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

Федоряка Дмитрий Сергеевич

Технология интерактивной визуализации тематических моделей

010900 — Прикладные математика и физика

Выпускная квалификационная работа бакалавра

Научный руководитель:
Профессор РАН, д. ф.-м. н.
Воронцов Константин Вячеславович

Москва
2017

Содержание

Аннотация

В данной работе исследуются методы визуализации тематических моделей коллекций текстовых документов. Рассматривается несколько способов визуализации тематических моделей, в том числе темпоральных и иерархических. Ставится задача построения спектра тематической модели. Предлагается несколько алгоритмов её решения и разрабатывается методика оценки их качества. Задача обобщается на случай иерархических тематических моделей. Описывается созданная информационная система VisARTM для автоматического создания и визуализации тематических моделей.

Ключевые слова: *тематическое моделирование, визуализация.*

1 Введение

Актуальность темы. Тематическое моделирование является важным инструментом статистического анализа текстовых коллекций. Наглядное представление тематической модели позволяет лучше изучить кластерную структуру коллекции и оценить качество тематической модели. На сегодняшний день создано много средств визуализации [1]. Тем не менее, есть задачи визуализации, не решённые ранее. Одна из них — построение тематического спектра — такого линейного упорядочивания тем, при котором близкие по смыслу темы находятся рядом.

Цели работы.

- Создание информационной системы с графическим веб-интерфейсом для визуализации тематических моделей;
- разработка алгоритмов построения тематического спектра;
- разработка методов оценивания качества тематического спектра.

Методы исследований. При разработке методов визуализации тематических моделей использовались элементы теории вероятностей, математической статистики и линейной алгебры. При решении задачи построения тематического спектра использовались также методы комбинаторной оптимизации. Для программной реализации разработанных алгоритмов использовались языки программирования python, C и javascript.

Научная новизна. Разработаны эффективные алгоритмы для построения тематического спектра для плоских и иерархических тематических моделей. Разработаны методы оценивания качества тематического спектра.

Практическая ценность. Разработана информационная система для автоматического построения и визуализации тематических моделей коллекций текстовых документов.

Основные положения, выносимые на защиту.

1. Разработаны алгоритмы построения тематического спектра.
2. Предложены методы оценивания качества тематического спектра.
3. Создана информационная система для визуализации тематических моделей.

Работа организована следующим образом. В главе 2 ставится задача вероятностного тематического моделирования и кратко описывается подход к её решению с помощью APTM. В главе 3 обсуждается визуализация тематических моделей и описывается информационная система VisARTM. В главе 4 описывается постановка и решение задачи построения тематического спектра. В главе 5 эта задача обобщается на случай иерархических тематических моделей.

2 Вероятностное тематическое моделирование

Тематическое моделирование — это одно из современных направлений статистического анализа текстов, активно развивающееся с конца 90-х годов. Приложения тематических моделей — информационный поиск, выявление трендов в новостных потоках или научных публикациях, анализ социальных сетей, классификация и категоризация документов, тематическая сегментация текстов, тегирование веб-страниц, обнаружение спама, рекомендательные системы [2].

Вероятностная тематическая модель (probabilistic topic model, PTM) коллекции текстовых документов описывает каждую тему дискретным распределением на множестве терминов, каждый документ — дискретным распределением на множестве тем.

2.1 Постановка задачи тематического моделирования

Пусть D — множество (коллекция) текстовых документов, W — множество (словарь) всех употребляемых в них терминов (слов или словосочетаний). Каждый документ — это последовательность n_d терминов $w_1, \ldots, w_{n_d}$ из словаря W.

Вероятностное пространство. Предполагается, что существует конечное множество тем T, и каждое вхождение термина w в документ d связано с некоторой темой $t \in T$. Коллекция документов рассматривается как случайная и независимая выборка троек $(w_i, d_i, t_i), i \in \overline{1,n}$ из дискретного распределения $p(w,d,t)$ на конечном вероятностном пространстве $W \times D \times T$. Термины w и документы d являются наблюдаемыми переменными, тема $t \in T$ является скрытой переменной.

Гипотезы. Принимается *гипотеза мешка слов* — предположение о том, что порядок слов в документе не имеет значения. Также принимается *гипотеза условной независимости* — предположение о том, что появление слова в документе, связанное с темой t зависит только от темы t, но не зависит от самого документа: $p(w|d,t) = p(w|t)$.

Задача матричного разложения. Согласно формуле полной вероятности и гипотезе условной независимости:

$$p(w|d) = \sum_{t \in T} p(w|t)p(t|d). \tag{2.1}$$

Рассмотрим матрицы $F \in \mathbb{R}^{|W| \times |D|}$, $\Phi \in \mathbb{R}^{|W| \times |T|}$ и $\Theta \in \mathbb{R}^{|T| \times |D|}$ такие, что

$$F_{wd} = p(w|d); \quad \varphi_{wt} = p(w|t); \quad \theta_{td} = p(t|d). \tag{2.2}$$

Тогда имеет место матричное равенство:

$$F = \Phi\Theta. \tag{2.3}$$

Матрица F оценивается частотами:

$$F_{wd} = \frac{n_{dw}}{n_d}, \tag{2.4}$$

где n_{dw} — число вхождений термина w в документ d, n_d — длина документа d.

Итак, задача вероятностного тематического моделирования — это задача матричного разложения (2.3), т.е. аппроксимации матрицы F, определённой в (2.4) произведением двух стохастических матриц Θ и Φ.

Тематика. Введём понятие *тематики*. Тематика объекта x — это условное вероятностное распределение распределение $p(t|x)$, где $t \in T$. В частности, тематика документа — это столбец матрицы Θ, а тематика термина — это строка матрицы Φ (с точностью до перенормировки по формуле Байеса: $p(t|w) = \varphi_{wt}\frac{p(t)}{p(w)}$). Далее понятие тематики будет применяться и к другим объектам (например, к паре термин-документ).

Задачу (2.3) можно рассматривать как задачу одновременного поиска тематик всех документов и терминов коллекции.

2.2 Решение задачи тематического моделирования

Задача оценивания параметров Φ, Θ сводится к максимизации правдоподобия выборки:

$$p(D;\Phi;\Theta) = \prod_{i=1}^{n} p(d_i, w_i) = \prod_{d\in D}\prod_{w\in d} p(w|d)^{n_{dw}} p(d)^{n_{dw}} \to \max_{\Phi,\Theta}. \tag{2.5}$$

Прологарифмировав правдоподобие, получим постановку задачи *вероятностного латентного семантического анализа* (probabilistic latent semantic analysis, PLSA) [3]:

$$\begin{cases} L(\varphi,\Theta) = \sum\limits_{d\in D}\sum\limits_{w\in d} n_{dw} \ln \sum\limits_{t\in T} \varphi_{wt}\theta_{td} \to \max\limits_{\Phi,\Theta}; \\ \sum\limits_{w\in W} \varphi_{wt} = 1; \quad \varphi_{wt} \geq 0; \\ \sum\limits_{t\in T} \theta_{td} = 1; \quad \theta_{td} \geq 0. \end{cases} \tag{2.6}$$

Эту задачу можно решить с помощью EM-алгоритма [4, 5]. На E-шаге нужно вычислять значение условных вероятностей:

$$p_{tdw} = \frac{\varphi_{wt}\theta_{td}}{\sum\limits_{s\in T} \varphi_{ws}\theta_{sd}}. \tag{2.7}$$

На M-шаге нужно вычислять оценки максимального правдоподобия:

$$\varphi_{wt} = \frac{n_{wt}}{\sum\limits_{v\in W} n_{vt}}; \quad n_{wt} = \sum_{d\in D} n_{dw} p_{tdw}; \tag{2.8}$$

$$\theta_{td} = \frac{n_{td}}{\sum\limits_{s\in T} n_{sd}}; \quad n_{sd} = \sum_{w\in d} n_{dw} p_{tdw}. \tag{2.9}$$

2.3 Аддитивная регуляризация

Задача (2.3) в общем случае имеет бесконечное множество решений. Действительно, если $F = \Phi\Theta$, то $F = (\Phi S)(S^{-1}\Theta)$ для всех невырожденных S. Чтобы сделать решение этой задачи единственным, было предложено [6] к логарифму правдоподобия добавить *регуляризатор* — функцию $R(\Theta, \Phi)$. В общем случае можно взять несколько таких функций и прибавить их линейную комбинацию:

$$L(\Phi,\Theta)+\sum_{i=1}^{k}\tau_i R_i(\Phi,\Theta)\to\max_{\Phi,\Theta} \tag{2.10}$$

Такой подход называется аддитивной регуляризацией тематических моделей (APTM). Он не только формально делает постановку задачи корректной, но и позволяет учитывать дополнительные предположения о порождающей модели. В частности, если в качестве регуляризатора взять функцию

$$R(\Phi,\Theta)=\beta_0\sum_{t,w}(\beta_w-1)\ln\varphi_{wt}+\alpha_0\sum_{d,t}(\alpha_t-1)\ln\theta_{td}, \tag{2.11}$$

то задача будет эквивалентной задаче *латентного размещения Дирихле* (Latent Dirichlet Allocation, LDA) [7], которая основана на предположении, что столбцы матриц Φ и Θ являются случайными векторами, порождаемыми распределениями Дирихле.

В [2], кроме регуляризатора (2.11) предлагаются также регуляризаторы для разреживания, частичного обучения, выделения предметных и фоновых тем, декоррелирования, отбора тем и повышения когерентности тем.

2.4 Некоторые обобщения

Мультимодальные ТМ. *Модальности* — это специальные типы терминов (кроме слов и словосочетаний, это могут быть авторы, теги, рубрики, источники, моменты времени и т.д.). Формально, в мультимодальной модели рассматривается несколько словарей, и оптимизируется линейная комбинация их правдоподобий (2.6).

Иерархические ТМ. Иерархическая ТМ состоит из нескольких *уровней*, каждый из которых является обычной ТМ, и матриц условных вероятностей, описывающих иерархические отношения между уровнями. Обозначим через $T_1,\dots,T_L$ множества тем на уровнях $1,\dots,L$. Тогда вводятся матрицы $\Psi^1,\dots,\Psi^{L-1}$, такие что

$$\psi^{\ell}_{ta}=p(t|a), \tag{2.12}$$

где $t\in T_{\ell+1}, a\in T_\ell$ и $\Psi^\ell\in\mathbb{R}^{|T_{\ell+1}|\times|T_\ell|}$.

Построение иерархической ТМ производится сверху вниз [8]. Первый уровень строится как обычная ТМ. Далее, каждый следующий уровень $\ell+1$ строится вместе с вычислением матрицы Ψ_ℓ. Функция правдоподобия, зависящая от $\Theta^{\ell+1}, \varphi^{\ell+1}$ и Ψ^ℓ прибавляется к оптимизируемому функционалу (2.6) в качестве регуляризатора. Полученная функция правдоподобия оптимизируется EM-алгоритмом.

2.5 BigARTM

BigARTM [9, 10] — это библиотека алгоритмов тематического моделирования с открытым кодом, основанная на подходе APTM. Она позволяет эффективно строить ТМ с использованием различных регуляризаторов. Также она поддерживает мультимодальные и иерархические ТМ.

Все исследования в данной работе проведены с использованием BigARTM, а один из результатов работы — создание пользовательского интерфейса, совместимого с этой библиотекой.

3 Визуализация тематических моделей в VisARTM

Обычно в прикладных задачах ТМ представляется несколькими матрицами большого размера ($|T| \sim 10^2, |W| \sim 10^4 - 10^5, |D| \sim 10^3 - 10^6$). Естественно возникает необходимость визуализировать эти матрицы в удобном для пользователя виде.

На сегодняшний день создано множество средств визуализации тематических моделей, некоторые из которых описаны в [1]. Эти средства различаются по способу представления информации и целям.

Основные цели тематических моделей — это тематический поиск (поиск по данному документу документов схожей тематики) и тематическая навигация — переход пользователя между тематически связанными объектами (документами, темами, терминами).

Для примера рассмотрим систему TMVE [11]. Для каждой темы она выдаёт ранжированные списки терминов и документов. для каждого документа — ранжированный список тем. Также для темы она отображает близкие темы, а для документа — близкие документы.

В данной работе этот подход берётся за основу и дополняется.

3.1 Визуализация тематики документов

Прежде всего, для каждого документа нужно отобразить его тематику. Согласно (2.2), $p(t|d) = \theta_{td}$, то есть каждый столбец матрицы Θ описывает тематику некоторого документа. Тогда $\arg\max\limits_t \theta_{td}$ — это наиболее вероятная тема документа d. Но часто документ принадлежит нескольким темам, поэтому полезно вывести несколько тем с наибольшей вероятностью в виде ранжированного списка.

Согласно модели вероятностного тематического моделирования, вхождение каждого слова в документ связано с определённой темой. Это позволяет объяснит пользователю, почему документ был отнесён к определённым темам. Обозначим $t^*(w,d)$ — наиболее вероятную тему, с которой связано вхождение слова w в документ d. Тогда, используя формулу Байеса,

$$t^*(w,d) = \arg\max_t p(t|d,w) = \arg\max_t \frac{p(w|t,d)p(t|d)}{p(w|d)} = \arg\max_t \varphi_{wt}\theta_{td}. \qquad (3.1)$$

В VisARTM документ визуализируется следующим образом. Отображается его текст. Справа от текста отображается круговая диаграмма, построенная по значениям $p(t|d)$, в которой показано минимальное количество тем с наибольшей вероятностью, сумма $p(t|d)$ для которых не меньше 95%. Каждой теме соответствует какой-то цвет. В тексте те слова, которые считаются терминами, подсвечены цветом темы, определённой по формуле (3.1). Пример такой визуализации — на рис. 1.

