UMAP: Lý thuyết và ứng dụng

Nguyễn Minh Ánh Nguyệt - 20110012

Ngày 18 tháng 6 năm 2023

Mục lục

1 Giới thiệu

Trong lĩnh vực khoa học dữ liệu, giảm chiều từ lâu đã đóng một vai trò quan trọng, đặc biệt là trong mảng trực quan hóa và tiền xử lý. Mục tiêu của nó là tìm ra các đặc trưng ẩn của bộ dữ liệu trong số chiều thấp hơn so với số chiều ban đầu. Trước đây, các thuật toán giảm chiều được phát triển chủ yếu được chia thành hai loại: dựa trên phương pháp phân tích thành tích các ma trận như phân tích thành phần chính (PCA), phân tích ma trận không âm (NMF), và sử dụng đồ thị các điểm gần nhất như t-SNE. Tuy nhiên, khi số chiều và số lượng dữ liệu tăng lên, vẫn còn tồn đọng rất nhiều khó khăn, đặc biệt là mất mát thông tin. Các thuật toán truyền thống thường chỉ giữ được cấu trúc địa phương hoặc cấu trúc toàn cục nên không thể giải quyết được vấn đề này. Để khắc phục chúng, UMAP đã được phát minh và cải tiến. UMAP đạt được kết quả giảm chiều dữ liệu với độ chính xác cao hơn, giảm thiểu sự mất mát thông tin và giữ được cấu trúc địa phương lẫn toàn cục của dữ liệu. UMAP dần trở thành một công cụ hiệu quả cho việc giảm chiều dữ liệu trong các dự án công nghiệp hiện nay.

Nhận thức được lợi ích của UMAP, bài tiểu luận khai thác về lý thuyết và ứng dụng của thuật toán này để hiểu rõ vì sao nó được áp dụng rộng rãi trong nhiều lĩnh vực và nhiều loại dữ liệu khác nhau như vậy. Cụ thể, chúng ta lần lượt tìm hiểu 5 phần: giới thiệu, một số kiến thức chuẩn bị, ý tưởng của UMAP, thuật toán UMAP và ứng dụng. Đầu tiên, một số kiến thức toán cần thiết sẽ được trình bày trong phần 2. Đây là các khái niệm khá trừu tượng, tuy nhiên nó đóng vai trò lớn trong việc tạo ra phương pháp tối ưu này. Sau đó, chúng ta sẽ tìm hiểu ý tưởng và cách mà UMAP đã xây dựng lên thuật toán ở phần 3 và tổng hợp các kiến thức trên để tạo nên các bước

triển khai của UMAP ở phần 4. Cuối cùng, chúng ta sẽ khai thác một ứng dụng phổ biến của UMAP trong trực quan hóa dữ liệu trong phần 5. Cụ thể, với một bộ dữ liệu văn bản, việc chọn lựa một phương pháp nhúng trong rất nhiều phương pháp là một điều khá khó. Vì vậy, UMAP đã giúp chúng ta giảm chiều và trực quan hóa để tiện cho việc đánh giá độ chính xác của từng phương pháp. Từ đó có sự chọn lựa phù hợp.

2 Một số kiến thức chuẩn bị

Trong phần này, ta chỉ trình bày các khái niệm vừa đủ để có thể hiểu các kết quả muốn nói gì, ngoài ra, phần lớn ta hướng tới việc trình bày ý tưởng hơn là đi vào chi tiết toán học.

2.1 Đa tạp Riemann

Trước tiên, ta sẽ trình bày nhanh một số khái niệm về tôpô, đa tạp và đa tạp Riemann.

Định nghĩa 2.1. Một **tôpô** trên tập hợp X là một họ τ các tập con của X thỏa mãn:

- Tập $\varnothing$ và X là các phần tử của τ.
- Hội các phần tử của τ cũng là một phần tử của τ.
- Giao hữu hạn các phần tử của τ cũng là một phần tử của τ.

Các phần tử của τ được gọi là các **tập mở**. Một tập hợp X cùng với một tôpô τ được gọi là **không gian tôpô**, ký hiệu bởi (X, τ). Ta thường ghi nhanh là X nếu không cần thiết phải làm rõ tôpô đang được đề cập. Các phần tử của X còn được gọi là **điểm** và các tập con của X chứa một tập mở còn được gọi là **lân cận**.

Ví dụ 2.2. Không gian mêtríc là không gian tôpô với các tập mở theo nghĩa mêtríc. Vì vậy, các tập con của không gian Euclid $\mathbb{R}^n$ đều là không gian tôpô với mêtríc cảm sinh.

Nhắc lại, nếu $U \subset \mathbb{R}^k$ là một tập mở thì $f : U \to \mathbb{R}^l$ được gọi là **trơn** nếu f và tất cả các các đạo hàm riêng mọi cấp của f tồn tại và liên tục. Tổng quát hơn cho miền xác định bất kỳ, ta có định nghĩa sau.

Định nghĩa 2.3. Cho $X \subset \mathbb{R}^k$ bất kỳ và $f : X \to \mathbb{R}^l$. Ta nói f **trơn** nếu với mỗi $x \in X$, tồn tại tập mở $U \subset \mathbb{R}^k$ chứa x và ánh xạ trơn $F : U \to \mathbb{R}^l$ sao cho $F|_{U \cap X} = f|_{U \cap X}$.

Như vậy, ta có thể trình bày được tổng quát của khái niệm về phép vi phôi (hay còn gọi là phép đổi biến) trong Giải tích 3.

Định nghĩa 2.4. Cho $X \subset \mathbb{R}^k, Y \subset \mathbb{R}^l$ và $f : X \to Y$. Ánh xạ f được gọi là **phép vi phôi** nếu f là song ánh, trơn với hàm ngược f^{-1} cũng trơn. Nếu f là phép vi phôi thì ta nói X và Y **vi phôi** với nhau.

Giống các khái niệm về sự giống nhau khác, chẳng hạn như song ánh hay đẳng cấu, hai không gian tôpô được xem là "như nhau ở cấu trúc trơn" nếu chúng vi phôi với nhau.

Định nghĩa 2.5. Cho $M \subset \mathbb{R}^k$ bất kỳ. Ta nói M là **đa tạp trơn** m-chiều với $m \in \mathbb{Z}^+$, nếu với mọi điểm $x \in M$, có một lân cận của M chứa x mà vi phôi với $\mathbb{R}^m$.

Ví dụ 2.6. $\mathbb{R}^n$ là đa tạp n-chiều. Mặt cầu $S^2 = \{x \in \mathbb{R}^3 \mid \|x\|_2 = 1\}$ là một đa tạp trơn 2-chiều, về mặt trực quan, mỗi người ở trên mặt cầu nhưng ta đều "cảm nhận" nơi chúng ta đang đứng giống như mặt phẳng $\mathbb{R}^2$.

Tiếp theo, ta quan tâm đến hình học của đa tạp trơn. Một vấn đề cổ điển là một đường đi trên đó thì có thể định nghĩa được chiều dài hay không? Như đã biết khi học về tích phân đường, chiều

dài của một đường đi được tham số hoá bởi $\gamma : [a, b] \to \mathbb{R}^m$ được cho bởi công thức

$$L[\gamma] = \int_a^b \|\gamma'(t)\|_2 \, dt.$$

Xét bài toán tương tự, giả sử có hai điểm p và q trong đa tạp trơn m-chiều M. Giả sử có tham số hoá $\gamma : [a, b] \to M$ sao cho $\gamma(a) = p, \gamma(b) = q$. Ta muốn định nghĩa phiếm hàm $L[\gamma]$ tương tự trên. Vậy ta cần hiểu $\gamma'(t)$ là gì và chuẩn trên không gian chứa $\gamma'(t)$? Trong [8, p.163], việc định nghĩa đạo hàm của một ánh xạ trơn giữa hai đa tạp là làm được và đạo hàm này cũng có tập xác định và tập giá trị là các không gian véctơ như hàm nhiều biến trên $\mathbb{R}^n$.

Định nghĩa 2.7. Cho $M \subset \mathbb{R}^k$ là một đa tạp trơn m-chiều. Giả sử điểm $x \in M$ có một lân cận được tham số hóa bởi φ, với $\varphi(u) = x$. **Không gian tiếp xúc** của M tại x, ký hiệu là TM_x, là không gian véctơ con m-chiều của $\mathbb{R}^k$ sinh bởi các véctơ $\frac{\partial \varphi}{\partial u_i}(u), i = \overline{1, m}$.

Ở đây, ta thừa nhận việc TM_x được định nghĩa tốt, không phụ thuộc vào cách chọn tham số hoá φ cũng như số chiều của TM_x là m. Bây giờ, nếu $f : M \to N$ là ánh xạ trơn và $x \in M$, khi đó, tồn tại F là một mở rộng trơn như 2.3 và ta định nghĩa được $df_x = dF_x|_{TM_x} : TM_x \to TN_{f(x)}$, trong đó lưu ý là F là hàm trơn trên một tập mở của $\mathbb{R}^k$ nên dF_x là đạo hàm Fréchét theo nghĩa đã biết. Ta thừa nhận ánh xạ đạo hàm df_x được định nghĩa tốt. Như vậy, đạo hàm của ánh xạ trơn giữa hai đa tạp thực sự là một mở rộng các khái niệm đã biết.

Nhận xét. Đoạn thẳng $[a, b]$ không thật sự là một đa tạp trong $\mathbb{R}$ mà nó là một đa tạp có biên. Ở đây, ta không đi quá sâu về chi tiết này.

Như vậy, việc còn lại là định nghĩa một chuẩn trong không gian tiếp xúc TM_x. Và đây chính là lí do tại sao ta lại cần mêtríc Riemann. Một **mêtríc Riemannn** g trên M là "một cách chọn trơn" cho tích trong g_x trên TM_x với mỗi $x \in M$. Một đa tạp trơn M cùng với mêtríc Riemann trên nó được gọi là **đa tạp Riemann** (M, g). Bây giờ, ta đã có thể định nghĩa được độ dài của một đường đi giữa hai điểm trong một đa tạp Riemann.

