Аракчеев Егор Алексеевич

студент, Факультет «Информатика и системы управления»

Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет), г. Москва

Горелова Ксения Игоревна

студент, Факультет «Информатика и системы управления»

Московский государственный технический университет имени Н.Э. Баумана

(национальный исследовательский университет), г. Москва

ПРИМЕНЕНИЕ АЛГОРИТМОВ ПОТОКОВОЙ КЛАСТЕРИЗАЦИИ ДЛЯ АНАЛИЗА НЕПРЕРЫВНО ПОСТУПАЮЩИХ ДАННЫХ

Аннотация

Цель статьи — рассмотреть особенности потоковой кластеризации и показать применение потокового варианта метода k-средних для обработки данных, поступающих мини-пакетами. В работе обобщены основные подходы к кластеризации потоков данных, включая Mini-batch k-means, CluStream, DenStream и скользящее окно. На основе экспериментальной выборки из 990 объектов выполнено сравнение пакетного KMeans, статического KMeans и MiniBatchKMeans. Показано, что пакетный алгоритм обеспечивает лучшее качество при хранении всей выборки, тогда как MiniBatchKMeans дает компромисс между точностью, скоростью и ограничением памяти. Отдельно рассмотрены ошибки, связанные с шумом, дрейфом концепции и выбором числа кластеров.

Ключевые слова: потоковые данные, кластеризация, MiniBatchKMeans, KMeans, дрейф концепции, шумовые объекты, ARI, Silhouette, анализ данных.

Annotation

The purpose of the article is to consider the features of streaming clustering and show the application of the streaming version of the k—means method for processing data received in mini-packets. The paper summarizes the main approaches to clustering data flows, including Mini-batch k-means, Clustering, Deepstream and sliding window. Based on an experimental sample of 990 objects, a comparison of batch KMeans, static KMeans and MiniBatchKMeans was performed. It is shown that the batch algorithm provides better quality when storing the entire sample, while MiniBatchKMeans provides a compromise between accuracy, speed, and memory limitations. Errors related to noise, concept drift, and the choice of the number of clusters are considered separately.

Keywords: streaming data, clustering, Mini Batch K Means, Means, concept drift, noise objects, AREA, Silhouette, data analysis.

Введение

В современных информационных системах значительная часть данных поступает не в виде заранее подготовленных таблиц, а как непрерывный поток событий. К таким источникам относятся сетевые журналы, телеметрия технических устройств, пользовательские действия в web-сервисах, события информационных систем и измерения датчиков. В этих условиях классический подход к анализу данных, при котором вся выборка заранее известна и хранится целиком, оказывается недостаточным.

Кластеризация используется для группировки объектов без заранее заданных классов. Однако традиционные алгоритмы часто требуют многократного просмотра всех данных и значительного объема памяти. В потоковой постановке данные могут быть потенциально бесконечными, а модель должна обновляться по мере поступления новых объектов. Поэтому применяются специальные алгоритмы, основанные на мини-пакетах, микрокластерах, скользящих окнах или затухающем весе старых наблюдений.

Цель статьи — показать, как потоковая кластеризация может применяться для анализа непрерывно поступающих данных, и оценить эффективность MiniBatchKMeans как простого и воспроизводимого решения для учебного эксперимента.

1. Особенности потоковой кластеризации

Потоковые данные представляют собой последовательность объектов, которые поступают во времени и обрабатываются в порядке появления. В отличие от статической выборки, поток не всегда можно полностью сохранить, повторно просмотреть или заранее очистить. Это особенно важно для систем мониторинга, где задержка анализа напрямую влияет на качество принимаемых решений.

Потоковая кластеризация должна удовлетворять нескольким требованиям. Во-первых, алгоритм должен работать инкрементно, то есть обновлять модель после получения нового объекта или небольшой группы объектов. Во-вторых, необходимо ограничивать потребление памяти, поскольку хранение всей истории потока может быть невозможно. В-третьих, алгоритм должен учитывать дрейф концепции — изменение распределения данных во времени. Например, нормальное состояние технического объекта может постепенно смещаться из-за износа, изменения нагрузки или внешних условий.

Еще одна важная проблема — наличие шума и выбросов. Отдельные наблюдения могут быть ошибочными либо не принадлежать ни к одному устойчивому кластеру. Если алгоритм не учитывает такие объекты, они искажают центры кластеров и снижают качество результата. Поэтому при выборе метода важно учитывать не только итоговую точность, но и устойчивость к выбросам, вычислительную сложность и способность адаптироваться к изменению потока.