3.2 Определение родительской темы документа

Одно из основных назначений тематического моделирования — кластеризация коллекции документов (т.е. «разложение по папкам»). Для этого нужно определить, к какой теме относится данный документ. Во всех описанных ниже визуализациях тема, к которой относится документ, определяется по формуле:

$$t^*(d) = \arg\max_t p(t|d) = \arg\max_t \theta_{td}. \qquad (3.2)$$

Также предполагается, что документ может принадлежать нескольким темам. Зададим $\varepsilon \in (0, 1]$ – порог. Определим

$$T^*(d) = \{\arg\max_t \theta_{td}\} \cup \{t|\theta_{td} > \varepsilon\}. \tag{3.3}$$

Заметим, что если $\varepsilon > \frac{1}{2}$, то (3.3) всегда будет содержать одну тему.

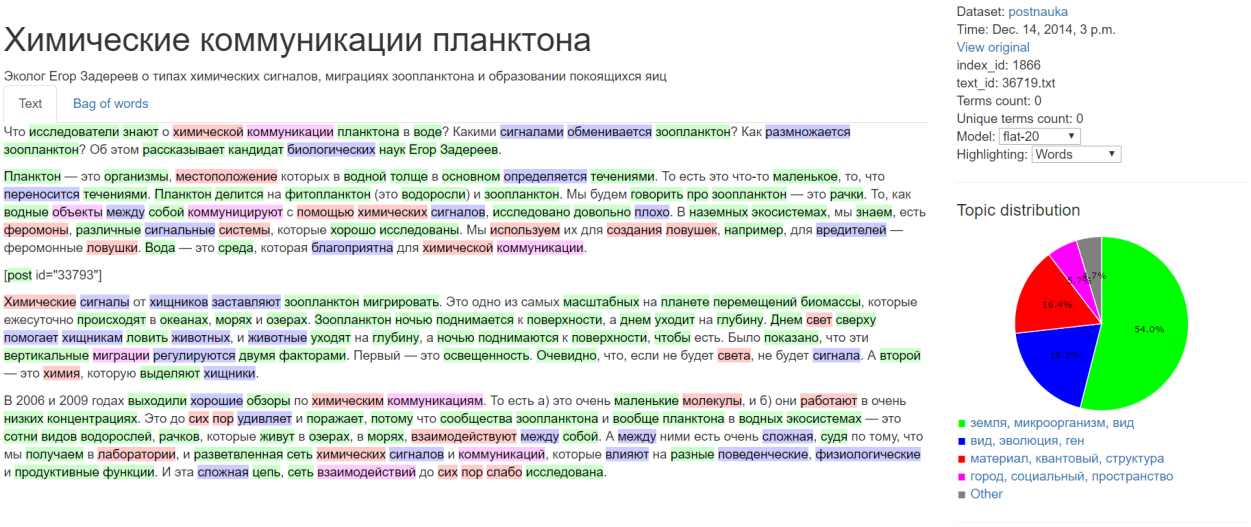


Рис. 1: Визуализация документа в VisARTM

3.3 Визуализация и именование тем

В VisARTM темы представляются ранжированным списком терминов и документов.

Термины в ранжированном списке сортируются по вероятностям $p(w|t)$, то есть по элементам столбца матрицы Φ, соответствующего данной теме. При этом выводятся 100 терминов с наибольшей вероятностью.

В ранжированный список документов включаются только те документы, которые считаются дочерними документами данной темы согласно формуле (3.3). При этом они сортируются по вероятности $p(t|d)$.

Также темам нужно дать короткие называния, дающие представление о содержании темы. Задача автоматического именования тем в тематических моделях на сегодняшний день не имеет универсального решения. В данной работе применяется простой подход — выбирается несколько (по умолчанию — 3) слов из описанного выше ранжированного списка и выводятся через запятую. Также есть возможность ручного именования тем.

Рассмотрим отдельно формирование списка терминов для мультимодальных тематических моделей. Простейший подход — вывод отдельных списков (отсортированных по $p(w|t)$) для каждой модальности. Если же требуется вывести общий список, нельзя сравнивать вероятности $p(w|t)$ для разных модальностей. В данной работе задаются веса модальностей $\alpha_1, \ldots, \alpha_{|M|}$, такие что $\alpha_i \geq 0$ и $\sum_{m=1}^{|M|} \alpha_m = 1$. При ранжировании вероятности из матрицы Φ умножаются на эти коэффициенты и по полученным числам производится сортировка.

Этот подход позволяет учитывать разные модальности, даже если размеры словарей модальностей сильно отличаются. Например, если одна модальность содержит 1000 терминов, а другая 100000, то без перенормировки, все термины в начале ранжированного списка будут принадлежать первой модальности. Также так можно исключить некоторые модальности (например, даты) из ранжированного списка, задав их коэффициент равным нулю.

3.4 Визуализация тематической модели во времени

Одна из задач визуализации тематических моделей — показывать изменение трендов в потоке информации с течением времени. Особенно это актуально для анализа новостей в СМИ. Для этого используются *темпоральные* тематические модели, учитывающие время публикации документа.

В VisARTM используется следующий подход: строится тематическая модель, а потом время разбивается на интервалы i, и документы группируются не только по темам, но и по времени. Время откладывается по оси абсцисс, а темы — по оси ординат. Есть два подхода — группировать документы, и показывать количество документов по данной теме в данный промежуток времени интенсивностью цвета (рис. 2a), или строить график количества документов от времени (рис 2b).

Также при не очень большом числе документов — несколько сотен в теме, их можно отобразить непосредственно в виде точек (рис. 2c). Тогда плотность точек свидетельствует о интенсивности публикаций, а наводя указатель на точки можно сразу видеть названия отдельных документов.

В описанном выше подходе предполагается, что документы монотематичны. Если это не так, надо вместо количества документов показывать темпоральную интенсивность. Пусть каждому документу d поставлено в соответствие время публикации $\tau : D \to [\xi_{min}, \xi_{max}]$. Также задано разбиение области значений этой характеристики на непересекающиеся интервалы i (дни, недели, месяцы, годы). Тогда темпоральная интенсивность темы:

$$p(i|t) = \sum_d p(i|d)p(d|t) \tag{3.4}$$

Так как интервалы не пересекаются, $p(i|d) = \big[\tau(d) \in i\big]$, поэтому

$$p(i|t) = \sum_{d \in D_i} p(d|t), \tag{3.5}$$

где $D_i = \{d \in D | \tau(d) \in i\}$.

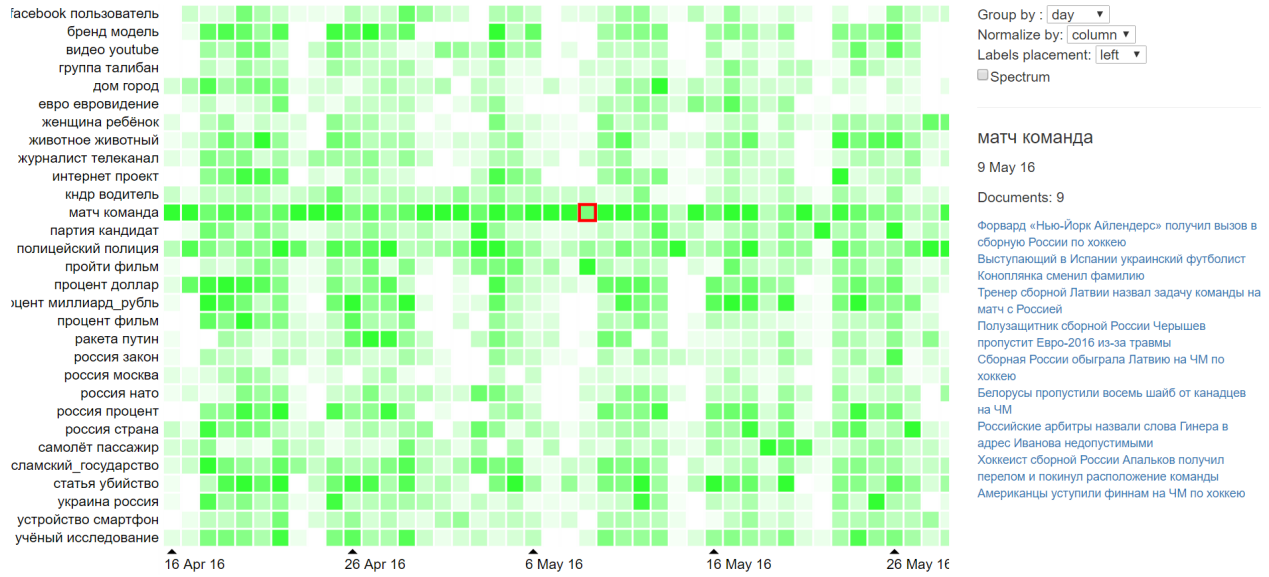
Применив ядерное сглаживание, можно определить темпоральную интенсивность темы как функцию времени:

$$\hat{p}(\tau|t) = \frac{1}{h} \sum_{d \in D} K\Big(\frac{\tau(d) - \tau}{h}\Big) p(d|t), \tag{3.6}$$

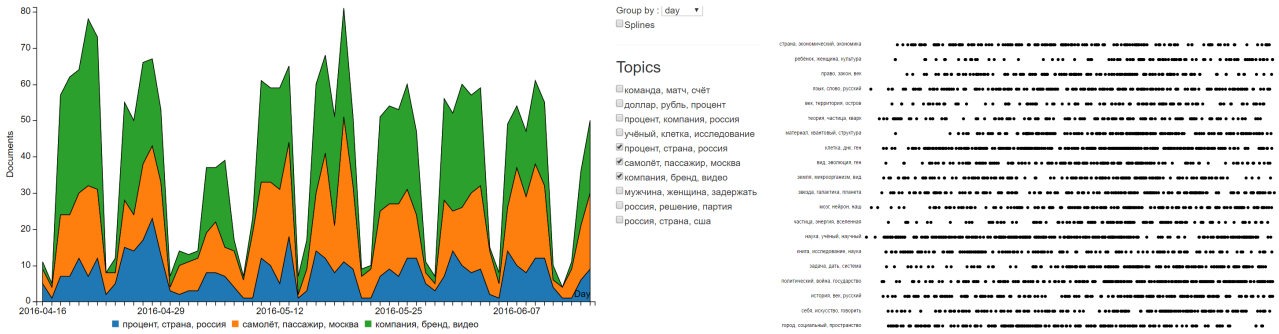
где h — ширина окна, $K(x)$ — ядерная функция.

Проверим, что эта функция является функцией плотности распределения случайной величины:

$$\int_{-\infty}^{\infty} \hat{p}(\tau|t)d\tau = \frac{1}{h}\sum_{d\in D} p(d|t) \int_{-\infty}^{\infty} K\Big(\frac{\tau(d)-\tau}{h}\Big)d\tau = \frac{1}{h}\sum_{d\in D} p(d|t)\cdot h = \sum_{d\in D} p(d|t) = 1 \quad (3.7)$$



(a) Сетка



(b) График (c) Точки

Рис. 2: Темпоральная визуализация в VisARTM

3.5 Визуализация иерархических тематических моделей

Иерархическая ТМ состоит из $L > 1$ уровней. Согласно (2.12), связь между уровнями иерархии описывается матрицами условных вероятностей $p(t|a)$. Таким образом, иерархическая ТМ может быть представлена многодольным графом, в котором вершины соответствуют темам, доли — уровням, а веса рёбер равны вероятностям $p(t|a)$.

Однако иерархические структуры естественно представлять деревом. Такое дерево может быть получено из многодольного графа, если у каждую тему t соединять ребром только с одной родительской темой a.

Очевидный подход состоит в том, чтобы в качестве родительской темы выбирать наиболее вероятную. Пусть $t \in T_{\ell+1}$ — тема $(\ell+1)$-го уровня. Пусть $\pi : T_{\ell+1} \to T_\ell$ — функция, определяющая родительскую тему. Тогда

$$\pi(t) = \arg\max_{a \in T_\ell} p(a|t) \tag{3.8}$$

По формуле Байеса, $p(a|t) = \frac{p(t|a)p(a)}{p(t)}$. Согласно (2.12), $p(t|a) = \psi_{ta}^{\ell}$.

Вероятность темы $p(t)$ определим из матрицы Θ следующим образом:

$$p(t) = \frac{1}{n}\sum_{d \in D} p(t|d) n_d = \frac{1}{n}\sum_{d \in D} \theta_{td} n_d \tag{3.9}$$

Итак,

$$p(a|t) = \psi_{ta}^{\ell} \frac{\sum_{d \in D} \theta_{ad}^{\ell} n_d}{\sum_{d \in D} \theta_{td}^{\ell+1} n_d} \tag{3.10}$$

и

$$\pi(t) = \arg\max_{a \in T_\ell} \psi_{ta}^{\ell} \frac{\sum_{d \in D} \theta_{ad}^{\ell} n_d}{\sum_{d \in D} \theta_{td}^{\ell+1} n_d} \tag{3.11}$$

Итак, теперь у нас есть дерево иерархии ТМ. Оно состоит из $L + 2$ уровней. На верхнем уровне находится всего одна вершина — корень. На нижнем уровне находятся документы. Родителями документов являются темы (родительская тема для документа определяется формулой (3.2)). Родителями тем из уровня $T_{\ell+1}$, $\ell = 1, \ldots, L-1$ являются темы из уровня T_i, определяемые соотношением (3.11). Родителями тем из T_1 является корень.

