q)$ – показатель релевантности; она равна единице, если документ d является релевантным запросу q, и равна нулю в противном случае. Когда это не вызывает недоразумений, будем обозначать показатель релевантности просто R.

В рамках такой модели естественным является ранжирование документов по оценённым вероятностям их релевантности запросу: $p(R(d,q) = 1)$. Такой подход лежит в основе *вероятностного принципа ранжирования*, предложенного Робертсоном в 1977 году [34]. Приведём его основные положения:

(1) релевантность документа запросу не зависит от других документов в коллекции;

(2) пертинентность документа для пользователя может зависеть от уже просмотренных им документов;

(3) *вероятностный принцип ранжирования*: если поисковая система в ответ на каждый запрос пользователя ранжирует документы в порядке убывания их вероятности быть релевантными запросу пользователя, и эта вероятность оценивается наиболее точно на основе доступных данных, то общее качество системы является наилучшим на основе доступных данных.

Для формального доказательства оптимальности этого принципа в конкретной ситуации необходимо зафиксировать критерий качества. Самой простой является модель бинарных потерь: потери на документе d_i составляют $L(d_i) = 1$, если он был выдан системой, будучи не релевантным, или не был выдан, будучи релевантным; в остальных случаях $L(d_i) = 0$. Общие потери являются суммой потерь на всех документах коллекции. Определим цель системы по данному запросу q и числу k так: вернуть k наиболее релевантных документов. Тогда справедлива следующая теорема.

Теорема (Robertson). Вероятностный принцип ранжирования является оптимальным в смысле минимизации ожидаемых потерь в модели бинарных потерь.

Доказательство этой теоремы приведено в [34]. Она требует использования аналитического представления всех распределений для получения оценок, что недостижимо на практике. Тем не менее, вероятностный принцип ранжирования стал основой для разработки современных моделей информационного поиска.

2.3. Критерии оценки качества поиска. Для оценки качества поиска необходимо иметь некоторое тестовое множество, содержащее «достоверную» информацию о том, какой документ является релевантным каким запросам. Обычно тестовое множество строится специальными экспертами и состоит из оценок релевантности для пар (запрос, документ). Оценки могут быть числовыми или категориальными. Так как оценки получают от людей, тестовое множество покрывает лишь малую часть всей базы поисковой системы и его получение является трудоёмким и дорогостоящим.

Классическими параметрами для оценки качества работы поисковой системы являются *точность* и *полнота*:

(1) точность (precision) – количество релевантных запросу документов в выдаче, делённое на общее количество документов в выдаче;

(2) полнота (recall) – количество релевантных запросу документов в выдаче, делённое на общее количество релевантных документов в базе поисковой системы.

Эти параметры поисковой системы недостаточно полно описывают качество её работы, так как не зависят от ранжирования выдачи. Кроме того, трудно оценить изменения качества, если, например, увеличилась точность поиска, а полнота снизилась.

Как уже отмечалось, ранжирование играет важную роль в поисковой системе, поэтому распространение получили меры качества ранжирования результатов. Самой распространённой на сегодняшней день является NDCG [19] – Normalized Discounted Commulative Gain. Так как размер выдачи по каждому запросу может быть различным, ограничиваются первыми k результатами. Формула для одного запроса следующая:

$$\mathrm{DCG}_k = \sum_{i=1}^{k} \frac{2^{r_i} - 1}{\log_2(1+i)},$$
$$\mathrm{NDCG}_k = \frac{\mathrm{DCG}_k}{\mathrm{IDCG}_k},$$

где r_i – оценка релевантности документа на позиции i, а IDCG_k – значение DCG_k при ранжировании по истинным значениям релевантности документов. Таким образом, максимальное значение NDCG составляет 1. Хотя эта мера учитывает положение документов в выдаче, у неё нет естественной интерпретации.

Ранжирование на основе вероятностного принципа можно рассматривать и как задачу бинарной классификации; тогда более естественной мерой качества может стать AUC [12] (Area Under (ROC) Curve, площадь под графиком). Для бинарного классификатора можно построить *кривую соотношения корректного и ложного обнаружения* (ROC curve, Receiver Operating Characteristics curve), отражающую зависимость между вероятностями ошибок первого и второго рода. В частности, если ответом классификатора является вероятность принадлежности «положительному» классу, то площадь под графиком этой кривой будет равна вероятности того, что случайно выбранному «положительному» экземпляру будет присвоено большее значение, чем «отрицательному». Если документы в выдаче упорядочиваются по оценкам вероятности принадлежности релевантному классу, то следующая статистика является несмещённой оценкой AUC [23]:

$$\hat{A} = \frac{S_0 - n_0(n_0+1)/2}{n_0 n_1},$$

где n_0,n_1 – число релевантных и нерелевантных документов в выдаче, $S_0 = \sum p_i$ – сумма номеров позиций релевантных документов в выдаче.

Как мера качества поиска эта величина имеет естественную интерпретацию: оценка вероятности того, что релевантный документ будет расположен выше нерелевантного.

2.4. Вероятностные графические модели. Многие современные вероятностные модели поведения пользователей представляются в виде *вероятностных графических моделей*. Вероятностные графические модели представляют совместное распределение случайных величин, участвующих в вероятностной модели решаемой задачи, в виде графа, рёбра которого (точнее, их отсутствие) кодируют условные независимости между величинами. Можно выделить следующие основные достоинства этого подхода.

1. Визуальное представление структуры вероятностной модели является наглядным, способствует быстрой разработке, изменению и представлению новых моделей.

2. Важные свойства модели, например условная независимость входящих в неё случайных величин, сводятся к свойствам графа, соответствующего данной модели.

3. Сложные, связанные с использованием модели вычисления, такие, как получение оценок параметров, могут быть сформулированы в виде алгоритмов на соответствующем графе.

Классический граф состоит из множества вершин и множества рёбер. В вероятностной графической модели вершины графа обозначают случайные величины, а рёбра – связи между ними. Графические модели разных типов различаются способом построения и видом графа. Модели, основанные на направленных графах, носят название *байесовских сетей доверия* (Bayesian belief networks) [2, 24, 28, 29, 37, 43]. Модели, основанные на ненаправленных графах, часто называют *марковскими случайными полями* (Markov random fields) [2, 18, 24, 30]. Ещё один тип вероятностных графических моделей представлен в аппарате *алгебраических байесовских сетей* [39–46].

Разные типы моделей отличаются друг от друга своими свойствами и задают немного разные семейства распределений, но основной алгоритм точного байесовского вывода – алгоритм передачи сообщений – в любом типе моделей удобно представлять на

фактор-графе. Фактор-граф даёт явное представление разложения сложной функции в произведение более простых: рассмотрим набор переменных $\boldsymbol{X} = \{X_i\}$ и набор функций (факторов) $f, f_1, \ldots, f_m$, таких, что $f(\mathbf{X}) = \frac{1}{Z}\prod_{j=1\ldots m} f_j(Y_j)$, $Y_j \subset \boldsymbol{X}$. Двудольный граф G является фактор-графом для такого разложения функции f, если:

- вершинами первой доли являются переменные X_i;
- вершинами второй доли являются функции f_j;
- рёбра соединяют функции с их аргументами;
- Z – нормировочная константа, определяемая как

$$Z = \int \ldots \int \prod_{j=1\ldots m} f(Y_j).$$

Чаще всего фактор-граф используют для представления плотности совместного распределения случайных величин, однако известны также варианты его использования в теории кодирования [24]. Представление в виде фактор-графа удобно для того, чтобы определить *алгоритм передачи сообщений* (в этом контексте его обычно называют Belief Propagation) [14,22]. Мы не будем подробно рассматривать базовый алгоритм передачи сообщений и отошлём читателя к книгам, посвящённым этой тематике [2, 24].

