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

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 адаптировали лучшие методы кластеризации изображений для задачи кластеризации аудио. Также мы построили бейзлайн для сравнения классических методов и глубинного обучения в задаче класстеризации 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 определенного метода, как правило, его применяют к размеченному датасету. 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:

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

Где KL это дивергенция Кульбака-Лейблера, H – энтропия. При подсчете NMI мы считаем распределения меток и псевдометок за X и Y, и вычисляем *оценку максимальноого правдоподобия* (далее ОМП) для энтропии и KL-дивергенции. NMI 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* [18] 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.

1.3 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 [5]. 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 обучение для кластеризации изображений, был DeepCluster [3]. 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 [16], является более продвинутым методом по сравнению с 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.

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

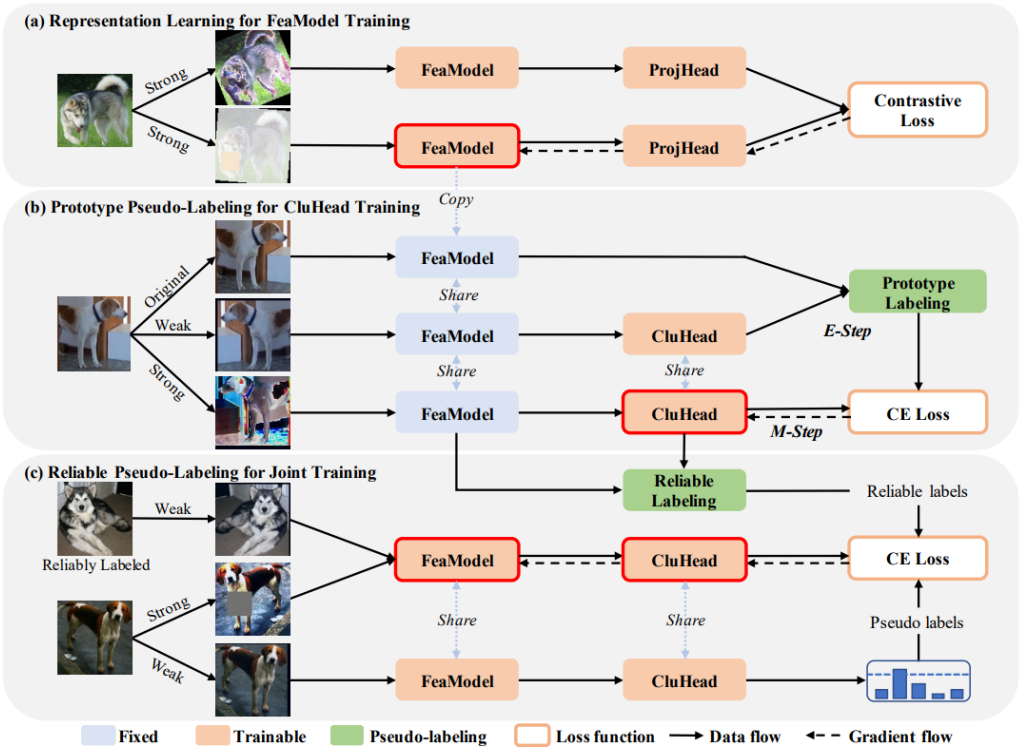


Рис. 2.1: Изображение взято из [16]. Этапы обучения метода SPICE. FeaModel - энкодер, ProjHead - проекционная голова, CluHead - кластеризационная голова. a) Обучение энкодера. b) Обучение кластеризационной головы. c) Совместное обучение энкодера и кластеризационной головы.

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. В качестве негативных примеров можно взять любые сэмплы отличные от текущего. 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 считаем параметры кластеризационной головы известными и ищем назначения кластеров объектам. На M-шаге мы считаем известными назначения кластеров и оптимизируем параметры кластеризационной головы. Рассмотрим по-

дробно 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} – логистическая функция потерь, $p_i' = \text{softmax}(p_i)$. p_i – это вероятности полученные из нижней ветки. Стоит отметить, что в итоге к логитам из нижней ветки дважды применяется операция softmax. Авторы объясняют 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, так как 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 – 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 метод FixMatch [19].

