VNUHCM - UNIVERSITY OF SCIENCE
FACULTY OF INFORMATION TECHNOLOGY

Knowledge Engineering Department

Report Lab 3

Project 2: INTRODUCTION TO OPENSSL

Course: Introduction to Cryptography

Students:
Lê Trường Thịnh (23127018)
Trần Lý Nhật Hào (23127187)

Instructor:
Trịnh Văn Minh

Ngày 14 tháng 1 năm 2026

Mục lục

1 GIỚI THIỆU

Phân cụm là bài toán học không giám sát đã được nghiên cứu sâu rộng trong nhiều thập kỷ qua. Trong số các mục tiêu phân cụm khác nhau, một trong những công thức phổ biến nhất là phân cụm k-means. Trong phân cụm k-means, bài toán được cho là một tập P các điểm dữ liệu trong không gian Euclid d chiều, và mục tiêu là tính toán một tập $C \subset \mathbb{R}^d$ các tâm với kích thước tối đa k sao cho tổng bình phương khoảng cách từ các điểm dữ liệu trong P đến các tâm gần nhất trong C là nhỏ nhất. Bài toán k-means rất được quan tâm và được chứng minh là NP-khó [5]. Hơn nữa, kết quả các nghiên cứu chỉ ra rằng ngay cả việc tìm một nghiệm cho bài toán k-means với tỷ lệ xấp xỉ nhỏ hơn 1.07 cũng là NP-khó [4]. Tỷ lệ xấp xỉ tốt nhất hiện nay cho bài toán k-means là 5.912 [3], dựa trên các phương pháp đối ngẫu nguyên thủy (primal-dual) và tập độc lập tựa lồng nhau (nested quasi-independent set). Đối với số chiều d hoặc số cụm k cố định, đã có một số thuật toán xấp xỉ $(1+\epsilon)$ như [9, 7]. Tuy nhiên, các thuật toán này với ngưỡng xấp xỉ rất chặt thường không mở rộng (scale) tốt cho các tập dữ liệu quy mô lớn. Do đó, nhiều phương pháp thực tiễn với thời gian tuyến tính đã được đề xuất, chẳng hạn như phương pháp k-means++ với tỷ lệ xấp xỉ $O(\log k)$ [1] và các phương pháp local search xấp xỉ hệ số hằng số [10]. Mặc dù các thuật toán thời gian tuyến tính này được sử dụng rộng rãi, tỷ lệ xấp xỉ lớn của chúng có thể làm giảm hiệu suất phân cụm trong các kịch bản đòi hỏi nghiệm phải rất tốt.

Để vượt qua rào cản không thể xấp xỉ và phát triển các thuật toán thực tiễn hơn, một loạt các nghiên cứu đã tập trung vào các thuật toán có hỗ trợ học (learning-augmented) [12]. Đối với bài toán phân cụm, Gamlath và tác giả cộng sự [8] đã đề xuất khôi phục phân cụm với nhãn nhiễu, trong đó các nhãn phân cụm dự đoán là thông tin bổ trợ. Bộ dự đoán có tham số tỷ lệ lỗi $\alpha \in [0, 1)$, sao cho kích thước của hiệu đối xứng giữa cụm dự đoán và cụm tối ưu tương ứng được chặn bởi α lần kích thước cụm tối ưu. Dựa trên mô hình này, người ta đề ra một thuật toán xấp xỉ $(1 + O(\alpha))$ với thời gian chạy đa thức khi giả định k và d cố định. Ergun và tác giả cộng sự [6] đã giới thiệu một mô hình phân cụm có hỗ trợ học khác, hướng đến việc thiết kế các thuật toán nhanh và thực tiễn. Trong mô hình này, bộ dự đoán cung cấp thông tin cho mỗi điểm dữ liệu dưới dạng nhãn dự đoán với độ tin cậy α, đảm bảo rằng có tối đa một phần α dương tính giả và âm tính giả trong mỗi cụm dự đoán. Dựa trên mô hình này, một cải tiến xấp xỉ $(1 + O(\alpha))$ có thể đạt được với thời gian chạy gần tuyến tính.

Trong bài báo này, tác giả tập trung chủ yếu vào bài toán phân cụm có hỗ trợ học (learning-augmented) được đề xuất bởi Ergun và tác giả cộng sự [6]. Động lực nghiên cứu bài toán này đến từ hai phía. Về mặt lý thuyết, k-means có hỗ trợ học có thể vượt qua rào cản không thể xấp xỉ, cho ra ngưỡng chất lượng rất chặt cùng khả năng mở rộng cao. Về mặt thực tế, các bộ dự đoán đáng tin cậy luôn có cho nhiều loại dữ liệu tự nhiên. Ví dụ, nhãn huấn luyện có thể đóng vai trò thông tin bổ trợ để tăng cường chất lượng phân cụm trên tập kiểm tra. Ngay cả khi không có nhãn, thực nghiệm cho thấy nhãn từ các phương pháp hiện có như k-means++ [1] hoặc các phương pháp

heuristic như Lloyd [11] cũng có thể là bộ dự đoán tốt. Tuy nhiên, như đã chỉ ra bởi Nguyen và tác giả cộng sự [13], ngay cả khi bộ dự đoán gần như tối ưu, chỉ cần một điểm dương tính giả nằm xa các tâm tối ưu cũng có thể ảnh hưởng nghiêm trọng đến cấu trúc phân cụm. Do đó, thách thức then chốt là thiết kế các thuật toán bền vững để giảm thiểu tác động của dương tính giả.

Dựa trên các phương pháp thống kê, Ergun và tác giả cộng sự [6] đề xuất thuật toán ngẫu nhiên đạt mức xấp xỉ $(1+20\alpha)$ trong thời gian $O(md\log m)$. Tuy nhiên, thuật toán này yêu cầu các điều kiện khắt khe về tỷ lệ lỗi α và kích thước cụm tối ưu. Để khắc phục, Nguyen và tác giả cộng sự [13] đề xuất phương pháp tìm kiếm xác định đạt kết quả xấp xỉ $(1+O(\alpha))$ tốt hơn trong thời gian $O(md\log m)$ với $\alpha \in [0, 1/2)$. Các thuật toán hiện tại chủ yếu dựa trên chiến lược sắp xếp để xấp xỉ tâm tối ưu. Vì sắp xếp yêu cầu thời gian logarit và có cận dưới là $O(m\log m)$, điều này hạn chế khả năng mở rộng khi xử lý dữ liệu quy mô cực lớn. Ngoài ra, thời gian chạyđ này không thể cải thiện thông qua các kỹ thuật giảm chiều như phương pháp JL vì bản thân việc chiếu cũng tốn ít nhất $O(md\log m)$. Thách thức trung tâm trong việc thiết kế các thuật toán nhanh hơn là xấp xỉ hiệu quả các tâm tối ưu trong từng chiều mà không cần sử dụng các chiến lược dựa trên sắp xếp.

Đằng sau các thuật toán: Việc sử dụng sắp xếp trong các nghiên cứu trước đây nhằm xác định vị trí các điểm lân cận để lọc, nhưng điều này dẫn đến chi phí nhân với $\log m$. Mục tiêu của tác giả trong bài báo này là thay thế việc sắp xếp bằng các kỹ thuật lấy mẫu (sampling) và ước lượng (estimation). Bằng cách này, tác giả có thể xác định các khoảng chứa tâm tối ưu và tinh chỉnh chúng trong thời gian tuyến tính $O(md)$. Điều này cho phép thuật toán phá vỡ rào cản tính toán của các phương pháp dựa trên sắp xếp truyền thống mà vẫn giữ được độ chính xác xấp xỉ trong phạm vi sai số dự đoán α.

Bảng 1: Kết quả so sánh các thuật toán k-means có hỗ trợ học

Phương pháp và Tài liệu tham khảo	Tỷ lệ xấp xỉ	Khoảng lỗi nhãn α	Độ phức tạp thời gian
Phân vùng và Sắp xếp [6]	$1+20\alpha$	$[\frac{10\log m}{\sqrt{m}}, 1/7]$	$O(md\log m)$
Sắp xếp [13]	$1+\frac{\alpha}{1-\alpha}+\frac{4\alpha}{(1-2\alpha)(1-\alpha)}$	$[0, 1/2)$	$O(md\log m)$
Fast-Sampling (Tác giả)	$1+\frac{\alpha}{1-\alpha}+\frac{4\alpha+\alpha\epsilon}{(1-2\alpha)(1-\alpha)}$	$[0, 1/2)$	$O(\epsilon^{-1}md\log(kd))$
Fast-Estimation (Tác giả)	$1+\frac{\alpha}{1-\alpha}+\frac{13\alpha-15\alpha^2}{(1-3\alpha-\epsilon)(1-2\alpha-\epsilon)}$	$(0, 1/3-\epsilon)$	$O(md)+\tilde{O}(\epsilon^{-5}kd/\alpha)$
Fast-Filtering (Tác giả)	$1+6\sqrt{\alpha}+\frac{36(\sqrt{\alpha}+\alpha)}{1-(3+\epsilon)\alpha}+\frac{9(\sqrt{\alpha}+\alpha)}{(1-\alpha)(1-(3+\epsilon)\alpha)}$	$O(md)+\tilde{O}(\epsilon^{-5}kd/\alpha)$	

Nếu so sánh tỷ lệ xấp xỉ theo lý thuyết, các thuật toán của tác giả không tốt hơn so với các thuật toán sắp xếp.