Định nghĩa 2.8. Cho đa tạp trơn M. Với $p, q \in M$, ta định nghĩa **khoảng cách** từ p tới q là

$$d(p, q) = \inf\{L[\gamma] \mid \gamma : [a, b] \to M \text{ trơn từng khúc}, \gamma(a) = p, \gamma(b) = q\},$$

trong đó

$$L[\gamma] = \int_a^b g_{\gamma(t)}(\gamma'(t), \gamma'(t))^{1/2} dt.$$

Ta còn gọi khoảng cách này là **khoảng cách trắc địa** trên (M, g).

Ví dụ 2.9. Tiếp tục ví dụ về mặt cầu S^2, nếu ta trang bị mêtríc Riemann trên S^2 thừa hưởng tích trong $\mathbb{R}^3$ thì khoảng cách trắc địa giữa hai điểm chính là khoảng cách theo đường chim bay.

Một câu hỏi tự nhiên là tại sao ta lại quan tâm khoảng cách trắc địa? Một lí do có thể thấy là vì nó là cách tự nhiên nhất để định nghĩa khoảng cách giữa hai điểm trên đa tạp (thông qua đường đi trên đa tạp).

Định lí 2.10. *Cho (M, g) là đa tạp Riemann và d là khoảng cách trắc địa. Khi đó, (M, d) là không gian mêtríc với tôpô cảm sinh trùng với tôpô của M.*

Ngoài ra, vì mêtríc Riemann g có thể xem như là một "tích trong" đặc biệt của M nên ta có thể bàn về hình học trên M chẳng hạn như thể tích trên M. Tuy nhiên, ở đây, ta sẽ không đi sâu hơn về chi tiết thể tích trên đa tạp.

2.2 Lý thuyết phạm trù

Định nghĩa 2.11. Một **phạm trù** $\mathscr{A}$ bao gồm

- một họ $\mathrm{ob}(\mathscr{A})$ các **vật**;
- với mỗi $A, B \in \mathrm{ob}(\mathscr{A})$, một họ $\mathscr{A}(A, B)$ các **cấu xạ** từ A vào B;
- với mỗi $A, B, C \in \mathrm{ob}(\mathscr{A})$, một ánh xạ

$$\begin{aligned} \mathscr{A}(B, C) \times \mathscr{A}(A, B) &\to \mathscr{A}(A, C) \\ (g, f) &\mapsto g \circ f, \end{aligned}$$

  được gọi là **hợp nối cấu xạ**;
- với mỗi $A \in \mathrm{ob}(\mathscr{A})$, một phần tử 1_A của $\mathscr{A}(A, A)$, được gọi là **cấu xạ đồng nhất** trên A,

thoả mãn các tiên đề sau:

- **tính kết hợp**: với mỗi $f \in \mathscr{A}(A, B), g \in \mathscr{A}(B, C)$ và $h \in \mathscr{A}(C, D)$, $(h \circ g) \circ f = h \circ (g \circ f)$;
- **luật về cấu xạ đồng nhất**: với mỗi $f \in \mathscr{A}(A, B)$, $f \circ 1_A = f = 1_B \circ f$.

Ta thường viết

$$\begin{aligned} A \in \mathscr{A} &\text{ thay cho } A \in \mathrm{ob}(\mathscr{A}); \\ f : A \to B \text{ hoặc } A \xrightarrow{f} B &\text{ thay cho } f \in \mathscr{A}(A, B); \\ gf &\text{ thay cho } g \circ f; \\ \mathrm{Hom}_{\mathscr{A}}(A, B) \text{ hoặc } \mathrm{Hom}(A, B) &\text{ thay cho } \mathscr{A}(A, B). \end{aligned}$$

Ví dụ 2.12. Phạm trù **Set** gồm các vật là các tập hợp với cấu xạ chính là các ánh xạ giữa các tập hợp, hợp nối cấu xạ chính là hợp nối ánh xạ và cấu xạ đồng nhất chính là ánh xạ đồng nhất theo nghĩa thông thường. Ta còn nói **Set** là phạm trù các tập hợp và các ánh xạ, hay ngắn gọn hơn là phạm trù các tập hợp.

Định nghĩa 2.13. Mọi phạm trù $\mathscr{A}$ có một **phạm trù đối** (hoặc **đối ngẫu**) bằng cách đảo ngược chiều của các mũi tên. Một cách hình thức, với mọi $A, B, C \in \mathscr{A}$,

- $\mathrm{ob}(\mathscr{A}^{op}) = \mathrm{ob}(\mathscr{A})$;
- $\mathscr{A}^{\mathrm{op}}(B, A) = \mathscr{A}(A, B)$;
- Cấu xạ đồng nhất của A trong $\mathscr{A}^{\mathrm{op}}$ là 1_A;
- $g \circ_{op} f = f \circ g$ nếu $A \xrightarrow{f} B \xrightarrow{g} C$ trong $\mathscr{A}^{op}$.

Phạm trù các phạm trù có vật là các phạm trù và cấu xạ chính là các hàm tử, là khái niệm sau.

Định nghĩa 2.14. Cho $\mathscr{A}$ và $\mathscr{B}$ là các phạm trù. Một **hàm tử** $F : \mathscr{A} \to \mathscr{B}$ bao gồm:

- một ánh xạ

$$\mathrm{ob}(\mathscr{A}) \to \mathrm{ob}(\mathscr{B}),$$

  viết dưới dạng $A \mapsto F(A)$;
- với mỗi $A, A' \in \mathscr{A}$, một ánh xạ

$$\mathscr{A}(A, A') \to \mathscr{B}(F(A), F(A')),$$

  được viết dưới dạng $f \mapsto F(f)$,

thoả mãn các tiên đề sau:

- $F(f' \circ f) = F(f') \circ F(f)$ nếu $A \xrightarrow{f} A' \xrightarrow{f'} A''$ trong $\mathscr{A}$;
- $F(1_A) = 1_{F(A)}$ với $A \in \mathscr{A}$.

Ta cũng thường gặp những toán tử giống như hàm tử nhưng đảo ngược mũi tên $A \to A'$ trong $\mathscr{A}$ thành $F(A) \leftarrow F(A')$ trong $\mathscr{B}$. Những hàm tử như thế được gọi là phản hàm tử.

Định nghĩa 2.15. Cho $\mathscr{A}$ và $\mathscr{B}$ là các phạm trù. Một **hàm tử phản biến** (hoặc **phản hàm tử**) từ $\mathscr{A}$ vào $\mathscr{B}$ là một hàm tử $\mathscr{A}^{\mathrm{op}} \to \mathscr{B}$.

Giả sử ta đã biết hai phạm trù $\mathscr{A}$ và $\mathscr{B}$. Ta muốn xây dựng $\mathscr{C}$ là phạm trù gồm các vật là các hàm tử giữa $\mathscr{A}$ và $\mathscr{B}$. Một cách để xây dựng các cấu xạ trong $\mathscr{C}$ là thông qua khái niệm **biến đổi tự nhiên**, chính là "cấu xạ giữa hai hàm tử".

Định nghĩa 2.16. Cho $\mathscr{A}$ và $\mathscr{B}$ là các phạm trù và các hàm tử $F, G : \mathscr{A} \to \mathscr{B}$. Một **phép biến đổi tự nhiên** $\alpha : F \to G$ là một họ các cấu xạ $\alpha_A : F(A) \to G(A)$ với $A \in \mathscr{A}$, sao cho với mọi $f : A \to A'$ thì biểu đồ

$$\begin{array}{ccc} F(A) & \xrightarrow{F(f)} & F(A') \\ {\scriptstyle \alpha_A}\downarrow & & \downarrow{\scriptstyle \alpha_{A'}} \\ G(A) & \xrightarrow[G(f)]{} & G(A') \end{array}$$

là giao hoán. Ta còn viết

$$\mathscr{A} \overset{F}{\underset{G}{\Downarrow \alpha}} \mathscr{B} \ .$$

Định nghĩa 2.17. Cho các phạm trù và hàm tử $\mathscr{A} \underset{G}{\overset{F}{\rightleftarrows}} \mathscr{B}$. Ta nói F là **liên hợp trái** của G và G là **liên hợp phải** của F nếu tồn tại các phép biến đổi tự nhiên $\eta : 1_{\mathscr{A}} \to G \circ F$ và $\epsilon : F \circ G \to 1_{\mathscr{B}}$ sao cho các biểu đồ sau

$$\begin{array}{ccc} F & \xrightarrow{F\eta} & FGF \\ & {\scriptstyle 1_F}\searrow & \downarrow{\scriptstyle \epsilon F} \\ & & F \end{array} \qquad \begin{array}{ccc} G & \xrightarrow{\eta G} & GFG \\ & {\scriptstyle 1_G}\searrow & \downarrow{\scriptstyle G\epsilon} \\ & & G \end{array}$$

đều giao hoán. Trong trường hợp trên ta viết $F \dashv G$ và nói F và G lập thành một **adjunction**.

Ở đây, các khái niệm đưa ra rất trừu tượng, nhưng về mặt ý tưởng ta có thể hình dung đây là sự mở rộng của khái niệm "giống nhau". Rõ ràng, nếu F và G thoả mãn $F \circ G = 1_{\mathscr{B}}$ và $G \circ F = 1_{\mathscr{A}}$ thì khái niệm này chính là sự giống nhau "mạnh" giữa hai phạm trù, ở đây, ta nới lỏng điều kiện để vừa đủ để làm việc.