2.3 TEMI

Метод кластеризации изображений TEMI [1] использует в качестве энкодера модель, предобученную без учителя, для получения представлений изображений. Сама статья описывает способ обучения ансамбля из кластеризационных голов.

2.3.1 PMI

Будем считать что изображение $x \in \mathcal{X}$ это случайная переменная из распределения с плотностью $p(x)$. Пусть $p(c)$ это вероятность того что x имеет класс $c \in \{1, \dots, C\}$. За $p(x, x')$ обозначим вероятность того, что изображения x и x' имеют один и тот же класс. Легко видеть, что $p(x, x') = \sum_{c=1}^{C} p(x|c)p(x'|c)p(c)$. Введем $q(c|x)$, который будет нашим обучаемым классификатором, сообщающим вероятность класса c для картинки x. Используя формулу Байеса, получаем $q(x|c) = q(c|x)p(x)/q(c)$. Тогда можно аппроксимировать $p(x, x')$ следующим образом:

$$q(x, x') = \sum_{c=1}^{C} q(x|c)q(x'|c)q(c).$$

Теперь определим *поточечную взаимную информацию* (PMI) как:

$$\text{pmi}(x, x') \stackrel{\text{def}}{=} \log \frac{q(x, x')}{p(x)p(x')} = \log \sum_{c=1}^{C} \frac{q(c|x)q(c|x')}{q(c)}$$

Оказывается, что наилучший классификатор $q^*(c|x)$, т. е. совпадающий с $p(c|x)$ во всех точках x с точностью до перестановки индексов кластеров, это классификатор, максимизирующий матожидание $\text{pmi}(x, x')$. Отметим, что выполнение этого свойства гарантируется только в случае вырожденного распредления $p(c|x)$, т. е. когда каждому изображению соответствует ровно 1 класс.

2.3.2 Самодистиляция

Обозначим энкодер за g и найдем для каждого изображения x его представление $g(x)$. В полученном признаковом пространстве найдем для каждого объекта x k ближайших соседей по косинусной похожести и обозначим их за S_x. Во время обучения мы будем брать случайный x из датасета и случайный x' из S_x. Предполагается, что x и x' будут в одном кластере с большой вероятностью.

Введем две кластеризационные головы: студента $h_s(\cdot)$ и учителя $h_t(\cdot)$. Каждая из голов это трехслойная полносвязная сеть. Чтобы получить $q_s(c|x)$ и $q_t(c|x')$ используем софтмакс с температурой τ на логитах $h_s(g(x))$ и $h_t(g(x'))$ соответственно.

Аппроксимируем PMI следующим образом:

$$\widetilde{\text{pmi}}(x, x') \stackrel{\text{def}}{=} \log \sum_{c=1}^{C} \frac{(q_s(c|x)q_t(c|x'))^{\beta}}{\tilde{q}_t(c)}$$

Гиперпараметр $\beta \in (0.5, 1]$ помогает избегать вырожденных решений, делая вклад крупных кластеров менее важным. $q(c)$ оценим экспоненциально движущимся средним:

$$\tilde{q}_t(c) \leftarrow m\tilde{q}_t(c) + (1 - m)\frac{1}{B}\sum_{i=1}^{B} q_t(c|x_i)$$

где B это размер батча, $m \in (0, 1)$ гиперпараметр инерции.

Чтобы получить лосс нужно взять $\widetilde{\text{pmi}}(x, x')$ со знаком минус и симметризовать его:

$$\mathcal{L}(x, x') \stackrel{\text{def}}{=} -\frac{1}{2}\left(\widetilde{\text{pmi}}(x, x') + \widetilde{\text{pmi}}(x', x)\right)$$

Параметры $h_s(\cdot)$ оптимизируются градиентным спуском, в то время как параметры

$h_t(\cdot)$ это экспоненциально движущееся среднее параметров головы студента.

2.3.3 Взвешивание пар

Так как не все пары соседей x и x' будут иметь один класс, введем вес, позволяющий уменьшить вклад ложноположительных пар в лосс:

$$w(x, x') = \sum_{c=1}^{C} q_t(c|x) q_t(c|x')$$

Также мы будем учить не одну пару голов, а сразу H пар. Причем для каждой головы для взвешивания пары будем использовать веса, полученные из всех голов. Таким образом, финальный лосс для i-ой пары голов выглядит так:

$$\mathcal{L}^i_{\text{TEMI}}(x, x') \stackrel{\text{def}}{=} \frac{1}{H} \sum_{j=1}^{H} w_j(x, x') \mathcal{L}^i(x, x')$$

2.4 Обучение представлений

Как уже было упомянуто 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 [10]. 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.

