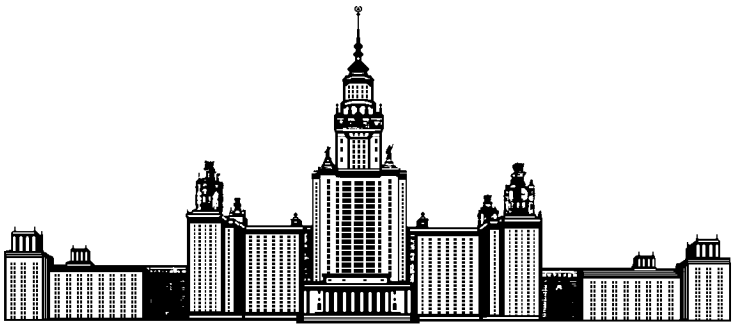
Московский государственный университет имени М. В. Ломоносова
Факультет Вычислительной Математики и Кибернетики
Кафедра Математических Методов Прогнозирования



Новоселов Вадим Владимирович

Темпоральные тематические модели новостных потоков с возможностью обнаружения новых тем и событий

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА

Научный руководитель:
Воронцов Константин Вячеславович

Москва, 2021

Содержание

1 Введение

Методы машинного обучения в последнее время находят всё более разнообразные применения в автоматической обработке новостных потоков. Они используются для решения таких задач, как обнаружение и отслеживание тем [1, 19, 30, 12], связывание новостных статей из различных источников [37, 39], распознавание новых сюжетов [34, 29, 35], определение противоречий, разногласий или поляризованности [36], описание тематических шаблонов в публикациях [27]. Решение некоторых из этих задач требует формирования больших корпусов размеченных новостных сообщений.

Целью данной работы является проверка гипотезы, что кластеризация новостей по событиям с последующей кластеризацией событий по темам может быть выполнена с приемлемым качеством без использования разметки, то есть методами обучения без учителя. Для формализации понятий «событие» и «тема» строится выборка пар новостей, которые размечаются экспертами на три класса как относящиеся к одному событию, либо к разным событиям но одной теме, либо не имеющие отношения друг к другу. Получаемая таким способом размеченная выборка используется не для оптимизации параметров модели, а для выбора из небольшого конечного множества моделей, то есть не как обучающая, а как валидационная. В литературе событие может определяться как семантический шаблон определённой структуры внутри скользящего окна или *рамочное событие*; также событие может определяться относительно построенного по коллекции текстов представления семантического графа, такой вид событий называют *графовым*; событие, представленное в виде кластера документов, называется *документальным событием* [7]. В данной работе рассматриваются именно документальные события.

Сложность кластеризации новостей связана с тем, что число кластеров (событий и тем) изменяется во времени — постоянно появляются новые кластеры, старые со временем теряют актуальность и перестают пополняться новыми новостными сообщениями. Кроме того, кластеры, как правило, несбалансированы — в любой момент времени имеются как плотные кластеры, состоящие из большого числа сообщений, так и малочисленные кластеры, состоящие из небольшого числа сообщений или даже из единственного сообщения. Определённые проблемы может вызывать гетерогенность источников: тексты сообщений могут иметь существенно различную длину, различные стилистические и жанровые особенности. В этой ситуации алгоритмы кластеризации с фиксированным числом кластеров оказываются неприменимы. Потоковый характер данных вынуждает использовать *инкрементные* или *онлайновые* методы кластеризации. Выделение нового события или темы должно происходить полностью автоматически, без учителя, поскольку в режиме реального времени нет никакой возможности спрашивать экспертов о том, какие события или темы являются новыми, и какими ключевыми словами они описываются. Предполагается, что разметка может не только заметно отставать от реального потока, но и вообще иметь фиксированный и весьма ограниченный объём. Новые события и темы должны надёжно выделяться даже в том случае, когда в размеченной выборке не было никаких похожих на них текстов.

В описанных условиях предлагается использовать валидационную выборку лишь для *слабого обучения*, то есть для подбора небольшого числа гиперпараметров по небольшим конечным сеткам значений [46].

Основные задачи исследования:

- Разработать методику автоматического отбора пар новостей для последующей экспертной разметки (на основе активного обучения).
- Разработать инкрементный алгоритм двухуровневой иерархической кластеризации новостного потока, где на нижнем уровне выделяются события, на верхнем уровне события группируются в темы.
- Разработать методику оценивания качества кластеризации по критериям точности и полноты классификации пар новостей при сравнении с асессорской разметкой. Также предложить количественную относительную оценку для сравнения качества кластеризации с уровнем экспертов.
- Разработать метод автоматического выбора числа тем для темпоральных тематических моделей.

2 Обзор

Распознавание документальных событий заключается в построении кластеров документов, относящихся к одному и тому же событию, в связи с чем в научной литературе термины *документальное событие*, *тема*, *история* часто смешиваются, хотя определённые различия между ними имеются [7, 47, 25, 38, 9]. Термин «история» обычно используется в приложении к новостной ленте, в то время как «событие» происходит в определённом месте и в определённое время [8]. «Тема» может в свою очередь состоять из нескольких событий. Таким образом, рассматриваемая в данной работе задача распознавания документальных событий близка к задачам *обнаружения и отслеживания тем* (topic detection and tracking) [40]. Сами кластеры могут рассматриваться как цепные структуры [42, 39, 16], как полносвязные структуры на графах [14, 17, 15], как упорядоченные в хронологическом порядке деревья [23, 3] или в виде изолированных кластеров–событий [21]. Иногда коллекции новостей кластеризуются без явного учёта упорядоченности во времени [44], однако это может приводить к ошибкам при образовании кластеров–событий. Явный учёт времени необходим для отнесения к различным кластерам семантически близких новостей; например, новости про извержения вулканов, если они произошли с достаточно большим перерывом между собой. Вопрос о проведении границ кластеров в подобных случаях связан с определением понятия «событие», которое в нашем случае формализуется через экспертную разметку.

Существует несколько часто используемых на практике подходов к определению необходимого представления новостного документа и события для распознавания документальных событий. Наиболее удобным в применении к инкрементной кластеризации является использование векторных

моделей, в которых новостные документы представлены в виде векторов взвешенных частот терминов [43, 37, 10]. При этом кластеризация основана на попарном сравнении новостных векторов с помощью различных функций расстояния: Хеллингера, косинусного, манхэттеновского, Кульбака–Лейблера, Йенсена–Шеннона и других [7, 11, 28, 4]. Расстояние между новостями можно определить и через функцию от графов, порождаемых сочетаниями сущностей в сравниваемых статьях, однако такой подход приводит к дополнительным вычислительным затратам на построение отнологических отношений и сравнение графов, нивелируя преимущества инкрементного подхода [13, 2, 22, 24]. Также в качестве признаков могут выступать векторы низкой размерности, построенные на принципах дистрибутивной семантики. Это позволяет частично решить проблему «проклятия размерности» векторных моделей, однако в случае отсутствия предварительно обученных моделей придётся постоянно обновлять векторные модели, поддерживая их актуальность [6, 31, 20]. Также можно снижать размерность другими методами, например, с помощью хэширования, в котором основной проблемой является подбор удачного семейства хэш–функций для максимизации полезной информации в хэшированных вложениях [33, 32]. Как аналог модели снижения размерности можно рассмотреть применение тематического моделирования для описания кластеров событий [27, 5, 26, 50]. Однако для онлайнового применения тематического моделирования требуется использование масштабируемой модели наподобие процесса Дирихле или онлайновой аддитивной тематической модели и их иерархических вариантов. Это приводит к увеличению времени инкрементного обучения по сравнению с более простыми методами, использующими частотные векторы и адаптивно настраиваемые пороговые правила [51, 48, 41, 49, 18].

3 Описания алгоритмов

3.1 Алгоритм выделения цепочек

На первом (нижнем) уровне иерархической кластеризации строятся цепочки новостей, связанных общим событием. Новостные сообщения о событии образуют цепочку в хронологическом порядке.

В данном разделе описывается алгоритм выделения цепочек новостей, связанных одним событием, на основе функции расстояния между текстами. Этот алгоритм не использует данные о разметке пар новостей, поэтому является алгоритмом обучения без учителя.

3.1.1 Описание алгоритма

Пусть на очередном шаге алгоритма имеется k цепочек: $C = \{C_1, C_2, ..., C_k\}$. Каждая цепочка задаётся **упорядоченным** множеством своих новостей, количество которых в каждой цепочке своё: $C_i = [d_1, d_2, ..., d_{n(i)}]$. Кроме цепочек, имеется новость $\hat{d}$, для которой нужно найти самую подходящую цепочку, либо создать новую.

Создание новой цепочки эквивалентно присоединению $\hat{d}$ к новой пустой цепочке $\hat{C}_0$. Множество всех цепочек, включая пустую, обозначается

$$\widehat{C} = \{\hat{C}_0, C_1, C_2, ...C_k\}.$$

Чтобы определить, к какой цепочке присоединять новость, необходимо определить расстояние от новости до цепочки $\rho(x, C_i)$. Определяется это расстояние через агрегирующую функцию F от расстояний $\mathrm{p}(x, d_j)$ между данной новостью x и всеми новостями d_j в цепочке:

$$\rho(x, C_i) = F\Big(\mathrm{p}(x, d_1), ..., \mathrm{p}(x, d_{n(i)})\Big).$$

Расстояние до «пустой» цепочки всегда равно некоторому порогу: $\rho(x, \hat{C}_0) = threshold$, который является гиперпараметром модели. Чем выше порог, тем ниже требования к «похожести» вектора новости и цепочки, тем реже будут создаваться новые цепочки.