Простой способ визуализировать такое дерево — изобразить вершины и рёбра. Но при большом числе документов это невозможно, т.к. все документы находятся на одном уровне дерева и они не поместятся на изображении. В VisARTM реализован подход «сперва обзор, фильтрация при приближении, детали по требованию» [12]. Будем использовать модель вложенных многоугольников (рис. 3a). Каждой вершине графа иерархии (т.е. корню, теме или документу) ставится в соответствие многоугольник на плоскости. При этом если $\pi(a) = b$, то многоугольник вершины a содержится в многоугольнике вершины b.

На рис. 3c изображена реализация этого подхода, когда для представления графа используются прямоугольники. При этом родительский прямоугольник делится на дочерние таким образом, чтобы они не перекрывались, полностью покрывали родительский прямоугольник и их соотношение сторон было максимально близко к единице. Пусть N — число дочерних прямоугольников, w и h — ширина и высота родительского прямоугольника. Тогда вычислим $N_h = [\sqrt{N\frac{w}{h}} + 0.5]$, $N_w = [N/N_h]$, $d = N - N_w \cdot N_h$. Разобьём родительский прямоугольник на N_h строк — прямоугольников ширины w и высоты $\frac{h}{N_h}$. Первые d строк разобьём вертикальным линиями на $N_w + 1$ прямоугольников, а оставшиеся строки — на N_w прямоугольников.

На рис. 3b изображена визуализация иерархической ТМ с помощью диаграмм Вороного. Для этого использована библиотека FoamTree [13].

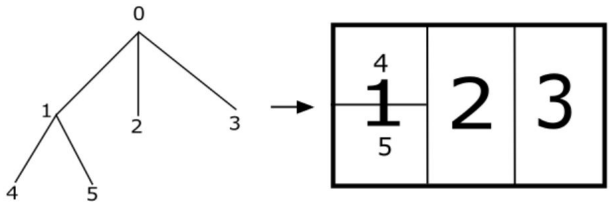
На рис. 3d изображена визуализация иерархической ТМ вложенными кругами.

Иногда документ или тема может принадлежать нескольким родительским темам. Определим порог иерархии ε и максимально допустимое количество родительских тем $Z \geq 1$. Тогда определим $\pi(t)$ — можество родительских вершин для темы:

$$\pi(t) = \left\{ \arg\max_{a \in T_\ell} p(a|t) \right\} \cup \left\{ a \in T_\ell \middle| p(a|t) \geq \varepsilon \right\} \tag{3.12}$$

Если в этом множестве оказалось больше, чем Z тем, выберем ровно Z с максимальным $p(a|t)$.

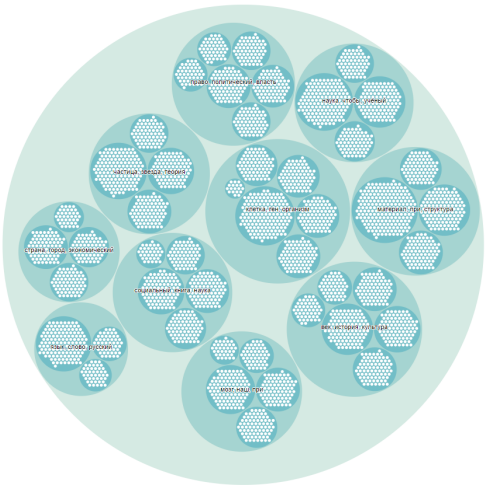
Теперь иерархия уже представляется не деревом, а слоистым графом. В нём будет $L+2$ слоёв: $\{\text{root}\}, T_1, \ldots, T_L, D$. Для всех таких t, a, что $\pi(t) = a$, в графе будет ребро (t, a). Для того, чтобы отображать такой граф рамках описанной выше модели вложенных многоугольников, предлагается каждый документ или тему, которые имеют несколько родителей, копировать вместе с поддеревьями и отображать несколько раз.



(a) Модель вложенных многоугольников

(b) FoamTree

(c) Вложенные прямоугольники



(d) Вложенные круги

Рис. 3: Иерархические визуализация в VisARTM

3.6 Описание VisARTM

Все вышеописанные визуализации были реализованы в информационной системе VisARTM. Кроме визуализаций, эта система содержит ряд других возможностей.

Назначение. Основное назначение VisARTM — предоставить исследователям, работающим с библиотекой BigARTM, возможность изучать тематические модели с помощью разных визуальных представлений.

Коллекции документов. Система работает с коллекциями документов (*datasets*) в формате Vowpal Wabbit. Она содержит встроенный конвертер из формата UCI в формат Vowpal Wabbit. Кроме того, VisARTM способен работать с документами в необработанном виде, предоставленных в виде набора текстовых файлов. Тогда документы разбиваются на слова, которые затем лемматизируются (с использованием pymorphy3 для русского языка или nltk для английского языка). Также VisARTM может выделить *энграммы* — сочетания рядом стоящих слов и отфильтровать слова — удалить из словаря слова, встречающиеся слишком часто или слишком редко.

Тематические модели. После загрузки коллекции в VisARTM можно загрузить модель BigARTM для этой коллекции. Для этого нужно загрузить модель в виде матриц Θ и Φ (и, возможно, матриц Ψ для иерархической модели). VisARTM также поддерживает автоматическое создание моделей с помощью BigARTM через web-интерфейс. Для этого нужно указать число слоёв и количество тем на каждом слое. Для плоских моделей также доступно подключение регуляризаторов. Таким образом, пользователь может загрузить коллекцию документов, построить тематическую модель и посмотреть её визуализации полностью через web-интерфейс, не прибегая к написанию кода.

Представления. Пользователь может просматривать список документов в коллекции. Для документа он может просматривать документ в виде текста или «мешка слов», его метаданные (включая теги), его распределение по темам некоторой модели (с подсветкой слов) и список близких по тематике документов (в смысле одного из расстояний, которые будут обсуждаться в разделе 4.2). Кликнув на слово в документе, пользователь попадает на страницу этого термина, на которой перечислены все документы, куда входит это слово (с указанием соседних слов) и распределение этого слова по темам некоторой модели. На странице темы перечислены все темы модели и возможные визуализации для этой модели.

Безопасность. В VisARTM есть пользователи и права. У каждой коллекции и модели есть владелец, и только он может менять эту коллекцию и модель. Если пользователь укажет, что коллекция является закрытой, то только он сможет просматривать информацию, относящуюся к ней (включая модели).

Тематические спектры. В VisARTM реализованы все алгоритмы построения тематических спектров, описанные в разделах 4 и 5. Пользователь может выбрать алгоритм и метрику, и во всех визуализациях, где важен порядок тем, они будут упорядочены согласно спектру. Реализованы также тематические спектры для иерархических моделей с любым количеством уровней.

Поиск. В системе можно искать документы по терминам в тексте или названии, а также по сочетаниям нескольких терминов.

Сбор асессорских оценок. Дополнительной возможностью VisARTM является сбор асессорских оценок о тематических моделях. Администратор может создать *задачу оценивания*, связанную с некоторой моделью. Затем один или несколько оценщиков (асессоров) обращаются к системе и получают *задания*, которые выполняют в браузере. По

завершении выполнения задания результат сохраняется на сервер, а пользователь может перейти к выполнению нового задания. В любой момент администратор может получить результаты оценивания. Чтобы создать задачу оценивания, администратор должен создать python-модуль, содержащий несколько функций: создание нового задания, создание контекста для задания, обработка запросов асессора в ходе выполнения задания, сохранение результата, агрегация результатов. Также он должен создать HTML-страницу задания, которая отображает полученный контекст задания и реагирует на действия асессора, посылая серверу нужные POST-запросы.

Реализация. VisARTM — это web-сервис, спроектированный согласно идеологии Model-View-Container с использованием фреймворка для веб-приложений django [14]. Серверная часть написана на языках python и C. Клиентская часть и дизайн — с помощью HTML, CSS, javascript и фреймворка Twitter Bootstrap [15].

Система использует ряд библиотек, в том числе:

- Научные библиотеки языка Python 3 из пакета Anaconda (в том числе scikit-learn [16]).
- Библиотеку тематического моделирования BigARTM [10].
- Алгоритм Lin-Kernighan-Helsgaun, собранный из исходников, взятых с [17].
- Библиотеку визуализации данных d3 [18].
- Библиотеку визуализации иерархических данных с помощью диаграмм Вороного FoamTree [13].

Данные хранятся частично в файловой системе, частично в базе данных под управлением PostgreSQL.

Использование системы. Система проектировалась для двух основных групп пользователей. Во-первых, она может использоваться как сайт, открыто доступный в сети Интернет. Предполагается, что этим сайтом будут пользоваться люди, которые хотят ознакомиться с возможностями тематического моделирования и построить тематические модели для своих коллекций. Во-вторых, исследователи, работающие с BigARTM могут развернуть систему локально на персональном компьютере и визуализировать модели, которые они строят с помощью BigARTM.

На момент защиты данной работы система развёрнута на одном из серверов Московского физико-технического института и доступна по адресу `http:\visartm.vdi.mipt.ru`. Кроме того, систему можно развернуть локально. Исходный код находится в репозитории [19], а инструкции по установке — в документации [20].

4 Тематический спектр

4.1 Постановка задачи

Одним из результатов обучения тематической модели является множество тем. Однако порядок тем в модели случаен: если применить одну и ту же произвольную перестановку к столбцам матрицы Φ и срокам матрицы Θ, получим ту же самую тематическую модель. Естественно возникает задача упорядочить темы по их семантической близости, чтобы визуализация тематической модели несла больше информации.

Пусть задана некоторая функция расстояния между темами $\rho : T \times T \to [0, +\infty)$. Изначально темы в множестве T имеют какой-то порядок, т.е. $T = \{t_1, t_2, \ldots, t_{|T|}\}$. Поставим задачу поиска такой перестановки тем, чтобы сумма расстояний между соседними в этой перестановке темами была минимальной:

$$\sum_{i=1}^{|T|-1} \rho(t_{\pi_i}, t_{\pi_{i+1}}) \to \min_{\pi \in S_{|T|}}, \tag{4.1}$$

где S_n — множество всех перестановок на n элементах.

Будем называть *тематическим спектром* тематической модели перестановку тем π^*, доставляющую минимум в задаче (4.1). Последовательность тем $t_{\pi_1^*}, t_{\pi_2^*}, \ldots, t_{\pi_{|T|}^*}$ также будем называть тематическим спектром.

4.2 Функции расстояния между темами

Рассмотрим матрицу Φ. В ней $|T|$ столбцов, каждый из которых описывает некоторую тему как вероятностное распределение над словарём. Далее будем рассматривать только такие функции ρ, которые являются функциями от соответствующих столбцов матрицы φ. Кроме того, потребуем, чтобы $\rho(t, t') = \rho(t', t)$ и $\rho(t, t) = 0$.

Рассмотрим несколько таких функций расстояния.

Евклидово расстояние:

$$\rho_E(t, s) = \sqrt{\sum_{w \in W} \big(\varphi_{wt} - \varphi_{ws}\big)^2}. \tag{4.2}$$

Манхэттенское расстояние:

$$\rho_M(t, s) = \sum_{w \in W} \big|\varphi_{wt} - \varphi_{ws}\big|. \tag{4.3}$$

Косинусное расстояние:

$$\rho_C(t, s) = 1 - \frac{1}{\|t\|\|s\|} \sum_{w \in W} \varphi_{wt}\varphi_{ws}; \quad \|t\| = \sqrt{\sum_{w \in W} \varphi_{wt}^2}. \tag{4.4}$$

Расстояние Хеллингера:

$$\rho_H(t, s) = \sqrt{\frac{1}{2} \sum_{w \in W} \big(\sqrt{\varphi_{wt}} - \sqrt{\varphi_{ws}}\big)^2}. \tag{4.5}$$

Расстояние Йенсена-Шеннона. Для измерения расстояния между вероятностными распределениями часто используется расстояние Кульбака-Лейблера:

$$D_{KL}(u\|v) = \sum_i u_i \ln \frac{u_i}{v_i}. \tag{4.6}$$

Однако эта функция расстояния несимметрична и неопределена, если вектор u или v содержит нулевые элементы. Поэтому также используют расстояние Йенсена-Шеннона:

$$D_{JS}(u,v) = \frac{1}{2}\left(D_{KL}\big(u\|\frac{u+v}{2}\big) + D_{KL}\big(v\|\frac{u+v}{2}\big)\right). \tag{4.7}$$

Можно показать, что

$$D_{JS}(u,v) = H\Big(\frac{u+v}{2}\Big) - \frac{1}{2}\big(H(u) + H(v)\big), \tag{4.8}$$

где $H(u) = -\sum_i u_i \ln u_i$ — энтропия. Энтропия может быть вычислена даже для векторов с нулевыми значениями, если полагать $0 \ln 0 = 0$. Значит, и расстояние Йенсена-Шеннона может быть вычислено для векторов с нулевыми значениями.