Для вывода на вероятностных графических моделях поведения пользователя большое значение имеет алгоритм Expectation Propagation (EP) [26,27]. Этот алгоритм в общем случае решает задачу поиска в экспоненциальном семействе распределений такого распределения, которое минимизирует КЛ-дивергенцию (расстояние Кульбака-Лейблера) с фиксированным распределением. Мы подробно рассмотрели алгоритм Expectation Propagation на примере недавно разработанной модели SCM в предыдущей работе [38]; в этом обзоре мы только рассказываем о базовых принципах моделей поведения пользователей и демонстрируем сами модели.

3. Гипотезы о поведении пользователя и простые модели.

3.1. Обозначения и определения. Многие модели поведения пользователей, или, как их ещё называют, клик-модели, используют схожую терминологию и обозначения, и прежде чем описывать конкретные модели, мы её введём.

При взаимодействии с поисковой системой пользователь делает запрос (query) q, получает в результате первые M документов (ссылок), просматривает их, кликает на некоторые из них или ни на какие. Эти действия рассматриваются в течение одного запроса или запросной сессии (query session). Большинство моделей рассматривает их как отдельные и часто независимые сущности, называя их также поисковыми сессиями (search session). В общем случае поисковая сессия состоит из нескольких последовательных запросов одного и того же пользователя, разделённых небольшими интервалами во времени. Оценки релевантности определяются для пар (запрос, документ), поэтому часто в моделях q явно не фигурирует.

Просмотр ссылки и клик трактуется как случайные события. Для конкретной запросной сессии переменная E_i обозначает просмотр описания ссылки на документ, показанный на позиции i, C_i – клик на этой позиции. Тогда вероятность просмотра документа на позиции i равна $p(E_i = 1)$, вероятность соответствующего клика – $p(C_i = 1)$.

Прежде чем строить модели, сформулируем явно те предположения о характере поведения пользователя при работе с поисковой системой, которые эти модели часто делают. Эти предположения вводились одновременно с появлением новых моделей (обычно относительно простых), пользующихся этими предположениями, поэтому далее в этом разделе мы будем рассматривать эти предположения вместе с соответствующими моделями.

3.2. Гипотеза просмотра. Большое распространение получила так называемая *гипотеза просмотра* (examination hypothesis) [33]:

$$p(C_i = 1) = p(E_i = 1)p(C_i = 1 \mid E_i = 1),$$

то есть пользователь делает клик только после просмотра описания. Эта гипотеза подтверждается экспериментами со слежением за взглядом пользователя из той же работы. Также авторы предложили использовать второй сомножитель как оценку релевантности документа запросу:

$$p(C_i = 1 \mid E_i = 1) = r_{\varphi(i)},$$
$$p(C_i = 1 \mid E_i = 0) = 0,$$

где $r_{\varphi(i)}$ считается релевантностью документа на позиции i.

3.3. Каскадная гипотеза: CM и DCM. В работе [8] вводится ещё одно популярное предположение – *каскадная гипотеза* (cascade hypothesis), которая заключается в том, что процесс просмотра описаний всегда начинается с первой позиции и строго линеен. Позиция просматривается, только если все предыдущие позиции были просмотрены. Во введённых обозначениях:

$$p(E_1 = 1) = 1,$$
$$p(E_{i+1} = 1 \mid E_i = 0) = 0,$$

В предлагаемой в [8] модели CM (Cascade Model) дополнительно устанавливается следующее ограничение:

$$p(E_{i+1} = 1 \mid E_i = 1) = 1 - C_i,$$

то есть пользователь продолжает просмотр до первого клика.

Модель DCM (Dependent Click Model) [17] обобщает каскадную гипотезу на несколько кликов по одной и той же выдаче:

$$p(E_{i+1} = 1 \mid E_i = 1, C_i = 1) = \lambda_i,$$
$$p(E_{i+1} = 1 \mid E_i = 1, C_i = 0) = 1,$$

где λ_i, $i = 1, \ldots, M$, – глобальные параметры поведения пользователя.

3.4. Гипотеза просмотра без каскадной гипотезы: модель UBM. Модель UBM (User Browsing Model) [11] основана на гипотезе просмотра, однако не следует каскадной гипотезе; вместо этого она предполагает, что вероятность просмотра E_i зависит от позиции предыдущего клика $r_i = \arg\max_{l<i}\{C_l = 1\}$ и расстояния от неё до текущей позиции $d_i = i - r_i$:

$$p(E_i = 1 \mid C_{r_i} = \beta_{r_i, d_i}.$$

Для первого клика $r_i = 0$. Величины в правой части являются параметрами поведения пользователя, их общее число $M(M+1)/2$. Для получения оценок этих параметров авторы предлагают алгоритм, требующий хранения в памяти наблюдений всех запросных сессий, что затрудняет его практическое применение. В работе [17] предлагается упрощённая модель и эффективный алгоритм получения оценок релевантности, но в сравнении с UBM она даёт худший результат.

Далее в развитии клик-моделей начинается новый этап – получение оценок релевантности выходит на передний план (многие модели пришли из исследований рекламы, и основным показателем для них являлась способность предсказывать клики). Важной новой идеей стало предположение о том, что последующие действия пользователя зависят от релевантности ранее просмотренных документов. В самом деле, пусть система выдала список из 10 документов, и на позиции 3 находится релевантный документ. Если на первых двух позициях находятся нерелевантные документы, то доля кликов на третьей позиции будет высокой, в противном случае можно ожидать, что она понизится вследствие того, что пользователи удовлетворят свою информационную потребность и прекратят поиск, даже не просмотрев этот документ. Эти идеи привели к более сложным моделям поведения пользователя, основанным на вероятностных графических моделях, – к ним мы и переходим в следующем разделе.

4. Модели поведения пользователя, основанные на вероятностных графических моделях.

4.1. Модель CCM (Click Chain Model). Данная модель была предложена в работе [7]. Сохраняя все стандартные обозначения и следуя гипотезе просмотра, модель вводит новый параметр для каждого показанного документа – релевантность $R_i \in [0,1]$, а также глобальные параметры α_1,α_2,α_3. Эти параметры влияют на поведение пользователя на странице поиска, которое изображено в виде блок-схемы на рис. 1.

Пользователь просматривает описание ссылки, далее с вероятностью R_i он кликает на неё (а с вероятностью $1 - R_i$ пропускает), затем с вероятностью α_1 завершает работу, а с вероятностью $1 - \alpha_1$ продолжает просматривать ссылки. Если же был сделан клик, то продолжение просмотра определяется релевантностью просмотренного документа и параметрами α_2,α_3. На рис. 2 модель представлена в виде байесовской сети. Наблюдаемые величины выделены заливкой.

Для завершения формального описания модели приведём все

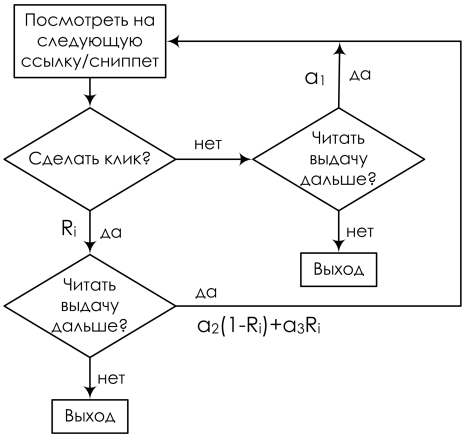

Рис. 1: Модель CCM: блок-схема

условные вероятности:

$$p(C_i = 1|E_i = 0) = 0,$$
$$p(C_i = 1|E_i = 1,R_i) = R_i,$$
$$p(E_{i+1} = 1|E_i = 0) = 0,$$
$$p(E_{i+1} = 1|E_i = 1,C_i = 0) = \alpha_1,$$
$$p(E_{i+1} = 1|E_i = 1,C_i = 1,R_i) = \alpha_2(1 - R_i) + \alpha_3 R_i.$$

Получение оценок релевантности документа производится путём вычисления апостериорных распределений $p(R_i|C_{1...M})$. Так как граф модели имеет циклы, авторы предлагают приближённый алгоритм, в котором удаляют некоторые рёбра, точнее, считают переменные C_i условно независимыми при условии R_i:

$$p(R_i|C_{1...M}) \propto p(R_i)p(C_{1...M}|R_i).$$

После этого удаётся получить замкнутые формулы в виде полиномов, но из-за их высокой степени для получения нормализационной константы и среднего значения используются численные методы. Для оценки параметров авторы используют приближённый метод максимального правдоподобия (так как граф модели содержит циклы).

