# Порождение структурно простых ранжирующих функций для задач информационного поиска

Кулунчаков Андрей

Московский физико-технический институт
Факультет управления и прикладной математики
Кафедра интеллектуальных систем

Научный руководитель д.ф.-м.н. В. В. Стрижов

Москва,
2017 г.

# Задача

## Цель

Предложить автоматическую процедуру порождения ранжирующих функций произвольной структуры, имеющих высокий MAP.

## Проблема

Переборный алгоритм порождает только очень простые функции. Алгоритмы генетического поиска дают сложные модели, быстро стагнируя в локальные минимумы.

## Предлагаемое решение

Замедлять стагнацию с помощью регуляризации. Выводить из стагнации, определяя момент ее начала с помощью структурных метрик на множестве моделей.

# Прикладная задача

## Значимость

Предлагаемый подход предназначен для улучшения систем информационного поиска, основанных на экспертных оценках релевантности документа запросам.

## Коллекции документов

Следуя традициям сообщества ИП, мы ставим своей целью построение ранжирующих функций, дающих высокий MAP на коллекциях TREC.

## Актуальность

Постоянное развитие TREC-сообщества, программных пакетов, связанных в т.ч. с ранжирующими функциями (напр. Terrier) демонстрирует актуальность поставленной задачи.

# Известные подходы к порождению функций

**Goswami et al. Exploring the space of ir functions, 2014**

Переборный алгоритм вычислительно сложен. Аналиризуются только функции с очень низкой структурной сложностью.

**Fan et al. A generic ranking function discovery framework by genetic programming for information retrieval, 2004.**

Генетический алгоритм порождения застревает в локальных минимумах, итоговые функции переусложнены.

**Billhardt P. et al. Using genetic algorithms to find suboptimal retrieval expert combinations, 2002.**

Генетический алгоритм поиска линейной комбинации ранжирующих функций возвращает параметры, существенно зависящие от коллекции.

# Постановка задачи

## Задача

Коллекция документов $C$ экспертно отранжирована по релевантности запросам $Q$ — $g : \bar{C} \times Q \to \{0, 1\}$. Необходимо аппроксимировать зависимость $g$ некоторой явной функцией.

## Постановка задачи

Оптимизационная задача:

$$f^* = \text{argmax}_f \big(\mathcal{F}(f) - \text{R}(f)\big),$$

где $R$ — регуляризатор, функционал качества $\mathcal{F}$ — MAP, а максимизация проводится по всем суперпозициям над порождающими $G$.

# Аргументы функции и критерии качества

## Аргументы ранжирующей функции

Переменные в $G$ — признаки пары документа и слова $(d, w)$ :

$$x_w^d = t_d^w \log\left(1 + \frac{l_{avg}}{l_d}\right), \quad y_w = \frac{N_w}{N},$$

где $N_w$ — # документов, содержащих $w$; $t_d^w$ — число слов $w$ в $d$, $l_d$ — длина $d$, $l_{avg}$ — средняя длина документа в коллекции. При этом значение функции $f$ на паре $(q, d)$ определяется:

$$f(q, d) = \sum_{w \in q} f(w, d) = \sum_{w \in q} f(x_w^d, y_w)$$

## Критерии качества

Идеальная ранжирующая модель $h$ такова, что для лучших моделей $\{f_i\}$ из (Goswami 2014) выполняется: $\mathcal{F}(h) > \mathcal{F}(f_i)$ — модель $h$ равномерно лучше, чем все эталонные функции $\{f_i\}$.

# Алгоритм порождения ранжирующих функций

1. Создаётся начальное множество случайных моделей.
2. Часть моделей скрещиваются — обмениваются случайными подмоделями:
   $f = \sin(x) + \mathbf{cos(xy)},\ g = \cos(x) + \mathbf{(x+y)} \rightarrow$
   $\rightarrow f = \sin(x) + \mathbf{(x+y)},\ g = \cos(x) + \mathbf{cos(xy)}.$
3. Некоторые модели мутируют — часть модели заменяется на случайную функцию:
   $f(x, y) = \sin(x) + \mathbf{cos(xy)} \rightarrow f = \sin(x) + \mathbf{ln(y)}.$
4. Определяется, попал ли алгоритм в локальный минимум. В случае попадания удаляем часть моделей и добавляем множество случайных.
5. Выбирается некоторое число лучших моделей согласно функционалу качества $\big(\mathcal{F} - \mathrm{R}\big)(f)$. Если требуемая точность достигнута, алгоритм останавливается. Иначе, возврат на 2 шаг.

