Московский государственный университет имени М.В. Ломоносова
Факультет вычислительной математики и кибернетики
Кафедра математических методов прогнозирования

Викулин Всеволод Александрович

# Ранжирование с помощью поведенческой информации

МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ

**Научный руководитель:**
д.ф.-м.н., профессор
А. Г. Дьяконов

Москва, 2018

# Содержание

**Аннотация**

Поведенческие факторы играют ключевую роль при решении задачи ранжирования. Они содержат в себе знания о реакции пользователей, которые взаимодействовали с поисковой выдачей. К сожалению, значение поведенческих факторов известно только для небольшой части поисковых запросов и документов. В данной работе предложен алгоритм оценки поведенческих признаков для запросов, которые редко задаются поисковой машине или вообще никогда не задавались. Был предложен механизм сглаживания поведенческих факторов через схожесть запросов. Был продемонстрирован способ составления обучающего и тестового множества для оценки схожести запросов. Предложена модель для оценки схожести запросов, проведены эксперименты по сравнению модели с базовыми подходами информационного поиска.

# 1 Введение

В современном мире поисковые системы все сильнее и сильнее входят в нашу жизнь. Они помогают быстро и качественно находить нужную информацию миллионам людей ежедневно. Ключевой задачей, которую должна решать любая поисковая система, является задача ранжирования.

Задача ранжирования заключается в сортировке найденных по данному пользовательскому запросу документов в соответствии с их релевантностью. Релевантность документа по запросу является мерой того, насколько информационная потребность пользователя будет удовлетворена, если ему поисковая машина покажет этот документ. Подробную информацию о задаче ранжирования можно найти в работе [1].

Релевантность документа может рассматриваться только в контексте какого-то конкретного запроса. Не имеет смысла говорить, что определенный документ в принципе является релевантным для всех возможных запросов, однако, можно оценить насколько данный документ релевантен по определенному запросу.

Все факторы, которые берут во внимание, когда определяют порядок на множестве документов, можно разделить на два больших множества:

- текстовые факторы
- поведенческие факторы

Текстовые факторы характеризуют, насколько сильно данный документ похож на данный запрос по тексту. Простейший текстовый фактор – это сколько общих слов содержит заголовок документа и запрос. Очевидно, что чем больше таких слов содержится, тем более релевантный документ с таким заголовком по данному запросу.

Поведенческие факторы характеризуют насколько пользователи поисковой системы были удовлетворены документом по данному запросу. Самый простой поведенческий фактор – это отношение кликов в документ по запросу к показам документа по запросу.

Поведенческие факторы подсказывают поисковой системе, что показывать по данному запросу, при учете той исторической информации, которая у нее есть по данному запросу. Например, если из исторической информации известно, что по конкретному запросу 80 % кликов приходило на сайт, который был на 8 позиции в выдаче, то стоит поставить его повыше, ведь из истории понятно, что он удовлетворяет информационную потребность пользователя.

У поведенческих факторов есть один большой недостаток – они известны только для небольшого подмножества запросов, так называемых высокочастотных запросов. Про огромное число запросов такой информации нет. Такие запросы соответственно называют низкочастотными. Важной и актуальной задачей в ранжировании является разработка методик, с помощью которых можно давать оценку поведенческим факторам для низкочастотных запросов, так называемое сглаживание поведенческой информации.

В данной работе предлагается производить сглаживания поведенческих факторов через модель похожести запросов. Предположим, что мы хотим оценить значение поведенческого признака для документа по заданному запросу. Мы рассматриваем все запросы, по которым у нас известно значение поведенческого признака с документом, находим эти поведенческие признаки. Имея модель похожести запросов, мы можем посчитать значение поведенческого признака для заданного запроса, усреднив все признаки с весами, равными значениям похожести запроса на заданный.

Задачами данной работы являются:

- Формальная постановка задачи сглаживания поведенческой информациями для задачи ранжирования
- Изучение существующих методов оценки поведенческой информации для низкочастотных запросов
- Формулировка задачи построения модели похожести поисковых запросов
- Проектирование новой системы, позволяющей оценивать поведенческую информацию через модель похожести запросов.

Новая предлагаемая система представляет собой систему поиска множества похожих высокочастотных запросов для низкочастотного запроса и оценку поведенческих факторов для документа и низкочастотного запроса.

Данная работа состоит из следующих разделов: постановка задачи и введение основных терминов, обзор существующих методов сглаживания поведенческой информации, описание предлагаемой методики сглаживания, описание результатов работы предложенного метода.

# 2 Постановка задачи

Путь имеется коллекция текстовых документов $D$, имеется множество текстовых запросов $Q$. Пусть для некоторых запросов из $Q$, которые мы будем называть $Q_{\text{train}}$, задан порядок для некоторой части документов из текстовой коллекции $D$, то есть для некоторого запроса $q \in Q_{\text{train}}$ известен правильный порядок для некоторых пар документов $d_i \in D$ и $d_j \in D$. Если $d_i \prec d_j$, то документ $d_j$ является более релевантным по запросу $q$, чем документ $d_i$.

Задача ранжирования заключается в построении решающего правила $a : Q \times D \to \mathbb{R}$, такого, что если $d_i \prec d_j$ по запросу $q$, то $a(q, d_i) < a(q, d_j)$.

Признаком для пары $(q, d)$ назовем какой-либо результат измерения пары $(q, d)$, иными словами, признаком называется отображение $f : Q \times D \to D_f$ , где $D_f$ – множество допустимых значений признака.

Таким образом, для задачи ранжирования мы имеем порядок для пар $(q, d)$ из некоторого обучающего множества, задача заключается в оценке этого порядка для всех остальных пар $Q \times D$.

Порядок на этапе обучения обычно задают одним из двух способов.

- Асессорская оценка. Асессора просят оценить для некоторого запроса список документов, то есть проставить им оценки в зависимости от релевантности данного документа к данному запросу. Асессор при выборе оценки руководствуется четкими критериями и правилами, описанными у него в инструкции.

- Поведенческая оценка. Оценить релевантность документа по данному запросу можно непосредственно из анализа поведения пользователей в поисковой сессии по данному запросу. Например, если документ в поисковой сессии был на первом месте в поисковой выдаче, но пользователь в него не кликнул, то скорее всего этот документ нерелевантен по данному запросу. Напротив, наличие клика в документ запроса иногда может свидетельствовать о релевантности документа по запросу. Более подробно о формировании поведенческих оценок можно найти в классической работе [8]. Однако, у данного подхода есть существенный недостаток – пользователи всегда взаимодействуют только с первыми документам из поисковой выдачи, то есть те документы, которые по каким-либо причинам не показывались поисковой системой на первых позициях, так и останутся навсегда без поведенческой оценки. Бороться с такой проблемой можно с помощью концепций многоруких бандитов [2, 3].

Существует большое число различных моделей, позволяющих строить решающее правило $a : Q \times D \to \mathbb{R}$. Описание самих моделей ранжирования существенно выходит за рамки данной работы, подробное описание существующих моделей можно найти в [7].

Поисковой машиной или поисковой системой будем называть алгоритм, решающий задачу ранжирования.

Поисковой сессией будем называть последовательность взаимодействий одного пользователя, задавшего один поисковой запрос, с поисковой машиной.

Поисковым серпом будем называть упорядоченное множество документов, выданное поисковой машиной, в ответ на запрос пользователя.

Напомним, что признаком в данной задаче мы будем понимать какое-либо измерение пары запрос-документ. Поведенческие признаки позволяют оценивать насколько документ релевантен по данному запросу, исходя из истории поисковых сессий. Приведем типичные примеры поведенческих признаков:

- Число переходов пользователей на данный документ по запросу

- Отношение числа переходов к числу показов данного документа по данному запросу. Документ можно считать показанным по данному запросу, если он находился, напри-

мер, в первой странице поискового серпа, то есть пользователь имел возможность, не изменяя страницу, перейти на этот документ.

- Число переходов пользователей на данный документ по запросу, после которых пользователи не возвращались обратно на поисковую систему
- Отношение числа таких переходов к общему числу переходов на документ
- Среднее время, за которое пользователь просматривает данный документ
- Среднее время, через которое пользователь переходит на данный документ из поискового серпа
- Количество возвращений на данный документ после первого посещения

Все поведенческие признаки строятся путем анализа поведения пользователей с поисковой системой. Они содержат информацию о том, как пользователи в среднем реагируют на данный документ, если они его видят в поисковой выдаче по какому-то определенному запросу. Поведенческие признаки имеют рад существенных недостатков, которые мы далее рассмотрим.

