Активное обучение и краудсорсинг: обзор методов оптимизации разметки данных

[1,2]Гилязев Р.А. <gilyazev@ispras.ru>
[1,3,4]Турдаков Д.Ю. <turdakov@ispras.ru>
[1] Институт системного программирования им. В.П. Иванникова РАН,
109004, Россия, г. Москва, ул. А. Солженицына, д. 25.
[2] Московский физико-технический институт,
141700, Московская область, г. Долгопрудный, Институтский пер., 9
[3]Московский государственный университет имени М.В. Ломоносова,
119991 ГСП-1, Москва, Ленинские горы
[4] Национальный исследовательский университет Высшая школа экономики,
101000, Москва, ул. Мясницкая, д. 20

Аннотация. Качественные аннотированные коллекции являются ключевым элементом при построении систем, использующих машинное обучение. В большинстве случаев создание таких коллекций предполагает привлечение к разметке данных людей, а сам процесс является дорогостоящим и утомительным для аннотаторов. Для оптимизации этого процесса был предложен ряд методов, использующих активное обучение и краудсорсинг. В статье приводится обзор существующих подходов, обсуждается их комбинированное применения, а также описываются существующие программные системы, предназначенные для упрощения процесса разметки данных.

Ключевые слова: активное обучение; краудсорсинг; аннотация корпусов; крауд-вычисления

DOI: 10.15514/ISPRAS-2018-30(2)-11

Для цитирования: Гилязев Р.А., Турдаков Д.Ю. Активное обучение и краудсорсинг: обзор методов оптимизации разметки данных. Труды ИСП РАН, том 30, вып. 2, 2018 г., стр. 215-250. DOI: 10.15514/ISPRAS-2018-30(2)-11

1. Введение

Увеличение размера обучающей выборки позволяет улучшить точность и полноту большинства прикладных решений, основанных на методах машинного обучения с учителем. Однако для получения обучающих выборок в большинстве задач приходится прибегать к ручной разметке данных, что приводит к удорожанию конечных решений, особенно если необходимо привлекать экспертов в предметной области. Для оптимизации процесса

разметки было предложено несколько подходов, которые будут рассмотрены далее.

Одним из направлений работ, позволяющих оптимизировать процесс разметки, является активное обучение. Методы активного обучения направлены на поиск самых информативных примеров для классификатора. На каждой итерации из неразмеченного множества каким-либо алгоритмом выбирается один пример, он предоставляется оракулу (эксперту) на разметку и классификатор заново обучается на обновленном наборе тренировочных примеров. Подробный обзор методов предложен в работе [37]. Активное обучение – важная техника, позволяющая существенного снизить объем размеченных данных, что подкрепляется как экспериментами так и теоретическим обоснованием.

При активном обучении подразумевается, что в каждый момент времени разметку производит только один человек – эксперт в задаче, который и является оракулом. Но часто важно распараллелить нагрузку, что позволяет разметить больше примеров. В этом помогают краудсорсинговые платформы: Amazon Mechanical Turk (MTurk)[1], CrowdFlower[2] , Яндекс.Толока[3] и др. Их цель объединить работников и работодателей. Каждый участник платформы может выложить задание, например, в виде набора из нескольких примеров для разметки, другой участник или исполнитель, увидев интересное ему задание, выполняет его за определенную плату, существенно меньшую стоимости привлечения эксперта.

Основным недостатком такого подхода является низкий уровень качества выполненных заданий. Так Килгаррифф [18] выделил три причины некачественной разметки: неоднозначность данных, плохое руководство для аннотаторов и недостаток мотивации или знаний у аннотатора.

Тем не менее, исследования показывают, что польза от использования краудсорсинга есть, и результат, как правило, достигается при агрегации ответов нескольких аннотаторов на одном и том же примере. Например, С. Новак и др. [31] рассмотрели задачу многоклассовой разметки картинок с привлечением экспертов и обычных исполнителей, используя краудсорсинговую систему MTurk. Авторы, померив несколько статистик согласия, пришли к выводу, что при качественном руководстве для аннотаторов нет смысла в нескольких метках для объекта, если аннотаторами выступают эксперты. В противном случае, несмотря на неплохую точность разметчиков (каппа-статистика была на пороге допустимого), разметка одного задания несколькими разметчиками позволяет получить существенно более качественный датасет.

Район Сноу и др. [40] рассмотрели различные задачи текстовой классификации. Данные были размечены с привлечением экспертов и аннотаторов из MTurk.

[1] https://www.mturk.com/
[2] https://www.crowdflower.com/
[3] https://toloka.yandex.ru/

Измерив корреляцию Пирсона между и теми и другими авторы пришли к выводу, что привлеченные аннотаторы размечают данные заметно хуже экспертов, но их учет позволяет улучшить результаты, в частности, в некоторых задачах обычное усреднение меток не-экспертов сравнимо с результатами одного эксперта уже при 4 аннотаторах.

Похожие результаты получили и другие исследователи, которые дали ответ на вопрос – что важнее при построение тренировочного множества для классификатора: покрытие или качественная разметка [2, 20]. То есть существуют две опции: потратить все ресурсы на то, чтобы разметить как можно больше данных по одному разу, или разметить небольшое количество примеров, но каждый несколькими аннотаторами. Лучшей стратегией является выбор качественной разметки при низком качестве исполнителей, а при высоком – покрытия.

Краудсорсинг широко применяется при решении задач, не поддающихся автоматическим вычислениям и требующих человеческих усилий. На пути получения максимальной пользы при использовании краудплатформ встают три фактора. Первый из них – это качество, то есть нужны алгоритмы, которые наилучшим образом определят настоящие метки из имеющихся. При этом, естественно, необходимо помнить о стоимости разметки – решить задачу увеличением числа аннотаторов для одного примера не всегда разумно – это второй параметр. И, в-третьих, иногда первоочередным фактором является быстрое получение размеченного корпуса, тогда необходимо минимизировать временные задержки при выполнении участниками задания.

Большинство работ направлены на изучение первых двух факторов, и существующие решения обычно учитывают оба. Существующие работы можно сгруппировать несколькими способами. Возможные группировки представлены в табл. 1. Основой алгоритма вывода меток в краудсорсинге является модель аннотатора. Чаще всего это одно или несколько чисел, характеризующих его надежность или компетентность на каждом классе. В дополнение к этому иногда моделируется зависимость аннотаторов, например, марковской сетью. Реже – для каждого аннотатора обучается отдельный классификатор. В некоторых алгоритмах нет явной модели аннотатора, но различие между ними важно. В остальных случаях предполагается, что исполнители равнозначны.

Еще один вид группировки – это способ описания примера. Многие алгоритмы вывода не связаны напрямую с машинным обучением, а направлены лишь на качественную разметку, таким образом признаковое описание объектов встречается не часто. Тем не менее учитывать характеристики примера необходимо. Поэтому моделируются его сложность – чем она выше тем, меньше правильных ответов ожидается получить и распределение тем – вектор характеризующий к каким заранее определенным темам относится задание.

Табл. 1. Классификация алгоритмов вывода меток
Table 1. Classification of ground truth inference algorithms

Моделирование аннотатора	Уровень компетентности	[4][44]
	Матрица ошибок	[3][19][34][53]
	Параметры классификатора	[14] [12]
	Структура зависимости	[19][42]
	Вектор компетентности(в зависимости от темы задания)	[5][51]
	Неявное моделирование	[9][15]
Моделирование примера	Вектор признаков	[14][34][42][45]
	Сложность	[17][44]
	Распределение тем	[5][51]
Вывод меток	Голосование большинством	[4][11][16][40]
	Максимизация правдоподобия EM алгоритмом	[3][34][43][44][53]
	Итерационный процесс	[4][15][51]
	Вывод в графической модели	[19][25][42]
	Решение оптимизационной задачи	[14][41]
Инициализация параметров	На основе результатов разметки тестовых примеров	[4][11][16][17][40][51]
	Голосование большинством	[3][47][34]
	Случайная инициализация	[15][19][34][42]
Распределение заданий	Случайное	[3][14][34][44][47]
	Итеративное	[7][11][38][43][45][54]
	Онлайн	[5][12][20] [51][53]
Тестирование качества алгоритмов	Качество вывода меток	[3][8][9][25][51]
	Качество классификатора, обученного на крауд-данных	[12][14][34][42]

Следующий способ группировки – это алгоритм вывода меток. Самым простым является голосование большинством и его улучшения. В других алгоритмах учитывается модель аннотатора и вывод происходит более сложными путями.

Важным этапом является инициализация параметров модели, например, компетентностей аннотаторов. От этого во многом зависит точность решения. Для максимального использования ресурсов и применения активного обучения важно оптимальное распределение заданий.

Наконец, есть два способа тестирования качества алгоритмов: качество вывода меток и качество классификатора обученного на полученных данных.

Вопрос скорости получения размеченного корпуса не столь популярен у исследователей. В работе [10] выделяют три вида временных задержек. Первая

из них связана с временем, которое проходит с момента отправки задания до начала его выполнения. Сюда входит время на привлечение исполнителя, изучения им руководства для аннотаторов и, возможно, обучение (прохождение тренировочных заданий). Второй вид – это непосредственно выполнение задания. И третий, которого мы коснемся в вопросе об онлайн распределении заданий, – это задержки связанные с исполнением алгоритма генерирующего задания, например, время выполнения итерации активного обучения.

Активное обучение и краудсорсинг (под краудсорсингом здесь подразумевается любое распараллеливание процесса), пожалуй, единственные методы оптимизации процесса разметки. И возникает естественный вопрос – как их объединить, чтобы достичь большего результата? В частности, как в этом случае распределить задания между участниками?

Предметом этой работы является обзор методов разметки данных с использованием краудсорсинга и способов применения активного обучения, когда имеется несколько параллельно работающих исполнителей. Также задачей является исследование фреймворков реализующих эти методы.

Далее статья организована следующим образом. В следующем разделе обсуждаются существующие обзоры по релевантным темам. В разд. 3 освещается вопрос обеспечения качества в краудсорсинге. В четвертом разделе описаны способы распределения заданий между участниками и онлайн взаимодействие пользователей и системы. Разд. 5 посвящен существующим фреймворкам, которые обеспечивают работу с крауд-вычислениями. Разд. 6 содержит заключение.

2. Существующие обзоры

Тема выявления истинных меток возникла относительно недавно, вместе с первыми краудсорсинговыми платформами, и привлекает внимание исследователей до сих пор. Это отмечает А.В. Пономарев в обзоре [55]. В работе рассматриваются методы обеспечения качества в крауд-вычислениях. Под крауд-вычислениями здесь понимается более широкая область, чем разметка примеров для задач классификации, – по сути, это любая обработка информации. Автор провел аналитическое исследование проблемы и выделил широкий спектр существующих методов решения: методы согласования (consensus), методы проектирования потока работ, методы централизованного назначения работ, теоретико-игровые методы, методы, основанные на учете свойств заданий, и методы, основанные на анализе действий пользователя и воздействии на него.

Подробное описание методов краудсорсинга приведено в обзоре [48]. Глобально авторы выделяют два вида работ: те, которые никак не используют признаки объектов, и работы, вывод в которых так или иначе связан с моделью машинного обучения. В последнем случае рассмотрены и методы активного обучения, но не затрагивается проблема параллельной разметки. Также

проведена классификация методов по основным предположениям, использующимся в выводе.

В работе [23] произведена классификация каждого из типов моделей в краудсорсинге. Затронуты такие темы как: моделирование заданий, управление качеством, стоимостью и временными задержками. Особое внимание уделено крауд-операторам, которые охватывают практически все виды крауд-вычислений. Такими операторами являются: операторы выбора, сортировки, агрегации, сопоставления объектов и другие. Для каждого оператора приведены способы управления качеством. Также дан краткий обзор существующих крауд-платформ и фреймворков, упрощающих работу с ними, оперируя информацией как реляционной базой данных.

