

**ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

KHOA HỆ THỐNG THÔNG TIN

-----🙞🙜🕮🙞🙜-----

**MẠNG XÃ HỘI**

**MẠNG NHÚNG THƯA THỚT CÓ THUỘC TÍNH THÔNG QUA**

**TỔNG HỢP THÔNG TIN TƯƠNG ỨNG CỦA NHỮNG NÚT LÂN CẬN**

**(Sparse Attributed Network Embedding via Adaptively Aggregating Neighborhood Information)**

**Lớp:** IS403.L22.HTCL

**Giảng viên phụ trách:** Th.S Nguyễn Thị Minh Phụng

Thành viên:

Đoàn Thục Quyên – 18521320

Hoàng Thủy Tiên – 18521320

Dương Bảo Nam – 18521320

TP.HCM – 15/11/2021

Nội dung

[**I.** **GIỚI THIỆU:** 3](#_Toc88556046)

[**II.** **CÔNG TRÌNH NGHIÊN CỨU:** 6](#_Toc88556047)

[**A.** **Network Embedding:** 6](#_Toc88556048)

[**B.** **Attention Mechanism:** 10](#_Toc88556049)

[**III.** **TRÌNH BÀY VẤN ĐỀ** 11](#_Toc88556050)

[**IV.** **MÔ HÌNH ĐỀ XUẤT** 13](#_Toc88556051)

[**A.** **Việc lấy thông tin của những nút lân cận** 13](#_Toc88556052)

[**B.** **Framework** 15](#_Toc88556053)

[**C.** **Training** 19](#_Toc88556054)

[**D.** **Thảo luận** 20](#_Toc88556055)

[**V.** **THÍ NGHIỆM** 21](#_Toc88556056)

[**A. Tập dữ liệu và đường cơ sở** 21](#_Toc88556057)

[**B. Dự đoán liên kết** 23](#_Toc88556058)

[**C. Phân loại nút** 24](#_Toc88556059)

[**D. Trực quan hóa Mạng** 25](#_Toc88556060)

[**E. Nhạy cảm với Tham số** 25](#_Toc88556061)

[**VI.** **KẾT LUẬN** 26](#_Toc88556062)

1. **GIỚI THIỆU:**

Phương pháp học đại diện mạng - network representation learning (NRL), dùng để nối các nút trong mạng dưới dạng vector có chiều nhỏ, đã gây chú ý bởi tiềm năng trong các ứng dụng của mạng khác nhau. Gần đây, mạng nhúng có thuộc tính (attributed network embedding) kết hợp đồng thời cấu trúc mạng và các thuộc tính của nút đã có được những kết quả tốt trong NRL. Tuy nhiên, hầu hết các phương pháp hiện tại không thể có kết quả như mong muốn trong những mạng thưa thớt có thuộc tính (sparse attributed networks) vì thiếu cấu trúc của thông tin (structural information).

Để làm giảm tác động tiêu cực của sự thưa thớt (sparseness), chúng ta giới thiệu mô hình tên là AANI (Adaptively Aggregating Neighborhood Information)- thông tin về việc tổng hợp các cùng lân cận tương ứng. Cụ thể, AANI khai thác thông tin của nút hiện tại và vùng lân cận của nó để có được nhiều thông tin của nút đại diện. Tại thời điểm đó, AANI giới thiệu cơ chế chú ý (attention mechanism) để tổng hợp tương ứng với các thông tin đã được làm mịn dựa vào mức độ quan trọng của chúng. Chúng tôi thử nghiệm mở rộng trên ba tập dữ liệu thực tế để chứng minh AANI hoạt động tốt hơn các phương pháp nhúng hiện đại trong những mạng nhúng có thuộc tính (sparse attributed networks)

Với sự phát triển của kỹ thuật nhúng, nhiều nhà nghiên cứu đã chú ý đến thuật toán nhúng mạng (network embedding algorithms), nhằm mục đích ánh xạ các nút trong mạng thành các vectơ có số chiều nhỏ. Nhúng mạng là một phương pháp học đặc trưng được áp dụng trong các ứng dụng mạng khác nhau như:

* Phân loại nút (node classification),
* Dự đoán liên kết (link prediction)
* Trực quan hóa mạng (network visualization)

Các công trình nghiên cứu trước đây tập trung vào việc duy trì cấu trúc thông tin cục bộ và toàn cầu trong mạng, chẳng hạn như DeepWalk và Node2vec. Các phương pháp này thường thực hiện các bước đi ngẫu nhiên ngắn trên mạng để tạo ra những chuỗi nút như trong luận văn Word2vec. Trong khi đó, thông tin hữu ích khác trong mạng, chẳng hạn như những thuộc tính của nút, phần lớn lại bị bỏ qua.

