ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГАОУ ВО НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Образовательная программа «Прикладная математика и информатика»

УДК 004.8

Отчет об исследовательском проекте на тему:

Кластеризация аудио

(промежуточный, этап 1)

Выполнил студент:

группы #БПМИ213, 3 курса Бонич Дмитрий Сергеевич

Принял руководитель проекта:

Сендерович Александра Леонидовна
Научный сотрудник
Факультет компьютерных наук НИУ ВШЭ

Москва 2024

Содержание

Аннотация

Существующие методы глубинного обучения для кластеризации изображений работают довольно хорошо. Однако вопрос качественной кластеризации аудио остается открытым. В этой работе мы планируем адаптировать лучшие методы кластеризации изображений для задачи кластеризации аудио.

Ключевые слова

Глубинное обучение, обучение без учителя, кластеризация

1 Введение

1.1 Постановка задачи

В задаче классификации мы имеем обучающую выборку, где для каждого объекта известен его класс и от нас требуется моделировать распределение классов на пространстве объектов. Задача кластеризации более сложная – необходимо разбить объекты на осмысленные группы не зная ни самих групп, ни их распределения.

1.2 Метрики качества

Чтобы замерить качество кластеризации как правило метод решает задачу на размеченном для кластеризации датасете, разумеется не используя метки классов. Полученные после кластеризации номера кластеров будем называть *псевдометками*. Псевдометки сравниваются с настоящими метками и качество определяется как некоторый вид корреляции между ними.

Одной из самых популярных метрик качества является NMI(Normalized Mutual Information) Пусть у нас есть пара случайных величин X и Y, тогда:

$$\mathrm{NMI}(X,Y) = \frac{\mathrm{KL}(p_{(X,Y)}|p_X \otimes p_Y)}{\sqrt{H(X)H(Y)}}$$

При подсчете NMI мы считаем распределения меток и псевдометок за X и Y и вычисляем *оценку максимального правдоподобия*[1]. NMI принимает значения от 0 до 1.

Еще одной используемой метрикой является ARI(Adjusted Rand Index). Это скорректированная версия метрики RI[2] (Rand Index). Определяется ARI следующим образом:

$$\mathrm{ARI}(X,Y) = \frac{\mathrm{RI}(X,Y) - \mathrm{E}[\mathrm{RI}(X,Y)]}{1 - \mathrm{E}[\mathrm{RI}(X,Y)]}$$

ARI принимает значения от 0 до 1.

Также в качестве метрики качества можно использовать долю правильных ответов как и в задаче классификации. Однако предварительно необходимо решить *задачу о назначениях*[3]. между псевдометками и метками. Мы переименуем псевдометки так, чтобы достичь максимального качества. Затем на переименованных псевдометках мы посчитаем долю правильных ответов, которую и используем в качестве метрики качества.

[1] далее ОМП
[2] https://en.wikipedia.org/wiki/Rand_index
[3] https://en.wikipedia.org/wiki/Assignment_problem

1.3 Классические методы решения

С точки зрения классических методов кластеризуемые объекты это точки в многомерном пространстве. Такие методы обычно имеют итерационную природу и решают задачу выделения кластеров точек находя их скопления, области с повышенной плотностью, некоторые структуры. Посмотрим на несколько примеров.

1.3.1 K-means

Одним из самых простых и известных методов является K-means. Он работает на базе EM-алгоритма[4]. K-means итерационно пытается найти два неизвестных набора переменных — номера кластеров для объектов и центры этих кластеров. Количество кластеров фиксировано и задается до начала работы алгоритма в качестве гиперпараметра k.

В начале своей работы классический K-means инициализирует все центры кластеров случайно. Затем чередуются E и M шаги до сходимости. На E-шаге мы назначаем каждому объекту номер кластера к центру которого объект ближе всего в качестве псевдометки. На M-шаге мы вычисляем ОМП для каждого центра кластера, то есть берем в качестве центра кластера усредненную точку из всех объектов с псевдометкой этого кластера.