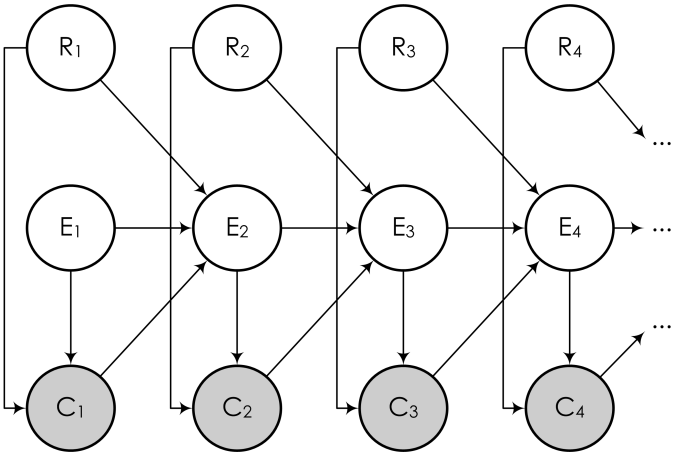

Рис. 2: Модель CCM: байесовская сеть

Хотя эта модель и учитывает влияние релевантности ранее просмотренных документов на поведение на текущей позиции, в ней есть одна неточность: смешивается привлекательность описания и релевантность соответствующего документа (релевантность влияет на клик ещё до того, как пользователь просмотрел саму страницу, а потом эта же величина влияет на последующие просмотры). Следующая модель разделяет эти два понятия.

4.2. Модель DBN (Dynamic Bayesian Network). Прежде всего необходимо отметить, что название этой модели является примером некорректного использования терминов: dynamic Bayesian network (динамическая байесовская сеть) – это специальный вид байесовских сетей, используемый для моделирования последовательностей случайных величин. Так как авторы не дали своей модели названия, в литературе, посвящённой клик-моделям, за ней закрепилось именно это название.

Данная модель также опирается на гипотезу просмотра и каскадную гипотезу. Вдобавок к традиционным обозначениям вводятся дополнительные случайные бинарные величины: A_i показывает, является ли документ привлекательным для пользователя (attractiveness), S_i показывает, удовлетворяет ли документ инфор-

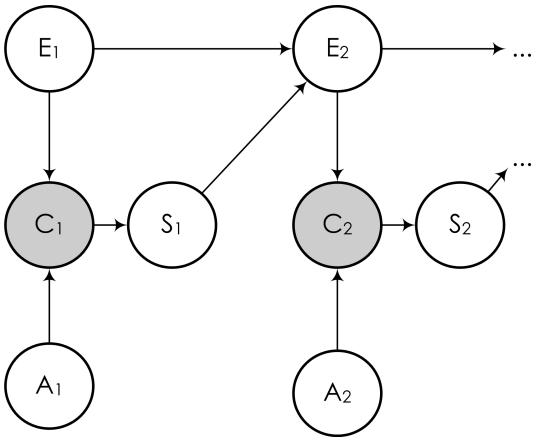

Рис. 3: Модель DBN: байесовская сеть

мационным потребностям пользователя (satisfaction). Последний параметр соответствует пертинентности, о которой говорилось в разделе 2.1. На рис. 3 приведена байесовская сеть для данной модели.

Теперь приведём набор условных вероятностей. Здесь и далее мы будем для ясности изложения использовать логические связки вместо выписывания условных вероятностей, равных нулям и единицам.

$$
\begin{aligned}
E_i = 1, A_i = 1 &\Leftrightarrow C_i = 1, \\
p(A_i = 1) &= a_i, \\
p(S_i = 1 | C_i = 1) &= s_i, \\
C_i = 0 &\Rightarrow S_i = 0, \\
S_i = 1 &\Rightarrow E_{i+1} = 0, \\
p(E_{i+1} = 1 | E_i = 1, S_i = 0) &= \gamma, \\
E_i = 0 &\Rightarrow E_{i+1} = 0
\end{aligned}
$$

В качестве оценки релевантности авторы используют следую-

щую условную вероятность:

$$r_i = p(S_i = 1|E_i = 1) = p(S_i = 1|C_i = 1)p(C_i = 1|E_i = 1) = a_i s_i.$$

Иначе говоря, оценкой релевантности служит вероятность того, что пользователь удовлетворился полученной информацией после просмотра ссылки. Для оценки параметров авторы используют алгоритм EM, а параметр γ считают фиксированным.

4.3. Модель TCM (Task-Centric Click Model). В модели TCM, предлагаемой в работе [35], поведение пользователя рассматривается более широко – на уровне всей поисковой сессии. Поведение пользователя теперь представляется на двух разных уровнях: на уровне сессии и на уровне одного запроса

Пользователь вводит запрос. Далее, если запрос соответствует его потребностям, он просматривает выдачу и может ввести новый запрос или прекратить поиск. Если выдача по запросу не удовлетворила его, он не сделает ни одного клика и переформулирует свой запрос. Вводятся новые бинарные случайные величины: M равна 1, когда запрос соответствует потребностям пользователя, N равна 1, если пользователь продолжает поиск. Так как мы наблюдаем сессию целиком, то по её завершении величина N является наблюдаемой. На рис. 4 поведение пользователя в модели TCM на уровне сессии представлено в виде блок-схемы.

Так как в поисковой сессии несколько запросов, связанных между собой, один и тот же документ может быть показан в выдаче по нескольким запросам (возможно, на разных позициях). Вполне логично, что если документ был показан пользователю не один раз, доля кликов на него после первого показа снижается; это же подтверждается и более подробными исследованиями [35]. Чтобы отразить эту зависимость, в модель на уровне запроса вводится дополнительная бинарная случайная величина F – «свежесть» документа. Теперь для каждого документа в выдаче пользователь просматривает его описание, а если он считает, что выдача не соответствует его потребностям или документ уже был показан ранее, то пропускает его. В противном случае всё определяется релевантностью документа. На рис. 5 это представлено в виде блок-схемы.

Теперь представим TCM в виде байесовской сети (рис. 6). На этом рисунке используется нотация плашек (plate notation): прямоугольниками объединены повторяющиеся фрагменты графа, переменные в них индексированы, диапазон их указан в углу плаш-

