BỘ GIÁO DỤC
VÀ ĐÀO TẠO

VIỆN HÀN LÂM KHOA HỌC
VÀ CÔNG NGHỆ VIỆT NAM

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ

Bùi Quốc

ỨNG DỤNG CỦA THUẬT TOÁN K-MEANS VÀO BÀI TOÁN PHÂN CỤM CỦA MẠNG LỚN

LUẬN VĂN THẠC SĨ TOÁN ỨNG DỤNG

HàNội - 2024

BỘ GIÁO DỤC
VÀ ĐÀO TẠO

VIỆN HÀN LÂM KHOA HỌC
VÀ CÔNG NGHỆ VIỆT NAM

HỌC VIỆN KHOA HỌC VÀ CÔNG NGHỆ

Bùi Quốc

ỨNG DỤNG CỦA THUẬT TOÁN K-MEANS VÀO BÀI TOÁN PHÂN CỤM CỦA MẠNG LỚN

LUẬN VĂN THẠC SĨ TOÁN ỨNG DỤNG

Mãsố: 8460112

NGƯỜI HƯỚNG DẪN KHOA HỌC:

TS. Đỗ Duy Hiếu

HàNội– 2024

LỜI CAM ĐOAN

Tôi cam đoan rằng luận văn này là kết quả của quá trình tìm hiểu, học hỏi và phát triển kiến thức của bản thân dưới sự hướng dẫn chuyên nghiệp của thầy Đỗ Duy Hiếu. Tất cả các thông tin và ý tưởng được trích dẫn từ các tác giả khác đều được nêu rõ nguồn gốc. Tôi hoàn toàn chịu trách nhiệm về những lời cam đoan này.

Hà Nội, tháng 5 năm 2024

Học viên

Bùi Quốc

LỜI CẢM ƠN

Trước hết, tôi muốn bày tỏ lòng biết ơn sâu sắc đến thầy Đỗ Duy Hiếu, người đã dành thời gian và công sức để hướng dẫn và hỗ trợ tôi trong việc chọn đề tài và xác định hướng nghiên cứu cho luận văn của mình. Thầy không chỉ là một người hướng dẫn khoa học tận tâm, mà còn là người đem đến những lời khuyên và sự động viên, khích lệ giúp tôi phát triển trong cả khía cạnh cá nhân và học thuật.

Trong thời gian học tại Viện Toán học, tôi rất biết ơn sự quan tâm, góp ý và hỗ trợ quý báu từ các giáo viên, đồng nghiệp và bạn bè. Tôi muốn gửi lời cảm ơn chân thành đến tất cả họ.

Tôi cũng muốn bày tỏ lòng biết ơn đến Viện Toán học và Học viện Khoa học và Công nghệ, Viện Hàn lâm Khoa học và Công nghệ Việt Nam, đã tạo điều kiện thuận lợi và cung cấp môi trường học tập cho tôi trong quá trình thực hiện luận văn này.

Cuối cùng, tôi muốn bày tỏ lòng biết ơn vô hạn đến gia đình đã luôn kiên nhẫn và yêu thương tôi không điều kiện.

Hà Nội, tháng 5 năm 2024

Học viên

Bùi Quốc

Mục lục

Lời cam đoan **i**

Lời cảm ơn **ii**

Mục lục **v**

Danh mục ký hiệu **vi**

Mở đầu **vi**

1 Kiến thức chuẩn bị **3**

1.1 Một số kiến thức lý thuyết đồ thị 3

1.2 Sơ lược về bước đi ngẫu nhiên trên đồ thị 7

1.3 Sơ lược về bài toán tìm kiếm cộng đồng mạng 8

1.3.1 Khoa học mạng 8

1.3.2 Mạng lớn và tìm kiếm cộng đồng mạng trong mạng lớn 9

1.3.3 Mạng lớn và cấu trúc cộng đồng 9

2 Một số phương pháp tọa độ hóa các đỉnh trong đồ thị **11**

2.1 Một số phương pháp tọa độ hóa đồ thị vô hướng 11

2.1.1 Các phương pháp tọa độ hóa dựa theo các thuật toán giảm số chiều 11

2.1.2 Phương pháp tọa độ hóa đồ thị vô hướng sử dụng bước đi ngẫu nhiên 23
2.2 Một số phương pháp tọa độ hóa đồ thị có hướng 23
2.2.1 Phương pháp tọa độ hóa phổ 23
2.2.2 Phương pháp tọa độ hóa đồ thị có hướng sử dụng bước đi ngẫu nhiên 25

3 Thuật toán *K*-Means, *K*-Means++, *K*-Means‖ 27
3.1 Thuật toán *K*-Means 27
3.1.1 Giới thiệu 27
3.1.2 Mô tả thuật toán *K*-Means 28
3.1.3 Cơ sở toán học 30
3.2 Thuật toán *K*-Means++ và *K*-Means‖ 33
3.2.1 Thuật toán *K*-Means++ 33
3.2.2 Thuật toán *K*-Mean ‖ 35
3.3 Một số thí nghiệm của Thuật toán *K*-Means++ và *K*-Mean‖ 36
3.3.1 So sánh giữa *K*-Means và *K*-Means++ 36
3.3.2 So sánh Thuật toán *K*-Means‖ và một số thuật toán khác 39

4 Một số thuật toán *K*-Means sử dụng hàm cosin 41
4.1 Độ tương đồng giữa các đỉnh sử dụng hàm cosin trong thị vô hướng 41
4.2 Độ tương đồng giữa các đỉnh sử dụng hàm cosin trong đồ thị có hướng 42
4.3 Một số thuật toán *K*-Means cosin 43
4.4 Một số thí nghiệm 46

4.4.1 Các mô hình đồ thị ngẫu nhiên và các tiêu chí đánh giá 46
4.4.2 Thí nghiệm trên đồ thị sinh ngẫu nhiên 49
4.4.3 Thí nghiệm trên dữ liệu thực 57
4.4.4 Nhận xét về các thí nghiệm 62

Kết luận **64**

Tài liệu tham khảo **66**

Danh mục ký hiệu

$\mathbb{R}$	Tập số thực.
$diag(d_1, d_2, ..., d_n)$	Ma trận đường chéo cỡ $n \times n$.
$A, A(G)$	Ma trận kề của đồ thị G.
$tr(M)$	Vết của ma trận M.
M^T	Ma trận chuyển vị của ma trận M.
I_k	Ma trận đơn vị cỡ $k \times k$.
L_{sym}	Ma trận Laplace chuẩn hóa.
P	Ma trận chuyển của bước đi ngẫu nhiễn trên đồ thị.
$[x_1, x_2, ..., x_n]$	Ma trận với các cột là các vector x_i.
Γ	Ma trận Diplacian.
Φ	Trạng thái dừng của bước đi ngẫu nhiên.
$C(i)$	Tọa độ của đỉnh i.
$\| \bullet \|_2$	Chuẩn Euclid.
L	Hàm mất mát của thuật toán K-Means.
$\frac{\partial f(x)}{\partial x_i}$	Đạo hàm riêng của hàm f theo biến x_i.
$g(x) = \theta(f(x))$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} = c$ với c dương, hữu hạn.
$g(x) = \mathcal{O}(f(x))$	Tồn tại $M > 0$ sao cho $0 \leq f(x) \leq Mg(x)$ với x đủ lớn.
$g(x) = \Omega(f(x))$	$\lim_{x\to\infty} \frac{f(x)}{g(x)} = 0$.
$\| \bullet \|_F$	Chuẩn Frobenius.

LỜI MỞ ĐẦU

Nghiên cứu về phát hiện cộng đồng trong mạng là một lĩnh vực quan trọng trong khoa học mạng, với nhiều ứng dụng đa dạng trong khoa học máy tính và các lĩnh vực khoa học khác [1, 2, 3]. Do đó, các nhà nghiên cứu đã tiến hành nhiều nỗ lực nghiên cứu, sử dụng nhiều phương pháp khác nhau như thuật toán *K*-means, các thuật toán dựa trên modularity, thuật toán Louvain, và các thuật toán sử dụng bước đi ngẫu nhiên. Trong số này, thuật toán cổ điển và nổi tiếng nhất là thuật toán *K*-means.

Thuật toán *K*-means thường được áp dụng cho bài toán phân cụm với một tập dữ liệu gồm các vector. Tuy nhiên, cũng có thể áp dụng cho bài toán phân cụm với một mạng (đồ thị) bằng cách gắn mỗi đỉnh của mạng với một vector trong không gian $\mathbb{R}^d$. Do đó, việc nghiên cứu thuật toán *K*-means để tìm kiếm cộng đồng mạng tương đương với việc nghiên cứu cách biểu diễn các đỉnh của mạng trong không gian vector. Trong Chương 2 của luận văn này, chúng tôi sẽ trình bày một số phương pháp biểu diễn các đỉnh của mạng.

Mặc dù thuật toán *K*-means là hiệu quả, việc chọn ngẫu nhiên các điểm khởi đầu có thể dẫn đến kết quả phân cụm không chính xác hoặc cần nhiều vòng lặp để hội tụ. Vì vậy, việc chọn một phương pháp khởi tạo tốt hơn là một vấn đề quan trọng. Trong Chương 3, chúng tôi sẽ trình bày hai phiên bản cải tiến của *K*-means là *K*-means++ và *K*-means $\|$ với phương pháp khởi tạo tâm ban đầu tốt hơn. Hơn nữa, trong Chương 4,

chúng tôi đề xuất ba thuật toán khởi tạo tâm ban đầu tốt hơn cho mạng.

Mục đích, đối tượng và phạm vi nghiên cứu:

- Đối tượng nghiên cứu: đồ thị lớn.

- Phạm vi nghiên cứu: Định nghĩa tính chất về đồ thị, thuật toán *K*-means và các biến thể của nó.

Phương pháp nghiên cứu:

- Đọc hiểu và trình bày hệ thống các kiến thức liên quan đến đề tài luận văn từ các tài liệu tham khảo chuyên ngành.

- Sử dụng phương pháp tọa độ hóa các đỉnh trên đồ thị thông qua bước đi ngẫu nhiên.

- Sử dụng lập trình Python để dự đoán, đánh giá kết quả của thuật toán mà chúng tôi đề xuất.

Cấu trúc và dự kiến kết quả đạt được của luận văn

Ngoài phần Lời mở đầu, Lời cảm ơn, Lời cam đoan, Kết luận và Tài liệu tham khảo, Luận văn được chia thành bốn chương.

- Chương 1: Kiến thức chuẩn bị.
- Chương 2: Phương pháp tọa độ hóa các đỉnh trong đồ thị.
- Chương 3: Thuật toán *K*-means, *K*-means++, *K*-means$\|$.
- Chương 4: Một số thuật toán *K*-means sử dụng hàm cosin.

Chương 1

Kiến thức chuẩn bị

1.1. Một số kiến thức lý thuyết đồ thị

Dưới đây, chúng tôi sẽ giới thiệu các kiến thức cơ bản về lý thuyết đồ thị được áp dụng cho các nội dung trong phần tiếp theo. Tài liệu tham khảo chính của phần này là trong [4].

Định nghĩa 1.1.1. *Đồ thị vô hướng G là một cặp $G = (V, E)$, trong đó V là tập hợp các đỉnh và E là một họ gồm các tập con có hai phần tử của V được gọi là một cạnh. Hai đỉnh x và y trong V được gọi là kề nhau nếu $\{x, y\} \in E$, khi đó x và y cũng được gọi là kề với cạnh $\{x, y\}$.*

Ta cũng có định nghĩa đồ thị có hướng như sau:

Định nghĩa 1.1.2. *Đồ thị có hướng G là một cặp có thứ tự $G = (V, E)$, ở đây V là một tập hợp các đỉnh, còn E là tập các cặp có thứ tự chứa các đỉnh phân biệt, được gọi là cung. Cụ thể hơn nếu $(a, b) \in E$ thì (a, b) là cung của G với đỉnh đầu là a, đỉnh cuối là b.*

Định nghĩa 1.1.3. *Trong đồ thị vô hướng $G = (V, E)$, một hành trình là một dãy các đỉnh $v_0v_1v_2...v_n$ sao cho mỗi cặp $\{v_i, v_{i+1}\}$ là một cạnh của G với mọi $i = 0, 1, ..., n-1$. Các cạnh $\{v_i, v_{i+1}\}, i = 0, 1, ..., n-1$ cũng được gọi là các cạnh của hành trình $v_0v_1...v_n$. Trong đó, n được gọi là độ dài, v_0 là đỉnh đầu, v_n*

là đỉnh cuối của hành trình. Một hành trình được coi là khép kín nếu đỉnh đầu và đỉnh cuối của nó trùng nhau.

Một hành trình được gọi là đường nếu các đỉnh trên đó đôi một khác nhau. Một hành trình khép kín được gọi là chu trình nếu nó có độ dài ít nhất là 3 và khi xoá đi đỉnh cuối thì trở thành đường.

Nhận xét 1.1.1. *Các định nghĩa trong mục này dựa chủ yếu vào tài liệu [4], trong các tài liệu khác các khái niệm về đồ thị trong mục này có thể được định nghĩa khác trong các tài liệu khác nhau.*

Định nghĩa 1.1.4. *Giả sử $G = (V, E)$ là một đồ thị vô hướng và $v \in V$. Ký hiệu*

$$N_G(v) := \{x \in V \mid x \neq v \text{ và } \{x, v\} \in E\}.$$

Khi đó $N_G(v)$ được gọi là tập các láng giềng của v. Trong trường hợp đồ thị G được hiểu ngầm, ta ký hiệu $N_G(v)$ đơn giản bằng $N(v)$.

Ta định nghĩa bậc của đỉnh v trong đồ thị G, ký hiệu là $deg_G(v)$ hay ngắn gọn là $\deg(v)$ nếu như G được hiểu ngầm, là số cạnh kề với đỉnh v. Trong luận văn này, ta sẽ xét

Khi đó, chúng ta định nghĩa ma trận đường chéo D như sau:

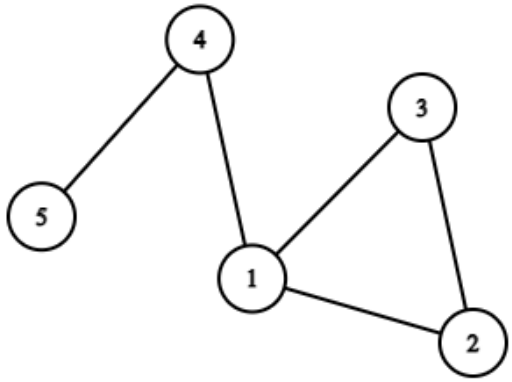
$$D := \operatorname{diag}(d_1, d_2, \ldots, d_n),$$

trong đó d_i là bậc của đỉnh v_i.

Định nghĩa 1.1.5. *Giả sử $G = (V, E)$ là một đồ thị có hướng và $v \in V$. Ta ký hiệu d_i^- và d_i^+ là số cạnh đi vào và đi ra đỉnh v_i.*

Khi đó $|N^+(v)|$ được gọi là bậc đi ra, còn $|N^-(v)|$ được gọi là bậc đi vào của v.

Khi đó hai ma trận chéo tương ứng với D trong trường hợp có hướng:



Hình 1.1: Đồ thị vô hướng G với 5 đỉnh

1. $D^{in} := diag(d_1^-, ..., d_n^-)$
2. $D^{out} := diag(d_1^+, ..., d_n^+)$

Định nghĩa 1.1.6. *Một đồ thị vô hướng $G = (V, E)$ được gọi là liên thông nếu với mọi đỉnh v_i, v_j phân biệt trong V, luôn tồn tại một hành trình có hướng từ v_i đến v_j.*

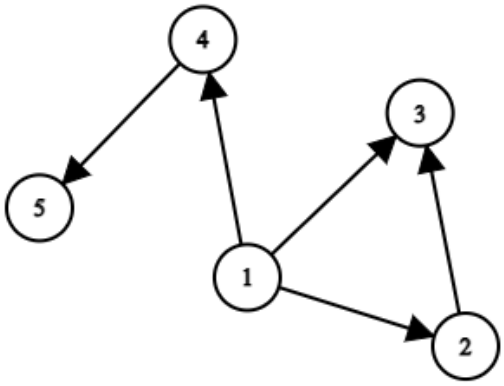
Định nghĩa 1.1.7. *Cho một đồ thị vô hướng (có hướng) $G = (V, E)$, ma trận kề của đồ thị G, ký hiệu là $A(G)$ hoặc A khi đã rõ G được định nghĩa như sau:*

$$A_{ij} := \begin{cases} 1 \text{ nếu} \{i, j\} \in E(\text{ tương ứng } (i, j) \in E); \\ 0 \text{ trong trường hợp còn lại.} \end{cases} \tag{1.1.1}$$

Nhận xét 1.1.2. *Nếu $G = (V, E)$ là đồ thị vô hướng thì ma trận kề của nó A là ma trận đối xứng. Điều này nói chung không còn đúng trong trường hợp có hướng.*

Trong luận văn này, với tập V có n đỉnh ta sẽ đồng nhất tập đỉnh V với tập n số tự nhiên khác không đầu tiên $\{1, 2, ..., n\}$.

Ví dụ 1.1.1. *Chúng ta xét đồ thị vô hướng trong Hình 1.1: Ta có ma trận kề*



Hình 1.2: Đồ thị có hướng G' với 5 đỉnh

$A(G)$ của đồ thị trên là

$$A(G) = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}.$$

Mâ trận $A(G)$ ở trên là ma trận đối xứng, hàng và cột thứ i tương ứng với đỉnh thứ i. Ví dụ đỉnh 1 *kề với đỉnh* 2, 3 *và* 4

Ví dụ 1.1.2. *Chúng ta xét đồ thị có hướng trong Hình 1.2: Tiếp theo ta xét ma trận kề:*

$$A(G') = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Định nghĩa 1.1.8. *Cho đồ thị $G = (V, E)$ vô hướng, ta định nghĩa ma trận*

Laplace như sau:

$$L := D - A.$$

Trong trường hợp đồ thị có hướng ta có thể định nghĩa ma trận Laplace bằng cách thay ma trận D bởi D^{out}. Trong luận văn này, nếu không nói gì thêm, ta sẽ luôn hiểu D là D^{out} khi đồ thị đang xét là đồ thị có hướng

Tiếp theo, chúng ta sẽ nhắc lại một số tính chất của ma trận kề A và ma trận Laplace L:

Mệnh đề 1.1.1 ([5])**.** *Cho G là một đồ thị vô hướng, với ma trận Laplace $L = D - A$ được định nghĩa như ở trên, ma trận Laplace có các tính chất sau:*

- *Ma trận L đối xứng.*
- *Ma trận L là ma trận nửa xác định dương.*
- *L có giá trị riêng nhỏ nhất bằng 0.*
- *Nếu G là đồ thị liên thông thì không gian con riêng ứng với giá trị riêng 0 là không gian một chiều sinh bởi vector $\mathbf{1}$.*
- *Trong trường hợp đồ thị G có k thành phần liên thông $A_1, A_2, ..., A_k$ thì bội của giá trị riêng 0 bằng số thành phần liên thông của G và không gian con riêng tương ứng có cơ sở $\{\mathbf{1}_{A_1}, ..., \mathbf{1}_{A_k}\}$. Trong đó A_i là các thành phần liên thông của G.*

1.2. Sơ lược về bước đi ngẫu nhiên trên đồ thị

Định nghĩa 1.2.1. *Gọi $G = (V, E)$ là một đồ thị có hướng được định nghĩa trên tập đỉnh V. Định nghĩa A là ma trận kề của G, có nghĩa là $a_{ij} = 1$ nếu có một cạnh hướng (hoặc cung) (i, j), và 0 trong trường hợp ngược lại. Với $i = 1, 2, ..., n$, ta nhắc lại bậc ra của đỉnh i, $d_i^+ = \sum_{j=1}^{n} a_{ij}$, và bậc vào của đỉnh i,*

$d_i^- = \sum_{j=1}^n a_{ji}$. Nói chung, d_i^+ không bằng d_i^-. Tuy nhiên, ta có $m = \sum_{i=1}^n d_i^+ = \sum_{i=1}^n d_i^-$, với m là số cạnh của G.