Во-первых, поведенческие факторы существенно зависят от позиции документа в поисковой выдаче. Действительно, предположим, что мы рассматриваем поисковую сессию, документ $d_2$ в которой находился на позиции номер 10. Предположим, что пользователь сначала перешел на некий иной документ $d_1$, который в свою очередь находился на первой позиции в выдаче. Мы хотим оценить привлекательность документов по данному запросу, то есть, например, посчитать число переходов в документ по запросу, деленное на число показов документа по запросу. Должны ли мы учитывать переход в документ $d_1$ с таким же весом, как и переход в документ $d_2$? Очевидно, что не должны, так как они находились существенно на разных позициях в выдаче. Пользователь мог перейти в документ $d_1$ случайно, а до документа $d_2$ ему пришлось просмотреть все документы между ними и решить, что документ $d_2$ предпочтительнее всех. Модели, которые позволяют строить поведенческие признаки, явным образом учитывая позицию документа в выдаче, называют кликовыми моделями.

Одним из важнейших типов кликовых моделей являются так называемые позиционные кликовые модели. Допустим, мы хотим оценить привлекательность документа $d$ по запросу $q$. Однако, мы наблюдаем только наличие перехода в документ $d$ по запросу $q$. Вероятность перехода зависит от двух факторов: вероятность пользователя найти этот документ в поисковом серпе (зависит от позиции документа в серпе) и от общей привлекательности документа (фактор самого документа). Очевидно, что в качестве поведенческого признака нам нужна оценка, несмещенная на позицию документа. Вероятность перехода $P(C = 1)$ можно оценить как:

$$P(C = 1) = P(E_r = 1) \cdot P(d_q = 1), \tag{1}$$

где $P(E_r = 1)$ вероятность того, что пользователь увидит документ, находящейся на позиции $r$ в серпе, а $P(d_q = 1)$ – вероятность, что документ $d$ релевантен по запросу $q$, которую мы хотим использовать в качестве поведенческого признака. Вероятность перехода можно оценить по принципу максимального правдоподобия через количество переходов и количество показов документа по запросу:

$$P(C = 1) = \frac{N_{clicks}(d, q)}{N_{shows}(d, q)}. \tag{2}$$

Вероятность дойти до позиции $r$ можно приближенно оценить через число кликов в документ на позиции $r$ и общее число показов позиции $r$.

$$P(E_r = 1) = \frac{N_{clicks}(r)}{N_{shows}(r)}. \tag{3}$$

Отсюда можно получить оценку релевантности документа к запросу, несмещенную на позицию $P(d_q = 1)$.

В данном случае была рассмотрена одна из простейших кликовых моделей. Существуют кликовые модели, которые оценивают вероятность $P(d_q = 1)$, моделируя поведения пользователя, взаимодействующего с поисковым серпом, так называемые каскадные модели. Рассмотрение этих и других кликовых моделей выходит за рамки работы, читатель может найти их подробное изложение в работах [4–6].

Другой проблемой, возникающей при построении поведенческих признаков, является небольшой размер пар $(q, d)$, для которых можно достоверно оценить поведенческие признаки.

Существует огромное число запросов, которые никогда ранее не задавались поисковой машине. Например, в 2017 году компания «Google» заявила [26], что примерно 15% запросов, которые они получают ежедневно от пользователей, никогда ранее не задавались. В меньших поисковых системах этот процент, очевидно, должен быть значительно выше. По таким запросам невозможно построить поведенческие признаки.

Если запрос ранее задавался поисковой система, то он мог задаваться не очень большое число раз. Это усложняет построение поведенческих признаков, так как при небольшом количестве поисковых сессий, поведенческая информация является очень шумной, и ей нельзя доверять.

Даже если запрос часто задавался поисковой машине, то поведенческая информация по этому запросу есть только у небольшого числа документов – тех документов, которые были на самом верху поискового серпа. До большинства документов, которые не оказались наверху серпа, пользователь просто не дошел, поэтому поведенческой информации для таких документов по запросу все равно нет.

Таким образом существует огромное число ситуаций, когда или поведенческой информации для пары $(q, d)$ не существует, или эта поведенческая информация очень шумная из-за небольшого числа поисковых сессий. Для всех этих ситуаций необходимо применять так называемый механизм **сглаживания поведенческой информации**.

Сглаживание поведенческой информации позволяет оценить поведенческие признаки для тех пар $(q, d)$, для которых нет поведенческой информации, или она крайне шумная. То есть задача сглаживания поведенческих признаков заключается в том, чтобы для поведенческого признака $f : Q \times D \to D_f$ построить его приближение $\hat{f} : Q \times D \to D_f$, а затем это приближение использовать для тех пар $(q, d)$, для которых неизвестно или неточно известно значение самого признака $f$.

Далее рассмотрим существующие подходы, применяемы для решения задачи сглаживания поведенческой информации.

# 3 Методы сглаживания поведенческой информации

Большинство алгоритмов, осуществляющих сглаживание поведенческой информации основано на методах кластеризации, то есть на методах, которые каким-либо образом позволяют распределять поведенческую информацию по всем похожим парам запрос-документ. Похожесть при этом может определяться по-разному или задаваться неявным образом.

Рассмотрим вероятностную постановку задачи ранжирования, которую ввели в работе [9]. Рассмотрим генеративную модель для запросов, документов и слов. Каждый запрос порождает мультиномиальное распределение для слов. Каждое слово в словаре порождает мультиномиальное распределение для документов. Введя такие распределения, можно в вероятностном смысле определить релевантность документа по запросу, как вероятность выбрать документ $d$ после задания запроса $q$. То есть:

$$Rel(d, q) = p(d|q) = \sum_{t_j \in q} p(t_j|q) \cdot p(d|t_j) \tag{4}$$

Добавляя определенную параметризацию на распределения для слов и на распределения для документов, несложно оценить $p(t|q)$, при чем, вводя другую параметризацию, мы будем получать существенно другую функцию ранжирования. Вероятность документа при наличии слова $p(d|t)$ можно оценить по принципу максимума правдоподобия через встречаемость слова $t$ в документе $n(t, d)$.

$$p(d|t) = \frac{n(d, t)}{\sum_{d_l \in D} n(d_l, t)} \tag{5}$$

Модель порождения документа $d$ по запросу $q$, заданная с помощью распределения над запросами и распределения над словами, может быть рассмотрена как блуждание по многодольному графу запросов, слов и документов. Схематичное представление этого графа представлено на рисунке 1.

Не ограничивая общности, будем считать, что мы проводим две итерации блуждания по этому графу. Будем считать, что после первого прохода при достижения документа $d$ с вероятностью $\alpha$ процесс заканчивается, или мы продолжаем второй проход от всех слов

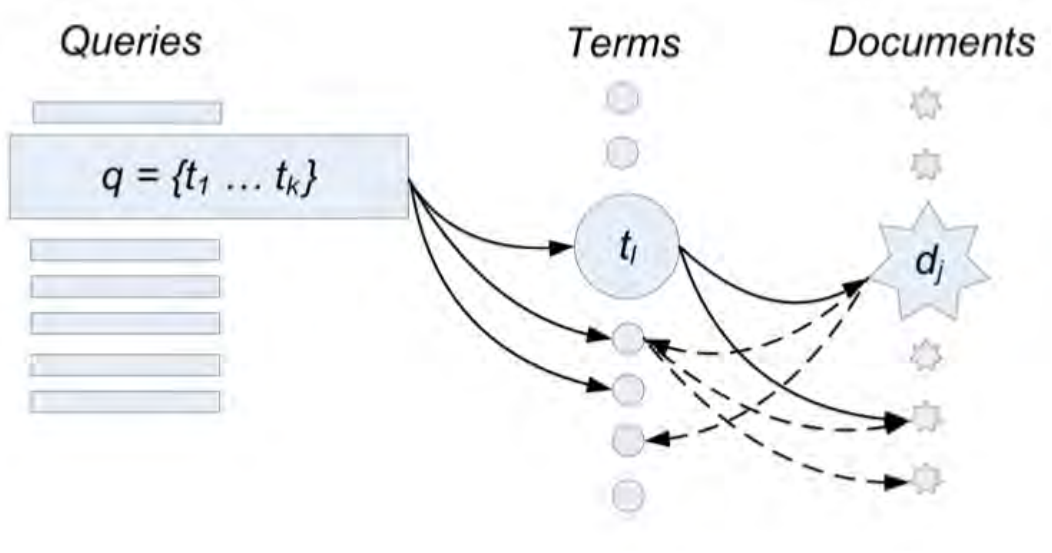


Рис. 1: Граф запросов, слов и документов

документов, до которых мы дошли, ко всем остальным документам. Тогда релеватность документа к запросу может быть переписана следующим образом:

$$Rel(d,q) = \sum_{t_j \in q} p(t_j|q) \cdot \alpha \cdot p(d|t_j) + (1-\alpha) \sum_{t_l \in d_j} p(d_j|d_j) \cdot p(t_l|d_j) \cdot p(d|t_l) \tag{6}$$

Этот процесс случайного блуждания производит как раз эффект сглаживания, так как вероятность $p(d|q)$, то есть вероятность дойти до документа $d$, отныне порождается не только словами $t_j$ из запроса $q$, но и теми словами $t_l$, которые мы нашли в тех документах, после того как сделали первый проход из слов запроса $q$.