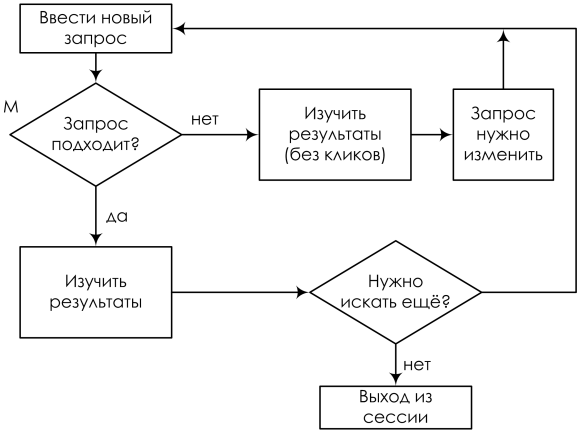

Рис. 4: Модель TCM: блок-схема уровня сессии

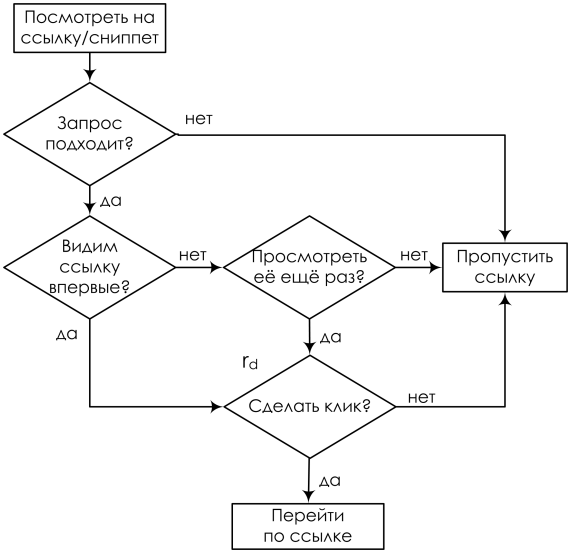

Рис. 5: Модель TCM: блок-схема уровня пользователя

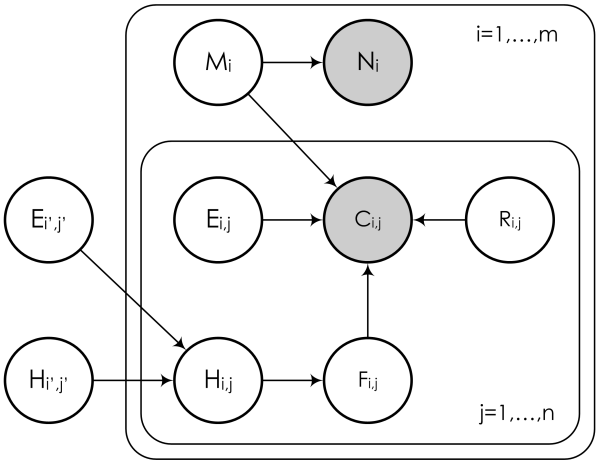

Рис. 6: Модель TCM: байесовская сеть

ки. Индекс i пробегает все запросы в текущей сессии, индекс j – документы, выданные по каждому запросу. Величины M, N и F мы уже описали, E, C и R имеют прежний смысл. Переменная H нужна для связи одинаковых документов, показанных по разным запросам: «свежесть» документа зависит не просто от того, выдавала ли его уже поисковая систем ранее, а от факта просмотра его пользователем. Индексы (i', j') обозначают номер запроса и позиции предыдущего показа документа с позиции (i, j).

Выпишем теперь явно все условные вероятности:

$$
\begin{aligned}
p(M_i = 1) &= \alpha_1, \\
p(N_i = 1 | M_i = 1) &= \alpha_2, \\
M_i = 0 &\Rightarrow N_i = 1, \\
p(F_{i,j} = 1 | H_{i,j} = 1) &= \alpha_3, \\
p(E_{i,j} = 1) &= \beta_{i,j}, \\
p(R_{i,j} = 1) &= r_{i,j}, \\
H_{i,j} = 0 &\Rightarrow F_{i,j} = 1, \\
H_{i,j} = 0 &\Leftrightarrow H_{i',j'} = 0, E_{i',j'} = 0, \\
C_{i,j} = 1 &\Leftrightarrow M_i = 1, E_{i,j} = 1, R_{i,j} = 1, F_{i,j} = 1.
\end{aligned}
$$

Для каждой пары запрос-документ параметр $r_{i,j}$ – релевантность документа. Параметр $\beta_{i,j}$ сходен с «привлекательностью» из модели DBN, однако здесь он влияет не на вероятность клика, а на вероятность просмотра; α_1, α_2 и α_3 являются глобальными параметрами поведения пользователя.

Как видно из описания, вид графа (наличие некоторых рёбер) зависит от конкретной сессии. Наличие неопределённого числа циклов затрудняет получение оценок параметров, и чтобы с этим справиться, авторы [35] предлагают подход, похожий на CCM, – предполагают дополнительные условные независимости между переменными, получая после этого приближённый вариант EM-алгоритма.

4.4. Модель SCM (Session Click Model). Эта модель и вывод на ней подробно описаны в нашей предыдущей работе [38], поэтому здесь для полноты картины ограничимся только кратким описанием. За основу модели SCM была взята модель TCM, однако на уровне запроса в SCM используется модель DBN, а переменные, отражающие свежесть ссылок, берутся из предыдущих выдач как наблюдаемые величины (если документ был показан ранее, это отражено в логах и можно использовать как простую бинарную величину). Таким образом, получается модель, структура графа которой не зависит от конкретной сессии, а лишь от числа запросов в сессии.

На рис. 7 приведена предлагаемая модель SCM (session click model) в виде байесовской сети. Назначение всех участвующих в модели случайных величин (все величины бинарные) описано в

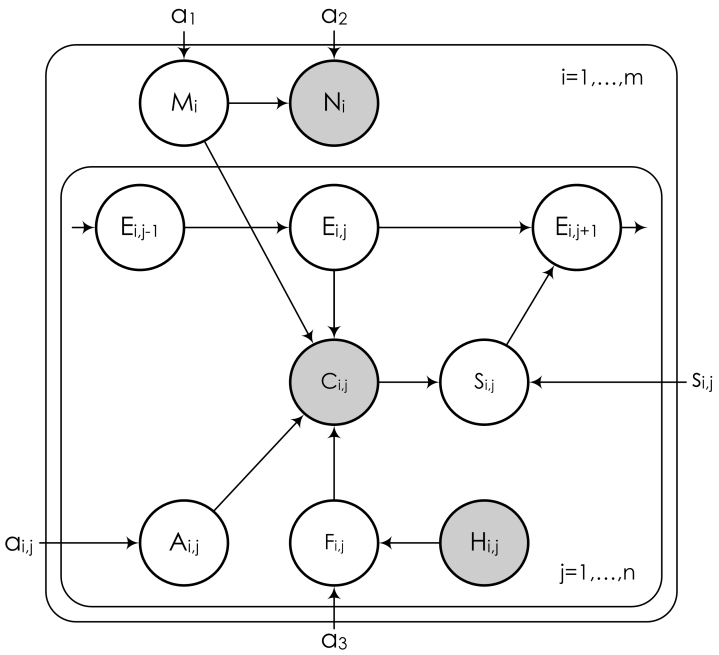

Рис. 7: Модель SCM: байесовская сеть

таблице 1. Ниже мы приводим набор условных вероятностей, завершающий формальное определение модели.

$$
\begin{aligned}
P(M_i = 1) &= \alpha_1, \\
P(N_i = 1 | M_i = 1) &= \alpha_2, \\
P(N_i = 1 | M_i = 0) &= 1, \\
P(F_{i,j} = 1 | H_{i,j} = 1) &= \alpha_3, \\
P(F_{i,j} = 1 | H_{i,j} = 0) &= 1, \\
P(A_{i,j} = 1) &= a_{i,j}, \\
E_{i,j-1} = 0 &\Rightarrow E_{i,j} = 0, \\
S_{i,j-1} = 1 &\Rightarrow E_{i,j} = 0,
\end{aligned}
$$

	Определение
N_i	Продолжает ли пользователь поиск. Наблюдаемая величина
M_i	Соответствует ли выдача по запросу i информационным потребностям пользователя
$E_{i,j}$	Просмотрел ли пользователь описание документа (i, j)
$C_{i,j}$	Клик на документ (i,j). Наблюдаемая величина
$A_{i,j}$	Привлёк ли документ внимание пользователя
$S_{i,j}$	Удовлетворяет ли документ (i,j) информационные потребности пользователя
$H_{i,j}$	Был ли ранее в этой сессии показан документ (i, j)
$F_{i,j}$	Считает ли пользователь документ (i,j) «свежим»

Таблица 1: Случайные величины модели SCM

$$
\begin{aligned}
E_{i,j} = 1 \Rightarrow & E_{i,j-1} = 1, S_{i,j-1} = 0, \\
P(S_{i,j} = 1 | C_{i,j} = 1) = & s_{i,j}, \\
P(S_{i,j} = 1 | C_{i,j} = 0) = & 0, \\
C_{i,j} = 1 \Leftrightarrow & M_i = 1, E_{i,j} = 1, S_{i,j} = 1, F_{i,j} = 1.
\end{aligned}
$$

Параметр $a_{i,j}$ определяет привлекательность документа, а точнее, соответствующего ему элемента на странице выдачи. Этот параметр влияет на вероятность клика. Параметр $s_{i,j}$ можно интерпретировать как пертинетность документа: если запрос соответствовал потребностям пользователя, и пользователь закончил просмотр выдачи кликом по этому документу, значит, документ оказался ему полезным.