Таким образом, тройка $(F, \mathrm{p}, threshold)$ задаёт параметрическую модель генерации цепочек. Сюда можно включить и способ векторизации новостей. Алгоритм поиска новой цепочки для вектора новости x выглядит следующим образом:

$$C_{new}(x) = arg\min_{c \in \widehat{C}} \Big(\rho(x, C)\Big)$$

То есть выбирается ближайшая цепочка, либо, если расстояние до $\hat{C}_0$ оказывается наименьшим, создаётся новая.

3.1.2 Гиперпараметры алгоритма

Алгоритм состоит четырёх частей:

1. Способ векторизация новостей
2. Функция расстояния p между новостями
3. Агрегирующая функция F, оценивающая расстояние между новостью и цепочкой
4. Порог $threshold$ образования новой цепочки

Способ векторизация новостей может проводиться любым алгоритмом векторизации документов. В работе сравниваются: TF-IDF, fasttext, BERT и ELMO.

Функция расстояния между новостями выбирается с учетом особенностей способа векторизации. Для подбора порога образования новой цепочки удобно взять такую метрику, которая принимает значения из $[0, 1]$. В противном случае её значения придётся дополнительно нормировать в этот промежуток.

Расстояние от новости до цепочки. В работе рассматриваются следующие функции агрегации расстояний до новостей в цепочке:

1. Среднее расстояние до всех новостей цепочки:

$$\rho(x, C_i) = \frac{1}{n(i)} \sum_k \mathrm{p}(x, d_k)$$

2. Расстояние до последнего вектора в цепочке:

$$\rho(x, C_i) = \mathrm{p}(x, d_{n(i)})$$

3. Скользящее среднее всех расстояний, начиная с конца:

$$\rho(x, C_i) = EWMA([\mathrm{p}(x, d_1), ..., \mathrm{p}(x, d_{n(i)})])$$

4. Расстояние до ближайшей новости в цепочке:

$$\rho(x, C_i) = \min_k(x, d_k)$$

Порог образования новой цепочки обозначает такое расстояние от новости до ближайшей цепочки, при превышении которого новость образует новую цепочку. Если расстояния до всех цепочек оказались выше данного порога, то создаётся новуая цепочка, содержащая единственную новость. Порог подбирается путём перебора всевозможных значений расстояний по критерию максимума качества кластеризации цепочек.

3.2 Алгоритм выделения тем

На втором уровне кластеризации цепочки событий объединяются в кластеры, на основе близости темы (далее просто «темы»). Такая кластеризация может быть как жёсткой (один документ – одна тема), так и мягкой (для каждого документа строится дискретное распределение на множестве тем). Тема объединяет в себе несколько событий, а каждое событие объединяет несколько новостей. Каждому документу присваивается тема (либо распределение тем) содержащей его цепочки, таким образом получается разбиение всего корпуса новостей по темам.

Жёсткая кластеризация Цепочки событий рассматриваются как изолированные документы, состоящие из текста всех документов, входящих в эти цепочки. Задача сводится к классической кластеризации документов, где каждый документ относится только к одному кластеру.

Существуют несколько различных алгоритмов жёсткой кластеризации, среди них можно выделить две основные группы: алгоритмы с фиксированным числом кластеров и алгоритмы с возможностью автоматически определять число кластеров. В работе сравниваются k-means, как представитель первой группы, и DBSCAN, как представитель второй.

Мягкая кластеризация отличается от жёсткой тем, что вместо отнесения цепочке к одной теме, строится целое распределение над темами для каждой цепочкой. Таким образом, одна цепочка относится ко всем темам, но с разной вероятностью.

Когда распределения тем всех цепочек являются вырожденными, мягкая кластеризация превращается в жёсткую. При этом все метрики оценки качества кластеризации сохраняются и переходят друг в друга.

Для построения мягкой кластеризации используется тематическое моделирование.

3.3 Темпоральная тематическая модель

Темпоральная тематическая модель строится в два шага:

1. Сначала применяется алгоритм построения цепочек событий над некоторым количеством новостей (например, над корпусом новостей, собранным за один день)
2. Затем, над полученными цепочками проводится мягкая кластеризация, число тем в которой определяется автоматически.

После того как коллекция пополняется новостями за новый период времени, данная процедура повторяется. Ожидается, что число тем с каждой итерацией алгоритма будет расти.

Предложенный алгоритм является инкрементным. Он не требует каждый раз перестраивать цепочки заново, достаточно просто добавлять новые документы к уже существующим цепочкам, а после этого дообучить тематическую модель на новом наборе цепочек, возможно изменив число тем.

Определение числа тем является проблемой. Нельзя утверждать, что с каждым днём число тем будет увеличиваться на одно и то же число. В реальности, темы имеют свойство как появляться, так и исчезать, при этом в разные дни могут произойти разное количество событий, которые будет связывать разное количество тем.

Таким образом, появляется необходимость автоматически подбирать число тем на новой итерации алгоритма. Для решения этой проблемы предлагается сначала применить алгоритм жёсткой кластеризации, который автоматически определяет число кластеров (например, DBSCAN), а затем дообучить мягкую кластеризацию на новом числе тем, равном числу получившихся кластеров.

У алгоритмов жёсткой кластеризации также присутствуют гиперпараметры, которые неявно задают число кластеров (например, α в DBSCAN). Такие параметры предлагается подбирать на отложенной выборке. Ожидается, что они либо не будут меняться со временем, либо их зависимость будет гораздо более простая, чем у числа тем, если бы мы оптимизировали их явно.

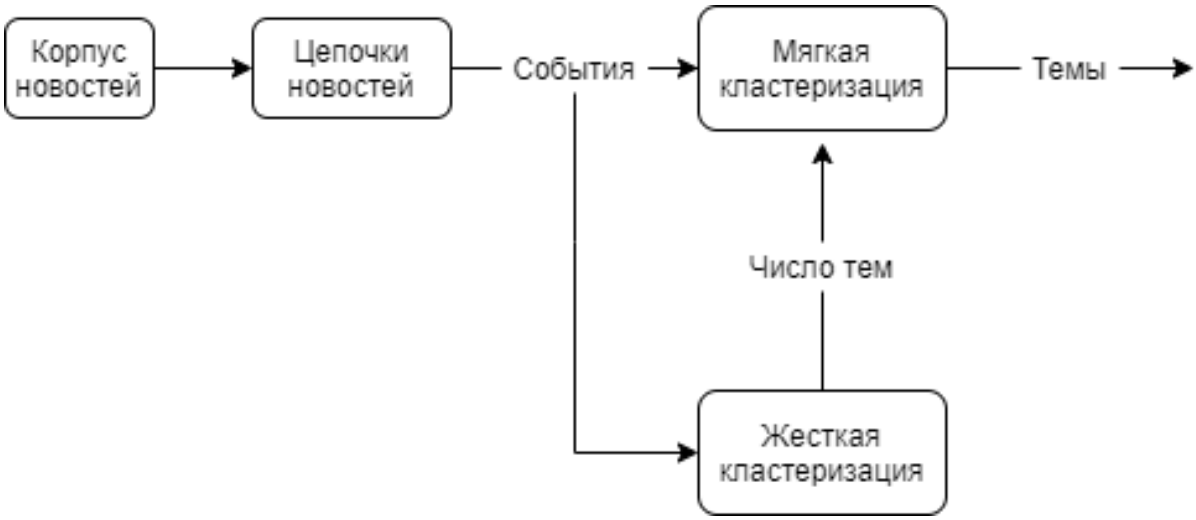


Рис. 1: Общая схема работы алгоритма темпорального тематического моделирования

4 Метрики качества

4.1 Оценки на основе разметки экспертов

Пусть $X_m = \{x_i \colon i = 1, \ldots, m\}$ — множество новостей; $y_{ijk} \in \{0, 1\}$ — оценка сходства пары новостей (x_i, x_j), поставленная экспертом $k \in K_{ij} \subset K$; K_{ij} — множество экспертов, оценивших данную пару новостей, K — множество всех экспертов. Оценка $y_{ijk} = 1$ означает, что эксперт k отнёс две новости к одному кластеру (событию).

Определим согласованную экспертную оценку пары новостей (x_i, x_j) по множеству экспертов K_{ij} как оценку, поставленную большинством экспертов:

$$y_{ij} = \Big[\sum_{k \in K_{ij}} y_{ijk} > \tfrac{1}{2} |K_{ij}| \Big]. \tag{1}$$

Чтобы голосование простым большинством было корректно, число экспертов $|K_{ij}|$ должно быть нечётным. В наших экспериментах оно всегда равнялось трём.

Есть ли необходимость вводить веса экспертов? В случае сильной несогласованности экспертов можно ввести весовые коэффициенты их надёжности w_k и заменить простое голосование (1) взвешенным:

$$y_{ij} = \big[\textstyle\sum_k w_k y_{ijk} > \frac{1}{2} \sum_k w_k \big], \tag{2}$$

оценив надёжность w_k как вероятность правильного ответа, точнее, как долю пар, для которых ответ эксперта совпадает с согласованной оценкой:

$$w_k = \frac{\sum_{i,j} \big[y_{ijk} = y_{ij} \big] \big[k \in K_{ij} \big]}{\sum_{i,j} \big[k \in K_{ij} \big]}. \tag{3}$$

Вычисления по формулам (2)–(3) производятся итерациями до сходимости [52]. Легко доказать, что в случае трёх экспертов, $|K_{ij}| = 3$, результаты простого и взвешенного голосования совпадают, если

минимальная и максимальная надёжность отличаются не более, чем в два раза. На наших данных это условие почти выполняется (Рис. 2), поэтому мы используем простое голосование.