2. Основные подходы к кластеризации потоков данных

В литературе выделяется несколько основных подходов к потоковой кластеризации. Наиболее простой вариант — модификация метода k-средних,

где центры кластеров обновляются не по всей выборке, а по очередным мини-пакетам. Такой подход хорошо масштабируется и прост в реализации, но требует заранее задать число кластеров и чувствителен к выбросам.

Другой подход основан на микрокластерах. Алгоритмы семейства CluStream хранят компактные статистические описания потока: количество объектов, суммы координат, суммы квадратов и временные характеристики. Затем на основе микрокластеров можно строить итоговые кластеры за выбранный период времени. Плотностные методы, например DenStream, развивают идеи DBSCAN и позволяют учитывать шум, но требуют настройки большего числа параметров. Скользящее окно, в свою очередь, строит модель только по последним объектам, что ускоряет адаптацию к изменениям, но может приводить к потере долгосрочных закономерностей.

Подход	**Преимущества**	**Ограничения**
Mini-batch k-means	Простота реализации, высокая скорость, малое потребление памяти	Нужно заранее задать k; чувствительность к шуму и форме кластеров
CluStream	Хранение компактной статистики потока; возможность анализа истории	Более сложная настройка и интерпретация микрокластеров
DenStream	Учет шума и кластеров произвольной формы	Больше параметров; выше вычислительная сложность
Скользящее окно	Быстрая адаптация к новым данным	Возможная потеря старых закономерностей

Для практической реализации выбран MiniBatchKMeans. Этот алгоритм поддерживает обновление модели методом partial_fit, позволяет имитировать поступление данных мини-пакетами и подходит для демонстрации базовой идеи потоковой обработки: модель не переобучается на всей выборке после каждого шага, а постепенно уточняется по новым данным.

3. Постановка эксперимента

Для проверки эффективности была подготовлена синтетическая выборка, имитирующая поток из 990 объектов. Основная часть потока содержит 900 объектов трех кластеров в двумерном пространстве признаков. Дополнительно добавлено 90 шумовых объектов, равномерно распределенных по области

наблюдения. После середины потока один из кластеров смещается, что имитирует дрейф концепции.

Использование синтетической выборки позволяет заранее знать истинные метки кластеров. Эти метки не применяются при обучении, но используются для объективной оценки результата через Adjusted Rand Index. Если бы использовались только реальные данные без разметки, оценка ограничивалась бы внутренними метриками, например Silhouette, которые не всегда отражают соответствие кластеров предметной области.

Параметр	**Значение**
Размер потока	990 объектов
Число основных кластеров	3
Доля шумовых объектов	90 объектов, около 9,1 %
Размер мини-пакета	50 объектов
Метрики качества	ARI, Silhouette, ARI после дрейфа
Среда реализации	Python, NumPy, scikit-learn

В эксперименте сравнивались три варианта: пакетный KMeans, обученный на всей выборке; статический KMeans, обученный только по первым объектам; потоковый MiniBatchKMeans, обновляемый по пакетам из 50 объектов. Пакетный KMeans использовался как ориентир качества, но не рассматривался как полноценное потоковое решение, поскольку требует хранить всю выборку.

4. Реализация на Python и метрики качества

Реализация выполнена на языке Python с использованием NumPy и scikit-learn. Для воспроизводимости применяется фиксированное значение random_state=42. Перед обучением данные масштабируются с помощью StandardScaler, так как метод k-средних основан на расстояниях и чувствителен к масштабу признаков. Основная потоковая часть строится вокруг метода partial_fit, который обновляет модель по очередному мини-пакету.

Метрика ARI показывает соответствие найденных кластеров истинным группам: значение 1 означает идеальное совпадение, значение около 0 соответствует случайному разбиению. Silhouette оценивает компактность и

разделимость кластеров: чем выше значение, тем лучше объекты отделены от соседних групп. Совместное использование этих метрик позволяет оценить как соответствие известной структуре данных, так и внутреннее качество разбиения.

5. Результаты эксперимента

Результаты сравнения показывают ожидаемое различие между пакетной, статической и потоковой обработкой. Пакетный KMeans получил наилучшие показатели, поскольку использовал всю выборку сразу. Однако такое преимущество достигается за счет нарушения потоковой постановки: алгоритм должен хранить все 990 объектов и выполнять обучение после получения полного набора данных.