1.3.2 DBSCAN

1.3.3 MeanShift

1.3.4 AgglomerativeClustering

2 Обзор литературы

При кластеризации сложных объектов таких как изображения или аудио недостаточно вытянуть данные объекта в вектор и применить классический алгоритм кластеризации для полученных точек. Чтобы алгоритмы кластеризации хорошо работали входное пространство точек должно обладать некоторыми свойствами. В идеале близкие в этом пространстве точки должны принадлежать к одной группе.

Таким образом задачу кластеризации сложных объектов можно разделить на две части – получение представлений объектов образующих пригодное для кластеризации пространство точек и сама кластеризация этих представлений. Метод переводящий объекты в признаки будем называть *энкодером*.

[4] https://en.wikipedia.org/wiki/Expectation%E2%80%93maximization_algorithm

2.1 DeepCluster

Одним из первых методов использующих глубинное обучения для кластеризации изображений был DeepCluster [1]. DeepCluster использует сверточную нейросеть f_θ в качестве энкодера. К полученным признакам применяется K-means и мы получаем псевдометку y_n для каждого изображения x_n. Затем к сверточной сети прикрепляется классификационная голова g_W. g_W предсказывает вероятности кластеров для объекта по его признакам полученным из f_θ. Псевдометки используются как настоящие для подсчета логистической функции потерь. Таким образом мы решаем следующую задачу оптимизации:

$$\min_{\theta,W} \frac{1}{N} \sum_{n=1}^{N} l(g_W(f_\theta(x_n)), y_n)$$

Здесь N – это размер батча, l логистическая мультиномиальная функция потерь

По усредненному значению функции потерь для батча делается проход назад и веса θ и W обновляются стохастическим градиентным спуском. Затем шаги кластеризации и оптимизации параметров сети повторяются заданное количество эпох.

Однако в такой постановке метод выраждается в тривиальные решения. Их существует 2 типа - пустые кластеры и тривиальная параметризация

Чтобы не допустить возникновения пустых кластеров сделаем следующую модификацию K-means. Пусть на некоторой итерации у нас появился пустой класс A. Тогда возьмем случайный непустой класс B. Добавим к центру B шум получив новый центр для кластера A. Переназначим классы точек из B так чтобы каждая точка имела класс ближайшего центра кластера.

Тривиальная параметризация возникает когда распределение классов становится сильно неравномерным. Тогда классификационной голове становится выгодно выдавать только несколько наиболее часто встречающихся классов игнорируя остальные. Чтобы этого избежать необходимо сэмплировать объекты для батча из равномерного распределения по псевдометкам с предыдущего шага.

2.2 SPICE

SPICE предложенный в [8] является более продвинутым методом нежели DeepCluster и имеет лучшее качество. Целью SPICE является обучение энкодера и кластеризационной головы. В качестве энкодера используется сверточная нейросеть. Кластеризационная голова это двухслойный MLP(Multilayer perceptron). Она принимает на вход признаки из энкодера

и выдает вероятности кластеров.

Метод состоит из 3-ех этапов. На первом этапе обучается энкодер. На втором этапе энкодер фиксируется и обучается только кластеризационная голова. На третьем этапе энкодер и кластеризационная голова обучаются вместе.