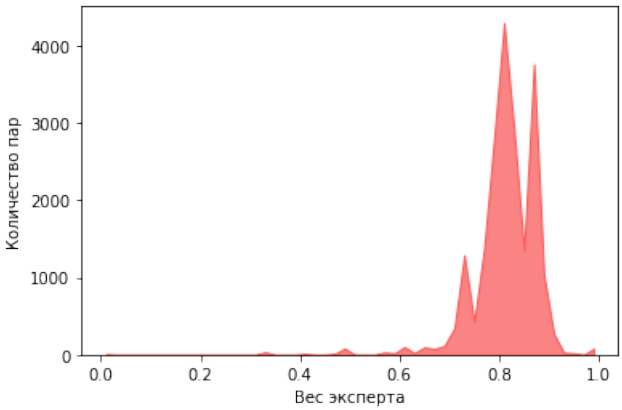


Рис. 2: Распределение надёжностей экспертов w_k, вычисленных алгоритмом (2)–(3).

Оценка качества жёсткой кластеризации. Рассмотрим алгоритм кластеризации $a\colon X \to C$, где C — множество кластеров. Обозначим через $a_{ij} = [a(x_i) = a(x_j)]$ бинарный индикатор того, что алгоритм a отнёс новости x_i, x_j к одному кластеру.

Точность кластеризации (precision) определим как долю схожих пар новостей среди всех размеченных пар, попавших в один кластер:

$$P(a) = \frac{\sum_{i,j} a_{ij} y_{ij}}{\sum_{i,j} a_{ij}}.$$

Полноту кластеризации (recall) определим как долю пар новостей, попавших в один кластер, среди всех схожих размеченных пар:

$$R(a) = \frac{\sum_{i,j} a_{ij} y_{ij}}{\sum_{i,j} y_{ij}}.$$

Низкая точность означает, что алгоритм строит слишком большие кластеры, а низкая полнота — что кластеры слишком раздроблены.

Для сравнения качества кластеризации с уровнем согласованности экспертов определим *относительную точность* (relative accuracy) как отношение доли случаев, когда решение алгоритма совпадает с решением экспертов к доле случаев, когда оценки двух экспертов совпадают:

$$A(a) = \frac{\sum_{i,j} \operatorname{avg}_k [a_{ij} = y_{ijk}]}{\sum_{i,j} \operatorname{avg}_{k,k'} [y_{ijk} = y_{ijk'}]},$$

где avg_k — усреднение по экспертам k из K_{ij}, $\operatorname{avg}_{k,k'}$ — усреднение по всем парам экспертов k, k' из K_{ij}. Если $A(a)$ превосходит 1, то можно утверждать, что алгоритм достигает человеческого уровня точности решения данной задачи.

Оценка качества вероятностной кластеризации. Рассмотрим алгоритм вероятностной (мягкой) кластеризации, который оценивает распределение вероятностей по кластерам $p(c \mid x)$, $c \in C$ для заданной новости x. Жёсткая кластеризация является частным случаем мягкой, когда $p(c \mid x) = 1$ для одного и только одного кластера c.

Определим близость кластеризации двух новостей $p(c \mid x_i)$ и $p(c \mid x_j)$ как вероятность того, что они относятся к одному кластеру:

$$a_{ij} = \sum_{c \in C} p(c \mid x_i)\, p(c \mid x_j). \tag{4}$$

При таком определении близости a_{ij} формулы для точности и полноты остаются в силе. Формула относительной точности A также остаётся в силе, если слагаемые в числителе записать в эквивалентном виде: $\big[a_{ij} = y_{ijk}\big] = a_{ij}y_{ijk} + (1 - a_{ij})(1 - y_{ijk})$, и вместо бинарных величин a_{ij} подставить вероятности (4).

5 Условные оценки качества кластеризации

Условными будем называть оценки качества, сделанные по части размеченной выборки пар новостей. Такие оценки позволяют проверять гипотезы о смещённости алгоритма в сторону ложно положительных $a_{ij} > y_{ij}$ или ложно отрицательных $a_{ij} < y_{ij}$ кластеризаций при определённых условиях. Для заданного условия $\beta_{ij} \in \{0, 1\}$ определим условные оценки точности и полноты:

$$P_\beta(a) = \frac{\sum_{i,j} \beta_{ij} a_{ij} y_{ij}}{\sum_{i,j} \beta_{ij} a_{ij}};$$
$$R_\beta(a) = \frac{\sum_{i,j} \beta_{ij} a_{ij} y_{ij}}{\sum_{i,j} \beta_{ij} y_{ij}}.$$

Если $P_\beta(a)$ значимо ниже $P(a)$, то алгоритм a при условии β_{ij} имеет тенденцию ошибочно объединять новости в один кластер. Если же $R_\beta(a)$ значимо ниже $R(a)$, то алгоритм a при условии β_{ij} имеет тенденцию ошибочно разделять новости, которые должны были бы оказаться в одном кластере.

Условный анализ позволяет автоматизировать поиск утечек качества кластеризации, лучше понимать их причины и эффективнее их устранять.

Смещённость из-за переобучения. Пусть имеется T моделей кластеризации $a^1, \ldots, a^T$, и из них необходимо выбрать лучшую. Для этого будем использовать скалярный критерий F_1-меры $F(a) = \frac{2P(a)R(a)}{P(a)+R(a)}$. Поскольку выбор даже из небольшого числа моделей по конечной выборке может приводить к переобучению, разобьём размеченную выборку пар новостей Z случайным образом на две части: Z_1 будем использовать в роли обучающей выборки для выбора модели $a^* = \arg\max_a F(a, Z_1)$, а Z_2 — для несмещённого оценивания качества выбранной модели $F(a^*, Z_2)$.

Таким образом, обычная в машинном обучении методика hold-out, когда выборка разделяется на обучающую и тестовую, является частным случаем условного оценивания при $\beta_{ij} = [(i, j) \in Z_1]$ и $\beta_{ij} = [(i, j) \in Z_2]$ соответственно.

Смещённость по источникам. Новостные агрегаторы сталкиваются с задачей кластеризации новостей из большого числа источников. Возникает опасение, что алгоритм кластеризации может ошибочно относить новости в один кластер не по сходству контента, а по сходству оформления или стилевых особенностей источника. Чтобы проверить гипотезу о наличии такой смещённости, определим индикатор $\beta_{ij} = [s(i) = s(j)]$, равный 1 тогда и только тогда, когда пара новостей (x_i, x_j) имеет общий источник. Для анализа отдельных источников или выявления источников с наибольшим вкладом в утечку качества, индикатор $\beta_{ij}(s) = \big[s = s(i) = s(j)\big]$ вводится как функция от номера источника s. Если $P_\beta(a)$ значимо ниже $P(a)$, то алгоритм a имеет тенденцию ошибочно объединять в один кластер новости из общего источника.

Смещённость по мощности кластера. Ещё одно опасение связано с тем, что некоторые эвристики кластеризации могут хуже работать на мелких кластерах, другие, наоборот, на крупных. Обозначим через m_i мощность кластера, к которому алгоритм a присоединяет новость i, на тот момент, когда происходит присоединение. Определим индикатор $\beta_{ij}(m) = \big[m \in \{m_i, m_j\}\big]$ как функцию от мощности кластера m. Графики точности, полноты или F_1-меры как функций от m позволяют выявлять, при каких m происходит утечка качества кластеризации.

Смещённость по длине текста. Обозначим через ℓ_i длину текста x_i в словах (или в десятках слов для получения более сглаженных графиков). Определим индикатор $\beta_{ij}(\ell) = \big[\ell \in \{\ell_i, \ell_j\}\big]$ как функцию от длины текста ℓ. Графики точности, полноты или F_1-меры как функций от ℓ позволяют проверять, насколько снижается качество кластеризации при сравнении слишком коротких или, наоборот, слишком длинных текстов новостей. Аналогичным образом можно проверять, не ухудшается ли качество кластеризации при сравнении текстов существенно различной длины. Для этого вводится другой индикатор, $\beta_{ij}(\delta) = \big[\delta = |\ell_i - \ell_j|\big]$.

6 Корпус новостей

Сырые данные Для задачи выделения цепочек используется корпус новостей с российских новостных сайтов: *gazeta.ru, interfax.ru, iz.ru, kommersant.ru, meduza.io, russian.rt.com, tvzvezda.ru.* В выборке присутствуют новости всех категорий, т.е. агрегируется весь новостной поток.