Метод	ARI без шума	ARI с шумом	Silhouette	ARI после дрейфа	Память
KMeans на всей выборке	0,917	0,787	0,678	0,834	990 объектов
KMeans по первым объектам	0,787	0,675	0,667	0,624	161 объект
MiniBatchKMeans	0,818	0,702	0,669	0,667	50 объектов

Статический KMeans, обученный только на первых объектах, хуже справился с изменением распределения: ARI после дрейфа снизился до 0,624. Это демонстрирует типичную проблему статических моделей: если распределение данных меняется, старые центры перестают соответствовать новым наблюдениям.

MiniBatchKMeans показал компромиссный результат. Его ARI без учета шума составил 0,818, что ниже пакетного KMeans, но выше статического подхода при условиях ограниченной памяти. Главное преимущество состоит в том, что алгоритм хранит и обрабатывает не всю выборку, а только очередной мини-пакет и текущие центры кластеров. Это делает выбранное решение полезным для базовой потоковой обработки, когда важны скорость и экономия памяти.

k	ARI без шума	Silhouette	Комментарий
2	0,537	0,487	Недостаточно кластеров, разные группы объединяются
3	0,818	0,669	Базовый вариант соответствует структуре выборки
4	0,813	0,669	Появляется лишнее разбиение одного кластера

5	0,857	0,656	ARI выше из-за дробления, но интерпретация хуже

Дополнительная проверка показала, что выбор числа кластеров влияет не только на численные метрики, но и на смысл результата. При k=2 разные группы объединяются. При k=4 и k=5 часть кластеров дробится, поэтому некоторые метрики могут формально улучшаться, но интерпретация становится хуже. Для данной выборки наиболее обоснованным остается k=3, так как оно соответствует известной структуре подготовленного потока.

6. Ошибки и направления улучшения

Первое ограничение связано с шумовыми объектами. MiniBatchKMeans не имеет специального класса «выброс», поэтому каждый шумовой объект принудительно относится к ближайшему центру. Это снижает ARI с 0,818 без учета шума до 0,702 с учетом шума. Для повышения качества можно добавить предварительную фильтрацию выбросов или использовать плотностный алгоритм, например DenStream.

Второе ограничение связано с дрейфом концепции. После смещения одного кластера качество потокового решения снижается, потому что старые центры продолжают влиять на модель. Улучшить результат можно с помощью скользящего окна или коэффициента забывания, при котором старые наблюдения постепенно получают меньший вес. В прикладных задачах также полезно применять детектор дрейфа, который запускает переинициализацию части центров.

Третья проблема относится к выбору числа кластеров. В реальных потоках значение k часто неизвестно заранее. Если задать слишком мало кластеров, алгоритм объединит разные группы; если слишком много — разобьет одну группу на несколько частей. Для улучшения можно периодически оценивать качество по внутренним метрикам и сравнивать несколько вариантов k на контрольном окне.

Таким образом, MiniBatchKMeans эффективен как базовый и понятный алгоритм потоковой кластеризации, но для промышленного применения его

желательно дополнять обработкой выбросов, механизмом адаптации к дрейфу и периодическим контролем параметров модели.

Заключение

Потоковая кластеризация применяется в задачах, где данные поступают непрерывно и не могут эффективно обрабатываться классическими многопроходными методами. В таких условиях важны инкрементальное обновление модели, ограничение потребления памяти, устойчивость к шуму и способность учитывать изменение распределения данных во времени.

В статье рассмотрены основные подходы к кластеризации потоков данных и подробно проанализирован MiniBatchKMeans как практическое решение для обработки мини-пакетов. Эксперимент показал, что пакетный KMeans обеспечивает лучшее качество, но требует хранения всей выборки. Статическая модель хуже всего адаптируется к дрейфу. Потоковый MiniBatchKMeans уступает пакетному решению по метрикам, однако сохраняет главное преимущество потокового анализа — возможность обновления модели при ограниченной памяти.

Итоговый вывод заключается в том, что MiniBatchKMeans можно использовать как базовый алгоритм для потоковой кластеризации телеметрии, журналов событий и других непрерывных данных. При наличии шума, сложной формы кластеров и выраженного дрейфа концепции его следует дополнять фильтрацией выбросов, скользящим окном, коэффициентом забывания или заменять на плотностные потоковые методы.

