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. Приведем пример такого метода.

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.

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*.

2.1 DeepCluster

Одним из первых методов использующих глубинное обучения для кластеризации изображений был DeepCluster [2]. DeepCluster использует сверточную нейросеть f_θ в качестве

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

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 [7] 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.

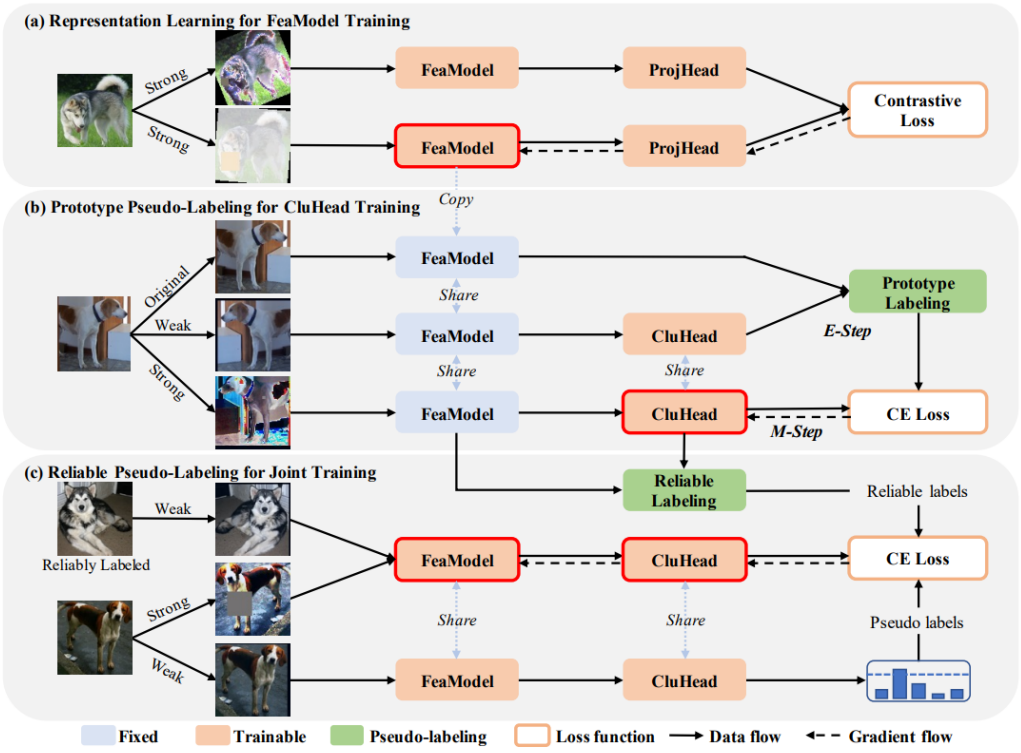

Рис. 2.1: Этапы обучения метода SPICE. FeaModel - энкодер, ProjHead - проекционная голова, CluHead - кластеризационная голова. a) Обучение энкодера. b) Обучение кластеризационной головы. c) Совместное обучение энкодера и кластеризационной головы.

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. Далее считается лосс максимизирующий косинусную похожесть результатов веток. Также лосс минимизирует косинусную похожесть результата верхней ветки для текущего сэмпла и негативных примеров. Градиентный спуск оптимизирует только нижнюю ветку, а верхняя обновляется как экспоненциально движущееся среднее 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 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. Для этого мы говорим, что у сэмпла x_i псевдометка y_i^s надежная, если среди N_s ближайших по косинусной похожести соседей к x_i доля объектов имеющих такую же псевдометку больше порога λ. N_s и λ это гиперпараметры.

В дальнейшем надежные метки фиксируются и не меняются все время обучения. В некотором смысле их можно считать настоящими метками. Поэтому для обучения энкодера и кластеризационной головы можно использовать любой алгоритм частичного обучения. Авторы SPICE использовали [8].

2.3 Обучение представлений

Как уже было сказано раннее, первый шаг для кластеризации сложных объектов это извлечение хороших признаков из объекта. Существует множество методов которые нацелены именно на эту задачу.

Один из примеров это фреймворк BYOL [4]. Он учит представления для изображений без учителя в предположении что признаки для двух аугментаций изображения должны быть в некотором смысле похожи. А именно проекция признаков для одной аугментации должна иметь как можно большую косинусную похожесть с проекцией признаков другой аугментации к которой применили MLP.

2.4 Случай аудио

Рассмотренные раннее методы занимаются кластеризацией изображений. Однако их можно адаптировать для кластеризации аудио. Для этого каждую аудиозапись можно перевести в изображение отражающее его структуру – спектрограмму[5]. Так например DeepCluster был адаптирован для аудио как DECAR [3].