Hầu hết, trong thực tế, các nút trong mạng luôn kết hợp với các thuộc tính của nó, và chúng được xem như là những thuộc tính của mạng.

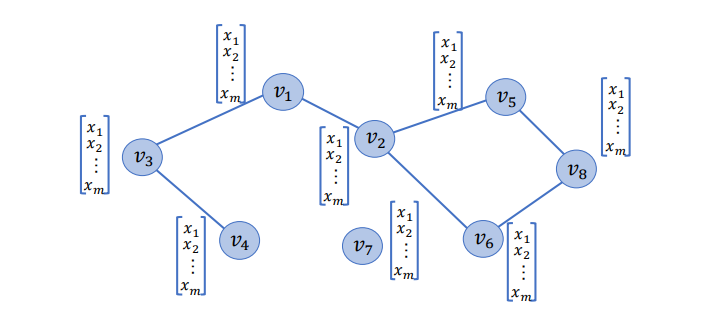
Ví dụ: trong mạng xã hội Facebook, độ tuổi, giới tính và nội dung bài đăng có thể được xem là các thuộc tính của nút nguồn (source node). Tương tự, trong mạng của các bài trích dẫn, sự xuất hiện của các từ quan trọng trong bài trích dẫn cũng có thể được xem là thuộc tính của nút.

Bởi vì, các thuộc tính nút thường có xu hướng tương đồng với các những nút lân cận, được gọi là hiệu ứng đồng tính. Nên các thuộc tính nút có thể tiết lộ cấu trúc mạng.

Nhiều phương pháp hiện có đã chứng minh được những lợi ích từ việc xem xét các thuộc tính:

- TADW lần đầu tiên sử dụng framework của kĩ thuật phân rã ma trận (matrix factorization) để kết hợp các thuộc tính văn bản của các nút.

- ASNE đề xuất một mô hình deep để giữ đồng thời cấu trúc và thuộc tính của vùng lận cận xung quanh. Nhưng hầu như chúng ta đều không thể đạt được khi mạng tương đối thưa thớt.



Hình 1: Ví dụ mạng lưới thưa thớt có thuộc tính. Các nút được đại diện bằng những hình tròn và thông tin thuộc tính của mỗi nút thì nằm bên cạnh nó.

Sự thưa thớt-sparseness có nghĩa là thiếu cấu trúc thông tin, vì có quá ít liên kết giữa các nút. Thuộc tính của nút là một thông tin phụ trợ quan trọng có thể làm giảm ảnh hưởng của sự thưa thớt trên mỗi nút đại điện. Tuy nhiên, hầu hết những phương pháp hiện tại chỉ có thể sử dụng thuộc tính của những nút hiện tại và bỏ quên thuộc tính của những nút lận cận của nó, do đó không thể tận dụng tối ưu những tiềm năng của thông tin thuộc tính. Ví dụ như trong hình 1, những phương pháp như AANE và ASNE chỉ sử dụng thuộc tính của những nút hiện tại v1 nhưng lại bỏ qua hoàn toàn những thuộc tính của các nút xung quanh. Gần đây, SEANO và ANRL cũng cải thiện bằng cách khai thác thuộc tính của những nút liền kề như v2 và v3. Mặc dù vậy nhưng họ không thể khai thác được những thuộc tính của những nút có thứ tự lân cận xa hơn như v4 và v5. Khai thác thuộc tính của nút có thứ tự lân cận xa hơn có thể làm giảm những ảnh hưởng tiêu cực của hiện tượng thưa thớt và lấy được nhiều thông tin của nút đại diện hơn.

Trong bài báo này, chúng ta đề xuất phương pháp mạng nhúng thưa thớt có thuộc tính (sparse attributed network embedding method) có tên là AANI (Adaptively Aggregate Neighborhood Information). Đầu tiên, chúng ta khai thác thông tin của những nút hiện tại và những nút lân cận của nó bằng cách làm min. Sau đó, chúng ta tổng hợp những thông tin đã được làm mịn bằng cơ chế chú ý dựa vào sự quan trọng tương ứng với từng nút đại diện. Sau khi có được thông tin tổng hợp, chúng ta đưa chúng vào một thuật toán perceptron[[1]](#footnote-2) nhiều lớp để lấy được những thông tin phi tuyến tính (non-linear information). Cuối cùng, để có được thông tin có cấu trúc, chúng ta đã thực hiện các bước đi ngẫu nhiên trong mạng và lấy số lớn nhất của những xác suất của nút hiện tại và những nút ngữ cảnh của nó.