2.5 Случай аудио

Рассмотренные раннее методы занимаются кластеризацией изображений. Однако их можно адаптировать для кластеризации аудио. Для этого каждую аудиозапись можно преобразовать в изображение, отражающее ее структуру – спектрограмму. Так например DeepCluster был адаптирован для аудио как DECAR [9].

Аналогичным образом BYOL был адаптирован для аудио как BYOL-A [15]. Важным отличием от классического BYOL является наличие специфичных для аудио аугментаций. Например используется аугментация Random Linear Fader, которая моделирует эффект приближения/удаления источника звука.

Еще один интересный пример получения представлений для аудио это wav2vec 2.0 [2]. 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 [12]. 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 Бейзлайн

Для получения бейзлайна будем использовать тройки: энкодер, метод понижения размерности, классический метод кластеризации. Сначала аудиозаписи проходят через энкодер превращаясь в векторы признаков. Затем полученные эмбеддинги уменьшаются с помощью метода понижения размерности. Наконец, к сжатым эмбеддингам применяется классический алгоритм кластеризации. В качестве метрик качества используются NMI и доля правильных ответов.

Для понижения размерности используются PCA [11] и t-SNE [13]. Для кластеризации используются K-means, DBSCAN [6], MeanShift [7], Agglomerative clustering [14].

В качестве датасета было использовано подмножество датасета DCASE 2018 task 5 [4]. В нем собраны аудиозаписи активности человека проживающего в отеле на протяжении одной недели. Всего классов 9.

Результаты можно видеть в Таблице 3.1. Стоит отметить, что помимо предобученной BYOL-A мы также замеряем её качество со случайной инициализацией весов, что отражено в таблице как Random BYOL-A.

Отметим, что для подсчета бейзлайна использовалась библиотека cuML[1]. Она содержит GPU имплементации многих классических алгоритмов кластеризации.

[1] https://docs.rapids.ai/api/cuml/stable

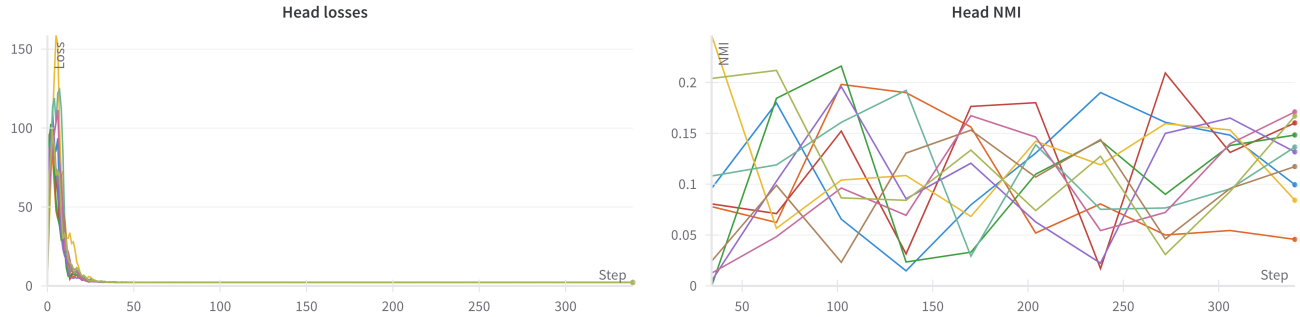
Encoder	Dimensionality reducer	Clustering method	NMI	accuracy
BYOL-A	None	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.39	0.51
		K-means	**0.59**	**0.62**
	PCA	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.54	0.53
		K-means	**0.59**	**0.62**
	t-SNE	AgglomerativeClustering	0.20	0.34
		DBSCAN	0.04	0.27
		K-means	0.50	0.48
Random BYOL-A	None	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.01	0.26
		K-means	0.56	0.58
	PCA	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.45	0.50
		K-means	0.56	0.58
	t-SNE	AgglomerativeClustering	**0.59**	**0.60**
		DBSCAN	0.20	0.28
		K-means	0.48	0.32
PaSST	None	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.29	0.44
		K-means	0.58	0.58
	PCA	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.51	0.53
		K-means	**0.61**	**0.60**
	t-SNE	AgglomerativeClustering	0.39	0.34
		DBSCAN	0.20	0.27
		K-means	**0.61**	0.56
wav2vec2	None	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.03	0.27
		K-means	0.31	**0.41**
	PCA	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.24	0.40
		K-means	0.31	**0.41**
	t-SNE	AgglomerativeClustering	0.01	0.26
		DBSCAN	0.03	0.26
		K-means	**0.32**	0.35