2.3 Fuzzy simplical set

Ở phần này, ta giới thiệu các phạm trù chính mà ta cần để hiểu UMAP. Đầu tiên, ta xây dựng một không gian tổng quát hơn không gian mêtríc để có thể định nghĩa được khoảng cách ∞ và làm nhẹ đi các ràng buộc.

Định nghĩa 2.18. Một không gian **giả mêtríc mở rộng** là một tập X và ánh xạ $d : X \times X \to [0, +\infty]$ thoả mãn với mọi $x, y, z \in X$,

a) $d(x, y) \geqslant 0$ và $d(x, x) = 0$;

b) $d(x,y) = d(y,x)$;

c) $d(x,z) = +\infty$ hoặc $d(x,z) \leqslant d(x,y) + d(y,z)$.

Định nghĩa 2.19. Cho X, Y là hai không gian giả mêtríc mở rộng và ánh xạ $f : X \to Y$. Ta nói f là **non-expansive map** nếu với mọi $x, y \in X$,

$$d_Y(f(x), f(y)) \leqslant d_X(x,y).$$

Phạm trù **EPMet** bao gồm các vật là các không gian giả mêtríc mở rộng và các cấu xạ là các non-expansive maps. Nếu thêm điều kiện hữu hạn cho các vật, ta được phạm trù **FinEPMet**.

Định nghĩa 2.20. Cho $(A, \leqslant_A)$ và $(B, \leqslant_B)$ là các tập được sắp thứ tự. Ta nói ánh xạ $f : A \to B$ là **bảo toàn thứ tự** nếu với mọi $x, y \in A, x \leqslant_A y$ thì $f(x) \leqslant_B f(y)$.

Phạm trù Δ gồm các vật là các tập có thứ tự hữu hạn $[n] = \{0, 1, \ldots, n\}$ với $n \in \mathbb{N}$ và các cấu xạ là các ánh xạ bảo toàn thứ tự. Cho $\mathscr{C}$ là một phạm trù, người ta gọi một **vật đơn hình** trong $\mathscr{C}$ là một phản hàm tử từ Δ vào $\mathscr{C}$. Chẳng hạn, nếu $\mathscr{C} = \mathbf{Set}$ thì ta được những **tập đơn hình** là tổng quát của khái niệm phức đơn hình trong Tôpô đại số, có thể xem thêm tại [11].

Định nghĩa 2.21. Fuzzy set là một cặp (X, μ), trong đó X là tập hợp và ánh xạ $\mu : X \to [0,1]$. Phạm trù **Fuzz** gồm các vật là fuzzy set và cấu xạ giữa (X, μ) và (Y, ν) là ánh xạ $f : X \to Y$ mà

$$\nu(f(x)) \leqslant \mu(x), \ \forall x \in X.$$

Như vậy, **fuzzy simplical set** chẳng qua là một phản hàm tử từ Δ vào **Fuzz**. Như vậy, phạm trù **sFuzz** các fuzzy simplical set với vật là các hàm tử $\Delta^{\mathrm{op}} \to$ **Fuzz** và các cấu xạ chính là các phép biến đổi tự nhiên giữa các hàm tử.

3 Ý tưởng của UMAP

Giả sử dữ liệu đầu vào là $X = \{X_1, \ldots, X_n\} \subset \mathbb{R}^N$. Cũng như các bài toán giảm chiều dữ liệu khác, ta cần tìm một biểu diễn của X dưới số chiều thấp. Không mất tính tổng quát, giả sử X nằm trên đa tạp Riemann (M, g). Nhắc lại rằng, từ 2.10, nếu gọi d là khoảng cách trắc địa trên M cảm sinh từ g thì (M, d) là không gian mêtríc và d sinh ra tôpô của đa tạp. Ý tưởng là ta muốn biểu diễn của X ở số chiều thấp vẫn giữ được một số thông tin từ các lân cận của mỗi điểm, tức là tôpô tương ứng. Như vậy, bước đầu, ta muốn tính xấp xỉ khoảng cách trắc địa từ mỗi điểm tới những điểm khác trong lân cận nào đó của nó.

Xác định lân cận và tính xấp xỉ khoảng cách trắc địa.

Với mỗi điểm X_i, ta cần chỉ ra một cách chọn lân cận N_i của nó. Nếu chọn N_i là quả cầu trong $\mathbb{R}^N$ thì ta phải nghĩ tới việc chọn bán kính như thế nào là tốt mà vẫn có được tôpô từ M. Một cách hiệu quả là chọn N_i bằng cách chọn k điểm gần nó nhất trong $\mathbb{R}^N$ với $k \in \mathbb{N}$ nào đó. Điều này là làm được bằng các thuật toán k-nearest neighbors.

Để tính được xấp xỉ khoảng cách trắc địa, ta sử dụng một kết quả trong bài báo gốc, tuy nhiên, ta chỉ nói ý tưởng cho đơn giản hoá vấn đề. Nếu g "đủ tốt" trên lân cận mở $U \subset M$ chứa p thì tồn tại quả cầu $B \subset U$ trong M có thể tích C_N sao cho $d(p,q) = \frac{1}{r} d_{\mathbb{R}^N}(p,q)$ với $d_{\mathbb{R}^N}$ là mêtríc trên $\mathbb{R}^N$ (không nhất thiết phải là khoảng cách Euclid) và r là bán kính của B trong $\mathbb{R}^N$.

Với mỗi X_i, ta giả sử g đủ tốt trên lân cận mở $U_i \subset M$ chứa X_i nào đó. Ta thấy ta cần giải quyết câu hỏi: Một quả cầu với thể tích cố định C_N thì có thể chứa bao nhiêu điểm dữ liệu? Vì nếu trả lời được câu hỏi trên thì sẽ chọn được k phù hợp để dùng được bổ đề trên. Câu hỏi trên tương đối phức tạp nhưng nếu về mặt trực quan, dưới góc nhìn xác suất, nếu dữ liệu phân bố đề trên M thì

$$\mathbb{P}(X_i \in B) = \frac{\mathrm{vol}(B)}{\mathrm{vol}(M)} = \frac{C_N}{\mathrm{vol}(M)},$$

do đó, số lượng điểm dữ liệu nằm trong quả cầu thể tích C_N là khoảng $\approx \frac{nC_N}{\text{vol}(M)}$. Như vậy, với k đủ bé thì có thể chọn được (ở đây, ta giả sử thể tích của M là hữu hạn). Giả sử, tại mỗi điểm, có quả cầu có thể tích C_N chứa k-nearest neighbors của nó.

Câu hỏi tiếp theo là làm sao để tính được quả cầu B_i ứng với mỗi X_i trong $\mathbb{R}^N$. Một cách tự nhiên là ta sẽ xem như quả cầu này là hình nhỏ nhất chứa X_i và k-nearest neighbors của nó, do đó, một cách tính là

$$d_M(X_i, X_j) \approx \frac{1}{r_i} d_{\mathbb{R}^N}(X_i, X_j),\ X_j \in B_i$$

với $r_i = \max\limits_{X_j \in B_i} d_{\mathbb{R}^N}(X_i, X_j)$. Tuy nhiên, do ở đây ta chỉ cần tính xấp xỉ và để đảm bảo tính ổn định, ta sẽ chọn σ_i là một hệ số phù hợp để

$$d_M(X_i, X_j) \approx \frac{1}{\sigma_i} d_{\mathbb{R}^N}(X_i, X_j),\ X_j \in B_i.$$

Vậy ta đã chỉ ra được cách tính khoảng cách trắc địa từ mỗi điểm tới k-nearest neighbors của nó.

Từ địa phương tới toàn cục

Ta muốn tìm một biểu diễn ở số chiều thấp lưu tất cả thông tin địa phương này. Điều đáng nói ở đây là ta không thể định lượng hoá được khoảng cách giữa mỗi điểm tới điểm khác và mục tiêu của ta là gồm các thông tin địa phương này lại thành một thể thống nhất. Trong [13], người ta chỉ ra được một adjunction như sau

$$\mathbf{sFuzz} \underset{\text{FinSing}}{\overset{\text{FinReal}}{\rightleftarrows}} \mathbf{EPMet}\ .$$

Tương ứng, trong bài báo gốc, người ta cũng chỉ ra được rằng

$$\mathbf{Fin\text{-}sFuzz} \underset{\text{FinSing}}{\overset{\text{FinReal}}{\rightleftarrows}} \mathbf{FinEPMet}\ ,$$

trong đó, ta không chỉ ra cụ thể phạm trù **Fin-sFuzz** được định nghĩa cụ thể như thế nào, ta chỉ cần biết đây là phạm trù con của **sFuzz** là đủ.

Giả sử ta có $(A, d) \in$ **FinEPMet** thì **FinSing**(A) trở thành một fuzzy simplical set, như vậy, **FinReal**$(A)[n]$ là fuzzy set thứ n, ta còn gọi đây là đơn hình n-chiều của **FinSing**(A). Vì chi phí tính toán, ta chỉ quan tâm tới các k-đơn hình với $k = 0$ hoặc $k = 1$ qua hàm tử **FinSing**. Cụ thể, **FinSing**$(A)[0] = (A, \mu)$ và **FinSing**$(A)[1] = (A \times A, \nu)$, trong đó

$$\nu(x, y) = e^{-d(x,y)},\ \forall x, y \in A.$$

Như vậy, các đơn hình 0-chiều và 1-chiều cho ta được một đồ thị có trọng số, ta còn nói ngắn gọn của hai đơn hình trên tạo thành **fuzzy graph**.

Bây giờ, ta cần tìm cách đưa thông tin địa phương về một đồ thị có trọng số sao cho không có điểm nào bị cô lập trong fuzzy graph tương ứng. Một cách xây dựng phù hợp là

$$d_i(X_j, X_k) = \begin{cases} d_M(X_j, X_k) - \rho_i, & \text{nếu } i = j \text{ hoặc } i = k \text{ và } X_j, X_k \text{ là một trong } k\text{-nn của } X_i \\ +\infty, & \text{khác} \end{cases}$$

với ρ_i là khoảng cách trắc địa từ X_i đến lân cận gần nhất mà không tính nó.