Tuy nhiên ở phần thực nghiệm Fast-Filtering cho thấy chi phí phân cụm giảm đi 1.5%. Điều này có thể do mặc dù theo lí thuyết là tệ nhưng trong thực tế tỷ lệ xấp xỉ thường nhỏ hơn (tốt hơn).

2 Các công trình liên quan

Thuật toán k-means++ có tỷ lệ xấp xỉ $\Theta(\log k)$ so với đáp án phân cụm tối ưu, đồng thời cải thiện tốc độ, độ chính xác (accuracy) nếu biết nhãn. Mặc dù các thuật toán thời gian tuyến tính như k-means++ được sử dụng rộng rãi, nhưng không phừ hợp để dùng nếu cần độ chính xác cao (như kết quả thực nghiệm của bài báo này). Thuật toán chỉ khác k-means ở bước đầu "seeding"(chọn các tâm ban đầu), việc chọn các tâm là lấy mẫu D^2 (D^2-sampling):

1. Chọn tâm cụm đầu tiên c_1 ngẫu nhiên từ tập dữ liệu P.
2. Chọn tâm cụm tiếp theo c_i, chọn $c_i = x$ với xác suất $P(x) \propto D(x)^2$:

$$P(x) = \frac{D(x)^2}{\sum_{y \in P} D(y)^2}$$

3. Chạy thuật toán k-means.

2.1 Các thuật toán dựa trên sắp xếp

- **Thuật toán của Ergun và cộng sự [6]:** Phương pháp ngẫu nhiên này đạt được tỷ lệ xấp xỉ $(1 + 20\alpha)$ trong thời gian $O(md \log m)$.
- **Thuật toán của Nguyen và cộng sự [13]:** Tác giả đề xuất phương pháp tìm kiếm định tính để đạt được tỷ lệ xấp xỉ cải tiến $(1+O(\alpha))$ cho $\alpha \in [0, 1/2)$ trong cùng thời gian $O(md \log m)$.

Ngoài ra trong 2 bài báo trên, các tác giả còn đề xuất thuật toán k-medians. Các phương pháp này đều cần phí sắp xếp $O(\log m)$

2.2 Clustering tương tác

Giống với mô hình của bài báo này yêu cầu có 1 oracle cho biết các nhãn hay các cụm dự đoán P_i. Thuật toán có thể đưa ra các câu hỏi dạng: "Điểm p và điểm q có thuộc cùng một cụm tối ưu hay không?"[2]. Một hướng tiếp cận nâng cao khác là phân cụm phân cấp Bayesian có tương tác [14]

3 Kiến thức cơ sở

Gọi $P \subset \mathbb{R}^d$ và k lần lượt là tập dữ liệu và số lượng cụm. Gọi m là kích thước của tập dữ liệu. Đối với hai điểm $p, q \in \mathbb{R}^d$ bất kỳ, ký hiệu $\delta(p, q)$ và $\delta^2(p, q)$ lần lượt là khoảng cách và bình phương khoảng cách giữa chúng. Cho một điểm $p \in \mathbb{R}^d$ và một tập các tâm $C = \{c_1, c_2, \ldots, c_k\}$, gọi $\delta(p, C) = \min_{c \in C} \delta(p, c)$ là khoảng cách từ p đến tâm gần nhất trong C.

Để ý có thể có nhiều cách phân cụm tối ưu nhưng ở đây tác giả chọn 1 cách cố định để phân tích.

Gọi $C^* = \{c_1^*, \ldots, c_k^*\}$ và $P(C^*) = \{P_1^*, \ldots, P_k^*\}$ là tập các tâm tối ưu và phân hoạch phân cụm tối ưu tương ứng. Mỗi tâm tối ưu $c_i^* \in C^*$ được biểu diễn bởi d tọa độ, tức là $c_i^* = (c_{i1}^*, c_{i2}^*, \ldots, c_{id}^*)$. Chi phí phân cụm của tập P đối với tập tâm C được định nghĩa là:

$$\delta^2(P, C) = \sum_{x \in P} \delta^2(x, C) \tag{1}$$

Cho một tập hợp $L(P) = \{P_1, P_2, \ldots, P_k\}$ đóng vai trò là bộ dự đoán, gọi $Q_i = P_i \cap P_i^*$ là tập các điểm dữ liệu trong cụm dự đoán P_i thuộc cụm tối ưu P_i^*. Gọi tọa độ chiếu của các điểm dữ liệu trong P_i và Q_i lên chiều thứ j lần lượt là P_{ij} và Q_{ij}. Gọi P_{ij}^* là tọa độ chiếu của các điểm trong P_i^* lên chiều thứ j. Gọi m_i và m lần lượt là kích thước của P_i và P. Với một tập điểm dữ liệu $V \subset \mathbb{R}^d$, gọi $\overline{V}$ là tâm hình học của tập V. Gọi $P(j)$ là tọa độ chiếu của toàn bộ các điểm trong P lên chiều thứ j. Gọi Δ_{max} là tỷ lệ chiều tối đa của các điểm dữ liệu được chiếu, được xác định bởi:

$$\Delta_{max} = \max_{1 \le j \le d} \frac{\max_{x,y \in P(j)} \delta(x, y)}{\min_{x,y \in P(j), x \neq y} \delta(x, y)} \tag{2}$$

Cụm từ "Aspect Ratio" được tác giả đề cập khi dịch về tiếng Việt để quen thuộc nhất thì chúng em sẽ gọi là **tỷ lệ khung hình**. Về bản chất thì cũng chỉ là tỉ lệ giữa khoảng cách lớn nhất của 2 điểm và khoảng cách nhỏ nhất của 2 điểm.

Với một số nguyên dương t, gọi $[t]$ là tập hợp các số nguyên từ 1 đến t.

Bài toán k-means hỗ trợ học: Cho tập dữ liệu $P \subset \mathbb{R}^d$ gồm m điểm, gọi C^* và $P(C^*) = \{P_1^*, P_2^*, \ldots, P_k^*\}$ lần lượt là một lời giải tối ưu và phân hoạch tương ứng. Trong thiết lập có hỗ trợ học, giả định rằng có quyền truy cập vào một bộ dự đoán dưới dạng phân hoạch nhãn $L(P) = \{P_1, P_2, \ldots, P_k\}$ được tham số hóa bởi tỷ lệ lỗi nhãn $\alpha \in [0, 1)$, thỏa mãn điều kiện $|P_i \cap P_i^*| \ge (1 - \alpha) \max\{|P_i|, |P_i^*|\}$. Mục tiêu của bài toán là tìm tập $C \subset \mathbb{R}^d$ các tâm sao cho $\delta^2(P, C)$ đạt giá trị nhỏ nhất.

Các hệ quả toán học trong phần này dựa trên tính chất cơ bản của không gian Euclid, trọng tâm $\overline{V}$ là điểm duy nhất tối thiểu hóa tổng bình phương khoảng cách (SSE) tới mọi điểm trong tập V. Logic của các bổ đề dưới đây cho phép phân rã chi phí phân cụm thành hai thành phần: chi phí trong cụm (độ liên kết - cohesion) và chi phí do khoảng cách từ tâm dự đoán đến tâm tối ưu.

Các bổ đề dưới đây là "dân gian truyền miệng"(folklore) rất phổ biến liên quan bài toán k-means clustering

Lemma 1. *Cho tập $X \subset \mathbb{R}^d$ có kích thước m và một điểm dữ liệu bất kỳ $c \in \mathbb{R}^d$, ta luôn có:*

$$\delta^2(X, c) = \delta^2(X, \overline{X}) + m \cdot \delta^2(c, \overline{X}) \tag{3}$$

[1]

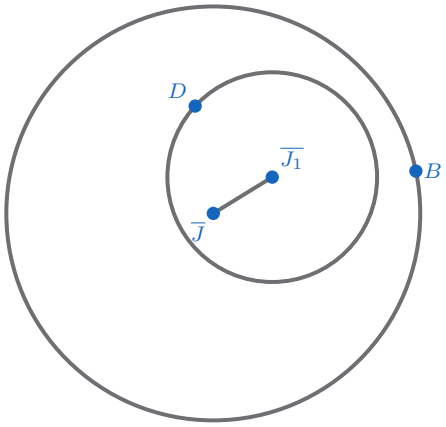
Bổ đề trên xuất phát từ quan sát trọng tâm $\overline{P_i}$ của các cụm dự đoán không đủ là nghiệm của bài toán. Vì dự đoán không đúng, có thể tồn tại 1 số điểm trong P_i nằm ngoài P_i^*. Nếu các điểm trong $P_i \setminus P_i^*$ nằm **rất xa** $\overline{P_i^*}$, trọng tâm cụm dự đoán sẽ bị lệch tuỳ ý dẫn đến chi phí tăng cụm tăng lên tuỳ ý.