Chúng ta ý rằng, đồ thị vô hướng có thể được coi là một trường hợp đặc biệt của đồ thị có hướng, trong đó ma trận kề là đối xứng và bậc ra và bậc vào của mỗi đỉnh là bằng nhau. Để tổng quát hoá ma trận chuyển mà chúng tôi sẽ định nghĩa sau đây, chúng ta sẽ ký hiệu $d_i^+ = d_i$, và $D^{out} = D$.

Một bước đi ngẫu nhiên trên đồ thị là một quá trình bắt đầu từ một đỉnh cho trước và di chuyển đến một đỉnh khác ở mỗi bước. Đỉnh tiếp theo trong quá trình đi được chọn một với xác suất như nhau trong các đỉnh lân cận của đỉnh hiện tại. Do đó, tại mỗi bước, xác suất chuyển từ đỉnh i đến đỉnh j được cho bởi $P_{ij} = \frac{A_{ij}}{d(i)}$. Định nghĩa này thiết lập ma trận chuyển P cho quá trình đi ngẫu nhiên. Rõ ràng rằng $P = D^{-1}A$. Ma trận chuyển đổi P thỏa mãn $\lim_{k\to\infty} P^k = P_\infty$, trong đó $(P_\infty)_{ij} = \phi_j$, thành phần thứ j của phân phối dừng duy nhất $\phi = (\phi_1, \phi_2, ..., \phi_n)$. Lưu ý rằng $\phi_i = d(i) / \sum d(j)$ (xem trong [6]).

1.3. Sơ lược về bài toán tìm kiếm cộng đồng mạng

1.3.1. Khoa học mạng

Khoa học mạng đã phát triển thành một lĩnh vực nghiên cứu quan trọng. Vì thế đã có nhiều nghiên cứu về chủ đề này, như tài liệu [7, 8]. Trong phần này, chúng ta sẽ giới thiệu một số kiến thức cơ bản về khoa học mạng.

Khoa học mạng có nhiều định nghĩa, một định nghĩa được nhiều người quan tâm trong [9] : Khoa học mạng là nghiên cứu về cấu trúc và hoạt động của các mạng bằng cách sử dụng các công cụ và lý thuyết toán học. Nó tập trung vào việc phân tích và mô tả đặc điểm và trạng thái của các

mạng. Nghiên cứu về mạng đã đóng góp vào sự hiểu biết và đánh giá các đặc tính thống kê của các mạng quy mô lớn.

1.3.2. Mạng lớn và tìm kiếm cộng đồng mạng trong mạng lớn

1.3.3. Mạng lớn và cấu trúc cộng đồng

Các mạng lớn với hàng nghìn đến hàng triệu đỉnh phổ biến trong nhiều lĩnh vực khoa học khác nhau. Mạng lớn thường có cấu trúc cộng đồng, trong đó các đỉnh hoặc nhóm đỉnh có mối liên kết mạnh mẽ bên trong cộng đồng và yếu hơn với các đỉnh hoặc nhóm đỉnh ở bên ngoài. Việc phát hiện cộng đồng trong mạng lớn là thách thức do chi phí tính toán cao và cấu trúc không đồng nhất.

Mạng thường được mô hình hoá dưới dạng đồ thị, trong đó các đỉnh được liên kết với nhau qua các cạnh. Cấu trúc cộng đồng là một đặc điểm tồn tại tự nhiên trong nhiều loại mạng thực tế, từ mạng xã hội đến mạng giao thông, và có vai trò quan trọng trong việc hiểu về cách mạng hoạt động và tương tác.

Việc xác định cộng đồng không chỉ giúp chúng ta hiểu rõ hơn về cấu trúc và tính chất của mạng, mà còn hỗ trợ trong việc giải quyết các vấn đề thực tế. Tuy đã có nhiều phương pháp và thuật toán được phát triển để phân tích cấu trúc cộng đồng trong mạng lớn, nhưng vẫn chưa có giải pháp tổng quát phù hợp cho mọi loại mạng do sự đa dạng và phức tạp của chúng. Điều này đặt ra một thách thức đối với nghiên cứu và ứng dụng thực tế.

Tìm kiếm cộng đồng mạng

Tìm kiếm cộng đồng trên mạng xã hội là một nhiệm vụ quan trọng trong phân tích mạng xã hội. Với sự phát triển của công nghệ thông tin,

mạng xã hội ngày càng mở rộng với quy mô lớn. Tuy nhiên, các thuật toán hiện tại thường gặp khó khăn trong việc xử lý các mạng xã hội quy mô lớn, do độ phức tạp tính toán lớn.

Mục tiêu của bài toán tìm kiếm cộng đồng mạng là từ mạng ban đầu, tìm ra các cộng đồng tồn tại trong đó và hiểu về mối quan hệ bên trong và giữa các cộng đồng. Cụ thể, chúng ta muốn tìm nhóm các đỉnh có liên kết mạnh với nhau. Điều này có thể được hiểu là bài toán phân cụm các đỉnh của đồ thị.

Chương 2

Một số phương pháp tọa độ hóa các đỉnh trong đồ thị

Trong chương này, chúng tôi sẽ trình bày các phương pháp tọa độ hóa các đỉnh trên đồ thị vô hướng và có hướng. Nội dung của chương này chủ yếu dựa vào tài liệu [10] và [11].

2.1. Một số phương pháp tọa độ hóa đồ thị vô hướng

2.1.1. Các phương pháp tọa độ hóa dựa theo các thuật toán giảm số chiều

Trong phần này, chúng tôi sẽ trình bày 5 phương pháp tọa độ hóa các đỉnh trong đồ thị vô hướng, dựa vào tài liệu [10].
Nhìn chung phương pháp tọa độ hóa các đỉnh trong đồ thị thường bắt nguồn từ bài toán giảm số chiều của dữ liệu. Trong bài toán này, chúng ta sẽ được cho trước một tập dữ liệu $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \subset \mathbb{R}^d$ và mục tiêu của chúng ta là biểu diễn các điểm dữ liệu này vào không gian có số chiều p sao cho p nhỏ hơn rất nhiều so với d. Các phương pháp giảm số chiều thường bao gồm hai bước chính

- Đầu tiên, chúng ta biểu diễn các điểm dữ liệu ban đầu trong không gian $\mathbb{R}^d$ thành một đồ thị với các đỉnh tương ứng với các điểm dữ

liệu ban đầu (thường thì đồ thị ta nhận được sẽ là đồ thị vô hướng với trọng số của các cạnh không âm thể hiện quan hệ giữa các điểm dữ liệu ban đầu).

- Bước tiếp theo, chúng ta tương ứng các điểm của đồ thị xây dựng từ bước một thành các điểm trong không gian $\mathbb{R}^p$ với p nhỏ hơn rất nhiều so với d.

Trong mục này, chúng tôi chủ yếu quan tâm đến ứng dụng của bước hai trong bài toán giảm số chiều của dữ liệu với ứng dụng trong bài toán tọa độ hóa các đỉnh trong đồ thị. Đầu tiên, chúng tôi sẽ trình bày hai hướng áp dụng chung nhất đó là tọa độ hóa trực tiếp và tọa độ hóa tuyến tính.

Phương pháp tọa độ hóa trực tiếp: Đầu tiên với phương pháp tọa độ hóa trực tiếp, ta có đầu vào là một đồ thị vô hướng $G = (V, E)$ với ma trận kề A. Khi đó phương pháp tọa độ hóa trực tiếp sẽ tương đương với việc giải bài toán tối ưu sau:

$$\begin{cases} min \sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2; \\ Y^T B Y = I. \end{cases}$$

Trong đó B là một ma trận đối xứng, xác định dương thường được gọi là ma trận ràng buộc, tùy vào từng cách chọn ma trận ràng buộc B, ta sẽ có các thuật toán tọa độ hóa khác nhau.
Có thể thấy, hàm mục tiêu $\sum_{i=1}^{n} \sum_{j=1}^{n} A_{ij} \|\mathbf{y}_i - \mathbf{y}_j\|_2^2$ sẽ là tổng khoảng cách các đỉnh i và j mà i kề với j, về mặt trực giác, nếu Y là một phép tọa độ hóa tốt thì khoảng cách của các vector tương ứng với các đỉnh kề nhau sẽ phải nhỏ nhất có thể, vì thế ta có bài toán tối ưu ở trên.

Bổ đề 2.1.1. *[5] Cho $L = D - A$ là ma trận Laplace của đồ thị vô hướng G, ta*

có:

$$tr(X^T LX) = \frac{1}{2} \sum_{i,j \in V} A_{ij} \|\mathbf{X}_i - \mathbf{X}_j\|_2^2.$$

Chứng minh. Ta có:

$$tr(X^T LX) = \sum_{j=1}^{p} (\mathbf{X}^{(j)})^T L \mathbf{X}^{(j)},$$

trong đó, $\mathbf{X}^{(i)}$ là vector cột thứ i của ma trận X.
Với mỗi $j = 1, 2, ..., p$, ta có:

$$\begin{aligned}
(\mathbf{X}^{(j)})^T L \mathbf{X}^{(j)} &= \sum_{i=1}^{n} d_i X_{ij}^2 - \sum_{i,l=1}^{n} X_{ij} A_{il} X_{lj} \\
&= \sum_{i,l=1}^{n} A_{il} X_{ij} (X_{ij} - X_{lj}) \\
&= \frac{1}{2} \sum_{i,l} A_{il} (X_{ij} - X_{lj})^2.
\end{aligned}$$

Từ đẳng thức này, chúng ta lấy tổng với j chạy từ 1 đến p và suy ra điều phải chứng minh. □

Từ Mệnh đề 2.1.1, chúng ta có nhận xét rằng, để giải bài toán tối ưu trên, chúng ta chỉ cần tìm giá trị cực tiểu cho $tr(X^T LX)$. Theo bổ đề 2.1.1 thì bài toán tối ưu trên có thể được viết lại như sau:

$$\begin{cases} \min tr(Y^T LY); \\ Y^T BY = I. \end{cases}$$

Lagrangian của bài toán tối ưu này là:

$$\mathcal{L} = Y^T LY - tr(Y^T BY - I),$$

ta giải phương trình

$$\frac{\partial L}{\partial Y} = 0,$$

tương đương với

$$2LY - 2BY\Lambda = 0,$$

trong đó $\Lambda = diag(\lambda_1, ..., \lambda_n)$ với $_i$ là giá trị riêng của L theo thứ tự từ bé đến lớn. suy ra

$$LY = BY\Lambda.$$

Vậy nghiệm của bài toán tối ưu trên là các hàng của ma trận $Y \in \mathbb{R}^{n\times p}$ thoả mãn phương trình trên.

Phương pháp tọa độ hóa tuyến tính: Một cách tiếp cận khác của tọa độ hóa đồ thị là tọa độ hóa tuyến tính. Trong phương pháp này ta sẽ tìm một ma trận $U \in \mathbb{R}^{d\times p}$ và chiếu các điểm dữ liệu gốc lên không gian vector sinh bởi các cột của ma trận U, cụ thể ta phải giải bài toán tối ưu sau:

$$\begin{cases} \min tr(U^T XLX^T U), \\ U^T XBX^T U = I. \end{cases}$$

Nghiệm của bài toán tối ưu trên tương tự như trong trường hợp tọa độ hóa trực tiếp là ma trận $U^T X$ gồm các cột là p vector riêng của ma trận Laplace L ứng với các giá trị riêng của L.

Phương pháp tọa độ hóa sử dụng ánh xạ riêng

Chúng ta khởi đầu với một đồ thị vô hướng $G = (V, E)$ có n đỉnh và m cạnh. A là ma trận kề của G, và $D = \text{diag}(d)$ là ma trận chéo với các phần tử trên đường chéo là các bậc của các đỉnh. Giả sử thêm rằng G là một đồ thị liên thông. Mục tiêu chính của việc tạo ra các tọa độ là để tương ứng mỗi đỉnh của đồ thị thành một vector trong không gian $\mathbb{R}^p$ với p nhỏ hơn nhiều so với n.

Đối với mỗi đỉnh $i \in V$, chúng ta tương ứng nó thành một vector $\mathbf{x}_i \in \mathbb{R}^p$. Sau đó, chúng ta tạo ma trận $X \in \mathbb{R}^{n\times p}$ với mỗi hàng biểu diễn

một vector $\mathbf{X}_i^T$. Mục tiêu của chúng ta là xây dựng ma trận X sao cho nếu hai đỉnh i và j gần nhau, thì khoảng cách giữa hai vector $\mathbf{X}_i$ và $\mathbf{X}_j$ (được tính bằng $\|\mathbf{X}_i - \mathbf{X}_j\|_2$) là tương đối "nhỏ". Để làm rõ hơn ý tưởng này, chúng ta xét bài toán tối ưu sau:

$$\min_{X, X^T\mathbf{1}=0} \sum_{i,j\in V} A_{ij}\|\mathbf{X}_i - \mathbf{X}_j\|_2^2.$$

Tuy nhiên, bài toán này có thể có nghiệm tầm thường như $X = 0$. Để tránh điều này, chúng ta có bài toán tối ưu được sửa đổi như sau:

$$\min_{X, X^T\mathbf{1}=0, X^TX=I_k} \sum_{i,j\in V} A_{ij}\|\mathbf{X}_i - \mathbf{X}_j\|_2^2. \tag{2.1.1}$$

Trước hết, chúng ta cần mệnh đề sau; Để giải quyết vấn đề này, trước hết chúng ta cần định lý sau:

Định lí 2.1.1 ([10])**.** *Cho L là ma trận Laplace của đồ thị vô hướng G, ta có:*

$$\min_{X^T\mathbf{1}=0, X^TX=I_p} tr(X^TLX) = \sum_{j=2}^{p+1} \lambda_j, \tag{2.1.2}$$

trong đó, $\lambda_2 \leq ... \leq \lambda_{p+1}$ là giá trị riêng của ma trận L. Và giá trị nhỏ nhất đạt được khi X là ma trận với các vector cột là các giá trị riêng tương ứng với các vector riêng $\lambda_2, ..., \lambda_{p+1}$ của ma trận Laplace L.

Chứng minh. Đầu tiên, chúng ta xem xét bài toán tối ưu sau:

$$\min_{X, \text{diag}(X^TX)=I_p} \text{tr}(X^TLX).$$

Lagrangian của bài toán tối ưu này được định nghĩa như sau:

$$\mathcal{L} = \text{tr}\left(X^TLX - (X^TX - I_p)\Gamma\right),$$

trong đó Γ là ma trận đường chéo với p thành phần trên đường chéo là các nhân tử Lagrange tương ứng với điều kiện $\text{diag}(X^TX) = I_p$.
Cho gradient theo X bằng 0, ta có:

$$LX = X\Gamma,$$

do đó, X là ma trận gồm các vector riêng của L ứng với các giá trị riêng của của ma trận Laplace L. Bên cạnh đó, kết hợp với điều kiện trực giao của X, ta có $tr(X^TLX) = \text{tr}(\Gamma)$.

Với điều kiện $X^T\mathbf{1} = 0$, giá trị riêng đầu tiên bị loại bỏ và $tr(X^TLX)$ đạt giá trị nhỏ nhất khi bằng $tr(\Gamma) = \sum_{j=2}^{p+1} \lambda_j$. □

Từ đó, mỗi đỉnh i trong đồ thị G sẽ được biểu diễn bằng một vector $\mathbf{X}_i$, nằm ở hàng thứ i của ma trận $X = [\mathbf{X}_2, \mathbf{X}_3, ..., \mathbf{X}_{p+1}]$, với $\mathbf{X}_2, \mathbf{X}_3, ..., \mathbf{X}_{p+1}$ là các vector riêng tương ứng với các giá trị riêng $\lambda_2, ..., \lambda_{p+1}$ của ma trận Laplace L.

Ràng buộc của bài toán tối ưu (2.1.1) liên quan đến ma trận hiệp phương sai của vector ngẫu nhiên $\mathbf{X}_i \in \mathbb{R}^p$ với đỉnh i được lấy mẫu theo phân phối đều. Một ràng buộc tự nhiên khác được suy ra từ việc lấy mẫu cạnh. Từ đó chúng ta có bài toán tối ưu khác, như sau:

$$\min_{X: X^T\mathbf{d}=0, X^TDX=I_p} \sum_{i,j\in V} A_{ij} \left\|\mathbf{X}_i - \mathbf{X}_j\right\|^2.$$

Cả ràng buộc căn giữa $X^Td = 0$ và ràng buộc hiệp phương sai $X^TDX = I_p$ tương ứng với việc các đỉnh được chọn với xác suất tỉ lệ với bậc của chúng. Để giải bài toán tối ưu trên, trước hết ta cần định lý sau:

Định lí 2.1.2. *[5] Cho $L_{sym} := I - D^{-1/2}AD^{1/2}$. là ma trận Laplace chuẩn hóa, ta có:*

$$\min_{X, X^Td=0, X^TDX=I_p} tr(X^TL_{sym}X) = \sum_{j=2}^{p+1} \lambda_j. \tag{2.1.3}$$

Giá trị nhỏ nhất đạt được khi X là ma trận với các cột là các vector riêng tương ứng với các giá trị riêng $\lambda_2, ..., \lambda_{p+1}$ của ma trận L_{sym}.

Như vậy, kết hợp với mệnh đề 2.1.1, thì nghiệm của bài toán tối ưu trên là nghiệm của bài toán giá trị riêng sau:

$$LV = DV\Lambda, V^T DV = I, \tag{2.1.4}$$

trong đó $\Lambda = diag(\lambda_1, \lambda_2, ..., \lambda_n)$, và $0 = \lambda_1 \leq \lambda_2 \leq .... \leq \lambda_n$. Ta cũng có $V = D^{-1/2}U$ với U là ma trận trực chuẩn gồm các vector riêng của ma trận Laplace chuẩn hóa L_{sym}.

Định lý này được chứng minh hoàn toàn tương tự như chứng minh Định lý 2.1.1.

Từ Định lý 2.1.2, chúng ta cũng kết luận được rằng kết quả của phép tọa độ hóa là các hàng của ma trận $V = D^{-1/2}U$, với các cột của ma trận U là các vector riêng ứng với các giá trị riêng $\lambda_2, ...\lambda_p$ của ma trận L_{sym}.