Смежной (или даже более общей чем выявление истины из крауд-данных) является тема выявления правды из нескольких источников. Последними могут выступать, например, новостные издания, сайты в сети или другие источники информации. Формально задача ставится так: есть множество источников S, множество объектов O и высказывания v_o^s, $o \in O$ – объект, к которому относится высказывание, $s \in S$ – источник. Для каждого объекта o есть истина v_o^*, у каждого источника есть надежность w_s. И, таким образом, имея множество объектов O и высказывания для них нужно найти истину, попутно выводя w_s.

В обзоре [24] методы решения такой задачи делят на три класса:

- итеративные: так как процесс вывода истинных меток и надежностей связаны, то часто сначала оценивают метки, а потом веса источников и так до сходимости;
- основанные на оптимизации, где в общем виде решается задача:
$$\arg\min_{v_o^*, w_s} \sum_{o \in O} \sum_{s \in S} w_s d(v_o^s, v_o^*)$$
где d - некоторая функция расстояния.
- основанные на вероятностных графических моделях.

Все эти методы похожи на методы выявления истинных меток (а некоторые совпадают), но большое внимание здесь уделяется зависимости источников между собой. Например, часто бывает, что одно издание копирует другое и дублироваться могут как ложные, так и истинные высказывания.

Многие исследователи описывают алгоритм вывода как итерационный процесс [23, 24]. Сначала инициализируются параметры модели, затем до сходимости повторяются два шага: вывод меток и переоценка параметров (см. Алгоритм 1).

Имеется несколько открытых реализаций и сравнений методов по данной теме. Одними из первых были системы BATC (2013) [13] и SQUARE (2013) [39]. Авторы обзора [48] реализовали более новые методы в системе CEKA (2015). Среди всех таких работ выделяется статья [52] со сравнением 17 алгоритмов и подробным описанием исследуемых данных. Общей тенденцией являются неплохие результаты алгоритмов, основанных на EM и его модификациях.

К сожалению, во всех обзорах методы, учитывающие признаки примера (связанные с машинным обучением), не сравниваются, в виду дороговизны реалистичного эксперимента и малого количества подходящих датасетов.

3. Основные методы вывода меток в краудсорсинге

В этом разделе рассматриваются методы агрегации меток в предположении, что система никак не влияет на процесс распределения заданий и всегда, за исключением оговоренных случаев, анализ производится после сбора всех ответов исполнителей. Начнем с формальной постановки задачи. Будем следовать классической постановке, встречающейся во многих работах [48, 55]. Пусть имеется N объектов $x_1, ..., x_N$, каждый их которых принадлежит одному из J классов $\{1, ..., J\}$. Также имеется K аннотаторов, каждый из которых некоторым объектам поставил в соответствие класс. Т.е. дана матрица меток $y_i^j \in \{0, ..., J\}$, где $i \in \{1, ..., N\}, j \in \{1, ..., K\}$, класс 0 означает, что соответствующий аннотатор не предоставил метки для примера i. Задача состоит в том, чтобы имея множество меток $\{y_i^j\}_{j=1}^K$ для каждого примера i предсказать правильную метку y_i (здесь предполагается, что она существует и единственна), т. е. нужно минимизировать эмпирический риск:

$$R = \frac{1}{N}\sum_{i=1}^{N} \mathbb{1}(y_i = \hat{y}_i)$$

Здесь $\hat{y}_i$ – предсказание метки, а $\mathbb{1}(x)$ – индикаторная функция, принимающая значение 1 при истинном агрументе x и 0 – в противном случае.

Очевидным решением является голосование большинством: для каждого примера выбирают метку, которая встречается чаще всего для данного примера. Этот способ еще называют мажоритарным голосованием (majority voting (MV)). Предположим, что $2k+1$ аннотаторов предоставили метку для некоторого примера i, причем каждый ставит правильную метку с одинаковой точностью p. Тогда вероятность того, что при мажоритарном голосовании получится правильный ответ:

$$\sum_{j=1}^{k} \binom{2k+1}{j} p^{2k+1-i}(1-p)^i$$

Эта формула часто используется для оценки необходимого числа меток, чтобы получить заданную точность [26].

Интересно вспомнить знаменитую теорему Кондорсье о жюри присяжных, в которой утверждается, что если число присяжных(аннотаторов) стремится к бесконечности, то при $p > 0.5$ вероятность выбрать правильный ответ стремится к 1, а при $p < 0.5$ к 0.

Как уже было сказано, краудсорсингу присущ шум, и метки получаются от людей с разным уровнем компетентности и опыта. Поэтому такой подход не

всегда хорош, поскольку каждый участник вносит одинаковый вклад в итоговый ответ. Логичным кажется модифицировать метод, добавив вес w_j каждому аннотатору как уровень его надежности, который отражает вероятность предоставить правильную метку для произвольного примера, а затем проводить голосование с весами:

$$y_i = \arg\max_{c \in [1,...,J]} \left(\sum_{j=1}^{K} w_j \mathbb{1}(y_i^j = c) \right)$$

В [50] детально рассмотрены алгоритмы голосования, если для каждого аннотатора известны q_j – вероятности предоставления правильного ответа, и доказано, что оптимальным является байесовское голосование, где вероятность класса пропорциональна его правдоподобию:

$$Pr(y_i = c) \propto \prod_{j=1}^{K} q_j^{\mathbb{1}(y_{ij}=c)} \left(\frac{1-q_j}{J-1} \right)^{\mathbb{1}(y_i^j \neq c)}$$

Нормировка позволяет также оценить вероятности классов.

Оценка характеристики аннотатора, например, значений q_j, является основной задачей. Часто для её нахождения используют тестовое множество примеров, с заранее известными ответами [11, 16, 40]. Иногда ненадежных аннотаторов вообще исключают из системы [21]. В системе ZenCrowds [4] привлекали разметчиков для решения задачи связывания именованных сущностей (entity linking). Аннотатор j описывался одним числом q_j – долей правильных ответов. На старте этот параметр получался из результатов разметки тестового множества, если такового не было, то полагалось $q_j = 0.5$. Затем до сходимости проводились итерации из двух шагов. Для каждого примера метка определялась взвешенным голосованием среди надежных аннотаторов, для этого выбирался порог надежности. После веса q_j считались заново в предположении, что полученные до этого метки истинны. В общем виде такая процедура описывается Алгоритмом 1.

---

input: матрица ответов L, L_{ij} - ответ участника с номером j на примере i
output: метки $\hat{\mathbf{y}} = [\hat{y}_1, ..., \hat{y}_N]$ и качество участников $\mathbf{q} = [\hat{q}_1, ..., \hat{q}_K]$
1: Инициализировать $\mathbf{q}$
2: Выполнять до сходимости:
3: for i:=1 to N do:
4: Оценить $\hat{y}_i$ на основе $\mathbf{q}$ и L
5: for j:=1 to K do:
6: Оценить q_j на основе $\hat{\mathbf{y}}$ и L

Алгоритм 1. Итеративный вывод
Algorithm 1. Iterative ground truth inference

Но этот подход не решает проблему того, что надежность может меняться в зависимости от данных или от истинной метки, к тому же привлечение экспертов для разметки теста вызывает дополнительные траты бюджета. Веса надежности являются лишь частью сложной модели, но тем не менее эта идея является ключевой при выводе истинных меток.

Дэвид и Скини [3] рассматривают надежность в более широком смысле. Они предложили для каждого аннотатора вычислять матрицу ошибок: $\pi_{qj}^{(k)}$ – вероятность того, что аннотатор k поставит метку j, если настоящей меткой является q. Теперь аннотатор характеризуется только матрицей ошибок. Обозначим p_{ij} априорное распределение классов для примера i.

Пусть $n_{il}^{(k)}$ – количество раз, которое аннотатор поставил метку l примеру i (предполагалось, что участник мог разметить один и тот же пример несколько раз). Тогда правдоподобие вероятностной модели запишется так:

$$\prod_{i=1}^{N}\left(\sum_{j=1}^{J} p_{ij}\prod_{k=1}^{K}\prod_{l=1}^{J}(\pi_{jl}^{(k)})^{n_{il}^{(k)}}\right)$$

Здесь использовались два важных предположения:

1. метка аннотатора не зависит от примера, а зависит только от истинной метки;
2. аннотаторы предоставляют метки независимо друг от друга.

Для решения задачи используют EM-алгоритм для нахождения параметров, максимизирующих правдоподобие. Итерации происходят следующим образом: при фиксированных $\pi_{qj}^{(k)}$ оцениваются вероятности классов, а затем при фиксированных p_{ij} находятся матрицы ошибок, максимизирующие правдоподобие.

Такую модель называют DS (анаграмма первых букв авторов). Очевидны недостатки модели: EM-алгоритм не гарантирует сходимость к оптимальному решению; необходимо правильно выбрать начальное параметры $\pi_{qj}^{(k)}$; нигде не используется сам пример x_i (в модели авторов не было признаков). И конечно, предположения 1 и 2 не всегда оправданы.

Эта идея в дальнейшем развивалась многими авторами и породила целый класс алгоритмов, которые так или иначе рассматривают модель аннотатора и часто решают задачу путем применения EM-алгоритма.

В работе Жанга и др. [49] для инициализации матрицы ошибок используют оригинальный подход, основанный на спектральном методе. Аннотаторы разбиваются на три группы, и для каждой группы вычисляются усредненные ответы исполнителей. Затем методом моментов оцениваются матрицы ошибок для каждой группы, как будто бы имеется всего 3 участника. На основе полученных оценок находятся начальные приближения для всех аннотаторов.

В алгоритме, предложенном Райкаром и др. [34], рассматривается задача с двумя классами. В таком случае модель аннотатора записывается двумя параметрами:

- чувствительность $\alpha^j = Pr[y^j = 1|y = 1]$ – вероятность того, что аннотатор j верно определит положительный класс и
- специфичность $\beta^j = Pr[y^j = 0|y = 0]$ – вероятность верно определить отрицательный класс.

В оригинальной модели (DS) нигде не использовался сам пример x_i. В работе [34] этот недостаток устраняется введением модели логистической регрессии:

$$Pr[y = 1|\mathbf{x}, \mathbf{w}] = \sigma(\mathbf{w}^T\mathbf{x})$$

где $\sigma(z) = \frac{1}{1+e^{-z}}$. Теперь метка зависит не только от векторов $\boldsymbol{\alpha}$ и $\boldsymbol{\beta}$, но и от $\mathbf{x}$ и вектора весов $\mathbf{w}$, и правдоподобие запишется следующим образом:

$$Pr[D|\theta] = \prod_{i=1}^{N} Pr[y_i^1, ..., y_i^K|\mathbf{x}_i, \boldsymbol{\alpha}, \boldsymbol{\beta}, \mathbf{w}]$$

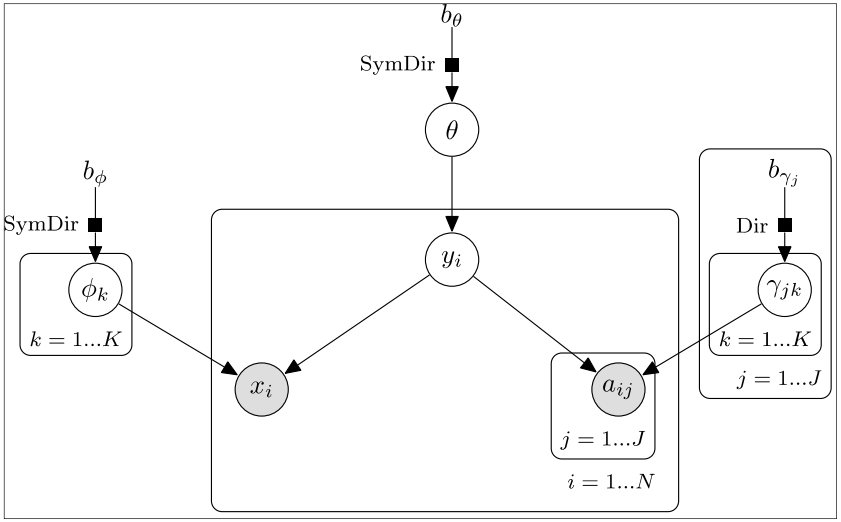


Рис. 1. Графическая модель вывода меток. Закрашенные ячейки соответствуют наблюдаемым переменным
Fig. 1. Graphical model for inferring ground truth labels. Shaded cells correspond to observable variables

$$\forall i : y_i|\boldsymbol{\theta} \sim Categorical(\boldsymbol{\theta})$$

$$\forall i : x_i|y_i, \boldsymbol{\psi} \sim Multinomial(|x_i|_1, \boldsymbol{\psi}_{\boldsymbol{y_i}})$$

$$\forall i : x_i|y_i, \boldsymbol{\psi} \sim Multinomial(|x_i|_1, \boldsymbol{\psi}_{\boldsymbol{y_i}})$$

Дальше выражение преобразуется с учетом предположений (1) и (2), и его логарифм представляется следующим образом:

$$ln(Pr[D|\theta]) = \sum_{i=1}^{n} y_i ln(p_i)a_i + (1-y_i)ln(1-p_i)b_i$$

где

$$p_i = \sigma(\mathbf{w}^T \mathbf{x}_i)$$
$$a_i = \prod_{j=1}^{K} [\alpha_j]^{y_i^j} [1-\alpha_j]^{1-y_i^j}$$
$$b_i = \prod_{j=1}^{K} [\beta_j]^{1-y_i^j} [1-\beta_j]^{y_i^j}$$

Максимум находится EM-алгоритмом: на E шаге оцениваются y_i, на M шаге значения α_i и β_i можно вычислить явно, а $\mathbf{w}$ находится градиентным спуском.