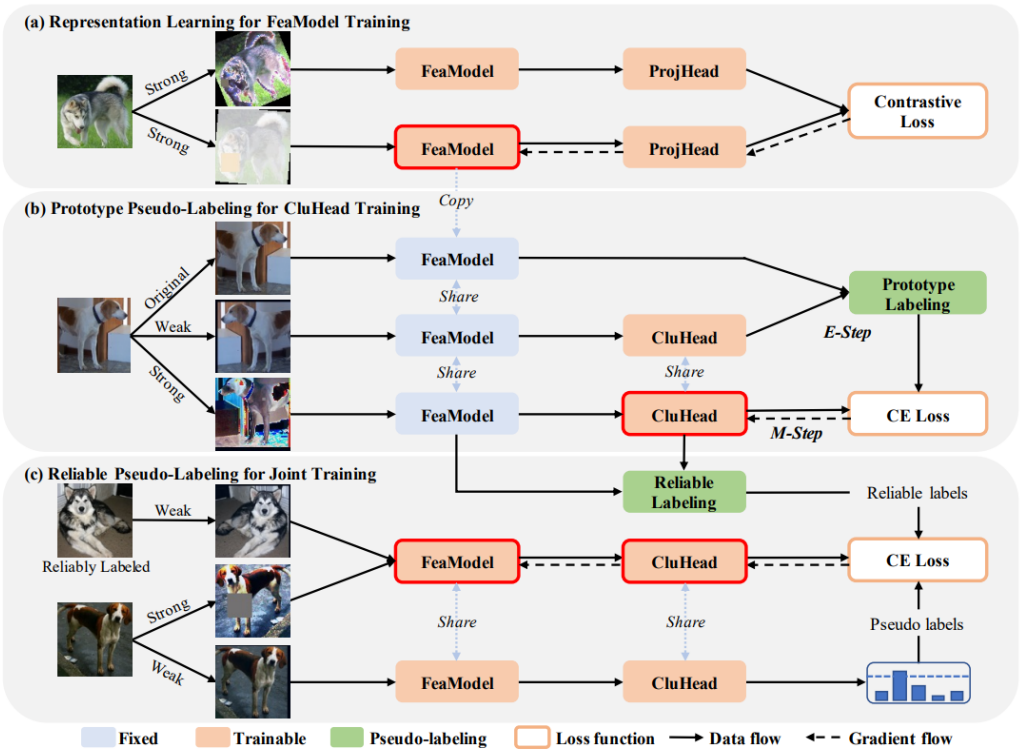

Рис. 2.1: Этапы обучения метода SPICE. FeaModel - энкодер, ProjHead - проекционная голова, CluHead - кластеризационная голова. a) Обучение энкодера. b) Обучение кластеризационной головы. c) Совместное обучение энкодера и кластеризационной головы.

Схематический принцип работы каждого этапа можно видеть на Рисунке 2.1. Перейдем к разбору каждого этапа по отдельности.

2.2.1 Обучение энкодера

Помимо энкодера будем учить проекционною голову, которая представляет из себя двухслойный MLP. Как показано на Рисунке 2.1(a) мы копируем энкодер и кластризационную голову, чтобы получить две ветки для обучения.

На вход веткам подаются разные аугментации одного и того же изображения. Аугментированное изображение проходит через энкодер соответствующей ветки и полученные признаки подаются на вход проекционной голове из той же ветки. В итоге из верхней и нижней веток мы получаем вектора z^+ и z соответственно. После этого мы считаем следующую функцию потерь:

$$\mathcal{L}_{fea} = -\ln\left(\frac{\exp(z^T z^+/\tau)}{\sum_{i=1}^{N_q}\exp(z^T z_i^-/\tau) + \exp(z^T z^+/\tau)}\right)$$

Здесь за z_i^- обозначены результаты верхней ветки для любых изображений отличных от текущего. Мы поддерживаем очередь из последних N_q таких негативных примеров. $\tau > 0$ – гиперпараметр температуры.

С помощью посчитанной функции потерь $\mathcal{L}_{fea}$ мы делаем проход назад по нижней ветке и делаем шаг градиентного спуска. Верхняя ветка обновляется как экспоненциально движущееся среднее[5] нижней.

Замечание. На самом деле вместо описанного алгоритма можно использовать любой метод обучения представлений для изображений без учителя.

2.2.2 Обучение кластеризационной головы

Энкодер на этом этапе фиксирован и не обучается. На данном этапе у нас есть два набора неизвестных переменных: оптимальные параметры кластеризационной головы и правильные назначения классов объектам. Поэтому для их нахождения мы можем воспользоваться EM-алгоритмом. На E-шаге мы будем считать параметры кластеризационной головы известными и будем искать назначения кластеров объектам. На M-шаге мы будем считать известными назначения кластеров и будем оптимизировать параметры кластеризационной головы. Рассмотрим подробно эти 2 шага.