1. **CÔNG TRÌNH NGHIÊN CỨU:**
2. **Network Embedding:**

Các công nghệ nhúng mạng có thể được bắt nguồn từ các phương pháp giảm chiều dữ liệu (đưa một vector có số chiều lớn, thường ở dạng thưa, về một vector có số chiều nhỏ, thường ở dạng dày đặc) dựa trên biểu đồ. Ví dụ như Locally Linear Embedding (LLE – thuật toán giảm chiều dữ liệu phi tuyến) và IsoMAP.

Những phương pháp này tìm hiểu cách nhúng dữ liệu trong khi duy trì kết cấu cục bộ manifold (đa tạp). Đa tạp tôpô *d* chiều là một không gian tôpô mà cục bộ của nó trông giống như một không gian euclidean *d* chiều ở các điểm gần nhau. Nói một cách chính xác, theo ngôn ngữ tôpô thì đa tạp *d* chiều là một không gian tôpô mà mỗi điểm có lân cận đồng phôi với tập con mở của . Ví dụ về đa tạp: nhìn một cách cục bộ thì có vẻ như bề mặt Trái Đất bằng phẳng chẳng khác nào không gian euclidean 2 chiều. Tuy nhiên, cấu trúc toàn cục lại là mặt cầu. Mặt cầu 3 chiều có thể xem là một đa tạp 2 chiều.

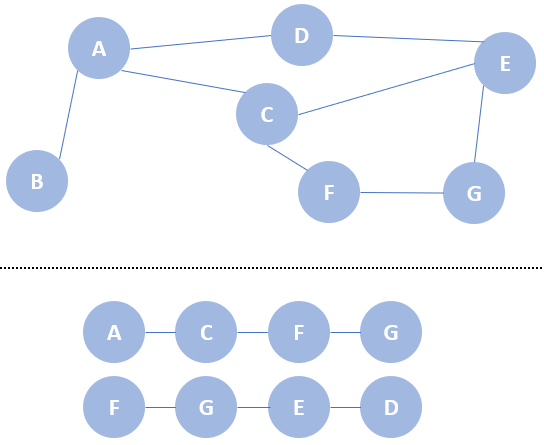
Vấn đề chính của những phương pháp này là chúng không thể được áp dụng cho các mạng quy mô lớn do độ phức tạp tính toán cao trong việc tính toán các vector riêng. Gần đây, neural networks đã có những bước phát triển vượt bậc trong nhiều lĩnh vực như xử lý ngôn ngữ tự nhiên và thị giác máy tính.

Một số thư viện sử dụng kĩ thuật Embedding như sklearn, Word2vec, FastText, ...v.v.

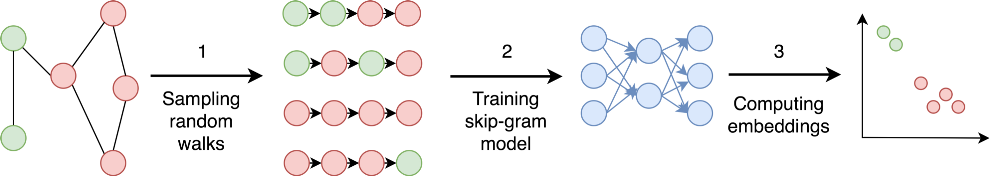
Word2vec có 2 model:

* CBOW: sử dụng các từ ngữ cảnh để dự đoán từ đích
* Skip-gram: sử dụng từ đích để dự đoán các từ ngữ cảnh xung quanh. ví dụ như text = "I love you so much". Khi dùng một window search có size 3 ta thu được : {(i,you),love},{(love,so),you},{(you,much),so}. Nhiệm vụ của nó là khi cho 1 từ center ví dụ là love thì phải predict các từ xung quanh là i, you.