Разметка Разметка состоит из пар новостей (x_i, x_j) и их меток класса. Метка класса каждой пары «2», если пара новостей принадлежит одному событию, «1», если одной теме и «0» в ином случае.

$$y_{ij} = \begin{cases} 2, & \text{(i, j) относятся к одному событию} \\ 1, & \text{имеют общую тему, но разные события} \\ 0, & \text{имеют разные темы и события} \end{cases}$$

- ***Событие*** – определенный факт общественно-политической жизни, явление действительности, которое произошло в определенное время в определенном месте.
- ***Тема*** – *объединение разных, но схожих по смыслу событий. Новости, относящиеся к одной теме, как правило, имеют много общей лексики (т.е. общих слов и словосочетаний) или общую читательскую аудиторию.*

Разметка, полученная от одного человека не будет обладать качеством репрезентативности, поэтому необходимо размечать данные большим количеством отличающихся друг от друга людей. Специально для этих целей был создан сервис Яндекс.Толока, который мы и использовали. Естественным образом возникает несколько шагов для получения разметки:

1. **Отбор пар для разметки асессорами.** За две недели было собрано примерно 15 тыс. новостей, что соответствует огромному количеству пар: C^2_{15000}, на их разметку может быть потрачено огромное количество денег. Понятно, что самый вероятный класс случайно взятой пары будет «0», а для экспериментов нас интересуют пары с примерным соотношением классов 1:2:3 для классов «0», «1» и «2» соответственно. Таким образом, встаёт задача обора пар для разметки с примерно таким же предполагаемым соотношением классов.
2. **Составление инструкции для разметчиков.** Очень важно коротко и ясно объяснить асессорам их задачу, чтобы они выполняли свою работу максимально точно.
3. **Контроль качества разметки.** Нельзя просто дать разметчикам сырые данные и через время забрать готовые идеально размеченные пары. На протяжении всего процесса разметки необходим контроль за качеством выполнения, чтобы не давать недобросовестным разметчикам портить данные. Также необходимо установить максимальное количество размеченных пар для одного человека, чтобы сохранить репрезентативность выборки. К счастью, платформа Яндекс.Толока предоставляет необходимый инструментарий для всех этих потребностей.
4. **Оценка результата.** Необходимо убедиться, что мы получили тот результат, которого добивались.

6.1 Отбор пар для разметки асессорами

Необходимо отобрать пары, соотношение классов которых после разметки с высокой вероятностью будет примерно 1:2:3 для классов «0», «1» и «2» соответственно. Для решение этой задачи предлагается использовать классификатор, обученный на небольшом корпусе вручную размеченных пар. Из множества пар-кандидатов в разметку подаются те пары, вероятность оказаться в нужном классе которых достаточно высока.

Первичная (ручная) разметка Если отбирать пары для ручной разметки случайно, мы столкнёмся с той же проблемой – случайно взятая пара с большой долей вероятности будет относиться к классу «0».

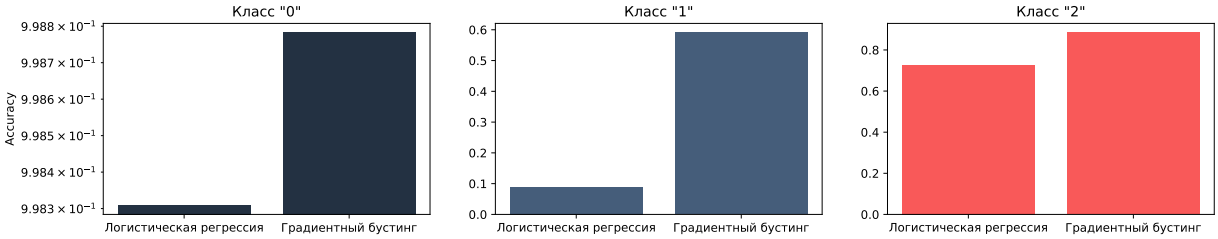


Рис. 3: Качество предсказания классов каждым из классификаторов

Поскольку у нас есть полностью рабочий unsupervised алгоритм построения цепочек новостей по событиям, а собираемая в данный момент разметка нужна только для настройки его гиперпараметров, можно взять ту комбинацию гиперпараметров, которая кажется нам максимально логичной и запустить процесс генерации цепочек.

Конечно, полученные цепочки не будут очень хорошими, но смотреть на них будет гораздо проще, чем на кучу неструктурированных данных. Остаётся лишь пройтись по всем полученным цепочкам, объединяя и дробя их с целью получить идеальное разбиение новостей на цепочки событий. Далее полученные цепочки нужно объединить в группы по темам. Пары новостей, входящие в одну цепочку помечаются классом «2», в одну группу по теме – классом «1», остальные пары относятся к классу «0». Таким образом собирается обучающая выборка для классификатора.

Отметим, что таким же способом можно отбирать пары для асессоров, но тогда придётся потратить больше времени на обработку результатов ненастроенного алгоритма. Преимущество классификатора в том, что в любой момент времени можно практически мгновенно отобрать практически любое количество пар для разметки, не затрачивая на это дополнительных сил.

Выбор классификатора В качестве классификаторов для выделения пар-кандидатов сравниваются два алгоритма: многоклассовая логистическая регрессия и градиентный бустинг. Реализация первого взята из библиотеки sklearn, а второго из библиотеки xgboost. Новости векторизуются при помощи TF-IDF, для нормализации слов используется лемматизация.

Необходимо выбрать тот классификатор, который во-первых правильно предскажет большее количество объектов каждого класса, а во-вторых покажет высокие вероятности истинных классов на отложенной выборке. Пусть в тестовой выборке N пар $a(x_i)$ – предсказание классификатора для i-ой пары, для оценки качества предсказания классов используется метрика *Accuracy*:

$$\text{Accuracy}(y) = \frac{\sum\limits_{i=1}^{N} \big[a(x_i) = y\big]\big[y_i = y\big]}{\sum\limits_{i=1}^{N} \big[y_i = y\big]}, \text{ где } y = \{0, 1, 2\} \text{ (метки классов)}$$

Результаты сравнения классификаторов представлены на Рис. 3. Видно, что xgboost лучше справляется с точностью классификации.

На Рис. 4 можно заметить, что у линейной регрессии, при увеличении вероятности предсказания для классов «1» и «2», количество правильно предсказанных пар не увеличивается. Это значит, что

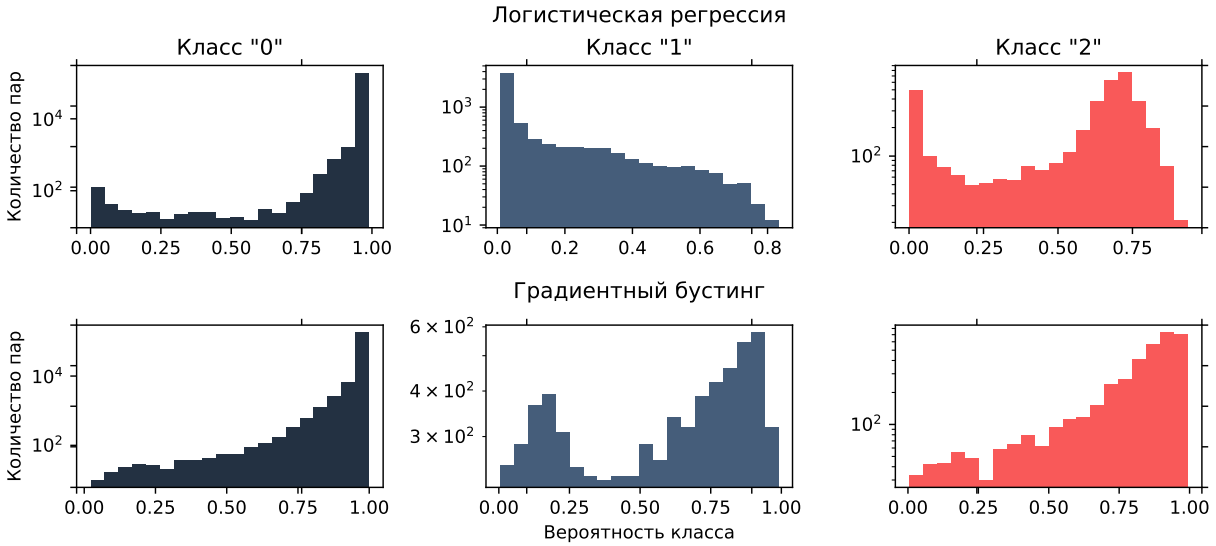


Рис. 4: Распределения вероятностей предсказания истинных классов

отобрать качественные пары-кандидаты не выйдет. В градиентном бустинге картина другая – чем выше вероятность класса, тем больше объектов он предсказывает верно, что позволяет качественно отобрать пары-кандидаты, взяв те, в которых алгоритм достаточно сильно уверен.

Таким образом, в качестве классификатора для отбора пар-кандидатов выбирается **xgboost**.

Отбор пар После обучения классификатора мы можем наконец-то отобрать пары-кандидаты для разметки. Более формально, нам нужно выбрать такое множество пар, среди которых соотношение истинных классов с высокой долей вероятности будет 1:2:3 для классов «0», «1» и «2» соответственно.

Данное множество пар ищется среди тех пар, вероятность отнести которые к классу «2» меняется в диапазоне $[x, 1]$, где x – какое-то значение порога вероятности отнести пару к классу «2». На Рис. 5 изображено соотношение истинных классов при различных порогах x, среди тех что классификатр отнёс к классу «2».

Оптимальное значение порога x должно быть выбрано таким образом, чтобы количество пар класса «1» лежало примерно посередине между количеством пар класса «2» и класса «0», при этом пар класса «0» было как можно меньше, а класса «2» – как можно больше. Необходимые соотношения сохраняются при значении $x = 0.1$ и $x = 0.8$. Отложенная выборка, для которой рисовался график, гораздо меньше той, что предстоит размечать, поэтому для надёжности выбирается значение $x = 0.8$.

Таким образом, на разметку асессорам подаются только те пары, вероятность отнести которые к классу «2» лежит в отрезке $[0.8, 1]$.