Итак, обозначая Φ_t — столбец матрицы Φ, соответствующий теме t, получаем расстояние Йенсена-Шеннона между темами:

$$\rho_{JS}(t,s) = H\Big(\frac{\Phi_t + \Phi_s}{2}\Big) - \frac{1}{2}\big(H(\Phi_t) + H(\Phi_s)\big). \tag{4.9}$$

Расстояние Жаккара. Эта функция расстояния показывает отношение количества слов в пересечении тем к количеству слов в объединении тем. Будем считать, что слово принадлежит теме, если оно в ней встречается с вероятностью большей, чем $\frac{1}{|W|}$.

$$\rho_J(t,s) = 1 - \frac{\left|\left\{w \in W | \varphi_{wt} > \frac{1}{|W|} \wedge \varphi_{ws} > \frac{1}{|W|}\right\}\right|}{\left|\left\{w \in W | \varphi_{wt} > \frac{1}{|W|} \vee \varphi_{ws} > \frac{1}{|W|}\right\}\right|}. \tag{4.10}$$

Также в экспериментах будем использовать расстояние Чебышева, как пример неудачной метрики:

$$\rho_{Ch}(t,s) = \max_{w \in W} \big|\varphi_{wt} - \varphi_{ws}\big|. \tag{4.11}$$

4.3 Решение задачи построения тематического спектра

Дальше нам будет удобно работать с матрицей расстояний $R \in \mathbb{R}^{|T|\times|T|}$, определяемой формулой $R[i,j] = \rho(t_i, t_j)$. Также введём обозначение $N = |T|$ — число тем. Тогда задача (4.1) запишется в виде

$$\sum_{i=1}^{N-1} R[\pi_i, \pi_{i+1}] \to \min_{\pi \in S_{|T|}}. \tag{4.12}$$

Заметим, что задача (4.12) — это задача поиска гамильтонового пути минимального веса в полном взвешенном неориентированном графе с N вершинами и матрицей весов R. Это NP-полная задача.

Данная задача сводится к задаче поиска гамильтонового цикла минимального веса в полном взвешенном неориентированном графе (задача коммивояжёра, Travelling Salesman Problem, TSP) [21].

Сведение. Добавим в исходный граф G новую вершину x и соединим её с остальными вершинами рёбрами веса 0. Получим новый граф G'. Найдём в G' гамильтонов цикл минимального веса, и затем удалим из него вершину x. Получим гамильтонов путь минимального веса в исходном графе G.

Доказательство. Пусть это не так. Тогда в G существует другой цикл меньшего веса. Соединим его края с новой вершиной, и получим гамильтонов цикл в графе G' меньшего веса, чем найденный.

Итак, задача (4.12) является NP-трудной, и поиск её точного решения алгоритмами полного перебора не имеет смысла уже при числе тем, больших 15, поэтому рассматривать их не будем.

Ниже рассмотрим несколько приближённых алгоритмов решения задачи (4.12).

4.3.1 Одномерная агломеративная иерархическая кластеризация

Общая идея этого подхода изложена в [22]. Она состоит в последовательном объединении кластеров по некоторому алгоритму. Ниже приведём алгоритм построения тематического спектра, использованный в данной работе.

Определим кластер как последовательность тем. Назовём крайними те темы, которые стоят в начале или конце кластера.

Изначально имеется N кластеров, каждый из которых содержит по одной теме. На каждой итерации алгоритма перебираются все пары крайних тем, не принадлежащих одному кластеру, и среди них выбирается пара (t, s) с наименьшим расстоянием между ними. Затем эти два кластера конкатенируются таким образом, чтобы эти две темы оказались соседними (возможно, один из кластеров для этого понадобится обратить).

Алгоритм повторяется до тех пор, пока не останется один кластер. Так как на каждой итерации количество кластеров уменьшается на единицу, будет выполнено ровно $N - 1$ итераций, на каждой из которых проверяется не более N^2 пар тем и выполняется слияние кластеров за $O(N)$. Таким образом, сложность этого алгоритма $O(N^3)$.

4.3.2 Алгоритмы многомерного шкалирования

Алгоритмы многомерного шкалирования широко применяются в визуализации данных в пространстве большой размерности. Основная идея таких алгоритмов состоит в том, чтобы отобразить множество точек из многомерного пространства в пространство низкой размерности (чаще всего двумерное) таким образом, чтобы попарные расстояния между образами точек как можно меньше отличались от попарных расстояний между прообразами.

В данной работе рассмотрены два алгоритма многомерного шкалирования: MDS (Multi-dimensinal scaling) [23] и t-SNE (t-distributed Stochastic Neighbor Embedding) [24]. В обоих случаях строилось отображение в одномерное пространство, и темы в спектре расставлялись в том же порядке, в каком следовали их образы в одномерном пространстве. Были использованы реализации алгоритмов t-SNE и MDS из пакета scikit-learn [16].

4.3.3 Симуляция отжига

Симуляция отжига (simulated annealing, SA)— один из вероятностных алгоритмов для решения решения задач дискретной оптимизации [25]. В [26] он применяется для решения задачи TSP. Ниже изложим алгоритм симуляции отжига, который был использован в данной работе для решения задачи (4.12).

Пусть $E(\pi) = \sum_{i=1}^{|T|-1} R[\pi_i, \pi_{i+1}]$ — энергия. Также введём температуру τ — положительную величину, которая некоторым образом зависит от номера итерации i.

При инициализации алгоритма берётся произвольная перестановка π длины T. Затем повторяется N_{steps} итераций, на каждой из которых случайно выбираются два индекса t и s и меняются местами элементы перестановки, стоящие а местах t и s. Обозначим новую перестановку π'. Пусть $\Delta E = E(\pi') - E(\pi)$. Вычислим вероятность перехода по формуле

$$P = \begin{cases} 1, & \text{если } \Delta E \leq 0 \\ e^{-\frac{\delta E}{\tau}}, & \text{если } \Delta E > 0 \end{cases} \tag{4.13}$$

Теперь с вероятностью P присвоим $\pi := \pi'$ и перейдём к следующей итерации.

Зависимость температуры от номера шага (Cooling Schedule) была взята такая:

$$\tau(i) = \exp\left(\ln(T_{\max}) - \frac{\ln(T_{\max}) - \ln(T_{\min})}{N_{\text{steps}}} \cdot i\right) \tag{4.14}$$

Начальная и конечная выбирались по формулам $T_{\min} = \overline{A} \cdot 10^{-5}, T_{\max} = \overline{R} \cdot 10^5$, где $\overline{R}$ — среднее значение в матрице R. Число итераций N_{steps} выбиралось от 10^6 до 10^8, в зависимости от числа тем.

4.3.4 Алгоритм Лина-Кернигана-Хельсгауна

Согласно [21], лучшим по точности эвристическим алгоритмом для решения TSP является алгоритм Лина-Кернигана-Хельсгауна (Lin, Kernighan, Helsgaun), описанный в [27]. Он находит точное решение для большинства задач из репозитория задач TSP, размер которых составляет несколько сотен вершин. Кроме того, он очень эффективен — время его работы приближённо оценивается, как $O(n^{2.2})$.

Этот алгоритм, как и алгоритм отжига — алгоритм улучшения. Изначально рассматривается некоторый гамильтонов цикл. Потом он разрезается на несколько частей, и эти части сшиваются в другом порядке, дающем уменьшение суммарного веса пути. Правильный выбор количества мест разреза, а также множество других эвристик, описанных в [27], делают алгоритм таким эффективным.

В данной работе используется готовая реализация этого алгоритма, собранная из исходного кода, взятого с сайта Кельда Хельсгауна [17].

4.4 Меры качества спектра

Пусть найдена некоторая перестановка π. Рассмотрим несколько способов численно оценить, насколько эта перестановка удовлетворяет свойствам спектра (эти свойства состоят в том, что семантически близкие темы находятся в перестановке, а не связанные по смыслу темы — далеко).

Для удобства, все меры качества будем определять так, чтобы меньшие значения соответствовали лучшему качеству.

4.4.1 Сумма расстояний между соседями

Прежде всего, сама оптимизируемая в (4.12) функция является мерой качества спектра:

$$NDS(\pi) = \sum_{i=1}^{N-1} R[\pi_i, \pi_{i+1}]. \tag{4.15}$$

4.4.2 Средний ранг соседа

Эта мера качества показывает степень семантической близости соседних тем в спектре.

Назовём *рангом* темы v относительно темы u её номер в списке всех тем w (кроме v), упорядоченных по возрастанию $\rho(w,u)$. Формально,

$$\operatorname{rank}(v|u) = \left|\left\{w \in \overline{1,N} \,\middle|\, R[w,u] < R[v,u]\right\}\right| \tag{4.16}$$

Примечание. Эта формула не верна, если для одной темы найдётся несколько тем, равноудалённых от неё (вероятность чего равна нулю). В этом случае при реальном вычислении ранга эти темы надо произвольным образом упорядочить. Здесь же будем считать, что $\forall x,y,z\ \rho(x,y) = \rho(x,z) \Leftrightarrow y = z$.

В частности, если x — ближайшая тема к теме y, то $\operatorname{rank}(x|y) = 1$.

Для каждой упорядоченной пары x, y соседних тем в спектре посчитаем ранг темы x относительно y и усредним полученные значения. Назовём эту меру качества *средним рангом соседа* (mean neighbor rank).

Формально,

$$MNR(\pi) = \frac{1}{2N-2} \sum_{i=1}^{N-1} \big(\operatorname{rank}(\pi_{i-1}|\pi_i) + \operatorname{rank}(\pi_i|\pi_{i-1})\big) \tag{4.17}$$

В идеальном случае, когда для каждой темы в спектре её соседи являются ближайшими и в метрическом пространстве, среди $2N-2$ пар N должны дать ранг 1, и ещё $N-2$ должны иметь ранг 2, то есть $MNR = \frac{3N-4}{2N-2} \approx 1.5$.

4.4.3 Кривая расстояний (DDC)

Определим следующую функцию для $d \in \overline{1, N-1}$:

$$\operatorname{DDC}(d) = \frac{1}{N-d} \sum_{i=1}^{N-d} R[i, i+d] \tag{4.18}$$

$\operatorname{DDC}(d)$ — это среднее расстояние в метрическом пространстве между темами, находящимися на расстоянии d в спектре. Ожидается, что для хорошего спектра кривая $DDC(d)$ будет возрастать.

4.5 Оценивание качества спектра с помощью асессоров

Метрики, описанные в разделе 4.4, не могут в полной мере описывать качество спектра, т.к. все они используют функцию ρ, и её же используют алгоритмы построения спектра. В конечном итоге, на спектр будет смотреть пользователь, поэтому необходимо проверить качество спектра с его точки зрения. Ниже будет рассмотрено два метода оценки качества спектра с помощью асессоров.

4.5.1 Оценивание близости тем

Одно из требований пользователя к хорошему спектру — чтобы близкие в спектре темы являлись действительно близкими по смыслу. Поэтому прежде всего нужно проверить качество используемых метрик, и выбрать из них лучшую.

Для этого предлагается показывать асессору темы и предлагать из остальных выбрать те, которые он считает близкими по смыслу и поставил бы рядом. Темы предлагается показывать в виде списка нескольких топ-слов (слов с наибольшим $p(w|t)$).

Часто близкие темы могут не содержать общих слов в числе топ-слов, но быть синонимами или описывать схожие сущности. Такие зависимости не удастся отследить автоматически без специальных словарей, поэтому асессорские оценки будут независимы с описанными выше метриками, и их использование для оценивания метрик имеет смысл.

Опишем меры качества для метрик, основанные на асессорских оценках. Пусть каждая тема была показана K раз (одному или разным пользователям). Рассмотрим матрицу B, в которой $B[i,j]$ — число пользователей, указавших тему j в числе близких к теме i. Симметризуем и отнормируем эту матрицу: $C = \frac{1}{2K}(B+B^T)$. В разделе 4.6 будет рассмотрено несколько мер качества спектров, основанных на матрице C.

Проблемой такого метода сборок оценок может быть то, что если тем очень много (больше сотни), пользователю будет очень сложно прочитать их все и выбрать ближайшие к данной. Поэтому предлагается показывать в качестве кандидатов не все темы, а несколько (например, 20) ближайших по какой-либо метрике.

4.5.2 Оценивание подпоследовательностей спектра

Опишем метод непосредственной оценки качества спектра с помощью асессоров. Для этого будем выбирать из спектра несколько тем (обозначим число выбранных тем K), и предлагать пользователю их упорядочить. Таким образом, получаем перестановку α на некотором подмножестве тем. Пусть J — индексное множество этих тем. Тогда рассмотрим коэффициент ранговой корреляции Кендалла:

$$\tau(\alpha|\pi) = 1 - \frac{4}{K(K-1)} \sum_{1 \le i < j \le K} \Big[(\alpha_i < \alpha_j) \veebar (\pi_{J_i} < \pi_{J_j}) \Big] \tag{4.19}$$

Рассчитаем ранговое качество спектра (rank spectrum quality):

$$\mathrm{RSQ}(\pi) = \frac{1}{\mathcal{A}} \sum_{\alpha \in \mathcal{A}} \Big| \tau(\alpha|\pi) \Big|, \tag{4.20}$$

где $\mathcal{A}$ — множество всех оценок.

Модуль берётся, потому что мы считаем реверс спектра тем же самым спектром.

Примечание. Данный метод не реализован в текущей версии VisARTM.

4.6 Меры качества, основанные на асессорских оценках

Опишем несколько способов оценивать качество спектра с использованием матрицы C асессорских оценок близости тем, получение которой описано в разделе **??**. Обозначения всех мер качества, использующих асессорские оценки, будут начинаться с буквы A. Все числовые меры качества будут подчиняться правилу «чем меньше — тем лучше».