Lemma 2. *Cho tập $J \subset \mathbb{R}$, gọi $J_1 \subseteq J$ với $|J_1| \geq (1-\zeta)|J|$, trong đó $0 \leq \zeta < 1$. Khi đó, mối liên hệ giữa chi phí của tập con và tập tổng thể được chặn bởi:*

$$\delta^2(\overline{J}, \overline{J_1}) \leq \frac{\zeta}{(1-\zeta)|J|}\delta^2(J, \overline{J}) \tag{4}$$

[13]

Bổ đề trên cũng đúng với $J \subset \mathbb{R}^d$, mặc dù tác giả chỉ ghi trên $\mathbb{R}$, xem chứng minh ở **??**.



Hình 1: Minh họa Bổ đề 2 trong không gian $\mathbb{R}^d$: Khoảng cách giữa tâm tập tổng thể $\overline{J}$ và tâm tập con $\overline{J_1}$ được chặn bởi hàm chi phí của J.

Bổ đề trên xuất phát từ quan sát liên hệ chi phí và kích thước tập con của cụm tối ưu. Ta muốn tìm $Q_i = P_i \cap P_i^*$ và lấy $\overline{Q_i}$ làm đáp án cho cụm i, điều này tự nhiên xuất phát từ dữ kiện Q_i của bài toán có hỗ trợ học.

$$|Q_i| \geq (1-\alpha)\max\{|P_i|, |P_i^*|\} \geq (1-\alpha)|P_i^*|$$
$$\Rightarrow |P_i^* \setminus Q_i| \leq \alpha m_i^*$$

Dùng bổ đề trên, ta có chặn trên chi phí phân cụm:

$$\begin{aligned}\delta^2(P_i^*, \overline{Q_i}) &= \delta^2(P_i^*, \overline{P_i^*}) + m_i^* \delta^2(\overline{P_i^*}, \overline{Q_i}) \text{ (bổ đề 1)} \\ &\leq \delta^2(P_i^*, \overline{P_i^*}) + m_i^* \frac{\alpha}{1-\alpha} \frac{\delta^2(P_i^*, \overline{P_i^*})}{m_i^*} \text{ (bổ đề 2)} \\ &= \left(1 + \frac{\alpha}{1-\alpha}\right) \delta^2(P_i^*, \overline{P_i^*})\end{aligned}$$

Như vậy ta có xấp xỉ $\left(1 + \frac{\alpha}{1-\alpha}\right)$ cho 1 cụm và cũng như cho bài toán. Cũng chính là chặn dưới và lí do xuất hiện của nó trong tỷ lệ xấp xỉ trong 1.

Khó khăn là Q_i chưa biết, vì vậy các thuật toán chính dưới đây tập trung vào việc loại bỏ các điểm outlier trong P_i, tìm một trọng tâm gần $\overline{Q_i}$, như vậy đồng thời giảm được khoảng cách đến $\overline{P_i^*}$ và chi phí.

Lemma 3. *Cho tập $X \subset \mathbb{R}^d$ và một giá trị $\alpha \in (0, 1]$, gọi $X' = \arg\min_{X'' \subseteq X, |X''| = \alpha|X|} \delta^2(X'', \overline{X''})$. Khi đó, ta có:*

$$\delta^2(X', \overline{X'}) \leq \alpha \cdot \delta^2(X, \overline{X}) \tag{5}$$

[13]

3.1 Các công cụ (toán)

Background Theorem 1 (Bất đẳng thức tam giác nới lỏng)**.** *Với mọi số thực $a, b \in \mathbb{R}$ và một tham số dương $\lambda > 0$, bất đẳng thức sau luôn thỏa mãn:*

$$(a+b)^2 \leq \left(1 + \frac{1}{\lambda}\right) a^2 + (1+\lambda) b^2$$

Trong không gian vector $\mathbb{R}^d$ với chuẩn Euclid $\|\cdot\|$, bất đẳng thức này tương đương với:

$$\|u+v\|^2 \leq \left(1 + \frac{1}{\lambda}\right) \|u\|^2 + (1+\lambda)\|v\|^2$$

Chứng minh. Ở đây nhóm em chứng minh cho 1 chiều, còn lại cũng tương tự. Ta bắt đầu bằng việc khai triển vế trái của bất đẳng thức:

$$(a+b)^2 = a^2 + 2ab + b^2$$

Ta áp dụng bất đẳng thức AM-GM. Với hai số thực dương x, y, ta luôn có $x^2 + y^2 \geq 2xy$. Chọn

$x = \frac{a}{\sqrt{\lambda}}$ và $y = b\sqrt{\lambda}$. Khi đó:

$$\left(\frac{a}{\sqrt{\lambda}}\right)^2 + (b\sqrt{\lambda})^2 \geq 2\left(\frac{a}{\sqrt{\lambda}}\right)(b\sqrt{\lambda})$$

$$\frac{a^2}{\lambda} + \lambda b^2 \geq 2ab$$

Thay thế chặn trên của $2ab$ vào khai triển ban đầu của $(a+b)^2$:

$$\begin{aligned}(a+b)^2 &= a^2 + 2ab + b^2 \\ &\leq a^2 + \left(\frac{a^2}{\lambda} + \lambda b^2\right) + b^2 \\ &= \left(a^2 + \frac{a^2}{\lambda}\right) + (b^2 + \lambda b^2) \\ &= \left(1 + \frac{1}{\lambda}\right) a^2 + (1+\lambda) b^2\end{aligned}$$

□

3.2 Các thuật toán con

3.2.1 Chọn phần tử hạng thứ i trong dãy

Hay chọn trung vị trong $O(m)$. Thay vì $O(m \log m)$

4 THUẬT TOÁN FAST-SAMPLING

Tóm tắt: abc

Ý tưởng tổng quát của thuật toán Fast-Sampling là xấp xỉ hiệu quả các tâm tối ưu trong từng chiều bằng cách xác định các tọa độ chất lượng cao mà không cần sử dụng các chiến lược dựa trên sắp xếp. Thách thức kỹ thuật chính nằm ở việc xử lý các âm tính giả mà không làm ảnh hưởng đáng kể đến các đảm bảo xấp xỉ. Mặc dù việc lấy mẫu trực tiếp một tập con nhỏ các tọa độ từ mỗi chiều của các cụm dự đoán có thể giúp xác định các điểm gần tâm tối ưu, các tọa độ được lấy mẫu theo phân phối đều có thể không xấp xỉ chính xác các tâm tối ưu, tiềm ẩn nguy cơ dẫn đến mất mát hằng số trong các đảm bảo xấp xỉ. Để giải quyết vấn đề này, thuật toán Fast-Sampling trước tiên xác định các tọa độ ứng viên gần với tọa độ của từng tâm tối ưu trong thời gian chạy tuyến tính đối với quy mô dữ liệu. Sau đó, các tọa độ ứng viên được xây dựng sẽ được sử dụng để xác định các khoảng phủ chính xác vị trí của các tâm tối ưu, cho phép xấp xỉ tốt hơn thông qua việc chia nhỏ các khoảng này kĩ (fine-grained).

Thuật toán Fast-Sampling đề xuất chủ yếu bao gồm hai giai đoạn sau: (1) ước lượng khoảng (bước 3-6 của Thuật toán 1); (2) xây dựng tọa độ ứng viên (bước 7 của Thuật toán 1). Trong giai đoạn ước lượng khoảng, đối với mỗi chiều của các cụm dự đoán, độ dài khoảng được ước tính thông qua các chiến lược lấy mẫu ngẫu nhiên. Các mẫu sau đó được điều chỉnh đối xứng (qua tâm) dựa trên các ước tính độ dài khoảng để xây dựng các khoảng có thể bao quanh tọa độ của các tâm tối ưu. Trong giai đoạn thứ hai, các khoảng thu được được chia thành các phần nhỏ hơn, mỗi phần tương ứng với một tọa độ ứng viên mới, cho phép xấp xỉ mịn các tâm tối ưu. Dưới đây là phân tích chi tiết cho thuật toán được đề xuất.

Thuật toán 1 Fast-Sampling

Đầu vào: Một bài toán k-means (P, k, d), một tập $(P_1, ..., P_k)$ các cụm với tỷ lệ lỗi α, và một tham số $\epsilon \in (0, 1]$.

Đầu ra: Một tập $C \subset \mathbb{R}^d$ các tâm với $|C| = k$.

1: **for** $i \in [k]$ **do**
2: **for** $j \in [d]$ **do**
3: Lấy mẫu ngẫu nhiên và độc lập để tạo tập U_{ij} từ P_{ij} với kích thước $O(\log(kd))$.
4: **for** $u \in U_{ij}$ **do**
5: Gọi $\mathcal{N}_{ij}(u)$ là tập $(1-\alpha)|P_i|$ tọa độ trong P_{ij} gần u nhất.
6: $l_{ij} = \sqrt{\frac{2\delta^2(\mathcal{N}_{ij}(u), \overline{\mathcal{N}_{ij}(u)})}{(1-\alpha)|P_i|}}$.
7: $s(u) = \{u + \epsilon' \lambda l_{ij} : \lambda \in [-\frac{1}{\epsilon'}, \frac{1}{\epsilon'}] \cap \mathbb{Z}\}$, với $\epsilon' = \sqrt{\frac{\epsilon}{48}}$.
8: $U'_{ij} = \bigcup_{u \in U_{ij}} s(u)$.
9: $u_1 = \arg\min_{u \in U'_{ij}} \delta^2(\mathcal{N}_{ij}(u), \overline{\mathcal{N}_{ij}(u)})$.
10: $I_{ij} = \mathcal{N}_{ij}(u_1)$.
11: $\hat{c}_i = (\overline{I}_{ij})_{j \in [d]}$.
12: **return** $\{\hat{c}_1, \hat{c}_2, ..., \hat{c}_k\}$.