Khi đó, **FinSing**(X, d_i) là đồ thị mà có ít nhất một cạnh kề với X_i có trọng số là 1, vì $d_M(X_i, X_j) = \rho_i$ thì $d_i(X_i, X_j) = 0$ nên trọng số trong fuzzy graph tương ứng là 1.

Bằng cách ghép các fuzzy graph ứng với từng X_i lại, ta được fuzzy graph chứa thông tin toàn cục từ đa tạp. Cụ thể hơn về quá trình "ghép" các fuzzy graph có thể hiểu như sau. Giả sử ta có hai fuzzy set (A, μ) và (A, ν) (trong bài toán của ta, ta hiểu $A = X \times X$), ta định nghĩa

$$(A, \mu) \cup (A, \nu) = (A, \mu \cup \nu),$$

trong đó

$$(\mu \cup \nu)(a) = \mu(a) \perp \nu(a) = \mu(a) + \nu(a) - \mu(a)\nu(a), \ \forall a \in A.$$

Biểu diễn ở số chiều thấp

Ta cần tìm một biểu diễn $Y = \{Y_1, \ldots, Y_n\} \subset \mathbb{R}^d$ với d là số chiều thấp mà ta muốn, ta nói Y_i là đại diện của X_i. Ý tưởng ở đây là ta sẽ cần so sánh giữa fuzzy graph ứng với Y và cái ứng với X. Một cách tự nhiên, ta khởi tạo một biểu diễn cho X là Y, ta cũng tính được fuzzy graph ứng với Y, như vậy, ta được hai đồ thị có trọng số. Để đánh giá xem Y có tốt không, người ta dùng fuzzy set cross entropy, cụ thể nếu (A, μ) và (A, ν) là các fuzzy set, ta định nghĩa

$$C((A, \mu), (A, \nu)) = \sum_{a \in A} \left(\mu(a) \log \frac{\mu(a)}{\nu(a)} + (1 - \mu(a)) \log \frac{1 - \mu(a)}{1 - \nu(a)} \right).$$

Một lưu ý là ta áp dụng fuzzy set cross entropy cho $A = [n] \times [n]$ với μ là trọng số trên fuzzy graph ở số chiều cao, $\mu(i, j)$ chỉ trọng số ứng với cạnh nối giữa X_i với X_j trong fuzzy graph tương ứng với X, tương tự với ν là trọng số trên fuzzy graph ở số chiều thấp ứng với Y. Bằng biến đổi đại số, ta viết lại được

$$C((A, \mu), (A, \nu)) = K - \sum_{a \in A} (\mu(a) \log \nu(a) + (1 - \mu(a)) \log(1 - \nu(a)))$$

với

$$K = \sum_{a \in A} (\mu(a) \log(\mu(a)) + (1 - \mu(a)) \log(1 - \mu(a)))$$

là hằng số. Như vậy, để cực tiểu hoá cross entropy, ta cần tìm ν để tối tiểu hoá biểu thức

$$- \sum_{a \in A} (\mu(a) \log \nu(a) + (1 - \mu(a)) \log(1 - \nu(a))).$$

Cụ thể hơn, ta sẽ lấy mẫu một cạnh có trọng số $\mu(a)$ và cập nhật tương ứng với giá trị $\nu(a)$. Sau đó, ta sử dụng phương pháp lấy mẫu âm bằng cách chọn một số mẫu có khả năng là mẫu âm (ví dụ có trọng số bằng 0) và cập nhật tương ứng với giá trị $1 - \nu(a)$. Tức là, ta cần cực đại hóa tương ứng với $\log \nu(a)$ và $\log(1 - \nu(a))$ lần lượt với a là mẫu dương hoặc mẫu âm. Việc ấy được thực hiện dựa trên thuật toán Stochastic Gradient Descent. Do đó, để giải quyết bài toán tối ưu này, ta cần tìm hàm khả vi $\nu(a)$.

4 Thuật toán UMAP

4.1 Các bước thực hiện thuật toán UMAP

Chúng ta biết rằng, những bài toán giảm chiều dựa trên đồ thị thường có hai bước chính cần được thực hiện, và UMAP cũng không phải là một ngoại lệ. Các bước đó là:

- Xây dựng đồ thị: Đầu tiên, ta cần tìm k điểm lân cận gần nhất cho mỗi điểm dữ liệu trong không gian ban đầu và khoảng cách giữa chúng. Sau đó, sử dụng các kết quả này để xây dựng một đồ thị (hay fuzzy graph được trình bày ở phần trên) được tổng hợp từ các thông tin địa phương. Bước này giúp tạo ra một đại diện cho các điểm dữ liệu thể hiện được cấu trúc tôpô của chúng.
- Graph Layout: Sau khi đã xây dựng fuzzy graph, ta chiếu dữ liệu xuống số chiều thấp hơn. Để thực hiện bước này, UMAP xây dựng fuzzy graph ứng với số chiều thấp và cố gắng tối ưu hóa để nó gần nhất có thể với cấu trúc tôpô của đồ thị ban đầu.

Mã giả sau đây sẽ mô tả chi tiết hơn thuật toán dựa trên hai bước trên.

Algorithm 1 UMAP(X, k, d, min-dist, n-epochs)

Input Tập dữ liệu X, k là số điểm lân cận ứng với một điểm dữ liệu, số chiều d sau khi giảm, min-dist để kiểm soát độ phân tán của dữ liệu ở số chiều thấp, n-epochs là số lần duyệt qua toàn bộ tập dữ liệu.

Output Y là đại diện cho cấu trúc liên kết của dữ liệu sau khi giảm chiều.

```
procedure UMAP
    # Xây dựng đại diện cấu trúc tôpô của dữ liệu ở số chiều cao.
    for all x ∈ X do
        fs-set[x] ← LocalFuzzySimplicialSet(X, x, k) # Lấy thông tin cục bộ
    top-rep ← ∪_{x∈X} fs-set[x] # Hợp các kết quả để có được đại diện ở số chiều cao
    # Tối ưu hóa và trả kết quả là đại diện cấu trúc tôpô của dữ liệu ở số chiều thấp.
    Y ← SpectralEmbedding(top-rep, d) # Khởi tạo
    Y ← OptimizeEmbedding(top-rep, Y , min-dist, n-epochs) # Tối ưu hóa
    return Y
```

Cụ thể, hai bước chính của thuật toán UMAP lần lượt được thực hiện như sau

1. Nhắc lại lý thuyết ở phần 3, ứng với mỗi điểm dữ liệu $X_i \in X$, ta sẽ xây dựng được giả mêtríc thông qua khoảng cách trắc địa

$$d_i(X_j, X_k) = \begin{cases} d_M(X_j, X_k) - \rho, & \text{nếu } i = j \text{ hoặc } i = k \\ \infty & \text{khác} \end{cases} \tag{4.1}$$

   Khi đó không gian giả mêtríc (X, d_i) được đưa qua hàm tử $FinSing$ và tác động vào [0], [1] để tạo thành đơn hình 0-chiều và 1-chiều chứa cấu trúc địa phương ứng với X_i. Sau đó, bằng phép hợp các đơn hình 0-chiều và 1-chiều lại với nhau, ta sẽ xây dựng được fuzzy graph đại diện cho cấu trúc ở số chiều cao. Các bước cụ thể được đề cập ở phần 4.2.

2. Phần 3 cũng chỉ ra rằng UMAP sẽ khởi tạo đại diện cho cấu trúc tôpô của dữ liệu ở số chiều thấp và tối ưu hóa nó để đạt được kết quả tốt nhất. Việc khởi tạo Y có thể được thực hiện bằng việc lấy ngẫu nhiên hoặc một số phương pháp khác như SVD, PCA. Tuy nhiên, trong bài trình bày này lựa chọn Speactral Embedding [2] [3] vì một số lí do được trình bày trong bài báo gốc [1]. Hơn nữa, kết quả mà thuật toán thể hiện trên thực nghiệm cho kết quả hội tụ nhanh và ổn định hơn. Cuối cùng, ta sẽ tối ưu hóa để có được kết quả tốt nhất có thể. Chi tiết được trình bày trong phần 4.4.

4.2 Xây dựng local fuzzy simplicial set

Trong phần này ta sẽ tìm đơn hình 0-chiều và 1-chiều ứng với một điểm dữ liệu $x \in X$ để nắm bắt được thông tin địa phương xung quanh x. Ý tưởng này được thực hiện bằng cách xấp xỉ khoảng cách trắc địa k điểm lân cận của x và biến đổi thành fuzzy graph.

Thuật toán được thực hiện theo các bước sau

Algorithm 2 LocalFuzzySimplicialSet(X, x, k)

Input Tập dữ liệu X, $x \in X$, k là số lân cận ứng với một điểm dữ liệu.
Output fs-set là 0-simplicies và 1-simplices ứng với x.

procedure LocalFuzzySimplicialSet
    # Xây dựng không gian giả metric ứng với điểm dữ liệu x.
    knn, knn-dists ← ApproxNearestNeighbors(X, x, k) # Thu được kết quả knn là k điểm lân cận của x và knn-dists tương ứng là khoảng cách của x đến k điểm đó.
    ρ ← knn-dists[1] # Khoảng cách đến lân cận gần nhất.
    σ ← SmoothKNNDist(knn-dists, k, ρ) # Tham số chuẩn hóa.
    fs-set$_0$ ← X. # Ứng với tập dữ liệu ban đầu
    fs-set$_1$ ← $\{([x, y], 0) | y \in X\}$ # Lấy tất cả cạnh có một đỉnh là x và trọng số khởi tạo là 0.
    for all $y \in$ knn **do**
        $d_{x,y} \leftarrow \max(0, \text{dist}(x, y) - \rho)/\sigma$ # Tính khoảng cách trong không gian giả mêtríc mở rộng đã được chuẩn hóa
        fs-set$_1$ ← fs-set$_1 \cup ([x, y], \exp(-d_{x,y}))$ # Thay thế trọng số của các cạnh xây dựng từ x và một trong k điểm lân cận của nó.
    return fs-set

1. Đầu tiên, ứng với điểm dữ liệu x, ta sẽ tìm ra k điểm lân cận của nó thông qua thuật toán Nearest Neighbors Descent [4]. Khi đó, ta thu được knn là k điểm lân cận của x và knn-dists là khoảng cách từ x tới các lân cận tương ứng. Lưu ý rằng, knn-dists được sắp theo thứ tự tăng dần và khoảng cách được tính theo mêtríc $d_{\mathbb{R}^N}$.