Для инициализации y_i используется мажоритарное голосование. Также отмечается, что алгоритм будет работать, если не все аннотаторы предоставили метки каждому примеру, и метод можно обобщить на любой вероятностный классификатор. Тестирование на наборе из нескольких реальных задач показывает, что метод превосходит мажоритарное голосование.

Однако, с другими методами сравнение не проводилось. Более детально метод и его расширения, включая задачу регрессии, описаны в поздней статье [35].

Важным дополнением является байесовское расширение метода. Если заранее имеются какие-либо предпочтения к аннотаторам, то авторы предполагают, что параметры $\boldsymbol{\alpha}$ и $\boldsymbol{\beta}$ принадлежат Beta распределению. Затем вместо ML ищется MAP оценка. Похожий подход встречается во многих работах. В общем случае вероятностные предположения накладываются на все переменные и система описывается более сложной графической моделью, вывод в которой уже не удается произвести EM-алгоритмом. На помощь приходят методы Монте Карло по схеме марковской цепи (Markov Chain Monte Carlo, MCMC).

Так, например, в работе [19] вводятся дополнительные параметры (предполагается, что матрица ошибок и распределение классов принадлежат распределению Дирихле);

$$\pi_j^{(k)} \sim Dir(\alpha_{j,1}^{(k)}, ..., \alpha_{j,J}^{(k)})$$
$$\mathbf{p} \sim Dir(\nu_1, ..., \nu_J)$$

Сами параметры $\alpha_{j,l}^{(k)}$ порождены экспоненциальным распределением $Exp(\lambda_{j,l}^{(k)})$. Тогда апостериорное распределение при условии независимости аннотаторов будет выглядеть так:

$$P(\boldsymbol{p}, \pi, y, \alpha|c) = \prod_{i=1}^{N} (p_{y_i} \prod_{j=1}^{M} \pi^{(k)}_{y_i, c_i^{(k)}}) p(\boldsymbol{p}|\nu) p(\pi|\alpha) p(\alpha|\lambda)$$

Вывод производится итерационно по схеме Гиббса. Также рассматривается случай зависимости участников, которая моделируется марковской сетью. Графическая модель из работы [42] учитывает признаки объекта. Имеется конечный набор факторов – линейных весов, соответствующих вектору признаков. Аннотатор характеризуется бинарной суммой этих факторов, а вероятность предоставить положительный класс описывается пробит-регрессией. В работе [8] введены параметры для моделирования распределения примеров в случае задачи текстовой классификации. Схема модели представлена на рисунке 1.

Интересный подход предложен Каргером и др. [15]. Рассматривается задача с двумя классами: 1 и -1. Также предполагается истинность допущений 1 и 2. Далее строится случайный (l, r) – регулярный двудольный граф $G(\{t_i\}_{i=1}^m \cup \{w_j\}_{j=1}^n, E)$, l – число вопросов для каждого примера, r – количество примеров которые размечает каждый аннотатор, число аннотаторов n определяется из $lm = rn$. Этот граф определяет распределение заданий для аннотаторов.

Вывод производится итеративно. Используются два типа сообщений $x_{i\to j}$, $y_{j\to i}$, где $(i,j) \in E$. $y_{j\to i}^{(0)}$ инициализируются случайно из нормального распределения $\mathcal{N}(1,1)$. Затем для заданного k_{max} и полученных меток $\{L_{ij}\}_{(i,j)\in E}$ выполняется k_{max} шагов:

- для всех $(i,j) \in E$:
$$x_{i\to j}^k \leftarrow \sum_{j' \in \delta(i)\setminus j} L_{ij'} y_{j'\to i}^{(k-1)}$$
- для всех $(i,j) \in E$:
$$y_{j\to i}^k \leftarrow \sum_{i' \in \delta(j)\setminus i} L_{i'j} x_{i'\to j}^{(k)}$$

$\delta(u)$ – соседи вершины u. Итоговые метки:

$$x_i = \sum_{j\in\delta(i)} L_{ij} y_{j\to i}^{(k_{max}-1)}$$

Значения $x_{i\to j}$, $y_{j\to i}$ легко интерпретируются: $x_{i\to j}$ – метка для примера i, полученная голосованием всех аннотаторов, кроме j-го, а $y_{j\to i}$ – надежность участника j, найденная без учета его предсказания для i-го примера. Авторы приводят теоретические обоснования корректности метода и асимптотики сходимости. Сравнения с мажоритарным голосованием и DS доказывают эффективность метода на синтетических данных. Однако в работе Лиу и др. [25] подчеркивается, что метод сложно обобщить на различные модели аннотатора, и нет уверенности в работе на реальных данных. Был предложен более общий подход, основанный на алгоритме распространения доверия.

Каргер утверждает, что его метод похож на алгоритм поиска собственного вектора матрицы LL^T. Разница только в том, что здесь одно из слагаемых не учитывается.

Гош и др. [9] предложили метод, полностью основанный на вычислении собственного вектора LL^T. Мотивация подхода следующая. Пусть есть два класса 1 и -1. Участник i размечает верно с вероятностью q_j, $\boldsymbol{y}$ – столбец настоящих меток, и A – матрица разметок. Тогда легко посчитать, что $E(L) = \boldsymbol{y}(2\boldsymbol{q} - 1)^T$, а $E(LL^T) = \kappa \boldsymbol{y}\boldsymbol{y}^T + (n - \kappa)I$, где $\kappa = \sum_j (2q_j - 1)$, I – единичная матрица. Максимальное собственное значение матрицы $E(LL^T)$ есть $\kappa||\boldsymbol{y}||^2 + (n - \kappa)$, а $\boldsymbol{y}$ – собственный вектор. Таким образом, в качестве вектора предсказаний алгоритм возвращает собственный вектор LL^T, соответствующий максимальному собственному значению.

Заметим, что не все подходы учитывают вектор признаков примера: сложно объединить модель вывода меток и какой-либо алгоритм машинного обучения. Тем не менее, находятся и другие способы различать объекты. Так, улучшить точность агрегации меток позволяют предположения о сложности примера.

В работе Уайтхилла и др. [44] решалась задача классификации картинок на 2 группы. Здесь параметр $1/\beta_i \in [0, +\infty)$, где $1/\beta_i = +\infty$ говорит о том, что картинка настолько сложная, что даже эксперт разметит ее правильно с вероятностью $1/2$, а $1/\beta_i = 0$ – что любой разметчик поставит этой картинке верный класс. Параметр $\alpha_j \in (-\infty, +\infty)$ отвечает за надежность аннотатора, $\alpha = +\infty$ соответствует идеальному разметчику, а $\alpha = -\infty$ – аннотатору, который всегда дает неправильный класс, 0 соответствует случайному выбору. Вероятность того, что метка l_{ij} для примера i от аннотатора j истинная, определяется как сигмоида $\sigma(\alpha_j\beta_i)$:

$$Pr[l_{ij} = y_i|\alpha_j, \beta_i] = \frac{1}{1 + e^{-\alpha_j\beta_i}}$$

Неизвестные параметры находятся все тем же EM-алгоритмом. Авторы назвали систему GLAD.

Похожим образом параметр сложности примера используется в системе ELICE [17].

Для начальной оценки параметров используются достоверно известные метки для n объектов корпуса. По этим меткам оцениваются надежности аннотаторов как разница числа верно и неверно размеченных примеров

$$\alpha_j = \frac{1}{n}\sum_{i=1}^{n}[\mathbb{1}(y_i = y_i^j) - \mathbb{1}(y_i \neq y_i^j)]$$

и сложность примера:

$$\beta_i = \frac{1}{M}\sum_{j=1}^{M}[\mathbb{1}(y_i = y_i^j)]$$

Для неразмеченных объектов метка сначала оценивается мажоритарным голосованием с весами α_j из первого шага. Затем полученные метки используются для оценки параметра β_i, и, наконец, итоговая метка получается так:

$$F_i = sign[\frac{1}{M}\sum_{j=1}^{M}\sigma(\alpha_j\beta_i)y_i^j]$$

Эксперименты с симуляцией разных типов разметчиков показали, что алгоритм более устойчив к большому числу шумных меток, эффективно работая когда всего 20% аннотаторов работают качественно. При этом экспертам понадобилось разметить всего 20 объектов. Подход применим только для двухклассовой задачи.

Далее была предложена модификация метода ELICE-2 [16], в которой извлекается польза от оппозиционных участников, заведомо неправильно выполняющих аннотации. Алгоритм похож на предыдущую версию, только параметры α_j и β_i домножаются на $(1 - E(p))$, где $E(p) = -plog(p) - (1 - p)log(1 - p)$ – энтропия доли настоящих меток для аннотатора или примера соответственно. Ясно, что случайные разметчики будут иметь высокую энтропию, хорошие и оппозиционные – низкую, но надежность последних так же, как и в ELICE, будет отличаться знаком. Финальная формула теперь учитывает неправильные метки оппозиционных разметчиков:

$$F_i = sign[\frac{1}{M}\sum_{j=1}^{M}\sigma(|c\alpha_j\beta_i|) * l_{ij} * sign(\alpha_j\beta_i)]$$

В системах DOCS [51] и CDAS [26] раскрывается идея того, что компетентность исполнителя связана с темой задания. В DOCS для каждого задания (примера) вводится вектор доменов – распределение по выделенным темам, а каждый исполнитель описывается набором чисел – компетентностью на каждом домене. Вывод происходит по классическому итерационному сценарию, с учетом доменов. В CDAS строится граф похожести заданий: если аннотатор хорошо справился с определенным заданием, то скорее всего он справится и с аналогичным. Идеи авторов систем DOCS и CDAS описываются подробнее в дальнейших разделах.

Имеются несколько подходов, которые строят классификатор явно, не производя вывод меток. Шенг и др. [38] предложили учитывать все собранные метки. Если для примера i есть L_i меток, то из него получается L_i обучающих примеров: каждой со своей меткой и примеру присваивается вес $1/L_i$, который обрабатывается классификатором.