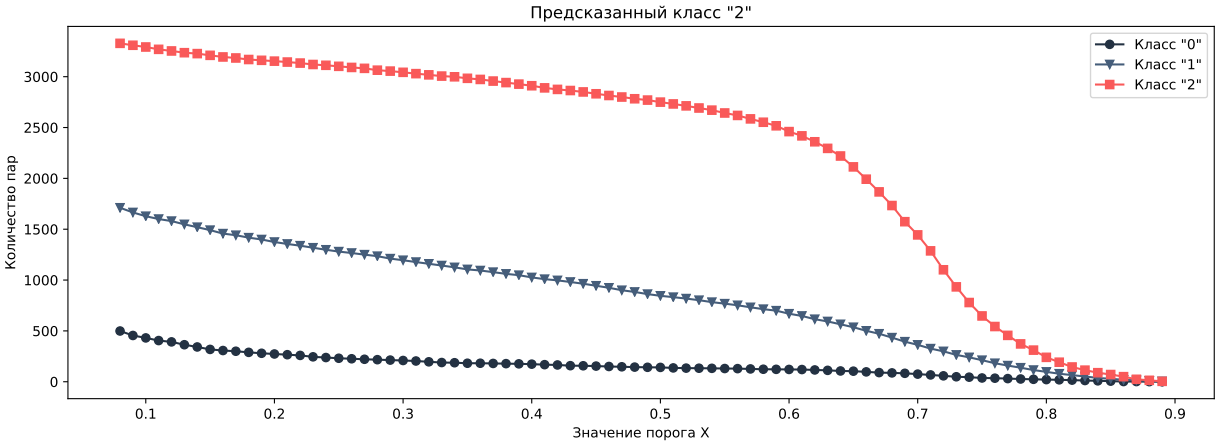


Рис. 5: Истинное соотношение классов пар, вероятность отнести которые к классу «2» лежит в диапазоне $[x, 1]$

6.2 Составление инструкции для разметчиков

Важным этапом получения разметки является процесс формулировки задания. В задании должны содержаться понятные определения того, что такое «событие» и «тема», а также подробные инструкции к задания с примерами, которые иллюстрируют трудности, которые могут возникнуть при их выполнении. Один из примеров, предложенных асессорам для обучения можно видеть на(Рис. 6.

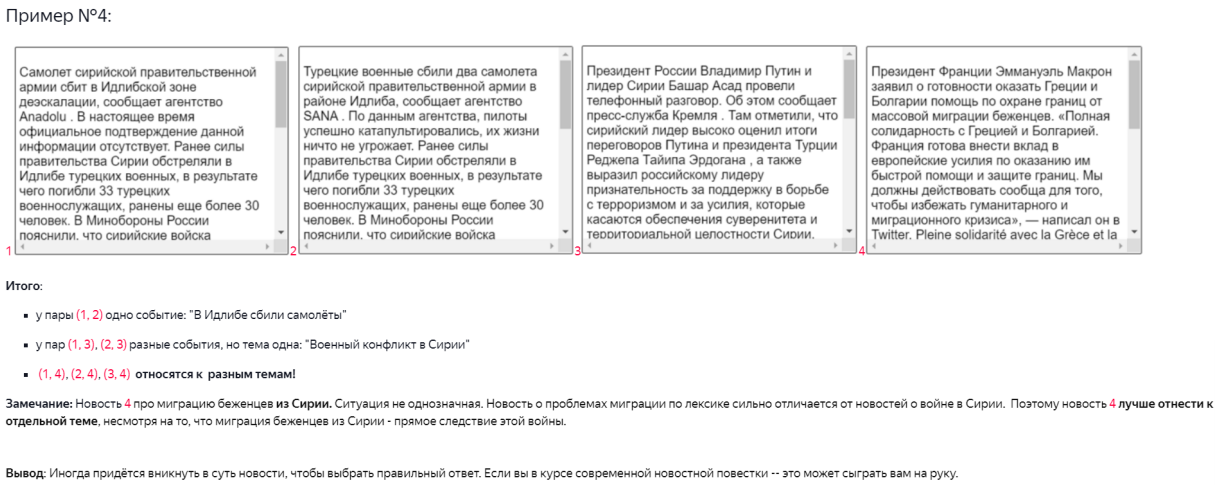


Рис. 6: Пример обучающего задания для асессоров

6.3 Контроль качества разметки

До того как разметчики приступили к выполнению задания, важно определить такой набор правил, который, с одной стороны, позволит максимально точно выявить недобросовестных разметчиков,

а с другой стороны не будет слишком строгим по отношению добросовестных асессорам, случайно допустившим ошибку. Предлагается следующий набор правил:

1. Если пропущенных подряд заданий ≥ 3, то заблокировать асессора на 10 минут.
2. Если процент неправильных контрольных заданий ≥ 34 и количество выполненных контрольных заданий, то заблокировать асессора на 5 дней.
3. Если выполненных заданий ≥ 30, то заблокировать асессора на 1 час. Такое правило необходимо для того, чтобы один человек не размечал слишком много пар, т.к. требуется получить как можно более репрезентативную разметку.
4. Если количество быстрых ответов ≥ 5, то заблокировать асессора на 3 часа.

Особое внимание стоит уделить пункту 2: из уже размеченных пар выбираются контрольные пары, которые нужны для контроля качества разметки. В контрольные задания нужно включать примерно в 5 раз меньше пар класса «1», чем остальных классов. Дело в том, что общность темы пары новостей интуитивно воспринимается каждым человеком по-разному, поэтому существует вероятность заблокировать добросовестного асессора просто так. Задача разметки класса «1» заключается в том, чтобы получить уровень интуитивного восприятия каждым человеком общность темы пары новостей.

Следующим важным этапом является установка уровня перекрытия и процента аудитории. Уровень перекрытия отвечает за то, сколько человек будут размечать одну и ту же новость. Перекрытие необходимо для повышения качества размеченных пар. Нам необходимо, чтобы каждую пару разметили 3 человека, поэтому уровень перекрытия можно оставить равным трём. Процент аудитории отвечает за процент людей с наибольшим рейтингом, которым будет доступно данное задание. Чем больше процент аудитории, тем выше скорость разметки, но ниже качество, т.к. задание становится доступно большему число менее опытных асессоров. Рекомендуется выставлять это значение в пределах от 40% до 60%.

Наконец, необходимо чтобы асессор прошёл тест из какого-то количества пробных заданий. Количество заданий в тесте и максимально допустимое количество ошибок подбирается в зависимости от качества или скорости желаемой разметки. Для разметки пар новостей тест состоит из 5-и пробных заданий, в которых содержится по 2 пары класса «0» и «2> и одна пара класса «1», асессор допускается к выполнению основного пула заданий только в том случае, если он не допустил ни одной ошибки.

6.4 Оценка результата

Всего асессоры разметили 20649 пар. Из них 4514 относятся к классу «0», 7460 к классу «1», 8675 к классу «2». Хоть классы не соотносятся в точности 1:2:3, основная цель была достигнута – большинство пар относятся к классам «2» и «1». Так как каждую пару размечало 3 человека, всего

имеется 6883 уникальные пары. В этих парах участвует 2978 уникальные новости. Такие новости далее будут называться «размеченные».

7 Эксперимент

7.1 Построение цепочек

При построении цепочек естественным образом возникает три гиперпараметра модели

- Способ векторизации новостей
- Способ оценки расстояния между двумя векторами новостей
- Способ агрегации расстояний между вектором новости и всеми векторами внутри цепочки, расстояние до которой мы измеряем
- Порог расстояния от новости до цепочек, при превышении которого новость образует новую цепочку.

Все параметры оптимизируются независимо. Когда первые три параметра фиксированы, четвёртый выбирается перебором по сетке. Таким образом, подбирается лучшая комбинация всех четырёх гиперпараметров.

Способ векторизации новостей. Между собой сравнивались такие способы векторизации как: BoW, TFIDF, Word2Vec, предобученные FastText и BERT. Все вектора сравнивались на косинусном расстоянии, в качестве агрегации расстояний бралось среднее по всем векторам цепочки, все значения сравнивались при наилучшем выборе порога. Лучшие результаты показал TFIDF, обученный на стороннем корпусе новостей. Для нормализации слов использовалась лемматизация.

Расстояния между векторами. Сравнивалось несколько способов измерения расстояния между векторами: косинусное, Евклидово и Манхеттеново расстояния. Два последних дополнительно нормировались в промежуток $[0, 1]$, чтобы иметь возможность подбирать порог.

1. Нормировка для расстояния Манхеттена: $\hat{\rho}(x, C) = \frac{\sqrt{\rho(x,C)}}{\sqrt{\rho(x,C)}+1}$
2. Нормировка для расстояния Евклида: $\hat{\rho}(x, C) = \frac{\rho^2(x,C)}{\rho^2(x,C)+1}$

Такие способы нормировки выбраны для удобства подбора порога: среднее расстояние между документами должно переводиться примерно в $\frac{1}{2}$.

Агрегация расстояний до элементов цепочки. Рассматривались следующие способы агрегации расстояний до новостей в цепочке: усреднение по всем новостям, минимальное расстояние в цепочке, расстояние до последней новости в цепочке и скользящее среднее с $\alpha = \frac{2}{n+1}$, где ширина окна $n = 4$ (оптимизации по n не проводилось). Результаты сравнения в Таблице 1.