2. Tham số ρ được tính bởi $\rho = \min_{y \in X} d(x, y) = \min_{y \in \text{knn}} d(x, y)$ là khoảng cách từ x đến điểm gần nhất trong R^N. Vì tính locally connected nên ρ sẽ đảm bảo x kết nối với ít nhất một điểm khác thuộc X với trọng số trong fuzzy graph bằng 1. Hơn nữa, curse of dimensionality sẽ làm cho khoảng cách từ x đến các điểm lân cận là rất xa nhưng sự chênh lệch giữa các khoảng cách ấy là nhỏ hơn rất nhiều, vì vậy ρ cũng giúp cải thiện vấn đề này và thuận lợi hơn trong chuyện tính toán.

3. Tham số σ được trình bày trong phần 4.3.

4. Khởi tạo fuzzy graph với fs-set$_0$ ứng với 0-simplices hay các dữ liệu ban đầu. Bên cạnh đó, fs-set$_1$ thể hiện cho 1-simplices, hay hiểu theo cách trực quan là một cạnh, được khởi tạo bằng cách nối x với tất cả các điểm thuộc X với trọng số bằng 0.

5. Xét k điểm gần nhất của x vừa tìm được ở bước 1. Ta chỉ quan tâm đến k điểm vì ở bước này chỉ cần trích xuất thông tin cục bộ ứng với x. Với mỗi điểm y là một trong k lân cận của x

    - Tính khoảng cách trong không gian giả mêtríc mở rộng đã được chuẩn hóa

    $$d(x, y) = \frac{\max\{0, dist(x, y) - \rho\}}{\sigma}$$

    trong đó $dist(x, y)$ chính là khoảng cách trong $\mathbb{R}^N$ được tính ở bước 1. Việc lấy max để đảm bảo rằng trọng số của các điểm mà khoảng cách từ nó đến x nhỏ hơn hoặc bằng ρ bằng 1. Ngoài ra, sự tồn tại σ ở mẫu sẽ được trình bày ở phần 4.3.

    - Tính trọng số của cạnh xây dựng từ $[x, y]$ theo công thức $w((x, y)) = \exp\{-d(x, y)\}$.

    - Cập nhật lại trọng số của cạnh xây dựng từ $[x, y]$ trong tập fs-set$_1$ bằng cách thay thế trọng số trước đó bằng trọng số mới $w(x, y)$.

6. Trả ra kết quả (0-simplices, 1-simplices) chứa thông tin cục bộ ứng với x.

4.3 Tính hệ số chuẩn hóa cho khoảng cách σ

Theo phần 4.2, σ được sử dụng trong công thức

$$d(x,y) = \frac{\max\{0, dist(x,y) - \rho\}}{\sigma}$$

Thêm vào đó, từ phần trước ta biết rằng giả mêtríc trên không gian giả mêtríc mở rộng được định nghĩa dựa trên khoảng cách trắc địa trên đa tạp

$$d_i(X_j, X_k) = \begin{cases} d_M(X_j, X_k) - \rho_M, & \text{nếu } i = j \text{ hoặc } i = k \\ \infty & \text{khác} \end{cases}$$

Mà trong phần trước chỉ ra rằng ta có thể định nghĩa khoảng cách trắc địa

$$d_M(x,y) = \frac{d_{\mathbb{R}^N}}{t}$$

Vì vậy ta định nghĩa lại khoảng cách trắc địa trên đa tạp

$$d_i(X_j, X_k) = \begin{cases} \frac{d_{\mathbb{R}^N}(X_j, X_k) - \rho}{t}, & \text{nếu } i = j \text{ hoặc } i = k \\ \infty & \text{khác} \end{cases}$$

Trong đó, ρ là khoảng cách từ x đến lân cận gần nhất theo $d_{\mathbb{R}^N}$. Hơn nữa, trong bài [1] có chỉ ra rằng, từ thực nghiệm đã cho thấy thuật toán sẽ làm việc nhanh và ổn định hơn thì ta chuẩn hóa các khoảng cách này sao cho tổng của tất cả các trọng số của phần trên bằng $\log_2 k$. Tức là ta cần tìm hằng số α sao cho

$$\sum_{i=1}^{k} \exp(\frac{-(\text{knn-dists}_i - \rho)}{\alpha t}) = \log_2(k)$$

Để đơn giản, ta sẽ tìm σ sao cho

$$\sum_{i=1}^{k} \exp(\frac{-(\text{knn-dists}_i - \rho)}{\sigma}) = \log_2(k)$$

Algorithm 3 SmoothKNNDist(knn-dists, k, ρ)

Input knn-dists thể hiện cho các khoảng cách từ điểm đang xét đến các điểm hàng xóm, l là số lượng điểm lân cận của điểm đang xét, ρ là khoảng cách từ điểm đang xét đến điểm gần nhất.

Output σ là tham số để chuẩn hóa khoảng cách.

1: **procedure** SmoothKNNDist
2: Tìm σ sao cho $\sum_{i=1}^{k} \exp(-(\text{knn-dists}_i - \rho)/\sigma) = \log_2(k)$.
3: return σ.

Bằng cách sử dụng thuật toán tìm kiếm nhị phân ta có thể xấp xỉ được σ thỏa các điều kiện trên. Cụ thể, khởi tạo khoảng $[l, r]$ mà $\sigma \in [l, r]$ và áp dụng tìm kiếm nhị phân trên $[l, r]$. Với $mid = \frac{l+r}{2}$ và $f(x) = \sum_{i=1}^{k} \exp(\frac{-(\text{knn-dists}_i - \rho)}{x})$ thì thuật toán sẽ dừng lại khi $|f(mid) - \log_2 k| < \epsilon$ với ϵ đủ nhỏ.

4.4 Tối ưu hóa để tìm ra đại diện ở số chiều thấp

Cuối cùng, sau khi đã có khởi tạo cho các điểm dữ liệu ở số chiều thấp, chúng ta sẽ sử dụng thuật toán Stochastic Gradient Descent để tìm ra đại diện đủ tốt cho cấu trúc liên kết của dữ liệu. Để đơn giản, chúng ta sẽ chọn đa tạp chính là $\mathbb{R}^d$ với d là số chiều sau sử dụng UMAP. Khi đó, mêtríc được sử dụng để đo khoảng cách giữa các điểm dữ liệu chính là khoảng cách Euclide. Mã giả sau sẽ thể hiện rõ hơn từng bước hoạt động

Cụ thể hơn,

Algorithm 4 OptimizeEmbedding(top-rep, Y , min-dist, n-epochs))

Input top-rep là đại diện cho cấu trúc liên kết ở số chiều cao, Y là khởi tạo cho đại diện ở số chiều thấp sử dụng Spectral embedding, min-dist là tham số để kiểm soát layout hay để khởi tạo hàm Ψ, n-epochs là số lượng lần duyệt qua tập dữ liệu.

Output Y là đại diện ở số chiều thấp.

procedure OptimizeEmbedding
  $\alpha \leftarrow 1.0$
  Từ min-dist ta xây dựng hàm Ψ, sau đó cố gắng cho hàm Φ gần với Ψ nhất có thể để tìm ra hai tham số của hàm Φ
  for $e \leftarrow 1, \ldots,$ n-epochs **do**
    for all $([a, b], p) \in$ top-rep$_1$ **do**
      if RANDOM()$\leqslant p$ **then**
        $y_a \leftarrow y_a + \alpha\nabla(\log(\Phi))(y_a, y_b)$
        for $i \leftarrow 1, \ldots,$ n-neg-samples **do**
          $c \leftarrow$ random sample from Y
          $y_a \leftarrow y_a + \alpha\nabla(\log(1 - \Phi))(y_a, y_c)$
    $\alpha \leftarrow 1.0 - e/$n-epochs
  return Y

1. Chọn learning rate bằng 1.0 là một hyperparameter của thuật toán SGD. Mặc dù chúng ta có thể khởi tạo bằng một giá trị khác, tuy nhiên, trong bài tiểu luận này, chúng ta sử dụng giá trị learning rate giống như trong bài báo gốc [1]. Chúng ta đã đánh giá và cho rằng giá trị này không quá lớn và không quá nhỏ để đảm bảo rằng sau mỗi n-epochs lần cập nhật, giá trị không quá gần 0.