На E-шаге мы сначала вычисляем признаки f_i из верхней ветки Рисунок 2.1(b). С помощью средней ветки мы получаем вероятности каждого кластера для каждого объекта из батча. Для каждого кластера мы выбираем топ $\frac{M}{K}$ объектов с максимальными вероятностями полученными из второй ветки и берем их признаки f_i как показано в 1.

$$\mathfrak{F}_k = \left\{ f_i \middle| i \in \text{argtopk}\left(P_{:,k}, \frac{M}{K}\right), \forall i = 1, \ldots, M \right\} \tag{1}$$

Здесь M это размер батча, а K количество кластеров задаваемое как гиперпараметр до начала работы алгоритма. P это матрица вероятностей где по строкам разложены вероятности кластеров для каждого объекта из батча.

Далее для каждого класса k мы вычисляем его центр γ_k, как среднее по точкам из $\mathfrak{F}_k$. Затем мы вычисляем косинусную похожесть между признаками f_i и центрами кластеров γ_k. За $\mathfrak{X}^k$ обозначим $\frac{M}{K}$ ближайших в данной метрике объектов к центру кластера γ_k. Скажем что

[5] https://en.wikipedia.org/wiki/Moving_average#Exponential_moving_average

все объекты в $\mathfrak{X}^k$ имеют одну и туже псевдометку, а именно $y_i^s = k \quad \forall x_i \in \mathfrak{X}^k$. Результатом E-шага будет следующее множество пар объект-псевдометка:

$$\mathfrak{X}^s = \{(x_i, y_i^s) \mid \forall x_i \in \mathfrak{X}^k, k = 1, \ldots, K\}$$

Будем называть псевдометки из $\mathfrak{X}^s$ протопсевдометками.

Перейдем к M-шагу. Используя протопсевдометки с E-шага и вероятности классов полученные из нижней ветки мы можем посчитать следующую функцию потерь:

$$\mathcal{L}_{clu} = \frac{1}{M} \sum_{i=1}^{M} L_{ce}(y_i^s, p_i')$$

Здесь L_{ce} это логистическая функция потерь, $p_i' = \text{softmax}(p_i)$. p_i это вероятности полученные из нижней ветки. Стоит заметить, что в итоге к логитам из нижней ветки два раза применяется операция softmax. Авторы обосновывают такое решение тем, что двойной softmax замедляет процесс обучения. Таким образом такая функция потерь меньше доверяет протопсевдометкам, что особенно полезно в начале обучения.

Функция потерь $\mathcal{L}_{clu}$ используется для прохода назад через кластеризационную голову. Затем делается шаг градиентного спуска.

Можно заметить что 2-ой этап довольно легковесный т. к. нам необходимо обучать только кластеризационную голову с малым количеством параметров. Поэтому авторы предлагают параллельно учить сразу несколько голов и выбирать лучшую по функции потерь $\mathcal{L}_{clu}$ для уменьшения дисперсии решения.

Стоит также обратить внимание, что распределения аугментаций для разных веток отличаются. Как видно на рисунке 2.1(b) верхней ветке на вход подается оригинал изображения, средней ветке слабо аугментированое изображение, а нижней сильно аугментированное.

2.2.3 Совместное обучение энкодера и кластеризационной головы

На данном этапе мы хотим дообучить энкодер и кластеризационную голову. С E-шага прошлого этапа мы можем получить набор $\{x_i, f_i, y_i^s\}_{i=1}^N$, где N это размер выборки. Для каждого объекта x_i выберем N_s ближайших соседей по косинусной похожести, обозначим метки объектов каждого такого множества за $\mathfrak{L}_i$. Введем величину r_i:

$$r_i = \frac{1}{N_s} \sum_{y \in \mathfrak{L}_i} I(y = y_i^s)$$

Введем гиперпараметр порога λ. Если $r_i > \lambda$, то будем считать такой объект надежно размеченным, иначе нет. Таким образом подмножество надежно размеченных объектов $\mathfrak{X}^r$ определяется следующим образом:

$$\mathfrak{X}^r = \{(x_i, y_i^s) \mid r_i > \lambda, \forall i = 1, \ldots, N\}$$

Во время обучения полученные надежные псевдометки и фиксируются.