4.6.1 Корреляция Пирсона

Прежде всего, оценим непосредственно качество используемой функции расстояния. Ожидается, что вычисленные с ей помощью расстояния (т.е. элементы матрицы R) должны коррелировать с оценками (т.е. элементами матрицы C). Поэтому используем коэффициент корреляции Пирсона между элементами этих двух матриц (assessment-metric correlation):

$$\begin{gathered}\mathrm{AMC}(\pi) = \frac{\sum_{i<j}(R_{ij} - \overline{R})(C_{ij} - \overline{C})}{\sqrt{\sum_{i<j}(R_{ij} - \overline{R})^2}\sqrt{\sum_{i<j}(C_{ij} - \overline{C})^2}}; \\ \overline{C} = \frac{2}{n(n-1)}\sum_{i<j} C_{ij}; \quad \overline{R} = \frac{2}{n(n-1)}\sum_{i<j} R_{ij}.\end{gathered} \tag{4.21}$$

В матрице расстояний R чем меньше значения, тем ближе темы, а в матрице пользовательских оценок C — наоборот, чем больше значения, тем ближе темы. Поэтому идеальной корреляции соответствует $\mathrm{UMC} = -1$, а если корреляция отсутствует, то $\mathrm{UMC} = 0$, то есть в эксперименте значения UMC будут отрицательными. В дальнейших экспериментах они так и будут приводиться, чтобы следовать принципу «чем меньше — тем лучше».

Заметим, что имеет смысл сравнивать разные метрики, используя AMC только с одной и той же матрицей оценок C.

4.6.2 Штраф за отдаление

Пройдёмся по всем парам тем (i, j), и будем штрафовать с весом C_{ij} эту пару, если темы не являются соседними в спектре. Причём чем дальше темы в спектре, тем больше штраф. Назовём эту меру качества *штрафом за отдаление* (distancing penalty).

$$\mathrm{ADP}(\pi) = \sum_{i<j} C_{ij}(|\pi_i^{-1} - \pi_j^{-1}| - 1). \tag{4.22}$$

4.6.3 Средняя несхожесть соседей

Естественно считать, что C_{ij} — асессорская оценка схожести тем i и j. Тогда назовём величину $1 - C_{ij}$ *несхожестью* тем i и j. Посчитаем среднюю несхожесть для всех пар тем, которые являются соседними в спектре. Назовём эту меру качества *средней несхожестью соседей* (mean neighbor dissimilarity):

$$\text{AMND} = 1 - \frac{1}{N-1}\sum_{i=1}^{N-1} C[\pi_i, \pi_{i+1}] \tag{4.23}$$

4.6.4 Доля несхожих соседей

Будем считать темы i и j *несхожими*, если ни один асессор не счёл близкими, т.е. $C_{ij} = 0$. Посчитаем долю таких тем среди соседних тем в спектре. Назовём эту меру качества долей несхожих соседей (dissimilar neighbors part).

$$\text{ADNP} = \frac{1}{N-1}\sum_{i=1}^{N-1} \Big[C[\pi_i, \pi_{i+1}] = 0\Big] \tag{4.24}$$

4.6.5 Кривая оценка-расстояние

По аналогии с (4.18) определим *кривую оценка-расстояние* (ADC) для $d \in \overline{1, N-1}$:

$$\text{ADC}(d) = \frac{1}{N-d}\sum_{i=1}^{N-d} C[i, i+d] \tag{4.25}$$

Здесь $\text{ADC}(d)$ — это средняя асессорская оценка близости для всех тем, находящихся в спектре на расстоянии d. Ожидается, что в хорошем спектре $ADC(1) \approx 1$, с ростом d эта функция должна быстро убывать, и $ADC(d) \approx 0$ при больших d.

4.7 Эксперименты

Эксперименты проводились на двух коллекциях:

- postnauka — публикации сайта `postnauka.ru` за 2012-2016 годы, 3446 документов, 35531 терминов);
- lenta — публикации сайта `lenta.ru` за апрель-июнь 2016 года, 8639 документов, 51634 терминов).

Для каждой коллекции была построена модель PLSA из 25 тем. Были построены спектры с помощью всех алгоритмов, описанных в разделе 4.3, и всех функций расстояния, описанных в разделе 4.2.

Были собраны асессорские оценки близости тем для обоих моделей по методике, описанной в разделе 4.5.1 с числом повторов K, равным 5. Эти оценки были использованы при вычислении мер качества, приведенных ниже.

4.7.1 Сравнение алгоритмов

Разные алгоритмы сравнивались по мерам качества NDS, MNR, ADP, AMND и ADNP с использованием одной и той же функции расстояния. В таблицах 1 и 2 приведены меры качества спектров, построенных разными алгоритмами (для коллекции postnauka с функцией расстояния Жаккара и для коллекции lenta с функцией расстояния Хеллингера). Для сравнения приводятся эти же меры качества, посчитанные для исходной перестановки тем (строка «No arranging»).

Во всех случая алгоритм LKH лучше остальных по всем мерам качества. Алгоритмы отжига и агломеративной кластеризации дают результаты, заметно лучшие, чем для исходной перестановки. Алгоритмы многомерного шкалирования MDS и t-SNE дают плохие результаты.

Отдельно на модельных сравнивались алгоритмы LKH и отжига, так как они оба непосредственно решают задачу (4.12). Для $N \in \overline{5,200}$ были случайно сгенерированы матрицы расстояний R, и задача (4.12) решалась обоими алгоритмами. На рис. 4a показано отношение NDS для перестановки, полученной алгоритмом отжига (с $N_{\text{steps}} = 10^8$) к NDS для перестановки, полученной алгоритмом LKH. При $N \geq 12$ это отношение стало больше единицы и продолжало расти с ростом N, то есть ответ алгоритма отжига становился хуже, чем ответ алгоритма LKH. Также сравнивалось реальное время работы этих алгоритмов (рис. 4b). Итак, алгоритм LKH работает быстрее и даёт лучший ответ, чем алгоритм симуляции отжига.

На основании экспериментов можно сделать вывод вывод, что алгоритм LKH — лучший алгоритм для построения тематического спектра. Он по умолчанию используется в VisARTM для построения тематических спектров. В экспериментах, описанных ниже, используется только он.

4.7.2 Сравнение функций расстояния

В разделе 4.2 описаны 6 функций расстояния, предположительно подходящих для оценки семантической близости тем: евклидово расстояние (euclidean), косинусное расстояние (cosine), манхэттенское расстояние (manhattan), расстояние Хеллингера (hellinger), расстояние Йенсена-Шеннона (jsd), расстояние Жаккара (jaccard). Будем называть их «*хорошими*» функциями расстояния. Также рассматривается расстояние Чебышева (chebyshev) как предположительно плохая функция расстояния.

В таблицах 3 и 4 спектры, построенные с использованием этих метрик, сравниваются по мерам качества MNR, ADP, AMND, ADNP, AMC.

На рис. 6 приведены кривые DDC и ADC. Каждый из первых семи графиков строился следующим образом: выбирается одна метрика (на графике изображена жирной линией), и по ней строится спектр. Затем этот спектр фиксируется, и для него и разных матриц R считается кривая DDC по формуле (4.18). Чтобы эти кривые можно было изображать на одном графике, недиагональные элементы матрицы R предварительно линейно преобразовываются так, чтобы расстояние между ближайшими темамии было равно 0, а между самыми удалёнными — 1.

На последнем графике на рис. 6 изображены кривые ADC (4.25) для спектров, построенных для каждой метрики.

Для каждой функции расстояния построим спектр, и зафиксировав его, построим DDC-кривую, подставляя разные матрицы R в (4.18)

Из таблиц и графиков видно, что функции расстояния cosine, minkovsky, hellinger, jsd и jaccard примерно одинаково хороши. Это видно из того, что их графики DDC очень близки, и все они резко растут при малых d. На кривой ADC малым d соответствуют большие значения оценок, а при $d \geq 5$ средние значения оценок не превышают C, то есть близкие в спектре темы близки по мнению асессоров, а далёкие — далеки.

Функция расстояния euclidean немного хуже, чем остальные «хорошие» функции.

Функция расстояния chebyshev сильно хуже всех остальных. На графиках DDC это выражается в том, что её график находится ниже всех остальных графиков, а графики

DDC для случая, когда спектр строился по метрике chebyshev, почти не растут в области малых d — это значит, что в спектре и близкие и далёкие темы в среднем имеют одинаковую семантическую близость, то есть спектр получился случайным. На графике ADC кривая для метрики chebyshev принимает малые значения при малых d и большие значения при больших d (по сравнению с остальными метриками), что свидетельствует о том, что эта метрика никак не соотносится с пользовательскими оценками.

На рис. 5 проиллюстрирована корреляция асессорских оценок (значений матрицы C) с расстояниями между темами (значениями матрицы R). Каждая точка иллюстрирует пару тем (t, s), её абсцисса равна $R[t, s]$, а ордината — $C[t, s]$.

На основании экспериментов можно сделать вывод, что функции расстояния euclidean, cosine, minkovsky, hellinger, jsd, jaccard примерно одинаково хороши как меры семантической близости тем (хоть функция euclidean несколько хуже остальных). Выбор оптимальной метрики может зависеть от конкретной модели. В VisARTM метрикой по умолчанию выбрано расстояние Жаккара, но пользователь может выбирать, какую метрику использовать.

4.7.3 Примеры спектров

На рис. 7 изображены тематические спектры, построенные с помощью алгоритма LKH. Каждая тема представлена десятью топ-словами, отсортированными по убыванию φ_{wt}.

Таблица 1: Сравнение алгоритмов (postnauka, расстояние Жаккара)

Алгоритм	NDS	MNR	ADP	AMND	ADNP
No arranging	17.9758	12.8125	154.40	0.91	0.62
LKH	**16.7725**	**2.5208**	**53.90**	**0.72**	**0.21**
Annealing	16.8223	3.0208	64.40	0.74	0.29
t-SNE	17.9245	12.7917	140.70	0.90	0.71
MDS	18.0651	14.0833	129.80	0.97	0.79
Agl. Clust.	16.8427	3.3125	55.60	0.75	0.33

Таблица 2: Сравнение алгоритмов (lenta, расстояние Хеллингера)

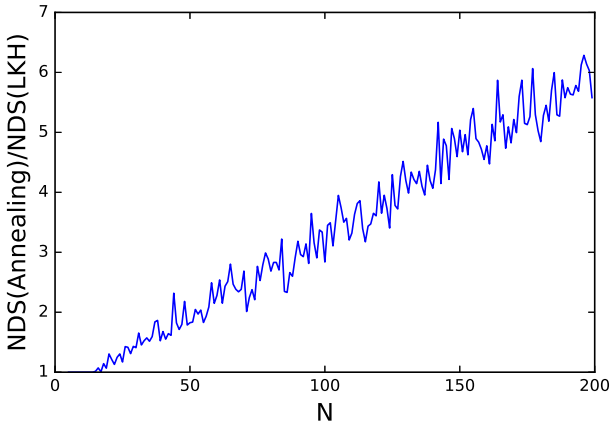
Алгоритм	NDS	MNR	ADP	AMND	ADNP
No arranging	20.4540	13.1667	174.90	0.97	0.83
LKH	**19.0180**	**3.0000**	**82.50**	**0.62**	**0.21**
Annealing	19.0661	3.4375	126.50	0.62	0.29
t-SNE	20.6573	14.9375	192.90	0.98	0.79
MDS	20.7519	15.8542	184.40	0.97	0.88
Agl. Clust.	19.0804	3.7917	94.70	0.62	0.25

Таблица 3: Сравнение метрик (postnauka)

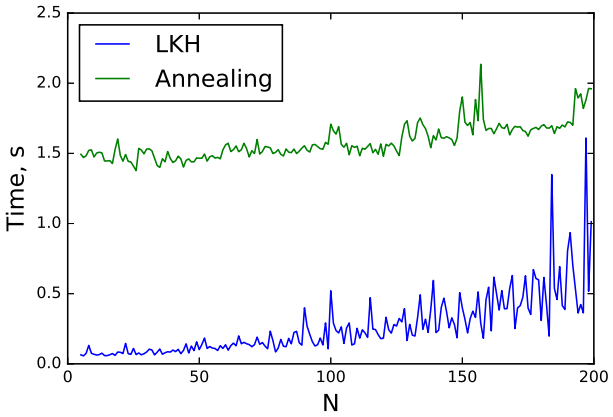
Метрика	MNR	ADP	AMND	ADNP	AMC
euclidean	7.2917	64.00	0.75	0.2917	-0.13
cosine	4.1875	**46.10**	0.70	0.2500	-0.36
manhattan	**2.2083**	54.20	0.72	**0.1667**	-0.49
hellinger	2.2292	66.00	**0.68**	0.2500	**-0.51**
jsd	2.2708	58.70	0.70	0.2083	-0.50
jaccard	2.5208	53.90	0.72	0.2083	-0.46
chebyshev	7.5625	127.80	0.85	0.4167	-0.06

Таблица 4: Сравнение метрик (lenta)

Метрика	MNR	ADP	AMND	ADNP	AMC
euclidean	7.0000	**75.20**	0.62	**0.2083**	-0.20
cosine	3.2917	87.60	**0.60**	0.2917	**-0.50**
manhattan	**2.8125**	97.40	0.65	0.2500	-0.46
hellinger	3.0000	82.50	0.62	**0.2083**	-0.48
jsd	2.9167	97.40	0.65	0.2500	-0.47
jaccard	2.9583	94.90	0.65	0.2500	-0.43
chebyshev	7.0625	171.10	0.80	0.5833	-0.09