Giải thích thuật toán: Trình tự của Fast-Sampling dựa trên việc khai thác cấu trúc của tập điểm trong từng chiều không gian. Việc lấy mẫu $O(\log(kd))$ điểm đảm bảo rằng với xác suất cao, ít nhất một điểm ứng viên u sẽ rơi vào vùng mật độ cao của cụm tối ưu. Tại bước 6, giá trị l_{ij} được tính toán dựa trên độ lệch chuẩn của tập lân cận gần nhất $\mathcal{N}_{ij}(u)$, đóng vai trò là thước đo khoảng cách đặc trưng để thiết lập lưới tìm kiếm. Kỹ thuật chia lưới trong bước 7 giúp chuyển đổi bài toán tìm kiếm liên tục thành rời rạc với sai số được kiểm soát bởi ϵ', từ đó tránh được việc phải sắp xếp toàn bộ dữ liệu, giúp duy trì độ phức tạp tuyến tính.

Đầu tiên, tác giả xem xét một chiều đơn lẻ $j \in [d]$ của một cụm dự đoán bất kỳ P_i với $i \in [k]$. Gọi $Q'_{ij} \subseteq Q_{ij}$ là tập hợp các tọa độ có kích thước $(1-\alpha)m_i$ và chi phí phân cụm nhỏ nhất. Bắt đầu từ bước 3 của Thuật toán 1, một tập U_{ij} được xây dựng bằng cách lấy mẫu ngẫu nhiên và độc lập

$O(\log(kd))$ mẫu từ P_{ij}. Mục tiêu ở đây là tìm các tọa độ gần với tọa độ của các tâm tối ưu. Tác giả sẽ chứng minh rằng, với xác suất nhất định, tồn tại ít nhất một tọa độ $u \in U_{ij}$ có thể xấp xỉ tốt trọng tâm của Q'_{ij}. Để phân tích xác suất thành công, tác giả định nghĩa G^{μ}_{ij} là tập hợp các tọa độ gần với Q'_{ij}. Bằng cách áp dụng chặn Union Bound cho xác suất thành công trên tất cả các chiều và các cụm dự đoán, tác giả có thể lập luận rằng với xác suất hằng số, tồn tại ít nhất một tọa độ $u \in U_{ij}$ sao cho $u \in G^2_{ij} \cap U_{ij}$.

Dựa trên các tọa độ đã lấy mẫu, trong các bước còn lại của giai đoạn ước lượng khoảng (bước 4-6), thuật toán Fast-Sampling ước tính độ dài khoảng để xác định các vùng tiềm năng có thể bao quanh trọng tâm của Q'_{ij}. Theo các hệ quả rút ra, có thể giả định rằng luôn tồn tại ít nhất một tọa độ $u \in U_{ij} \cap G^2_{ij}$. Sau đó, trong bước 5, thuật toán xác định tập $\mathcal{N}_{ij}(u)$ gồm $(1-\alpha)m_i$ tọa độ trong P_{ij} gần u nhất. Các bổ đề cho thấy cả chặn dưới và chặn trên cho $\delta^2(Q'_{ij}, \overline{Q}'_{ij})$ đều có thể được thiết lập bằng cách sử dụng $\mathcal{N}_{ij}(u)$. Nếu điểm được lấy mẫu u thuộc G^2_{ij}, bằng cách xác định tập $\mathcal{N}_{ij}(u)$, tác giả có thể thu được các khoảng bao quanh Q'_{ij} với độ dài xác định.

Trong giai đoạn xây dựng tọa độ ứng viên (bước 7), thuật toán tiếp tục chia các khoảng thành các khối nhỏ hơn, trong đó độ dài mỗi khối được tham số hóa bởi $\epsilon' = \sqrt{\epsilon/48}$. Do đó, tập ứng viên U'_{ij} được xây dựng ở bước 8 sẽ chứa ít nhất một tọa độ u' đủ gần với trọng tâm của Q'_{ij}.

Bắt đầu từ bước 9, thuật toán Fast-Sampling liệt kê tất cả các tọa độ ứng viên đã xây dựng và $(1-\alpha)m_i$ lân cận gần nhất của chúng để xác định tập hợp tọa độ có chi phí phân cụm nhỏ nhất. Sau đó, trọng tâm của tập hợp này được chọn để làm tọa độ cho tâm. Gọi I_{ij} là tập hợp các tọa độ được tìm thấy ở bước 10. Các bổ đề chứng minh rằng khoảng cách giữa Q_{ij} và I_{ij} có thể được chặn bằng cách sử dụng $I_{ij} \cap Q_{ij}$ để bắc cầu. Kết hợp các kết quả này, tác giả thiết lập được chặn cho khoảng cách giữa I_{ij} và P^*_{ij}. Tổng hợp lại, thuật toán có thể đưa ra nghiệm xấp xỉ $(1+O(\alpha))$ cho bài toán k-means có hỗ trợ học trong thời gian $O(\epsilon^{-1}md\log(kd))$.

Lemma 4. *Với mọi $Q_{ij} = P^*_{ij} \cap P_{ij}$, gọi Q'_{ij} là tập con của Q_{ij} có kích thước $(1-\alpha)m_i$ và chi phí phân cụm nhỏ nhất. Gọi $G^{\mu}_{ij} = \{x \in Q'_{ij} : \delta^2(x, \overline{Q'_{ij}}) \leq \mu\delta^2(Q'_{ij}, \overline{Q'_{ij}})/|Q'_{ij}|\}$ là tập hợp các tọa độ "tốt"với hằng số $\mu > 1$. Khi đó:*

$$|G^{\mu}_{ij}| \geq \frac{\mu - 1}{\mu}|Q'_{ij}|$$

Corollary 0.1. *Với xác suất hằng số, đối với mỗi cụm dự đoán và mỗi chiều , tồn tại ít nhất một tọa độ sao cho .*

Lemma 5. *Cho một tọa độ bất kỳ $u \in G^2_{ij} \cap U_{ij}$, bất đẳng thức sau luôn thỏa mãn:*

$$\delta^2(Q'_{ij}, \overline{Q'_{ij}}) \leq \delta^2(\mathcal{N}_{ij}(u), \overline{\mathcal{N}_{ij}(u)}) \leq 3\delta^2(Q'_{ij}, \overline{Q'_{ij}}) \tag{6}$$

trong đó $\overline{Q'_{ij}}$ và $\overline{\mathcal{N}_{ij}(u)}$ lần lượt là tâm hình học của tập Q'_{ij} và $\mathcal{N}_{ij}(u)$.

Corollary 0.2. *Với xác suất hằng số, đối với mỗi chiều $j \in [d]$ của mỗi cụm $i \in [k]$, tồn tại ít nhất*

một tọa độ $u' \in U'_{ij}$ sao cho:

$$\delta(u', \overline{Q'_{ij}}) \leq \sqrt{\frac{\epsilon\delta^2(Q'_{ij}, \overline{Q'_{ij}})}{2(1-\alpha)m_i}} \tag{7}$$

trong đó $\overline{Q'_{ij}}$ là tâm hình học của tập tối ưu Q'_{ij}.

Lemma 6. *Giới hạn sau đây luôn đúng đối với tập hợp các tọa độ I_{ij} được xác định bởi thuật toán Fast-Sampling so với tập giao Q_{ij}:*

$$\delta^2(\overline{I_{ij}}, \overline{Q_{ij}}) \leq \frac{(4\alpha + \alpha\epsilon)\delta^2(Q_{ij}, \overline{Q_{ij}})}{|Q_{ij}|(1-2\alpha)}$$

Lemma 7. *Khoảng cách giữa tọa độ của tâm thuật toán $\overline{I_{ij}}$ và tâm tối ưu $\overline{P^*_{ij}}$ bị chặn bởi:*

$$\delta^2(\overline{I_{ij}}, \overline{P^*_{ij}}) \leq \left(\frac{\alpha}{1-\alpha} + \frac{\alpha(4+\epsilon)}{(1-2\alpha)(1-\alpha)}\right)\frac{\delta^2(P^*_{ij}, \overline{P^*_{ij}})}{|P^*_{ij}|}$$

Theorem 1. *Tồn tại một thuật toán k-means có hỗ trợ học (Fast-Sampling) trả về một giải pháp xấp xỉ $(1 + O(\alpha))$ trong thời gian $O(\epsilon^{-1}md\log(kd))$ với xác suất hằng số, trong đó tỷ lệ lỗi nhãn thỏa mãn $\alpha \in [0, 1/2)$.*

5 Fast-Estimation

Mặc dù thuật toán Fast-Sampling có thời gian chạy tuyến tính trong khi vẫn duy trì các đảm bảo về mặt xấp xỉ, nhưng vẫn có $O(\log(kd))$ khi thực hiện chặn hội tụ xác suất, có thể ảnh hưởng trong thực tế của thuật toán khi xử lý các tập dữ liệu quy mô cực lớn. Để giải quyết vấn đề này, trong phần này, tác giả đề xuất một thuật toán dựa trên lấy mẫu nhanh hơn mang tên Fast-Estimation. Thuật toán Fast-Estimation có thể xấp xỉ hiệu quả tọa độ của từng cụm dự đoán trong thời gian chạy tuyến tính, với một sự đánh đổi nhỏ trong các đảm bảo về chất lượng phân cụm.