Таблица 3.1: Сравнение энкодеров, методов уменьшения размерности и методов кластеризации. Жирным шрифтом выделены лучшие значения метрик для каждого энкодера.

4 Эксперименты

4.1 SPICE для аудио

Применим SPICE к задаче кластеризации аудио. Первый этап SPICE нацелен на обучение энкодера, поэтому мы можем пропустить его взяв предобученный энкодер. Для этой задачи был выбран BYOL-A предобученный на Audioset [8]. В качестве аугментаций второго этапа будем использовать аугментации из BYOL-A. Будем параллельно учить 10 голов для кластеризации. Результаты обучения кластеризационной головы на втором этапе SPICE можно увидеть на Рисунке 4.1



(a) График функции потерь для каждой головы

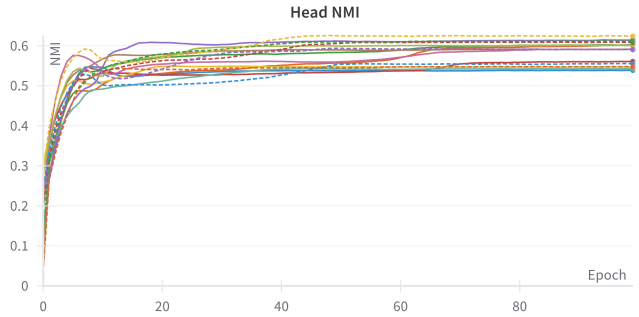
(b) NMI для каждой головы

Рис. 4.1: Результаты адаптации SPICE (этап 2) к аудио

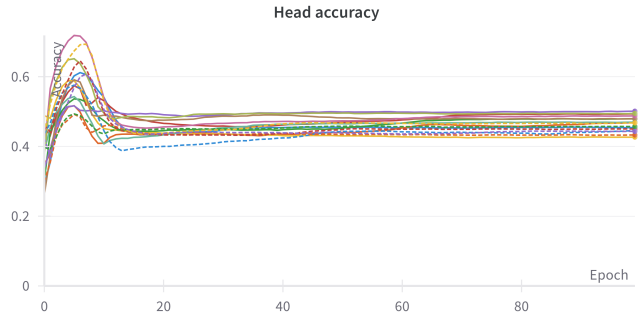
Результаты неутешительные. Решение вырождается в несколько кластеров, остальные остаются пустыми/почти пустыми. При этом из-за случайных аугментаций решение не сходится и мы наблюдаем «скачущий» NMI.

4.2 TEMI для аудио

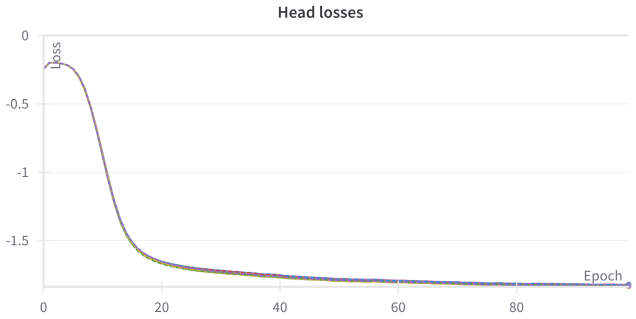
Применение TEMI для кластеризации аудио – прямолинейная задача, ведь сам метод опирается только на эмбединги объектов из обучающей выборки. Авторы TEMI получили наилучшее качество используя энкодер с архитектурой трансформера. Поступим также и используем PaSST предобученный на Audioset. Как и в оригинальной статье, обучим ансамбль из 16 голов. Результаты можно увидеть на Рисунке 4.2.