Аналогичным образом BYOL был адаптирован для аудио как BYOL-A [6]. Важным отличием от классического BYOL является наличие специфичных для аудио аугментаций. Например используется аугментация Random Linear Fader, которая симулирует эффект приближения/отдаления источника звука.

Еще один интересный пример получения представлений для аудио это wav2vec 2.0 [1]. Он работает с аудио как с последовательностью и кодирует её с помощью трансформера. Чтобы получить представление для всего аудио в целом можно к примеру усреднить полученную последовательность представлений.

Существуют также методы обучения с учителем представлений для аудио. Примером может служить PaSST [5]. Он разбивает спектрограмму на патчи, которые передаются в трансформер. Чтобы ускорить обучение в трансформер подаются не все патчи, а только их часть. Как оказывается такой подход не только ускоряет обучение, но и является техникой регуляризации улучшающей качество.

3 План дальнейшей работы

В дальнейшем планируется выбрать размеченный аудио датасет и построить бейзлайн для кластеризации. Он будет устроен следующим образом. Некоторому энкодеру на вход будет подаваться спектрограмма. Полученные признаки будут уменьшены некоторым методом уменьшения размерности, например PCA[6]. Затем малоразмерные признаки будут кластеризованны некоторым классическим методом, например K-means. Планируется добавить в бейзлайн несколько различных комбинаций (энкодер, метод уменьшения размерности, метод кластеризации). В качестве основных метрик качества планируется использовать NMI и долю правильных ответов. После этого планируется адаптировать метод SPICE для аудио и возможно другие новые методы кластеризации изображений и сравнить их с бейзлайном.

[5] https://en.wikipedia.org/wiki/Spectrogram
[6] https://en.wikipedia.org/wiki/Principal_component_analysis

Список литературы

[1] Alexei Baevski, Yuhao Zhou, Abdelrahman Mohamed и Michael Auli. “wav2vec 2.0: A Framework for Self-Supervised Learning of Speech Representations”. в: *Advances in Neural Information Processing Systems*. под ред. H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan и H. Lin. т. 33. Curran Associates, Inc., 2020, с. 12449—12460. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/92d1e1eb1cd6f9fba3227870bb6d7f07-Paper.pdf.

[2] 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.

[3] Sreyan Ghosh, Ashish Seth и Sharma Umesh. “DECAR: Deep Clustering for learning general-purpose Audio Representations”. в: 2022. URL: https://api.semanticscholar.org/CorpusID:251878867.

[4] Jean-Bastien Grill, Florian Strub, Florent Altché, Corentin Tallec, Pierre Richemond, Elena Buchatskaya, Carl Doersch, Bernardo Avila Pires, Zhaohan Guo, Mohammad Gheshlaghi Azar, Bilal Piot, koray kavukcuoglu koray, Remi Munos и Michal Valko. “Bootstrap Your Own Latent - A New Approach to Self-Supervised Learning”. в: *Advances in Neural Information Processing Systems*. под ред. H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan и H. Lin. т. 33. Curran Associates, Inc., 2020, с. 21271—21284. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/f3ada80d5c4ee70142b17b8192b2958e-Paper.pdf.

[5] Khaled Koutini, Jan Schlüter, Hamid Eghbalzadeh и Gerhard Widmer. “Efficient Training of Audio Transformers with Patchout”. в: *ArXiv* abs/2110.05069 (2021). URL: https://api.semanticscholar.org/CorpusID:238583346.

[6] Daisuke Niizumi, Daiki Takeuchi, Yasunori Ohishi, Noboru Harada и Kunio Kashino. “BYOL for Audio: Exploring Pre-Trained General-Purpose Audio Representations”. в: *IEEE/ACM Transactions on Audio, Speech, and Language Processing* 31 (2023), с. 137—151. DOI: 10.1109/TASLP.2022.3221007.

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

[8] Kihyuk Sohn, David Berthelot, Nicholas Carlini, Zizhao Zhang, Han Zhang, Colin A Raffel, Ekin Dogus Cubuk, Alexey Kurakin и Chun-Liang Li. “FixMatch: Simplifying Semi-Supervised Learning with Consistency and Confidence”. в: *Advances in Neural Information Processing Systems*. под ред. H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan и H. Lin. т. 33. Curran

Associates, Inc., 2020, c. 596—608. URL: https://proceedings.neurips.cc/paper_files/paper/2020/file/06964dce9addb1c5cb5d6e3d9838f733-Paper.pdf.