Ý tưởng chính: trước tiên tạo ra các tọa độ ứng viên có khả năng xấp xỉ chặt chẽ tọa độ của các tâm tối ưu. Sau đó, trong mỗi chiều của từng cụm dự đoán, một bộ ước lượng (estimator) được xây dựng bằng cách lấy mẫu theo phân phối đều. Bộ ước lượng này được thiết kế để cung cấp các ước tính chi phí phân cụm chính xác cho các tập con tọa độ có kích thước $(1-\alpha)m_i$. Cụ thể, đối với mỗi chiều của từng cụm dự đoán, bộ ước lượng được xây dựng bằng cách chọn ngẫu nhiên một tập S_{ij} từ P_{ij}. Mỗi tọa độ được lấy mẫu sau đó được gán một trọng số bằng nhau, vì vậy xấp xỉ chi phí phân cụm thông qua các mẫu trọng số thay vì tính toàn bộ cụm dự đoán. Với các bộ ước lượng đã xây dựng, việc tìm kiếm tập hợp các tọa độ có chi phí phân cụm tối thiểu có thể được thực hiện trong thời gian hạ tuyến tính (sub-linear), loại bỏ nhân với $O(\log(kd))$ khỏi thời gian chạy của thuật toán Fast-Sampling.

Thuật toán 2 Fast-Estimation

Đầu vào: Một bài toán k-means (P, k, d), một tập các phân vùng $(P_1, P_2, ..., P_k)$ với tỷ lệ lỗi α, và tham số $0 < \epsilon < 0.5$.

Đầu ra: Một tập $C \subset \mathbb{R}^d$ các tâm với $|C| = k$.

1: **for** $i \in [k]$ **do**
2: **for** $j \in [d]$ **do**
3: Lấy mẫu ngẫu nhiên và độc lập một tập U_{ij} từ P_{ij} với kích thước $O(\log(kd))$, sau đó khởi tạo $U'_{ij} = \emptyset$ và $\epsilon_1 = \frac{\epsilon}{126}$.
4: **for** $q = 0$ to $O(\log(m\Delta^2_{max}))$ **do**
5: $l_{ij} = \sqrt{\frac{2^{q-1}}{(1-\alpha)m_i}}$.
6: **for** $u \in U_{ij}$ **do**
7: $s(u) = \{u + \epsilon_2 \lambda l_{ij} : \lambda \in [-\frac{1}{\epsilon_2}, \frac{1}{\epsilon_2}] \cap \mathbb{Z}\}$, với $\epsilon_2 = \sqrt{\frac{\epsilon_1}{32}}$.
8: $U'_{ij} = U'_{ij} \cup s(u)$.
9: Lấy mẫu ngẫu nhiên và độc lập một tập S_{ij} từ P_{ij} với kích thước $O\left(\frac{\log(m^3 d \log^3(m\Delta^2_{max})/\epsilon_1^2)\log(m\Delta^2_{max})}{\alpha\epsilon_1^4}\right)$, gán cho mỗi điểm trong S_{ij} một trọng số $\frac{m_i}{|S_{ij}|}$.
10: Xây dựng bộ ước lượng ω sao cho $\forall u \in U'_{ij}$, $\omega(u) = \sum_{p \in S_{ij} \setminus F(u)} \frac{m_i}{|S_{ij}|}\delta^2(p, u)$, trong đó $F(u)$ là tập hợp $(1 + 3\epsilon_1)\alpha|S_{ij}|$ điểm xa u nhất trong S_{ij}.
11: $c_{ij} = \arg\min_{u \in U'_{ij}} \omega(u)$.
12: Gọi I_{ij} là tập hợp $(1 - 2\alpha - \alpha\epsilon)m_i$ tọa độ gần c_{ij} nhất từ P_{ij}.
13: $\hat{c}_i = (I_{ij})_{j \in [d]}$.
14: **return** $\{\hat{c}_1, \hat{c}_2, ..., \hat{c}_k\}$.

Phân tích thuật toán:

Trong bước 3, đối với mỗi chiều của cụm dự đoán, thuật toán chọn một mẫu ngẫu nhiên U_{ij} để xấp xỉ tọa độ của các tâm tối ưu. Theo Lemma 4, với xác suất hằng số, tồn tại ít nhất một tọa độ được lấy mẫu $u \in U_{ij}$ sao cho $\delta(u, Q'_{ij}) \leq \sqrt{2\delta^2(Q'_{ij}, \overline{Q'_{ij}})/|Q'_{ij}|}$. Sau đó, từ bước 4 liệt kê tất cả các độ dài khoảng ứng viên để xây dựng tập hợp các tọa độ ứng viên. Không mất tính tổng quát, có thể giả sử khoảng cách cặp tối thiểu giữa các tọa độ trong P_{ij} là 1 và khoảng cách cặp tối đa là Δ_{max}. Do đó, trong bước 5, tồn tại ít nhất một lần đoán q cho độ dài thoả $\sqrt{\frac{2\delta^2(Q'_{ij}, \overline{Q'_{ij}})}{(1-\alpha)m_i}} \leq l_{ij} \leq \sqrt{\frac{4\delta^2(Q'_{ij}, \overline{Q'_{ij}})}{(1-\alpha)m_i}}$. Tiếp theo, trong các bước 7-8, theo Lemma 5, cũng tồn tại ít nhất một tọa độ $u' \in U'_{ij}$ sao cho u' đủ gần với trọng tâm của Q'_{ij}, tức là $\delta(u', Q'_{ij}) \leq \sqrt{\epsilon_1\delta^2(Q'_{ij}, \overline{Q'_{ij}})/|Q'_{ij}|}$.

Đối với mỗi $u \in U'_{ij}$, gọi $\mathcal{N}_{ij}(u)$ là tập hợp $(1 - \alpha)m_i$ tọa độ gần nhất từ P_{ij} đến u. Gọi $O(u) = P_{ij} \setminus \mathcal{N}_{ij}(u)$ là tập hợp αm_i tọa độ xa nhất từ P_{ij} đến u. Trước khi xây dựng bộ ước lượng ω (bước 9-10), tác giả bắt đầu bằng cách chia $\mathcal{N}_{ij}(u)$ thành $\gamma = \frac{(1+\epsilon_1)\log(m\Delta^2_{max})}{\epsilon_1}$ khối. Cụ thể, đối với mỗi $u \in U'_{ij}$, $\mathcal{N}_{ij}(u)$ được phân rã thành γ khối (ký hiệu là $\mathcal{B}^1_u, \mathcal{B}^2_u, \ldots, \mathcal{B}^\gamma_u$) dựa trên khoảng cách từ các tọa độ trong $\mathcal{N}_{ij}(u)$ đến u, trong đó $\mathcal{B}^l_u = \{x \in \mathcal{N}_{ij}(u) : (1 + \epsilon_1)^l \leq \delta^2(x, u) < (1 + \epsilon_1)^{l+1}\}$.

Các "khối"này có thể hình dung là các phần tiếp nối giữa khối cầu có bán kính $(1+\epsilon_1)^l$ và $(1+\epsilon_1)^{l+1}$

trong $\mathbb{R}^d$

Sau đó, các khối này được chia tiếp thành hai nhóm dựa trên kích thước: $\mathcal{L}(u) = \{\mathcal{B}_u^l : |\mathcal{B}_u^l| \geq \frac{\epsilon_1^2 \alpha m_i}{(1+\epsilon_1)\log(m_i \Delta_{max}^2)}, l \in [\gamma]\}$ là nhóm các khối lớn và $\mathcal{S}(u) = \{\mathcal{B}_u^1, \ldots, \mathcal{B}_u^\gamma\} \setminus \mathcal{L}(u)$ là nhóm các khối nhỏ.

Sự hội tụ xác suất:

Mục tiêu là xấp xỉ tốt từng khối lớn trong $\mathcal{L}(u)$ đồng thời cho phép bỏ qua các tọa độ trong các khối nhỏ.

1. **Biến ngẫu nhiên:** Đối với mỗi mẫu $p \in S_{ij}$, xét biến ngẫu nhiên chỉ thị cho việc p rơi vào một khối cụ thể.
2. **Áp dụng Bất đẳng thức Chernoff:** Với kích thước mẫu $|S_{ij}|$ được chọn, kỳ vọng số điểm rơi vào mỗi khối lớn đủ lớn để xác suất sai lệch quá ϵ_1 lần kỳ vọng bị chặn bởi một hàm mũ âm. Cụ thể, $Pr(|X - \mathbb{E}[X]| \geq \epsilon_1 \mathbb{E}[X]) \leq 2e^{-\epsilon_1^2 \mathbb{E}[X]/3}$.
3. **Chặn hội tụ (Union Bound):** Bằng cách lấy tổng xác suất lỗi trên tất cả các khối và các tọa độ ứng viên, tác giả đảm bảo rằng bộ ước lượng ω hoạt động chính xác với xác suất cao trên toàn không gian ứng viên.