Каджино [14] предложил алгоритм, который обобщает логистическую регрессию на случай нескольких аннотаторов. Для простоты рассматриваются два класса. Общая модель описывается как $\sigma(\boldsymbol{w_o}^T\boldsymbol{x})$. Для каждого аннотатора

j строится своя модель: $\sigma(\boldsymbol{w}_j^T \boldsymbol{x})$, где параметр $\boldsymbol{w}_j$ рассматривается как отклонение от общей модели, что выражается следующим априорным распределением:

$$Pr[\boldsymbol{w}_0|\eta] \sim \mathcal{N}(0, \eta^{-1} I)$$
$$Pr[\boldsymbol{w}_j|\boldsymbol{w}_0, \lambda] \sim \mathcal{N}(\boldsymbol{w}_0, \lambda^{-1} I)$$

Затем, как обычно, максимизируется логарифм апостериорного распределения, что эквивалентно такой функции ошибки:

$$-\sum_{j=1}^{J} \sum_{i \in I_j} l(y_i^j, \sigma(\boldsymbol{w}_j^T \boldsymbol{x}_i)) + \lambda/2 \sum_{j=1}^{J} ||\boldsymbol{w}_j - \boldsymbol{w}_0||^2 + \eta/2 ||\boldsymbol{w}_0||^2$$

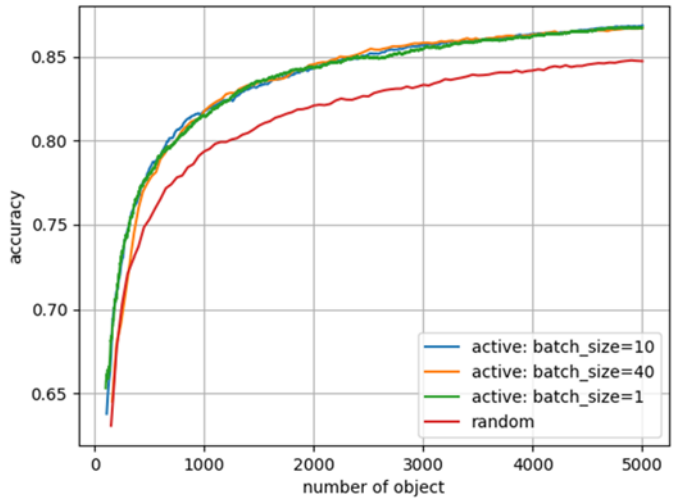
$l(y,p) = -ylog(p) - (1-y)log(1-p)$ – кросс-энтропия. Оптимизация происходит следующим образом. Если даны $\boldsymbol{w}_j$, то $\boldsymbol{w}_0$ выписывается аналитически. Оптимизация по $\boldsymbol{w}_j$ независима и делается по отдельности. То есть для каждого $\boldsymbol{w}_j$ делается шаг оптимизации, затем считается $\boldsymbol{w}_0$.

В конце разлела коснемся практической стороны темы. Сравнения [13, 48, 52] наиболее популярных методов показывают, что нет явного лидера: при двух классах неплохо работают DS [3] и его модификации (RY [34]), а сложные модели с большим количеством параметров не всегда применимы. Иногда совсем простые методы бывают эффективны. Например, метод [47], основанный на кластеризации, оказался успешней остальных в сравнении [48] на многоклассовых задачах. Каждый пример здесь описывается вектором размерности $|J|$ – число классов. Каждая компонента – количество меток соответствующего класса. Ответ получается кластеризацией этих векторов на $|J|$ классов алгоритмом к-средних, центрами кластеров инициализируются те примеры, для которых максимально число голосов за данный класс.

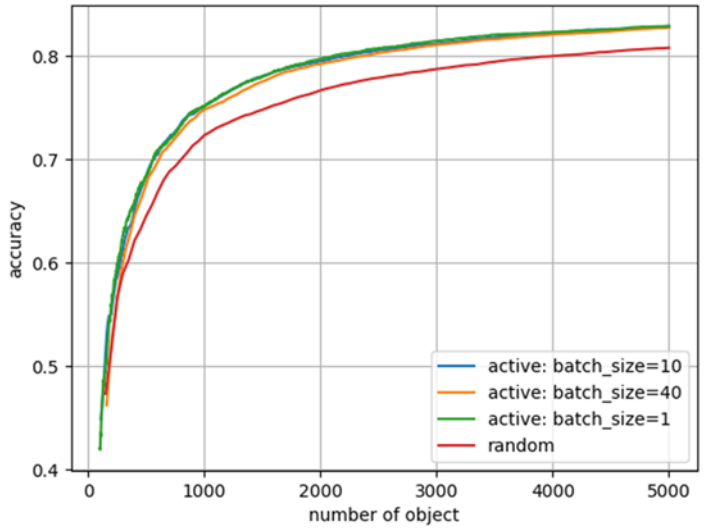
4. Распределение заданий

Как уже было сказано, почти все методы предыдущего раздела предполагали простейший сценарий распределения заданий. Заранее выбиралось множество объектов, разметку которых мы хотим получить. Затем задания, сформированные по несколько примеров в каждом, отправлялись на краудплатформу, каждое по несколько раз. После выполнения всех заданий результаты обрабатывались тем или иным алгоритмом. Другими словами, разметка происходила в оффлайн режиме.

В этом разделе будут рассмотрены различные варианты распределения заданий, в которых разметка происходит в итерационном режиме, т.е. анализ производится не после аннотации всех объектов, а после каждой итерации разметки. Такой способ позволяет эффективно использовать доступные ресурсы, в частности, применить активный выбор примеров и аннотаторов.



(a) IMDB, 2 класса
(a) IMDB, 2 classes



(b) Lenta.ru, 8 классов
(b) Lenta.ru, 8 classes

Рис. 2: Влияние размера пакета на качество: (a) Анализ тональности отзывов о фильмах с сайта IMDB, 2 класса; (b) Классификация новостей с сайта lenta.ru на 8 классов
Fig. 2. Tradeoff between batch size and model quality: (a) Sentiment analysis of movies review from IMDB, 2 classes; (b) News classification from lenta.ru, 8 classes

Классическое активное обучение предполагает, что примеры поставляются на разметку по одному. Эффективность в данном случае подкрепляется также теоретическими обоснованиями.

Но на практике это долго: каждый раз нужно переобучать классификатор и неясно как быть если метки приходят из краудсорсинга (отправлять задания с одним примером было бы затратно). Поэтому нередко генерируют не один пример, а пакет, состоящий из нескольких заданий. Существующие эксперименты подтверждают разумность этого шага. Например, в [17] размеры пакета 10–40 считаются приемлемыми.

То же показывают и наши эксперименты на нескольких задачах текстовой классификации. Вместо привлечения аннотаторов использовались датасеты с заранее известными истинными метками, таким образом происходила симуляция активного обучения. В качестве алгоритма активного обучения был выбран один из наиболее простых и популярных методов – для пополнения обучающего множества выбирались те примеры, на которых вероятностный классификатор был наименее уверен, а именно, с наименьшей разностью вероятностей двух наиболее популярных предсказанных классов. Классификатор – логистическая регрессия, признаки – «мешок» слов. В качестве датасетов были выбраны следующие:

- анализ тональности отзывов с двумя классами, IMDB, датасет из 25 тыс. примеров;
- классификация новостей с сайта Lenta.ru на 8 групп, около 500 тыс. примеров

В обоих случаях классы сбалансированы. На рис. 2 приведены результаты. Показано изменение метрики точности от числа примеров в обучающем множестве в зависимости от режима: активное обучение с различными размерами пакета и случайный выбор. Результаты усреднены по нескольким (трем) запускам итераций. Графики показывают, что использование активного обучения дает прирост в метрике качества по сравнению со случайной разметкой, и при размере пакета 10 получается результат, сравнимый с классическим подходом, где размер пакета 1.

Существующие подходы к пакетному распределению заданий можно разделить на две группы. Первая – итерационное планирование: планировщик после обработки очередной порции примеров, выбирает подходящие объекты или аннотатора и отправляет задание на крауд-платформу. Во второй группе работ, исполнитель сам сообщает о своей готовности выполнить разметку, и система поставляет ему примеры в онлайн режиме. В случае одного аннотатора оба подхода соответствуют традиционному активному обучению. Интерес представляет параллельная работа нескольких исполнителей.

4.1 Итеративная разметка

Наиболее простым подходом является варьирование числа меток для каждого примера в зависимости от его сложности. Так, в [20] число аннотаторов на один пример определяется динамически: после того как пример размечен несколько раз, подсчитывается согласие аннотаторов на нём. Разметка происходит до тех пор, пока не будет достигнут консенсус или не будет превышен допустимый порог числа повторений. В [29] это число для каждого примера считается заранее: все объекты кластеризуются, представители кластеров размечаются в тестовом режиме, на основе согласия на выбранных примерах итоговое число определяется для каждого кластера.

Более результативным является применение активного обучения в том или ином виде. В одной из первых работ [38] на эту тему примеры для разметки выбираются активно, а все собранные метки обрабатываются классификатором. Для итеративного выбора примера предлагается несколько эвристик. Первая считает неопределенность на основе имеющихся меток для примера – в простом случае это может быть энтропия меток. Вторая использует какой-либо алгоритм активного обучения. Эти эвристики объединяются подсчетом среднего геометрического неопределенностей полученных в обоих методах.

Большинство же работ, обсуждаемых в этом разделе, направлены на выбор аннотатора и примера либо одновременно, либо сначала примера, а потом аннотатора. То есть новое задание выдается конкретному человеку, который вероятно справится с ним лучше остальных.

Одной из таких работ является [45]. Метка аннотатора t на примере x_i моделируется нормальным распределением с центром в истинном классе:

$$p(y_i^{(t)}; x_i, z_i) = \mathcal{N}(y_i^{(t)}; y_i, \sigma_t(x_i));$$

y_i – истинная метка.

$$\sigma_t(x_i) = \frac{1}{1 + exp(-\boldsymbol{w}_t^T \boldsymbol{x}_i - \gamma_t)}.$$

Вероятность положительного класса рассматривается как логистическая регрессия:

$$p(z = 1|x_i) = \frac{1}{1 + exp(-\boldsymbol{\alpha}^T \boldsymbol{x}_i - \beta)}$$

Параметры аннотаторов $\boldsymbol{w}_i^T$, γ_t и веса регрессии $\boldsymbol{\alpha}$, β находятся поиском оценки максимального правдоподобия EM-алгоритмом.

Далее выбираются примеры, для которых текущая модель не уверена:

$$\arg\min_x (0.5 - p(y|x))^2$$

Решением является гиперплоскость $\boldsymbol{\alpha}^T \boldsymbol{x}_i + \beta = 0$. Затем ищется аннотатор с наименьшей дисперсией $\sigma_t(x)$. В итоге получается такая задача оптимизации:

$$\min_{x,p}(C(\boldsymbol{\alpha}^T\boldsymbol{x}+\beta)+p^T[\boldsymbol{w}_1,...,\boldsymbol{w}_T]x+p^T\gamma)$$

$p=[p_1,...,p_T]$ – распределение аннотаторов, $\sum_{i=1}^{T}p_i=1$, $\gamma=[\gamma_1,...,\gamma_T]$.

Таким образом, одновременно находятся и пример x^*, и аннотатор к нему. Конечно, x^* может не оказаться среди данных. Тогда выбирается ближайший к нему по Евклидовой метрике.

Нетрудно выделить общую схему алгоритмов активного обучения с несколькими аннотаторами. Сначала из неразмеченного множества выбирается самый информативный пример:

$$x_{i+1}=\arg\min_{x\in X_u}A(x)$$

Затем выбирается подходящий аннотатор, который больше всего уверен в нем:

$$t_{i+1}=\arg\max_{t\in T}Q_t(x_{i+1})$$

И в конце цикла все необходимые параметры переоцениваются.

Следовательно, нужно подобрать функции неуверенности классификатора $A(x)$ и надежности исполнителей $Q_t(x)$; при этом желательно, чтобы классификатор и аннотаторы были связаны. В качестве A и Q в работе [7] используется энтропия предсказаний классов и линейная комбинация признаков соответственно, в работе [54] – SVM с радиальными базисными функциями в обоих случаях.