Phương pháp tọa độ hóa của Thuật toán PCA

Sau đây tôi sẽ không trình bày cụ thể toàn bộ thuật toán PCA (Principle Component Analysis) mà sẽ chỉ tập trung vào phần sau của thuật toán nhằm phục vụ cho việc trình bày phương pháp tọa độ hóa dựa trên Thuật toán PCA. Trước hết ta sẽ trình bày sơ lược lại thuật toán PCA: Ta có tập dữ liệu đầu vào: $\{\mathbf{x}_i\}_{i=1}^n \subset \mathbb{R}^d$, và $\{\mathbf{y}_i\} \subset \mathbb{R}^p$ là kết quả nhận được sau khi thực hiện giảm số chiều của tập dữ liệu ban đầu. Mục đích chính của thuật toán PCA là chiếu các điểm dữ liệu lên một không gian vector p chiều với p nhỏ hơn nhiều so với d. Đầu tiên ta cần tìm một ma trận trực giao $U = [\mathbf{u}_1, \mathbf{u}_2, ..., \mathbf{u}_p] \in \mathbb{R}^{n \times p}$, hình chiếu của các vector $\mathbf{x}_i$ cho bởi: $UU^T\mathbf{x}_i$, đây là một vector trong không gian sinh bởi các vector cột của U. Mục tiêu của chúng ta là tìm U sao cho sai khác giữa $\mathbf{x}_i$ và $UU^T\mathbf{x}_i$ là bé

nhất có thể với mọi $i = 1, 2, ..., n$. Thật vậy ta cần cực tiểu hóa đại lượng sau:

$$\frac{1}{n}\sum_{i=1}^{n} \|\mathbf{x}_i - U^T U\mathbf{x}_i\|^2.$$

Để đơn giản cho tính toán ta giả sử trung bình của các điểm dữ liệu bằng 0, tức là $\frac{1}{n}\sum_{i=1}^{n} x_i = 0$. Ta có:

$$\frac{1}{n}\sum_{i=1}^{n} \|x_i - U^T U x_i\|^2 = \frac{1}{n}\sum_{i=1}^{n} \left(\|x_i\|^2 - \|U^T U x_i\|^2\right).$$

Để tìm cực tiểu của hàm trên, chúng ta chỉ cần tối đa hóa biểu thức $\|U^T U X_i\|^2$ cho mọi $i = 1, 2, ..., n$. Do đó, chúng ta cần tối đa hóa biểu thức sau:

$$\begin{aligned}
\|U^T U X\|_F^2 &= \text{tr}\left[\left(UU^T X\right)^T \left(UU^T X\right)\right] \\
&= \text{tr}\left(X^T UU^T UU^T X\right) \\
&= \text{tr}\left(X^T UU^T X\right) \\
&= \text{tr}\left(U^T XX^T U\right).
\end{aligned}$$

Từ đây ta có bài toán tối ưu sau:

$$\begin{cases} \max\limits_{U} tr\left(U^T XX^T U\right); \\ U^T U = I. \end{cases} \tag{2.1.5}$$

Ở đây, U là ma trận chiếu. Đặt $S = XX^T$, với $X = [\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n]$ trong đó $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ là các điểm dữ liệu ban đầu trong không gian $\mathbb{R}^d$. Ta thấy rằng phương pháp tọa độ hóa dựa theo thuật toán PCA là một trường hợp riêng của phương pháp tọa độ hóa tuyến tính đã trình bày ở phần đầu của phần 2.1.1 nếu thay B bởi I và $-L$ bởi I.

Đối với trường hợp đầu vào là một mạng (đồ thị), thì trong [10], các tác giả áp dụng thuật toán PCA đó bằng cách thay thế S bởi ma trận Laplace

L của đồ thị. Như vậy, khi đầu vào là một đồ thị thì chúng ta có bài toán tối ưu sau:

$$\begin{cases} \max\limits_{U} tr(U^T L U); \\ U^T U = I. \end{cases} \tag{2.1.6}$$

Khi đó bài toán tối ưu này là bài toán đã được giải ở phần đầu của Phần 2.1.1, cụ thể ở đây U được thay bởi $X^T U$, B được thay bởi I và L được thay bởi $-L$. Từ đó, chúng ta thu được nghiệm là ma trận U, trong đó các cột của U là các vector riêng tương ứng với các giá trị riêng của ma trận Laplace L (chú ý rằng các giá trị riêng được viết theo thứ tự từ lớn đến bé). Kết quả của phép tọa độ hóa là các hàng của U.

Phương pháp tọa độ hóa sử dụng thuật toán LLE (locally linear embbeding)

Trước hết, chúng tôi sẽ nhắc lại thuật toán LLE [10]. Thuật toán LLE có dữ liệu đầu vào là tập $X = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \subset \mathbb{R}^d$ và đầu ra là các tập $Y = \{\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n\} \subset \mathbb{R}^p$ với p nhỏ hơn nhiều so với d, thuật toán bao gồm ba bước chính như sau:

- Bước đầu tiên, chúng ta sẽ xây dựng đồ thị k-NN (k nearest neighbor) của tập X. Cụ thể, với mỗi điểm $\mathbf{x}_i$, ta sẽ tạo ra một ma trận tương ứng $\mathbf{X}_i$ trong đó mỗi cột là vector điểm dữ liệu láng giềng gần nhất của $\mathbf{x}_i$:

  $$\mathbf{X}_i := [\mathbf{x}_{i_1}, \mathbf{x}_{i_2}, ..., \mathbf{x}_{i_k}],$$

  Ở đây, $\mathbf{x}_{i_1}, ..., \mathbf{x}_{i_k}$ là k điểm lân cận gần nhất của điểm $\mathbf{x}_i$. Chúng ta sẽ lựa chọn một giá trị k đủ lớn để đảm bảo rằng đồ thị k-NN của X là một đồ thị liên thông.

- Tiếp theo, với mỗi điểm $\mathbf{x}_i$, chúng ta sẽ tìm tổ hợp tuyến tính của k điểm láng giềng gần nhất của $\mathbf{x}_i$ sao cho biểu diễn tuyến tính đó xấp

xỉ tốt nhất cho $\mathbf{x}_i$. Ta cần tối ưu hàm mất mát $\epsilon(\overline{W}) := \sum_{i=1}^{n} \|\mathbf{x}_i - \sum_{j=1}^{k} \overline{W}_{ij}\mathbf{x}_{ij}\|_2^2$. Từ đó, chúng ta có bài toán tối ưu sau:

$$\begin{cases} \min\limits_{\overline{W}} \epsilon(\overline{W}), \\ \sum_{j=1}^{n} \overline{W}_{ij} = 1 \quad \forall i \in \{1, 2, ..., n\}. \end{cases} \tag{2.1.7}$$

Cuối cùng, sau khi đã có ma trận $W \in \mathbb{R}^{n\times k}$ trong đó $W = [\mathbf{w}_1, \mathbf{w}_2, ..., \mathbf{w}_n]$ với $\mathbf{w}_i$ là vector chứa các hệ số trong biểu diễn tuyến tính của $\mathbf{x}_i$, lưu ý rằng, hàm mục tiêu trong bài toán 2.1.7 có thể được viết gọn lại như sau:

$$\epsilon(\bar{W}) = \sum_{i=1}^{n} \|\mathbf{x}_i - \mathbf{x}_i\bar{w}_i\|_2^2 \tag{2.1.8}$$

- Bước cuối cùng chính là áp dụng phương pháp tọa độ hóa tuyến tính cho đồ thị tương ứng với ma trận đối xứng W vừa tìm được. Cụ thể, ở bước này, sau khi đã có ma trận W từ bước hai, ta giải bài toán tối ưu sau đây để tìm biểu chiều thấp của các điểm dữ liệu gốc trong không gian $\mathbb{R}^p$:

$$\begin{cases} \min\limits_{Y} \sum_{i=1}^{n} \|\mathbf{y}_i - \sum_{j=1}^{n} W_{ij}\mathbf{y}_j\|_2^2; \\ \frac{1}{n}\sum_{i=1}^{n} \mathbf{y}_i\mathbf{y}_i^T = I; \\ \sum_{i=1}^{n} \mathbf{y}_i = 0. \end{cases} \tag{2.1.9}$$

Ở đây $Y = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]^T \in \mathbb{R}^{n\times p}$ là ma trận với các vector hàng $\mathbf{y}_i \in \mathbb{R}^p$ chính là kết quả của $\mathbf{x}_i$ sau khi thực hiện thuật toán LLE.

Ở đây, chúng ta sẽ chỉ tập trung quan tâm Bước 3 của của thuật toán LLE. Từ đó ứng dụng cho bài toán tọa độ hóa đồ thị. Bài toán 2.1.9 có thể được

viết lại như sau:

$$\begin{cases} \min\limits_{Y} tr(Y^T M T); \\ \frac{1}{n} Y^T Y = I; \\ Y^T \mathbf{1} = 0, \end{cases} \tag{2.1.10}$$

trong đó

$$M := (I - W)\,(I - W)^T \in \mathbb{R}^{n \times n}.$$

Xét tổng các phần tử của hàng thứ i bất kỳ của ma trận M:

$$\begin{aligned} \sum_{j=1}^{n} M_{ij} &= \sum_{j=1}^{n} \Big((I - W)(I - W)^T \Big)_{ij} \\ &= \sum_{j=1}^{n} \Big(I_{ij} - W_{ij} - W_{ji} + (WW^T)_{ij} \Big) \\ &= 1 - \sum_{j=1}^{n} (W_{ij} + W_{ji}) + \sum_{j=1}^{n} \sum_{k=1}^{n} (W_{ij} W_{jk}) \\ &= 1 - \sum_{j=1}^{n} \left(W_{ij} + W_{ji} \right) + \sum_{j=1}^{n} \left(W_{ij} \sum_{k=1}^{n} W_{jk} \right) \\ &= 1 - \sum_{J=1}^{n} W_{ij} - \sum_{j=1}^{n} W_{ji} + \sum_{j=1}^{n} W_{ij} = 0. \end{aligned}$$

Từ đây ta có thể thay thế ma trận M trong thuật toán gốc với một ma trận Laplace của một đồ thị vô hướng.

Lưu ý thêm rằng điều kiện ràng buộc thứ 2 có thể được bỏ qua và khi đó ta thấy thuật toán tọa độ hóa thông qua LLE là một trường hợp riêng của phương pháp tọa độ hóa trực tiếp đã trình bày ở phần đầu Phần 2.1.1 với $L = M$ và $B = \frac{1}{n} I$.

Vậy kết quả cuối cùng là các hàng của ma trận Y thỏa mãn:

$$LY = \frac{1}{n} Y \Lambda.$$

Phương pháp tọa độ hóa sử dụng thuật toán MDS

Ta xét phương pháp Kernel MDS [10] cổ điển sử dụng kernel được định nghĩa như sau:

$$K := -\frac{1}{2}HDH,$$

trong đó D là ma trận với các thành phần là bình phương các khoảng cách Euclid giữa các điểm dữ liệu ban đầu và

$$H := I - \frac{1}{n}\mathbf{1}\mathbf{1}^T.$$

Trong đó $\mathbf{1}$ là vector với các thành phần toàn số 1. Xét tổng các phần tử trong một hàng i bất kỳ của K ta có

$$\begin{aligned}
\sum_{j=1}^{n} K_{ij} &= \sum_{j=1}^{n}\left(-\frac{1}{2}HDH\right)_{ij} = \sum_{j=1}^{n}\left(-\frac{1}{2}\left(I-\frac{1}{2}\mathbf{1}\mathbf{1}^T\right)D\left(I-\frac{1}{n}\mathbf{1}\mathbf{1}^T\right)\right)_{ij} \\
&= \frac{1}{2}\sum_{j=1}^{n}\left(-D_{ij}+\frac{1}{n}\sum_{i'=1}^{n}D_{i'j}+\frac{1}{n}\sum_{j'=1}^{n}\left(D_{ij'}-\frac{1}{n}\sum_{k'=1}^{n}D_{k'j'}\right)\right) \\
&= \left(-\frac{1}{2}\sum_{j=1}^{n}D_{ij}+\frac{1}{2n}\sum_{j=1}^{n}\sum_{i'=1}^{n}D_{i'j}\right)+\left(\frac{1}{2}\sum_{j=1}^{n}\sum_{j'=1}^{n}D_{ij'}-\frac{1}{2n^2}\sum_{j=1}^{n}\sum_{j'=1}^{n}\sum_{k'=1}^{n}D_{k'j'}\right) \\
&= 0.
\end{aligned}$$

Từ đây thay vì sử dụng kernel K như trong MDS thì ta có thể được thay nó bởi ma trận Laplace của một đồ thị vô hướng $G = (V, E)$. Sau cùng ta cần giải bài toán tối ưu sau:

$$\begin{cases} \min\limits_{Y} tr(Y^T L Y); \\ Y^T Y = I. \end{cases} \tag{2.1.11}$$

Từ đây thuật toán MDS có thể nói là trường hợp riêng của phương pháp tọa độ hóa trực tiếp đã trình bày ở phần đầu mục 2.1.1 với trực tiếp với

$B = I$. Vậy, kết quả của phép tọa độ hóa là các hàng của ma trận X, trong đó X là ma trận với các cột là các vector riêng ứng với các giá trị riêng của ma trận Laplace L.

2.1.2. Phương pháp tọa độ hóa đồ thị vô hướng sử dụng bước đi ngẫu nhiên

Tiếp theo, tôi sẽ trình bày phương pháp tọa độ hóa dựa trên bước đi ngẫu nhiên. Cho $G = (V, E)$ là một đồ thị vô hướng và liên thông. Trong [6], tác giả đã có nhận xét rằng: nếu hai đỉnh i và j thuộc cùng một cộng đồng thì xác suất đi đến một đỉnh l bất kỳ sau t bước giữa chúng là gần bằng nhau, nghĩa là:

$$P_{il}^t \simeq P_{jl}^t, \text{ với mọi } l. \tag{2.1.12}$$

Sau đó các tác giả đã định nghĩa khoảng cách giữa chúng:

$$R_{ij}(t) := \|D^{-1/2}P_{i\bullet}^t - D^{-1/2}P_{j\bullet}^t\|.$$

Từ ý tưởng này, chúng ta có thể tương ứng mỗi i với vector $D^{-1/2}P_{i\bullet}^t$. Tức là chúng ta có thể coi tọa độ của mỗi đỉnh i là:

$$C(i) := D^{-1/2}P_{i\bullet}^t = \left\{d_1^{-1/2}P_{i1}^t, d_2^{-1/2}P_{i2}^t, ..., d_n^{-1/2}P_{in}^t\right\}. \tag{2.1.13}$$

Rõ ràng với cách tọa độ hóa trên, ta coi độ tương đồng giữa hai đỉnh i và j tương ứng với khoảng cách giữa hai vector tương ứng $C(i)$ và $C(j)$.

2.2. Một số phương pháp tọa độ hóa đồ thị có hướng

2.2.1. Phương pháp tọa độ hóa phổ

Trước hết để tiện trình bày, ta xét trường hợp với G là một đồ thị hai phần, ta ký hiệu $G = (V_1, V_2, E)$ khi đó ma trận kề của G có dạng:

$$A = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix}$$

Khi đó ta cũng sẽ thấy rằng ma trận đường chéo D sẽ có dạng

$$D = \begin{bmatrix} D_1 & 0 \\ 0 & D_2 \end{bmatrix},$$

với $D_1 = diag(B\mathbf{1})$ và $D_2 = diag(B^T\mathbf{1})$. Khi đó ta có kết quả tọa độ hóa có dạng

$$X = \begin{bmatrix} X_1 \\ X_2 \end{bmatrix},$$

với X_1 và X_2 là kết quả tọa độ hóa của các đỉnh trong V_1 và V_2:

$$P_1X_1 = X_1M,$$

$$P_2X_2 = X_2M.$$

Ta cũng nhận thấy rằng nếu μ là một giá trị riêng của P thì $-\mu$ cũng là giá trị riêng của P.

Quay trở lại với trường hợp có hướng. Một cách đơn giản để tọa độ hóa các đỉnh của đồ thị có hướng $G = (V, E)$ với ma trận kề A đó là coi đồ thị của ta như một đồ thị thị hai phần với ma trận $B = A$. Như vậy mỗi đỉnh của G được xuất hiện hai lần trong đồ thị hai phần, một phiên bản là nguồn và một phiên bản là đích của các cạnh. Ta gọi đồ thị G' là đồ thị gương (mirror graph). Kết quả của phép tọa độ hóa đồ thị G được lấy từ ánh xạ riêng Laplacian của một phần trong đồ thị gương X_1. Một cách khác là sử dụng phân tích giá trị kỳ dị (SVD) phân tích này cho ta một xấp xỉ chiều thấp tốt cho với mọi ma trận. Trước hết, như ở phần trên, ta sẽ xét phân tích SVD của ma trận B tương ứng với đồ thị hai phần $G = (V_1, V_2, E)$:

$$\begin{cases} BW = D_1U\Sigma; \\ B^TU = D_1W\Sigma. \end{cases} \tag{2.2.1}$$

Trong đó

$$U^T D_1 U = W^T D_2 W = I,$$

với $D_1 = diag(B_1), D_2 = diag(B_1^T)$ và $\Sigma = diag(\sigma_1, ..., \sigma_r)$ là ma trận đường chéo các giá trị kỳ dị lớn hơn 0:

$$\sigma_1 \geq \sigma_2 \geq ... \geq \sigma_r > 0.$$

Từ đây ta cũng dễ có r là hạng của ma trận B. Ta có thể thay thế ma trận B bởi một ma trận kề của bất kỳ một đồ thị có hướng (vô hướng) và khi đó thay vì làm việc với đồ thị gốc thì ta làm việc với đồ thị gương.

Ta dễ dàng kiểm tra được phân tích SVD của ma trận B (giờ có thể coi như ma trận kề của một đồ thị vô hướng hoặc thậm chí có hướng) tương ứng với phân tích phổ của ma trận chuyển P cuả đồ thị G với ma trận kề

$$A = \begin{bmatrix} 0 & B \\ B^T & 0 \end{bmatrix}.$$

Ta có

$$P \begin{bmatrix} U \\ V \end{bmatrix} = \begin{bmatrix} U \\ V \end{bmatrix} \Sigma.$$

Từ đây ta thấy rằng, các vector kỳ dị của B là các vector riêng của P ứng với các giá trị riêng dương, khi đó tọa độ hóa X_1,X_2 cho bởi k vector kỳ dị của B trừ vector kỳ dị đầu tiên.

2.2.2. Phương pháp tọa độ hóa đồ thị có hướng sử dụng bước đi ngẫu nhiên

Cho đồ thị có hướng, liên thông mạnh G, với $\Phi^{1/2} = \text{diag}[\sqrt{\phi_u}]$. Yanhua và Z. L. Zhang [14] đã định nghĩa ma trận Laplace chuẩn hóa $\Gamma = [\Gamma_{uv}]$ cho đồ thị có hướng (viết tắt là Diplacian) như sau.

Định nghĩa 2.2.1 ([14]). *Diplacian* Γ *được định nghĩa như sau*

$$\Gamma = \Phi^{1/2}(I - P)\Phi^{-1/2}. \tag{2.2.2}$$

Trong [11], các tác giả đã định nghĩa khoảng cách giữa các đỉnh trên đồ thị có hướng dựa vào thời điểm chạm của bước đi ngẫu nhiên. Sau đó, các tác giả đã đưa ra mối liên hệ của khoảng cách sử dụng thời điểm chạm và phân tích giá trị kỳ dị của ma trận Laplace chuẩn hóa Γ. Từ đó, các tác giả giới thiệu phương pháp tọa độ hóa thông qua phân tích giá trị kỳ dị như sau. Ta xét phân tích giá trị kỳ dị của ma trận Laplace chuẩn hóa $\Gamma = U\Sigma W^T$, với $W = [w_1, w_2, ..., w_n]$. Đặt $W_k = [w_1, w_2, \ldots, w_k]$ trong đó w_i là cột thứ i của ma trận W. Với mỗi đỉnh u trong đồ thị có hướng G, ta có:

$$C(i) = \left(\phi_i^{-1/2} V_1(i), \ldots, \phi_i^{-1/2} V_k(i), \phi_i^{-1/2} U_1(i), \ldots, \phi_i^{-1/2} U_k(i)\right). \tag{2.2.3}$$

Lưu ý rằng vai trò của các vector riêng phải và trái trong phân tích SVD của ma trận Laplace chuẩn hóa trong việc tìm phân cụm của đồ thị là như nhau. Do đó để biểu diễn tọa độ của các đỉnh bằng cách dùng cá vector phải như ở trên thì việc thay vào bằng cách vector trái cũng cho ta kết quả tương tự. Sau đây là sơ lược phương pháp tọa độ hóa:

- Đầu tiên, ta xét phân tích giá trị kỳ dị của ma trận Laplace chuẩn hóa $\Gamma = U\Sigma V^\top$, với U, V, và Σ lần lượt là các ma trận với các vector cột là các vector kỳ dị trái, phải và ma trận đường chéo với các phần tử trên đường chéo là các giá trị kỳ dị.