Với bộ ước lượng đã được chứng minh là hội tụ về giá trị thực, việc tìm c_{ij} tại bước 11 nhanh hơn vì số lượng ứng viên $|U'_{ij}|$ chỉ phụ thuộc logarit vào Δ_{max} và m, trong khi việc tính toán mỗi giá trị $\omega(u)$ chỉ tốn thời gian phụ thuộc vào kích thước mẫu $|S_{ij}|$ thay vì kích thước toàn bộ dữ liệu m_i. Cuối cùng, bằng cách sử dụng Lemma 7, Theorem 2 có thể được chứng minh để độ phức tạp thời gian tuyến tính $O(md) + \tilde{O}(\epsilon^{-5}kd/\alpha)$ cho bài toán có hỗ trợ học.

Lemma 8. *Giả sử S_{ij} là một mẫu được lấy ngẫu nhiên từ cụm dự đoán P_{ij} với kích thước mẫu $|S_{ij}| = \tilde{O}(1/\alpha\epsilon_1^4)$. Với xác suất ít nhất $1 - \frac{\epsilon_1}{m^3 d \log^2(m\Delta_{\max}^2)}$, các bất đẳng thức sau đây đồng thời xảy ra cho mọi khối lớn $\mathcal{B}_u^l \in \mathcal{L}(u)$ và tập các điểm xa nhất $\mathcal{O}(u)$:*

$$(1-\epsilon_1)\mathbb{E}[|\mathcal{B}_u^l \cap S_{ij}|] \leq |\mathcal{B}_u^l \cap S_{ij}| \leq (1+\epsilon_1)\mathbb{E}[|\mathcal{B}_u^l \cap S_{ij}|]$$

$$(1-\epsilon_1)\mathbb{E}[|\mathcal{O}(u) \cap S_{ij}|] \leq |\mathcal{O}(u) \cap S_{ij}| \leq (1+\epsilon_1)\mathbb{E}[|\mathcal{O}(u) \cap S_{ij}|]$$

Lemma 9. *Gọi $\mathcal{J}(u)$ là tập hợp các tọa độ nằm trong các khối nhỏ đối với một tọa độ ứng viên u. Với xác suất ít nhất $1 - \frac{\epsilon_1}{m^3 d \log^2(m\Delta_{\max}^2)}$, giao của tập mẫu S_{ij} và $\mathcal{J}(u)$ bị chặn như sau:*

$$|\mathcal{J}(u) \cap S_{ij}| \leq 2\epsilon_1 \alpha |S_{ij}|$$

Lemma 10. *Cho một tọa độ ứng viên bất kỳ $u \in U'_{ij}$. Với xác suất cao (xác suất hằng số), ước lượng $\omega(u)$ thỏa mãn các chặn sau:*

$$\frac{\delta^2(P_{ij} \setminus \mathcal{F}^\dagger(u), u)}{1 + 7\epsilon_1} \leq \omega(u) \leq (1+\epsilon_1)^2 \delta^2(\mathcal{N}_{ij}(u), u)$$

trong đó:

- $\mathcal{F}^{\dagger}(u)$ *là tập hợp gồm* $(2+20\epsilon_1)\alpha m_i$ *tọa độ xa nhất từ* P_{ij} *đến* u.
- $\mathcal{N}_{ij}(u)$ *là tập hợp gồm* $(1-\alpha)m_i$ *tọa độ gần nhất trong* P_{ij} *đến* u.

Lemma 11. *Với tập hợp các tọa độ* I_{ij} *được xác định bởi thuật toán Fast-Estimation, chặn sau đây luôn thỏa mãn:*

$$\delta^2(\overline{I_{ij}}, \overline{Q_{ij}}) \leq \frac{13\alpha - 15\alpha^2}{(1-3\alpha-\epsilon)(1-2\alpha-\epsilon)} \frac{\delta^2(Q_{ij}, \overline{Q_{ij}})}{|Q_{ij}|}$$

Theorem 2. *Thuật toán Fast-Estimation xấp xỉ* $(1+O(\alpha))$ *cho bài toán k-means có hỗ trợ học (learning-augmented) trong thời gian* $O(md) + \tilde{O}(\epsilon^{-5}kd/\alpha)$ *với xác suất hằng số, với tỷ lệ lỗi nhãn* $\alpha \in (0, 1/3-\epsilon)$.

6 Fast-Filtering

Đối với Fast-Sampling và Fast-Estimation, các tâm được tạo ra bằng cách tìm xấp xỉ tọa độ trong từng chiều không gian. Tuy nhiên, quy trình lấy mẫu này có thể làm phát sinh các sai số tích lũy, dẫn đến sự suy giảm chất lượng phân cụm tổng thể. Trong phần này, dựa trên các thuật toán Fast-Sampling và Fast-Estimation, tác giả đề xuất một thuật toán heuristic thực tiễn hơn mang tên Fast-Filtering nhằm bảo toàn tốt hơn chất lượng phân cụm trong khi vẫn duy trì được thời gian chạy hiệu quả.

Thuật toán đề xuất được trình bày trong Thuật toán 3, với ý tưởng chủ đạo là trực tiếp tìm kiếm các xấp xỉ tâm cho từng cụm dự đoán thay vì xấp xỉ từng chiều độc lập. Tại bước 2, một tập hợp các mẫu được rút ra một cách ngẫu nhiên và độc lập từ mỗi cụm dự đoán để đóng vai trò là các tâm ứng viên. Sau đó, trong các bước 3-4, các bộ ước lượng được xây dựng dựa trên những ý tưởng tương tự từ thuật toán Fast-Estimation. Dựa trên các bộ ước lượng này, tâm ứng viên có chi phí phân cụm tối thiểu được lựa chọn tại bước 5 để xác định các khoảng chứa $(1-\alpha)m_i$ điểm gần nhất. Cuối cùng, tại bước 7, các trọng tâm của các tập điểm đã xác định được chọn làm các tâm cuối cùng. Trong Phụ lục A.4, tác giả cung cấp phân tích lý thuyết cho thuật toán Fast-Filtering và chỉ ra rằng, với việc điều chỉnh số lượng lân cận gần nhất cùng kích thước mẫu R_1 và R_2, thuật toán này có thể đưa ra một nghiệm xấp xỉ $(1+O(\sqrt{\alpha}))$.

Giải thích thuật toán:

Thuật toán này giải quyết vấn đề "sai số tích lũy"bằng cách làm trực tiếp trên vecto thay vì gộp kết quả từ d bài toán đơn chiều.

- **Ước lượng nhanh:** Ý tưởng giống Fast-Estimation. Tại bước 4, thay vì tính toán tổng bình phương khoảng cách δ^2 trên toàn bộ tập dữ liệu P_i (vốn tốn thời gian $O(m_i d)$), tác giả sử dụng tập mẫu S_i có kích thước R_2 nhỏ hơn nhiều. Trọng số $\frac{m_i}{|S_i|}$ đảm bảo rằng kỳ vọng của bộ ước lượng $\omega(u)$ sẽ hội tụ về giá trị chi phí thực tế của cụm.

Thuật toán 3 Fast-Filtering

Đầu vào: Bài toán k-means (P, k, d), tập các phân vùng $(P_1, P_2, \ldots, P_k)$ với tỷ lệ lỗi α, các tham số $R_1 > 0, R_2 > 0$ và $0 < \epsilon < 1$.

Đầu ra: Một tập $C \subset \mathbb{R}^d$ các tâm với $|C| \leq k$.

1: **for** $i = 1..k$ **do**
2: Lấy mẫu ngẫu nhiên và độc lập một tập U_i từ P_i với kích thước R_1.
3: Lấy mẫu ngẫu nhiên và độc lập một tập S_i từ P_i với kích thước R_2, và gán cho mỗi điểm trong S_i một trọng số $\frac{m_i}{|S_i|}$.
4: Xây dựng bộ ước lượng ω sao cho $\forall u \in U_i, \omega(u) = \sum_{p \in S_i \setminus F(u)} \frac{m_i}{|S_i|} \delta^2(p, u)$, trong đó $F(u)$ là tập hợp $(1+\epsilon)\alpha|S_i|$ điểm trong S_i có khoảng cách xa nhất đối với u.
5: $c_i = \arg\min_{u \in U_i} \omega(u)$.
6: Gọi I_i là tập hợp $(1-\alpha)m_i$ điểm trong P_i gần c_i nhất.
7: $\hat{c}_i = \bar{I}_i$.
return $\{\hat{c}_1, \hat{c}_2, \ldots, \hat{c}_k\}$.

- **Loại bỏ nhiễu (Filtering):** Một đóng góp quan trọng của tác giả là việc định nghĩa tập $F(u)$ gồm các điểm xa nhất. Trong bài toán có hỗ trợ học, cụm dự đoán P_i có thể chứa tới αm_i điểm âm tính giả (nhiễu). Nếu các điểm nhiễu này nằm rất xa tâm thực, chúng sẽ kéo trọng tâm lệch khỏi vị trí tối ưu. Việc loại bỏ $F(u)$ trong quá trình ước lượng giúp "cô lập"ảnh hưởng của các điểm ngoại lệ này, giúp việc chọn c_i trở nên bền bỉ (robust) hơn.
- **Trọng tâm:** Sau khi đã xác định được một tâm ứng viên tốt c_i, thuật toán thực hiện một bước tinh chỉnh tại bước 6 và 7. Tập I_i đại diện cho phần "lõi"sạch nhất của cụm. Theo Lemma 1, trọng tâm $\bar{I}_i$ là điểm duy nhất tối thiểu hóa tổng bình phương khoảng cách tới tất cả các điểm trong tập đó. Do đó, $\hat{c}_i$ chính là nghiệm tối ưu địa phương cho tập điểm đã được lọc nhiễu.