Затем каждую эпоху мы вычисляем консистентные псевдометки:

$$y_j^u = \begin{cases} \text{argmax}(p_j) & \text{если } max(p_j) \geq \eta, \\ -1 & \text{иначе} \end{cases}$$

Здесь p_j вероятности кластеров для объекта j полученные после прогонки слабой аугментации объекта через энкодер и кластеризационную голову Рисунок 2.1(c). η гиперпараметр порог. Теперь мы можем посчитать следующий функционал ошибки:

$$\mathcal{L}_{joint} = \frac{1}{L}\sum_{i=1}^{L} \mathcal{L}_{ce}(y_i^s, \mathcal{C}(\mathcal{F}(\alpha(x_i)))) + \frac{1}{U}\sum_{i=1}^{U} \mathbb{I}(y_j^u \geq 0)\mathcal{L}_{ce}(y_j^u, \mathcal{C}(\mathcal{F}(\beta(x_j))))$$

Тут первое слагаемое отвечает за надежно размеченные сэмплы и их количество равняется L. Второе слагаемое отвечает за сэмплы с консистентыми псевдометками и их количество равняется U. $\mathcal{F}$ и $\mathcal{C}$ это энкодер и кластеризационная голова соответственно. α и β это слабая и сильная аугментации соответственно.

С помощью $\mathcal{L}_{joint}$ делается проход назад и шаг градиентного спуска по параметрам энкодера и кластеризционной головы.

Замечание. После получения надежных псевдометок можно применить любой метод частичного обучения[6].

[6] https://en.wikipedia.org/wiki/Weak_supervision#Semi-supervised_learning

3 План дальнейшей работы

4 Примеры

4.1 Ссылки на статьи

Ссылки на статьи оформляются с помощью пакета `biblatex`, например [2]. В описании статье в bib файле нужно обязательно указывать место публикации работы (журнал или конференцию) и год. Обратите внимание, что для описания статей из разных источников в списке литературы используются разные команды в bib файле: статья из журнала [4], статья с конференции [2], книга [7], глава книги [6]. Если статься еще не опубликована нигде, а только выложена на arXiv, то на нее тоже можно сослаться [3], но предпочтительно ссылаться на опубликованную версию, если она уже существует. Если вы хотите сослаться на сайт, то можно либо так же внести его в список литературы [5] (рекомендуется, если таких ссылок у вас много из-за особенностей темы вашего проекта), либо использовать ссылку внизу страницы[7]. При работе с онлайн ресурсами не забывайте указывать дату обращения к этому ресурсу, так как в отличие от опубликованных статей, эти ресурсы могут измениться в любой момент.

4.2 Рисунки

Все рисунки в тексте должны иметь подписи и вы на них должны ссылаться в тексте. Например, на Рисунке 4.1 изображен пример графика. Не забывайте подписывать все оси на графиках, добавлять легенду и пояснять все обозначения, а также используйте адекватного размера шрифты и толщину линий на графиках (все должно быть видно и понятно без многократного увеличения). На рисунке из примера явно не хватает обозначения синей линии в легенде.

4.3 Таблицы

Все таблицы в тексте тоже должны иметь подписи и вы на них должны ссылаться в тексте. Например, в Таблице 4.1 показаны результаты примерного эксперимента.

[7]Книги доступны по ссылке: `http://www-cs-faculty.stanford.edu/~uno/abcde.html`, дата обр. 16.05.2013