- Sau đó, ta chọn ra k cột đầu tiên của V và U và xây dựng ma trận $V_k = [\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k]$ và $U_k = [\mathbf{u}_1, \mathbf{u}_2, \ldots, \mathbf{u}_k]$. Với mỗi đỉnh i của đồ thị G, ta gán tọa độ cho đỉnh i với vector thứ i của ma trận $R = [\Phi^{-1/2} U_k, \Phi^{-1/2} V_k]$. Hơn nữa, tọa độ của của i có thể được biểu thị như sau:

$$C(i) = \left(\phi_i^{-1/2} u_{i1}, \ldots, \phi_i^{-1/2} u_{ik}, \phi_i^{-1/2} v_{i1} \ldots, \phi_i^{-1/2} v_{ik}\right) \tag{2.2.4}$$

Chương 3

Thuật toán K-Means, K-Means++, K-Means$\|$

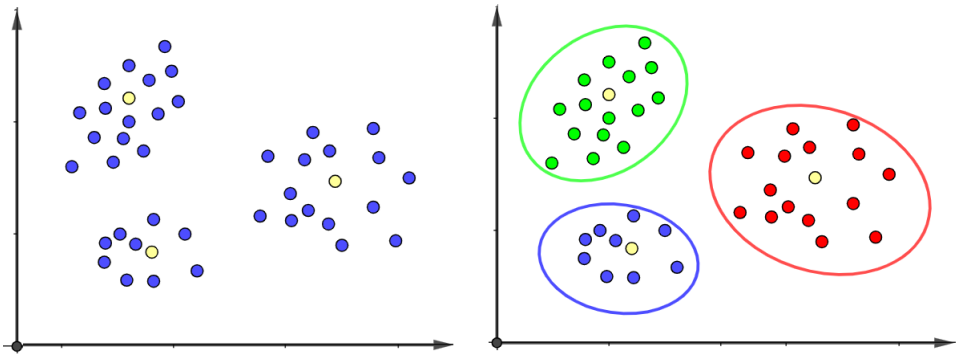
Trong phần này, chúng tôi giới thiệu thuật toán K-Means và hai phiên bản cải tiến của nó là K-Means++ và K-Means$\|$. Nội dung của phần này chủ yếu dựa trên các tài liệu từ [15] và [16].

3.1. Thuật toán K-Means

3.1.1. Giới thiệu

K-Means là một thuật toán không giám sát. Trong thuật toán này, chúng ta không có thông tin về nhãn của các điểm dữ liệu trước. Mục tiêu của thuật toán là phân chia các điểm dữ liệu vào các cụm sao cho khoảng cách giữa các điểm trong cùng một cụm nhỏ hơn đáng kể so với khoảng cách giữa các điểm thuộc các cụm khác nhau.

Hình 3.1 minh họa một ví dụ của một bộ dữ liệu có ba cụm. Mỗi cụm được biểu diễn bằng một điểm đại diện màu vàng, được gọi là tâm của cụm. Ý tưởng đơn giản nhất là với một điểm bất kỳ, chúng ta xác định điểm đó thuộc về cụm nào bằng cách xem nó gần với tâm nào nhất.



Hình 3.1: Minh họa về tập dữ liệu gồm có ba cụm. Hình bên trái là tập dữ liệu ban đầu. Hình bên phải là kết quả phân cụm sử dụng thuật toán *K*-Means.

3.1.2. Mô tả thuật toán *K*-Means

Thuật toán *K*-Means nhận đầu vào là tập dữ liệu $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \in \mathbb{R}^n$ và số lượng cụm cho trước K. Đầu ra của thuật toán là các tâm $\mathbf{M}$ và vector nhãn cho mỗi điểm dữ liệu $\mathbf{Y}$. Quá trình thực hiện thuật toán như sau: Đầu tiên, chọn K điểm bất kỳ làm các tâm khởi tạo. Sau đó gán mỗi điểm dữ liệu vào cụm có tâm gần nó nhất. Nếu việc gán dữ liệu vào từng cụm ở bước 2 không thay đổi so với vòng lặp trước, thì dừng thuật toán. Ngược lại, ta tiếp tục cập nhật các tâm cho từng cụm bằng cách tính trung bình của các điểm dữ liệu đã được gán vào cụm đó ở bước 2 trong vòng lặp trước. Cuối cùng, quay lại bước 2 để tiếp tục quá trình gán và cập nhật đến khi không có sự thay đổi trong việc gán dữ liệu vào các cụm. Từ đó, chúng ta có thể viết thuật toán *K*-Means dưới dạng giả mã như sau:

Thuật toán 1 Thuật toán *K*-Means trong không gian Euclid

Input: Tập dữ liệu $\mathbf{X} = \{\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n\} \subset \mathbb{R}^n$.
Output: Phân hoạch $P = (C_1, C_2, ..., C_K)$ với các tâm $\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K$.

Khởi tạo: Chọn K tâm $\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K$ ban đầu ngẫu nhiên.
while *chưa hội tụ* **do**
  for *j=1,...,k* **do**
    $C_j \leftarrow \emptyset$.
  end
  for *i=1,...,n* **do**
    $j^* \leftarrow arg \min_{j=1,2,...,K} \|\mathrm{x}_i - m_j\|^2$
    $C_{j^*} \leftarrow C_{j^*} \cup \{x_i\}$
  end
  for $j = 1, 2, ..., K$ **do**
    $m_j = \frac{1}{|C_j|} \sum_{x_i \in C_j} x_i$.
  end
end

Thuật toán *K*-Means đơn giản và có thời gian chạy thực khá nhanh, dẫu vậy thuật toán này cũng có một số hạn chế. Đầu tiên, thuật toán không cho ta cách tìm số cụm và việc lựa chọn số cụm là ngẫu nhiên. Điều này có thể ảnh hưởng nghiêm trọng đến chất lượng phân cụm. Cụ thể là nếu ta chọn số cụm K lớn hay nhỏ hơn quá nhiều số với số cụm thực tế. Hơn nữa, kết quả phân cụm sau cuối cũng như tốc độ hội tụ của thuật toán phụ thuộc rất nhiều vào việc khởi tạo tâm, trong khi thuật toán *K*-Means gốc hoàn toàn chọn ngẫu nhiên các tâm khởi tạo. Cuối cùng, ta thấy rằng *K*-Means nhạy cảm với cỡ và hình dạng của các cụm. Thuật toán hoạt động không tốt khi các cụm có kích thước và hình dạng không đồng nhất.

3.1.3. Cơ sở toán học

Bài toán: Cho tập dữ liệu đầu vào gồm n điểm trong không gian $\mathbb{R}^d$ và số lượng K cụm cho trước, mục tiêu của ta là chỉ ra tâm của mỗi cụm và phân các điểm dữ liệu vào các cụm tương ứng. Ta cũng giả sử thêm rằng mỗi điểm dữ liệu chỉ thuộc vào đúng một cụm.

Giả sử chúng ta có một tập hợp gồm n điểm dữ liệu được biểu diễn bởi $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}, ...\mathbf{x_n}]$, và K (với K nhỏ hơn rất nhiều so với n) là số cụm đã được xác định trước. Nhiệm vụ của chúng ta là tìm các tâm $\mathbf{m_1}, \mathbf{m_2}, ...\mathbf{m_K}$ và gán nhãn cho mỗi điểm dữ liệu. Đặt $\mathbf{y_i} = [y_{i1}, y_{i2}, ...y_{iK}]$ là vector nhãn của điểm dữ liệu $\mathbf{x_i}$, trong đó $y_{ij} = 1$ nếu $\mathbf{x_i}$ thuộc cụm j, và $y_{ij} = 0$ nếu $\mathbf{x_i}$ không thuộc cụm j.

Hàm mất mát và bài toán tối ưu: Trong bài toán này, chúng ta cần tìm một tập hợp các tâm $M = \{\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K\}$ và gán nhãn cho mỗi điểm dữ liệu $Y = [\mathbf{y_1}, \mathbf{y_2}, .., \mathbf{y_n}]$, với $\mathbf{y}_i$ là vector nhãn của điểm $\mathbf{x}_i$, sao cho khoảng cách $\|\mathbf{x}_i - \mathbf{m}_K\|_2^2$ giữa mỗi điểm $\mathbf{x}_i$ và tâm gần nhất là nhỏ nhất có thể. Để làm được điều này, chúng ta định nghĩa hàm mất mát như sau:

$$L(\mathbf{Y}, \mathbf{M}) = \sum_{i=1}^{n} \sum_{j=1}^{K} y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2. \tag{3.1.1}$$

Ở đây, $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, ..., \mathbf{y}_n]$ và $\mathbf{M} = [\mathbf{m}_1, \mathbf{m}_2, ..., \mathbf{m}_K]$. Từ đó, để tìm các tâm cụm và gán nhãn cho các điểm dữ liệu, chúng ta cần giải bài toán tối ưu sau:

$$\begin{cases} \min\limits_{\mathbf{Y},\mathbf{M}} \sum_{i=1}^{n} \sum_{j=1}^{K} y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2. \\ y_{ij} \in \{0, 1\} \ \forall i, j; \ \sum_{j=1}^{K} y_{ij} = 1 \ \forall i. \end{cases} \tag{3.1.2}$$

Tối ưu hàm mất mát: Một cách đơn giản để giải bài toán trên là lần lượt cố định M và Y và tối ưu theo biến còn lại. Như vậy ta sẽ phải giải xen kẽ liên tiếp hai bài toán tối ưu sau đây:

Cố định M, tối ưu theo Y: Giả sử ta đã có các tâm M, ta sẽ tìm các vector nhãn y_i để hàm mất mát đạt giá trị nhỏ nhất với cách chọn tâm M cố định. Khi các tâm là cố định, bài toán tìm các vector nhãn cho toàn bộ dữ liệu có thể được chia nhỏ thành bài toán tìm vector nhãn cho từng điểm dữ liệu $\mathbf{x}_i$ như sau:

$$\begin{cases} \mathbf{y}_i = \arg\min_{\mathbf{y}_i} \sum_{j=1}^{K} y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2; \\ y_{ij} \in \{0,1\}\ \forall i,j;\ \sum_{j=1}^{K} y_{ij} = 1\ \forall i. \end{cases} \tag{3.1.3}$$

Vì chỉ có đúng một thành phần của vector nhãn $\mathbf{y}_i$ bằng 1 nên bài toán (2) có thể tiếp tục được viết dưới dạng đơn giản hơn:

$$j^* = \arg\min_j \|\mathbf{x}_i - \mathbf{m}_j\|_2^2.$$

Từ đây ta dễ dàng thấy rằng mỗi điểm dữ liệu x_i sẽ có nhãn $\mathbf{y_i}$ với $y_{ij^*} = 1$ nếu m_{j^*} là tâm gần x_i nhất.

Cố định Y, tối ưu theo M: Giả sử ngược lại rằng chúng ta đã gán nhãn cho mỗi điểm dữ liệu. Bây giờ, với mỗi cụm, chúng ta cần tìm một tâm mới sao cho hàm mất mát được tối ưu. Bài toán tìm tâm cho mỗi cụm được biểu diễn như sau:

$$\mathbf{m}_j^* = \arg\min_{\mathbf{m}_j} \sum_{i=1}^{N} y_{ij} \|\mathbf{x}_i - \mathbf{m}_l\|_2^2. \tag{3.1.4}$$

Để giải bài toán tối ưu trên ta giải phương trình:

$$\frac{\partial(l(\mathbf{m}_j))}{\partial \mathbf{m}_j} = 2 \sum_{i=1}^{N} y_{ij}(\mathbf{m}_j - \mathbf{x}_i) = 0.$$

Từ đó ta suy ra:

$$\mathbf{m}_j = \frac{\sum_{i=1}^{N} y_{ij} \mathbf{x}_i}{\sum_{i=1}^{N} y_{ij}}.$$

Trong đó $l(\mathbf{m}_j) = \sum_{i=1}^{N} y_{ij} \|\mathbf{x}_i - \mathbf{m}_j\|_2^2$.

Nhận xét 3.1.1. *Từ đây ta có thể nói tâm mới:* $\mathbf{m}_j$ *sẽ là trọng tâm của các điểm trong cụm* j.

Một cách chứng minh khác cho nghiệm tối ưu của bài toán tối ưu hàm mất mát theo M với Y có định là sử dụng trực tiếp kết quả sau:

Bổ đề 3.1.1. *Cho* $\mathbf{z}_1, \mathbf{z}_2, .., \mathbf{z}_n \in \mathbb{R}^d$, *và* $\bar{\mathbf{z}} := \frac{\sum_{i=1} \mathbf{z}_i}{n}$, *khi đó với mọi* $\mathbf{z} \in \mathbb{R}^d$, *ta có*

$$\sum_{i=1}^{n} \|\mathbf{z}_i - \bar{\mathbf{z}}\|^2 \leq \sum_{i=1}^{n} \|\mathbf{z}_i - \mathbf{z}\|^2.$$

Chứng minh. Ta có:

$$\begin{aligned}
\sum \|\mathbf{z}_i - z\|^2 &= \sum_{i=1}^{n} \|(\mathbf{z}_i - \bar{\mathbf{z}}) + (\bar{\mathbf{z}} - z)\|^2 \\
&= \sum_{i=1}^{n} (\|\mathbf{z}_i - \bar{\mathbf{z}}\|^2 + \|\bar{\mathbf{z}} - \mathbf{z}\|^2 + 2(\mathbf{z}_i - \bar{\mathbf{z}})(\bar{\mathbf{z}} - \mathbf{z})) \\
&= \sum_{i=1}^{n} \|\mathbf{z}_i - \bar{\mathbf{z}}\|^2 + \sum_{i=1}^{n} \|\bar{\mathbf{z}} - \mathbf{z}\|^2 + 2 \sum_{i=1}^{n} (\mathbf{z}_i \bar{\mathbf{z}} - \mathbf{z}_i \mathbf{z} - \bar{\mathbf{z}}\bar{\mathbf{z}} + \bar{\mathbf{z}}\mathbf{z}) \\
&= \sum_{i=1}^{n} \|\mathbf{z}_i - \bar{\mathbf{z}}\|^2 + n\|\bar{\mathbf{z}} - \mathbf{z}\|^2 + 2(n\bar{\mathbf{z}}\bar{\mathbf{z}}) - n\bar{\mathbf{z}}\mathbf{z} - n\bar{\mathbf{z}}\bar{\mathbf{z}} + \mathbf{n}\bar{\mathbf{z}}\mathbf{z}) \\
&= \sum_{i=1}^{n} \|\mathbf{z}_i - \bar{\mathbf{z}}\|^2 + n\|\bar{\mathbf{z}} - \mathbf{z}\|^2 \\
&\geq \sum_{i=1}^{n} \|\mathbf{z}_i - \bar{\mathbf{z}}\|^2.
\end{aligned}$$

□

Trước khi đi vào định lý về sự hội tụ của thuật toán, chúng ta viết lại hàm mất mát tại bước thứ t như sau:

$$L(M^{(t)}, \mu^{(t)}) = \sum_{i=1}^{n} \|x_i - m_{\mu^{(t)}(i)}\|^2.$$

Trong đó $\mu^{(t)}(i)$ là số thứ tự của cụm mà điểm x_i thuộc về trong bước thứ t và $m_{\mu^{(t)}(i)}$ là tâm của cụm của điểm $\mathbf{x}_i$ tại vòng lặp thứ t. Cuối cùng

chúng tôi xin trình bày tính hội tụ của thuật toán K-Means qua định lý sau:

Định lí 3.1.1. *Thuật toán K-Means hội tụ.*

Chứng minh. Ta sẽ chứng minh định lý trên theo hai bước:

- Bước 1, ta sẽ chứng minh $L(M^{(t)}, \mu^{(t+1)}) < L(M^{(t)}, \mu^{(t)})$.
- Bước 2, ta có $L(M^{(t+1)}, \mu^{(t+1)}) \leq L(M^{(t)}, \mu^{(t+1)})$.

Bước một được chứng minh hiển nhiên từ việc các đỉnh sẽ được cho vào cụm có tâm gần với đỉnh đó nhất. Để chứng minh bước 2 ta sử dụng Bổ đề 3.1.1:

$$\begin{aligned} L(M^{(t+)}, \mu^{(t+1)}) &= \sum_{i=1}^{n} \|x_i - m^{(t+)}_{\mu^{(t+1)}(i)}\|^2 \\ &= \sum_{l=1}^{K} \sum_{\mu^{(t+1)}(i)=l}^{n} \|x_i - m^{(t+1)}_{\mu^{(t+1)}(i)}\|^2 \\ &\leq \sum_{l=1}^{K} \sum_{\mu^{(t+1)}(i)=l} \|x_i - m^{(t)}_{\mu^{(t+1)}(i)}\|^2 \\ &= \sum_{i=1}^{n} \|x_i - m^{(t)}_{\mu^{(t+1)}(i)}\|^2 = L(M^t, \mu^{(t+1)}). \end{aligned}$$

□

3.2. Thuật toán K-Means++ và K-Means‖

3.2.1. Thuật toán K-Means++

K-Means++ [15] đem lại sự cải thiện đáng kể trong việc khởi tạo các tâm ban đầu so với phương pháp khởi tạo ngẫu nhiên thông thường, đồng thời tăng cường chất lượng của quá trình phân cụm. Bằng cách này, nó

giúp tăng tốc độ hội tụ và đạt được kết quả tốt hơn trong việc xác định các cụm.

Ở mỗi bước của quá trình khởi tạo, chúng tôi tính toán khoảng cách ngắn nhất từ mỗi điểm dữ liệu x đến tâm gần nhất đã được chọn trước đó, được ký hiệu bằng $D(x)$. Sau đó, chúng tôi sử dụng quy trình sau: Ban đầu, chúng tôi chọn một tâm c_1 ngẫu nhiên từ tập dữ liệu $\mathcal{X}$ với xác suất đồng đều. Đối với các tâm tiếp theo c_i, chúng tôi chọn $c_i = x' \in \mathbf{X}$ với xác suất tỷ lệ với $\frac{D(x')^2}{\sum_{x\in\mathcal{X}} D(x)^2}$. Chúng tôi lặp lại quá trình này cho đến khi chúng tôi đã chọn được tổng cộng K tâm. Sau đó, các bước tiếp theo được thực hiện như trong thuật toán K-Means.