Sự kết hợp giữa lấy mẫu ngẫu nhiên để tìm ứng viên và bộ lọc thống kê để đánh giá chi phí cho phép Fast-Filtering đạt được sự cân bằng giữa tốc độ tính toán và độ chính xác phân cụm.

Theorem 3. *Cho* $R_1 = O\left(\frac{\log k}{1-2\alpha}\right)$ *và* $R_2 = O\left(\frac{\log(m^3 d \log^3(m\Delta^2)/\epsilon^2)\log(m\Delta^2)}{\alpha\epsilon^4}\right)$*, trong đó* Δ *là tỷ lệ chiều của tập dữ liệu. Với xác suất hằng số, Thuật toán 4 (Fast-Filtering) trả về nghiệm xấp xỉ* $(1+O(\sqrt{\alpha}))$ *cho bài toán k-means có hỗ trợ học trong thời gian* $O(md) + \tilde{O}\left(\frac{kd}{\epsilon^4(1-2\alpha)\alpha}\right)$ *với* $\alpha \in (0, 1/3 - \epsilon)$.

Corollary 3.1. *Cho kích thước mẫu* $R_1 = \Theta\left(\frac{\log k}{1-2\alpha}\right)$*. Với mỗi cụm dự đoán* $i \in [k]$*, với xác suất hằng số, tồn tại ít nhất một điểm dữ liệu u trong tập mẫu* U_i *sao cho* $u \in G_2(P_i^*)$*, trong đó* $G_2(P_i^*)$ *là tập hợp các điểm nằm gần tâm tối ưu.*

Corollary 3.2. *Cho*

$$R_2 = O\left(\frac{\log(m^3 d \log^3(m\Delta^2)/\epsilon_1^2)\log(m\Delta^2)}{\alpha\epsilon_1^4}\right)$$

Với một điểm dữ liệu bất kỳ $u \in U_i$, với xác suất cao, bộ ước lượng $\omega(u)$ thỏa mãn:

$$\frac{\delta^2(P_i \setminus \mathcal{Z}^\dagger(u), u)}{1+7\epsilon_1} \leq \omega(u) \leq (1+\epsilon_1)^2\delta^2(H_i(u), u)$$

trong đó $H_i(u)$ là tập hợp $(1-\alpha)m_i$ điểm gần u nhất trong P_i, và $\mathcal{Z}^\dagger(u)$ là tập hợp $(2+20\epsilon_1)\alpha m_i$ điểm xa u nhất.

Lemma 12. *Khoảng cách giữa trọng tâm của tập hợp đã lọc $\overline{I}_i$ và tâm tối ưu c_i^* bị chặn như sau:*

$$\delta^2(\overline{I}_i, c_i^*) \leq \frac{9\delta^2(P_i^*, c_i^*)}{(1-(3+\epsilon)\alpha)m_i}$$

Lemma 13. *Chi phí phân cụm của tập Q_i đối với tâm của tập hợp đã lọc $\overline{I}_i$ thỏa mãn chặn cụ thể sau:*

$$\delta^2(Q_i, \overline{I}_i) \leq \delta^2(Q_i, c_i^*) + \left(5\sqrt{\alpha} + \frac{36(\sqrt{\alpha}+\alpha)}{1-(3+\epsilon)\alpha}\right)\delta^2(P_i^*, c_i^*)$$

Lemma 14. *Tổng chi phí phân cụm của cụm tối ưu P_i^* đối với trọng tâm $\overline{I}_i$ bị chặn bởi:*

$$\delta^2(P_i^*, \overline{I}_i) \leq \left(1 + 6\sqrt{\alpha} + \frac{36(\sqrt{\alpha}+\alpha)}{1-(3+\epsilon)\alpha} + \frac{9(\sqrt{\alpha}+\alpha)}{(1-\alpha)(1-(3+\epsilon)\alpha)}\right)\delta^2(P_i^*, c_i^*)$$

7 Fast-Sampling (k-median)

Trong mục này, tác giả trình bày cách mở rộng các phương pháp dựa trên lấy mẫu được đề xuất cho bài toán k-median có hỗ trợ học. Thách thức chính ở đây nảy sinh từ sự khác biệt trong các mục tiêu tối ưu hóa. Cụ thể, đối với một tập hợp tọa độ $S \subset \mathbb{R}^d$ bất kỳ, tâm hình học của S không còn đóng vai trò là tâm phân cụm tối ưu cho S theo mục tiêu k-median, khiến việc xác định các tọa độ hoặc tâm ứng viên chất lượng cao trở nên khó khăn. Kết quả là, các thuật toán k-median có hỗ trợ học hiện có thường gặp khó khăn trong việc đạt được các đảm bảo xấp xỉ chất lượng cao.

Để vượt qua thách thức này, mục tiêu của tác giả là sử dụng các chiến lược dựa trên lấy mẫu để xây dựng một tập U_i các tâm nằm gần các tâm phân cụm tối ưu cho mỗi cụm dự đoán P_i. Sau đó, bằng cách rời rạc hóa lưới, tác giả có thể tạo ra các tâm ứng viên có khả năng xấp xỉ tốt các tâm phân cụm tối ưu. Cuối cùng, bằng cách liệt kê các tâm ứng viên đã xây dựng, tác giả chứng minh rằng chi phí phân cụm của mỗi cụm tối ưu có thể được xấp xỉ tốt bằng cách sử dụng tâm tốt nhất được chọn từ quá trình liệt kê.

Bảng 7 cung cấp một so sánh chi tiết các kết quả cho bài toán k-median có hỗ trợ học. Tác giả cũng đưa ra một biểu đồ (Hình 2) về tỷ lệ xấp xỉ so với tỷ lệ lỗi α. Có thể thấy từ bảng rằng kết quả tốt nhất hiện nay đạt được xấp xỉ $(1+O(\alpha))$ với $\alpha \in [0, 1/2)$ [13]. So với các kết quả tiên tiến nhất, thuật toán Fast-Sampling có thể đạt được các đảm bảo chất lượng phân cụm tốt hơn với thời gian chạy kém hơn một chút đối với số chiều d cố định.

Bảng 2: Kết quả so sánh các thuật toán k-median có hỗ trợ học

Phương pháp và Tài liệu tham khảo	**Tỷ lệ xấp xỉ**	**Khoảng lỗi nhãn** α	**Độ phức tạp thời gian**
Phân vùng và Sắp xếp [6]	$1+\tilde{O}((k\alpha)^{1/4})$	Hằng số nhỏ	$O(md\log^3 m+\text{poly}(k,\log m))$
Sắp xếp [13]	$1+\frac{\alpha(7+10\alpha-10\alpha^2)}{(1-\alpha)(1-2\alpha)}$	$[0,1/2)$	$O\left(\frac{md\log^3 m\log^2(k/\delta)}{1-2\alpha}\right)$
Fast-Sampling (Tác giả)	$1+\frac{\alpha(6+4\epsilon-4\alpha-3\epsilon\alpha)}{(1-\alpha)(1-2\alpha)}$	$(0,1/2)$	$O\left(\frac{md\log(kd)\log(m\Delta)}{1-2\alpha}\cdot\left(\frac{\sqrt{d}}{\epsilon\alpha}\right)^{O(d)}\right)$

Mô tả cụ thể cho thuật toán k-median có hỗ trợ học được trình bày trong Thuật toán 5. Ý tưởng chung đằng sau thuật toán là trước tiên tạo ra các tâm ứng viên có thể xấp xỉ chặt chẽ các tâm phân cụm tối ưu cho mỗi cụm dự đoán. Sau đó, bằng cách chọn tâm tốt nhất với chi phí k-median nhỏ nhất, tác giả chứng minh rằng thuật toán đề xuất có thể đưa ra các đảm bảo xấp xỉ tốt hơn cho bài toán k-median có hỗ trợ học. Dưới đây, tác giả đưa ra phân tích cụ thể cho thuật toán được đề xuất.

Thuật toán 4 Fast-Sampling (k-median)

Đầu vào: Bài toán k-median (P,k,d), tập các phân vùng $(P_1,\ldots,P_k)$ với tỷ lệ lỗi α, tham số $\epsilon\in(0,1]$.

Đầu ra: Tập $C\subset\mathbb{R}^d$ gồm k tâm sao cho $|C|=k$.