Одним из недостатков таких подходов является то, что модель склонна отдавать предпочтения одним и тем же исполнителям. Зависимость модели от одного аннотатора является нежелательным эффектом: предсказания алгоритма будут смещены, и в какой-то момент модель начнет считать все метки этого исполнителя истинными. К тому же у других участников пропадает возможность проявить себя.

Родригес и др. [36] предложили для оценки характеристики исполнителя использовать два параметра: α_j – чувствительность и β_j – специфичность, а для разметки выбирать исполнителя с наибольшим ожиданием предоставить правильный ответ:

$$j^*=\arg\max_j[\alpha_j p(y=1|x^*,L,Y)+\beta_j(1-p(y=1|x^*,L,Y))]$$

L – множество размеченных примеров, Y – ответы исполнителей. В качестве $p(y=1|x,L,Y)$ используется адаптированная к нескольким аннотаторам модель гауссовского классификатора [33]. Пример x^* выбирается активным обучением. Поскольку α_j и β_j вычисляются исходя из оцененных меток, предлагается не учитывать метки аннотатора при оценке его параметров. Например, если чувствительность вычисляется следующим образом:

$$\alpha_j=\frac{\sum_{i=1}^{N}y_i^j p(y_i=1|L,Y)}{\sum_{i=1}^{N}p(y_i=1|L,Y)}$$

то вероятности $p(y_i=1|L,Y)$ заменяются на $p(y_i=1|L\backslash L^j,Y\backslash Y^j)$. Таким образом решается проблема зависимости от одного исполнителя.

```
L, U - множество размеченных и неразмеченных примеров
A - множество аннотаторов
Iter - число итераций
1: for i:=1 to Iter do:
     // Выбрать пример для разметки из множества U с учетом имеющихся ответов в L
2:   x = U.sample(L)
     // Выбрать аннотатора для разметки x, на основе имеющихся ответов в L
3:   j = A.choice(x, L)
     // Получить метку для x
4:   y = get_answer(j, x)
5:   L.Update(x, y)
```

Алгоритм 2. Активное обучение с выбором аннотатора
Algorithm 2. Active learning with annotator selection

Иногда выбираются сразу нескольких аннотаторов, или аннотаторы выбираются случайно – пропорционально надежности. Некоторые алгоритмы рассчитаны на возможность привлечения специалистов: если согласие исполнителей низкое, то пример может быть предоставлен на экспертную оценку [11, 30], а в работе [46] после выбора примера сразу решается, следует ли привлечь эксперта или же обычного исполнителя к его разметке.

Велиндер и Перона [43] реализовали онлайн версию EM-алгоритма. Аннотатор описывается вероятностью предоставления правильного ответа на каждом классе. Далее вводится несколько типов аннотаторов: E – эксперты, B – множество аннотаторов, дающих некачественные ответы, и остальные. Итерации происходят следующим образом: очередной пример предоставляется на разметку аннотатору из множества E (если оно пусто, то любому аннотатору не из B). Затем оцениваются $p(y)$ – апостериорное распределение классов данного объекта, что соответствует E шагу. Процедура продолжается до тех пор, пока выполняется условие $\max_y p(y)<\tau$, где τ – порог. То есть разметка продолжается до тех пор, пока нет уверенности в какой-либо метке, либо пока не достигнуто максимальное число шагов. После заново оцениваются параметры исполнителей (M шаг), и на основе этого пересчитываются множества E и B.

Отметим, что онлайн выбор аннотатора в некотором смысле относится к задаче многоруких бандитов, но отклик (награда) не может быть вычислен явно – сложно определить полезность исполнителя по одному ответу. Поэтому приходится использовать различные эвристики. Так, в [27] модель в каждый момент времени находится в одном из двух режимов: исследование (exploration) – оценка компетентности исполнителей или использование (exploitation).

Пусть $E(t)$ – множество примеров, которые были размечены в первом режиме до момента времени t. Если $|E(t)| < D_1 log(t)$ или имеется пример $x_k \in E(t)$, размеченный меньше чем $D_2 log(t)$ раз, то в момент $t+1$ на разметку всем исполнителям предоставляется либо новый пример, либо x_k соответственно. Затем взвешенным голосованием переоцениваются истинные метки и оцениваются надежности исполнителей. Это режим исследования. В режиме использования случайно выбирается новый пример и предоставляется самым надежным участникам.

В работе [41] предложен подход к решению задачи привлечения фриланс-работников с неизвестным рейтингом и разной стоимостью. В этом случае отклик оценивается как качество проделанной работы. Отличие от задачи многоруких бандитов состоит в ограничении на количество выполненных заданий одним участником (может физически не хватить времени) и также в том, что за один раз можно дать одно и то же задание многим людям (дернуть за несколько ручек).

Формально задача в работе [41] ставится так: даны бюджет B и участники со стоимостью одного задания c_i, ограничением на количество заданий l_i и неизвестным распределением полезности. Необходимо распределить задания так, чтобы максимизировать сумму полезностей при ограничении на бюджет.

Решение делится на 2 фазы. На первом этапе выбирается такое ϵ, что, пока это возможно, тратится ϵB бюджета. Участники упорядочиваются по возрастанию c_i и по циклу получают задания. За задание выставляется оценка. Для каждого исполнителя оценивается математическое ожидание полезности $\hat{\mu}_i$ – среднее оценок. На втором этапе при найденной полезности $\hat{\mu}_i$, стоимости c_i, бюджете $(1-\epsilon)B$ и ограничениях l_i эвристическими методами решается известная в теории сложности вычислений задача о рюкзаке.

4.2 Онлайн разметка

С точки зрения организации процесса разметки итеративное планирование неэффективно использует доступные ресурсы. Нужно ждать, пока конкретный исполнитель выполнит задание, нет возможности загрузить работой свободных аннотаторов, и тем самым параллельная разметка затруднительна. Если разметку выполняют не участники крауд-платформ, а заинтересованные люди с высокой компетентностью, оптимизация процесса становится критичной.

Хотелось бы, чтобы планирование выглядело так. Скажем, каждый участник готов уделить аннотированию несколько минут в день. Он в удобное ему время заходит в систему и выполняет задание в режиме онлайн: система выдает по одному или по несколько примеров, обрабатывает ответы и с минимальной задержкой предоставляет следующее задание. Под такие критерии попадает простейший случай разметки – достаточно выбирать задания случайным образом, принимая во внимание только то, что одному человеку допустимо

разметить не больше одного примера. Но существуют ли оптимизированные методы онлайн разметки, если целью является построение качественного классификатора в задаче? В частности, возможно ли применение активного обучения в таком случае?

Процесс онлайн активного обучения с несколькими аннотаторами можно организовать в виде двух очередей. Первая очередь содержит примеры, ожидающие разметку, она пополняется пакетами, состоящими из нескольких примеров. Вторая очередь хранит ответы участников. Когда обработано определенное количество ответов, запускается очередная итерация активного обучения, и очередь обновляется.

Такой асинхронный процесс позволяет сократить время ожидания аннотатора, пока система генерирует следующий вопрос, но, к сожалению, он неэффективен, если качество аннотаций невысокое, и каждый пример требуется разметить несколькими людьми, прежде чем перейти к следующей итерации. Такая схема обсуждается в [10] и [20]. Когда пользователь открывает задание на крауд-платформе, система перенаправляет его на сервер заказчика, где объекты для разметки предоставляются в режиме онлайн.

В работе [12] для распределения заданий было предложено весь неразмеченный датасет D заранее разбивать на части $D = D_1 \cup ... \cup D_K$, $D_i = U_i \cup \mathcal{L}_i$, где K – число исполнителей, $\mathcal{L}$ и U_i – множества размеченных и неразмеченных примеров соответственно. Каждый пример попадает в одно и то же число частей: $|\{D_k | x_i \in D_k\}| = m$, $\forall x_i \in D$. Для каждого датасета D_i обучается отдельный классификатор $f_i(x)$. Функция ошибки $L(D)$ пытается одновременно оптимизировать все классификаторы $f_i(x)$:

$$L(D) = \sum_{i=1}^{K} \sum_{x_j \in \mathcal{L}_i} L_i(y_i^j, f_i(x_j)) + \\ + \sum_{1 \le i \ne j \le K} \sum_{x_k \in D_i \cap \mathcal{L}_j} L_{ij}(y_k^j, f_i(x_k)) + \\ + \lambda \sum_{i=1}^{K} \Omega(||f_i||_H)$$

Здесь L_i – функция ошибки классификатора i, а L_{ij} учитывает ошибку классификатора i на примерах, размеченных участником j и содержащихся в множестве $\mathcal{L}_i$; последнее слагаемое отвечает за регуляризацию. Таким образом, классификаторы не являются независимыми.

Разметка происходит активным обучением. Для исполнителя i выбирается пример, который находится ближе всего к границе решающего правила $f_i(x)$.

После разметки параметры, связанные с аннотатором i оптимизируются согласно формуле; остальные параметры считаются фиксированными. В

качестве классификаторов в работе [12] используется метод опорных векторов. Итоговый классификатор получается усреднением алгоритмов $f_i(x)$.

input: множество неразмеченных примеров U
output: метки $\hat{\mathbf{y}} = [\hat{y}_1, ..., \hat{y}_N]$, качество участников $\mathbf{q} = [\hat{q}_1, ..., \hat{q}_K]$
L - матрица ответов, M - матрица распределений классов для всех примеров
1: Инициализировать $\mathbf{q}$
2: Выполнять:
// Получить номер исполнителя
3: j = get_requestor()
// Выбрать подходящий пример на основе q_j и априорных распределений M
4: i = U.sample(q_j, M)
5: L.Update(i, j, get_answer(j, x_i))
// Переоценить M на основе имеющихся ответов L и надежностей $\mathbf{q}$
6: M.Update(L, $\mathbf{q}$)
// Для каждого исполнителя j оценить q_j на основе M и L
7: for j:=1 to K do:
8: q_j.Update(L, M)

Алгоритм 3. Вывод в онлайн разметке
Algorithm 3. Inference in online annotation

Существуют несколько систем, которые поддерживают онлайн разметку произвольных данных, но не связаны с построением классификатора: DOCS [51], QASQA [53], iCrowd [5]. В качестве вывода обычно применяется итерационная схема с байесовским голосованием, по сути это онлайн-версия Алгоритма 1: при получении очередной аннотации примера сначала переоценивается ожидаемая метка для этого объекта, а затем с учетом обновленной метки заново вычисляются параметры исполнителей (т.е. в отличие от Алгоритма 1 параметры оцениваются после каждой аннотации, а не после получения всех аннотаций).

Опишем, как происходит назначение задания. В каждый момент времени для всех примеров хранится текущее апостериорное распределение меток в виде матрицы $M_{i,k}$, содержащей вероятности того, что пример i принадлежит классу k. Для исполнителя j оцениваются ожидаемые ответы: $Q_{ia}^{(j)}$ – вероятность предоставить класс a примеру i. В качестве априорного распределения классов выбирается матрица M. В простейшем случае, когда исполнитель описывается только вероятностью q_j предоставить правильный ответ, значения $Q_{ia}^{(j)}$ вычисляются так:

$$Q_{ia}^{(j)} = q_j M_{i,a} + \frac{1-q_j}{J-1}(1 - M_{i,a})$$

Для каждого возможного ответа a переоценивается матрица M:

$$M_{i,k}^{|a} \propto M_{i,k}(q_j)^{\mathbb{1}(a=k)}(\frac{1-q_j}{J-1})^{\mathbb{1}(a\neq k)}$$

В итоге выбираются те примеры, на которых максимально изменение метрики неопределенности. Например, в DOCS это разность текущей и ожидаемой энтропии: $H(M_i) - H(M_i^{|})$, где

$$H(M_i^{|}) = \sum_{a=1}^{J} H(M_i^{|a}) Q_{ia}^{(j)}$$

Алгоритм 3 описывает подход в общем виде.