Do các bước sau hoàn toàn tương tự thuật toán K-Means nên chúng ta chỉ cần mô tả lại bước chọn tâm khởi tạo của thuật toán K-Means++ dưới dạng giả mã như sau:

Thuật toán 2 Khởi tạo tâm ban đầu của thuật toán K-Means++

Input: Tập dữ liệu $\mathbf{X}$, số cụm K.
Output: Tập S là tập các tâm khởi tạo.
Chọn một tâm c_1 ngẫu nhiên từ $\mathbf{X}$;
$S = \{c_1\}$;
$\ell = 1$;
while $\ell < K$ **do**
  $\ell \leftarrow \ell + 1$
  Chọn $c_\ell = x' \in \mathbf{X}$ với xác suất $\frac{D(x')^2}{\sum_{x\in\mathbf{X}} D(x)^2}$.
  $S \leftarrow S \cup \{c_\ell\}$.
end

Về mặt lý thuyết sự ưu việt của thuật toán K-Means++ được thể hiện qua định lý sau:

Định lí 3.2.1 ([15]). *Giả sử sau khi thuật toán K-Means++ hội tụ, ta thu được tập tâm M. Khi đó hàm mất mát tương ứng L sẽ thỏa mãn:*

$$E[L] \leq 8(lnK + 2)L_{OPT}. \tag{3.2.1}$$

Trong đó, L_{OPT} là giá trị cực tiểu của hàm mất mát ứng với một cách chọn tâm tối ưu M_{OPT}.

Nhận xét 3.2.1. *Theo định lý trên ta có thể thấy giá trị trung bình của hàm mất mát ứng với tập tâm M thu được từ thuật toán K-Means++ thuộc lớp $Oln(K)$ của giá trị hàm mất mát L_{OPT} tương ứng với một cách chọn tâm tối ưu M_{OPT}.*

3.2.2. Thuật toán K-Mean $\|$

Bây giờ chúng tôi sẽ trình bày thuật toán K-Means$\|$ (còn gọi là scalable K-Means), một phiên bản cải tiến của thuật toán K-Means trong việc khởi tạo các tâm. Mặc dù thuật toán này chủ yếu được lấy cảm hứng từ K-Means++, nhưng nó sử dụng kỹ thuật lấy mẫu vượt quá mức với hệ số $\ell = \Omega(K)$, không giống như K-Means++; một cách trực giác, ℓ nên được coi như $\Theta(K)$.

Thuật toán này bắt đầu bằng việc chọn một tâm ban đầu ngẫu nhiên với xác suất đồng đều cho các điểm dữ liệu và tính toán giá trị của hàm mất mát $\psi = L_X(C)$ với $L_X(C) = \sum_{x \in X} \min_{i=1,\dots,K} \|\mathbf{x} - \mathbf{c}_i\|^2 \sum_{x \in \mathbf{X}} d^2(x, C)$, với $d(x, C) := \min_{i=1,\dots,K} \|x - c_i\|$. Sau đó, tiến hành với số lần lặp $\log \psi$, trong mỗi lần lặp, ta có một tập các tâm S, mỗi $\mathbf{x}$ được chọn với xác suất $\frac{\ell d^2(x,S)}{L_X(S)}$. Các điểm được chọn sau đó được thêm vào S, đại lượng $L_X(S$ được cập nhật, và lặp lại. Ta sẽ thấy rằng, số điểm dự kiến được chọn trong mỗi lần lặp là ℓ và cuối cùng, số phần tử kỳ vọng trong S là $\ell \cdot \log Ł$, lớn hơn K.

Để giảm số lượng tâm, ta sẽ gán trọng số cho các điểm trong S và sau đó sẽ sắp xếp lại các điểm có trọng số này để lấy được chính xác K tâm. Dưới đây là phần giả mã của thuật toán K-Means$\|$:

Tác giả trong bài báo [16] có nhận xét rằng thuật toán K-Means$\|$ mang lại ưu điểm lớn hơn so với K-Means++ bằng cách giảm đáng kể thời gian

Thuật toán 3 Khởi tạo tâm ban đầu của thuật toán K-means $\|$

Input: Tập dữ liệu $\mathbf{X}$, số cụm K.
Output: Tập S gồm các điểm được chọn làm tâm ban đầu.
$S \leftarrow$ lấy một điểm bất kỳ ngẫu nhiên từ $\mathbf{X}$;
$\psi \leftarrow L_X(S)$;
$n \in \mathbb{N} \cap O(\log \psi)$;
for $i = 1, ..., n$ **do**
    $S' \leftarrow$ chọn mỗi điểm $x \in X$ độc lập với xác suất $p_x = \frac{\ell \cdot d^2(\mathbf{x}, S)}{L_X(S)}$;
    $S \leftarrow S \cup S'$;
end
Cho $\mathbf{x} \in S$, đặt w_x là số điểm trong $\mathbf{X}$ gần với $\mathbf{x}$ hơn các điểm khác trong S; chọn K đỉnh làm tâm từ tập S với xác suất được lấy của mỗi đỉnh x tỉ lệ với thuận với w_x.

tính toán. Thay vì phải tính toán xác suất lặp lại cho từng điểm dữ liệu như K-Means++, K-Means$\|$ chỉ cần thực hiện một số lần lấy mẫu dựa trên chi phí ban đầu của việc phân cụm. Điều này không chỉ giảm bớt gánh nặng tính toán mà còn tăng cường hiệu suất của thuật toán, đặc biệt là trong các tình huống có tập dữ liệu lớn.

3.3. Một số thí nghiệm của Thuật toán K-Means++ và K-Mean$\|$

3.3.1. So sánh giữa K-Means và K-Means++

Chúng ta sẽ so sánh hiệu suất giữa K-Means và K-Means++ trên ba bộ dữ liệu khác nhau. Tập dữ liệu đầu tiên là tập dữ liệu nhân tạo Norm25 [15]. Hai tập dữ liệu còn lại là hai bộ dữ liệu thực, tập dữ liệu Cloud [17] và Intrusion [18]. Trong nghiên cứu [15], tác giả đã xem xét sự khác biệt giữa hai thuật toán trên ba trường hợp với các giá trị K là 10, 25 và 50. Mỗi trường hợp, tác giả đã thực hiện 20 lần chạy. Các bảng dưới đây ghi lại giá trị nhỏ nhất của hàm tiềm năng, giá trị trung bình của hàm tiềm năng và thời gian chạy tương ứng. Lưu ý rằng với K-Means, thời gian được

tính bằng giây, trong khi với *K*-Means++, tỉ lệ phần trăm cải thiện so với *K*-Means được tính trực tiếp theo công thức:

$$\left(1 - \frac{L(K\text{-Means++})}{L(K\text{-Means})}\right) 100\%.$$

Trong đó L(*K*-Means) và L(*K*-Means++) lần lượt là giá trị hàm mất mát thu được bởi thuật toán *K*-Means và *K*-Means++.

Bốn Bảng 3.1, 3.2, 3.3 và 3.4 thể hiện kết quả của thuật toán *K*-Means và *K*-Means++. Mỗi bảng gồm ba cột chính như sau:

- Cột đầu tiên hiển thị giá trị trung bình của hàm mất mát $E[L]$. Cột nhỏ bên trái tương ứng với giá trị trung bình của hàm mất mát của thuật toán *K*-Means, trong khi cột nhỏ bên phải biểu diễn tỷ lệ phần trăm cải thiện của giá trị trung bình của hàm mất mát nhận được từ thuật toán *K*-Means++ so với giá trị trung bình của hàm mất mát nhận được từ thuật toán *K*-Means++.

- Cột thứ hai hiển thị giá trị nhỏ nhất của hàm mất mát: $minL$. Cột nhỏ bên trái tương ứng với giá trị nhỏ nhất của hàm mất mát của thuật toán *K*-Means, trong khi cột nhỏ bên phải biểu diễn tỷ lệ phần trăm cải thiện của giá trị nhỏ nhất của hàm mất mát nhận được từ thuật toán *K*-Means++ so với giá trị trung bình của hàm mất mát nhận được từ thuật toán *K*-Means++.

- Cột thứ ba hiển thị giá trị trung bình của thời gian chạy thực: $E[T]$. Cột nhỏ bên trái tương ứng với giá trị trung bình của thời gian chạy của thuật toán *K*-Means, trong khi cột nhỏ bên phải biểu diễn tỷ lệ phần trăm cải thiện của giá trị trung bình của thời gian chạy của thuật toán *K*-Means++ so với giá trị trung bình của thời gian chạy của thuật toán *K*-Means++.

	$E[L]$		min L		$E[T]$	
k	K-Means	K-Means++	K-Means	K-Means++	K-Means	K-Means++
10	$1.365 \cdot 10^5$	8.47%	$1.174 \cdot 10^5$	0.93%	0.12	46.72%
25	$4.233 \cdot 10^4$	99.96%	$1.914 \cdot 10^4$	99.92%	0.90	87.79%
50	$7.750 \cdot 10^3$	99.81%	$1.474 \cdot 10^1$	0.53%	2.04	-1.62%

Bảng 3.1: Kết quả thực nghiệm trên tập dữ liệu Norm25 có 10000 điểm dữ liệu với số chiều của dữ liệu là 15 trong bài báo [15]

	$E[L]$		min L		$E[T]$	
K	K-Means	K-Means++	K-Means	K-Means++	K-Means	K-Means++
10	$7.921 \cdot 10^3$	22.33%	$6.284 \cdot 10^3$	10.37%	0.08	51.09%
25	$3.637 \cdot 10^3$	42.76%	$2.550 \cdot 10^3$	22.60%	0.11	43.21%
50	$1.867 \cdot 10^3$	39.01%	$1.407 \cdot 10^3$	23.07%	0.16	41.99%

Bảng 3.2: Kết quả thực nghiệm trên tập dữ liệu Cloud có 1024 điểm dữ liệu và chiều của dữ liệu là 15 trong bài báo [15].

	$E[L]$		min L		$E[T]$	
K	K-Means	K-Means++	K-Means	K-Means++	K-Means	K-Means++
10	$3.387 \cdot 10^8$	93.37%	$3.206 \cdot 10^8$	94.40%	63.94	44.49%
25	$3.149 \cdot 10^8$	99.20%	$3.100 \cdot 10^8$	99.32%	257.34	49.19%
50	$3.079 \cdot 10^8$	99.84%	$3.076 \cdot 10^8$	99.87%	917.00	66.70%

Bảng 3.3: Kết quả thực nghiệm trên tập dữ liệu Intrusion có 494019 điểm dữ liệu và chiều của dữ liệu là 35 trong bài báo [15].

	$E[L]$		min L		$E[T]$	
K	K-Means	K-Means++	K-Means	K-Means++	K-Means	K-Means++
10	$3.698 \cdot 10^4$	49.43%	$3.684 \cdot 10^4$	54.59%	2.36	69.00%
25	$3.288 \cdot 10^4$	88.76%	$3.280 \cdot 10^4$	89.58%	7.36	79.84%
50	$3.183 \cdot 10^4$	95.35%	$2.384 \cdot 10^4$	94.30%	12.20	75.76%

Bảng 3.4: Kết quả thực nghiệm trên tập dữ liệu SPAM có 4601 điểm dữ liệu và chiều của dữ liệu là 58 trong bài báo [15].

Nhận xét 3.3.1. *Từ các bảng này, chúng ta dễ dàng nhận thấy rằng kết quả thực nghiệm của K-Means++ vượt trội so với K-Means cả về giá trị của hàm L và thời gian chạy. Cụ thể, trong mọi trường hợp, K-Means++ luôn có giá trị của hàm mục tiêu cao hơn tối thiểu* 10% *so với K-Means. Đồng thời, thời gian tính toán của K-Means++ cũng nhỏ hơn từ* 20 *đến* 1000 *lần so với thuật toán K-Means gốc.*

3.3.2. So sánh Thuật toán K-Means$\|$ và một số thuật toán khác

So sánh So sánh Thuật toán K-Means$\|$ với Thuật toán Partition, K-Means: Trong phần này, tác giả trong bài báo [16] trình bày kết quả thực nghiệm so sánh thời gian chạy giữa thuật toán K-Means$\|$, thuật toán Partition [19] và phương pháp chọn tâm ngẫu nhiên của thuật toán K-Means. Kết quả được trình bày trong Bảng 3.5:

Random		$K = 500$	$K = 1000$
Partition		420.2	$1,021.7$
K-Means $\|\|$, $\ell =$	$0.1K$	230.2	222.6
	$0.5K$	69.0	46.2
	K	75.6	89.1
	$2K$	69.8	86.7
	$10K$	75.7	101.0

Bảng 3.5: Kết quả thực nghiệm trên tập dữ liệu KDDDCUP1999 với thời gian tính theo phút trong bài báo [16].

So sánh số ứng viên chọn làm tâm trước khi chọn chính xác K tâm giữa thuật toán K-Means$\|$ và Partition: Cả hai thuật toán K-Means $\|$ và Partition đều sẽ chọn ra một tập các ứng cử viên tâm với lực lượng lớn hơn nhiều so với K và sau đó sẽ phân cụm lại để lấy ra đúng K tâm. Bảng dưới đây so sánh số ứng cử viên được chọn làm tâm nhận được trước khi phân cụm lại của thuật toán K-Means $\|$ và Partition với số tâm cho trước là $K = 500$ và $K = 1000$ trên tập dữ liệu KDDDCUP1999. Kết quả được

		$K = 500$	$K = 1000$
Partition		9.5×10^5	1.47×10^6
K-Means $\|\|, \ell =$	$0.1K$	602	1,240
	$0.5K$	591	1,124
	K	1,074	2,234
	$2K$	2,321	3,604
	$10K$	9,116	7,588

Bảng 3.6: Bảng trên gồm số lượng tâm trước khi phân cụm lại với tập dữ liệu KDDDCUP1999 trong bài báo [16].

	$K = 20$	$K = 50$	$K = 100$
Random	176.4	166.8	60.4
K-Means++	38.3	42.2	36.6
K-Means $\|\|$ $\ell = 0.5k, r = 5$	36.9	30.8	30.2
K-Means $\|\|$ $\ell = 2k, r = 5$	23.3	28.1	29.7

Bảng 3.7: Bảng thể hiện số vòng lặp trong bài báo [16].

trình bày trong Bảng 3.6:

So sánh số vòng lặp của các thuật toán K-Means $\|\|$, K-Means++ và K-Means: Bảng 3.7 so sánh số vòng lặp của các thuật toán K-Means $\|\|$ với K-Means++ và K-Means với số tâm cho trước lần lượt là 20,50 và 100 trên tập dữ liệu KDDDCUP1999.

Nhận xét 3.3.2. *Từ các Bảng 3.5, 3.6 và 3.7, chúng ta có thể dễ dàng nhận thấy ưu điểm về tốc độ chạy và tốc độ hội tụ của các phương pháp chọn tâm trong K-Means $\|\|$. Trong Bảng 3.6, biểu diễn số tâm trước khi phân cụm lại của K-Means $\|\|$ và Partition. Như chúng ta có thể thấy, số lượng ứng cử viên để chọn làm tâm ít hơn rất nhiều so với Partition (gần 10^4 lần). Bảng cuối cùng thể hiện một ưu điểm khác của K-Means $\|\|$ với số vòng lặp Lloyd tương ứng là thấp hơn nhiều so với K-Means++ và phương pháp chọn tâm ngẫu nhiên.*

Chương 4

Một số thuật toán K-Means sử dụng hàm cosin

Trong phần này, chúng tôi sẽ đề xuất 03 phương pháp khởi tạo tâm ban đầu để cải thiện Thuật toán K-Means. Trước hết, chúng tôi sẽ nhắc lại hai phương pháp tọa độ hóa sử dụng bước đi ngẫu nhiên mà chúng tôi đã trình bày chi tiết ở Chương 2.

4.1. Độ tương đồng giữa các đỉnh sử dụng hàm cosin trong thị vô hướng

Trong [6], các tác giả đã có nhận xét rằng: nếu hai đỉnh i và j thuộc cùng một cộng đồng thì xác suất đi đến một đỉnh l bất kỳ sau t bước giữa chúng là gần bằng nhau, nghĩa là:

$$P_{il}^t \simeq P_{jl}^t, \text{ với mọi } l. \tag{4.1.1}$$

Từ đó, các tác giả đã định nghĩa khoảng cách giữa các đỉnh trên một đồ thị vô hướng như sau:

$$R_{ij}(t) := \|D^{-1/2}P_{i\bullet}^t - D^{-1/2}P_{j\bullet}^t\|.$$

Chúng ta thấy rằng định nghĩa khoảng cách này tương đương với việc tọa độ hóa mỗi đỉnh u của đồ thị là vector $D^{-1/2}P_{i\bullet}^t$, tức là:

Từ ý tưởng này chúng tôi tương ứng mỗi i với vector $D^{-1/2}P^t_{i\bullet}$,

$$C(i) := D^{-1/2}P^t_{i\bullet} = \left\{ d_1^{-1/2}P^t_{i1}, d_2^{-1/2}P^t_{i2}, ..., d_n^{-1/2}P^t_{in} \right\}, \tag{4.1.2}$$

trong đó, $C(i)$ ký hiệu là tọa độ của đỉnh i. Từ 4.1.1, chúng ta có thể nhận xét rằng nếu hai đỉnh i, j thuộc cùng một cộng đồng thì góc tạo bởi hai vector $C(i)$ và $C(j)$ sẽ tương đối nhỏ. Nói cách khác

$$cos(C(i), C(j)) \simeq 1. \tag{4.1.3}$$

Vì độ dài các vector có thể tính được và sử dụng hàm cosin cho ta một đánh giá chính xác bằng cách so sánh với 1 nên ta sẽ sử dụng (4.1.3) để xác định xem hai đỉnh có thuộc cùng một cộng đồng hay không.

4.2. Độ tương đồng giữa các đỉnh sử dụng hàm cosin trong đồ thị có hướng

Đối với một đồ thị có hướng liên thông mạnh G, như đang trình bày ở mục 2.2.2 với phân tích SVD của ma trận Diplacian của đồ thị G như trong [11]

$$\Gamma = \Phi^{1/2}(I - P)\Phi^{-1/2} = U\Sigma V^T. \tag{4.2.1}$$

Sau đó, các tác giả tọa độ hóa mỗi đỉnh u của đồ thị có hướng G như sau:

$$C(i) = \left(\phi_i^{-1/2}V_1(i), \ldots, \phi_i^{-1/2}V_k(i), \phi_i^{-1/2}U_1(i), \ldots, \phi_i^{-1/2}U_k(i) \right). \tag{4.2.2}$$

Tương tự như trường hợp của đồ thị vô hướng, chúng ta cũng có thể quan sát hai đỉnh i và j thuộc cùng một cộng đồng nếu góc được tạo bởi hai vector $C(i)$ và $C(j)$ là nhỏ, tức là tương đương với điều kiện:

$$cos(C(i), C(j)) \simeq 1.$$

4.3. Một số thuật toán K-Means cosin

Trước khi trình bày thuật toán cải thiện của K-Means dựa vào hàm Cosin chúng ta trình bày một số định nghĩa sau:

Định nghĩa 4.3.1. *Gọi $N_\theta(j)$ là tập các đỉnh mà vector tương ứng với nó tạo với vector tương ứng với j một góc đủ nhỏ, cụ thể ta có:*

$$N_\theta(j) = \{i \mid cos(C(i), C(j)) \geq \theta, i \in V\}.$$

Ta ký hiệu $d(j, C)$ là khoảng cách giữa đỉnh j tới tập C và được định nghĩa cụ thể như sau:

$$d(j, C) = \min_{i \in C} \{\|(C(i), C(j))\|\}.$$

Tương tự như các thuật toán K-Means $\|$ và thuật toán Partition, trước khi chọn ra chính xác K tâm, ta tìm được một danh sách các ứng cử viên tiềm năng. Sau đó, chúng ta sẽ chọn chính xác K tâm từ danh sách các ứng cử viên này. Trong không gian $\mathbb{R}^n$, ta thấy rằng các tâm cụm là các đỉnh với mật độ cao. Vì thế, nếu một đỉnh c là tâm, thì đại lượng $|N_\theta(c)|$ thường sẽ lớn. Dựa trên quan sát này, ta đề xuất thuật toán sau để tìm ra tập S các ứng cử viên để có thể chọn làm tâm.