В работе [9] были проведены эксперименты, которые показали, что подобная система позволяет улучшить качество поиска. Аналогичную систему сглаживания в виде случайного блуждания можно найти во многих работах, посвященных данной теме.

Данный подход, к сожалению, работает очень медленно. Ограничение в 2 прохода было сделано из-за очень длительного времени прохода при больших коллекциях документов, а размер коллекций в поисковых системах с каждым годом только растет, что делает техническую реализацию метода крайне тяжелой задачей.

В работе [10] представлен метод сглаживания через блуждание по двудольному графу запросов и документов, уже без использования напрямую слов запроса и документа.

Двудольный граф запросов и документов представлен на рисунке 2. В левой доли данного графа представлены запросы, а в правой – документы.

Каждое ребро этого графа связано с вероятностью $p(d|q)$ – перейти с данного запроса в данный документ $q$. Построим из этих вероятностей матрицу перехода $A_{i,j} = p(d_j|q_i)$. Нормализуем эту матрицу, чтобы ее можно было считать матрицей перехода из запросов в документы. Аналогичным образом построим матрицу переходов $B_{i,j}$, путем транспонирования и перенормирования матрицы $A$, чтобы $B_{i,j}$ была матрицей перехода из документов

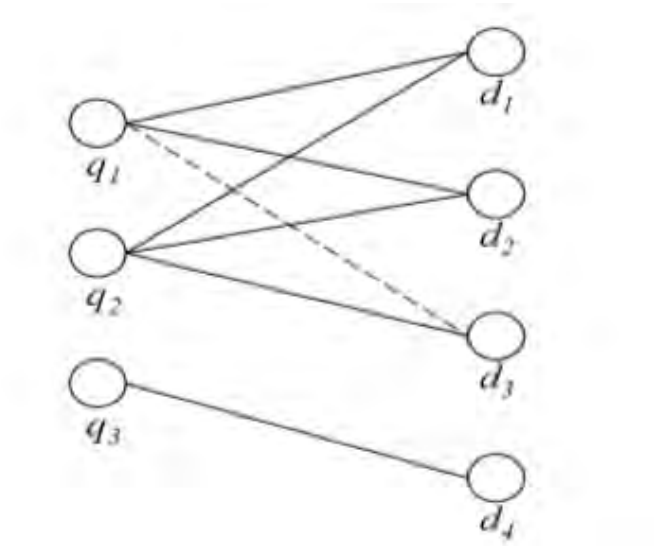


Рис. 2: Граф запросов и документов

в запросы. В данном случае нам неважно, что пользователи задают именно запросы, а переходят в документы, мы оперируем запросами и документами просто как равноправными вершинами двудольного графа.

Вероятности $p(d_j|q_i)$ можно оценить просто по принципу максимума правдоподобия, как число переходов в документ $j$ по запросу $i$, деленное на число показов документа $j$ по запросу $i$.

Легко заметить, что, оперируя матрицами $A$ и $B$, можно посчитать вероятности перехода из каждой вершины в каждую за любое число шагов. Зная эту вероятность, можно уже посчитать похожесть двух запросов как вероятность перейти из вершины одного запроса в вершину другого за $2s$ шагов. Очевидно, что эта вероятность равна $(A \cdot B)^s$. Далее можно считать, что документы, которые имеют высокую вероятность $p(d|q_1)$ у запроса $q_1$ можно показывать и по запросу $q_2$, при условии, что $sim(q_1, q_2) > \alpha$.

На рисунке 2 можно наглядно увидеть, как работает данный метод. Изначально между запросом $q_1$ и документом $d_3$ не было никакой вероятности перехода – просто не было сессий, где по запросу $q_1$ пользователь перешел бы на $d_3$. Однако, после нескольких итераций случайного блуждания между $q_1$ и $d_3$ начинает возникать вероятность перехода, так как запрос $q_1$ похож на запрос $q_2$ своей связанностью с документами $d_1$ и $d_2$, а по запросу $q_2$ были сессии с переходами в документ $d_3$.

Заметим, что в данном случае метод сглаживания построен не на общих словах, которые порождают документы, как было в предыдущей работе, а напрямую связан с тем, в какие документы часто переходят по запросу. Это, с одной стороны, упрощает реализацию и ускоряет работу данного метода, а с другой стороны имеет серьезный недостаток – невозможно сделать сглаживания для тех запросов, которые ранее никогда не задавались поисковой машине.

Подобные идеи сглаживания через случайные блуждания используются во множестве

других работ. В работе [11] авторы совместили сглаживание через случайное блуждание и сглаживание через языковое моделирование. В работе [12] проводилось сглаживание через случайное блуждание в двудольном графе, в котором в вершинах были метаданные. Это могут быть, например, признаки, которые характеризуют данный запрос или документ. Авторы показали, что, используя дополнительно в сглаживании метаданные, можно добиться более высокого качества в определении похожести запросов и документов.

Помимо сглаживания через случайные блуждания существуют другие варианты сглаживания.

Существуют работы, где похожесть запросов для сглаживания определялась по тексту двух запросов. В работе [13] авторы пытались оценить похожести двух запросов через различные метрики над их текстами. Как метрика семантической похожести авторами предлагалась взаимная информация между словами из двух разных запросов:

$$PMI(w_i, w_j) = \log \frac{p(w_i, w_j)}{p(w_i) \cdot p(w_j)} \tag{7}$$

В этом выражении $p(w_i)$, $p(w_j)$ и $p(w_i, w_j)$ вероятность встретить в запросе слово $w_i$, слово $w_j$ и в паре семантических похожих запросов слова $w_i, w_j$ соответственно. Эти вероятности легко оценить по счетчикам встречаемости, а затем посчитать выражение для $PMI$.

Помимо семантических расстояний между запросами, авторы предлагают использовать лексические расстояния. Как лексическое расстояние между запросами авторы предлагают использовать расстояние Левенштейна, то есть минимальное количество операций вставки, удалений и замены символов, с помощью которых можно из первого запроса получить второй. Хорошо известно, что расстояние Левенштейна может быть эффективно посчитано методами динамического программирования за $O(N \cdot M)$, где $N$ и $M$ длины двух рассматриваемых запросов.

Комбинируя лексические и семантические метрики, авторам удалось построить алгоритм, позволяющий давать как лексическую, так и семантическую оценку близости двух различных запросов. Имея такую оценку, можно в дальнейшем использовать ее для сглаживания, то есть переносить поведенческую информацию с одного запроса на близкие к нему другие запросы.

Можно оценивать похожесть запросов не с помощью какого-то отдельного алгоритма, а прямо через модель ранжирования. В работе [14] училась модель ранжирования в виде:

$$f(q, d) = h(q, d) + \frac{1}{|S_q|} \sum_{q_i \in S_q} s(q, q_i) \cdot h(q_i, d) \tag{8}$$

То есть предсказание модели $f(q, d)$ для запроса $q$ и документа $d$ представляется в виде базового предсказания качества документа $d$ к запросу $q$ $h(q, d)$ и еще нормированной суммы по всем запросам-кандидатам $S_q$. Каждое слагаемое этой суммы представляет из себя произведение похожести базового запроса с кандидатом $s(q, q_i)$, умноженное на базовое предсказание качество документа $d$ к запросу кандидату $q_i$.

В качестве функции потерь можно использовать, например, классическую квадратичную функцию потерь:

$$E = \sum_{q,d} (y_{q,d} - f(q,d))^2 \tag{9}$$

В итоге функция ошибки $E$ итеративно минимизируется. Сначала фиксируется $s$, минимизируется $h$, на следующем шаге фиксируется $h$, а минимизируется $s$. Путем многократного повторения этой процедуры находятся итоговая функция похожести $s$ и базовое предсказание $h$.

Авторы протестировали данный метод на нескольких наборах данных и пришли к выводу, что такой подход к сглаживанию значительно превосходит классические модели поиска, где предсказание модели по сути представляет собой только функцию $h(g,d)$, без дополнительных предсказаний на похожих запросах кандидатах.

У данного подхода, к сожалению, есть один большой недостаток: для получения предсказания модели для пары $(q,d)$ необходимо получить предсказание модели для множества пар $(q_i,d)$. В современных поисковых системах модель $h(q,d)$ зачастую является крайне сложной за счет огромного числа различных факторов, которые используются в поисковых системах. Получение предсказания этой модели на множестве других запросах $q_i$, ровно как и построение самих признаковых описаний для пар $(q_i,d)$, является вычислительно крайне дорогой задачей.

# 4 Модель похожести запросов

Рассмотрим модель сглаживания поведенческой информации, предлагаемую в данной работе.

Здесь и далее обозначим через $q$ запрос, по которому мы хотим оценить поведенческих признаки для документа $d$. Запросами $q'$ будем называть запросы, которые часто задаются пользователями, то есть по ним есть поведенческая информация.