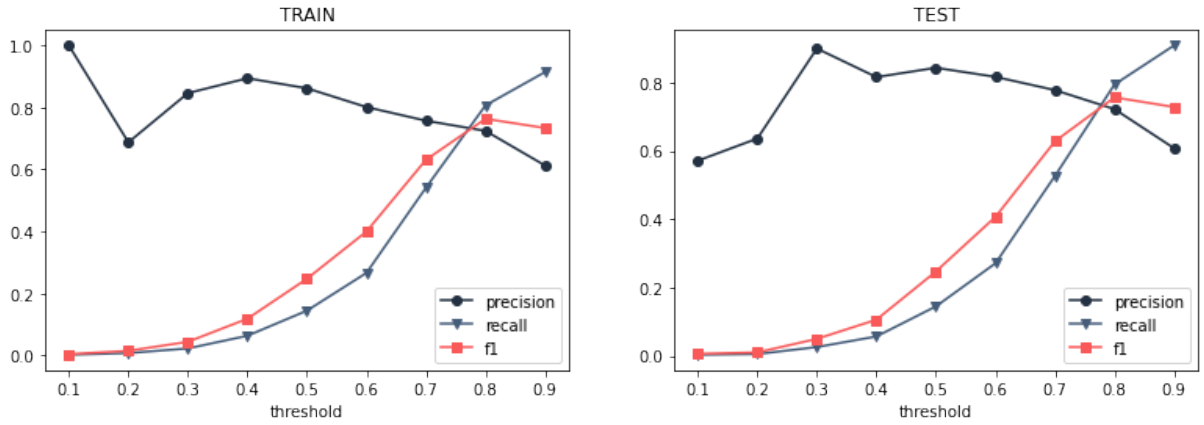


Рис. 7: Процедура выбора оптимального порога образования новой цепочки (threshold). Порог подбирается по обучающей (TRAIN) выборке. Итоговое качество оценивается по тестовой (TEST) выборке.

	MEAN	WMEAN	MIN	LAST
COSINE	**0.76**	**0.61**	**0.74**	**0.67**
EUCLIDEAN	0.73	0.58	0.71	0.58
MANHATTAN	0.59	0.58	0.59	0.58

Таблица 1: Сравнение качества (F1 мера) цепочек для различных функций расстояния от новости до новостей в цепочке и способов их агрегации в расстояния от новости до цепочки. Значения взяты при наилучших значений порога.

Видно, что из-за нормализации Евклидово и Манхеттеново расстояния в некоторых случаях практически идентичны (округлено до сотых). Возможно, если более точно подобрать методы нормализации, значения получатся более различными, но в этом нет особого смысла, так как со временем статистики расстояний всё-равно изменятся. Таким образом, самая лучшая метрика – косинусное расстояние, а лучшая функция агрегации расстояний до новостей в цепочке – их среднее.

Оценка влияния порога на результат. Выяснилось, что при слишком низком пороге возникает тенденция образования большого числа цепочек из малого числа элементов, модель становится слишком чувствительной к мельчайшим отличиям в контенте новостей. При слишком высоком пороге возникает обратная проблема: многие цепочки содержат новости, не принадлежащие одному кластеру, цепочки становятся слишком длинными и могут выродится в единственную цепочку с длиной, равной мощности корпуса. Это можно понять из значений точности и полноты (Рис. 7).

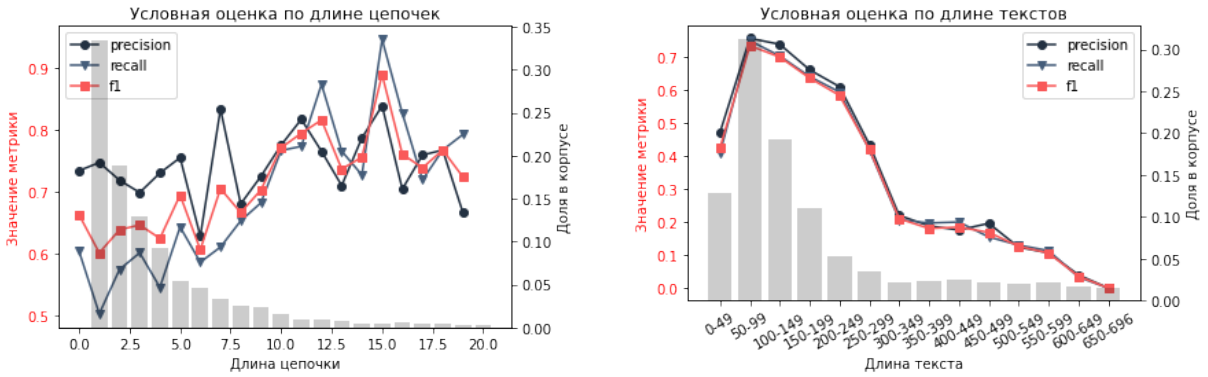


Рис. 8: Условные оценки качества по длине цепочек и текста.

После выбора порога, максимизирующего f1 меру, качество итоговой модели можно более детально оценить при помощи условных оценок. Например, если применить условную оценку смещённости по источникам, то наилучший алгоритм a_{best} покажет $P_\beta(a_{best}) = 0.68$, $R_\beta(a_{best}) = 0.67$. Это меньше чем $P(a_{best}) = 0.75$ и $R(a_{best}) = 0.73$, что означает, что алгоритм немного переобучается на специфичной для источников лексике.

Рассмотрим условные оценки по длине цепочек и текстов. На Рис. 8 показано, как меняется качество модели в зависимости от роста длины цепочек и текстов. Видно, что наибольшая потеря качества происходит на цепочках длины 1, которых модель генерирует больше всего. Эту проблему можно исправить, повышая порог, но жертвуя качеством.

Наилучший алгоритм a_{best} построения цепочек даёт относительное качество (relative accuracy) $A(a_{best}) = 1.01$. Это означает, что с точки зрения качества классификации пар наш алгоритм сопоставим в человеком.

7.2 Построение тем

На данном этапе цепочки новостей уже построены. Необходимо провести кластеризацию второго уровня, сравнить качество мягкой (тематической) и жёсткой кластеризации. Каждая цепочка рассматривается как отдельный документ. Считается, что все новости в цепочке попадают в тот кластер, что и цепочка, которая их содержит. В случае мягкой кластеризации, новостям присваивается распределение тем своей цепочки.

Жёсткая кластеризация. Первым делом исследуется качество алгоритмов жёсткой кластеризации. В качестве таких алгоритмов рассматриваются KMeans и DBSCAN (Рис. 11). По f1 мере оба алгоритма достигают примерно одинакового качества.

Относительное качество при наилучшем выборе параметров для обоих алгоритмов достигает значения $A(a) = 0.98$. Это означает, что качество выделения тем обоих алгоритмов лишь немного ниже человеческого.

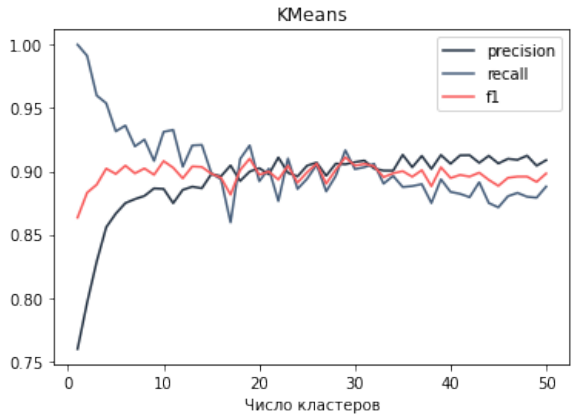


Рис. 9: KMeans при различном числе кластеров

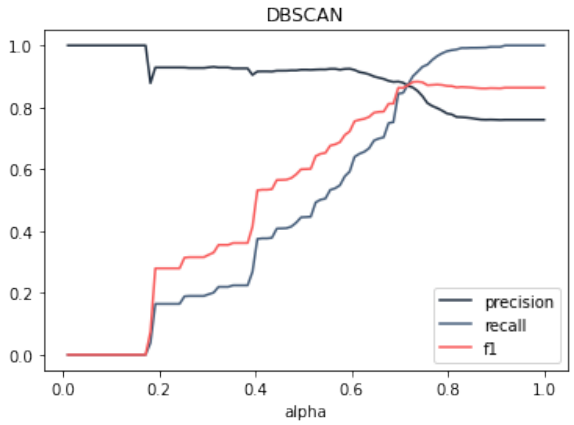


Рис. 10: DBSCAN при различных значениях α

Рис. 11: Сравнение качества жёсткой кластеризации на цепочках, построенных за весь промежуток времени

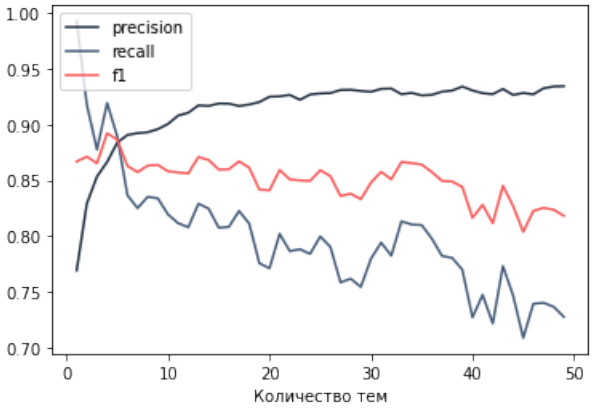


Рис. 12: Зависимость качества мягкой кластеризации от числа тем на цепочках за весь промежуток времени

Мягкая кластеризация. Мягкая кластеризация реализуется при помощи тематического моделирование, с помощью нахождения распределения вероятностей тем над цепочками. В качестве инструмента построения тематической модели используется open-source библиотека BigARTM.