Ý tưởng của thuật toán là chọn một danh sách các ứng cử viên tiềm năng như sau: Trước hết, ta chọn một đỉnh i_0 từ tập hợp các đỉnh V sao cho i_0 có số lượng đỉnh lân cận $|N_\theta(i_0)|$ lớn nhất. Tiếp theo, ta lặp lại quá trình này với tập hợp các đỉnh được cập nhật V được loại bỏ đi tất cả các đỉnh láng giềng của i_0 từ tập V. Quá trình lặp lại tiếp tục cho đến khi tập V không còn chứa bất kỳ đỉnh nào nữa. Từ đó, chúng ta có thể viết giả mã của thuật toán này như sau:

Thuật toán 4 Thuật toán tìm danh sách ứng cử viên làm tâm

Input: Graph G, θ.
Output: Tập S các ứng cử viên ưu tiên chọn làm tâm
Tính tọa độ $C(i)$ cho mỗi đỉnh $i \in V$: $C(i) \leftarrow$ được tính thông qua 4.1.2 nếu G là vô hướng thì ta sử dụng 2.2.3;
$S = \emptyset$;
$V = V(G)$;
while $V \neq \emptyset$ **do**
    $i_0 := \arg\max_{i \in V}\{|N_\theta(i)|\}$
    $S \leftarrow S \cup \{i_0\}$
    $V \leftarrow V \setminus N(i_0)$
end

Sau đây ta xét đến cách chọn các tâm, ta sẽ lựa chọn các tâm từ tập S dựa theo ý tướng rằng các tâm sẽ phải xa nhau và mỗi tâm c cần có $N_\theta(c)$ đủ lớn. Vì vậy, ta chọn tâm c_0 đầu tiên từ S sao cho $|N_\theta(c_0)|$ lớn nhất. Giả sử rằng ta đã tìm được ℓ tâm là $C = \{c_1, ..., c_\ell\}$ với $\ell < k$, tâm tiếp theo, ký hiệu là $c_{\ell+1}$, sẽ được chọn ít nhất thỏa mãn một trong hai tiêu chuẩn, đó là:

- Đảm bảo khoảng cách $d(c_{\ell+1}, C)$ đủ lớn.
- Có tập lân cận $N_\theta(c_{\ell+1})$ lớn.

Chúng tôi đề xuất ba thuật toán dựa trên hai tiêu chí trên. Thuật toán đầu tiên chọn ứng cử viên tiếp theo dựa trên việc chúng phải cách xa nhau nhất có thể. Thuật toán thứ hai chọn ứng cử viên tiếp theo dựa trên việc chúng có tập lân cận đủ lớn. Cuối cùng, trong thuật toán thứ ba, chúng tôi đề xuất kết hợp cả hai tiêu chí trên.

Sau đây là chi tiết thuật toán, chúng tôi sẽ mô tả giả mã của thuật toán đầu tiên. Chúng tôi gọi thuật toán này là là **Thuật toán K-Means Cosin 1**.

Thuật toán 5 Thuật toán K-Means Cosin 1

Input: Đồ thị G, số cụm K, tập S thu dược từ Thuật toán 4
Output: Kết quả phân cụm.
$c_1 \leftarrow S[0]$
$C \leftarrow \{c_1\}$
$\ell \leftarrow 1$
while $\ell < K$ **do**
  $\ell \leftarrow \ell + 1$
  $c_\ell \leftarrow \arg\max_{v \in S \setminus C} d(v, C)$
  $C \leftarrow C \cup \{c_\ell\}$
end
Thực hiện thuật toán K-Means với K tâm khởi tạo trong tập C.

Cho một tập S được chọn như trong Thuật toán 4, chúng ta thấy rằng các đỉnh trong tập S cũng tương đối xa nhau, nên ta tìm tâm tiếp theo $c_{\ell+1}$ phụ thuộc vào độ lớn của từ tập $N_\theta(c_{\ell+1})$. Thuật toán này được gọi là **Thuật toán K-Means Cosin 2**.

Thuật toán 6 Thuật toán K-Means Cosin 2

Input: Đồ thị G, số cụm K, tập S thu được từ Thuật toán 4
Output: Kết quả phân cụm
$c_1 \leftarrow S[0]$
$C \leftarrow \{c_1\}$
$\ell \leftarrow 1$
while $\ell < k$ **do**
  $\ell \leftarrow \ell + 1$
  $c_\ell \leftarrow S[\ell - 1]$
  $C \leftarrow C \cup \{c_\ell\}$
end
Thực hiện thuật toán K-Means với K tâm khởi tạo trong tập C.

Trong thuật toán thứ ba, việc chọn tâm tiếp theo $c_{\ell+1}$, chúng tôi sẽ tích hợp cả tiêu chuẩn về khoảng cách và độ lớn của lân cận. Bước đầu, với mỗi đỉnh v trong tập $S \setminus C$, ta tính đại lượng: $x_v := \frac{|N_\theta(v)|}{\sum_{u \in S \setminus C} |N_\theta(u)|}$, và $y_v := \frac{d^2(v,C)}{\sum_{u \in S \setminus C} d^2(u,C)}$. Chúng tôi đề xuất chọn $c_{\ell+1}$ cực đại hóa hàm $f(x_v, y_v) =$

$(x_v y_v)^{1/2}$. Từ đây ta có:

$$c_{\ell+1} = arg \max_{v \in S \setminus C} \left\{ \left(\frac{|N_\theta(v)|}{\sum_{u \in S \setminus C} |N_\theta(u)|} \frac{d^2(v, C)}{\sum_{u \in S \setminus C} d^2(u, C)} \right)^{1/2} \right\} \quad (4.3.1)$$

Ta gọi thuật toán tứ ba này là **Thuật toán K-Means Cosin 3**.

Thuật toán 7 Thuật toán K-Means Cosin 3

Input: Đồ thị G, số cụm K, tập S thu dược từ Thuật toán 4
Output: Kết quả phân cụm
$c_1 \leftarrow S[0]$
$C \leftarrow \{c_1\}$
$\ell \leftarrow 1$
while $\ell < k$ **do**
  $\ell \leftarrow \ell + 1$
  $c_\ell \leftarrow arg \max_{v \in S \setminus C} \left\{ \left(\frac{|N_\theta(v)|}{\sum_{u \in S \setminus C} |N_\theta(u)|} \frac{d^2(v,C)}{\sum_{u \in S \setminus C} d^2(u,C)} \right)^{1/2} \right\}$
  $C \leftarrow C \cup \{c_\ell\}$
end
Thực hiện thuật toán K-Means với K tâm khởi tạo trong tập C.

4.4. Một số thí nghiệm

4.4.1. Các mô hình đồ thị ngẫu nhiên và các tiêu chí đánh giá

Các mô hình đồ thị ngẫu nhiên

Đánh giá hiệu suất của một thuật toán phát hiện cộng đồng là một phần quan trọng và thu hút sự quan tâm của nhiều nhà nghiên cứu. Trong quá trình này, dữ liệu đóng một vai trò quan trọng và đôi khi có thể là một thách thức. Đối với các thí nghiệm, việc sử dụng dữ liệu thực tế là điều không thể tránh khỏi. Tuy nhiên, để mở rộng phạm vi của các nghiên cứu và thử nghiệm, việc tạo ra dữ liệu mô phỏng ngẫu nhiên cũng là một phương tiện quan trọng.

Các dữ liệu mô phỏng ngẫu nhiên cung cấp một cách tiếp cận tiện lợi để đánh giá hiệu suất của các thuật toán phát hiện cộng đồng. Đặc biệt, trong các dữ liệu này, chúng ta thường biết trước cấu trúc cộng đồng, điều này giúp tiện lợi trong việc đánh giá chất lượng của các thuật toán. Dưới đây là hai mô hình sinh dữ liệu đồ thị ngẫu nhiên mà chúng ta sẽ sử dụng trong quá trình này:

Mô hình phân vùng l-planted: Mô hình sinh đồ thị đầu tiên mà chúng tôi giới thiệu là mô hình phân vùng l-planted. Mô hình này sử dụng các tham số sau để tạo ra một đồ thị ngẫu nhiên với một số tính chất nhất định: số lượng nhóm l, số lượng đỉnh mỗi nhóm b, xác suất giữa các cụm p_{in} và xác suất trong cụm p_{out}:

- Độ trung bình của một đỉnh là $E[k] = p_{in}(g-1) + p_{out}g(l-1)$.
- Tất cả cộng đồng có kích thước như nhau.
- Tất cả các đỉnh có độ gần như nhau bởi vì mỗi cộng đồng có thể được coi như một đồ thị ngẫu nhiên được đề xuất bởi Erdos và Renyi. Trong những đồ thị ngẫu nhiên đó, mỗi cặp đỉnh được kết nối với xác suất bằng nhau p_{in} mà độc lập với các cặp khác.

Mô hình sinh đồ thị phân vùng Gauss: Mô hình sinh đồ thị tiếp theo là mô hình sinh đồ thị phân vùng Gauss, giúp khắc phục phần nào nhược điểm của mô hình phân vùng l-planted về phân bố độ của các đỉnh. Khác với mô hình phân vùng l-planted, kích thước của cộng đồng trong mô hình sinh đồ thị phân vùng Gauss là một biến ngẫu nhiên theo phân phối Gauss. Các tham số cần được xác định cho mô hình sinh đồ thị phân vùng Gauss là:

- Số lượng đỉnh trong đồ thị: N;

- Trung bình của kích thước cộng đồng: m và phương sai của kích thước cộng đồng: σ;
- Xác suất cạnh giữa p_{in} và trong cụm p_{out}.

Các tiêu chí đánh giá

Đầu tiên, chúng ta sẽ nhắc lại khái niệm về hàm modularity, một đại lượng quan trọng được định nghĩa trong [20], giúp chúng ta đánh giá chất lượng của một phân cụm. Đối với cặp đỉnh i và j, số cạnh kỳ vọng giữa chúng được tính bằng $d_i d_j/(2m)$, trong đó d_i là bậc của đỉnh i, d_j là bậc của đỉnh j, và m là tổng số cạnh trong đồ thị. Số cạnh thực tế giữa hai đỉnh i và j được biểu diễn bởi thành phần A_{ij} ở vị trí (i, j) trong ma trận kề A của đồ thị G. Do đó, chúng ta có thể tính hiệu số giữa số cạnh thực tế và số cạnh kỳ vọng, tức là $A_{ij} - d_i d_j/(2m)$.

Ký hiệu g_i là nhãn của cụm mà đỉnh i thuộc vào. Dựa trên những khái niệm này, chúng ta có thể định nghĩa modularity Q như sau:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{d_i d_j}{2m} \right] \delta_{g_i g_j} \tag{4.4.1}$$

khi $\delta_{g_i g_j}$ bằng 1 với $g_i = gj$ và bằng 0 nếu ngược lại. Hệ số $1/2m$ chỉ mang tính thuận tiện mà không ảnh hưởng đến việc tối ưu đại lượng Q. Nếu i vầ j là hai đỉnh cùng cộng đồng thì đại lượng $A_{ij} - \frac{d_i d_j}{2m}$ "nên" dương, và nếu đại lượng đó dương thật và ta gán nhãn cho i và j cùng cụm thì $\delta_{g_i g_j} = 1$, khi đó ta sẽ "được thưởng" một lượng dương lầ $A_{ij} - \frac{d_i d_j}{2m}$, từ đó dẫn đến việc hàm Q lớn sẽ tương ứng với việc phân cụm tốt.

Một tiêu chí khác để chúng tôi đánh giá các kết quả phân cụm đó là thông tin tương hỗ chuẩn hóa (normalized mutual information) mà từ giờ chúng tôi sẽ gọi tắt là NMI [21], đo độ tương đồng giữa phân cụm thật Y

(nhãn gán từ đầu) với phân cụm dự đoán C, cụ thể ta có:

$$\mathrm{NMI}(Y, C) = \frac{2 \times \mathrm{MI}(Y, C)}{\mathrm{H}(Y) + \mathrm{H}(C)}, \tag{4.4.2}$$

ở đây:

- $\mathrm{MI}(Y, C)$ là thông tin tương hỗ giữa Y và C,
- $\mathrm{H}(Y)$ và $\mathrm{H}(C)$ lần lượt là entropy của Y and C.

NMI là giá trị từ 0 tới 1, với giá trị cao thì tương ứng với chất lượng phân cụm tốt hơn. Đây là một tiêu chuẩn thường dùng để đánh giá một phân cụm.

4.4.2. Thí nghiệm trên đồ thị sinh ngẫu nhiên

Trong phần này, chúng tôi sử dụng hai mô hình sinh đồ thị ngẫu nhiên là phân vùng Gauss và phân vùng l-planted để tạo ra các đồ thị cho các thí nghiệm của mình. Để đánh giá tính hợp lý và hiệu quả của các thuật toán mà chúng tôi đã đề xuất, chúng tôi sẽ thực hiện 10 lần thử với mỗi thuật toán và so sánh kết quả thu được từ ba thuật toán đó với phân cụm được tạo ra từ cách tạo đồ thị, sử dụng tiêu chí đánh giá NMI. Cần lưu ý rằng, nếu giá trị NMI so sánh kết quả phân cụm của thuật toán và phân cụm gốc được sinh ra từ cách tạo đồ thị là 1, tức là kết quả phân cụm của thuật toán trùng khớp hoàn toàn với phân cụm được tạo ra từ cách tạo đồ thị, điều này ngụ ý rằng thuật toán cho kết quả tốt nhất.

Chúng tôi lưu ý rằng, trong các thí nghiệm ở phần này, khi áp dụng các Thuật toán K-Means Cosin, chúng tôi sẽ lấy giá trị $\theta = 0,9$ đối với các đồ thị được tạo ra với $p_{out} = 0,05$ và $p_{out} = 0,07$, và giá trị $\theta = 0,8$ đối với các đồ thị được tạo ra với $p_{out} = 0,01$ và $p_{out} = 0,03$ cho cả hai mô hình đồ thị ngẫu nhiên. Đồng thời, số cụm K sẽ bằng số cụm sinh ra bằng cách tạo đồ thị.

Thí nghiệm sử dụng Mô hình sinh đồ thị phân vùng Gauss

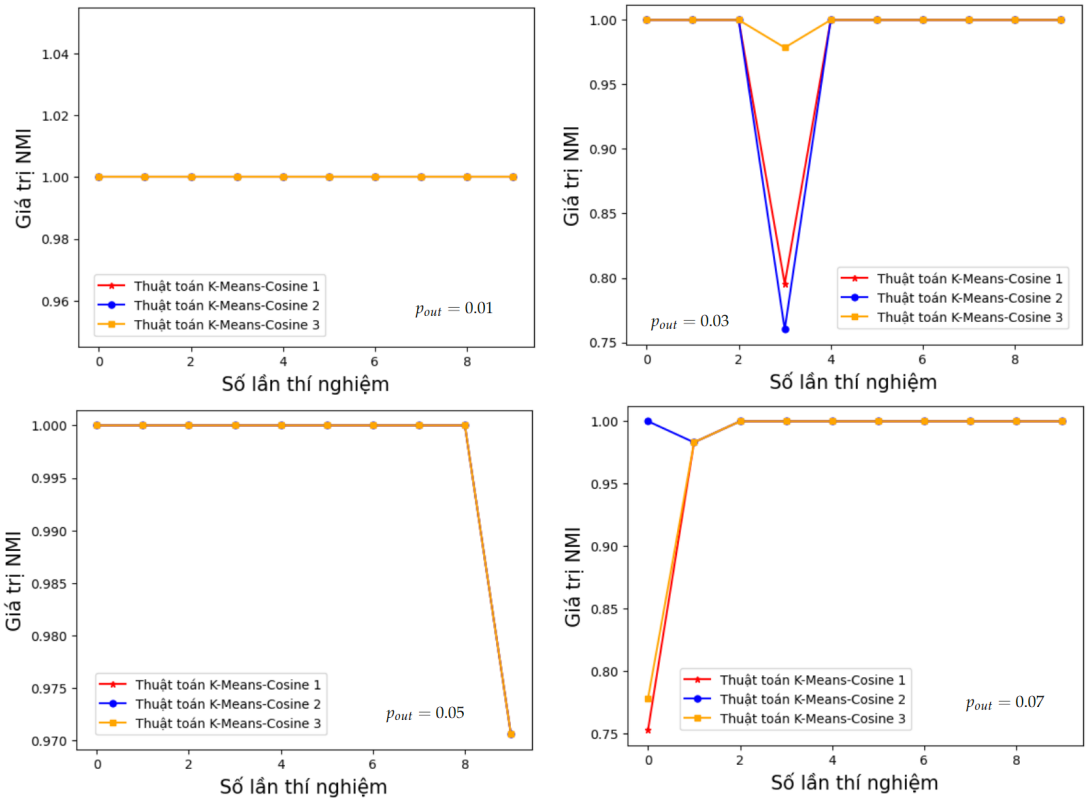
Trong thí nghiệm này, chúng tôi sử dụng mô hình sinh số ngẫu nhiên Gauss để tạo ra các đồ thị với số đỉnh dao động từ vài trăm đến hàng ngàn đỉnh. Đối với mỗi đồ thị, chúng tôi giữ $p_{in} = 0,7$ không đổi, và thay đổi giá trị của p_{out} trong các giá trị $0,01$, $0,03$, $0,05$, và $0,07$, tương ứng với các đồ thị có cấu trúc cộng đồng rõ ràng đến các đồ thị có cấu trúc cộng đồng không rõ ràng. Chúng tôi đặt phương sai của kích thước cộng đồng $\delta = 2,5$.

Thí nghiệm 1: sử dụng Mô hình sinh đồ thị phân vùng Gauss

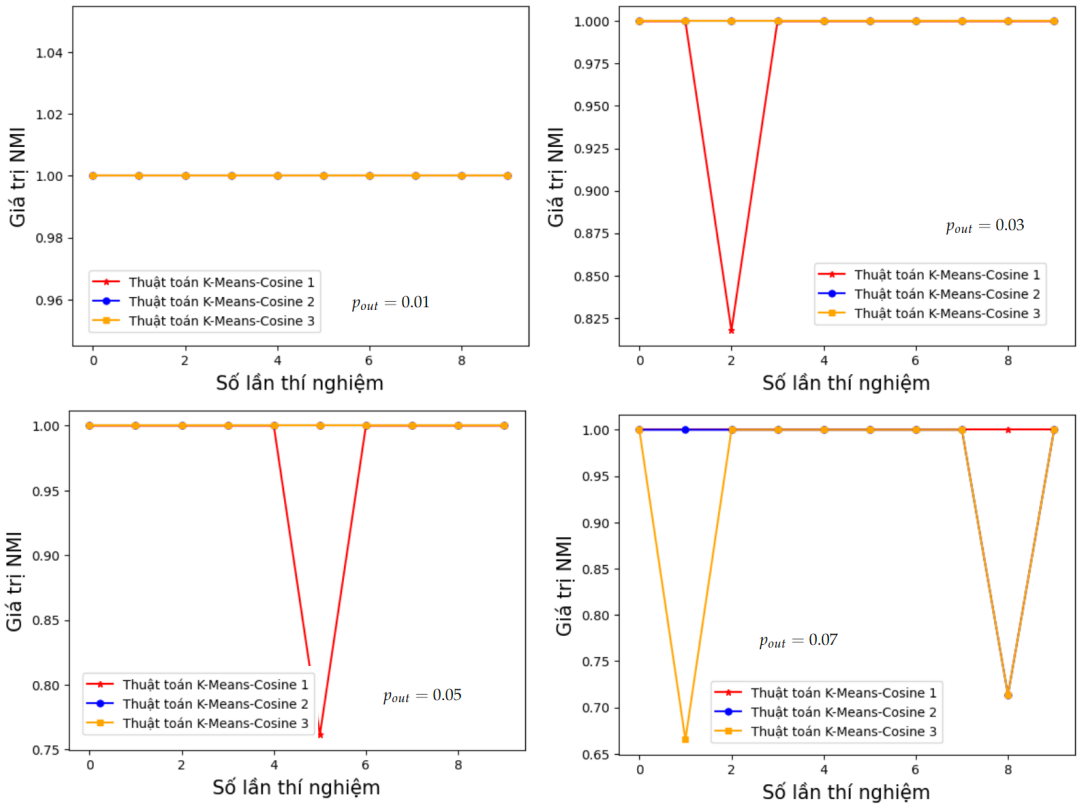
Trong thí nghiệm này, chúng tôi sử dụng mô hình sinh đồ thị phân vùng Gauss với một số đỉnh N và giá trị trung bình của kích thước cộng đồng m được lấy từ phân phối đều trong khoảng $N \in [300; 500]$, $m \in [80; 120]$. Chúng tôi biểu diễn các kết quả của NMI trong Hình 4.1. **Nhận xét:** Đối với đồ thị được sinh ra bằng mô hình phân vùng Gauss và có ít đỉnh, khi đồ thị có cấu trúc phân cụm rõ ràng thì cả 10 thí nghiệm của chúng tôi đều cho kết quả chính xác hoàn toàn (NMI=1), còn khi cấu trúc phân cụm không rõ ràng thì có 9 thí nghiệm cho kết quả chính xác hoàn toàn (NMI=1) và 1 trường hợp có độ chính xác cao.