# Определение попадания в локальный минимум

Имеется множество моделей $P = \{f_j\}_{j=1}^{|P|}$ и метрики $\mu(f_i, f_j)$. Значение $\mu$ на всем множестве $P$ определяется как:

$$\mu(P) = \frac{\sum_{k<j} \mu_i(f_k, f_j)}{avlen(P)},$$

где средняя длина моделей в множестве $P$:

$$avlen(P) = \frac{1}{|P|} \sum_{j=1}^{|P|} |f_j|.$$

**Попадание в локальный минимум** определяется, как опускание значения $\mu(P)$ ниже порога Thresh, который определяется эмпирически.

# Три метрики $\mu_1, \mu_2, \mu_3$

Суперпозиции $f_i$, $f_j$ представляются в виде помеченных деревьев $T_i$ и $T_j$.

**$\mu_1$**

$$\mu_1(f_i, f_j) = |T_i| + |T_j| - 2|T_{ij}|,$$

где $T_{ij}$ — наибольший общий подграф деревьев $T_i$ и $T_j$.

**$\mu_2$**

Вторая метрика $\mu_2(f_i, f_j)$ — редакторское расстояние между строковыми представлениями деревьев $T_i$ и $T_j$.

**$\mu_3$**

Третья метрика $\mu_3(f_i, f_j)$ — редакторское расстояние между самими деревьями $T_i$ и $T_j$.

# Регуляризаторы

Для контроля структурной сложности модели в функционале качества присутствует регуляризатор. Исследуются три варианта ($m$ — значение MAP(f)):

1. $R_1(f) = p \cdot m \cdot I(|f| < CT)$,
   где $CT$ — пороговая сложность модели, $p \in (0, 1)$.
2. $R_2(f) = p \cdot m \cdot I(|f| \geq CT) \cdot (|f| - CT)$,
   где $C > 0$ — некоторая константа.
3. $R_3(f) = p \cdot m \cdot |f|^* \cdot \log(|f| + 1)$,
   где $|f|^*$ — количество переменных $x$, $y$ в модели.

# Вычислительный эксперимент

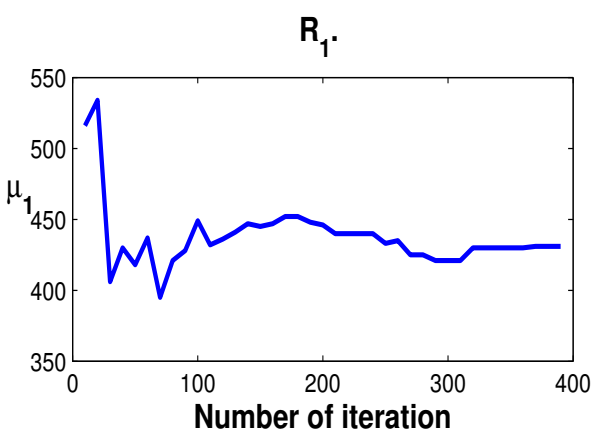
## Выборки

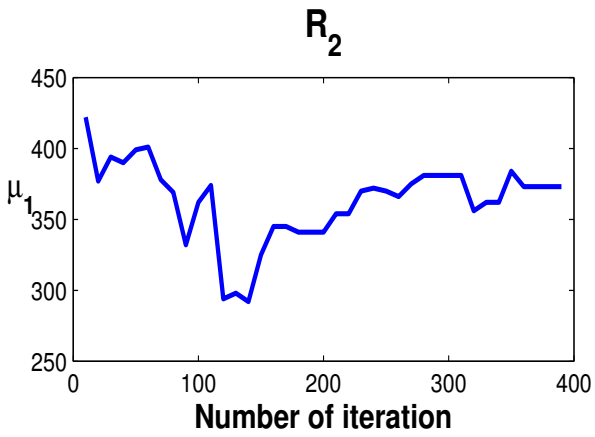
Используются выборки Trec5-8 для анализа моделей, генерируемых генетическим алгоритмом. При этом Trec7 — обучающая, а остальные — контрольные. Все выборки имеют примерно по 500 000 документов и 50 запросов к ним.