2. Với tham số min_dist được đưa vào từ input ta xây dựng hàm

$$\Psi(x) = \begin{cases} 1 & \text{nếu } \|x - y\|_2 \leqslant min\_dist \\ \exp -(\|x - y\|_2 - min\_dist) & \text{otherwise} \end{cases}$$

   Như chúng ta đã biết, ở số chiều cao, để định nghĩa khoảng cách giữa hai điểm dữ liệu x và y, ta cần biết khoảng cách từ x đến điểm dữ liệu gần nhất ρ để đảm bảo rằng các cạnh được xây dựng từ x với các điểm mà khoảng cách nó nhỏ hơn ρ sẽ có trọng số bằng 1. Tương tự như vậy, ở số chiều thấp, chúng ta sẽ áp dụng ý tưởng đó bằng cách tạo ra tham số min_dist. Khi đó, ứng với một điểm dữ liệu x, bất kỳ điểm y mà $\|x - y\| \leqslant$ min_dist thì trọng số của cạnh xây dựng từ 2 điểm ấy bằng 1. Ngược lại, trọng số sẽ được tính thông qua công thức $\exp -(\|x - y\|_2 - min\_dist)$

3. Sau khi đã xây dựng được 0-simplices và 1-simplices lần lượt là Y và Ψ ở số chiều thấp, ta sẽ tiến hành tối ưu hóa để tìm ra kết quả tốt nhất có thể. Tuy nhiên lưu ý rằng, do sử dụng thuật toán SGD nên ta cần tìm hàm mục tiêu khả vi. Trong thực tế, UMAP sử dụng "the family of curves" có dạng $\Phi(x, y) = (1 + a(\|x - y\|_2^2)^b)^{-1}$ để thực hiện việc ấy, trong đó hai a, b là hằng số thu được từ việc sử dụng non-linear least squares với mong muốn Φ có thể gần nhất với Ψ. Cụ thể trong [7], tận dụng hàm curve_fit của thư viện scipy [6], thuật toán đã tìm được tham số a, b sao cho mean square error giữa Φ và Ψ là nhỏ nhất.

4. Lặp lại các bước $5 - 6$ n-epochs lần - tương ứng với số lần duyệt qua toàn bộ bộ dữ liệu.

5. Lần lượt duyệt qua tất cả các cạnh của fuzzy graph ở số chiều cao, trong đó mỗi cạnh của top_rep gồm 3 thành phần $([a, b], p)$ với a, b là hai đỉnh của cạnh và p là trọng số của cạnh đó. Nhắc lại, mục tiêu của chúng ta là bảo toàn cấu trúc liên kết của dữ liệu như giả định thứ ba có trình bày ở mục 2. Tức là, khi xét $([a, b], p)$ ta cũng mong muốn ứng với số chiều thấp cũng sẽ tồn tại $[(y_a, y_b), p]$. Vì vậy trong thực tế, chúng ta lấy tập đỉnh của không gian

thấp chiều bằng đúng tập đỉnh của không gian nhiều chiều để đảm bảo được thứ tự của chúng, sau đó ứng với $[a, b]$ ta sẽ có được $[y_a, y_b]$ tương ứng và thực hiện tối ưu hóa.

- Lấy cạnh $([y_a, y_b], q])$ tương ứng với $([a, b], p), q \leqslant p$.
- Chúng ta biết rằng, $([a, b], p)$ thể hiện b là một trong k điểm lân cận của a, tức là ứng với công thức ở phần 2.3 nhiệm vụ cần làm là cực đại $p \log \Phi([y_a, y_b])$ - kéo hai điểm này lại gần nhau, mà p là cố định nên ta cần cực đại $\log(\Phi([y_a, y_b]))$. Hơn nữa, mỗi lần ta chỉ lấy một cạnh, vì vậy bằng cách sử dụng stochastic gradient descent ta có thể để cập nhật y_a bằng công thức

  $$y_a \leftarrow y_a + \alpha \Delta(\log(\Phi))(y_a, y_b)$$

  để tìm được điểm y_a sao cho $\log(\Phi([y_a, y_b]))$ lớn nhất có thể. Cụ thể hơn từ [7].
- Chọn ra $n\_neg\_samples$ mẫu $([a, c], w)$ mà c không lân cận với a ($w \approx 0$), khi đó ta có tương ứng $([y_a, y_c], h)$. Theo phần 2.3 ta cần cực đại $w(1 - \log(\Phi([y_a, y_c])))$ với ý nghĩa đẩy ra những điểm không lân cận với nhau, mà w là cố định nên ta cần cực đại $1 - \log(\Phi([y_a, y_c]))$. Tương tự trên, ta cũng cập nhật y_a bằng công thức

  $$y_a \leftarrow y_a + \alpha \Delta(\log(1 - \Phi))(y_a, y_c)$$

6. Cập nhật learning rate thông qua công thức $\alpha \leftarrow 1.0 - e/\text{n-epoch}$. Tức là α sẽ giảm dần theo thời gian để hội tụ dễ hơn.
7. Trả ra kết quả là Y sau khi đã tối ưu hóa.

5 Ứng dụng

UMAP trong trực quan hóa dữ liệu

Mục tiêu

Sử dụng UMAP để đánh giá phương pháp nhúng nào là tốt nhất.

Mô tả dữ liệu

Bài tiểu luận sử dụng là bộ fetch_20newsgroups tải từ thư viện sklearn. Dữ liệu dưới dạng văn bản bao gồm khoảng 18000 bản tin của 20 chủ đề tương ứng là nhãn của chúng.

Ý tưởng chính

Như chúng ta đã biết, một trong những ứng dụng phổ biến của UMAP là trực quan hóa dữ liệu. Thông thường, chúng ta thường dùng thuật toán này để ép dữ liệu về một hoặc hai chiều để tiện cho quan sát và phân tích. Dựa trên ý tưởng này, bài tiểu luận sẽ chia dữ liệu thành hai tập để huấn luyện và kiểm tra, sau đó sử dụng UMAP để giảm chiều dữ liệu sau khi sử dụng nhiều phương pháp embedding khác nhau như Bag of word, Tf-idf, Pre-train model, RNN-LSTM đối với tập huấn luyện. Tiếp theo, sử dụng model đã huấn luyện để chuyển đổi dữ liệu trong tập test, kết quả sẽ được trực quan hóa cùng với nhãn của dữ liệu. Khi đó, chúng ta sẽ quan sát kết quả và đưa ra nhận xét về độ chính xác của từng phương pháp được sử dụng ở trên. Ngoài ra, việc đánh giá còn được sử dụng thêm Silhouette score và Trustworthiness score.

Chi tiết

1. **Tiền xử lý dữ liệu**

   Với bài trình bày này gồm ba bước

- Xóa đi các kí tự đặc biệt, không phải từ bằng hàm re.
- Tách đoạn văn bản thành các từ bằng hàm word_tokenize của nltk.
- Xóa những từ quá ngắn và stopword là những từ xuất hiện quá nhiều và hầu như không đóng góp ý nghĩa trong câu, đồng thời sử dụng ps.stem để đưa từ về dạng gốc của nó.

2. **Bag of word**

   Bag of Words (BoW) là một kỹ thuật xử lý ngôn ngữ tự nhiên phổ biến trong Machine Learning và NLP (Natural Language Processing). Kỹ thuật BoW chuyển đổi một văn bản thành một tập hợp các từ (word) và đếm số lần xuất hiện của mỗi từ trong văn bản đó. Kỹ thuật này không quan tâm đến thứ tự các từ trong văn bản và chỉ tập trung vào tần suất xuất hiện của từ.

   Khi sử dụng kỹ thuật BoW, ứng với một văn bản ta tạo ra một véctơ đặc trưng cho mỗi văn bản với số chiều bằng số lượng từ trong từ điển, trong đó mỗi phần tử của vector tương ứng với một từ trong từ điển. Giá trị của từng phần tử trong vector là số lần xuất hiện của từ tương ứng trong văn bản. Cụ thể, chúng ta sẽ sử dụng hàm CountVectorizer của sklearn với số đặc trưng giới hạn tối đa là 1000.

3. **Tf-idf**

   Tf-idf là một kỹ thuật xử lý ngôn ngữ tự nhiên phổ biến để đánh giá tầm quan trọng của một từ trong một văn bản dựa trên tần suất xuất hiện của từ đó trong văn bản cũng như sự phân bố của từ đó trong tập dữ liệu. Trong đó, Term frequency (tf) là số lần xuất hiện của một từ trong văn bản và inverse document frequency (idf) là một chỉ số đo lường tầm quan trọng của từ đó trong toàn bộ tập dữ liệu. Công thức tính tf-idf như sau:

   $$tf - idf = tf(t, d) * idf(t)$$

   với

   - $tf(t, d)$ là tần suất xuất hiện của từ t (từ cần đánh giá) trong văn bản d.
   - $idf(t)$ là chỉ số đo lường tầm quan trọng của từ t trong toàn bộ tập dữ liệu, được tính bằng công thức:

     $$idf(t) = \log(N/df(t))$$

     Trong đó:

     - N là số lượng văn bản trong tập dữ liệu.
     - $df(t)$ là số lượng văn bản trong tập dữ liệu mà từ t xuất hiện trong đó.

   Ứng với một văn bản Tf-idf sẽ trả ra một véctơ đặc trưng mà mỗi cột tương ứng với một từ trong từ điển được sử dụng trong tất cả các văn bản. Giá trị tại mỗi phần tử là giá trị tf-idf của từ tương ứng và văn bản tương ứng. Cụ thể, chúng ta sẽ sử dụng hàm TfidfVectorizer của sklearn với số đặc trưng giới hạn tối đa là 1000.

4. **Pre-train model - Glove**

   Pre-trained model GloVe (Global Vectors for Word Representation) là một mô hình nhúng từ (embedding model) được huấn luyện trên một lượng lớn văn bản để biểu diễn các từ dưới dạng các vector đặc trưng. Mô hình GloVe đã được huấn luyện trên một lượng lớn dữ liệu văn bản, bao gồm cả các tiểu thuyết, báo chí, bài báo khoa học và các trang web trên Internet. Nhờ vậy, mô hình GloVe có thể biểu diễn các từ trong ngôn ngữ một cách rất hiệu quả. Việc sử dụng pre-trained model GloVe embedding trong các tác vụ xử lý ngôn ngữ tự nhiên giúp

cho việc biểu diễn từ và câu trở nên đơn giản hơn, giảm thiểu sự phụ thuộc vào việc huấn luyện từ đầu trên các tập dữ liệu lớn và phức tạp.