5. Обзор фреймворков

5.1 Крауд-платформы

В этом подразделе описываются несколько известных платформ для работы с крауд-вычислениями. В таких платформах есть два типа участников: заказчики, которые публикуют задания, и участники-исполнители или аннотаторы. Задания, как правило, представляются в виде набора из нескольких примеров для которых требуется аннотация, их называют HIT (Human Intelligence Task). Это могут быть различные задачи классификации, машинного перевода, сопоставления сущностей и многие другие.

Опишем подробнее использование одной из таких платформ – Яндекс.Толока. Перед публикацией заданий нужно создать проект и написать инструкцию к выполнению заданий. Затем необходимо оформить интерфейс в виде HTML текста. Задания можно добавлять набором по несколько штук, называемых пулом. Для каждого пула указывается максимальное время выполнения, а также перекрытие – количество пользователей, которые должны выполнить задание. Агрегация меток производится большинством голосов.

Для дополнительного контроля качества добавляются контрольные задания с заранее известными ответами, которые для исполнителя внешне ничем не отличаются от обычных. Таким образом, имеется возможность блокирования пользователей, которые либо часто ошибаются на контрольных вопросах, либо выполняют задания подозрительно быстро. Имеется также возможность добавления обучающих заданий, являющихся квалификационным тестом. После запуска в системе отображается прогресс по задаче.

Наиболее популярными англоязычными краудсорсинговыми платформами являются MTurk и CrowdFlower. Они предоставляют готовые шаблоны для оформления заданий, кроме того есть интерфейс для построения собственного дизайна задания средствами CSS и Javascript. Кроме обычного подхода, применяемого в Яндекс.Толока, существуют несколько других способов использования системы. Во-первых, имеются API, позволяющие выполнять операции добавления заданий, получать различные статистики и ответы на задания с помощью высокоуровневых языков программирования. Во-вторых, при выполнении задания платформа может перенаправлять пользователей на

сайт заказчика. Это может быть особенно полезно при использовании онлайн методов разметки.

5.2 Крауд-оптимизаторы

Существует несколько систем, предназначенных для упрощения и оптимизации работы с краудплатформами: хранения и обработки информации об аннотаторах и полученных аннотаций, формирования заданий и отправки их на крауд-платформу, улучшения качества вывода меток. Такие системы выступают посредниками между платформой и заказчиком. Эти системы уже упоминались в предыдущих частях. Важно, что они имеют косвенное отношение к разметке тренировочного множества для алгоритмов машинного обучения, их задачей является получение аннотаций для заданного набора примеров. И признаки объектов практически не используются.

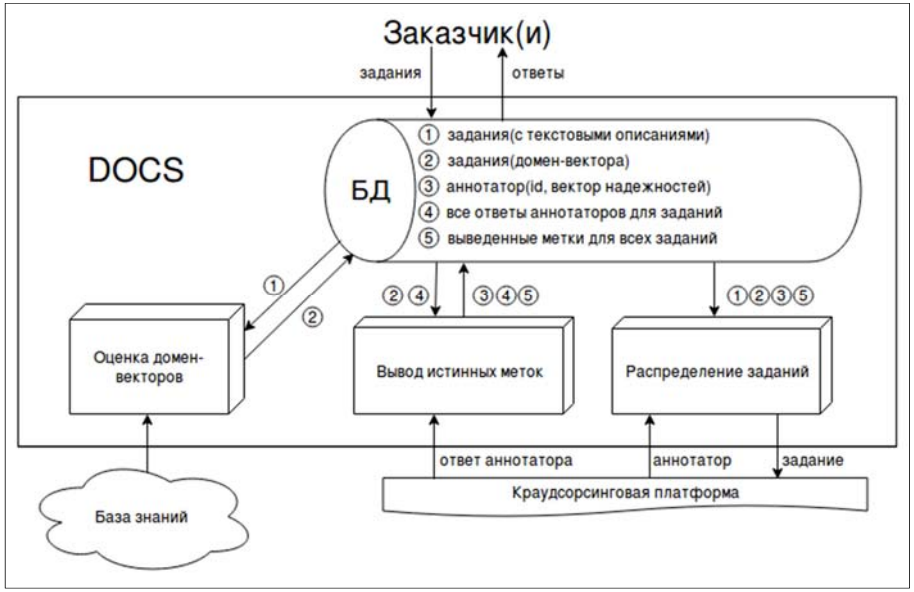


Рис. 3. Архитектура DOCS
Fig. 3. DOCS architecture

В самой простой такой системе Askit [1] не вводятся дополнительные параметры, и алгоритм работает только с полученными метками. Из множества примеров для повторной разметки выбираются те, у которых максимальна мера неопределенности. Для каждой возможной метки примера вычисляется энтропия всех меток, если к ним добавить новую. Для подсчета

неопределенности предлагается два способа: максимум этих значений и их среднее. Система CDAS [26] моделирует надежности исполнителей и определяет необходимое число вопросов для достижения определенного качества.

К другим подходам применима схема, описанная в предыдущей части для онлайн разметки. В QASCA [53] аннотатор моделируется вероятностью предоставить верный класс, а вероятность ответить любым неверным классом считается одинаковой. В качестве метрики выбирается не разность энтропии, а разность вероятностей наиболее уверенных классов в ожидаемом от аннотатора распределении и текущем.

В iCrowd [5] и DOCS [51] для оценки ожидаемых ответов $Q_{ia}^{(j)}$ используются более сложные методы, связанные с идеей зависимости компетентности исполнителя от темы задания. В iCrowd строится взвешенный граф похожести примеров.

В DOCS объект и аннотатор описываются векторами из нескольких чисел, соответствующих доменам. Объект – это вектор распределения доменов, аннотатор – вектор вероятностей предоставить правильный ответ на каждом домене. Возникает три задачи.

Во-первых, по имеющимся примерам нужно определить их вектора доменов. Постановка аналогична постановке задачи тематического моделирования [56]. Но авторы вместо использования стандартных методов предлагают свой подход. Для примера выделяют все сущности. Затем для каждой сущности находят распределения концептов. Эти два шага осуществляются с помощью готового викификатора. Каждый концепт имеет бинарный вектор домена (принадлежит/не принадлежит). Чтобы для данного набора концептов посчитать домен-вектор, нужно сложить эти бинарные вектора по всем сущностям и нормализовать. А чтобы посчитать вектор для всего примера, нужно найти математическое ожидание по распределению концептов.

Во-вторых, после получения каждой аннотации нужно переоценить надежности и метки. Это производится стандартным итеративным алгоритмом, но с учетом доменов. Сначала для каждого домена байесовским голосованием оценивается распределение меток для всех примеров. Окончательное распределение получается взвешенным суммированием этих распределений пропорционально весам доменов. Затем переоцениваются надежности аннотаторов на каждом домене. Для ускорения переоцениваются только параметры, непосредственно связанные с новой аннотацией – распределение меток соответствующего примера и компетентности аннотаторов, разметивших этот пример.

И третья задача – подбор задания участнику. Если он еще не выполнил ни одного задания, то предоставляется тестовое множество примеров. Иначе оцениваются метки, которые ожидаются от аннотатора (в зависимости от его

компетентности на доменах) и выбираются те примеры, для которых максимальна разность энтропии имеющихся меток и ожидаемых.

Авторы DOCS провели подробные сравнения системы с Askit, QASCA, iCrowd на крауд-платформе MTurk и с алгоритмами тематического моделирования для выявления доменов. Причём для каждой системы запускались независимые процессы разметки. Результаты показали, что DOCS выигрывает у конкурентов по всем показателям.

5.3 Крауд-СУБД

Популярными являются крауд системы, которые рассматривают данные в виде реляционных баз Qurk [28], Deco [32], CDB [22], CrowdOp [6]. В общем случае они устроены следующим образом. Пользователь загружает в систему данные в виде таблиц, возможно с пропущенными значениями, и вводит обычный SQL запрос. Система анализирует запрос, затем генерирует план в виде простых вопросов на краудплатформу и производит крауд оптимизации при исполнении этого плана. В одной из первой такой системе CrowdDB вводят три типа вопросов: заполнение пропущенных значений, сопоставление одинаковых объектов (join) и сравнение объектов по какомулибо показателю.

Важной задачей таких систем является построение оптимального плана запросов. Так, в Deco (2012), CrowdOP (2015) оптимизация строится на основе деревьев, а в недавней работе CDB (2017) на основе взвешенного графа объектов.

6. Заключение

Тема разметки данных с помощью краудсорсинга активно изучается последние несколько лет. За это время появилось много различных методов и реализаций. Так как основной особенностью крауд-систем является различный уровень компетентности и мотивированности исполнителей, то главная черта решений – это оценка уровня надежности исполнителей.

Оффлайн разметка является наиболее исследованной темой, это подтверждает наличие большого количества обзоров и сравнений [23, 48, 52, 55]. Для задач бинарной классификации лидерами остаются алгоритмы основанные на модели матрицы ошибок [3] и его модификации. Для многоклассовых задач имеется небольшое количество алгоритмов. Проблемой остается разрыв между разметкой данных и построением тренировочного множества для классификатора – практически все алгоритмы слабо учитывают признаковое описание примеров.

Более успешным является использование итеративного процесса разметки. Полагаем, что дальнейший фокус исследований будет смещен в эту сторону, в частности, к методам активного обучения. Здесь необходимо устранить эффект зависимости от одного исполнителя. Онлайн активное обучение лишено этого

недостатка, и также позволяет распараллелить работу исполнителей. Однако такие методы встречаются редко.

Также заметим, что акцент в современных работах смещается в сторону практических реализаций, набирают популярность крауд-оптимизаторы и их аналоги. К сожалению, основной целью таких работ является не построение классификатора для данной задачи, а точная разметка данных.

Список литературы

[1]. Rubi Boim, Ohad Greenshpan, Tova Milo, Slava Novgorodov, Neoklis Polyzotis, and Wang-Chiew Tan. Asking the right questions in crowd data sourcing. In Proc. of the 28th International Conference on Data Engineering (ICDE), 2012, pp 1261–1264.

[2]. Anthony Brew, Derek Greene, and Pa´draig Cunningham. Using crowdsourcing and active learning to track sentiment in online media. In Proc. of the 19th European Conference on Artificial Intelligence, pp. 145–150.

[3]. P. Dawid, A. M. Skene, A. P. Dawidt, and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. Applied Statistics, vol. 28, № 1, 1979, pp. 20-28.

[4]. Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudr´e-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In Proc. of the 21st international conference on World Wide Web, 2012, pp. 469–478.

[5]. Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In Proc. of the ACM SIGMOD International Conference on Management of Data, 2015, pp. 1015–1030.

[6]. Ju Fan, Meihui Zhang, Stanley Kok, Meiyu Lu, and Beng Chin Ooi. Crowdop: Query optimization for declarative crowdsourcing systems. IEEE Transactions on Knowledge and Data Engineering, vol. 27, № 8, 2015, pp. 2078–2092.

[7]. Meng Fang, Xingquan Zhu, Bin Li, Wei Ding, and Xindong Wu. Self-taught active learning from crowds. In Proc. of the 12th International Conference on Data Mining (ICDM), 2012, pp. 858–863.

[8]. Paul Felt, Robbie Haertel, Eric K Ringger, and Kevin D Seppi. Momresp: A bayesian model for multi-annotator document labeling. In Proc. of the Ninth International Conference on Language Resources and Evaluation, 2014. pp. 3704–3711.

[9]. Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators?: crowdsourcing abuse detection in user-generated content. In Proc. of the 12th ACM conference on Electronic commerce, 2011, pp. 167–176.

[10]. Daniel Haas, Jiannan Wang, Eugene Wu, and Michael J. Franklin. Clamshell: Speeding up crowds for low-latency data labeling. Proc. of the VLDB Endowment, vol. 9, № 4, 2015, pp. 372–383.

[11]. Shuji Hao, Steven C. H. Hoi, Chunyan Miao, and Peilin Zhao. Active crowdsourcing for annotation. In Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, vol. II, 2015, pp. 1–8.

[12]. Gang Hua, Chengjiang Long, Ming Yang, and Yan Gao. Collaborative active learning of a kernel machine ensemble for recognition. In Proc. of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 1209– 1216.