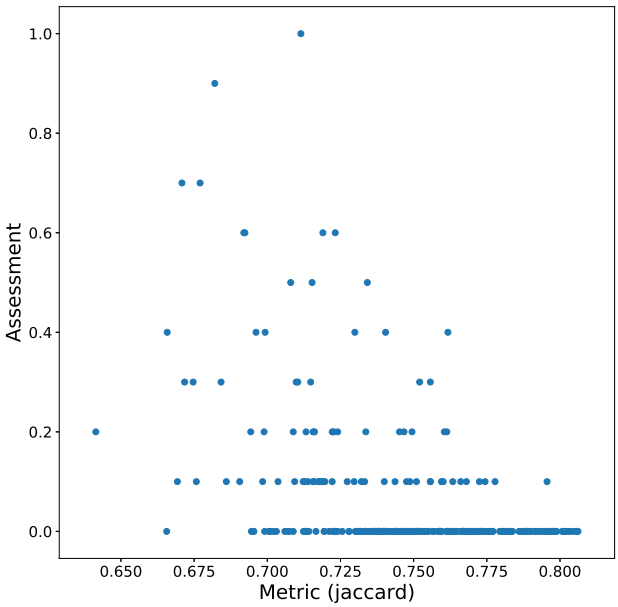


(a) Относительное качество

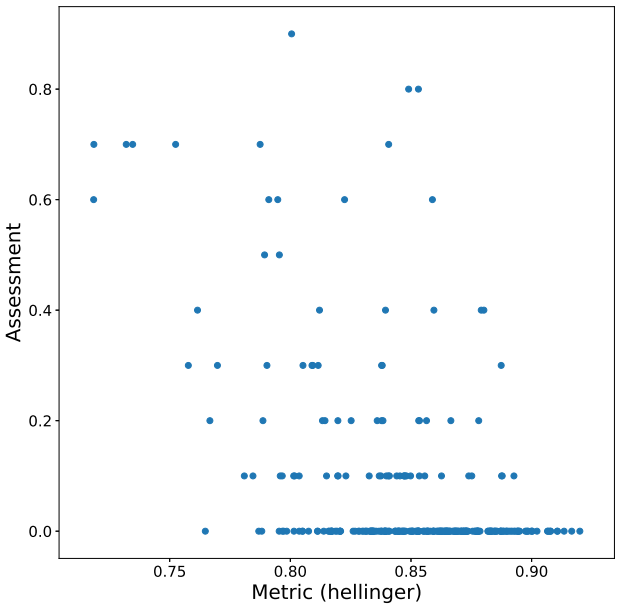


(b) Время работы

Рис. 4: Сравнение алгоритмов LKH и отжига



(a) postnauka



(b) lenta

Рис. 5: Связь метрик с асессорскими оценками

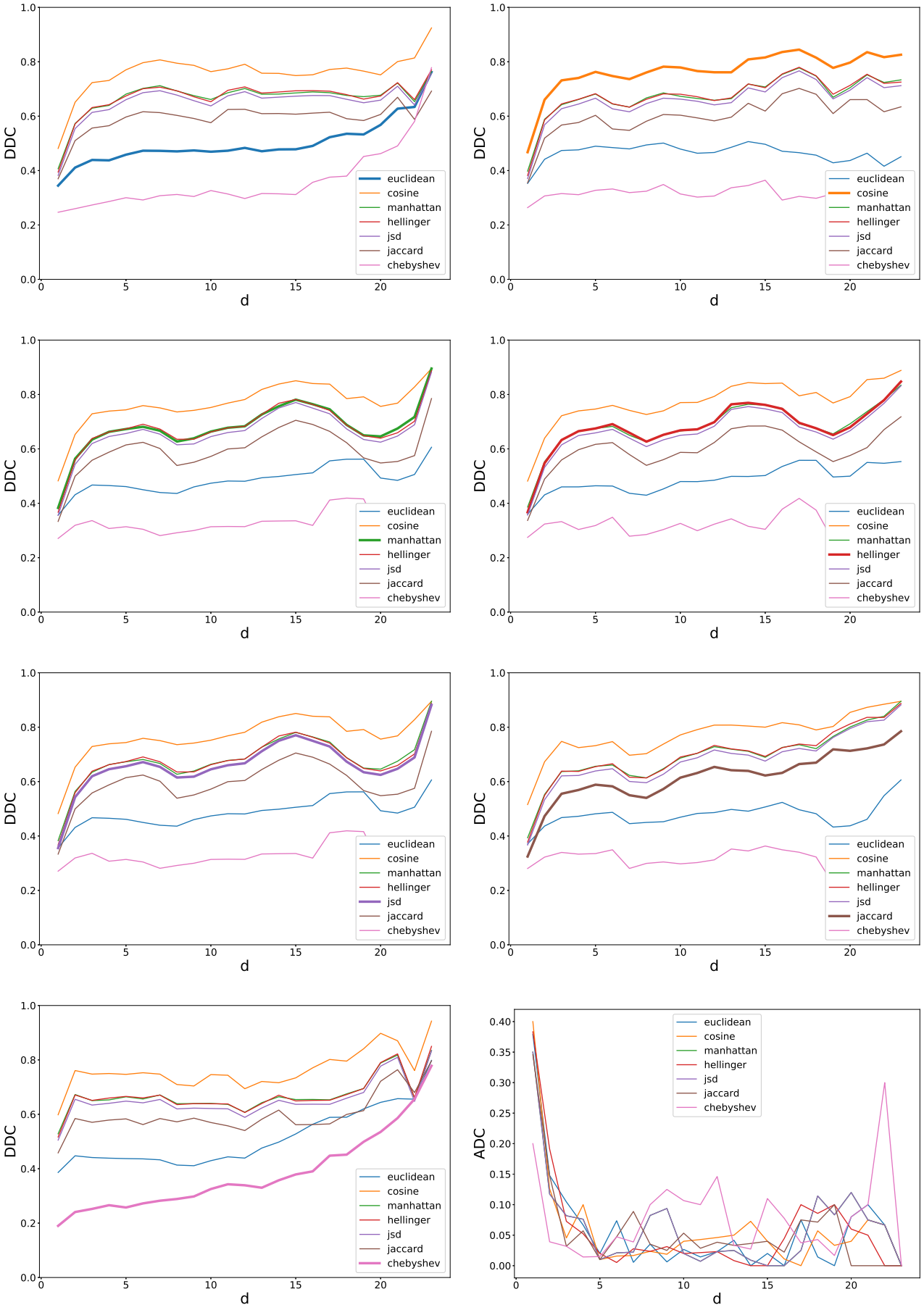


Рис. 6: Кривые DDC и ADC (lenta)

1. остров, земля, период, там, территория, океан, где, более, вид, найти, вулкан, находиться, южный
2. растение, япония, раса, при, более, чем, например, исследование, вид, страна, население
3. вид, эволюция, самец, мозг, самка, животное, отбор, ген, более, птица, наш, между, чтобы, чем, друг
4. мозг, нейрон, при, заболевание, наш, пациент, состояние, система, болезнь, сон, исследование
5. клетка, музей, стволовой, ткань, организм, чтобы, опухоль, система, использовать, технология
6. клетка, ген, днк, организм, молекула, геном, белок, белка, бактерия, система, процесс, жизнь
7. система, материал, задача, структура, метод, компьютер, дать, при, химический, область, химия
8. квантовый, свет, волна, атом, информация, фотон, сигнал, использовать, два, при, частота, состояние
9. частица, энергия, кварк, взаимодействие, магнитный, электрон, масса, физика, бозон, протон, модель
10. звезда, галактика, земля, планета, вселенная, дыра, чёрный, объект, солнце, масса, наш, система
11. теория, пространство, вселенная, закон, физика, математический, уравнение, число, два, мир, система
12. наш, сеть, информация, дать, объект, культура, задача, например, образ, память, слово, разный
13. язык, слово, русский, например, говорить, словарь, речь, разный, языковой, текст, два, лингвист
14. наука, учёный, научный, потому, чтобы, лекция, хороший, университет, сейчас, наш, заниматься
15. экономический, экономика, страна, чтобы, более, рынок, компания, цена, решение, деньга, работа, чем
16. страна, война, государство, политический, россия, советский, власть, политика, германия, стать
17. ребёнок, женщина, мужчина, жизнь, культура, общество, себя, семья, социальный, советский, женский
18. город, пространство, социальный, городской, общество, место, культурный, жизнь, более, современный
19. исследование, социальный, поведение, группа, решение, and, the, теория, проблема, наука
20. социальный, социология, мир, теория, объект, социологический, действие, событие, социолог, наука
21. политический, философия, идея, наука, свобода, понятие, революция, история, философ, век, себя
22. право, власть, закон, король, век, римский, бог, себя, церковь, правовой, политический, суд, два
23. век, история, русский, исторический, имя, традиция, христианский, культура, историк, текст, уже
24. себя, искусство, литература, говорить, потому, мир, сам, миф, жизнь, слово, текст, роман, век
25. книга, фильм, автор, кино, pcourse, num, читатель, посвятить, тема, история, исследование, работа

(a) postnauka, LKH, расстояние Жаккара

1. спортсмен, допинг, олимпиада, рио, де, россия, проба, жанейро, wada, олимпийский_игра, соревнование
2. команда, матч, счёт, клуб, победа, чемпионат, турнир, минута, футболист, встреча, летний, футбол
3. евро, евровидение, страна, россия, конкурс, франция, болельщик, англия, украина, футбол, певец
4. пройти, мероприятие, россия, акция, фестиваль, москва, фильм, участник, картина, театр, музей
5. фильм, сериал, продукт, актёр, компания, продукция, процент, россия, книга, товар, картина, сезон
6. россия, москва, турист, процент, россиянин, страна, отель, рейс, путешественник, город, тысяча
7. процент, доллар, рубль, нефть, цена, россия, баррель, страна, уровень, вырасти, рынок, рост
8. компания, миллиард_рубль, процент, миллиард_доллар, россия, сумма, миллион_доллар, банк, банка
9. закон, законопроект, документ, реклама, использование, деятельность, поправка, внести, организация
10. россия, страна, керченский_пролив, российский, боинг, работа, чайка, ряд, гражданин, аэропорт
11. партия, кандидат, журналист, праймериза, выбор, единый_россия, госдума, выборы
12. россия, украина, крым, решение, киев, депутат, вопрос, отношение, страна, мнение, право, москва
13. россия, страна, турция, сша, ес, евросоюз, москва, санкция, отношение, украина, вопрос, государство
14. россия, сирия, исламский_государство, сша, нато, иго, запретить, террорист, страна, боевик
15. ракета, путин, россия, запуск, глава_государство, союз, спутник, президент
16. учёный, клетка, исследование, исследователь, ген, университет, оказаться, процент, помощь, организм
17. земля, животное, учёный, животный, тысяча, звезда, планета, обнаружить, кошка, территория, жизнь
18. самолёт, километр, машина, борт, пассажир, вертолёт, погибнуть, лайнер, пилот, час, район, яхта
19. полицейский, полиция, мужчина, задержать, автомобиль, улица, москва, пострадать, life
20. статья, убийство, задержать, суд, отношение, ук_рф, подозревать, следствие, обвинять, трамп, часть
21. ребёнок, женщина, мужчина, летний, дом, сын, семья, мальчик, жена, полиция, дочь, школа, врач
22. видео, youtube, ролик, фото, фотография, канал, снимка, auto, instagram, девушка, страница, группа
23. facebook, пользователь, интернет, страница, twitter, пост, написать, соцсеть, вконтакте, аккаунт
24. устройство, смартфон, компания, мотоциклист, игра, байкер, видео, миллион_доллар, робот, молодая
25. бренд, модель, компания, обувь, основать, одежда, релиз, коллекция, редакция, часы, поступить

(b) lenta, LKH, расстояние Хеллингера

Рис. 7: Примеры тематических спектров

5 Тематический спектр для иерархических моделей

5.1 Постановка задачи

Пусть теперь имеется двухуровневая тематическая модель. То есть есть два множества тем T_1 и T_2, для каждого из которых задана своя матрица расстояний (R_1 и R_2), и кроме того, задана матрица иерархии $H \in \mathbb{R}^{|T_1|\times|T_2|}$, в которой $H[t,s]=1$, если тема s нижнего уровня считается дочерней темой темы t верхнего уровня, и $H[t,s]=0$ иначе.

Предлагается визуализировать такую модель в виде двух списков тем (рис. 10). Темы нижнего уровня предлагается соединять с их родительскими темами на верхнем уровне.

Возникает сразу три функционала, которые нужно оптимизировать. Это суммы расстояний между соседями на обоих уровнях:

$$NDS(\pi_1) = \sum_{t=1}^{|T_1|-1} R_1[\pi_1[t], \pi_1[t+1]]; \tag{5.1}$$

$$NDS(\pi_2) = \sum_{s=1}^{|T_2|-1} R_2[\pi_2[s], \pi_2[s+1]], \tag{5.2}$$

а также число пересечений рёбер в графе спектра (Spectrum Crosses Count).

$$SCC(\pi_1,\pi_2) = \sum_{t_1<t_2}\sum_{s_1<s_2} H[t_1,s_1]H[t_2,s_2]\Big[(\pi_1[t_1] < \pi_1[t_2]) \veebar (\pi_2[s_1] < \pi_1[s_2])\Big]. \tag{5.3}$$

Поясним формулу (5.3). Перебираются все возможные пары рёбер $\big((t_1,s_1),(t_2,s_2)\big)$ в графе спектра (условия $t_1 < t_2$ и $s_1 < s_2$ накладываются для того, чтобы каждое ребро учесть ровно один раз и не учитывать пары рёбер, имеющих общую вершину), и для каждой пары проверяется, пересекаются ли эти рёбра. Они пересекаются, если концы рёбер идут в разном порядке на первом и втором уровне, т.е. если $\Big(\pi_1[t_1] < \pi_1[t_2] \wedge (\pi_2[s_1] > \pi_1[s_2])\Big)$ или $\Big(\pi_1[t_1] > \pi_1[t_2] \wedge (\pi_2[s_1] < \pi_1[s_2])\Big)$, что можно кратко записать с использованием операции «исключающее или».