Не ограничивая общности, будем считать, что мы хотим оценить поведенческий признак – вероятность клика в документ $d$ по запросу $q$. Если бы для этой пары была поведенческая информация, то мы могли бы эту вероятность оценить по принципу максимума правдоподобия через число кликов в документ по запросу, деленное на число показов документа по запросу:

$$p(click|q,d) = \frac{n_{clicks}(q,d)}{n_{shows}(q,d)} \tag{10}$$

К сожалению, напрямую оценить это значением мы не можем, однако для любого запроса $q' \in Q'(d)$, где $Q'(d)$ – множество запросов, для которых имеется поведенческая информация для документа $d$, возможно оценить:

$$p(click|q', d) = \frac{n_{clicks}(q', d)}{n_{shows}(q', d)} \tag{11}$$

Зная выражение для $p(click|q', d)$, дальше можно оценить $p(click|q, d)$ по принципу, похожий на принцип сглаживания из работы [14]:

$$p(click|q, d) = \sum_{q' \in Q'} sim(q, q')p(click|q', d); \tag{12}$$

Отметим, что значения похожести запросов $sim(q, q')$ необходимо отнормировать, чтобы они суммировались в единицу:

$$\sum_{q' \in Q'} sim(q, q') = 1 \tag{13}$$

Данный подход обладает несколькими достоинствами. Во-первых, такое сглаживание можно проводить для запросов, которые задаются впервые. Во-вторых, при эффективной реализации модели $sim(q, q')$ и при выборе небольшого множества $Q'$, можно проводить достаточно быстрое сглаживание, так как оно происходит только по текстам запросам без какой-либо дополнительной информации.

Таким образом для сглаживания нам необходимо выучить модель похожесть запросов $sim(q, q')$, а именно решающее правило $a_{sim} : Q \times Q \to \mathbb{R}$.

## 4.1 Формирование обучающего и проверочного множества

Рассмотрим предлагаемую схему формирования обучающего и проверочного множества для обучения модели похожести двух запросов $a_{sim}$.

Пусть $D_q = \{d : \exists click(q, d)\}$, то есть это множество документов, по которым были клики по запросу $q$. Будем считать, что два запроса $q_1$ и $q_2$ похожи, если:

$$\frac{|D_{q_1} \cap D_{q_2}|}{|D_{q_1} \cup D_{q_2}|} > \text{threshold}, \tag{14}$$

где threshold порог, после которого мы считаем, что запросы похожие. Эта формула реализует простую концепцию – запросы являются похожими, если по ним люди переходят на одинаковые документы. Таким образом можно получить большое число пар похожих друг на друга запросов.

Для построения модели $a_{sim}$ с помощью классических моделей машинного обучения с учителем помимо класса похожих запросов необходимо задать класс непохожих запросов. В данной работе рассматривалось два варианта построения пар непохожих запросов:

- Случайное сэмплирование. Будем считать непохожими запросами любую случайную пару запросов $(q_i, q_j)$. Этот метод позволяет быстро генерировать негативные примеры в огромных количествах. Основным недостатком данного метода является низкое качество негативных примеров – даже очень простая модель с большой точностью сможет

отличить похожую пару запросов от двух совершенно случайных запросов. Далее этот процесс будем называть random sampling.

- Сэмплирование примеров, в которых наиболее уверена модель. Данный процесс обучения модели происходит итеративно. Пусть на итерации $i$ была построена модель $a^i_{sim}$. Тогда будем строить модель $a^{i+1}_{sim}$ на тех же положительных примерах, что и модель $a^i_{sim}$, а в качестве отрицательных примеров, будем брать такие случайные пары $(q_k, q_l)$, у которых $a^i_{sim}(q_k, q_l) > \text{thr}$, где thr некоторый порог. То есть в качестве негативных примеров, будем брать те случайные пары запросов, которые имели высокое значение похожести у предыдущей модели. Такой подход имеет очевидный недостаток – медленная скорость построения итоговой модели, однако негативные примеры с каждой итерацией становятся все сложнее и сложнее, после какой-то итерации их должно быть трудно отделить от позитивных. Этот процесс в литературе называют hard negative mining.

Перейдем к построению тестовых данных, на которых будем оценивать качество моделей. При формировании методики оценки моделей, важно понимать, что при такой постановке задачи нельзя в качестве оценки использовать метрики классификации. Совершенно безразлично, какой процент похожих запросов модель реально найдет. Важно, чтобы значение $sim(q_i, q_j) = a(q_i, q_j)$ для двух похожих запросов $q_i, q_j$ было выше, чем для двух непохожих запросов, так как именно с этими весами в дальнейшем будет производиться сглаживание поведенческих признаков. Важно, чтобы поведенческие признаки похожих запросов шли с большим весом, чем поведенческие признаки непохожих.

В качестве позитивных примеров в тестовое множество будем брать те пары похожих запросов, которые не попали в обучающее множество. Важно отметить, что в качестве негативных примеров при тестировании нельзя использовать случайные примеры, значения метрики качества будут слишком высокими даже для простой функции похожести, такой как, например, число общих слов. Очевидно, что у случайной пары запросов маловероятно будут общие слова.

Исходя из вышеизложенного, в данной работе предлагается следующая методика для оценки качества модели похожести запросов. Тестовый набор данных состоит из троек: базовый запрос ($q_{base}$), запрос, похожий на базовый ($q_{true\_sim}$) и запрос, похожий на базовый по тексту, но не являющийся похожим по метрике (14) ($q_{false\_sim}$). Похожесть по тексту между $q_{base}$ и $q_{false\_sim}$ определялась через процент одинаковых слов у этих двух запросов. Отметим, что из-за условия на наличие общих слов между $q_{base}$ и $q_{false\_sim}$, вполне вероятно, что они на самом деле являются похожим, просто для них не набралось достаточно общих кликнутых документов в поисковой истории. По этой причине все тестовые тройки должны быть провалидированы асессором на предмет таких ошибок.

В качестве метрики качества предлагается использовать процент троек, в которых вы-

полнялось условие:

$$a(q_{base}, q_{false\_sim}) < a(q_{base}, q_{true\_sim}). \tag{15}$$

Во всех экспериментах, которые будут описаны далее, подразумевается использование именно этой метрики качества.

## 4.2 Базовые модели

В данном разделе будут описаны некоторые популярные модели, часто применяемые в информационном поиске и во многих других задачах обработки естественного языка. Все эти модели можно применять для оценки похожести запросов.

Самой простой моделью, которую можно использовать для определения похожести запросов, является модель общих слов. В этой модели похожесть $sim(q_1, q_2)$ равна количеству одинаковых слов в запросах $q_1$ и $q_2$. Очевидно, что из-за вышеизложенной методики построения тестового множества качество такой модели может вполне быть меньше 50%.

В качестве следующего класса моделей рассмотрим вероятностные модели информационного поиска. В этих моделях, как было показано в работе [15], вес совпадения слова $i$ в документе $d_j$ $w_{i,d_j}$ не одинаков для всех слов, как было в предыдущей модели, а в общем виде может быть записан как:

$$w_{i,d_j} = L_{i,d_j} \cdot G_i, \tag{16}$$

где $L_{i,d_j}$ – вес слова $i$ в документе $d_j$ (так называемый локальный вес слова), $G_i$ – глобальный вес слова, то есть его абсолютная важность совпадения без учета самого документа. Теперь будем считать, что мы ищем не слова запроса в документе, как в обычном информационном поиске, а слова запроса в другом запросе. Воспользовавшись предыдущими обозначениями, будем считать похожесть как:

$$sim(q_i, q_j) = \sum_{i \in q_i} w_{i,q_j} \tag{17}$$

В данной работе было рассмотрено две вероятностных модели.

В первой модели $G_i = \log \frac{N}{n(q_i)}$, $L_{i,q_j} = tf(i, q_j)$, где $N$ – общее число запросов коллекции, $n(q_i)$ – число запросов, содержащие слова $i$, $tf(i, q_j)$ – число вхождений слова $i$ в запрос $q_j$. Модель с такими $L_{i,q_j}$ и $G_i$ часто называют TF–IDF моделью.

В модели BM25 [16] глобальный вес слова $G_i$ считается так же, как в модели TF–IDF, а локальный вес считается по формуле $L_{i,q_j} = \frac{tf(i,q_j)\cdot(1+k)}{k\cdot((1-b)+b\cdot\frac{dl_{q_j}}{dl_{avgd}})+tf(i,q_j)}$, где $dl_{q_j}$ – длина запроса $q_j$, $dl_{avgdl}$ – средняя длина запроса в коллекции, $k$ и $b$ – коэффициенты. Обычно данная модель превосходит по качеству модель TF–IDF за счет явного учета длины документа в локальном весе $L_{i,q_j}$.

Предыдущие модели были основаны на прямом совпадении слов – значение похожести увеличивается на $w_{i,q_j}$ если слово $i$ встретилось в запросе $q_j$. Однако запросы $q_i$ и $q_j$ вообще могут не иметь общих слов, но при этом быть очень похожими. Для этого обычно применяют

так называемые семантические модели, которые должны оценить семантическую похожесть слов из двух разных вопросов, а не точное пересечение.