Dựa trên ý tưởng chủ đạo từ word2vec, thuật toán DeepWalk thực hiện cắt ngắn các bước đi ngẫu nhiên tới các nút láng giềng trên mạng dưới dạng chuỗi (chuỗi nút) và đưa chúng vào mô hình skip-gram để tìm hiểu các nút đại diện từ đó dự đoán hiệu quả ngữ cảnh xung quanh của nó.



Hình 2: *Ví dụ: từ 2 đỉnh A và F, bước đi ngẫu nhiên với bước nhảy = 3 sẽ sinh ra các chuỗi như: A -> C -> F -> G, F -> G -> E -> D, ...*

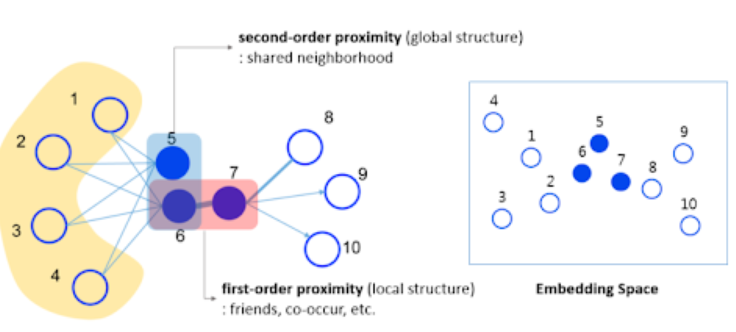


Hình 3: Các bước trong thuật toán Deepwalk

Node2vec mở rộng DeepWalk bằng cách sử dụng một chiến lược linh hoạt hơn để tạo chuỗi nút. Đó là thêm 2 thông số:

* *p* (hay còn lại là return parameter) quy định khả năng quay lại 1 nút trước đó trên bước đi ngẫu nhiên. Với p lớn thì ít có khả năng quay lại nút trước đó và ngược lại.
* *q* (hay còn lại là in-out parameter) quy định khả năng bước đi "gần" hay "xa" so với nút ban đầu. Với *q* > 1 thì bước đi ngẫu nhiên có xu hướng đi quanh quẩn nút ban đầu và ngược lại, đi ra xa khỏi nút ban đầu.

Ngoài các phương pháp dựa trên các bước đi ngẫu nhiên, LINE (Line Large-scale Information Network Embedding) đề xuất một hàm mục tiêu được thiết kế cẩn thận giúp khoảng cách bậc nhất và bậc hai giữa các nút trong cấu trúc mạng cục bộ và toàn cầu được bảo toàn.

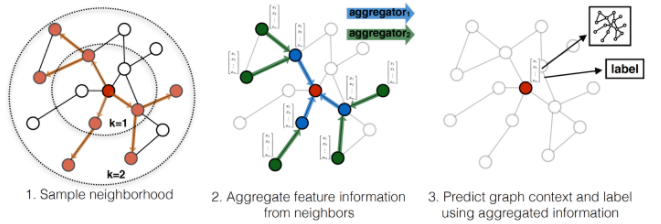


Hình 4: Bảo toàn cấu trúc mạng toàn cầu và cục bộ trong LINE

GraRep đề xuất một phương pháp nắm bắt thông tin quan hệ bậc *k (k>2)* để cải thiện hiệu suất biểu diễn của nút đại diện. Khác với những cách hoạt động trên, SDNE đề xuất một mô hình bán giám sát (semi-supervised model là mô hình khi có một lượng lớn dữ liệu nhưng chỉ 1 phần được gán nhãn) có thể khai thác thông tin phi tuyến tính hiệu quả và nắm bắt cấu trúc gần nhau tại cùng một thời điểm.

Các phương pháp nói trên chỉ có thể sử dụng cấu trúc thông tin trong mạng. Tuy nhiên, thuộc tính của các nút là thông tin quan trọng dùng để khai phá cấu trúc mạng. Người ta đã chứng minh rằng thông tin thuộc tính có thể có lợi đến việc biểu diễn nút. TADW lần đầu tiên đề xuất DeepWalk liên kết với văn bản, có thể kết hợp các tính năng văn bản của các nút vào các biểu diễn mạng. Tuy vậy, TADW chỉ có thể xử lý các thuộc tính văn bản. AANE là một phương pháp phân tán có thể học cách biểu diễn giảm chiều thông qua việc phân tích mối quan hệ quy kết và tăng cường sự khác biệt nhúng giữa các nút được kết nối. ASNE đề xuất một mô hình deep để giữ mối quan hệ phức hợp giữa cấu trúc mạng và các thuộc tính của nút.