Формально задача построения тематического спектра для иерархической ТМ ставится следующим образом:

$$\Big(NDS(\pi_1), NDS(\pi_2), SCC(\pi_1,\pi_2)\Big) \to \min_{\pi_1,\pi_2}. \tag{5.4}$$

5.2 Построение иерархического спектра

Предлагается решать задачу (5.4), поднимаясь снизу вверх, то есть сначала упорядочить темы на нижнем уровне, а потом, зафиксировав эту перестановку, упорядочить темы на верхнем уровне. Но на нижнем уровне будем решать задачу, учитывающую иерархию. Для этого модифицируем матрицу D_2, умножив расстояния между темами, у которых есть хотя бы один общий родитель, на некоторый коэффициент $\beta \in (0,1)$:

$$D'_2[s,s'] = \begin{cases} \beta D_2[s,s'], & \text{если } \exists t \in T_1 : (H[t,s]=1) \wedge (H[t,s']=1), \\ D_2[s,s'], & \text{иначе.} \end{cases} \tag{5.5}$$

Теперь найдём перестановку $\hat{\pi}_2 = \arg\min\limits_{\pi_2} NDS(\pi_2)$, (например, алгоритмом LKH, описанным в предыдущей главе). Затем, зафиксировав перестановку тем на нижнем уровне, найдём перестановку тем верхнего уровня, минимизирующую число пересечений:

$$\hat{\pi}_1 = \arg\min_{\pi_1} SCC(\pi_1, \hat{\pi}_2). \tag{5.6}$$

5.3 Решение задачи о минимизации числа пересечений

Решение задачи (5.6) рассмотрено в [28]. В данной работе рассмотрим её точное решение с помощью сведения к задаче целочисленного программирования, предложенного в [29] и три эвристики.

5.3.1 Точное решение

Сначала для удобства переформулируем задачу. Будем говорить о темах как о вершинах двудольного графа. Пусть первая доля — это темы верхнего уровня, а вторая доля — темы нижнего уровня. Пусть $N = |T_1|$, $M = |T_2|$. Так как на нижнем уровне перестановка вершин фиксирована, будем считать, что вершины нижнего уровня имеют номера $1, 2, \ldots, M$. Тогда задача принимает вид

$$SCC(\pi) = \sum_{i_1=1}^{N} \sum_{i_2=i_1+1}^{N} \sum_{j_1=1}^{M} \sum_{j_2=1}^{j_1-1} H[\pi(i_1), j_1] H[\pi(i_2), j_2] \to \min_{\pi}. \tag{5.7}$$

Заметим, что для каждой пары рёбер их пересечение зависит только от взаимного расположения вершин первой доли. Поэтому сумма всех пересечений состоит из суммы независимых вкладов от каждой пары вершин первой доли. Введём матрицу $C \in \mathbb{R}^{N \times N}$, в которой C_{ij} — это вклад в общее число пересечений от рёбер, выходящих из вершин i и j при условии, что i идёт раньше, чем j, то есть

$$C_{ij} = \begin{cases} \sum\limits_{k=1}^{M} H[i,k] \sum\limits_{l=1}^{k-1} H[j,l], & \text{если } i \neq j \\ 0, & \text{если } i = j \end{cases} \tag{5.8}$$

Введём переменные x_{ij} для $i, j \in \overline{1, N}$, такие что:

$$x_{ij} = \begin{cases} 1, & \text{если } \pi^{-1}(i) < \pi^{-1}(j); \\ 0, & \text{если } \pi^{-1}(i) \geq \pi^{-1}(j). \end{cases} \tag{5.9}$$

Другими словами, x_{ij} — это индикатор того, что в перестановке π тема i предшествует теме j.

Тогда

$$SCC(\pi) = \sum_{i=1}^{N} \sum_{j=1}^{N} x_{ij} C_{ij}. \tag{5.10}$$

Теперь рассмотрим, какие значения x_{ij} допустимы. Прежде всего, они могут принимать только значение 0 или 1. Далее, для любых i, j, таких, что $i \neq j$ или вершина i

предшествует вершине j или наоборот, т.е. ровно одно из чисел x_{ij}, x_{ji} равно 1, а другое равно 0, т.е. $x_{ij} + x_{ji} = 1$. Кроме того, отношение предшествования должно быть транзитивным, т.е. из $x_{ij} = 1$ и $x_{jk} = 1$ должно следовать $x_{ik} = 1$. Это значит, что в любой тройке x_{ij}, x_{jk}, x_{ki} числа не могут быть одновременно равны нулю или единице, из чего следует, что $1 \leq x_{ij} + x_{jk} + x_{ki} \leq 2$.

Итак, имеем следующую оптимизационную задачу:

$$\begin{cases} \sum\limits_{i=1}^{N}\sum\limits_{j=1}^{N} x_{ij}C_{ij} \to \min\limits_{\boldsymbol{x}}; \\ x_{ij} \in \{0,1\}; \\ x_{ij} + x_{ji} = 1 \ \forall i,j \in \overline{1,N}, i \neq j; \\ 1 \leq x_{ij} + x_{jk} + x_{ki} \leq 2 \ \forall i,j,k \in \overline{1,N}, i \neq j \neq k. \end{cases} \tag{5.11}$$

Перепишем выражение (5.10) с учётом того, что $x_{ii} = 0$ и $x_{ij} + x_{ji} = 1$:

$$\begin{aligned} SCC(\pi) = \sum_{i<j} x_{ij}C_{ij} + \sum_{i>j} x_{ij}C_{ij} = \sum_{i<j} \Big[x_{ij}C_{ij} + x_{ji}C_{ji}\Big] = \\ = \sum_{i<j} \Big[x_{ij}C_{ij} + (1 - x_{ij})C_{ji}\Big] = \sum_{i<j} \Big[x_{ij}(C_{ij} - C_{ji})\Big] + \underbrace{\sum_{i<j} C_{ji}}_{\text{константа}}. \end{aligned} \tag{5.12}$$

Оказывается, целевой функционал можно выразить только через те x_{ij}, для которых $i < j$. При этом можно также избавиться от всех ограничений вида $x_{ij} + x_{ji} = 1$, а ограничения вида $1 \leq x_{ij} + x_{jk} + x_{ki} \leq 2$ переписать, подставив $x_{ki} = 1 - x_{ik}$. Итак, задаче (5.11) эквивалентна задача

$$\begin{cases} \sum\limits_{i=1}^{N}\sum\limits_{j=1+1}^{N} x_{ij}(C_{ij} - C_{ji}) \to \min\limits_{\boldsymbol{x}}; \\ x_{ij} \in \{0,1\}; \\ 0 \leq x_{ij} + x_{jk} - x_{ik} \leq 1 \ \forall (i,j,k) : 1 \leq i < j < k \leq N. \end{cases} \tag{5.13}$$

У этой задачи $\frac{N(N-1)}{2}$ переменных и $\frac{N(N-1)(N-2)}{3}$ ограничений. Это — задача целочисленного линейного программирования [30]. В данной работе она решается с использованием решателя CBC MILP Solver [31], который использует метод ветвей и границ.

Когда найдено решение задачи , легко восстановить начальную перестановку. Сначала восстановим все элементы матрицы инцидентности: вычислим $x_{ji} = 1 - x_{ij}$ для всех $i, j : i < j$. Теперь x — это матрица инцидентности некоторого ациклического графа. Если выполнить в нём топологическую сортировку, получим искомую перестановку $\hat{\pi}$.

Однако её можно получить ещё проще. Заметим, что для i-й вершины в оптимальной перестановке $(i-1)$ вершины идёт до неё и $N - i$ вершин идут после неё. Значит, в соответствующей ей строке матрицы x находится ровно $(N - i)$ единиц. Поэтому

$$\hat{\pi} = \operatorname{argsort}\Big(\big(-\sum_{j=1}^{N} x_{ij}\big)_{i \in \overline{1,N}}\Big), \tag{5.14}$$

где $\operatorname{argsort}(x_1, \ldots, x_N)$ — такая перестановка $\hat{\pi}$ на N элементах, что

$$\forall i,j \quad \hat{\pi}_i^{-1} < \hat{\pi}_j^{-1} \Leftrightarrow i < j. \tag{5.15}$$

5.3.2 Эвристика барицентров

Вычислим для каждой темы верхнего уровня барицентр — среднее значение номера дочерней темы.

$$B(i) = \frac{\sum_{j=1}^{M} H[i,j] \cdot j}{\sum_{j=1}^{M} H[i,j]} \tag{5.16}$$

Тогда определим

$$\hat{\pi} = \operatorname{argsort}\Big(B(1), B(2), \dots B(|T_1|)\Big), \tag{5.17}$$

5.3.3 Эвристика медиан

Вычислим для каждой темы верхнего i уровня медиану номеров дочерних тем

$$M(i) = \operatorname{median}(\{j | H[i,j] = 1\}), \tag{5.18}$$

и найдём оптимальную перестановку как

$$\hat{\pi} = \operatorname{argsort}\Big(M(1), M(2), \dots M(|T_1|)\Big). \tag{5.19}$$

5.3.4 Эвристика быстрой сортировки

Введём на темах отношение $Q(i,j) = [C_{ij} \leq C_{ji}]$. Если бы темы можно было упорядочить полностью без пересечений, то это упорядочивание можно было бы получить сортировкой, которая использует Q вместо отношения «меньше или равно».

Действительно, пусть $\hat{\pi}$ — такая перестановка, что $\forall i,j\ (\pi^{-1}(i) \leq \pi^{-1}(j)) \to (C_{ij} \leq C_{ji})$. Но если возможна перестановка без пересечений, то для любой пары чисел C_{ij} и C_{ji} одно из них равно нулю. Значит, $\forall i,j\ (\pi^{-1}(i) \leq \pi^{-1}(j)) \to (C_{ij} = 0)$. Тогда $SCC(\hat{\pi}) = 0$ по формуле (5.10).

Предлагается провести такую сортировку в общем случае. Для этого модифицируем алгоритм быстрой сортировки. Рассмотрим следующую рекурсивную процедуру:

Функция Sort(C, π, i, j)

1: Если $i > j$ — выход;
2: $p \leftarrow \operatorname{Random}(\overline{i,j})$;
3: $A \leftarrow \Big\{\pi[k] \Big| k \in \overline{i,j}, k \neq p, C[\pi[k],\pi[p]] < C[\pi[p],\pi[k]]\Big\}$;
4: $B \leftarrow \Big\{\pi[k] \Big| k \in \overline{i,j}, k \neq p, C[\pi[k],\pi[p]] \geq C[\pi[p],\pi[k]]\Big\}$;
5: $\pi[i,\dots,j] \leftarrow A, \pi[p], B$;
6: $\operatorname{Sort}(C, \pi, i, i + |A| - 1)$;
7: $\operatorname{Sort}(C, \pi, j - |B| + 1, j)$.

Эта процедура переставляет элементы некоторого отрезка перестановки. Вначале произвольно выбирается элемент в этом отрезке, слева от него помещаются элементы, «меньшие» его в смысле отношения Q, справа — «большие». Затем процедура повторяется для полученных двух подотрезков.

Чтобы использовать этот алгоритм, нужно создать массив $\pi = [1, 2, \ldots, N]$ и вызвать Sort(C, π, 1, N).

5.4 Обобщение на большее число уровней

Если в иерархии $N > 2$ уровней, предлагается следующий алгоритм. Модифицируем матрицу D_N по формуле (5.5), используя матрицу иерархии между двумя последними уровнями, и упорядочим темы на последнем уровне в соответствии с этой матрицей. Затем будем подниматься вверх, и упорядочивать темы i-го уровня при помощи принципа центра масс, считая центры масс по номерам тем $(i+1)$-го уровня.

5.5 Эксперименты

5.5.1 Сравнение алгоритмов минимизации числа пересечений

Описанные в 5.3 алгоритмы сравнивались на модельных данных. Создавался двудольный граф с $N \in \overline{5, 100}$ вершинами в первой доле и $M = 500$ вершинами во второй доле. В него было случайно добавлено $\frac{NM}{100}$ рёбер. Для него точно решалась задача о минимальном количестве пересечений (с помощью CBC MILP Solver). Затем находилось минимальное количество пересечений с помощью эвристик барицентров, медиан и быстрой сортировки. Также применялась эвристика QuickSort-10N, когда $10N$ раз искалась оптимальная перестановка с помощью эвристики быстрой сортировки, и из них выбиралась та, которая обеспечивала минимальное количество пересечений рёбер.

На рис. 8a приведены относительные ошибки эвристик, равные $\frac{SCC}{SCC_{opt}}$, где SCC — количество пересечений, найденное эвристикой, а SCC_{opt} — минимальное количество пересечений, найденное при точном решении оптимизационной задачи.

На рис. 8b приведено время работы CBC MILP Solver и эвристики QuickSort-10N. Видно, что время решения оптимизационной задачи зависит не только от N, но и от данных, и в некоторых случаях может быть намного больше, чем в среднем.

В VisARTM при $N \leq 50$ количество пересечений минимизируется точно с помощью CBC MILP Solver, а при $N > 50$ — приближённо при помощи эвристики QuickSort-10N.