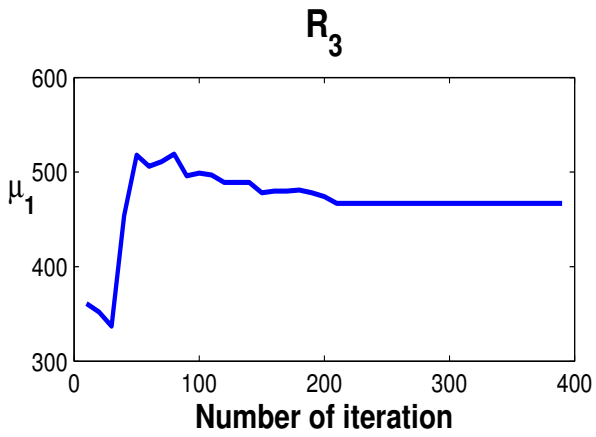
## Вычислительный эксперимент

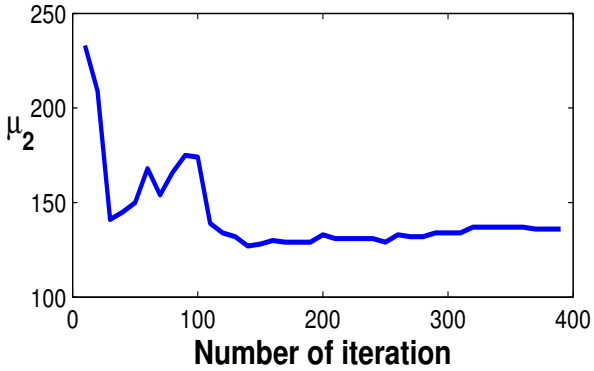
Запускается алгоритм при разных регуляризаторах на выборке Trec7. Выбирается регуляризатор, который наиболее отдаляет точку стагнации и при этом не позволяет моделям значительно переусложняться. Выберается метрика, которая наиболее точно определяет начало стагнации.
После этого запускается алгоритм снова и итоговые модели сравниваются с теми, что были отобраны в Goswami, 2014.