(a) NMI для каждой головы



(b) Accuracy для каждой головы



(c) График функции потерь для каждой головы

Рис. 4.2: Результаты адаптации TEMI к аудио

NMI лучшей головы по лоссу - 0.62, доля правильных ответов - 0.47. NMI получился на 1 процентный пункт лучше чем у бейзлайна. Значение доли правильных ответов сильно хуже. Такое несоответствие натолкнуло нас на мысль, что данный метод нашел осмысленные кластеры отличные от классов размеченных людьми. Подробнее эта гипотеза исследуется в следующем разделе.

5 Исследование кластеров

В данном разделе мы опишем смысл кластеров, найденных различными методами. Для этого для каждого кластера мы прослушаем 10 случайных записей из него и дадим ему наиболее подходящее описание. Также мы визуализируем объекты на плоскости с помощью t-SNE, чтобы сравнить предсказанные кластеры с настоящими метками.

Датасет DCASE 2018 task 5 размечен на 9 классов, которые представляют из себя активности человека живущего в отеле: «просмотр телевизора», «работа», «отсутствие», «социальная активность», «готовка», «мытье посуды», «уборка пылесосом», «потребление пищи» и «другое»

5.1 Кластеры в бейзлайне

Возьмем конфигурацию PaSST + PCA + K-means из бейзлайна, как лучшую, и рассмотрим кластеры в ней. Мы получили следующие кластеры:

- 0 – «Работа»
- 1 – «Просмотр телевизора»
- 2 – «Тишина»
- 3 – «Уборка пылесосом», «Готовка»
- 4 – «Потребление пищи», «Готовка», «Работа»
- 5 – «Просмотр телевизора»
- 6 – «Социальная активность»
- 7 – «Тишина»

Сравнение бейзлайн кластеризации с оригинальными метками можно увидеть на Рисунке 5.1. Для визуализации были взяты признаки из PaSST и уменьшены в размерности с помощью t-SNE.