Pre-trained model sẽ cho ra ứng với một từ là một véctơ, mỗi cột tương ứng với một chiều trong không gian véctơ đặc trưng của các từ đó. Giá trị tại mỗi phần tử của ma trận là giá trị véctơ đặc trưng của từ tương ứng trong không gian véctơ đó. Trong bài tiểu luận này, mỗi văn bản sẽ ứng với một ma trận là trung bình các ma trận nhúng của các từ. Hơn nữa, số cột là 100 vì chúng ta sẽ sử dụng bộ glove 100d. Ta sẽ lần lượt thực hiện các bước

- Tải bộ glove, đọc file glove.6B.100d.txt.
- Ứng với mỗi từ trong tập dữ liệu, ta tìm chỉ số tương ứng trong glove.
- Với một văn bản, lấy index đã tìm được ở bước 2 để tìm ma trận nhúng của từng từ và tính trung bình để ra được ma trận đại diện cho văn bản đó.

5. **LSTM**

   LSTM (Long Short-Term Memory) là một kiến trúc mạng neuron nhân tạo trong học sâu (deep learning), cấu trúc của nó khá phức tạp, ta có thể tham khảo thêm tại [10]. LSTM trong việc nhúng được thực hiện bằng cách chia văn bản thành các từ và mỗi từ được biểu diễn dưới dạng một vector đặc trưng sử dụng các kỹ thuật nhúng từ (word embedding) như Word2Vec, GloVe, FastText,... Sau đó, các vector đặc trưng này được đưa vào một mạng LSTM để tạo ra một vector đặc trưng cho cả câu hoặc đoạn văn bản. Trong quá trình huấn luyện, các tham số của mạng LSTM được điều chỉnh để sao cho vector đặc trưng được tạo ra bởi LSTM có thể tốt nhất biểu diễn ý nghĩa của câu hoặc đoạn văn bản. Lưu ý rằng, ta sẽ tạo ra một bài toán cụ thể cho mạng LSTM này, chẳng hạn dự đoán hoặc phân loại, để giúp cho việc nhúng được chính xác và hiệu quả hơn.

   Tương tự như sử dụng pre-train model, kết quả cuối cùng sẽ là một ma trận mỗi cột tương ứng với một chiều trong không gian vector đặc trưng của các từ đó. Số chiều tùy thuộc vào chúng ta lựa chọn và điều chỉnh, cụ thể ở bài này là 200. Để đạt được kết quả trên, ta lần lượt thực hiện

   - Chia tập huyến luyện thành hai tập train - để huấn luyện và valid - để đánh giá.
   - Nhiệm vụ để mô hình học trong trường hợp này là phân loại các văn bản trong tập dữ liệu về các nhóm, trong đó có 20 nhóm tương ứng với 20 nơi viết ra các bản tin tức này.
   - Mô hình được định nghĩa bao gồm
     - Lớp đầu tiên để tạo vector nhúng ban đầu cho các từ, có thể sử dụng random nhưng trong bài này, ta sẽ sử dụng luôn kết quả của phương pháp sử dụng glove.
     - Lớp bi-LSTM để có thể học được ngữ nghĩa của câu từ hai chiều.
     - Lấy trung bình để tạo ra được một véctơ đại diện duy nhất cho một văn bản.
     - Lớp fully connected.
     - Lớp softmax để phân loại.
   - Thực hiện các bước huấn luyện.
   - Sử dụng mô hình vừa huấn luyện ở trên để lấy các véctơ đại diện cho các đoạn văn bản của cả tập huấn luyện và tập thử nghiệm bằng cách chạy lại mô hình nhưng không cho mô hình học, sau đó lấy kết quả ẩn của lớp LSTM.

6. **Áp dụng thuật toán UMAP** Ta sẽ sử dụng UMAP để giảm chiều kết quả sau khi áp dụng các phương pháp trên với cái siêu tham số được lựa chọn là

- n_neighbors=30.
- min_dist=0.1.
- n_components=2.
- metric='cosine'.
- random_state=42

Các siêu tham số này được lựa chọn dựa trên thử nghiệm.

7. **Các phương pháp để đánh giá**

   - Hình ảnh được hiển thị thông qua thư viện mathplotlib. Bằng trực quan, ta có thể đánh giá được độ chính xác của phương pháp dựa trên việc đánh giá xem các đại diện của cùng một nhóm có được phân bố gần nhau không.
   - Silhouette score là một phương pháp được sử dụng để đánh giá chất lượng của các cụm. Silhouette score được tính toán bằng cách đo độ tương đồng giữa một điểm dữ liệu với các điểm trong cùng một cụm và với các điểm trong các cụm khác.

     Cụ thể, đối với mỗi điểm dữ liệu, ta tính toán hai giá trị:

     – $a(i)$: Trung bình khoảng cách giữa điểm dữ liệu i và các điểm dữ liệu khác trong cùng cụm với i.

     – b(i): Trung bình khoảng cách giữa điểm dữ liệu i và các điểm dữ liệu trong cụm khác gần nhất với i.

     Sau đó, Silhouette score của điểm dữ liệu i được tính bằng công thức:

     $$s(i) = \frac{b(i) - a(i)}{max(a(i), b(i))}$$

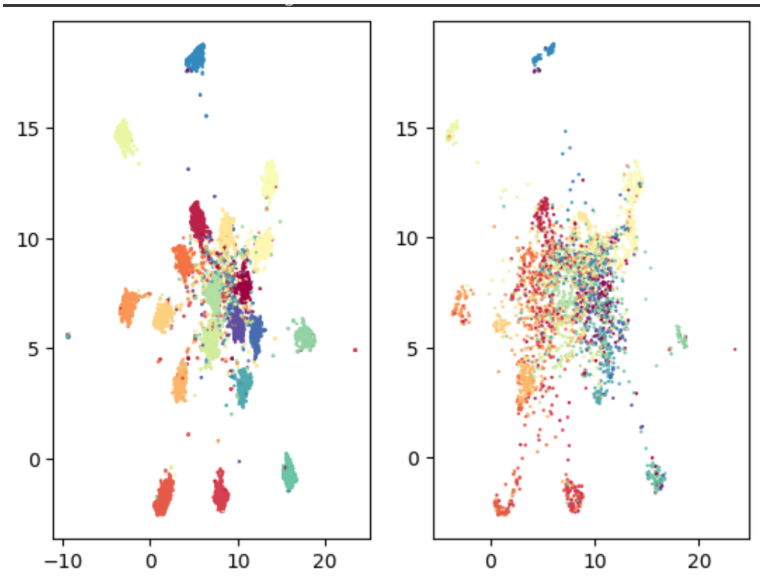
     Trong đó, giá trị $s(i)$ nằm trong khoảng $[-1, 1]$, với giá trị càng lớn thì cụm của điểm dữ liệu i càng tốt. Silhouette score của toàn bộ cụm được tính bằng trung bình của tất cả các điểm dữ liệu trong cụm đó.
   - Trustworthiness score được sử dụng để đánh giá chất lượng của biểu diễn chiều thấp dữ liệu bằng cách sử dụng k điểm gần nhất của $x \in X$, một ánh xạ được coi là đáng tin cậy nếu k điểm này ở không gian có số chiều cao cũng gần điểm x trong không gian có số chiều thấp. Trustworthiness score trong UMAP là một số trong khoảng từ 0 đến 1, trong đó điểm số càng gần 1 thì cấu trúc cục bộ của dữ liệu được bảo tồn tốt trong không gian có số chiều thấp.

Kết quả

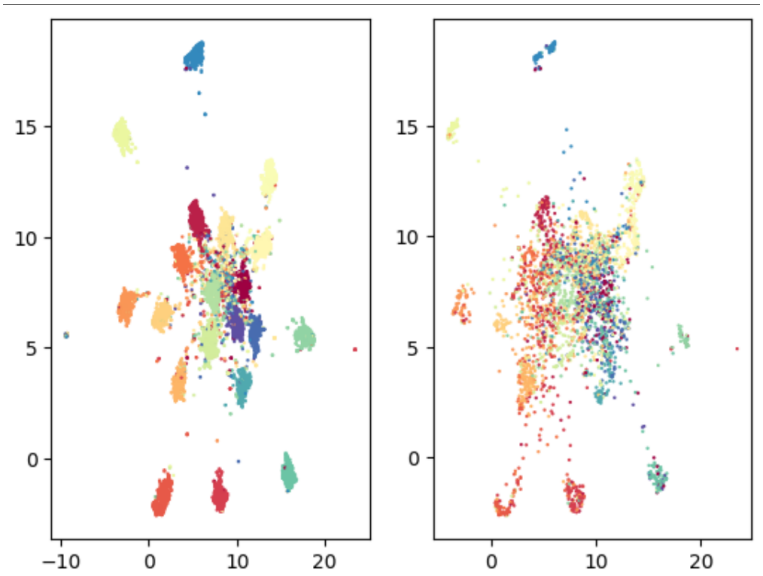
Trực quan hóa

Kết quả sau khi sử dụng UMAP của tập huấn luyện và thử nghiệm được lần lượt hiển thị ứng với các phương pháp

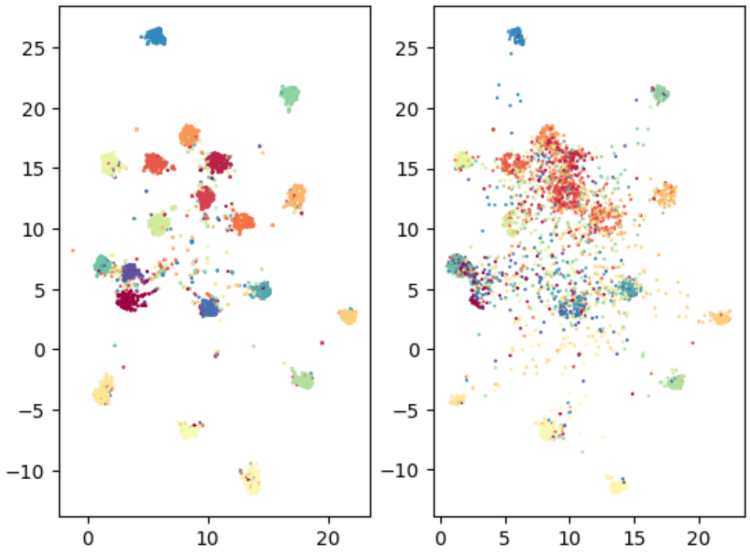
- Bag of word.
- Tf-idf.
- Pre-train model.
- LSTM.