Thí nghiệm 2: sử dụng Mô hình sinh đồ thị phân vùng Gauss:

Trong thí nghiệm này, chúng tôi sử dụng Mô hình sinh đồ thị phân vùng Gauss với một số đỉnh N, giá trị trung bình của kích thước cộng đồng m được lấy từ phân phối đều trong các khoảng tương ứng sau: $N \in [800; 1500]$, $m \in [150; 250]$. Chúng tôi biểu diễn các kết quả của NMI trong Hình 4.2. **Nhận xét:** Đối với đồ thị được sinh ra bằng mô hình phân vùng Gauss và có số đỉnh trung bình, khi đồ thị có cấu trúc phân cụm rõ ràng



Hình 4.1: Biểu đồ này minh họa giá trị NMI trong **Thí nghiệm 1**; mỗi biểu đồ là kết quả thử nghiệm trên 10 đồ thị được tạo ngẫu nhiên bằng cách sử dụng Mô hình sinh đồ thị phân vùng Gauss với các thông số là $N \in [300;500]$, $m \in [80;120]$.



Hình 4.2: Biểu đồ này minh họa giá trị NMI trong **Thí nghiệm 2**; mỗi biểu đồ là kết quả thử nghiệm trên 10 đồ thị được tạo ngẫu nhiên bằng cách sử dụng Mô hình sinh đồ thị phân vùng Gauss với các thông số là $N \in [800; 1500]$, $m \in [150; 250]$.

thì cả 10 thí nghiệm của chúng tôi đều cho kết quả chính xác hoàn toàn (NMI=1), còn khi cấu trúc phân cụm không rõ ràng thì có 8 hoặc 9 thí nghiệm cho kết quả chính xác hoàn toàn (NMI=1) và các trường hợp còn lại có độ chính xác cao.

Thí nghiệm 3: sử dụng Mô hình sinh đồ thị phân vùng Gauss:

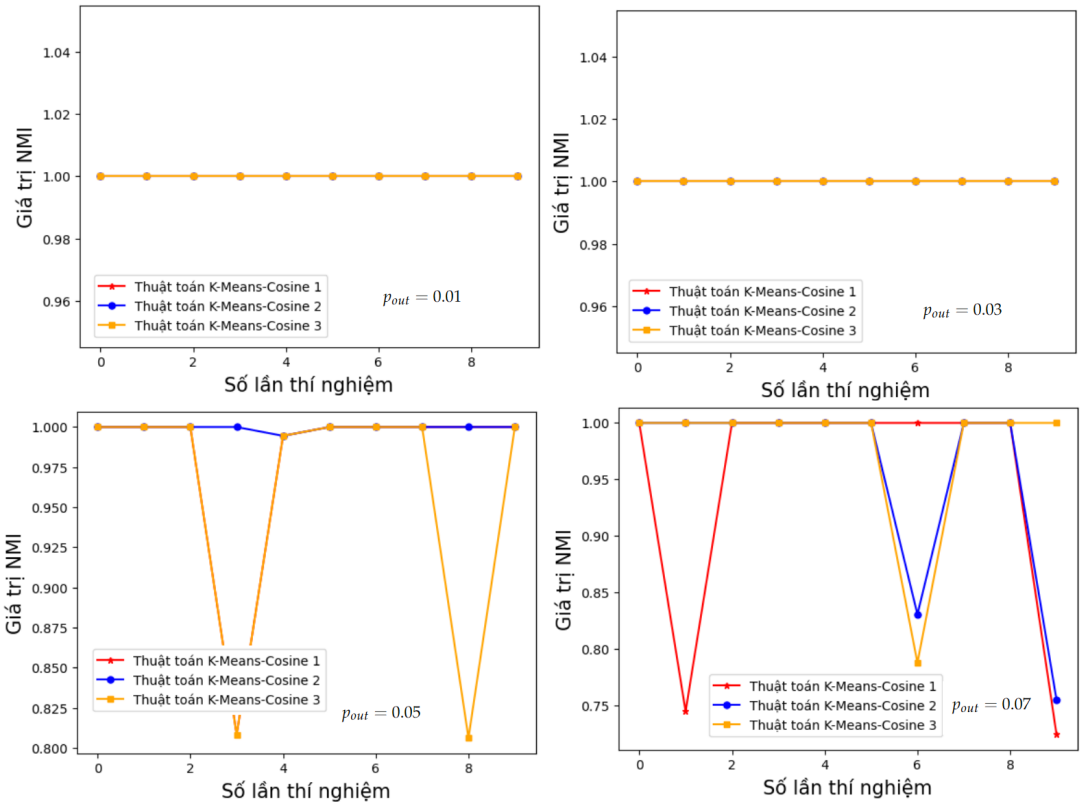
Trong thí nghiệm này, chúng tôi sử dụng Mô hình sinh đồ thị phân vùng Gauss với một số đỉnh N, giá trị trung bình của kích thước cộng đồng m được lấy từ phân phối đều trong các khoảng tương ứng sau: $N \in [2000; 3000]$, $m \in [300; 500]$. Chúng tôi biểu diễn các kết quả của NMI trong Hình 4.3. **Nhận xét:** Đối với đồ thị được sinh ra bằng mô hình phân vùng Gauss và có số đỉnh lớn, khi đồ thị có cấu trúc phân cụm rõ ràng thì cả 10 thí nghiệm của chúng tôi đều cho kết quả chính xác hoàn toàn (NMI=1), còn khi cấu trúc phân cụm không rõ ràng thì có 7 hoặc 8 thí nghiệm cho kết quả chính xác hoàn toàn (NMI=1) và các trường hợp còn lại có độ chính xác cao.

Thí nghiệm sử dụng Mô hình sinh đồ thị phân vùng l-planted:

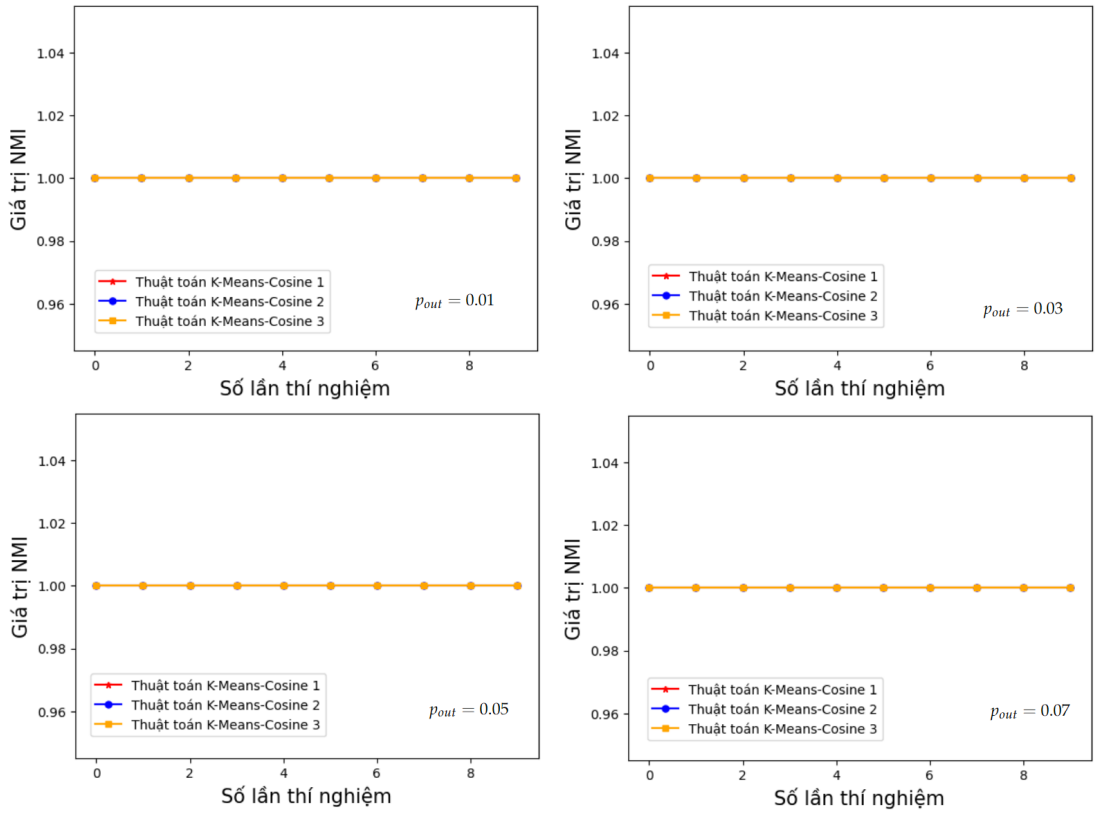
Với mô hình phân vùng l-planted, chúng tôi cũng cố định $p_{in} = 0.7$ và thay đổi các giá trị p_{out} trong các giá trị $0,01; 0,03; 0,05$ và 0.07, tương ứng với các đồ thị có cấu trúc cộng đồng rõ ràng đến các đồ thị có cấu trúc cộng đồng không rõ ràng.

Thí nghiệm 4: sử dụng Mô hình sinh đồ thị phân vùng l-planted:

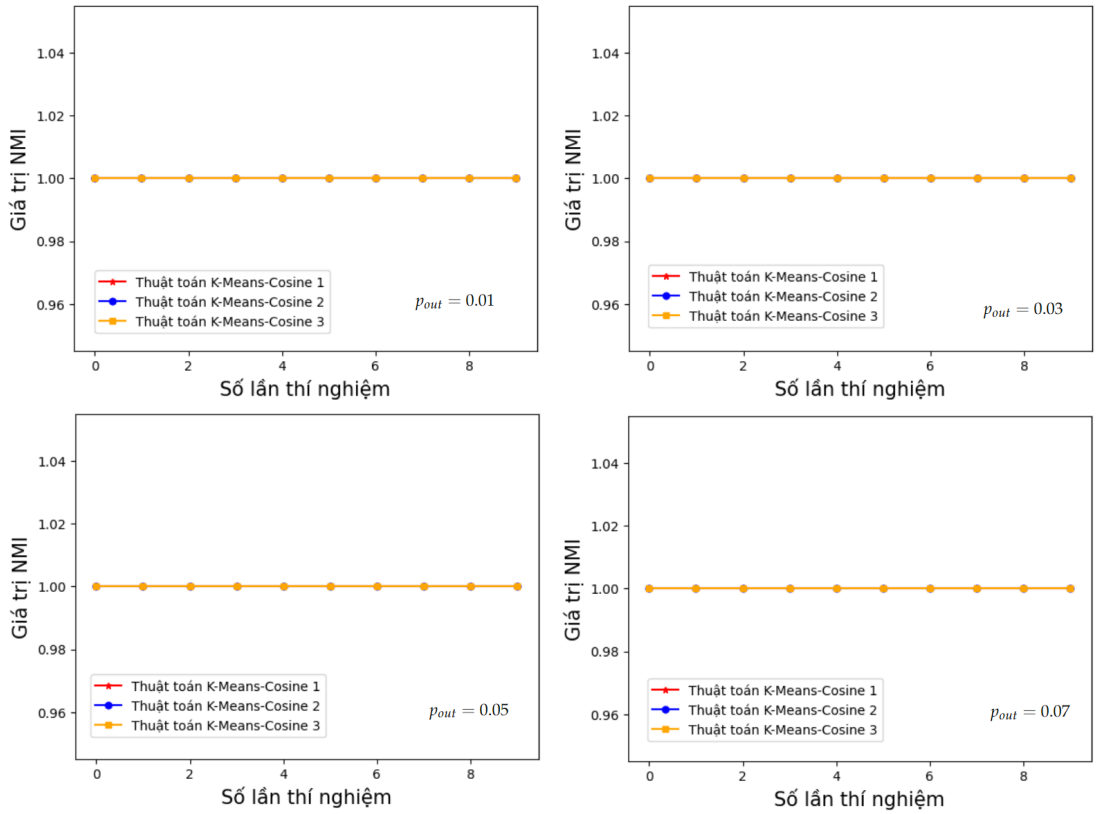
Sử dụng mô hình phân vùng l-planted với số cộng đồng l, kích thước của mỗi cộng đồng g được lấy từ phân phối đồng nhất trong các khoảng tương ứng sau: $g \in [100; 200]$, $l \in [5; 10]$. Chúng tôi biểu diễn các kết quả NMI trong Hình 4.4. **Nhận xét:** Đối với đồ thị được sinh ra bằng mô hình



Hình 4.3: Biểu đồ này minh họa giá trị NMI trong **Thí nghiệm 3**; mỗi biểu đồ là kết quả thử nghiệm trên 10 đồ thị được tạo ngẫu nhiên bằng cách sử dụng Mô hình sinh đồ thị phân vùng Gauss với các thông số là $N \in [2000; 3000]$, $m \in [300; 500]$.



Hình 4.4: Biểu đồ này minh họa giá trị NMI trong **Thí nghiệm 4**; mỗi biểu đồ là kết quả thử nghiệm trên 10 đồ thị được tạo ngẫu nhiên bằng cách sử dụng Mô hình sinh đồ thị phân vùng l-planted với các thông số là $g \in [100;200]$, $l \in [5;10]$.



Hình 4.5: Biểu đồ này minh họa giá trị NMI trong **Thí nghiệm 5**; mỗi biểu đồ là kết quả thử nghiệm trên 10 đồ thị được tạo ngẫu nhiên bằng cách sử dụng Mô hình sinh đồ thị phân vùng l-planted với các thông số là $g \in [10; 15]$, $l \in [200; 300]$.

phân vùng l-planted và có số đỉnh bé, khi đồ thị có cấu trúc phân cụm rõ ràng hay không rõ ràng thì cả 10 thí nghiệm của chúng tôi đều cho kết quả chính xác hoàn toàn (NMI=1).

Thí nghiệm 5: sử dụng Mô hình sinh đồ thị phân vùng l-planted:

Sử dụng mô hình phân vùng l-planted với số cộng đồng l, kích thước của mỗi cộng đồng g được lấy từ phân phối đồng nhất trong các khoảng tương ứng sau: $g \in [10; 15]$, $l \in [200; 300]$. Chúng tôi biểu diễn các kết quả NMI trong Hình 4.5. **Nhận xét:** Đối với đồ thị được sinh ra bằng mô hình

phân vùng l-planted và có số đỉnh lớn, khi đồ thị có cấu trúc phân cụm rõ ràng hay không rõ ràng thì cả 10 thí nghiệm của chúng tôi đều cho kết quả chính xác hoàn toàn (NMI=1).

4.4.3. Thí nghiệm trên dữ liệu thực

Trong phần này, chúng tôi thực hiện các thí nghiệm để so sánh hiệu quả của ba thuật toán *K*-Means Cosin mà chúng tôi đề xuất trong Phần 4.3 và hai thuật toán *K*-Means++ và *K*-Means $\|$ trên một số dữ liệu thực.

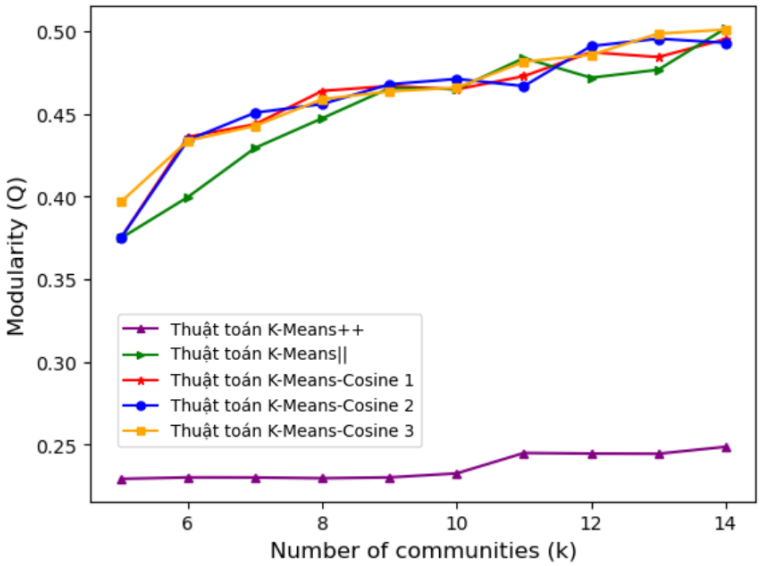
Đối với mỗi bộ dữ liệu, chúng tôi tiến hành thí nghiệm trên một loạt các giá trị của số cụm (K). Sau đó, với mỗi kết quả phân cụm thu được, chúng tôi tính giá trị modularity (sử dụng công thức (4.4.1)) tương ứng. Cuối cùng, chúng tôi biểu diễn các giá trị modularity đó trên một biểu đồ.

Chúng tôi lưu ý rằng, trong một số thí nghiệm, kết quả của thuật toán *K*-Means++ có thể kém hơn so với các thuật toán còn lại đáng kể. Do đó, khi vẽ biểu đồ, chúng tôi sẽ tạo hai biểu đồ: một biểu đồ bao gồm tất cả các thuật toán và một biểu đồ khác loại bỏ kết quả của thuật toán *K*-Means++ để dễ quan sát.

Thí nghiệm 1: Mạng "A Song of Ice and Fire": Đây là một đồ thị với đỉnh là các nhân vật trong tiểu thuyết viễn tưởng T "A Song of Ice and Fire" của George R. R. Martin [22]. Mỗi cạnh giữa hai nhân vật tương ứng một lần hai nhận vật đều đã đề cập đến người kia với hơn 50 từ; bội của cạnh thể hiện số lần xảy ra của hiện tượng này. Mạng này là một đồ thị gồm 796 đỉnh và 2823 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán *K*-Means Cosin của chúng tôi với hệ số $\theta = 0,5$. Kết quả thu được trình bày trong Hình 4.6.