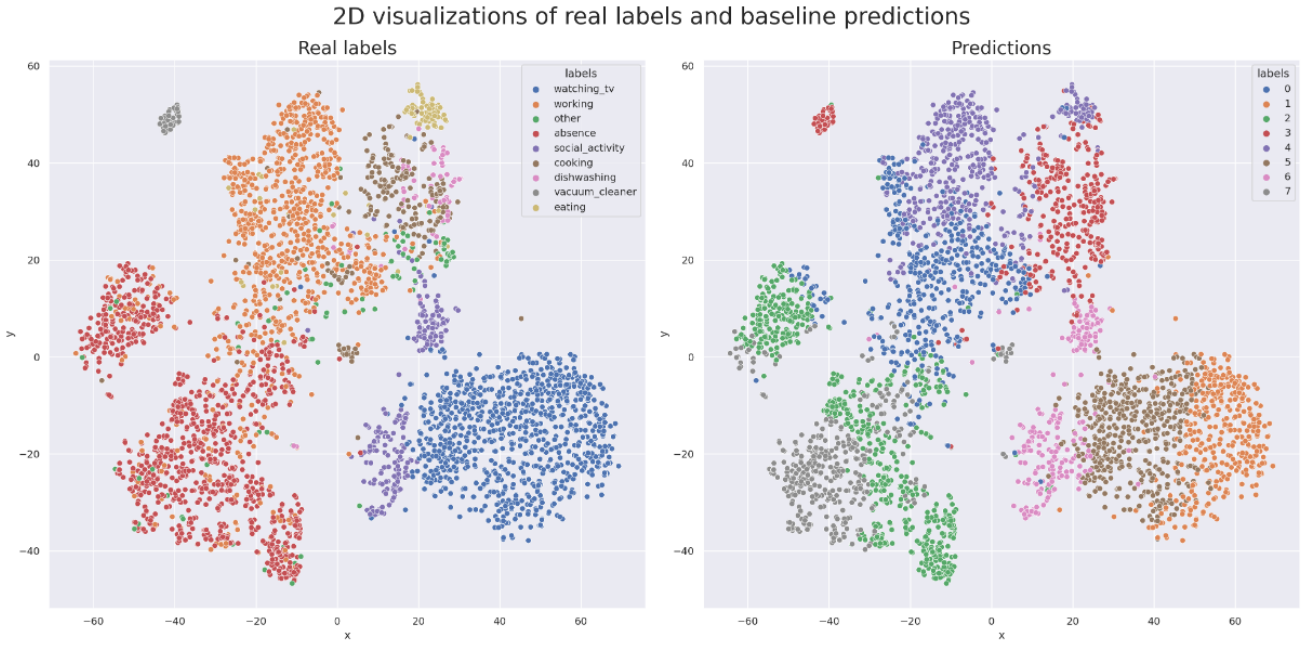


Рис. 5.1: Сравнение настоящих меток (слева) и бейзлайн кластеризации (справа)

Можно заметить, что некоторые классы или их части схлопнулись в один. Другие же разбились на несколько подклассов, но различия между этими подклассами выявить не удалось.

5.2 Кластеры в TEMI для аудио

Кластеры выделенные TEMI следующие:

- 0 – «Работа» (печать на клавиатуре)
- 1 – «Просмотр телевизора» (без музыкального сопровождения, только речь)
- 2 – «Социальная активность», «уборка пылесосом», «другое»
- 3 – «Работа» (клики, скроллинг мышкой)
- 4 – «Слабые звуки» (пение птиц, чих)
- 5 – «Просмотр телевизора» (с музыкальным сопровождением)
- 6 – «Активность на кухне» (готовка, потребление пищи, мытье посуды)
- 7 – «Тишина»
- 8 – «Тишина» (с небольшими волнениями, возможно звук от стиральной машины)

Сравнение кластеризации TEMI с оригинальными метками можно увидеть на Рисунке 5.2. Для визуализации были взяты признаки из предпоследнего слоя лучшей головы в ансамбле и уменьшены в размерности с помощью t-SNE.

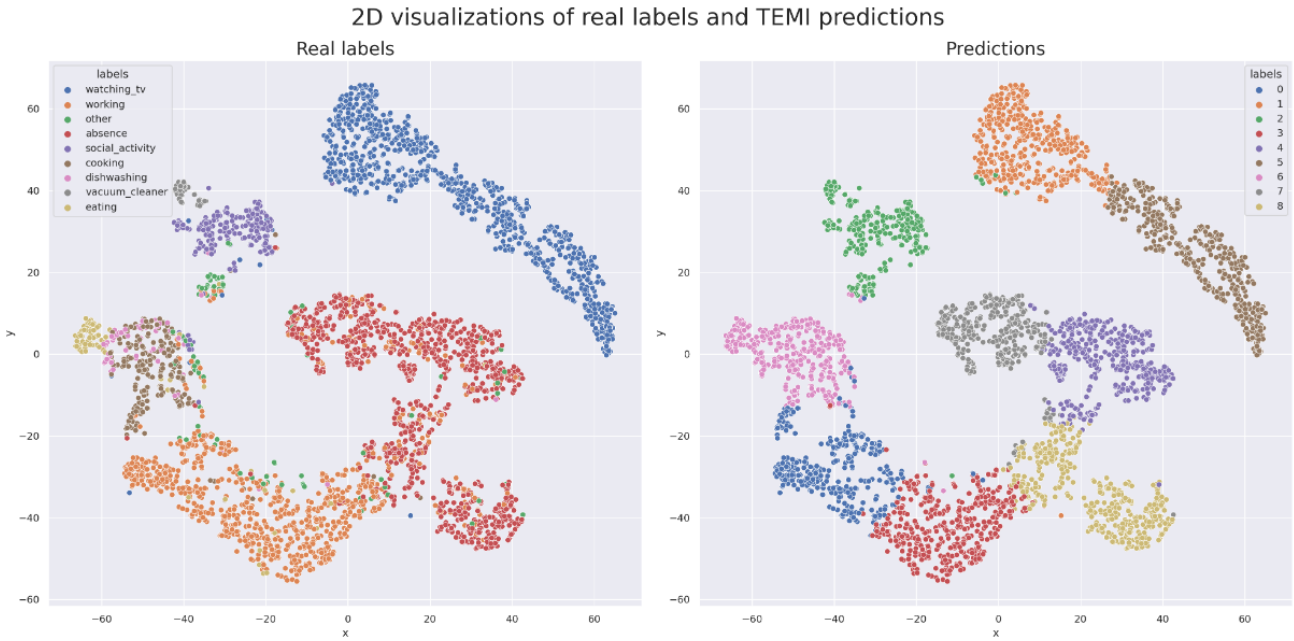


Рис. 5.2: Сравнение настоящих меток (слева) и кластеризации TEMI (справа)