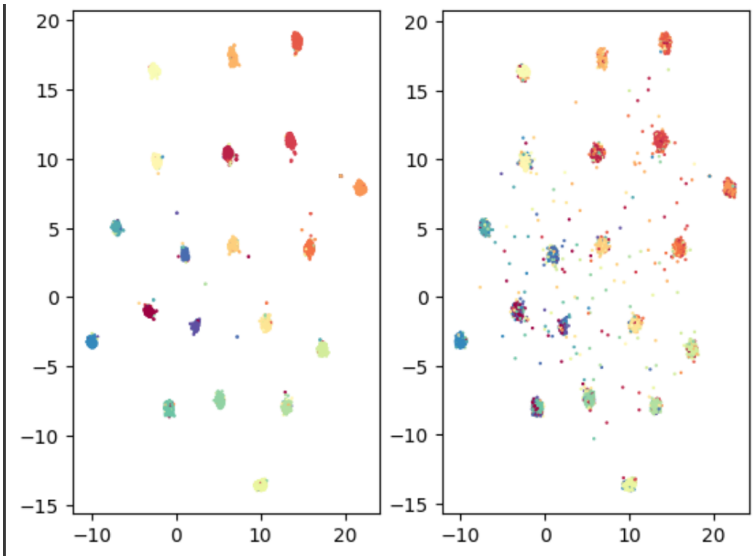
Hình 1: Bag of word



Hình 2: Tf-idf



Hình 3: Pre-train model



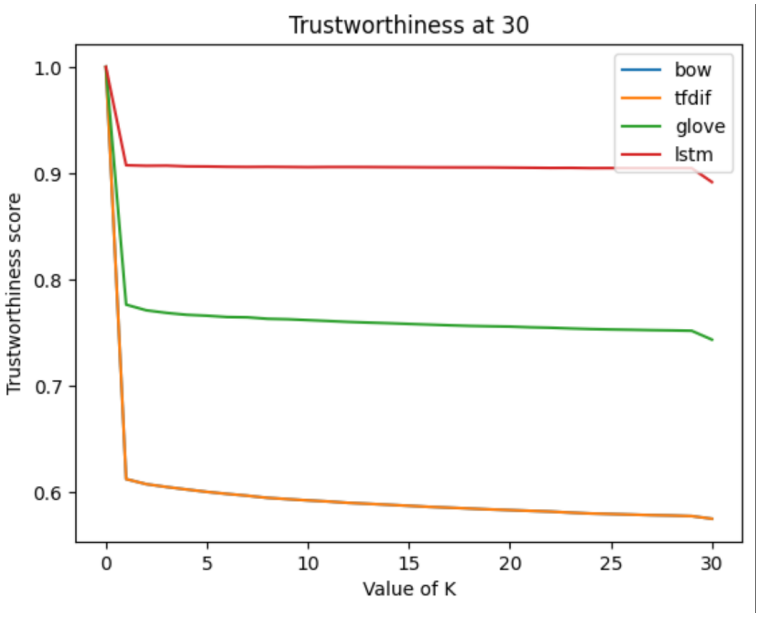
Hình 4: LSTM

Silhouette score

- Bag of word.
  - Tập huấn luyện: 0.47847196
  - Tập thử nghiệm: -0.1658458
- Tf-idf.
  - Tập huấn luyện: 0.47847196
  - Tập thử nghiệm: -0.1658458
- Pre-train model.
  - Tập huấn luyện: 0.5970492
  - Tập thử nghiệm: -0.109202586
- LSTM.
  - Tập huấn luyện: 0.8254086
  - Tập thử nghiệm: 0.09736216

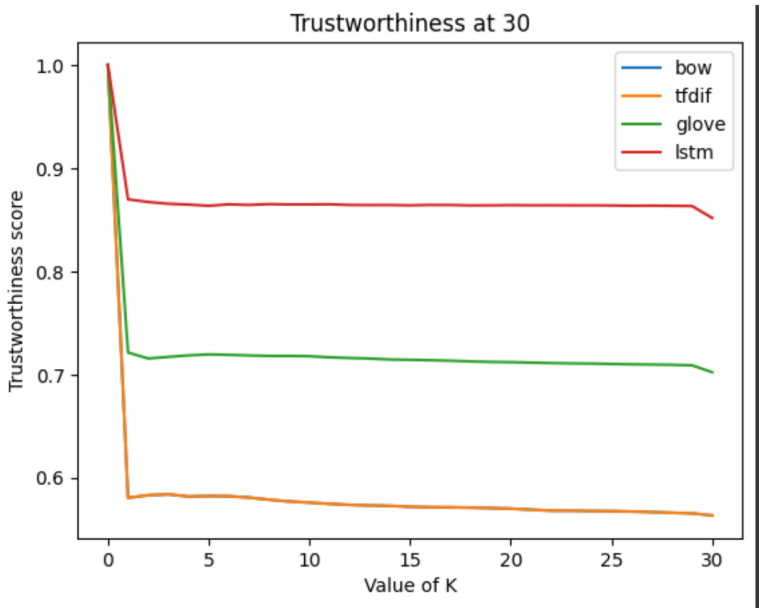
Trustworthiness score

- Tập huấn luyện.



Hình 5: Tập thử nghiệm

- Tập thử nghiệm.



Hình 6: Tập thử nghiệm

Nhận xét - Đánh giá

Đối với kết quả từ việc trực quan hóa dữ liệu, ta có thể thấy rằng ở tập huấn luyện, cả 4 phương pháp đều cho thấy sự liên kết cục bộ khá tốt khi có thể phân cụm được các đoạn văn bản có cùng nhãn. Tuy nhiên đối với cấu trúc toàn cục, phương pháp thứ 3 và 4 có phần thể hiện tốt hơn. Điều này có thể đến từ việc chúng ta hạn chế số lượng đặc trưng của hai phương pháp trên nên thông tin bị mất đi khá nhiều. Ngoài ra, phương pháp số 4 cho ra kết quả ít nhiễu hơn phương pháp số 3 do việc mô hình đã học được ngữ nghĩa hai chiều từ bi-LSTM và có thể vì bộ dữ liệu khá lớn nên phù hợp với phương pháp deep learning. Kết quả cũng tương tự đối với tập thử nghiệm, hai phương pháp đầu có output khá tương đồng và mất đi tính cục bộ lẫn toàn cục. Phương pháp số 3 có phần cải thiện hơn nhưng nhiễu vẫn còn rất nhiều. Và kết quả tốt nhất vẫn là phương pháp số 4 với việc giữ được cả cấu trúc cục bộ và toàn cục.

Tiếp theo, khi so sánh các Silhouette score kết quả cũng thể hiện sử dụng LSTM cho kết quả tốt nhất. Ngoài ra, phương pháp thứ 3 vẫn cho số liệu tốt hơn hai phương pháp còn lại.

Về Trustworthiness score, kết quả cho ta thấy được UMAP đã hoạt động tốt và giữ lại cấu trúc nhiều nhất với phương pháp số 4. Phương pháp số 3 tuy không bằng nhưng kết quả vẫn được xem là cao và chấp nhận được so với hai phương pháp còn lại. Vì số đặc trưng cao hơn nên Trustworthiness score của hai phương pháp đầu cho ra không được tốt.

Như vậy, nhờ UMAP ta có thể đánh giá được độ chính xác của các phương pháp nhúng. Từ đó có quyết định phù hợp với từng bộ dữ liệu.

Hướng mở rộng trong tương lai

- Tạo pipeline để tiền xử lý dữ liệu phù hợp cho mỗi phương pháp.
- Mở rộng giới hạn chiều cho phương pháp bag of word và tf-idf. Kết hợp với SVD hoặc PCA giảm thiểu số chiều trước khi đưa vào UMAP.
- Thực hiện các bước để lựa chọn được bộ siêu tham số của UMAP cho kết quả tốt nhất.
- Thử nghiệm với một tập dữ liệu khác.

Tài liệu

[1] L. McInnes, J. Healy, J. Melville, *UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction*.

[2] M. Belkin, P. Niyogi, *Laplacian eigenmaps and spectral techniques for embedding and clustering. In Advances in neural information processing systems*.

[3] M. Belkin, P. Niyogi, *Laplacian eigenmaps for dimensionality reduction and data representation*.

[4] W. Dong, C. Moses, K. Li, *Efficient k-nearest neighbor graph construction for generic similarity measures*.

[5] A. Jackson, *The mathematics of UMAP*.

[6] Curve fitting, Spicy Doccument of Curve fitting, Spicy

[7] Source code UMAP, Souce code

[8] Huỳnh Quang Vũ, Bài giảng Tôpô.

[9] T. Leinster, Basic Category Theory, Cambridge University Press, 2014.

[10] Wiki, LSTM LSTM

[11] G. Friedman, An elementary illustrated introduction to simplicial sets.

[12] B. Andrews, Lectures on Differential Geometry.

[13] D. Spivak, Metric realization of fuzzy simplicial sets.