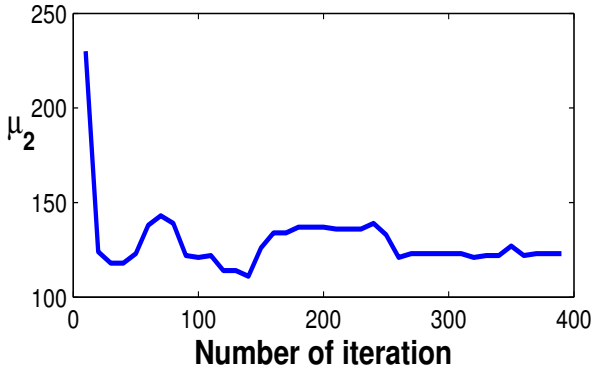
# Анализ метрик и регуляризаторов

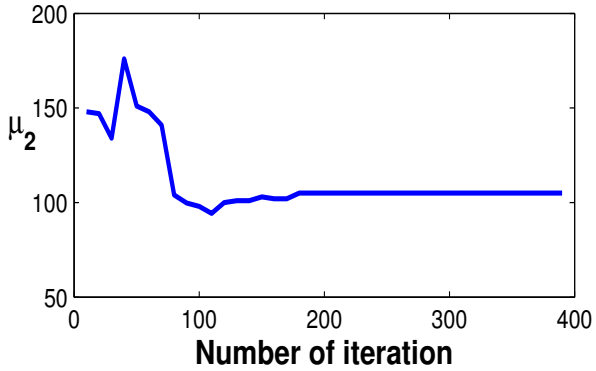


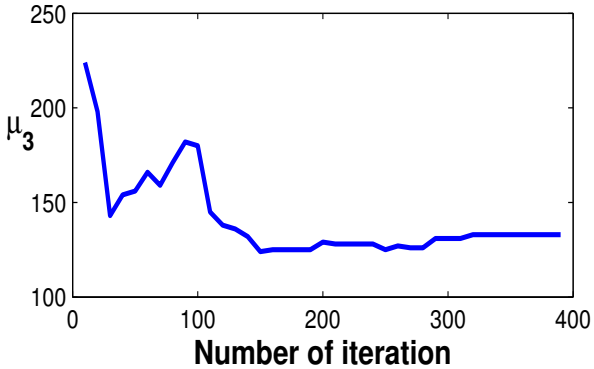



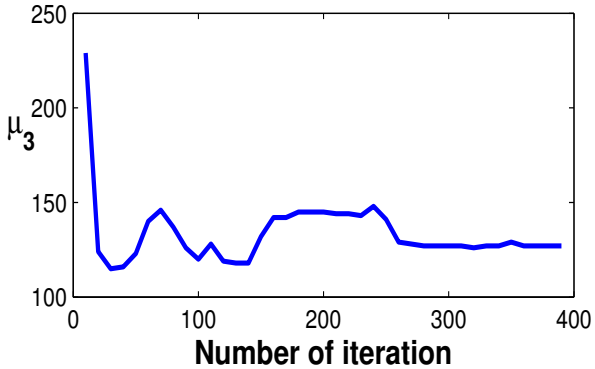



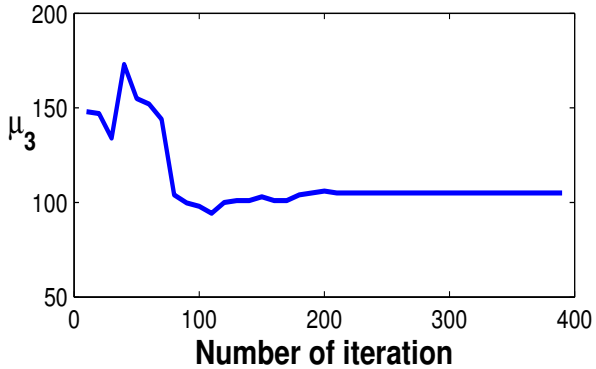



# Анализ метрик и регуляризаторов

1. Метрика $\mu_1$ не отражает момент стагнации — на графике нет заметных минимумов и последующих прямолинейных участков.
2. Метрики $\mu_2$ и $\mu_3$ фактически неотличимы по динамике. Выберается $\mu_2$, как более эффективно вычислимое.
3. Регуляризатор $R_1$ слишком жесткий — алгоритм стагнирует уже после первых итераций.
4. Похожая ситуация наблюдается и для $R_2$ — метрики существенно уменьшаются уже на первых итерациях.
5. Напротив, с регуляризатором $R_3$ уменьшение более плавное. Далее используем **третий регуляризатор**.

# Итоговые функции на Trec7($\mu_2, R_3$)

После 600 итераций отобраны функции $\{h_j\}$ ($\ln(x) \to \ln(x+1)$, а $g(x) = \ln\ln(x)$). Модели $\{h_j\}$ сравниваются с функциями $\{f_i\}$, признанными лучшими в работе Goswami,2014. Функции $\{f_i\}$ в среднем на множестве коллекций лучше, чем все модели сложностью не более 8 и известные ранжирующие модели $BM25, LGD, LM_{DIR}$:

| # | Эталонные функции | # | Итоговые функции |
|---|---|---|---|
| $f_1$ | $e^{\sqrt{\ln(x/y)}}$ | $h_1$ | $g\left(\frac{g(x)}{\sqrt{\ln(x)+x}}\right) - \ln(y)$ |
| $f_2$ | $\sqrt{\frac{\ln(x)}{\sqrt{y}}}$ | $h_2$ | $g\left(\frac{g(x)}{\sqrt{\frac{1}{2}\ln(x)+x}}\right) - \ln(y)$ |
| $f_3$ | $\sqrt[4]{\frac{x}{y}}$ | $h_3$ | $g\left(\ln\left(\frac{g(x)}{\sqrt{\frac{1}{2}\ln(x)+x}}\right) - \ln(y)\right)$ |
| $f_4$ | $\sqrt{y+\sqrt{\frac{x}{y}}}$ | $h_4$ | $g\left(\frac{g(x)}{\sqrt{g(\sqrt{x})+x}}\right) - \ln(y)$ |
| $f_5$ | $\sqrt{\sqrt{\frac{x}{y}}\cdot e^{-y}}$ | $h_5$ | $g\left(\frac{g(x)}{\sqrt{\ln(x)+\ln(y)}}\right) - \ln(y)$ |
| $f_6$ | $\sqrt{\sqrt{x}+\sqrt{\frac{x}{y}}}$ | $h_6$ | $g\left(\frac{g(\ln(x))}{\sqrt{\ln(x)+x}}\right) - \ln(y)$ |

# Сравнение функций на разных выборках

| Значение 50·MAP для функций $f_i$ и $h_j$ | | | | |
|---|---|---|---|---|
| Функция | Trec5 | Trec6 | Trec7 | Trec8 |
| $f_1$ | 8.79 | 13.71 | 10.04 | 13.9 |
| $f_2$ | 8.52 | 13.00 | 9.22 | 13.07 |
| $f_3$ | 8.91 | 13.62 | 9.91 | 13.71 |
| $f_4$ | 8.91 | 13.62 | 9.91 | 13.71 |
| $f_5$ | 8.91 | 13.62 | 9.91 | 13.71 |
| $f_6$ | 8.87 | 13.61 | 9.89 | 13.70 |
| $h_1$ | 8.97 | 13.69 | 10.60 | 14.40 |
| $h_2$ | 9.47 | 13.72 | 10.65 | 14.40 |
| $h_3$ | 9.56 | 13.79 | 10.63 | 14.38 |
| $h_4$ | 9.23 | 13.71 | 10.50 | 14.37 |
| $h_5$ | 8.86 | 13.39 | 10.44 | 14.36 |
| $h_6$ | 8.10 | 13.48 | 10.42 | 14.36 |

# Выводы

Полученные функции заметно улучшают известные модели из Goswami, 2014. Модели $h_{1,2,3,4}$ лучше их даже на всех тестовых коллекциях.

Хотя структурная сложность моделей $h_j$ примерно в два раза выше, чем у функций $f_i$, $h_j$ имеют компактный вид.

| Структурная сложность $f_i$ и $h_j$ | | | |
|---|---|---|---|
| Функция | Сложность | Функция | Сложность |
| $f_1$ | 6 | $h_1$ | 14 |
| $f_2$ | 6 | $h_2$ | 15 |
| $f_3$ | 5 | $h_3$ | 16 |
| $f_4$ | 7 | $h_4$ | 16 |
| $f_5$ | 8 | $h_5$ | 15 |
| $f_6$ | 8 | $h_6$ | 15 |

# Заключение

1. Приведен анализ регуляризаторов и структурных метрик в задаче порождения ранжирующих функций.
2. Разработан генетический алгоритм поиска ранжирующих функций, который не склонен к попаданию в локальные минимумы.
3. Порождаемые модели не переусложнены и имеют компактное описание.
4. Получаемые ранжирующие функции равномерно лучше структурно простых эталонных моделей на рассматриваемом множестве тестовых коллекций TREC.

# Список работ, подготовленных в магистратуре

Публикации:

- Kulunchakov A. S., Strijov V. V., Generation of simple structured Information Retrieval functions by genetic algorithm without stagnation , Expert Systems With Applications, 2017,
- Kulunchakov A. S., Creation of parametric rules to rewrite algebraic expressions in Symbolic Regression, JMLDA, 2017

Подготовлены:

- Kulunchakov A. S., Strijov V. V., Isomorphism-based simplification of forecasting models
- Kulunchakov A. S., Strijov V. V., Generation of structural features from time series in problems of classification and clustering
- Kulunchakov A., Mairal J., Juditsky A., Harchaoui Z., Regularization and averaging schemes for stochastic least-squares with drifting optimum