1: **for** mỗi $i\in[k]$ **do**
2: Lấy mẫu ngẫu nhiên và độc lập tập U_i từ P_i với kích thước $O\left(\frac{\log(kd)}{1-2\alpha}\right)$, sau đó khởi tạo $U_i'=\emptyset$.
3: **for** $q=0$ đến $O(\log(m\Delta))$ **do**
4: $l_i=2^{q-1}/(1-\alpha)m_i$.
5: **for** mỗi $u\in U_i$ **do**
6: Gọi $G(u)$ là lưới tâm u với độ dài cạnh $2l_i$.
7: Phân rã $G(u)$ thành các lưới con nhỏ hơn với độ dài cạnh $(1-\alpha)\alpha\epsilon_1 l_i/\sqrt{d}$, và gọi $s(u)$ là tập các tâm của các lưới con này, với $\epsilon_1<\epsilon/4$.
8: $U_i'=U_i'\cup s(u)$.
9: $u_i=\arg\min_{u\in U_i'}\delta(\mathcal{N}_i(u),u)$, trong đó $\mathcal{N}_i(u)$ là tập $(1-\alpha)m_i$ điểm trong P_i gần u nhất.
10: $\hat{c}_i=u_i$.
11: **return** $\{\hat{c}_1,\hat{c}_2,\ldots,\hat{c}_k\}$.

Không mất tính tổng quát, tác giả có thể giả định rằng khoảng cách từng đôi tối thiểu giữa các điểm dữ liệu trong P là 1 trong khi khoảng cách từng đôi tối đa là Δ. Lưu ý rằng điều này có thể thực hiện được bằng các kỹ thuật tỉ lệ chuẩn. Theo Bổ đề 4, trong mỗi bước 2 của Thuật toán 5, với xác suất ít nhất $1-1/k$, có thể tìm thấy ít nhất một tâm $u\in U_i$ sao cho $\delta(u,c_i^*)\le 2\delta(P_i^*,c_i^*)/|P_i^*|\le\frac{2\delta(P_i^*,c_i^*)}{(1-\alpha)m_i}$, trong đó bước cuối cùng tuân theo thực tế là $|P_i^*|\ge|Q_i|\ge(1-\alpha)m_i$. Sau đó, trong bước 3 của Thuật toán 5, vì thuật toán liệt kê tất cả các giá trị có thể giữa 1 và $\log(m\Delta)$, tồn tại ít nhất một dự đoán cho bán kính phân cụm (bước 4 của Thuật toán 5) sao cho $\delta(P_i^*,c_i^*)/(1-\alpha)m_i\le l_i\le 2\delta(P_i^*,c_i^*)/(1-\alpha)m_i$. Do đó, trong bước 6 của Thuật toán 5, lưới có tâm

tại u với độ dài cạnh $2l_i$ ($G(u)$) sẽ chứa tâm phân cụm tối ưu c_i^*. Sau đó, trong bước 7 của Thuật toán 5, bằng cách phân rã lưới $G(u)$ thành các lưới con nhỏ hơn với độ dài cạnh $(1-\alpha)\alpha\epsilon_1 l_i/\sqrt{d}$ cho một số $\epsilon_1 < \epsilon/4$, tâm phân cụm tối ưu c_i^* cũng phải thuộc về một trong các lưới con. Vì lưới con có độ dài cạnh $(1-\alpha)\alpha\epsilon_1 l_i/\sqrt{d}$, cũng tồn tại ít nhất một $u' \in U_i'$ sao cho u' đủ gần với c_i^*, tức là $\delta(u', c_i^*) \leq (1-\alpha)\alpha\epsilon_1 l_i \leq \alpha\epsilon\delta(P_i^*, c_i^*)/m_i$. Gọi u_i là điểm được chọn trong bước 9 của Thuật toán 5. Đối với bất kỳ điểm dữ liệu nào $u \in U_i'$, gọi $N_i(u)$ là tập hợp các điểm gần nhất $(1-\alpha)m_i$ trong P_i tới u. Do đó, ta có

$$\begin{aligned}
\delta(\mathcal{N}_i(u_i), u_i) &\leq \delta(\mathcal{N}_i(u'), u') \\
&\leq \delta(\mathcal{N}_i(u'), c_i^*) + |\mathcal{N}_i(u')|\delta(u', c_i^*) \\
&\leq \delta(\mathcal{N}_i(u_i), c_i^*) + m_i \cdot \left(\frac{\alpha\epsilon\delta(P_i^*, c_i^*)}{m_i}\right) \\
&\leq \delta(\mathcal{N}_i(u_i), c_i^*) + \alpha\epsilon\delta(P_i^*, c_i^*)
\end{aligned}$$

Trong đó:

- Bất đẳng thức đầu tiên tuân theo tính tối tiểu của u_i trong tập U_i'.
- Bất đẳng thức thứ hai áp dụng bất đẳng thức tam giác cho từng điểm trong $\mathcal{N}_i(u')$.
- Bước cuối cùng sử dụng giới hạn $|\mathcal{N}_i(u_i)| \leq m_i$ và khoảng cách tâm $\delta(u', c_i^*)$ đã thiết lập.

Corollary 3.3. *Đối với một cụm dự đoán P_i, với xác suất ít nhất $1 - 1/k$, tâm u_i được chọn bởi thuật toán Fast-Sampling cho mục tiêu k-median thỏa mãn:*

$$\delta(\mathcal{N}_i(u_i), u_i) \leq \delta(\mathcal{N}_i(u_i), c_i^*) + \alpha\epsilon\delta(P_i^*, c_i^*)$$

trong đó $\mathcal{N}_i(u_i)$ là tập hợp $(1-\alpha)m_i$ điểm gần nhất trong P_i đến u_i, và c_i^* là tâm phân cụm tối ưu cho cụm thứ i.

Lemma 15. *Đối với hàm mục tiêu k-median, khoảng cách giữa tâm thuật toán u_i và tâm phân cụm tối ưu c_i^* thỏa mãn:*

$$\delta(u_i, c_i^*) \leq \frac{(2+\alpha\epsilon)\delta(P_i^*, c_i^*)}{(1-2\alpha)m_i}$$

Lemma 16. *Đối với hàm mục tiêu k-median, chi phí phân cụm của tập Q_i đối với tâm được chọn u_i thỏa mãn chặn sau:*

$$\delta(Q_i, u_i) \leq \delta(Q_i, c_i^*) + \frac{\alpha(4+3\epsilon)}{1-2\alpha}\delta(P_i^*, c_i^*)$$

trong đó c_i^* là tâm tối ưu của cụm thứ i.

Lemma 17. *Với mỗi cụm $i \in [k]$, với xác suất ít nhất $1 - 1/k$, chi phí phân cụm k-median của*

cụm tối ưu P_i^* đối với tâm thuật toán u_i bị chặn bởi:

$$\delta(P_i^*, u_i) \leq \left(1 + \frac{6\alpha + 4\alpha\epsilon - 4\alpha^2 - 3\epsilon\alpha^2}{(1-\alpha)(1-2\alpha)}\right) \delta(P_i^*, c_i^*)$$

trong đó c_i^* là tâm tối ưu của cụm thứ i.

Tài liệu

[1] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. In *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 1027–1035, 2007.

[2] Maria-Florina Balcan and Avrim Blum. Clustering with interactive feedback. In *International Conference on Algorithmic Learning Theory*, pages 316–328. Springer, 2008.

[3] Vincent Cohen-Addad, Hossein Esfandiari, Vahab Mirrokni, and Shyam Narayanan. Improved approximations for euclidean k-means and k-median, via nested quasi-independent sets. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, pages 1621–1628, 2022.

[4] Vincent Cohen-Addad and C. S. Karthik. Inapproximability of clustering in lp metrics. In *Proceedings of the 60th IEEE Annual Symposium on Foundations of Computer Science*, pages 519–539, 2019.

[5] Sanjoy Dasgupta. The hardness of k-means clustering. In *Technical Report*, 2008.

[6] Jon C. Ergun, Zhili Feng, Sandeep Silwal, David Woodruff, and Samson Zhou. Learning-augmented k-means clustering. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[7] Zachary Friggstad, Mohsen Rezapour, and Mohammad R. Salavatipour. Local search yields a ptas for k-means in doubling metrics. *SIAM Journal on Computing*, 48(2):452–480, 2019.

[8] Buddhima Gamlath, Silvio Lattanzi, Ashkan Norouzi-Fard, and Ola Svensson. Approximate cluster recovery from noisy labels. In *Proceedings of the 35th Conference on Learning Theory*, pages 1463–1509, 2022.

[9] Ragesh Jaiswal, Amit Kumar, and Sandeep Sen. A simple d^2-sampling based ptas for k-means and other clustering problems. *Algorithmica*, 70(1):22–46, 2014.

[10] Silvio Lattanzi and Christian Sohler. A better k-means++ algorithm via local search. In *Proceedings of the 36th International Conference on Machine Learning*, pages 3662–3671, 2019.

[11] Stuart Lloyd. Least squares quantization in pcm. *IEEE Transactions on Information Theory*, 28(2):129–137, 1982.

[12] Michael Mitzenmacher and Sergei Vassilvitskii. Algorithms with predictions. *Communications of the ACM*, 65(7):33–35, 2022.

[13] Thy Dinh Nguyen, Anamay Chaturvedi, and Huy Nguyen. Improved learning-augmented algorithms for k-means and k-medians clustering. In *Proceedings of the 10th International Conference on Learning Representations*, 2022.

[14] Sharad Vikram and Sanjoy Dasgupta. Interactive bayesian hierarchical clustering. In *International Conference on Machine Learning*, pages 2081–2090. PMLR, 2016.