Nhận xét: Đối với mạng "A Song of Ice and Fire", trong các giá trị K từ 5 đến 14 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết



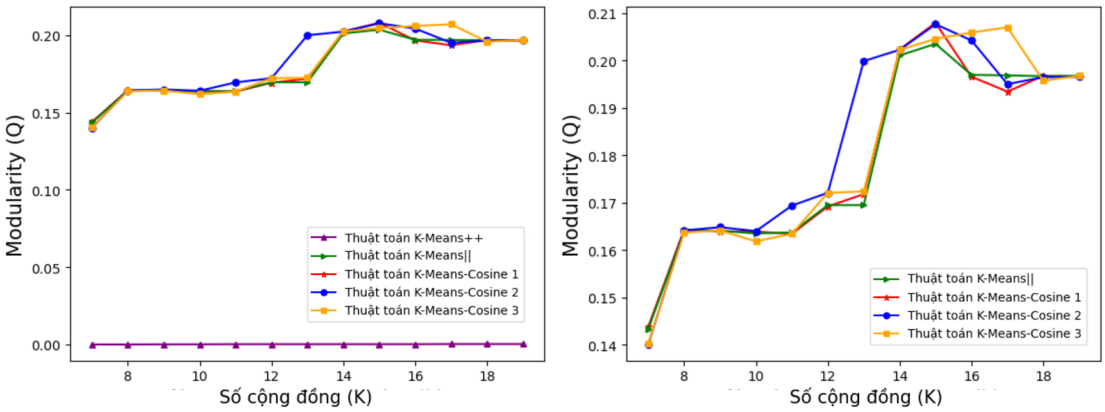
Hình 4.6: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**A Song of Ice and Fire**" khi áp dụng các thuật toán

quả tốt hơn thuật toán K-Means++ và hầu hết các trường hợp cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means ||.

Thí nghiệm 2: Mạng "DNC co-recipients": Đây là đồ thị vô hướng gồm những người nhận cùng một email trong vụ rò rỉ email của ủy ban quốc gia Đảng dân chủ năm 2016 [22]. Một kết xuất email của DNC đã bị rò rỉ vào năm 2016 và tập dữ liệu này chứa những người từ kết xuất đó dưới dạng các đỉnh và một lợi thế khi hai người nhận được cùng một email, tức là khi hai người nằm trong danh sách người nhận của cùng một email. Nhiều cạnh biểu thị nhiều email. Mạng này là một đồ thị gồm 906 đỉnh và 10429 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán K-Means Cosin của chúng tôi với hệ số $\theta = 0,5$. Kết quả thu được trình bày trong Hình 4.7

Nhận xét: Đối với mạng "DNC co-recipients", trong các giá trị K từ 7 đến 19 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means++ và hầu hết các trường hợp cả ba thuật toán



Hình 4.7: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**DNC co-recipients**" khi áp dụng các thuật toán

của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means ||.

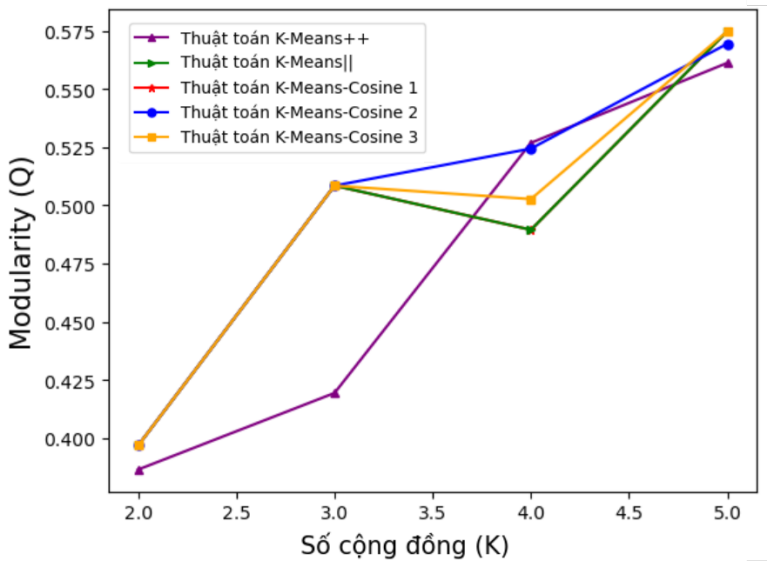
Thí nghiệm 3: Mạng "American football": Đồ thị này chứa các đỉnh là các trận đấu của cao đẳng Division IA colleges trong giải thu 2000." Kết quả hay thông tin sân khách sân nhà không có trong tập dữ liệu gốc của đồ thị này. Mạng này là một đồ thị gồm 115 đỉnh và 613 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán *K*-Means Cosin của chúng tôi với hệ số $\theta = 0,8$. Kết quả thu được trình bày trong Hình 4.8.

Nhận xét: Đối với mạng "American football", trong các giá trị K từ 2 đến 5 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết quả tương đương thuật toán K-Means++ và tất cả trường hợp cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn hoặc tương đương với thuật toán K-Means ||.

Thí nghiệm 4: Mạng "Celegans Metabolic": Đây là mạng lưới trao đổi chất của Caenorhabditis elegans từ dự án DIMACS10, được mua lại từ trang web của Alex Arena, lấy từ bài viết của Jeong và cộng sự, dựa trên dữ liệu từ cơ sở dữ liệu WIT. Mạng này là một đồ thị gồm 453 đỉnh và 2025 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán *K*-Means Cosin của



Hình 4.8: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**American football**" khi áp dụng các thuật toán

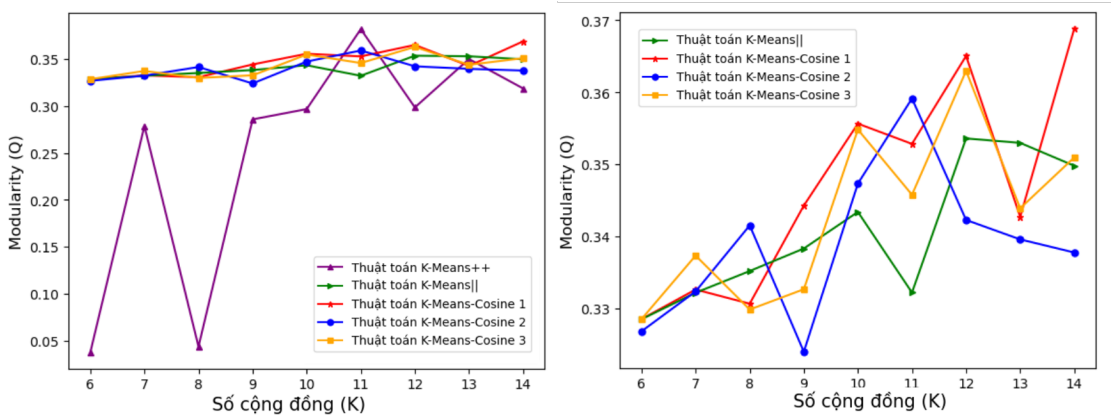
chúng tôi với hệ số $\theta = 0,8$. Kết quả thu được trình bày trong Hình 4.9.

Nhận xét: Đối với mạng "Celegans Metabolic", trong các giá trị K từ 6 đến 15 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means++ và hầu hết các trường hợp cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means $\|$.

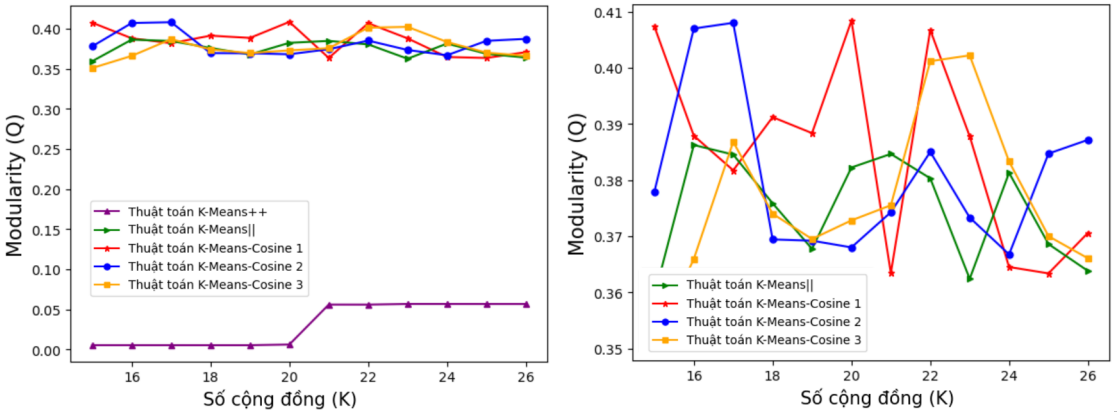
Thí nghiệm 5: Mạng "Yeast": Mạng vô hướng này chứa các tương tác protein có trong nấm men. Nghiên cứu cho thấy rằng các protein có hàm lượng cao có vai trò quan trọng hơn đối với sự tồn tại của nấm men so với các protein khác. Mỗi đỉnh đại diện cho một protein và mỗi cạnh đại diện cho sự tương tác trao đổi chất giữa hai protein. Mạng này là một đồ thị gồm 1870 đỉnh và 2277 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán K-Means Cosin với hệ số $\theta = 0,4$. Kết quả thu được trình bày trong Hình 4.10.

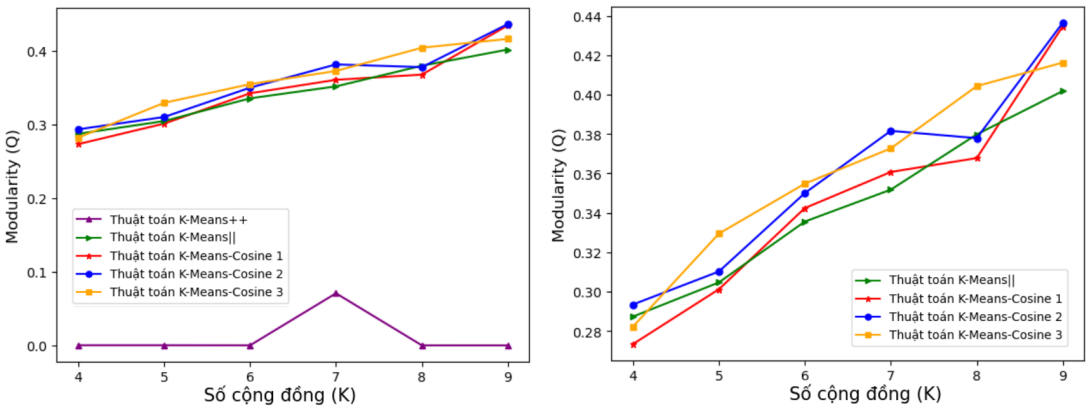
Nhận xét: Đối với mạng "Yeast", trong các giá trị K từ 15 đến 26 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật



Hình 4.9: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**Celegans Metabolic**" khi áp dụng các thuật toán



Hình 4.10: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**Yeast**" khi áp dụng các thuật toán



Hình 4.11: Bảng so sánh các giá trị modularity tương ứng với các phân cụm của mạng "**Similarities (DBpedia)**" khi áp dụng các thuật toán

toán K-Means++ và hầu hết các trường hợp cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means ||.

Thí nghiệm 6: Mạng "Similarities (DBpedia)": Đây là đồ thị xây dựng từ DBpedia. Nó chứa các liên kết "tương tự" giữa các trang của Wikipedia. Đây là một đồ thị vô hướng và các cạnh không có bội. Mạng này là một đồ thị gồm 430 đỉnh và 565 cạnh.

Đối với mạng này, chúng tôi áp dụng ba thuật toán K-Means Cosin với hệ số $\theta = 0,4$. Kết quả thu được trình bày trong Hình 4.11.

Nhận xét: Đối với mạng "Similarities (DBpedia)", trong các giá trị K từ 4 đến 9 mà chúng tôi khảo sát, cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means++ và hầu hết các trường hợp cả ba thuật toán của chúng tôi đều cho kết quả tốt hơn thuật toán K-Means ||.

4.4.4. Nhận xét về các thí nghiệm

Thông qua các thí nghiệm trên các đồ thị được sinh ngẫu nhiên bằng hai mô hình sinh đồ thị ngẫu nhiên và các dữ liệu thực, chúng ta có các nhận xét sau:

- Đối với các đồ thị được sinh bằng mô hình sinh đồ thị phân vùng l-planted, các thuật toán *K*-Means Cosin của chúng tôi cho kết quả phân cụm trùng khớp hoàn toàn (NMI=1) với phân cụm gốc khi tạo đồ thị.

- Đối với các đồ thị được sinh bằng mô hình sinh đồ thị phân vùng Gauss, các thuật toán *K*-Means Cosin của chúng tôi cho kết quả phân cụm trùng khớp hoàn toàn (NMI=1) với phân cụm gốc khi tạo đồ thị khi p_{out} bé, tức là các đồ thị có cấu trúc cộng đồng rõ ràng. Và khi p_{out} lớn, tức là các đồ thị có cấu trúc cộng đồng không quá rõ ràng thì các thuật toán của chúng tôi cũng phần lớn cho kết quả chính xác, và còn lại cho kết quả với dộ chính xác cao.

- Khi so sánh các thuật toán *K*-Means Cosin của chúng tôi với hai thuật toán *K*-Means++, *K*-Means $\|$ trên các dữ liệu thực, thì kết quả phân cụm của các thuật toán của chúng tôi có kết quả tương đương hoặc tốt hơn.

Qua đó, chúng ta có thể thấy rằng các thuật toán *K*-Means Cosin của chúng tôi có độ chính xác cao và là những thuật toán đáng được quan tâm.

Kết luận

Trong luận văn này, chúng tôi đã nghiên cứu và tìm hiểu ứng dụng của thuật toán K-Means vào các mạng lớn. Cụ thể, chúng tôi tôi đã nghiên cứu và tìm hiểu các vấn đề sau:

- Trong Chương 1: Chúng tôi đã trình bày lại những kiến thức chuẩn bị như: các kiến thức cơ sở về lý thuyết đồ thị, bước đi ngẫu nhiên trên đồ thị, lý thuyết về khoa học mạng, cộng đồng mạng...
- Trong chương 2, chúng tôi đã tìm hiểu và trình bày lại các phương pháp toạ độ hoá các đỉnh của đồ thị vô hướng và có hướng như: phương pháp tọa độ hóa bằng bước đi ngẫu nhiên cũng như các phương pháp tọa độ hóa lấy ý tưởng từ những phương pháp giảm chiều của dữ liệu như LLE, PCA, hay phương pháp phổ.
- Trong chương 3, chúng tôi đã trình bày lại thuật toán K-Means và hai phiên bản cải thiện của nó là K-Means++ và K-Means $\|$.
- Ở Chương 4, Chúng tôi đã đề xuất ba thuật toán K-Means-Cosin là các phiên bản cải thiện của thuật toán K-Means áp dụng cho mạng. Cụ thể, chúng tôi đã coi mỗi đỉnh của đồ thị là một véc tơ trong không gian $\mathbb{R}^n$ và sử dụng hàm cosin để đo lường các đỉnh cùng cụm hay khác cụm. Từ đó chúng tôi đưa ra các phương pháp khởi tạo tâm ban đầu tương ứng với các thuật toán K-Means cosin mà chúng tôi đã đề xuất.

- Cuối cùng, chúng tôi đã thực hiện các thử nghiệm trên cả các đồ thị được sinh bằng các mô hình sinh đồ thị ngẫu nhiên là mô hình phân vùng l- planted và Gauss cũng như thực hiện các thí nghiệm so sánh các thuật toán của chúng tôi với thuật toán *K*-Means++ và *K*-Means || trên một số dữ liệu thực.

Trong thời gian tới, chúng tôi sẽ tiếp tục nghiên cứu về cách chọn tâm ban đầu cho thuật toán *K*-Means. Đặc biệt, chúng tôi sẽ nghiên cứu, phát triển phương pháp chọn tâm mà chúng tôi đã sử dụng ở Chương 4, để nhưng tâm tiếp theo được chọn thỏa mãn vừa xa các tâm cũ vừa là tâm của một cụm đủ lớn.

Tài liệu tham khảo

[1] Flake, Gary William, Steve Lawrence, C. Lee Giles and Frans Coetzee. “Self-Organization and Identification of Web Communities.” Computer 35 (2002): 66-71.

[2] Kleinberg, Jon and Steve Lawrence. “The Structure of the Web.” Science 294 (2001): 1849 - 1850.

[3] Ravasz, Erzsébet, Audrey Somera, D. A. Mongru, Zoltán N. Oltvai and Albert-László Barabási. “Hierarchical Organization of Modularity in Metabolic Networks.” Science 297 (2002): 1551 - 1555.

[4] Ngô Đắc Tân. "Lý thuyết tổ hợp và đồ thị". 2004. NXB Đại học Quốc gia Hà Nội.

[5] Ulrike von Luxburg. "A Tutorial on Spectral Clustering". Springer, (2007).

[6] Pons, Pascal and Matthieu Latapy. “Computing Communities in Large Networks Using Random Walks”. Computer and Information Sciences - ISCIS (2005): 284–293.

[7] Raphael Michele Coscia. "The Atlas for the Aspiring Network Scientist". arXiv:2101.00863v2. (2021).

[8] Katharina A. Zweig. "Network Analysis Literacy". Springer Vienna, (2016).

[9] Brandes, Ulrik, Garry Robins, Ann McCranie and Stanley Wasserman. “What is network science?” Network Science 1 (2013): 1 - 15.

[10] Ghojogh, Benyamin, Mark Crowley, Fakhri Karray and Ali Ghodsi. “Elements of Dimensionality Reduction and Manifold Learning.” Elements of Dimensionality Reduction and Manifold Learning (2023).

[11] Dang, T.D., Do, D.H., Phan, T.H.. "Community detection in directed graphs using stationary distribution and hitting times methods." Social Network Analysis and Mining, 13(2023), 1-30.

[12] Nguyễn Xuân Dũng. "Nghiên cứu các thuật toán rút gọn đồ thị và ứng dụng để phát triển cộng đồng trên mạng xã hội". Luận văn tiến sĩ, 2021.

[13] Newman, Mark E. J.. and M. Girvan. "Finding and evaluating community structure in networks". Phys. Rev. E 69 (2004) 026–113.

[14] Li, Yanhua and Zhi-Li Zhang. “Digraph Laplacian and the Degree of Asymmetry.” Internet Mathematics 8 (2012): 381 - 401.

[15] Arthur, David and Sergei Vassilvitskii. “k-means++: the advantages of careful seeding.” ACM-SIAM Symposium on Discrete Algorithms (2007).

[16] Bahmani, Bahman, Benjamin Moseley, Andrea Vattani, Ravi Kumar and Sergei Vassilvitskii. “Scalable K-Means++.” Proc. VLDB Endow. 5 (2012): 622-633.

[17] Philippe Collard's. Cloud cover database.ftp://ftp.ics.uci.edu/pub/machine-learningdatabases/undocumented/taylor/cloud.data.

[18] KDD Cup 1999dataset. http://kdd.ics.uci.edu//databases/kddcup99/kddcup99.ht

[19] N. Ailon, R. Jaiswal, and C. Monteleoni. Streaming k-means approximation. In NIPS, pages 10–18, 2009.

[20] Newman, Mark E. J.. "Spectral methods for network community detection and graph partitioning." Physical review. E, Statistical, nonlinear, and soft matter physics 88 4 (2013): 042822 .

[21] Kvålseth, Tarald O.. "Entropy and Correlation: Some Comments." IEEE Transactions on Systems, Man, and Cybernetics 17 (1987): 517-519.

[22] Kunegis, Jérôme. "KONECT: the Koblenz network collection." Proceedings of the 22nd International Conference on World Wide Web (2013).