[13]. Nguyen Quoc Viet Hung, Nguyen Thanh Tam, Lam Ngoc Tran, and Karl Aberer. An evaluation of aggregation techniques in crowdsourcing. In Proc. of the International Conference on Web Information Systems Engineering, 2013, pp. 1–15.

[14]. Hiroshi Kajino, Yuta Tsuboi, and Hisashi Kashima. A convex formulation for learning from crowds. Transactions of the Japanese Society for Artificial Intelligence, vol. 27, № 3, , 2012, pp. 133–142.

[15]. David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 24), 2011, pp. 1953–1961.

[16]. Faiza Khan Khattak. Toward a Robust and Universal Crowd Labeling Framework. PhD Thesis, Columbia University, 2017, 168 p.

[17]. Faiza Khan Khattak and Ansaf Salleb-Aouissi. Quality control of crowd labeling through expert evaluation. In Proceedings of the NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds, vol. 2, 2011, 5 p.

[18]. Adam Kilgarriff and Adam Kilgarriff. Gold standard datasets for evaluating word sense disambiguation programs. Computer Speech and Language, vol. 12, № 3, 1998, pp. 453-472.

[19]. Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In Proc. of the Fifteenth International Conference on Artificial Intelligence and Statistics, 2012, pp. 619–627, 2012.

[20]. Florian Laws, Christian Scheible, and Hinrich Schütze. Active learning with amazon mechanical turk. In Proc. of the Conference on Empirical Methods in Natural Language Processing, 2011, pp. 1546–1556.

[21]. Kyumin Lee, James Caverlee, and Steve Webb. The social honeypot project: protecting online communities from spammers. In Proc. of the 19th International Conference on World Wide Web, 2010, pp 1139–1140.

[22]. Guoliang Li, Chengliang Chai, Ju Fan, Xueping Weng, Jian Li, Yudian Zheng, Yuanbing Li, Xiang Yu, Xiaohang Zhang, and Haitao Yuan. Cdb: optimizing queries with crowd-based selections and joins. In Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data, 2007, pp. 1463–1478.

[23]. Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael J Franklin. Crowdsourced data management: A survey. IEEE Transactions on Knowledge and Data Engineering, vol. 28, № 9, 2016, pp. 2296–2319.

[24]. Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. A survey on truth discovery. ACM SIGKDD Explorations Newsletter, vol. 17, № 2, 2016, pp. 1–16.

[25]. Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 25), 2012, pp. 692–700.

[26]. Xuan Liu, Meiyu Lu, Beng Chin Ooi, Yanyan Shen, Sai Wu, and Meihui Zhang. Cdas: a rowdsourcing data analytics system. Proc. of the VLDB Endowment, vol. 5, № 10, 2012, pp. 1040–1051.

[27]. Yang Liu and Mingyan Liu. An online learning approach to improving the quality of crowd-sourcing. ACM SIGMETRICS Performance Evaluation Review, vol. 43, 2015, pp. 217–230.

[28]. Adam Marcus, Eugene Wu, David R Karger, Samuel Madden, and Robert C Miller. Crowdsourced databases: Query processing with people. Proc. of the 5th Conference on Innovative Data Systems Research (CIDR). 2011, pp. 211-214.

[29]. Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. Scaling up crowdsourcing to very large datasets: a case for active learning. Proc. of the VLDB Endowment, vol. 8, № 2, 2014, pp. 125–136.

[30]. An Thanh Nguyen, Byron C Wallace, and Matthew Lease. Combining crowd and expert labels using decision theoretic active learning. In Proc. of the Third AAAI Conference on Human Computation and Crowdsourcing, 2015, pp. 120-129.

[31]. Stefanie Nowak and Stefan Ru¨ger. How reliable are annotations via crowdsourcing: A study about interannotator agreement for multi-label image annotation. In Proc. of the International Conference on Multimedia Information Retrieval, 2010, pp. 557– 566.

[32]. Aditya Ganesh Parameswaran, Hyunjung Park, Hector Garcia-Molina, Neoklis Polyzotis, and Jennifer Widom. Deco: declarative crowdsourcing. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 1203–1212.

[33]. Carl Edward Rasmussen. Gaussian processes in machine learning. In Advanced Lectures on Machine Learning, Lecture Notes in Computer Science, vol 3176, 2004, pp. 63–71.

[34]. Vikas C Raykar, Shipeng Yu, Linda H Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In Proc. of the 26th Annual international conference on machine learning, 2009, pp. 889–896.

[35]. Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. Journal of Machine Learning Research, vol. 11, 2010, pp. 1297–1322.

[36]. Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Gaussian process classification and active learning with multiple annotators. In Proc. of the International Conference on Machine Learning, 2014, pp. 433–441.

[37]. Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin– Madison, 2009, 65 p.

[38]. Victor S. Sheng, Foster J. Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 614– 622.

[39]. Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In Proc. of the First AAAI Conference on Human Computation and Crowdsourcing, 2013, pp. 156-164.

[40]. Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast–but is it good?: Evaluating non-expert annotations for natural language tasks. In Proc. of the Conference on Empirical Methods in Natural Language Processing, 2008, pp. 254–263.

[41]. Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. Artificial Intelligence, vol. 214, issue 1, 2014, pp. 89–111.

[42]. Fabian L Wauthier and Michael I Jordan. Bayesian bias mitigation for crowdsourcing. In In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 24), 2011, pp. 1800–1808.

[43]. Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, 2010, pp. 25–32.
[44]. Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Proc. of the Neural Information Processing Systems 2009 Conference (Advances in neural information processing systems 22), 2009, pp. 2035–2043.
[45]. Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. Active learning from crowds. In Proc. of the 28th International Conference on International Conference on Machine Learning, 2011, pp. 1161–1168.
[46]. Chicheng Zhang and Kamalika Chaudhuri. Active learning from weak and strong labelers. In Proc. of the Neural Information Processing Systems 2015 Conference (Advances in neural information processing systems 28), 2015, pp. 703–711.
[47]. Jing Zhang, Victor S Sheng, Jian Wu, and Xindong Wu. Multi-class ground truth inference in crowdsourcing with clustering. IEEE Transactions on Knowledge and Data Engineering, vol. 28, № 4, 2016, pp. 1080–1085.
[48]. Jing Zhang, Xindong Wu, and Victor S Sheng. Learning from crowdsourced labeled data: a survey. Artificial Intelligence Review, vol. 46, № 4, 2016, pp. 543–576.
[49]. Yuchen Zhang, Xi Chen, Denny Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In Proc. of the Neural Information Processing Systems 2014 Conference (Advances in neural information processing systems 27), 2014, pp. 1260–1268.
[50]. Yudian Zheng, Reynold Cheng, Silviu Maniu, and Luyi Mo. On optimality of jury selection in crowdsourcing. In Proceedings of the 18th International Conference on Extending Database Technology, 2015, pp. 193-204.
[51]. Yudian Zheng, Guoliang Li, and Reynold Cheng. Docs: a domain-aware crowdsourcing system using knowledge bases. Proc. of the VLDB Endowment, vol. 10, № 4, 2016, pp. 361– 372.
[52]. Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? Proc. of the VLDB Endowment, vol. 10, № 5, 2017, pp. 541–552.
[53]. Yudian Zheng, Jiannan Wang, Guoliang Li, Reynold Cheng, and Jianhua Feng. Qasca: A quality-aware task assignment system for crowdsourcing applications. In Proceedings of the ACM SIGMOD International Conference on Management of Data, 2015, pp. 1031–1046.
[54]. Jinhong Zhong, Ke Tang, and Zhi-Hua Zhou. Active learning from crowds with unsure option. In Proc. of the 24th International Conference on Artificial Intelligence, 2015, pp. 1061–1068.
[55]. Пономарев А. В. Методы обеспечения качества в системах крауд-вычислений: аналитический обзор. Труды СПИИРАН, вып. 54, 2017, pp. 152–184. DOI: 10.15622/sp.54.7
[56]. Коршунов А., Гомзин А. Тематическое моделирование текстов на естественном языке. Труды ИСП РАН, 23, 2012, стр. 215-244. DOI: 10.15514/ISPRAS-2012-23-13

Active learning and crowdsourcing: a survey of annotation optimization methods

[1,2]R. A. Gilyazev <gilyazev@ispras.ru>
[1,3,4]D. U. Turdakov <turdakov@ispras.ru>
[1]Ivannikov Institute for System Programming of the RAS,
25, Alexander Solzhenitsyn Str., Moscow, 109004, Russia
[2]Moscow Institute of Physics and Technology (State University),
9, Institutskiy per., Dolgoprudny,
[3]Lomonosov Moscow State University,
GSP-1, Leninskie Gory, Moscow, 119991, Russia Moscow Region, 141700, Russia
[4]National Research University Higher School of Economics (HSE),
20, Myasnitskaya Ulitsa, Moscow, 101000, Russia

Abstract. High quality labeled corpora play a key role to elaborate machine learning systems. Generally, creating of such corpora requires human efforts. So, annotation process is expensive and time-consuming. Two approaches that optimize the annotation are active learning and crowdsourcing. Methods of active learning are aimed at finding the most informative examples for the classifier. At each iteration from the unplaced set, one algorithm is chosen by an algorithm, it is provided to the oracle (expert) for the markup and the classifier is newly trained on the updated set of training examples. Crowdsourcing is widely used in solving problems that can not be automated and require human effort. To get the most out of using crowdplatforms one needs to to solve three problems. The first of these is quality, that is, algorithms are needed that will best determine the real labels from the available ones. Of course, it is necessary to remember the cost of markup - to solve the problem by increasing the number of annotators for one example is not always reasonable - this is the second problem. And, thirdly, sometimes the immediate factor is the rapid receipt of the marked corpus, then it is necessary to minimize the time delays when the participants perform the task. This paper aims to survey existing methods based on this approaches and techniques to combine them. Also, the paper describes the systems that help to reduce the cost of annotation.

Keywords: active learning; crowdsourcing; learning from crowds; annotation; ground truth inference

DOI: 10.15514/ISPRAS-2018-30(1)-11

For citation: Gilyazev R.A., Turdakov D.Y. Active learning and crowdsourcing: a survey of data markup optimization methods. Trudy ISP RAN/Proc. ISP RAS, vol. 30, issue 2, 2018, pp. 215-250 (in Russian). DOI: 10.15514/ISPRAS-2018-30(1)-11

References

[1]. Rubi Boim, Ohad Greenshpan, Tova Milo, Slava Novgorodov, Neoklis Polyzotis, and Wang-Chiew Tan. Asking the right questions in crowd data sourcing. In Proc. of the 28th International Conference on Data Engineering (ICDE), 2012, pp 1261–1264.
[2]. Anthony Brew, Derek Greene, and Pa´draig Cunningham. Using crowdsourcing and active learning to track sentiment in online media. In Proc. of the 19th European Conference on Artificial Intelligence, pp. 145–150.

[3]. P. Dawid, A. M. Skene, A. P. Dawidt, and A. M. Skene. Maximum likelihood estimation of observer error-rates using the em algorithm. Applied Statistics, vol. 28, № 1, 1979, pp. 20-28.

[4]. Gianluca Demartini, Djellel Eddine Difallah, and Philippe Cudr´e-Mauroux. Zencrowd: leveraging probabilistic reasoning and crowdsourcing techniques for large-scale entity linking. In Proc. of the 21st international conference on World Wide Web, 2012, pp. 469–478.

[5]. Ju Fan, Guoliang Li, Beng Chin Ooi, Kian-lee Tan, and Jianhua Feng. icrowd: An adaptive crowdsourcing framework. In Proc. of the ACM SIGMOD International Conference on Management of Data, 2015, pp. 1015–1030.

[6]. Ju Fan, Meihui Zhang, Stanley Kok, Meiyu Lu, and Beng Chin Ooi. Crowdop: Query optimization for declarative crowdsourcing systems. IEEE Transactions on Knowledge and Data Engineering, vol. 27, № 8, 2015, pp. 2078–2092.