Из всех гиперпараметров модели на валидации подбирается только число тем, остальные параметры выбираются из соображений здравого смысла и общих наблюдений за поведением алгоритма. Дополнительно используются модальности: заголовков, текстов и тэгов новостей, входящих в цепочку. На Рис. 12 видно, что тематическое моделирование даёт сопоставимое с жёсткой кластеризацией качество, при этом оно имеет преимущество в виде построения распределения тем над цепочками новостей. Такое распределение даёт нам более точное понимание связей между цепочками.

Автоматический выбор числа тем. На первом уровне кластеризации число цепочек можно регулировать за счёт изменения порога *threshold*.

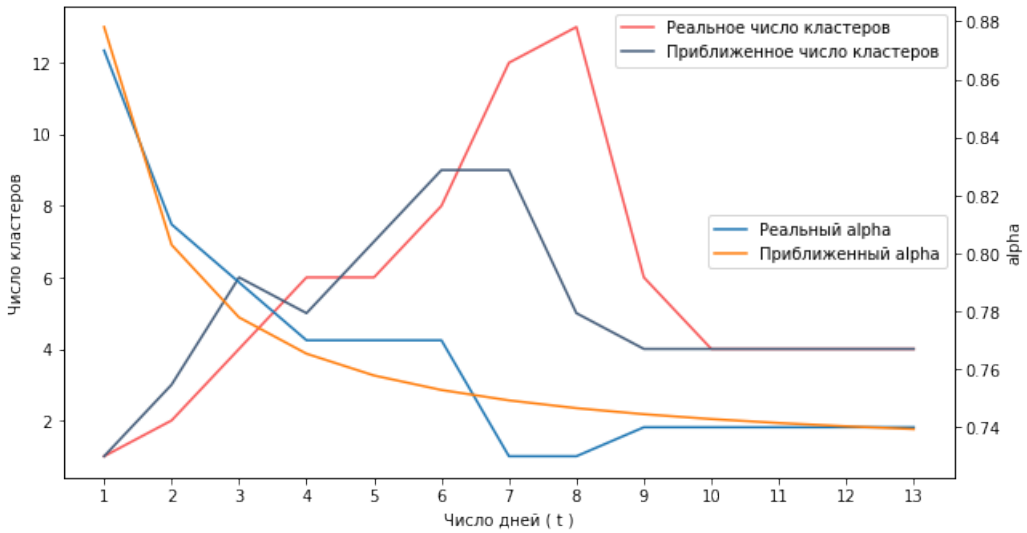


Рис. 13: Алгоритм DBSCAN: сравнение реальных и приближенных оптимальных значений параметра α и соответствующие им количества кластеров.

На втором уровне кластеризации немного сложнее. Классическое тематическое моделирование, на основе которого строится верхнеуровневая кластеризация, не даёт возможность автоматически выбирать число тем. Для решения этой проблемы предлагается использовать алгоритм жёсткой кластеризации DBSCAN. Он имеет лишь один гиперпараметр α, который отвечает за «близость» объектов в каждом кластере, а также неявно задаёт это самое число кластеров.

Чтобы определить сколько тем присутствует в корпусе в определённый момент времени, мы подбираем параметр α на отложенной выборке, затем приближаем его зависимость от числа дней монотонной функцией $\hat{\alpha}(t)$. После чего остаётся взять значение этой функции в нужный момент времени и запустить DBSCAN с параметром $\alpha = \hat{\alpha}(t)$ на цепочках. В качестве количества тем берётся число получившихся кластеров. Для приближения мы использовали гиперболическую регрессию, так как гипербола монотонно убывает и даёт хорошую точность аппроксимации. Результаты приближения показаны на Рис. 13.

$$\hat{\alpha}(t) = 0.7278 + \frac{0.1503}{t}$$

7.3 Онлайновый алгоритм двухуровневой кластеризации

Целью данного раздела является исследовать то, как поведёт себя наш алгоритм двухуровневой кластеризации при инкрементном пополнении корпуса новостей. Поток новостей моделируется при помощи разделения всего датасета по дням (всего 13 дней). Каждую итерацию цепочки пополняются новостями за новый день, далее обновляется кластеризация второго уровня, после чего измеряется качество (Рис. 14).

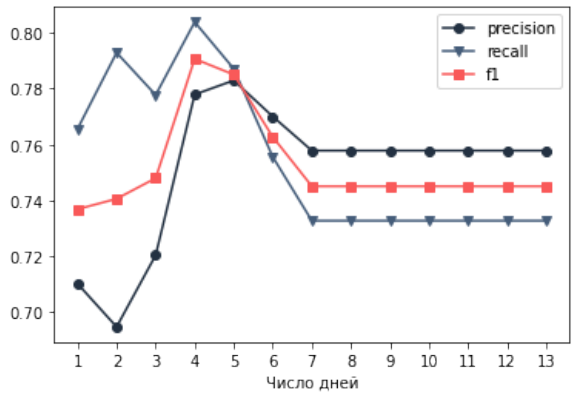


Рис. 14: Изменение качества кластеризации первого уровня (цепочек новостей) со временем

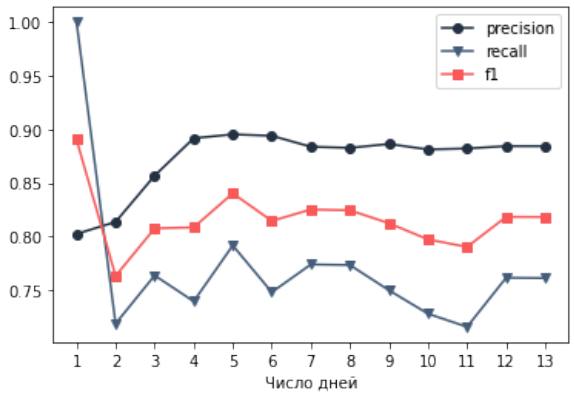


Рис. 15: Изменение качества кластеризации второго уровня (тематической модели) со временем

В качестве алгоритма для второго уровня кластеризации используется тематическое моделирование, так как конечной целью является получить невырожденное распределение тем над цепочками.

При очередной итерации алгоритма тематическую модель необходимо дообучить, добавив новые слова и изменив число тем. Если при поступлении новой порции данных обучать тематическую модель с нуля, но с обновленными параметрами, то восстановить связь новых тем с темами за предыдущий период будет очень трудно, для этого алгоритм обучения должен быть онлайновым:

1. При поступлении новой порции данных добавить в модель новые слова, документы (в нашем случае цепочки), изменить число тем. Размеры матриц Φ и Θ поменяются.

2. В матрице Φ каждый новый столбец инициализировать равномерным распределением по старым словам. В матрице Θ каждый новый столбец инициализировать вырожденным распределением в той теме, к которой DBSCAN отнёс соответствующий документ.

3. Обучить новую модель на обновлённом корпусе цепочек.

Важно отметить, что в нашей реализации число тем никогда не уменьшалось, поэтому оно всегда было $\geq$ того, что показывает DBSCAN. Вместо этого используется регуляризатор отбора тем [45], который зануляет нерелевантные темы во всех документах, отчего некоторые строки матрицы Θ становятся полностью нулевыми. Результаты работы модели представлены на Рис. 15

В разметке отсутствуют новости после седьмого дня, поэтому, в силу своей инкрементности, кластеризация первого уровня практически не меняет качество с того момента. На изменение качества кластеризации второго уровня отсутствие разметки сильного влияния не оказывает, так как эта кластеризация не жёсткая, а распределения тем со временем меняется.

Качество кластеризации первого уровня на последнем дне не отличается от того, что было ранее (Табл. 1), так как никаких изменений в дизайне эксперимента на этом уровне не было. Качество кластеризации второго уровня оказалось ниже чем то, что было ранее (Рис. 12) так как теперь

тематическая модель обучена в онлайн режиме, а число тем подбирается автоматически. Кроме того, f1 мера на втором уровне почти не меняется со временем, это значит, что наш алгоритм стабильно неплохо угадывает оптимальное число тем.

8 Результаты

Собран датасет новостей из различных издательств. Датасет был собран с нуля, все парсеры новостных сайтов писались вручную.

Разработана методика крадсорсинга для построения размеченного датасета. Пары новостей для разметки отбирались при помощи классификатора. Разметка проводилась на платформе Яндекс.Толока, для асессоров были составлены поясняющие задания с примерами, проведён контроль качества разметки. Каждую пару новостей размечали три человека.

Предложены оценки качества кластеризации на основе размеченных пар новостей. Введены условные оценки, которые позволяют более детально оценить поведение алгоритма. Предложена метрика относительного качества, позволяющая оценить насколько качество алгоритма отличается от качества человека.

Разработан алгоритм обучения без учителя для формирования цепочек новостей, относящихся к одному событию. Проведен подбор гиперпараметров и выбрана наилучшая комбинация, дающая качество кластеризации, сопоставимое с человеческим. Проведён дополнительный анализ наилучшего алгоритма на основе условных оценок качества. Выполнено требование инкрементности алгоритма.

Разработан алгоритм объединения цепочек новостей в темы. Проведено сравнение алгоритмов жёсткой и мягкой кластеризации, доказано отсутствие между ними разницы с точки зрения качества кластеризации, получено качество сопоставимое с человеческим.

Предложен способ определения оптимального числа тем с использованием алгоритмов жёсткой кластеризации. Приведён пример его использования для построения онлайновых тематических моделей.

Предложен метод темпорального тематического моделирования новостных потоков. Проведена оценка качества его работы.

Список литературы

[1] J. Allan, J. G. Carbonell, G. Doddington, J. Yamron, and Y. Yang. Topic detection and tracking pilot study final report. 1998.