В работе рассматривалось несколько семантических моделей. Одна из них была популярной моделью Word2Vec [17]. Эта модель позволяет для каждого слова сопоставить вектор чисел, причем у похожих по смыслу слов, эти вектора будут похожими. Модель обучается двумя различными способами:

1. Модель пытается предсказать для каждого слова его контекст в некотором окне размера $W$.

2. Модель пытается предсказать для контекста размера $W$ недостающее в контексте слово.

Обе модели часто используются и показывают примерно одинаковые результаты. Первый вариант, как правило, быстрее обучается.

Для предсказания похожести двух запросов $q_i, q_j$ предлагается сделать следующее – усреднним вектора для всех слов из запросов $q_i$ и $q_j$, получим вектора $v_{q_i}, v_{q_j}$. Тогда похожесть будем считать по формуле:

$$sim(q_i, q_j) = \frac{(v_{q_i}, v_{q_j})}{|v_{q_i}| \cdot |v_{q_j}|} \tag{18}$$

В качестве еще одной семантической модели была использована модель LSA [18]. Суть метода заключается в том, что запросы $q_i$ и $q_j$ проецируются в пространство меньшей размерности, в этом пространстве им соответствуют вектора $v_{q_i}$ и $v_{q_j}$, а затем значение похожести считается по формуле (18).

Для оценки похожести можно использовать и модели обучения с учителем. Сопоставим каждому слову из словаря размерности $|V|$ число от 1 до |V|. Сопоставим запросу $q$ вектор размерности $|V|$, в котором все элементы равны 0, кроме ячеек, позиции которых соответствуют словам из запроса $q$. В этих ячейках стоят единицы. Такой способ кодирования часто называют «one hot кодирование». Тогда для каждый пары запросов $q_i, q_j$ можно объединить эти вектора и считать такие объединения признаковым пространством. Далее с таким пространством может работать модель обучения с учителем, то есть предсказывать, является ли данная пара похожей, а затем выход модели можно интерпретировать как степень похожести $sim(q_i, q_j)$. В данной работе применялась модель Gradient Boosting Machine [19].

Данная модель обучения с учителем будет часто применяться в данной работе. Была выбрана конкретно эта модель по нескольким фактора

- Во многих задачах с открытыми данными на одинаковых признаковых пространствах эта модель показывала более высокие результаты, чем другие.

- Модель позволяет строить нелинейные разделяющие поверхности.

- Для модели существуют эффективные готовые реализации. Например, Xgboost [1].

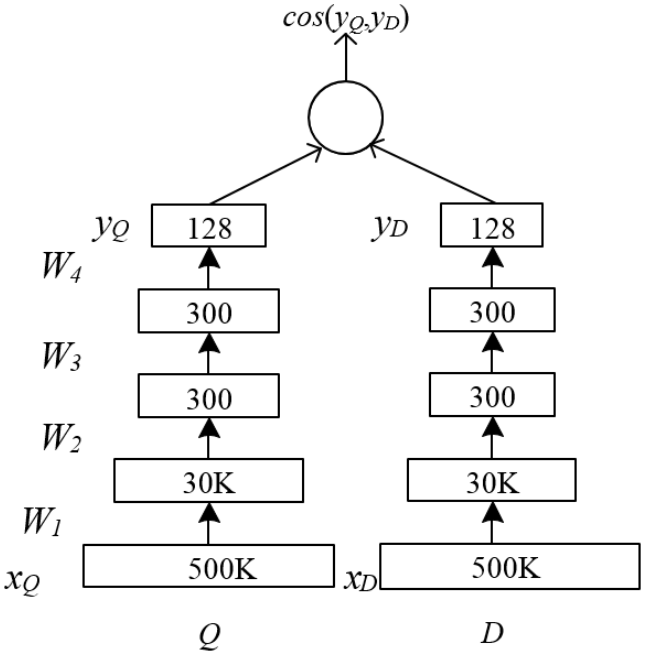


Рис. 3: Модель DSSM

Помимо градиентного бустинга в качестве модели обучения с учителем часто используются нейронные сети. Для предсказания похожести запросов была выбрана популярная в задаче ранжирования архитектура DSSM [21, 22]. Архитектура нейронной сети схематично представлена на рисунке 3. Эта модель представляет собой глубокую нейронную сесть, которая проецирует запрос $q$ и документ $d$ в пространство меньшей размерности, а затем считается похожесть двух векторов по формуле (18). В оригинальной статье она настраивается таким образом, чтобы похожесть между запросом и документом, по которым был клик, была высокая, а похожесть между запросом и документом, по которым клика не было, была низкая. В данной работе архитектура была абсолютно такая же, только кодировались и считалось выражение (18) между двумя запросами. У похожих запросов выражение (18) должно быть высоким, у непохожих – низким. Это выражение и интерпретируется в качестве $sim(q_i, q_j)$.

Перейдем к рассмотрению модели, предлагаемой в данной работе для оценки похожести запросов. Модель состоит из 3 независимых частей, рассмотрим их поочередно.

## 4.3 Модель совпадения слов

Первой моделью рассмотрим модель совпадения слов. Эта модель построена по той же структуре, что и вероятностные модели TF–IDF и BM25, которые были рассмотрены ранее. Будем считать значение похожести двух запросов по следующей формуле:

[1] `https://github.com/dmlc/xgboost`

$$sim(q_i, q_j) = \sum_{i \in q_i} w_{i,q_j}, \tag{19}$$

где $w_{i,q_j}$ вес слова $i$ в запросе $q_j$. В вероятностных моделях суммирование в подобной формуле всегда идет по словам запроса, а затем эти слова ищутся в документе. Так как в нашей задаче запросы, для которых мы ищем похожесть совершенно равноценные, то формулу можно переписать:

$$sim(q_i, q_j) = \frac{1}{2}(\sum_{i \in q_i} w_{i,q_j} + \sum_{j \in q_j} w_{j,q_i}) \tag{20}$$

Как и в вероятностных моделях разложим $w_{i,g_j}$:

$$w_{i,g_j} = L_{i,q_j} \cdot G_i, \tag{21}$$

где $L_{i,q_j}$ – локальный вес слова $i$ в запросе $q_j$, $G_i$ – глобальный вес слова $i$.

В качестве $L_{i,q_j}$ попробуем два варианта: модель TF–IDF и модель BM25, а глобальные веса $G_i$ будем учить моделью логистической регрессии. Эта модель соответствует следующему признаковому описанию: в качестве признаков для пары $(q_i, q_j)$ подается вектор размерности $|V|$ (размер словаря), в котором все элементы равны 0, кроме ячеек, позиции которых соответствуют общим словам из запросов. В этих ячейках стоят веса $L_{i,q_j}$.

Таким образом данная модель явным образом выучивает насколько запросы $q_i, q_j$ стали более похожими, если у них совпало слово $i$.

Если мы отказываемся от явного выучивания модели, аналогичной вероятностным моделям информационного поиска, то необязательно использовать линейную модель. В данной работе такое же признаковое пространство еще подавалось модели Gradient Boosting Machine [19]. Эта модель уже не будет соответствовать вероятным моделям информационного поиска, однако она может учитывать нелинейные зависимости в данных.

## 4.4 Модель несовпадения слов

Модель несовпадения слов логически дополняет предыдущую модель совпадения слов. Таким же образом, как предыдущая модель совпадения добавляла в значение $sim(q_i, q_j)$ всякое совпадение слов из запросов, модель несовпадения слов должна отнимать из этого значения всякое несовпадение слов.

Вероятностные модели информационного поиска для такой задачи в явном виде не предназначены, попробуем немного их модернизировать в соответствии с вышеупомянутой логикой. Будем искать веса для несовпадений слов таким же образом:

$$w'_{i,g} = L'_{i,q} \cdot G'_i, \tag{22}$$

где $G'_i$ – глобальный вес несовпадения слова $i$, $L'_{i,q}$ – локальный вес слова $i$ в запросе $q$. Значение похожести запросов будем оценивать следующим образом:

$$sim(q_i, q_j) = \frac{1}{2} \cdot (\sum_{i \in q_i, i \notin q_j} w'_{i,q_i} + \sum_{j \in q_j, j \notin q_i} w'_{j,q_j}). \quad (23)$$

Таким же образом, как для модели совпадения слов, представляя пару $q_i, q_j$ в признаковом пространстве через $L'_{i,q}$ можно выучить веса $G'_i$ линейной моделью. На этом же признаковом пространстве можно использовать нелинейную модель. В экспериментах использовалась Gradient Boosting Machine в качестве модели несовпадения слов.

## 4.5 Семантическая модель

Две предыдущие модели использовали четкое совпадение или несовпадение запросов по словам. Семантическая модель должна быть построена таким образом, чтобы без пословного сравнения оценивать похожесть запросов $sim(q_i, q_j)$. В данной работе исследовалось несколько возможных вариантов построения семантической модели. У всех них была неизменная структура – каждому слову из словаря |V| должен сопоставиться определенный вектор таким образом, чтобы затем, имея вектора слов для каждого запроса, можно было судить об их семантической близости или об отсутствие таковой. Разница только в технологии обучения моделей.