[7]. Meng Fang, Xingquan Zhu, Bin Li, Wei Ding, and Xindong Wu. Self-taught active learning from crowds. In Proc. of the 12th International Conference on Data Mining (ICDM), 2012, pp. 858–863.

[8]. Paul Felt, Robbie Haertel, Eric K Ringger, and Kevin D Seppi. Momresp: A bayesian model for multi-annotator document labeling. In Proc. of the Ninth International Conference on Language Resources and Evaluation, 2014. pp. 3704–3711.

[9]. Arpita Ghosh, Satyen Kale, and Preston McAfee. Who moderates the moderators?: crowdsourcing abuse detection in user-generated content. In Proc. of the 12th ACM conference on Electronic commerce, 2011, pp. 167–176.

[10]. Daniel Haas, Jiannan Wang, Eugene Wu, and Michael J. Franklin. Clamshell: Speeding up crowds for low-latency data labeling. Proc. of the VLDB Endowment, vol. 9, № 4, 2015, pp. 372–383.

[11]. Shuji Hao, Steven C. H. Hoi, Chunyan Miao, and Peilin Zhao. Active crowdsourcing for annotation. In Proc. of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology, vol. II, 2015, pp. 1–8.

[12]. Gang Hua, Chengjiang Long, Ming Yang, and Yan Gao. Collaborative active learning of a kernel machine ensemble for recognition. In Proc. of the IEEE International Conference on Computer Vision (ICCV), 2013, pp. 1209– 1216.

[13]. Nguyen Quoc Viet Hung, Nguyen Thanh Tam, Lam Ngoc Tran, and Karl Aberer. An evaluation of aggregation techniques in crowdsourcing. In Proc. of the International Conference on Web Information Systems Engineering, 2013, pp. 1–15.

[14]. Hiroshi Kajino, Yuta Tsuboi, and Hisashi Kashima. A convex formulation for learning from crowds. Transactions of the Japanese Society for Artificial Intelligence, vol. 27, № 3, , 2012, pp. 133–142.

[15]. David R Karger, Sewoong Oh, and Devavrat Shah. Iterative learning for reliable crowdsourcing systems. In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 24), 2011, pp. 1953–1961.

[16]. Faiza Khan Khattak. Toward a Robust and Universal Crowd Labeling Framework. PhD Thesis, Columbia University, 2017, 168 p.

[17]. Faiza Khan Khattak and Ansaf Salleb-Aouissi. Quality control of crowd labeling through expert evaluation. In Proceedings of the NIPS 2nd Workshop on Computational Social Science and the Wisdom of Crowds, vol. 2, 2011, 5 p.

[18]. Adam Kilgarriff and Adam Kilgarriff. Gold standard datasets for evaluating word sense disambiguation programs. Computer Speech and Language, vol. 12, № 3, 1998, pp. 453-472.

[19]. Hyun-Chul Kim and Zoubin Ghahramani. Bayesian classifier combination. In Proc. of the Fifteenth International Conference on Artificial Intelligence and Statistics, 2012, pp. 619–627, 2012.

[20]. Florian Laws, Christian Scheible, and Hinrich Schütze. Active learning with amazon mechanical turk. In Proc. of the Conference on Empirical Methods in Natural Language Processing, 2011, pp. 1546–1556.

[21]. Kyumin Lee, James Caverlee, and Steve Webb. The social honeypot project: protecting online communities from spammers. In Proc. of the 19th International Conference on World Wide Web, 2010, pp 1139–1140.

[22]. Guoliang Li, Chengliang Chai, Ju Fan, Xueping Weng, Jian Li, Yudian Zheng, Yuanbing Li, Xiang Yu, Xiaohang Zhang, and Haitao Yuan. Cdb: optimizing queries with crowd-based selections and joins. In Proceedings of the 2017 ACM SIGMOD International Conference on Management of Data, 2007, pp. 1463–1478.

[23]. Guoliang Li, Jiannan Wang, Yudian Zheng, and Michael J Franklin. Crowdsourced data management: A survey. IEEE Transactions on Knowledge and Data Engineering, vol. 28, № 9, 2016, pp. 2296–2319.

[24]. Yaliang Li, Jing Gao, Chuishi Meng, Qi Li, Lu Su, Bo Zhao, Wei Fan, and Jiawei Han. A survey on truth discovery. ACM SIGKDD Explorations Newsletter, vol. 17, № 2, 2016, pp. 1–16.

[25]. Qiang Liu, Jian Peng, and Alexander T Ihler. Variational inference for crowdsourcing. In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 25), 2012, pp. 692–700.

[26]. Xuan Liu, Meiyu Lu, Beng Chin Ooi, Yanyan Shen, Sai Wu, and Meihui Zhang. Cdas: a rowdsourcing data analytics system. Proc. of the VLDB Endowment, vol. 5, № 10, 2012, pp. 1040–1051.

[27]. Yang Liu and Mingyan Liu. An online learning approach to improving the quality of crowd-sourcing. ACM SIGMETRICS Performance Evaluation Review, vol. 43, 2015, pp. 217–230.

[28]. Adam Marcus, Eugene Wu, David R Karger, Samuel Madden, and Robert C Miller. Crowdsourced databases: Query processing with people. Proc. of the 5th Conference on Innovative Data Systems Research (CIDR). 2011, pp. 211-214.

[29]. Barzan Mozafari, Purna Sarkar, Michael Franklin, Michael Jordan, and Samuel Madden. Scaling up crowdsourcing to very large datasets: a case for active learning. Proc. of the VLDB Endowment, vol. 8, № 2, 2014, pp. 125–136.

[30]. An Thanh Nguyen, Byron C Wallace, and Matthew Lease. Combining crowd and expert labels using decision theoretic active learning. In Proc. of the Third AAAI Conference on Human Computation and Crowdsourcing, 2015, pp. 120-129.

[31]. Stefanie Nowak and Stefan Ru¨ger. How reliable are annotations via crowdsourcing: A study about interannotator agreement for multi-label image annotation. In Proc. of the International Conference on Multimedia Information Retrieval, 2010, pp. 557– 566.

[32]. Aditya Ganesh Parameswaran, Hyunjung Park, Hector Garcia-Molina, Neoklis Polyzotis, and Jennifer Widom. Deco: declarative crowdsourcing. In Proceedings of the 21st ACM International Conference on Information and Knowledge Management, 2012, pp. 1203–1212.

[33]. Carl Edward Rasmussen. Gaussian processes in machine learning. In Advanced Lectures on Machine Learning, Lecture Notes in Computer Science, vol 3176, 2004, pp. 63–71.
[34]. Vikas C Raykar, Shipeng Yu, Linda H Zhao, Anna Jerebko, Charles Florin, Gerardo Hermosillo Valadez, Luca Bogoni, and Linda Moy. Supervised learning from multiple experts: whom to trust when everyone lies a bit. In Proc. of the 26th Annual international conference on machine learning, 2009, pp. 889–896.
[35]. Vikas C Raykar, Shipeng Yu, Linda H Zhao, Gerardo Hermosillo Valadez, Charles Florin, Luca Bogoni, and Linda Moy. Learning from crowds. Journal of Machine Learning Research, vol. 11, 2010, pp. 1297–1322.
[36]. Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. Gaussian process classification and active learning with multiple annotators. In Proc. of the International Conference on Machine Learning, 2014, pp. 433–441.
[37]. Burr Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin– Madison, 2009, 65 p.
[38]. Victor S. Sheng, Foster J. Provost, and Panagiotis G. Ipeirotis. Get another label? improving data quality and data mining using multiple, noisy labelers. In Proc. of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2008, pp. 614– 622.
[39]. Aashish Sheshadri and Matthew Lease. Square: A benchmark for research on computing crowd consensus. In Proc. of the First AAAI Conference on Human Computation and Crowdsourcing, 2013, pp. 156-164.
[40]. Rion Snow, Brendan O'Connor, Daniel Jurafsky, and Andrew Y. Ng. Cheap and fast–but is it good?: Evaluating non-expert annotations for natural language tasks. In Proc. of the Conference on Empirical Methods in Natural Language Processing, 2008, pp. 254–263.
[41]. Long Tran-Thanh, Sebastian Stein, Alex Rogers, and Nicholas R Jennings. Efficient crowdsourcing of unknown experts using bounded multi-armed bandits. Artificial Intelligence, vol. 214, issue 1, 2014, pp. 89–111.
[42]. Fabian L Wauthier and Michael I Jordan. Bayesian bias mitigation for crowdsourcing. In In Proc. of the Neural Information Processing Systems 2011 Conference (Advances in neural information processing systems 24), 2011, pp. 1800–1808.
[43]. Peter Welinder and Pietro Perona. Online crowdsourcing: rating annotators and obtaining cost-effective labels. In Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Workshops, 2010, pp. 25–32.
[44]. Jacob Whitehill, Ting-fan Wu, Jacob Bergsma, Javier R Movellan, and Paul L Ruvolo. Whose vote should count more: Optimal integration of labels from labelers of unknown expertise. In Proc. of the Neural Information Processing Systems 2009 Conference (Advances in neural information processing systems 22), 2009, pp. 2035–2043.
[45]. Yan Yan, Romer Rosales, Glenn Fung, and Jennifer G Dy. Active learning from crowds. In Proc. of the 28th International Conference on International Conference on Machine Learning, 2011, pp. 1161–1168.
[46]. Chicheng Zhang and Kamalika Chaudhuri. Active learning from weak and strong labelers. In Proc. of the Neural Information Processing Systems 2015 Conference (Advances in neural information processing systems 28), 2015, pp. 703–711.
[47]. Jing Zhang, Victor S Sheng, Jian Wu, and Xindong Wu. Multi-class ground truth inference in crowdsourcing with clustering. IEEE Transactions on Knowledge and Data Engineering, vol. 28, № 4, 2016, pp. 1080–1085.
[48]. Jing Zhang, Xindong Wu, and Victor S Sheng. Learning from crowdsourced labeled data: a survey. Artificial Intelligence Review, vol. 46, № 4, 2016, pp. 543–576.

[49]. Yuchen Zhang, Xi Chen, Denny Zhou, and Michael I Jordan. Spectral methods meet em: A provably optimal algorithm for crowdsourcing. In Proc. of the Neural Information Processing Systems 2014 Conference (Advances in neural information processing systems 27), 2014, pp. 1260–1268.
[50]. Yudian Zheng, Reynold Cheng, Silviu Maniu, and Luyi Mo. On optimality of jury selection in crowdsourcing. In Proceedings of the 18th International Conference on Extending Database Technology, 2015, pp. 193-204.
[51]. Yudian Zheng, Guoliang Li, and Reynold Cheng. Docs: a domain-aware crowdsourcing system using knowledge bases. Proc. of the VLDB Endowment, vol. 10, № 4, 2016, pp. 361– 372.
[52]. Yudian Zheng, Guoliang Li, Yuanbing Li, Caihua Shan, and Reynold Cheng. Truth inference in crowdsourcing: is the problem solved? Proc. of the VLDB Endowment, vol. 10, № 5, 2017, pp. 541–552.
[53]. Yudian Zheng, Jiannan Wang, Guoliang Li, Reynold Cheng, and Jianhua Feng. Qasca: A quality-aware task assignment system for crowdsourcing applications. In Proceedings of the ACM SIGMOD International Conference on Management of Data, 2015, pp. 1031–1046.
[54]. Jinhong Zhong, Ke Tang, and Zhi-Hua Zhou. Active learning from crowds with unsure option. In Proc. of the 24th International Conference on Artificial Intelligence, 2015, pp. 1061–1068.
[55]. A.V. Ponomarev. Quality Control Methods in Crowd Computing: Literature Review. SPIIRAS Proceeding, issue 54, 2017, pp. 152–184 (in Russian). DOI: 10.15622/sp.54.7.
[56]. A. Korshunov, A. Gomzin. Topic modeling in natural language texts. Trudy ISP RAN/Proc. ISP RAS, vol. 23, 2012, pp. 215-244 (in Russian). DOI: 10.15514/ISPRAS-2012-23-13.