[2] M. T. Altuncu, S. N. Yaliraki, and M. Barahona. Content-driven, unsupervised clustering of news articles through multiscale graph partitioning. *arXiv preprint arXiv:1808.01175*, 2018.

[3] J. Ansah, L. Liu, W. Kang, S. Kwashie, J. Li, and J. Li. A graph is worth a thousand words: Telling event stories using timeline summarization graphs. In *The World Wide Web Conference*, pages 2565–2571, 2019.

[4] T. R. Bandaragoda, D. De Silva, and D. Alahakoon. Automatic event detection in microblogs using incremental machine learning. *Journal of the Association for Information Science and Technology*, 68(10):2394–2411, 2017.

[5] R. C. Barranco, A. P. Boedihardjo, and M. S. Hossain. Analyzing evolving stories in news articles. *International Journal of Data Science and Analytics*, 8(3):241–256, 2019.

[6] Ö. Can and S. Tekir. Automatic story construction from news articles in an online fashion. *arXiv preprint arXiv:2007.10399*, 2020.

[7] Y. Chen, Z. Ding, Q. Zheng, Y. Qin, R. Huang, and N. Shah. A history and theory of textual event detection and recognition. *IEEE Access*, 8:201371–201392, 2020.

[8] C. Cieri, S. Strassel, D. Graff, N. Martey, K. Rennert, and M. Liberman. Corpora for topic detection and tracking. In *Topic detection and tracking*, pages 33–66. Springer, 2002.

[9] M. Cordeiro and J. Gama. Online social networks event detection: a survey. In *Solving Large Scale Learning Tasks. Challenges and Algorithms*, pages 1–41. Springer, 2016.

[10] M. Fedoryszak, B. Frederick, V. Rajaram, and C. Zhong. Real-time event detection on social data streams. In *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2774–2782, 2019.

[11] X. Fu, E. Ch'ng, U. Aickelin, and L. Zhang. An improved system for sentence-level novelty detection in textual streams. 2015.

[12] D. G. Ghalandari and G. Ifrim. Examining the state-of-the-art in news timeline summarization. *arXiv preprint arXiv:2005.10107*, 2020.

[13] G. Glavaš and J. Šnajder. Recognizing identical events with graph kernels. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 797–803, 2013.

[14] M. S. Hossain, C. Andrews, N. Ramakrishnan, and C. North. Helping intelligence analysts make connections. In *Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[15] M. S. Hossain, P. Butler, A. P. Boedihardjo, and N. Ramakrishnan. Storytelling in entity networks to support intelligence analysts. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1375–1383, 2012.

[16] M. S. Hossain, J. Gresock, Y. Edmonds, R. Helm, M. Potts, and N. Ramakrishnan. Connecting the dots between pubmed abstracts. *PloS one*, 7(1):e29509, 2012.

[17] B. F. Keith Norambuena and T. Mitra. Narrative maps: An algorithmic approach to represent and extract information narratives. *Proceedings of the ACM on Human-Computer Interaction*, 4(CSCW3):1–33, 2021.

[18] V. Krishnan and J. Eisenstein. Nonparametric bayesian storyline detection from microtexts. *arXiv preprint arXiv:1601.04580*, 2016.

[19] V. Lavrenko, J. Allan, E. DeGuzman, D. LaFlamme, V. Pollard, and S. Thomas. Relevance models for topic detection and tracking. In *Proceedings of the second international conference on Human Language Technology Research*, pages 115–121. Citeseer, 2002.

[20] D. Liang, G. Wang, and J. Nie. A dynamic evolutionary framework for timeline generation based on distributed representations. *arXiv preprint arXiv:1905.05550*, 2019.

[21] M. Linger and M. Hajaiej. Batch clustering for multilingual news streaming. *arXiv preprint arXiv:2004.08123*, 2020.

[22] B. Liu, W. Guo, D. Niu, J. Luo, C. Wang, Z. Wen, and Y. Xu. Giant: Scalable creation of a web-scale ontology. In *Proceedings of the 2020 ACM SIGMOD International Conference on Management of Data*, pages 393–409, 2020.

[23] B. Liu, D. Niu, K. Lai, L. Kong, and Y. Xu. Growing story forest online from massive breaking news. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 777–785, 2017.

[24] B. Liu, D. Niu, H. Wei, J. Lin, Y. He, K. Lai, and Y. Xu. Matching article pairs with graphical decomposition and convolutions. *arXiv preprint arXiv:1802.07459*, 2018.

[25] X. Liu, M. Wang, and B. Huet. Event analysis in social multimedia: a survey. *Frontiers of Computer Science*, 10(3):433–446, 2016.

[26] I. Mele, S. A. Bahrainian, and F. Crestani. Event mining and timeliness analysis from heterogeneous news streams. *Information Processing & Management*, 56(3):969–993, 2019.

[27] I. Mele and F. Crestani. Event detection for heterogeneous news streams. In *International Conference on Applications of Natural Language to Information Systems*, pages 110–123. Springer, 2017.

[28] S. Miranda, A. Znotiņš, S. B. Cohen, and G. Barzdins. Multilingual clustering of streaming news. *arXiv preprint arXiv:1809.00540*, 2018.

[29] S. Moran, R. McCreadie, C. Macdonald, and I. Ounis. Enhancing first story detection using word embeddings. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, pages 821–824, 2016.

[30] T. A. Oghaz, E. Ç. Mutlu, J. Jasser, N. Yousefi, and I. Garibay. Probabilistic model of narratives over topical trends in social media: A discrete time model. In *Proceedings of the 31st ACM Conference on Hypertext and Social Media*, pages 281–290, 2020.

[31] N. Panagiotou, C. Akkaya, K. Tsioutsiouliklis, V. Kalogeraki, and D. Gunopulos. First story detection using entities and relations. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3237–3244, 2016.

[32] S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to twitter. In *Human language technologies: The 2010 annual conference of the north american chapter of the association for computational linguistics*, pages 181–189, 2010.

[33] S. Petrović, M. Osborne, and V. Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *Proceedings of the 2012 conference of the north american chapter of the association for computational linguistics: Human language technologies*, pages 338–346, 2012.

[34] S. Phuvipadawat and T. Murata. Breaking news detection and tracking in twitter. In *2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology*, volume 3, pages 120–123. IEEE, 2010.

[35] Y. Rao, Q. Li, Q. Wu, H. Xie, F. L. Wang, and T. Wang. A multi-relational term scheme for first story detection. *Neurocomputing*, 254:42–52, 2017.

[36] H. Rashkin, E. Choi, J. Y. Jang, S. Volkova, and Y. Choi. Truth of varying shades: Analyzing language in fake news and political fact-checking. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2931–2937, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics.

[37] S. Ribeiro, O. Ferret, and X. Tannier. Unsupervised event clustering and aggregation from newswire and web articles. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 62–67, 2017.

[38] Z. Saeed, R. A. Abbasi, O. Maqbool, A. Sadaf, I. Razzak, A. Daud, N. R. Aljohani, and G. Xu. What's happening around the world? a survey and framework on event detection techniques on twitter. *Journal of Grid Computing*, 17(2):279–312, 2019.

[39] D. Shahaf and C. Guestrin. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 623–632, 2010.

[40] B. Shi, T.-B. Le, N. Hurley, and G. Ifrim. Story disambiguation: Tracking evolving news stories across news and social streams. *arXiv preprint arXiv:1808.05906*, 2018.

[41] P. Srijith, M. Hepple, K. Bontcheva, and D. Preotiuc-Pietro. Sub-story detection in twitter with hierarchical dirichlet processes. *Information Processing & Management*, 53(4):989–1003, 2017.

[42] M. Toprak, Ö. Özkahraman, and S. Tekir. A news chain evaluation methodology along with a lattice-based approach for news chain construction. In *Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism*, pages 95–99, 2017.

[43] D. Trilling and M. van Hoof. Between article and topic: News events as level of analysis and their computational identification. *Digital Journalism*, 8(10):1317–1337, 2020.

[44] S. Vadrevu, C. H. Teo, S. Rajan, K. Punera, B. Dom, A. J. Smola, Y. Chang, and Z. Zheng. Scalable clustering of news search results. In *Proceedings of the fourth ACM international conference on Web search and data mining*, pages 675–684, 2011.

[45] K. Vorontsov, A. Potapenko, and A. Plavin. Additive regularization of topic models for topic selection and sparse factorization. pages 193–202, 04 2015.

[46] F. Wang, R. J. Ross, and J. Kelleher. Bigger versus similar: selecting a background corpus for first story detection based on distributional similarity. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2019)*, pages 1312–1320, 2019.

[47] W. Xiang and B. Wang. A survey of event extraction from text. *IEEE Access*, 7:173111–173137, 2019.

[48] D. Zhou, H. Xu, X.-Y. Dai, and Y. He. Unsupervised storyline extraction from news articles. In *IJCAI*, pages 3014–3021, 2016.

[49] D. Zhou, H. Xu, and Y. He. An unsupervised bayesian modelling approach for storyline detection on news articles. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1943–1948, 2015.

[50] D. Zhou, X. Zhang, and Y. He. Event extraction from twitter using non-parametric bayesian mixture model with word embeddings. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 808–817, 2017.

[51] Воронцов and Потапенко. Аддитивная регуляризация тематических моделей. In *Доклады Академии наук*, volume 456, pages 268–271, 2014.

[52] Гилязев and Турдаков. Активное обучение и краудсорсинг: обзор методов оптимизации разметки данных. *Труды ИСП РАН*, 30:215–250, 2018.