В качестве оценки релевантности документа, как и в DBN, будем использовать

$$
\begin{aligned}
r_{i,j} &= P(S_{i,j} = 1 | E_{i,j} = 1, M_i = 1, F_{i,j} = 1) = \\
&= P(S_{i,j} = 1 | C_{i,j} = 1) P(C_{i,j} = 1 | E_{i,j} = 1, M_i = 1, F_{i,j} = 1) = \\
&= a_{i,j} s_{i,j}.
\end{aligned}
$$

Модель имеет три глобальных параметра поведения пользователя: α_1, α_2 и α_3.

5. Совмещение нескольких признаков для ранжирования

5.1. Введение. Ранжирующая функция для поиска Okapi BM25 [21] была одним из первых способов реализации вероятностной модели поиска, однако уже она использовала несколько признаков документа и запроса, такие, как длина документа и запроса в словах, частота вхождения текста запроса в документ и т.д. Эта функция была построена вручную: конкретная формула и числовые значения параметров были получены авторами исходя из семантики использованных признаков.

На сегодняшний день многими авторами предлагается большое количество признаков запроса и документа, которые могут быть полезны для ранжирования: доля входящих и исходящих ссылок документа, длина URL-адреса, возраст документа и многие другие. Любая модель поведения пользователя из тех, что мы рассмотрели выше, тоже на самом деле добавляет ещё один признак для функции ранжирования. Современные поисковые системы используют до нескольких сотен различных признаков для ранжирования, поэтому ручное построение ранжирующей функции просто невозможно.

На сегодняшний день наиболее популярным является метод построения ранжирующей функции на основе машинного обучения. Пары (запрос, документ) представляются в системе векторами своих признаков, а на основе оценок экспертов формируется тренировочное множество, с помощью этой информации ранжирующая функция строится автоматически на основе того или иного метода машинного обучения. В зависимости от типа оценок, которые предоставили эксперты, различают несколько подходов.

1. *Поточечный* (*pointwise*). Каждой паре (запрос-документ) поставлена в соответствие своя оценка. Если оценка категориальная, то получается задача классификации – нужно для пар (запрос-документ) предсказать класс; если оценка численная – задача регрессии.

2. *Попарный* (*pairwise*). Для пар документов, соответствующих одному запросу, эксперты дают «парные предпочтения» – какой документ более релевантен запросу. В рамках этого подхода работают алгоритмы RankNet [4] и RankBoost [13].

3. *Списочный* (*listwise*). Эксперты сами ранжируют несколько документов, соответствующих каждому запросу. Упорядочение, получаемое с помощью построенной ранжирующей функции, должно согласовываться с экспертным. Примером применения этого подхода является LambdaRank [5, 6, 9].

Все крупные поисковые системы (Google, Yandex, Yahoo, Bing и др.) так или иначе используют ранжирующие функции на основе методов машинного обучения, хотя для самых популярных запросов построение ранжирования часто на практике делается вручную (распределение частоты запросов таково, что значительную долю всех запросов составляют буквально считанные тысячи запросов, и на них, конечно, поисковая система должна работать идеально).

В этом разделе мы рассматриваем несколько методов построения ранжирующей функции по множеству признаков.

5.2. Деревья принятия решений. Деревья принятия решений применяются в машинном обучении для построения алгоритмов классификации или регрессии на основе нескольких признаков. Пусть входные данные представляются векторами числовых признаков $(x_1,...,x_n)$. Тогда дерево принятия решений – это бинарное дерево, в листьях которого хранятся значения целевой переменной y (для классификации – номер класса, для регрессии – значение функции). В остальных узлах дерева хранится номер признака k и число z из диапазона значений этого признака. Классификация нового объекта осуществляется следующим рекурсивным алгоритмом обхода: сравнивать значение k-го признака нового объекта с z текущей вершины, если оно меньше – переходить к правому поддереву, иначе – к левому, если попали в лист – вернуть соответствующее значение целевой переменной.

Построение дерева по набору тренировочных примеров $T = \{(x_1,...,x_n,y)\}$ происходит рекурсивно от корня: выбирается номер признака и значение z, а затем множество примеров разбивается на два согласно значениям k-го признака, и так рекурсивно строятся поддеревья, пока все примеры в листе не будут иметь одинаковое значение целевой переменной (этот процесс всегда закончится, хотя бы одноэлементным множеством). Основной вопрос здесь в том, как выбрать очередной атрибут и значение для разбиения (splitting criterion). Распространение получили следующие критерии.

1. Для классификации на k классов $C_1,...,C_k$.

(a) Прирост информации

$$Info(T) = -\sum_{i=1}^{k} \frac{\text{freq}(C_i,T)}{|T|} \log_2 \left(\frac{\text{freq}(C_i,T)}{|T|} \right),$$

где $\text{freq}(C_i,T)$ – число тренировочных примеров из класса C_i. Эта величина в терминологии теории информации называется энтропией множества T. Её можно интерпретировать как среднее количество информации, необходимое для однозначного определения класса одного примера (использование двоичного алгоритма даёт оценку в битах). Оптимальным разбиением будет такое, которое максимизирует следующее выражение:

$$\text{InfoGain}(T,x_k,z) = \text{Info}(T) - \frac{|T_L|}{|T|}\text{Info}(T_L) - \frac{|T_R|}{|T|}\text{Info}(T_R),$$

где T_L,T_R – множества примеров, оказавшихся при таком разбиении в левом и правом поддеревьях соответственно. Возможные значения z для x_k можно искать следующим образом: отсортировать множество T по значениям x_k и брать середины промежуточных интервалов (фактически медиану).

(b) Индекс Джини

$$\text{Gini}(T) = 1 - \sum_{i=1}^{k} \left(\frac{freq(C_i,T)}{|T|} \right)^2;$$

в предыдущих обозначениях оптимальное разбиение теперь максимизирует следующее выражение:

$$\text{GiniGain}(T,x_k,z) = \text{Gini}(T) - \frac{|T_L|}{|T|}\text{Gini}(T_L) - \frac{|T_R|}{|T|}\text{Gini}(T_R).$$

Выбор z – как в предыдущем случае.

2. Для задачи регрессии обычно используют среднеквадратичное отклонение в качестве критерия разбиения. Разбиение максимизирует следующее выражение:

$$S(T,x_k,z) = se(T) - se(T_L) + se(T_R),$$

где $se(T) = \frac{1}{|T|}\sum_{i \in T}(y_i - m(T))^2$ и $m(T) = \frac{1}{|T|}\sum_{i \in T} y_i$. Выбор z осуществляется аналогично предыдущим пунктам. Рекурсивный алгоритм построения дерева в этом случае можно останавливать, когда $se(T)$ станет меньше определенного порога. Тогда значением целевой переменной в листе будет среднее по оставшимся там примерам.

В конкретных алгоритмах классификации и регрессии на основе деревьев принятия решений помимо критериев разбиения используются различные модификации самой рекурсивной процедуры построения. Так, алгоритм классификации ID3 [31] работает только с категориальными признаками, и поэтому разбиения дерева не бинарные, а по всем значениям соответствующего атрибута. Алгоритм C4.5 [32] является его развитием и позволяет использовать дискретные и непрерывные числовые атрибуты, а также пропуски в данных (когда для некоторых примеров известны значения не всех признаков). Алгоритм CART [3], применимый как для классификации, так и для регрессии, использует технику отсечения (pruning) поддеревьев для уменьшения размера получившегося дерева.