5.5.2 Поиск оптимального параметра β

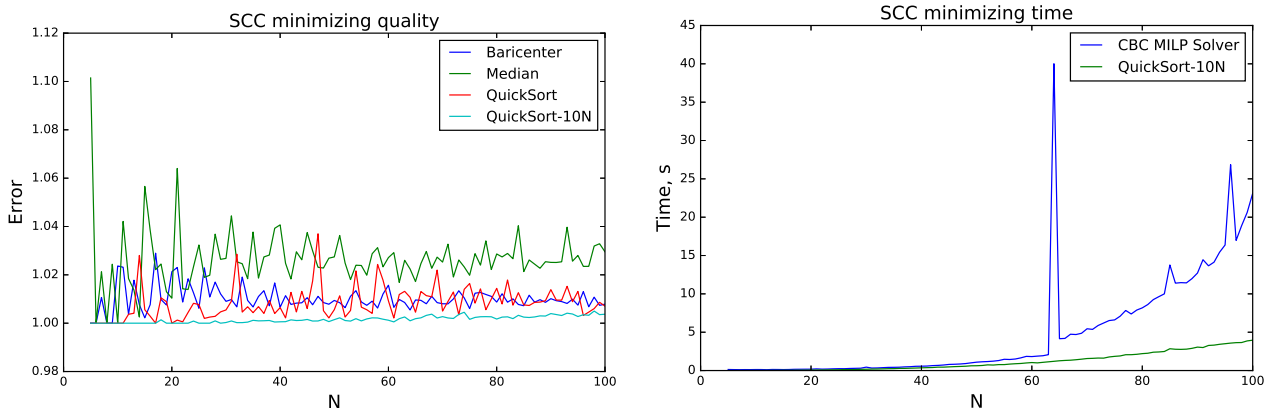
Для коллекций postnauka и lenta были построены иерархические тематические модели из двух уровней (с 10 темами на первом уровне и 30 темами на втором уровне). Для них были построены двухуровневые спектры при разных значениях параметра β (в диапазоне от -0.2 до 0.2). Число пересечений минимизировалось точным алгоритмом. На рис. 9 показаны зависимости трёх мер качества (суммарное расстояния между соседями на обоих уровнях NDS_1 и NDS_2, а также число пересечений SCC) от параметра β.

NDS_2 достигает своего минимума при $\beta = 1$, потому что в этом случае оптимизируется только этот функционал, а два других игнорируются. При уменьшении β до 0.9 NDS_2 увеличивается, но уменьшаются NDS_1 и SCC. При этом SCC очень быстро падает до нуля (то есть удаётся разместить темы в спектре вообще без пересечений рёбер). При $\beta \in [0.5, 0.9]$ все три показателя качества не меняются. При меньших значениях

показатель NDS_2 увеличивается, что значит, что темы на нижнем уровне перемешиваются. Оптимальными являются значения $\beta \in [0.5, 0.9]$. В VisARTM по умолчанию используется $\beta = 0.8$.

5.5.3 Примеры спектров

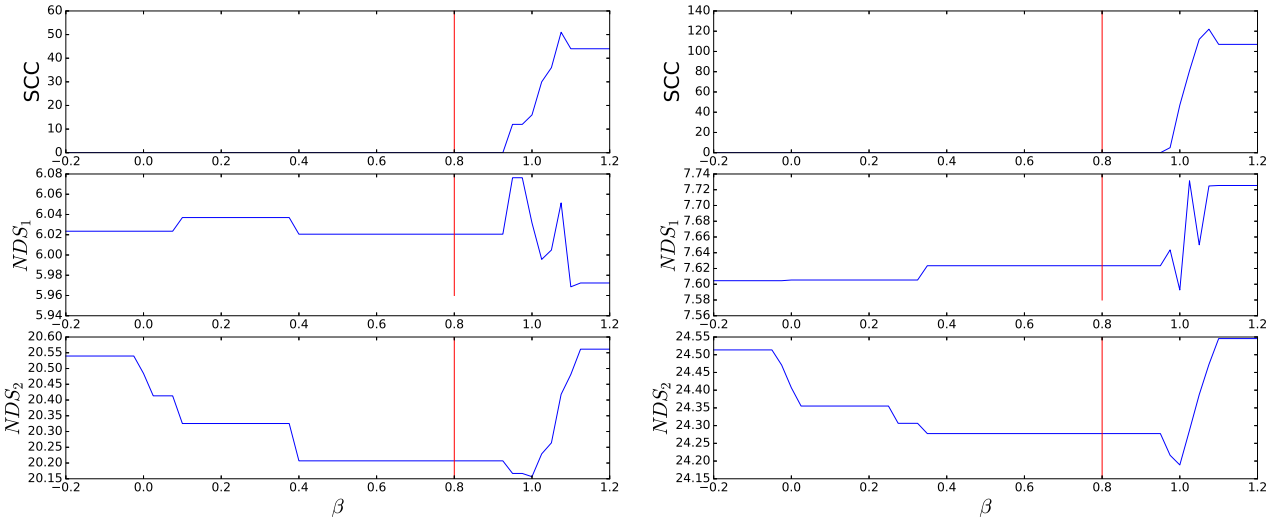
На рис. 10 изображёны графы двухслойных тематических спектров для коллекций postnauka и lenta.



(a) Относительная ошибка эвристик

(b) Время работы

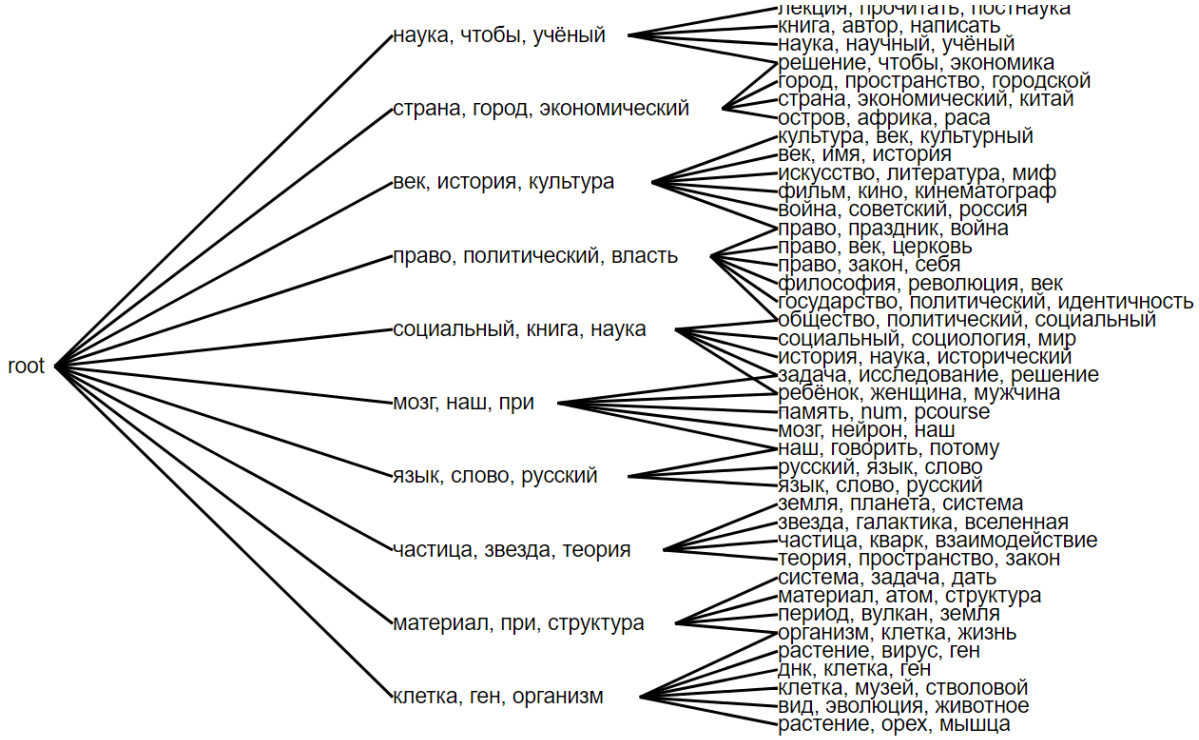
Рис. 8: Сравнение алгоритмов минимизации числа пересечений рёбер в спектре



(a) postnauka, jaccard

(b) lenta, jaccard

Рис. 9: Зависимость мер качества иерархического спектра от параметра β



(a) potnauka

(b) lenta

Рис. 10: Примеры иерархических тематических спектров

6 Заключение

В данной работе были поставлены задачи построения тематического спектра для плоских и иерархических тематических моделей. Было предложено несколько алгоритмов решения этих задач и разработаны методики оценивания качества спектров. На основании экспериментов сделан вывод, что лучше всего решать задачу построения тематического спектра, сводя её к задаче коммивояжёра, и решая последнюю алгоритмом Лина-Кернигана-Хельсгауна.

Реализована информационная система для визуализации тематических моделей VisARTM, в которую интегрированы разработанные алгоритмы построения тематического спектра.

Список литературы

[1] Айсина Р.М. Обзор средств визуализации тематических моделей коллекций текстовых документов // Машинное обучение и анализ данных. — 2015. — Т. 1, № 11. — С. 1584–1618.

[2] Воронцов К.В. Тематическое моделирование в BigARTM: теория, алгоритмы, приложения. — 2015. — URL: http://www.machinelearning.ru/wiki/images/b/bc/Voron-2015-BigARTM.pdf.

[3] Hofmann Thomas. Probabilistic latent semantic indexing // Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval / ACM. — 1999. — P. 50–57.

[4] Dempster Arthur P, Laird Nan M, Rubin Donald B. Maximum likelihood from incomplete data via the EM algorithm // Journal of the royal statistical society. Series B (methodological). — 1977. — P. 1–38.

[5] Воронцов К.В., Потапенко А.А. Модификации EM-алгоритма для вероятностного тематического моделирования // Машинное обучение и анализ данных. — 2013. — Т. 1, № 6. — С. 657–686.

[6] Воронцов К.В. Аддитивная регуляризация тематических моделей коллекций текстовых документов // Доклады РАН. — Т. 455. — 2014. — С. 268–271.

[7] Blei David M, Ng Andrew Y, Jordan Michael I. Latent dirichlet allocation // Journal of machine Learning research. — 2003. — Vol. 3, no. 1. — P. 993–1022.

[8] Chirkova NA, Vorontsov KV. Additive Regularization for Hierarchical Multimodal Topic Modeling // JMLDA. — 2016. — Vol. 2, no. 2.

[9] BigARTM: библиотека с открытым кодом для тематического моделирования больших текстовых коллекций / К.В. Воронцов [и др.] // Аналитика и управление данными в областях с интенсивным использованием данных. XVII Международная конференция DAMDID/RCDL. — 2015.

[10] Online documentation for BigARTM. — URL: http://docs.bigartm.org/en/stable/index.html.

[11] Chaney Allison June-Barlow, Blei David M. Visualizing Topic Models. // ICWSM. — 2012.

[12] Shneiderman Ben. The eyes have it: A task by data type taxonomy for information visualizations // Visual Languages, 1996. Proceedings., IEEE Symposium on / IEEE. — 1996. — P. 336–343.

[13] Online documentation for FoamTree. — URL: https://get.carrotsearch.com/foamtree/latest/api/index.html.

[14] The Django book. — URL: http://djangobook.com/.

[15] URL: http://getbootstrap.com/.

[16] scikit-learn: Machine Learning in Python.— URL: http://scikit-learn.org/stable/.

[17] LKH Algorithm.— URL: http://www.akira.ruc.dk/~keld/research/LKH/.

[18] D3.js — Data-Driven Documents.— URL: https://d3js.org/.

[19] Репозиторий VisARTM.— URL: https://github.com/bigartm/visartm.

[20] VisARTM documentation.— URL: http://visartm.vdi.mipt.ru/docs.

[21] Karla L. Hoffman Manfred Padberg Giovanni Rinaldi. Travelling Salesman Problem // Encyclopedia of Operations Research and Management Science.— Springer, 2013.— P. 1573–1578.

[22] Lance Godfrey N, Williams William Thomas. A general theory of classificatory sorting strategies II. Clustering systems // The computer journal.— 1967.— Vol. 10, no. 3.— P. 271–277.

[23] Kruskal Joseph B. Multidimensional scaling by optimizing goodness of fit to a nonmetric hypothesis // Psychometrika.— 1964.— Vol. 29, no. 1.— P. 1–27.

[24] Maaten Laurens van der, Hinton Geoffrey. Visualizing data using t-SNE // Journal of Machine Learning Research.— 2008.— Vol. 9, no. 11.— P. 2579–2605.

[25] Suman Balram. Simulated Annealing // Encyclopedia of Operations Research and Management Science.— Springer, 2013.— P. 1395–1404.

[26] Aarts Emile HL, Korst Jan HM, van Laarhoven Peter JM. A quantitative analysis of the simulated annealing algorithm: A case study for the traveling salesman problem // Journal of Statistical Physics.— 1988.— Vol. 50, no. 1-2.— P. 187–206.

[27] Helsgaun Keld. An effective implementation of the Lin–Kernighan traveling salesman heuristic // European Journal of Operational Research.— 2000.— Vol. 126, no. 1.— P. 106–130.

[28] З.В. Апанович. Методы визуализации информации при помощи графов.— URL: http://mmc2.nsu.ru/default.aspx?db=book_apanovich2&int=VIEW&el=1897&templ=I206.

[29] A branch-and-cut approach to physical mapping of chromosomes by unique end-probes / Thomas Christof [et al.] // Journal of Computational Biology.— 1997.— Vol. 4, no. 4.— P. 433–447.

[30] Karla L. Hoffman Ted K. Ralphs. Integer and Combinatorial Optimization // Encyclopedia of Operations Research and Management Science.— Springer, 2013.— P. 771–783.

[31] CBC User Guide.— URL: https://www.coin-or.org/Cbc/.