СПИСОК ЛИТЕРАТУРЫ

1. Скобцов В. Ю., Новоселова Н. А. Исследование алгоритмов потоковой кластеризации при решении задачи анализа данных телеметрии малых космических аппаратов // CyberLeninka. URL: https://cyberleninka.ru/article/n/issledovanie-algoritmov-potokovoy-klasterizatsii-pri-reshenii-zadachi-analiza-dannyh-telemetrii-malyh-kosmicheskih-apparatov.

2. Печеный Е. А. Самоорганизующаяся кластеризация потока больших данных // CyberLeninka. URL: https://cyberleninka.ru/article/n/samoorganizuyuschayasya-klasterizatsiya-potoka-bolshih-dannyh.

3. Панамарева О. Н. Реализация кластеризации новостных потоков на основе векторных представлений текста // CyberLeninka. URL: https://cyberleninka.ru/article/n/realizatsiya-klasterizatsii-novostnyh-potokov-na-osnove-vektornyh-predstavleniy-teksta.

4. Панферова Е. В. Сравнительная оценка методов кластеризации в работе с большими данными // CyberLeninka. URL: https://cyberleninka.ru/article/n/sravnitelnaya-otsenka-metodov-klasterizatsii-v-rabote-s-bolshimi-dannymi.

5. Aggarwal C. C., Han J., Wang J., Yu P. S. A Framework for Clustering Evolving Data Streams // VLDB, 2003. URL: https://www.vldb.org/conf/2003/papers/S04P02.pdf.

6. Cao F., Ester M., Qian W., Zhou A. Density-Based Clustering over an Evolving Data Stream with Noise // SIAM, 2006. URL: https://www.cs.sfu.ca/~ester/papers/SDM2006.DenStream.final.pdf.

7. Zhang T., Ramakrishnan R., Livny M. BIRCH: An Efficient Data Clustering Method for Very Large Databases // SIGMOD, 1996. URL: https://www2.cs.sfu.ca/CourseCentral/459/han/papers/zhang96.pdf.

8. Sculley D. Web-scale k-means clustering // WWW, 2010. URL: https://www.ccs.neu.edu/home/vip/teach/DMcourse/2_cluster_EM_mixt/notes_slides/sculey_webscale_kmeans_approx.pdf.

9. Scikit-learn documentation. MiniBatchKMeans. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html.

list of literature

1. V. Skobtsov, Yu., Novoselova, N. A. Investigation of algorithms of streaming clustering in solving the problem of analyzing telemetry data of small spacecraft // Cyberleninka. URL: https://cyberleninka.ru/article/n/issledovanie-algoritmov-

potokovoy-klasterizatsii-pri-reshenii-zadachi-analiza-dannyh-telemetrii-malyh-kosmicheskih-apparatov.

2. Pecheny E. A. Self-regulating classification of points of contact // CyberLeninka. URL: https://cyberleninka.ru/article/n/samoorganizuyuschayasya-klasterizatsiya-potoka-bolshih-dannyh.

3. Panamareva O. N. Classification of new approaches based on the modern state approach // CyberLeninka. URL: https://cyberleninka.ru/article/n/realizatsiya-klasterizatsii-novostnyh-potokov-na-osnove-vektornyh-predstavleniy-teksta.

4. Pancherova E. V. Control work on classification methods in working with big data // CyberLeninka. URL: https://cyberleninka.ru/article/n/sravnitelnaya-otsenka-metodov-klasterizatsii-v-rabote-s-bolshimi-dannymi.

5. Aggarwal K. S., Khan J., Wang J., Yu P. S. A framework for clustering evolving data flows // VLDB, 2003. URL: https://www.vldb.org/conf/2003/papers/S04P02.pdf .

6. Cao F., Esther M., Qian W., Zhou A. Density-based clustering in a dynamic data stream with noise // SIAM, 2006. URL: https://www.cs.sfu.ca /~ester/papers/SDM2006.DenStream.final.pdf.

7. Zhang T., Ramakrishnan R., Livni M. BIRCH: an effective method of data clustering for very large databases // SIGMOD, 1996. URL: https://www2.cs.sfu.ca/CourseCentral/459/han/papers/zhang96.pdf .

8. Scully D. Clustering of k-funds on a web scale // WWW, 2010. URL: https://www.ccs.neu.edu/home/vip/teach/DMcourse/2_cluster_EM_mixt/notes_slides /sculey_webscale_kmeans_approx.pdf.

9. Scikit-learn documentation. MiniBatchKMeans. URL: https://scikit-learn.org/stable/modules/generated/sklearn.cluster.MiniBatchKMeans.html.