Алгоритмы на основе деревьев принятия решений обладают рядом достоинств:

(1) не требуют подготовки входных данных;

(2) полученное дерево имеет ясную и наглядную интерпретацию;

(3) позволяют использовать как категориальные, так и числовые признаки;

(4) допускают эффективный алгоритм применения полученной модели (спуск по дереву).

К их недостаткам можно отнести следующее:

1. задача построения оптимального дерева принятия решений является NP-полной [38];

2. у изолированного полного дерева очень низкая обобщающая способность, необходимо использовать приёмы на основе усечений [37]; кроме того, решение неустойчиво: при добавлении небольшого числа новых тренировочных примеров структура получаемого дерева может кардинально измениться;

3. не все свойства и взаимосвязи атрибутов могут быть выражены в структуре дерева (например, чётность значений признаков и связи отношений атрибутов типа XOR).

Ещё одной важной особенностью деревьев принятия решений является возможность оценить *предсказательную способность* (*predictor importance*) признаков: какие признаки наиболее «важны» для принятия верного решения, а какие и вовсе не несут полезной информации. Один из возможных способов определить предсказательную способность (на примере деревьев регрессии) таков:

$$\mathrm{imp}(x_k) = \frac{1}{|T|} \sum_{v \in V_{x_k}} S(T_v, x_k, z),$$

где V_{x_k} – множество вершин, разбиение в которых происходит по признаку x_k, T_v – поддерево с корнем в v. Выражения под знаком суммы уже были подсчитаны при построении дерева, поэтому дополнительных вычислений не требуется. Видно, что если по некоторому признаку разбиений не производилось, то его предсказательная способность равна 0.

5.3. Усиление (boosting). Деревья принятия решений легки в построении, но обладают своими недостатками. Возникает идея использовать несколько деревьев так, чтобы их совместная эффективность увеличилась. Алгоритмы машинного обучения, направленные на улучшение работы «простых» алгоритмов классификации и регрессии, получили общее название *boosting* (*усиление*). Рассмотрим этот метод на примере регрессии, следуя [15].

Рассмотрим задачу минимизации функции нескольких переменных без ограничений: $\min_{x \in R^d} F(x)$. Для её решения можно применить метод градиентного спуска, начиная с некоторого x_0: $x_{k+1} = x_k - \alpha_k \nabla F(x_k)$.

Этот алгоритм не всегда находит глобальный минимум функции, и его можно считать «жадным»: на каждой итерации делается локально оптимальный шаг в направлении, противоположном градиенту.

Теперь вернёмся к задаче регрессии. По набору тренировочных примеров $T = \{(x_i, y_i)\}_{i=1}^N$ мы хотим построить такую функцию h, что $h(x_i) \approx y_i$, $i = 1...N$ (здесь мы считаем x_i вектором числовых признаков). Используя подход *минимизации эмпирического риска* [18], будем искать функцию h в некотором классе H функций,

минимизируя функцию потерь (loss function); в качестве функции потерь возьмём для примера сумму квадратов отклонений:

$$\min_{h \in H} L(h) = \min_{h \in H} \frac{1}{2} \sum_{i=1}^{N} (y_i - h(x_i)).$$

Теперь можно было бы применить метод градиентного спуска в пространстве функций H:

$$h_{k+1}(x) = h_k(x) - \alpha_k \nabla L(h_k(x)).$$

Однако мы не можем вычислить ∇L во всех точках x (в общем случае это должна быть функция от x), а лишь в конечном числе точек:

$$\nabla L(h_k(x_i)) = -(y_i - h_k(x_i)),\ i = 1...N.$$

Основная идея *усиления* в том, чтобы на каждой итерации находить аппроксимацию градиента функции потерь по имеющимся значениям в точках тренировочного набора. Рассмотренные в предыдущем разделе деревья принятия решений можно использовать как класс аппроксимирующих функций. Приведённый ниже алгоритм из [15] использует деревья решений для регрессии для аппроксимации градиента квадратичной ошибки. Таким образом, получаемая функция h будет являться линейной комбинацией деревьев регрессии.

Алгоритм Gradient Boost
Вход: $T = \{(x_i, y_i)\}_{i=1}^{N}$
Выход: такая h, что $h(x_i) \approx y_i$, $i = 1...N$
Алгоритм

1. $h_0(x) = 1/N \sum_{i=1}^{N} y_i$.
2. Повторять M раз:
   (a) $r_{ik} = \nabla L(h_k(x_i))$;
   (b) построить дерево регрессии t_k по тренировочному множеству $\{(x_i, r_{ik})\}_{i=1}^{N}$;
   (c) $h_{k+1}(x) = h_k(x) - \eta(t_k(x))$.
3. $h(x) = h_M(x)$.

Выбор параметров η и M осуществляется с помощью стандартной в машинном обучении техники: используется отдельное *валидационное* множество примеров, на котором оценивается поведение получаемой функции при изменении параметров (например, величина ошибки).

Усиление деревьев решений стало стандартом де-факто для решения задачи ранжирования, и многие современные алгоритмы на нём основаны (в том числе упомянутые в начале этого раздела).

5.4. Построение ранжирующей функции. Используя изложенные в предыдущих двух разделах методы, можно было бы строить ранжирующую функцию для документов по признакам запроса и документа, считая целевой переменной оценку эксперта. Однако такой подход даёт слабые результаты, преимущественно потому, что меры качества ранжирования считаются и усредняются по запросам, а не по всем возможным парам запроса и документа.

Основным направлением в области применения машинного обучения к задаче ранжирования является использование целевых мер качества (таких, как NDCG) в ходе минимизации эмпирического риска. Одним из способов реализации этого приёма является добавление специального признака «идентификатор запроса» к вектору признаков пары (запрос, документ). Примером может служить алгоритм NDCGBoost [40].

Приведём пример построения ранжирующей функции, основанный на [41]. Пусть дано тренировочное множество

$$T = \{(qid_i, x_i, r_i)\}_{i=1}^{N},$$

где qid_i – идентификатор запроса, x_i –вектор признаков пары (запрос, документ), r_i – экспертная оценка релевантности. Перейдём от оценок релевантности отдельных документов к так называемым «парным предпочтениям». Для тренировочных примеров с одинаковыми qid возьмём все пары документов и посчитаем разность оценок релевантности для них. Получим новое множество тренировочных примеров:

$$T'_k = \{(qid_k, x_i, x_j, r_i - r_j)\}_{i,j \in D(qid_k)}, \quad T' = \bigcup_k T'_k.$$

Теперь можно рассматривать задачу регрессии на парах документов – целевой переменной будет разность оценок их релевантно-

сти: если она больше 0, то первый документ должен располагаться выше в ранжированном списке, иначе – ниже. Применяя к этой задаче алгоритм Gradient Boost, мы получим некоторую функцию $h(qid,x,y)$. Теперь нам нужно вернуться к ранжированию отдельных документов. Это можно сделать следующим образом: $g(qid,x) = \sum_i h(qid,x,y_i)$, где суммирование ведётся по всем парам документов для запроса qid, в которых первым элементом пары выступает документ x.

Чем же был обусловлен выбор парных предпочтений и почему следует ожидать, что этот подход даст лучшее качество ранжирования, чем построение ранжирующей функции по исходному тренировочному множеству? Дело в том, что если рассматривать AUC в качестве меры качества, можно показать, что эта величина будет равна доле инверсий в ранжированном списке (число пар, для которых истинное значение релевантности документа, ранжированного выше, меньше, чем ранжированного ниже). Таким образом, если оценки экспертов являются бинарными, можно считать, что предлагаемый подход минимизирует AUC как эмпирический риск.