В первом варианте модель строилась аналогично модели DSSM. После получения вектора для каждого слово из запросов $q_i, q_j$, векторы слов каждого из запросов усреднялись, а затем между этими средними векторами считалась близость (18), отшкалированная на отрезок $[0, 1]$. У семантически близких запросов это значение должно быть близко к единице. Для обучения использовалась логистическая функция потерь. В дальнейшем в экспериментах эту модель будем называть Semantic Mean Distance.

В другом варианте семантическая модель старалась предсказывать слова из похожих запросов. Представим такой граф, в котором вершиной являются запрос, а ребро между запросом $q_i$ и запросом $q_j$ существует, если эти два запроса похожи. Затем выделим в этом графе компоненты сильной связности – эта процедура позволит нам выделить похожие группы запросов. Дальше будем для каждого слова из запроса $q_i$ предсказывать все слова из его компоненты сильной связности. Эта процедура обучения аналогично обучению модели word2vec, только в этом случае роль окна выполняет компонента сильной связности. Для предсказания похожести двух запросов используется такая же процедура, как в Semantic Mean Distance. Эту модель далее будем называть Semantic Strong Components.

Еще один вариант семантической модели основан на так называемой матрице семантических расстояний, которая часто применяется в моделях ранжирования [23, 24]. Пусть запрос $q_i$ состоит из $T_1$ слов, а запрос $q_j$ из $T_2$ слов. Тогда назовем матрицей взаимодействия размерности $T_1 \times T_2$ матрицу $M$, каждый элемент в которой равен $M_{i,j} = w_i \bigotimes v_j$, где $w_i$ векторное представление слова $i$ из запроса $q_i$ размерности $h$, а $v_j$ – векторное представление слова $j$ из запроса $q_j$ той же размерности, а символ $\bigotimes$ означает некоторую функцию похожести

$\mathbb{R}^h \times \mathbb{R}^h \to \mathbb{R}$ . Схематичное представление матрицы взаимодействий представлено на рисунке 4. В качестве функции похожести в работе использовалась формула (18).

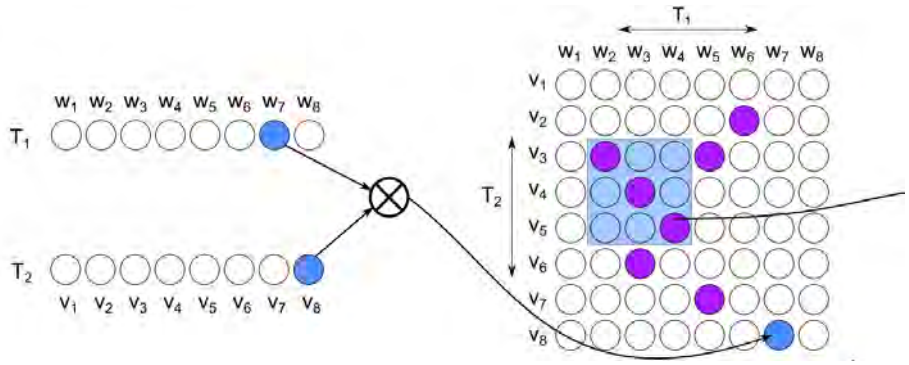


Рис. 4: Матрица взаимодействий

Дальше матрица $M_{i,j}$ преобразуется в вектор длины максимального размера матрицы $M_{i,j}$ в обучающем множестве, в котором лишние ячейки заменяются нулями. Затем этот вектор используется для предсказания похожести запросов. В качестве функции потерь использовалась логистическая функция потерь. Эту модель будем называть Semantic Matrix Classifier.

В более сложном варианте, после такого обучения из этой матрицы извлекаются признаки, основанные на статистике взаимодействий слов – максимальное взаимодействия слов, среднее взаимодействие слов, минимальное взаимодействие и так далее. На этих признаках обучается новый классификатор, в данной работе использовалась модель Gradient Boosting Machine. Эту модель будем называть Semantic Matrix Meta Classifier.

# 5 Эсперименты

Все эксперименты проводились на данных отдела поиска компании Mail.Ru Group. Важно отметить, что так как размер данных в поисковых системах довольно объемный, а процесс формирования похожих запросов подразумевает поиск всех документов, в которых был переход по данному запросу, то процесс формирования обучающего множества осуществлялся с помощью технологии MapReduce [20].

В результате была составлена обучающая выборка, состоящая примерно из 25 миллионов пар похожих запросов. Примеры пар запросов из обучающего множества представлены в таблице 1.

Для проверочного множества было получено около 5 тысяч троек, которые были проверены асессором. Напомним, что тестовое множество состоит троек запросов. Второй запрос похож на первый, а третий запрос похож на первый только по тексту, но не по смыслу. Примеры таких троек представлены в таблице 2.

В качестве метрики считается процент троек, где значение похожести у первых двух запросов, выше, чем у первого и третьего.

Таблица 1: Похожие запросы из обучающего множества

| Запрос 1 | Запрос 2 |
|---|---|
| куплю шотландского кота | продажа кошек |
| решения ответы математике 6 класс | гдз по математике |
| погода в Рязани | прогноз погоды Рязань |
| купить квартиру в Москве | продажа недвижимости Москва |

Таблица 2: Проверочное множество

| Базовый запрос | Похожий на базовый | Непохожий на базовый |
|---|---|---|
| администрация Омск | администрации Омска сайт | администрация Ижевска |
| Флеш первый сезон | смотреть онлайн Флеш 1 сезон | Наруто первый сезон |
| смотреть дневники вампира | сериал дневники вампира онлайн | смотреть Флеш |
| фильмы с Кэмерон Диас | новое кино Кэмерон Диаз | фильмы с Уиллисом |

Перед применением моделей весь текст был приведен к нижнему регистру и отлемматизирован, были отброшены знаки препинания.

## 5.1 Модель похожести запросов

В таблице 3 представлены результаты всех моделей на тестовом множестве.
Из этой таблицы важно отметить следующее:

- Модель BM25 превосходит модель TF–IDF. Обычно так происходит и для задачи ранжирования
- Ансамбль лексическая модель (BM25) + семантическая (word2vec) превосходит просто семантическую модель
- GBM модель на словах двух запросов не смогла показать высокого качества
- Линейная модель на BM25 признаках работает лучше, чем на TF–IDF признаках
- В случае модели совпадения слов линейная модель, которая по сути обучает веса у аналогичной ей вероятностной модели, работает намного лучше, чем GBM
- В качестве модели несовпадения лучше работает GBM. Разница между TF–IDF и BM25 несущественна

Таблица 3: Результаты моделей на тестовом множестве

| Модель | Качество |
|---|---|
| Число общих слов | 46.2 % |
| TF–IDF | 73.8 % |
| BM25 | 77.9 % |
| LSA | 58.6 % |
| word2vec | 60.5 % |
| BM25 + word2vec | 79.2 % |
| GBM на текстах двух вопросов | 59.3 % |
| DSSM | 83.3 % |
| TF–IDF модель совпадения | 82.9 % |
| BM25 модель совпадения | 82.9 % |
| GBM модель совпадения | 79.3 % |
| TF–IDF модель несовпадения | 85.0 % |
| BM25 модель несовпадения | 85.6 % |
| GBM модель несовпадения | 87.1 % |
| Semantic Mean Distance | 72.7 % |
| Semantic Strong Components | 71.5 % |
| Semantic Matrix Classifier | 78.1 % |
| Semantic Matrix Meta Classifier (SMMC) | 81.2 % |
| BM25 модель совпадения + BM25 модель несовпадения | 87.8 % |
| BM25 модель совпадения + GBM модель несовпадения | 89.8 % |
| BM25 модель совпадения + BM25 модель несовпадения + SMMC | 89.5 % |
| BM25 модель совпадения + GBM модель несовпадения + SMMC | 91.0 % |

- В качестве семантической модели с большим преимуществом выигрывает Semantic Matrix Meta Classifier
- Композиции всех трех моделей показывают самое высокое качество. Это согласуется с последними результатами в задаче ранжирования. В работе [25] было показано, что лексическая и семантическая модели дают более высокое качество, чем каждая по отдельности.

Знаком «+» в таблице показано ансамблирование моделей. Например, в случае двух моделей, финальная похожесть двух запросов $sim(q_i, q_j) = \alpha \cdot a_1(q_i, q_j) + \beta \cdot a_2(q_i, q_j)$, где $a_1, a_2$ – ансамблируемые модели, а $\alpha, \beta$ – веса, которые можно подобрать на валидационном множестве. Так как модели выдают похожесть в разных масштабах, предсказания моделей следует отнормировать.

Ранее в работе обсуждалось два способа генерации негативных примеров:

- random sampling
- hard negative mining

Так как сложность предсказания GBM и DSSM намного выше сложности предсказания линейной модели, а hard negative mining крайне затратная процедура, то hard negative mining использовался только для линейных моделей. Остальные модели обучались только с помощью random sampling. В таблице 3 показано качество лучшего варианта.