В данном случае кластеризация кажется более осмысленной, чем у бейзлайна. Так например класс «Работа» разбивается на два понятных кластера – работа с мышкой и работа с клавиатурой.

6 Заключение

К сожалению нам не удалось добиться значительно лучшего качества кластеризации методами глубинного обучения. Однако, кластеры, полученные при использовании TEMI, хорошо интерпретируемы в отличии от кластеров, полученных классическими методами.

Также важно отметить, что в данной работе была опробована лишь малая часть существующих методов кластеризации изображений. Например метод SeCu [17] выглядит многообещающе. В нем для обучения учитываются только пары изображений из одного кластера, что позволяет избежать смещения обучения из-за ограниченности размера батча. Помимо этого метод использует сразу несколько техник для избежания вырожденного решения.

Список литературы

[1] Nikolas Adaloglou, Felix Michels, Hamza Kalisch и Markus Kollmann. “Exploring the Limits of Deep Image Clustering using Pretrained Models”. в: *34th British Machine Vision Conference 2023, BMVC 2023, Aberdeen, UK, November 20-24, 2023*. BMVA, 2023.

[2] 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.

[3] 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.

[4] G. Dekkers и P. Karsmakers. *DCASE 2018, Task 5: Monitoring of domestic activities based on multichannel acoustics - Development dataset*. URL: https://zenodo.org/records/1247102 (дата обр. 12.02.2024).

[5] Arthur P Dempster, Nan M Laird и Donald B Rubin. “Maximum likelihood from incomplete data via the EM algorithm”. в: *Journal of the royal statistical society: series B (methodological)* 39.1 (1977), с. 1—22.

[6] Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu и др. “A density-based algorithm for discovering clusters in large spatial databases with noise”. в: *kdd*. т. 96. 34. 1996, с. 226—231.

[7] Keinosuke Fukunaga и Larry Hostetler. “The estimation of the gradient of a density function, with applications in pattern recognition”. в: *IEEE Transactions on information theory* 21.1 (1975), с. 32—40.

[8] Jort F. Gemmeke, Daniel P. W. Ellis, Dylan Freedman, Aren Jansen, Wade Lawrence, R. Channing Moore, Manoj Plakal и Marvin Ritter. “Audio Set: An ontology and human-labeled dataset for audio events”. в: *Proc. IEEE ICASSP 2017*. New Orleans, LA, 2017.

[9] Sreyan Ghosh, Ashish Seth и Sharma Umesh. “DECAR: Deep Clustering for learning general-purpose Audio Representations”. в: 2022.

[10] 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.

[11] Cadima J. Jolliffe IT. "Principal component analysis: a review and recent developments." в: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences.* (2016).

[12] Khaled Koutini, Jan Schlüter, Hamid Eghbalzadeh и Gerhard Widmer. "Efficient Training of Audio Transformers with Patchout". в: *ArXiv* abs/2110.05069 (2021).

[13] Laurens van der Maaten и Geoffrey Hinton. "Visualizing Data using t-SNE". в: *Journal of Machine Learning Research* 9.86 (2008), с. 2579—2605.

[14] Frank Nielsen. "Hierarchical Clustering". в: февр. 2016, с. 195—211. ISBN: 978-3-319-21902-8. DOI: 10.1007/978-3-319-21903-5_8.

[15] 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.

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

[17] Qi Qian. "Stable Cluster Discrimination for Deep Clustering". в: окт. 2023, с. 16599—16608. DOI: 10.1109/ICCV51070.2023.01526.

[18] Lyle Ramshaw и Robert Endre Tarjan. "On Minimum-Cost Assignments in Unbalanced Bipartite Graphs". в: 2012.

[19] 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, с. 596—608.