6. Заключение. В этой работе мы представили обзор вероятностных моделей поведения пользователя системы интернет-поиска. Эти модели – ещё одно подтверждение «интуитивности интерфейса» вероятностных графических моделей, ведь разработчики моделей поведения пользователя обычно начинают с простых и логичных предположений об этом поведении, затем формализуют их в виде вероятностной графической модели, а потом остаётся только подобрать алгоритм приближённого вывода (или воспользоваться алгоритмом передачи сообщений, если модель не содержит циклов). Можно ожидать появление и новых моделей поведения пользователей – этому способствует то, что задача улучшения качества ранжирования представляет огромный практический интерес для крупных игроков на рынке поисковых систем. Проводятся даже специальные конкурсы, посвящённые ранжированию [48].

Наш обзор был посвящён моделям, которые используют кликлоги поисковой системы; однако ими не исчерпывается всё многообразие возможных подходов. Так, например, в недавней работе [1] предлагается новый подход, позволяющий провести очень хорошо сфокусированные эксперименты, в которых оценивается, насколько успешен был пользователь в своём поиске, а затем использовать

результаты этих экспериментов для улучшения качества ранжирования. Мы полагаем, что в этой области исследований можно ожидать новых прорывов в самое ближайшее время.

Литература

1. *Ageev M., Guo Q., Lagun D., Agichtein E.* Find it if you can: a game for modeling different types of web search success using interaction data // Proceedings of the 34th Annual ACM SIGIR Conference. 2011. P. 345–354.

2. *Bishop C. M.* Pattern Recognition and Machine Learning. Springer, 2006.

3. *Breiman L., Friedman J. H., Olshen R. A., Stone C. T.* Classification and Regression Trees. New York: Chapman & Hall, 1984.

4. *Burges C. J., Shaked T., Renshaw E., Lazier A., Deeds M., Hamilton N., Hullender G.* Learning to rank using gradient descent // Proceedings of the 22nd International Conference on Machine Learning. 2005. P. 89–96.

5. *Burges C. J. C.* From RankNet to LambdaRank to LambdaMART : An Overview: Tech. rep.: Microsoft Research, 2010.

6. *Burges C. J. C., Svore K. M., Bennett P. N., Pastusiak A., Wu Q.* Learning to Rank Using an Ensemble of Lambda-Gradient Models // Journal of Machine Learning Research. 2011. Vol. 14. P. 25–35.

7. *Chapelle O., Zhang Y.* A dynamic Bayesian network click model for web search ranking // Proceedings of the 18th International Conference on World Wide Web. 2009. P. 1–10.

8. *Craswell N., Zoeter O., Taylor M., Ramsey B.* An experimental comparison of click position-bias models // Proceedings of the 1st ACM International Conference on Web Search and Data Mining. 2008. P. 87–94.

9. *Donmez P., Svore K. M., Burges C. J. C.* On the local optimality of LambdaRank // Proceedings of the 32nd Annual ACM SIGIR Conference. ACM, 2009. P. 460–467.

10. *Dupret G., Liao C.* A model to estimate intrinsic document relevance from the clickthrough logs of a web search engine // Proceedings of the 3rd ACM International Conference on Web Search and Data Mining. 2010. P. 181–190.

11. *Dupret G., Piwowarski B.* A user browsing model to predict search engine click data from past observations // Proceedings of the 31st Annual ACM SIGIR Conference. 2008. P. 331–338.

12. *Fawcett T.* An introduction to ROC analysis // Pattern Recognition Letters. 2006. Vol. 27, N. 8. P. 861–874.

13. *Freund Y., Iyer R., Schapire R. E., Singer Y.* An efficient boosting algorithm for combining preferences // Journal of Machine Learning Research. 2005. Vol. 4. P. 933–969.

14. *Frey B. J.* Extending Factor Graphs so as to Unify Directed and Undirected Graphical Models // Proceedings of the 19th Conference on Uncertainty in Artificial Intelligence. Acapulco, Mexico: Morgan Kaufmann Publishers Inc., 2003. P. 257–264.

15. *Friedman J.* Greedy function approximation: a gradient boosting machine // Annals of Statistics. 2001. Vol. 29. P. 1180.

16. *Guo F., Liu C., Kannan A., Minka T., Taylor M., Wan Y., Faloutsos C.* Click chain model in web search // Proceedings of the 18th International Conference on World Wide Web. 2009. P. 11–20.

17. *Guo F., Liu C., Wang Y.-M.* Efficient multiple-click models in web search // Proceedings of the 2nd ACM International Conference on Web Search and Data Mining. 2009. P. 124–131.

18. *Hastie T., Tibshirani R., Friedman J.* Elements of Statistical Learning. Springer, New York, 2008.

19. *Jarvelin K., Kekalainen J.* Cumulated gain-based evaluation of IR techniques // ACM Transactions on Information Systems. 2002. Vol. 20, N. 4. P. 422–446.

20. *Joachims T., Granka L. A., Pang B., Hembrooke H., Gay G.* Accurately Interpreting Clickthrough Data as Implicit Feedback // Proceedings of the 28th Annual ACM SIGIR Conference. 2005. P. 154–161.

21. *Jones K. S., Walker S., Robertson S. E.* A Probabilistic Model of Information Retrieval: Development and Comparative Experiments (parts 1 and 2) // Information Processing and Management. 2000. Vol. 36, N. 6. P. 779–840.

22. *Kschischang F. R., Frey B. J., Loeliger H.-A.* Factor Graphs and the Sum-Product Algorithm // IEEE Transactions on Information Theory. 2001. Vol. 47, N. 2. P. 498–519.

23. *Ling C. X., Huang J., Zhang H.* AUC: a Statistically Consistent and more Discriminating Measure than Accuracy // Proceedings of the International Joint Conference on Artificial Intelligence 2003. 2003. P. 519–526.

24. *MacKay D. J.* Information Theory, Inference and Learning Algorithms. Cambridge University Press, 2003.

25. *Manning C. D., Raghavan P., Schütze H.* Introduction to Information Retrieval. Cambridge University Press, 2008.

26. *Minka T.* Expectation Propagation for approximate Bayesian inference // Proceedings of the 17^{th} Conference on Uncertainty in Artificial Intelligence / Ed. by J. S. Breese, D. Koller. San Francisco, CA: Morgan Kaufmann Publishers Inc., 2001. P. 362–369.

27. *Minka T.* A family of algorithms for approximate Bayesian inference: Ph.D. thesis / Massachusetts Institute of Technology. 2001.

28. *Pearl J.* Probabilistic reasoning using graphs // Uncertainty in Knowledge-Based Systems / Ed. by B. Bouchon, R. R. Yager. Springer-Verlag, 1987. P. 201–202.

29. *Pearl J.* Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. NY etc.: Morgan Kaufmann, 1994.

30. *Prince S.* Computer Vision: Models, Learning, and Inference. Cambridge University Press, 2012.

31. *Quinlan J. R.* Induction of Decision Trees // Machine Learning. 1986. Vol. 1, N. 1. P. 81–106.

32. *Quinlan J. R.* C4.5: Programs for Machine Learning. San Francisco: Morgan Kaufmann, 1993.

33. *Richardson M., Dominowska E., Ragno R.* Predicting clicks: estimating the click-through rate for new ads // Proceedings of the 16th International Conference on World Wide Web. 2009. P. 521–530.

34. *Robertson S. E.* Probability ranking principle in IR // Journal of Documentation. 1977. Vol. 33, N. 4. P. 294–304.

35. *Zhang Y., Chen W., Wang D., Yang Q.* User-click modeling for understanding and predicting search-behavior // Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, New York, NY, USA, 2011. P. 1388–1396.

36. *Заболотский В. П., Юсупов Р. М.* Основные проблемы устойчивости перехода к информационному обществу // Труды СПИИРАН. 2002. Т. 1, № 1. С. 13–26.