В режиме hard negative mining для каждой пары похожих запросов как негативный пример бралась случайная пара запросов, предсказание модели на которой было больше, чем для данной позитивной пары.

На рисунках 5, 6, 7, 8 показана зависимость качества линейных моделей от числа негативных примеров в случае random sampling и от числа итераций в случае hard negative mining. Из этих графиков видно, что оба метода генерации негативных примеров в итоге показывают примерно одинаковое качество. Однако, стоит отметить, что процесс генерации негативных примеров с помощью hard negative mining является значительно более трудоемким.

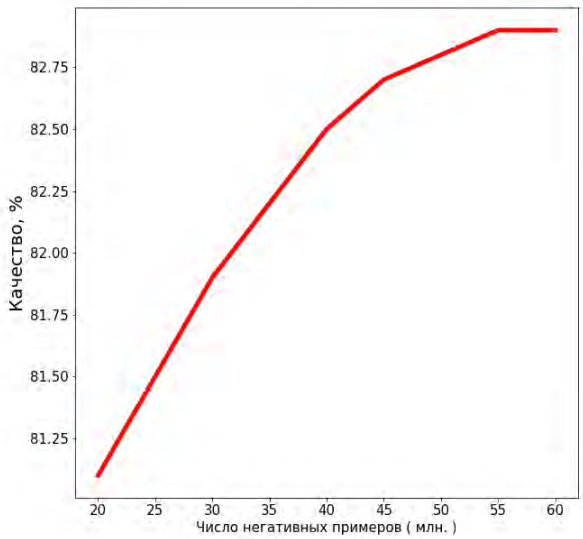


Рис. 5: Качество BM25 модели совпадения от числа негативных примеров.

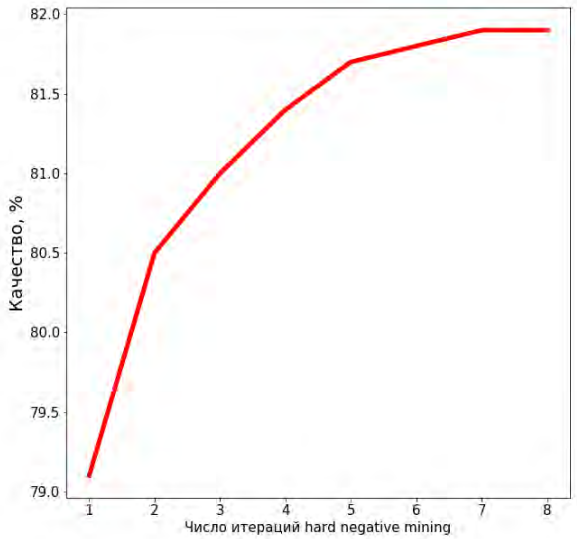


Рис. 6: Качество BM25 модели совпадения от итерации hard negative mining.

На рисунках 9, 10 показана зависимость качества GBM модели в зависимости от числа итераций в случаях моделей совпадения и несовпадения соответственно. Из этих графиков видно, что качество модели совпадения на 10 тысячах итераций выходит примерно на константу, когда качество модели несовпадения продолжает расти. Это может быть связано с тем фактом, что в среднем число несовпадений слов в двух запросах из обучающего множества

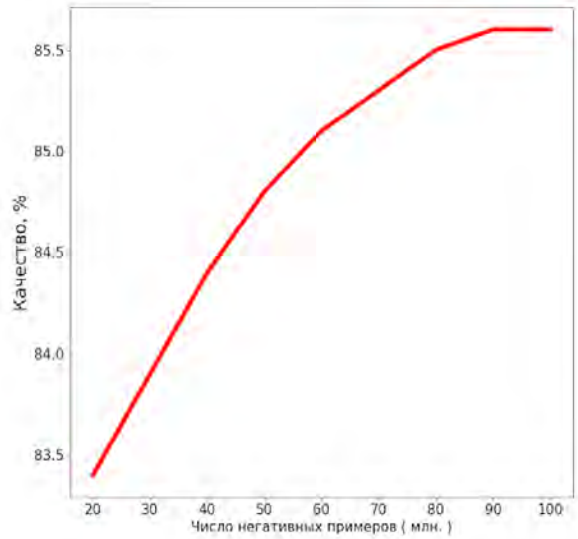


Рис. 7: Качество BM25 модели несовпадения от числа негативных примеров.

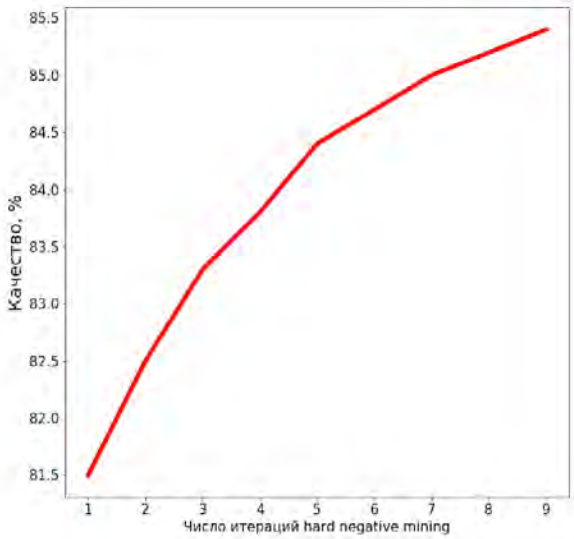


Рис. 8: Качество BM25 модели несовпадения от итерации hard negative mining.

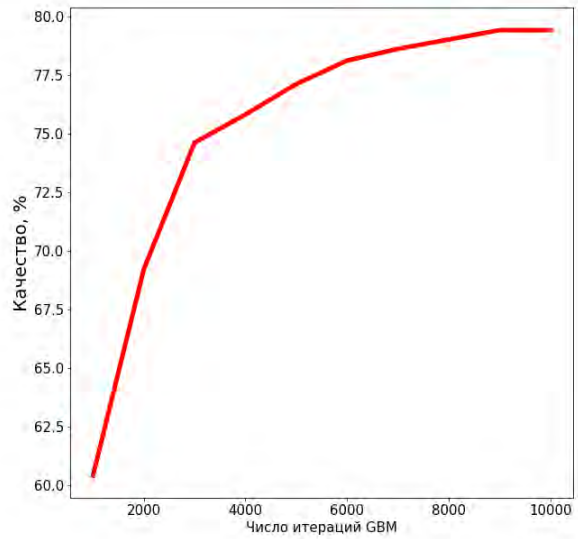


Рис. 9: Качество GBM модели совпадения от числа итераций GBM.

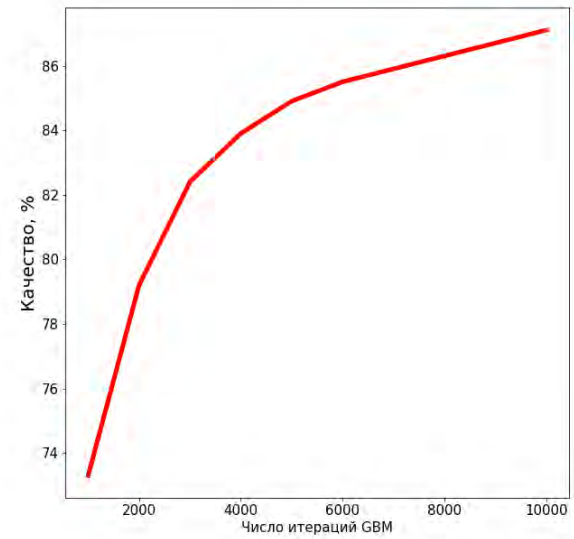


Рис. 10: Качество GBM модели несовпадения от числа итераций GBM.

выше, чем число совпадений слов. Стоит отметить, что качество модели несовпадения значительно выше. Если бы не заранее поставленный лимит в 10 тысяч итераций, то качество продолжило бы незначительно расти. Данный лимит был поставлен исключительно из-за времени обучения моделей. Например, GBM модель несовпадения на обучающем множестве

из 125 миллионов объектов обучалась на 10 тысяч итераций около суток. Для сравнения, линейная модель несовпадения на таком же объеме данных обучается порядка часа.

Стоит отметить, что, хотя GBM модель несовпадения значительно превосходит линейную модель по качеству, она значительно увеличивает скорость предсказания финального ансамбля. Ансамбль из моделей «BM25 модель совпадения + BM25 модель несовпадения + SMMC» хоть и проиграет по итоговому качеству ансамблю «BM25 модель совпадения + GBM модель несовпадения + SMMC», но выполняет предсказание на тестовом наборе данных примерно в 100 раз быстрее. Если в задаче нахождения похожих запросов важна скорость работы, то стоит рассматривать такой ансамбль, хоть и итоговое качество будет немного хуже.