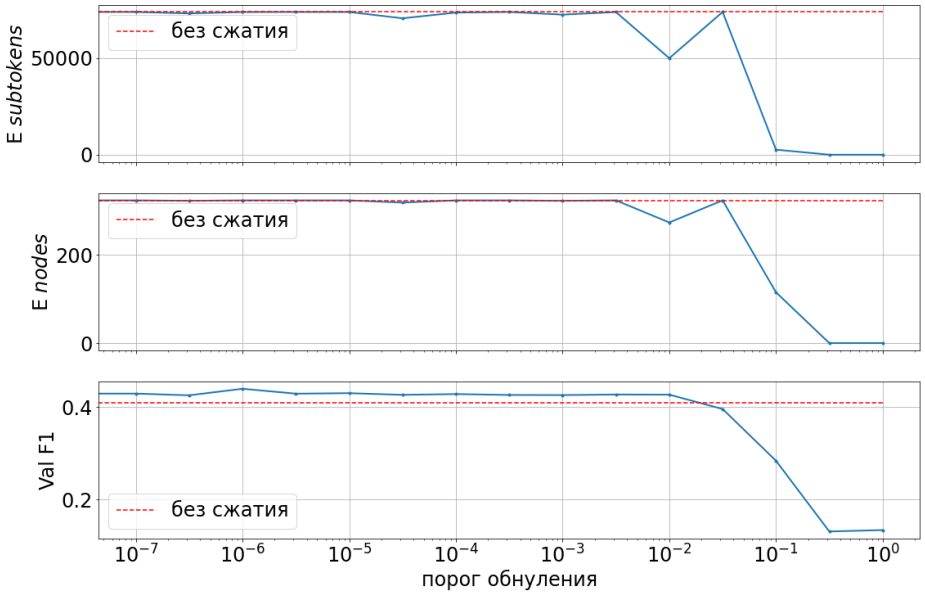

Рис. 4.1: Пример графика. Тут должна быть подпись, поясняющая что происходит на рисунке (краткая, но достаточная для понимания основной идеи графика).

Таблица 4.1: Пример таблички. Тут должна быть подпись, поясняющая что происходит в таблице (краткая, но по делу).

	Val			Test				
	Prec	Rec	F1	Prec	Rec	F1	nodes	subtokens
запуск 1	0.4894	0.3775	0.4263	0.4824	0.3683	0.4177	10029	179
запуск 2	0.4887	0.3739	0.4237	0.4891	0.3724	0.4228	10039	177
запуск 3	0.4820	0.3751	0.4219	0.4838	0.3677	0.4178	10037	180
среднее	**0.4867**	**0.3755**	**0.4239**	**0.4851**	**0.3695**	**0.4195**		
дисперсия	0.0041	0.0019	0.0022	0.0036	0.0025	0.0029		

4.4 Формулы

Формулы стоит центрировать, а также нумеровать, если вы ссылаете на них в тексте. Также не забывайте пояснять все обозначения в формулах. Например, запишем следующую задачу оптимизации:

$$\theta* = \min_{\theta} F(\theta), \tag{2}$$

где F – квадратичная функция от параметра θ. При необходимости, далее в тексте можно сослаться на формулу (2). При этом, в зависимости от конкретных формул, можно использовать разные слова: формула, уравнение, задача оптимизации и т.п.

Список литературы

[1] Mathilde Caron, Piotr Bojanowski, Armand Joulin и Matthijs Douze. “Deep Clustering for Unsupervised Learning of Visual Features”. в: *Proceedings of the European Conference on Computer Vision (ECCV)*. сент. 2018.

[2] Nadezhda Chirkova, Ekaterina Lobacheva и Dmitry Vetrov. “Bayesian Compression for Natural Language Processing”. в: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2018.

[3] Nadezhda Chirkova, Ekaterina Lobacheva и Dmitry Vetrov. “Bayesian Compression for Natural Language Processing”. в: *arXiv preprint, arXiv:1810.10927, version 2* (2018).

[4] George D. Greenwade. “The Comprehensive Tex Archive Network (CTAN)”. в: *TUGBoat* 14.3 (1993), с. 342—351.

[5] Donald Knuth. *Knuth: Computers and Typesetting*. URL: http://www-cs-faculty.stanford.edu/~uno/abcde.html (дата обр. 16.05.2013).

[6] Donald E. Knuth. “Fundamental Algorithms”. в: Addison-Wesley, 1973. гл. 1.2.

[7] Donald E. Knuth. *The Art of Computer Programming*. Four volumes. Seven volumes planned. Addison-Wesley, 1968.

[8] Chuang Niu и Ge Wang. *SPICE: Semantic Pseudo-labeling for Image Clustering*. 2021. arXiv: 2103.09382 [cs.CV].