37. *Николенко С. И., Тулупьев А. Л.* Самообучающиеся системы. М.: МЦНМО, 2009. 288 с.

38. *Николенко С. И., Фишков А. А.* SCM: новая вероятностная модель поведения пользователей интернет-поиска // Труды СПИИРАН. 2012.

39. *Сироткин А. В.* Байесовские сети доверия: дерево сочленений и его вероятностная семантика // Труды СПИИРАН. 2006. Т. 1, № 3. С. 228–239.

40. *Сироткин А. В.* Вычислительная сожность алгоритмов локального апостериорного вывода в алгебраических байесовских сетях // Труды СПИИРАН. 2011. С. 188–214.

41. *Сироткин А. В., Тулупьев А. Л.* Локальный априорный вывод в алгебраических байесовских сетях: комплекс основных алгоритмов // Труды СПИИРАН. 2007. № 5. С. 100–111.

42. *Тулупьев А. Л., Николенко С. И., Никитин Д. А., Сироткин А. В.* Синтез апостериорных оценок истинности суждений в интегрированных базах знаний: детерминированный вариант // Известия высших учебных заведений: Приборостроение. 2006. № 11. С. 35–39.

43. *Тулупьев А. Л., Николенко С. И., Сироткин А. В.* Байесовские сети: логико-вероятностный подход. СПб.: Наука, 2006. 608 с.

44. *Тулупьев А. Л., Николенко С. И., Сироткин А. В.* Синтез апостериорных оценок при поступлении свидетельств с неопределенностью в интегрированную систему знаний о неточных вероятностях // Известия высших учебных заведений: Приборостроение. 2006. № 11. С. 39–44.

45. *Тулупьев А. Л., Сироткин А. В., Николенко С. И.* Синтез согласованных оценок истинности утверждений в интеллектуальных информационных системах // Известия высших учебных заведений: Приборостроение. 2006. № 7. С. 20–26.

46. *Тулупьев А. Л., Сироткин А. В., Николенко С. И.* Байесовские сети доверия: логико-вероятностный вывод в ациклических направленных графах. СПб.: Изд-во С.-Петербургского ун-та, 2009. 400 с.

47. *Яндекс.* Поиск в интернете: что и как ищут пользователи. Информационный бюллетень «Яндекс», `http://download.yandex.ru/company/yandex_search_autumn_2008_ru.pdf`. 2008.

48. *Яндекс.* Конкурс «Интернет–математика». `http://imat-relpred.yandex.ru/`. 2011.

Николенко Сергей Игоревич — к.ф.-м.н.; научный сотрудник лаборатории математической логики ПОМИ РАН, доцент кафедры математических и информационных технологий, старший научный сотрудник проблемной лаборатории вычислительной биологии СПбАУ НОЦНТ РАН; snikolenko@gmail.com; СПбАУ НОЦНТ РАН, ул. Хлопина, д. 8, корп. 3, г. Санкт-Петербург, 194021, РФ; р.т. +7(812)297-2145, факс +7(812)448-6998.

Sergey I. Nikolenko — Ph.D.; Researcher at Steklov Mathematical Institute, Adjunct Prof. at the Chair of Mathematical and Information Technologies, Senior Researcher at the Laboratory of Algorithmic Biology at St. Petersburg Academic University; snikolenko@gmail.com; St. Petersburg Academic University, ul. Khlopina, d. 8, korp. 3, St. Petersburg, 194021, Russia; office phone +7(812)297-2145, fax +7(812)448-6998.

Фишков Александр Александрович — студент Санкт-Петербургского Государственного Политехнического Университета; jetsnguns@gmail.com; СПбГПУ, Политехническая, 29, 195251, Санкт-Петербург, РФ; р.т. +7(812)297-2095, факс +7(812)552-6080. Научный руководитель – С.И. Николенко.

Alexander Fishkov — M.Sc. student at St. Petersburg Polytechnical University; jetsnguns@gmail.com; St. Petersburg Polytechnical University, ul. Polytechnicheskaya, d. 29, St. Petersburg, 195251, Russia; office phone +7(812)297-2095, fax +7(812)552-6080. Advisor – S.I. Nikolenko.

Поддержка исследований. Работа была поддержана грантами РФФИ 11-01-00760-а и 11-01-12135-офи-м-2011, грантом Президента РФ для молодых кандидатов наук МК-6628.2012.1, грантом Президента РФ для ведущих научных школ НШ-3229.2012.1 и Федеральной целевой программой «Научные и научно-педагогические кадры инновационной России» за 2009–2013 гг.

Рекомендовано ТиМПИ СПИИРАН, зав. лаб. Тулупьев А.Л., д.ф.-м.н.

Статья поступила в редакцию 22.04.2012.

РЕФЕРАТ

Николенко С.И., Фишков А.А. **Обзор моделей поведения пользователей для задачи ранжирования результатов поиска.**

Модели поведения пользователей – одно из основных направлений исследований в области улучшения интернет-поиска; это обычно вероятностные модели, обучающиеся из данных о пользовательских действиях (click logs). Мы представляем обзор современных моделей поведения пользователей, а также рассказываем о том, как модели поведения комбинируются с другими признаками в функции ранжирования.

Сначала в работе вводятся основные принципы информационного поиска и определяется место вероятностных моделей в информационном поиске. Мы напоминаем такие основные понятия информационного поиска как релевантность и пертинентность, а также рассматриваем наиболее часто применяющиеся критерии оценки качества поиска: NDCG (normalized discounted cumulative gain) и AUC (area under [ROC] curve).

Затем рассматриваются основные гипотезы о поведении пользователя, из которых исходят создатели моделей: гипотеза просмотра и каскадная гипотеза; мы иллюстрируем эти гипотезы при помощи относительно простых вероятностных моделей CM (cascade model), DCM (dependent click model) и UBM (user browsing model).

Основное содержание работы посвящено более сложным вероятностным моделям поведения пользователя, представленным в виде вероятностных графических моделей. Мы рассматриваем модели CCM (click chain model), DBN (dynamic Bayesian network), TCM (task-centric click model) и недавно предложенную модель SCM (session click model).

Наконец, в последней части обзора мы описываем, как именно эти и другие модели можно внедрить в реальную поисковую систему, т.е. как из множества признаков (в том числе результатов различных моделей поведения пользователя) можно обучить единую ранжирующую функцию; рассматриваются алгоритмы усиления (boosting), в том числе градиентное усиление, предназначенное для задач регрессии и применённое к ранжированию в алгоритме LambdaRank.

SUMMARY

Nikolenko S.I., Fishkov A.A. **A survey of user behaviour models for search results ranking.**

User behaviour models represent one of the most important issues in research directed at improving web search results; they are usually probabilistic models learned from logs of user actions (click logs). We present a survey of modern user behaviour models and review how behaviour models are combined with other features in the ranking function.

At first, we introduce basic principles of information retrieval and how probabilistic models enter this field. We remind such basic notions of information retrieval as relevance and pertinence and also consider most popular metrics for evaluating search quality, namely NDCG (normalized discounted cumulative gain) and AUC (area under [ROC] curve).

Then we consider the basic hypotheses concerning user behaviour that user behaviour models rely upon, namely the examination hypothesis and the cascade hypothesis. We illustrate these hypotheses with relatively simple probabilistic models CM (cascade model), DCM (dependent click model), and UBM (user browsing model).

The main part of this work is devoted to more complex probabilistic user behaviour models usually represented as probabilistic graphical models. We consider the following models: CCM (click chain model), DBN (dynamic Bayesian network), TCM (task-centric click model), and the recently proposed SCM (session click model).

Finally, in the last part of this survey we study how these and other models can be implemented in a real-life search engine, that is, how a set of features (including the predictions of various user behaviour models) can be incorporated to learn a single ranking function. We consider boosting algorithms, including gradient boosting intended for regression problems and applied to ranking in the LambdaRank algorithm.