## 5.2 Сглаживание поведенческой информации

Оценим, как влияет сглаживание на качество финальной модели ранжирования. Применим лучший ансамбль из предыдущего пункта для сглаживания поведенческого признака в задаче ранжирования.

Обучающее множество представляло из себя 50 тысяч пар $(q, d)$, для каждой пары известна асессорская оценка – релевантен данный документ к запросу или нет. Для этих пар было извлечено 10 текстовых признаков, которые свидетельствовали о текстовой похожести запроса и документа и один простейший поведенческий признак – число кликов в данный документ по данному запросу, деленное на число показов. Будем решать эту задачу как задачу бинарной классификации. Как модель классификации возьмем модель GBM.

Как метрику качества классификации будем использовать ROC-AUC на пятикратной кросс-валидации. Эта метрика качества равна вероятности, что для двух случайно выбранных объектов $x_1$ из класса 1 и $x_0$ из класса 0, вероятность принадлежности к классу 1 у объекта $x_1$ больше, чем у $x_0$.

В качестве сглаживания попробуем несколько различных вариантов. Во-первых, вариант предлагаемый в работе по формуле (12). Далее будем называть его Mean weighted sim. В качестве более простой альтернативы, будем просто переносить поведенческий признак у самого близкого по похожести запроса. Этот метод назовем Max sim.

Сглаживание будем применять, только если поведенческий признак у данной пары запрос документ отсутствует. В другом варианте будем применять сглаживание всегда, но сглаженный признак будем вставлять не вместо оригинального, а отдельным признаком.

Результаты сглаживания представлены в таблице 4. Из нее видно, что при любой технике сглаживания более выгодно использовать сглаженный признак в качестве отдельного признака, а не заменять исходный. Можно сделать вывод, что сглаживание через взвешенное усреднение по формуле (12) работает лучше, чем просто перенос поведенческого признака с самого похожего запроса.

Таблица 4: Качество сглаживания

| Вариант сглаживания | AUC |
|---|---|
| Без сглаживания | 0.882 |
| Max sim | 0.890 |
| Max sim, новый признак | 0.902 |
| Mean weighted sim | 0.901 |
| Mean weighted sim, новый признак | 0.906 |

# 6 Заключение

В ходе данной работы было проделано следующее:

- Предложен метод сглаживания поведенческой информации, основанный на текстовой похожести запросов
- Предложен метод нахождения похожих запросов через клики
- Предложен метод оценки похожести запросов через ансамбль трех моделей: модели совпадения слов, модели несовпадения слов и семантической модели
- Проведено исследование двух методов генерации негативных примеров: random sampling и hard negative mining
- Проведено сравнение предложенных моделей с базовыми моделями задачи ранжирования
- Проведен эксперимент со сглаживанием поведенческого признака перед обучением финальной модели ранжирования

На основе результатов проделанной работы были сделаны следующие выводы:

- Задачу оценки похожести запросов можно разделить на три независимых модели: модель совпадения слов, модель несовпадения слов, семантическую модель. Вместе они показывают значительно более высокое качество, чем каждая из них по отдельности.
- Веса в вероятностных моделях, таких как BM25, можно обучать линейными моделями, и это дает значительный прирост в качестве
- Линейная модель совпадения дает значительно более высокое качество, чем GBM, так как напрямую обучает вероятностную модель
- Для линейных моделей оба метода генерации негативных примеров работают примерно одинаково, однако hard negative mining работает значительно дольше

- В качестве семантической модели лучше всего обучить классификатор на выученных семантических разложениях
- С помощью взвешенного усреднения поведенческих признаков можно получить признак, который значительно улучшит качество финальной модели ранжирования

На защиту в данной работе выносятся следующие результаты:

1. Метод сглаживания поведенческой информации через модель похожести запросов
2. Построение модели похожести через три независимых модели: модель совпадения слов, модель несовпадения слов и семантическую модель
3. Эксперименты, сравнивающие предложенную модель похожести запросов с классическими моделями информационного поиска

Работа была представлена на научной конференции «Ломоносов – 2018» [27].

# Список литературы


[1] *Manning C., Raghavan P., Schütze H.* Introduction to Information Retrieval // Cambridge University Press New York, New York, USA, 2008.

[2] *Yue Y., Joachims T.* Interactively optimizing information retrieval systems as a dueling bandits problem // Proceedings of the 26th Annual International Conference on Machine Learning, Montreal, Canada, 2009, P. 1201-1208.

[3] *Hofmann K., Schuth A., Whiteson S., Rijke M.* Reusing historical interaction data for faster online learning to rank for IR // Proceedings of the sixth ACM international conference on Web search and data mining, Rome, Italy, 2013, P. 183-192.

[4] *Chuklin A., Markov I., Rijke M.* Click Models for Web Search. Morgan & Claypool, 2015.

[5] *Dupret G., Piwowarski B.* A user browsing model to predict search engine click data from past observations // Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval, Singapore, Singapore, 2008, P. 331–338.

[6] *Chapelle O., Zhang Y.* A dynamic bayesian network click model for web search ranking // Proceedings of the 18th international conference on World wide web, Madrid, Spain, 2009, P. 1–10.

[7] *Li H.* A Short Introduction to Learning to Rank // IEICE Transactions on Information and Systems, 2011, Vol. 94-D, P. 1854–1862.

[8] *Joachims T.* Optimizing search engines using clickthrough data // Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, Edmonton, Canada, 2002, P. 133–142.

[9] *Bilenko M., White R.* Mining the search trails of surfing crowds: identifying relevant websites from user activity // Proceedings of the 17th international conference on World Wide Web, Beijing, China, 2008, P. 51–60.

[10] *Gao J., Yuan W., Li X., Deng K., Nie J-Y.* Smoothing clickthrough data for web search ranking // Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval, Boston, USA, 2009, P. 355–362.

[11] *Yi X., Allan J.* Discovering Missing Click-through Query Language Information for Web Search // Proceedings of the 20th ACM international conference on Information and knowledge management, Glasgow, Scotland, 2011, P. 153–162.

[12] *Wu W., Li H.* Learning query and document similarities from click-through bipartite graph with metadata // Proceedings of the sixth ACM international conference on Web search and data mining, Rome, Italy, 2013, P. 687–696.

[13] *Bona F., Riezler S., Hall K.* Learning dense models of query similarity from user click logs // The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Los Angeles, USA, 2010, P. 474–482.

[14] *Zhou K., Li X., Zha H.* Collaborative ranking: improving the relevance for tail queries // Proceedings of the 21st ACM international conference on Information and knowledge management, Maui, USA, 2012, P. 1900–1904.

[15] *Robertson S., Zaragoza H.* The Probabilistic Relevance Framework: BM25 and Beyond // Journal Foundations and Trends in Information Retrieval, 2009, Vol. 3, P. 333–389.

[16] *Robertson S., Walker S.* Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. // Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieva, New York, USA, 1994, P. 232–241.

[17] *Milkov T.* Distributed Representations of Words and Phrases and their Compositionality // arXiv preprint arXiv:1310.4546, 2013.

[18] *Deerwester S., Dumais S., Landauer T., Furnas G., Harshman A.* Indexing by latent semantic analysis // Journal of the American Society for Information Science, 1990, Vol. 41, P. 391–407.

[19] *H.Friedman J.* Greedy Function Approximation: A Gradient Boosting Machine // The Annals of Statistics , 2001, Vol. 29, P. 1189–1232.

[20] *Dean J., Ghemawat S.* MapReduce: simplified data processing on large clusters // Magazine Communications of the ACM, 2008, Vol. 51, P. 107–113.

[21] *Huang P., He X., Gao J., Deng L., Acero A., Heck L.* Learning deep structured semantic models for web search using clickthrough data // Proceedings of the 22nd ACM international conference on Information & Knowledge Management, San Francisco, USA, 2013, P. 2333–2338.

[22] *Shen Y., He X., Gao J., Deng L., Mesnil G.* Learning semantic representations using convolutional neural networks for web search // Proceedings of the 23rd International Conference on World Wide Web, Seou, Korea, 2014, P. 373–374.

[23] *Pang L., Lan Y., Guo J., Xu J., Wan S., Cheng X.* Text matching as image recognition // Proceeding AAAI'16 Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, Phoenix, USA, 2016, P. 2793–2799.

[24] *Hu B., Lu Z., Li H., Chen Q.* Convolutional neural network architectures for matching natural language sentences // Proceeding NIPS'14 Proceedings of the 27th International Conference on Neural Information Processing Systems, Montreal, Canada, 2014, P. 2042–2050.

[25] *Mitra B., Diaz F., Craswell N.* Learning to Match using Local and Distributed Representations of Text for Web Search // Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 2017, P. 1291–1299.

[26] Our latest quality improvements for Search. `https://blog.google/products/search/our-latest-quality-improvements-search/`

[27] *Викулин В.А.* Ранжирование с помощью поведенческой информации // Сборник тезисов XXV международной научной конференции Ломоносов–2018. Издательский отдел факультета вычислительной математики и кибернетики МГУ имени М.В. Ломоносова, 2018, С. 102–103.