Tuy nhiên, các phương pháp này chỉ sử dụng thông tin thuộc tính của nút hiện tại nhưng bỏ qua các thuộc tính của vùng lân cận của nó, điều này là thiếu sót trong các mạng phân bổ thưa thớt. SEANO và ANRL sử dụng các thuộc tính của các nút lân cận để nâng cao hiệu quả biểu diễn của các nút đại diện. Tuy nhiên, chúng không thể khai thác các thuộc tính vùng lân cận có bậc cao hơn, điều này có thể làm giảm bớt tác động tiêu cực của sự thưa thớt. GraphSAGE tạo các lần nhúng bằng cách lấy mẫu và tổng hợp các đặc trưng (feature) từ vùng lân cận của nút.



Hình 5: Các bước hoạt động của GraphSAGE

Khác với GraphSAGE, chúng ta đã áp dụng một tập hợp chiến lược khác và giới thiệu cơ chế chú ý (attention mechanism) để phân bổ trọng số thích ứng dựa trên tầm quan trọng của thông tin. Có một số công trình nghiên cứu khám phá trong cách học bán giám sát để kết hợp thông tin nhãn cho thuộc tính mạng được nhúng, chẳng hạn như TriDNR và LANE. Tuy nhiên, thường khó lấy thông tin nhãn của nút trong các tình huống của thế giới thực. Vì việc thu thập dữ liệu có nhãn tốn rất nhiều thời gian và có chi phí cao. Rất nhiều loại dữ liệu thậm chí cần phải có chuyên gia mới gán nhãn được (ảnh y học chẳng hạn). Ngược lại, dữ liệu chưa có nhãn có thể được thu thập với chi phí thấp từ internet.

Hơn nữa, có một số khám phá các biểu diễn trong mạng động (dynamic networks). DNE mở rộng mô hình skip-gam trong môi trường động với hiệu quả cao. JODIE nghiên cứu nhúng động của người dùng và các item từ một chuỗi các tương tác thời gian. Trong bài báo này, chúng ta tập trung vào nhúng mạng tĩnh (static network).

1. **Attention Mechanism:**

Gần đây, cơ chế chú ý đã chứng minh được hiệu quả của nó trong nhiều lĩnh vực như xử lý ngôn ngữ tự nhiên và thị giác máy tính. Cơ chế chú ý được lấy cảm hứng từ cơ chế thị giác của con người, khi cơ chế thị giác của con người phát hiện một vật phẩm, nó thường sẽ không quét toàn bộ cảnh từ đầu đến cuối; thay vào đó, nó sẽ luôn tập trung vào một phần cụ thể theo nhu cầu của người đó. Khi một người nhận thấy rằng đối tượng họ muốn chú ý thường xuất hiện trong một phần cụ thể của cảnh, họ sẽ biết rằng trong tương lai, đối tượng sẽ nhìn vào phần đó và có xu hướng tập trung sự chú ý của họ vào khu vực đó. Trong lĩnh vực máy học, chú ý là một phương pháp để tổng hợp một tập các nút thành một nút, thường thông qua một nút tra cứu. Kết quả thường được gọi là nút ngữ cảnh, vì nó chứa ngữ cảnh phù hợp với thời gian hiện tại.

Một vài công trình nghiên cứu đã đưa cơ chế chú ý vào mạng nhúng sử dụng thuộc tính. SANE áp dụng cơ chế chú ý để đánh giá hợp lí sức mạnh tương tác giữa mỗi nút và các nút ngữ cảnh của nó để giải quyết vấn đề thuộc tính thưa thớt. GAT đề xuất một kiến trúc mạng nơ-ron hoạt động trên dữ liệu mạng thông qua tận dụng các lớp tự chú ý được che đậy.

Khác với SANE và GAT, mô hình đề xuất của chúng ta sử dụng cơ chế chú ý để đánh giá tầm quan trọng của thông tin được thu thập từ nút hiện tại và vùng lân cận của nó một cách hợp lí.

1. **TRÌNH BÀY VẤN ĐỀ**

* Giới thiệu một số ký hiệu được sử dụng trong bài báo:



Biểu thị một mạng phân bổ thưa thớt

*Trong đó:*

- là tập các nút của đồ thị

- là biểu diễn cạnh mã hóa nối từ nút tới nút của đồ thị

- mã hóa thông tin thuộc tính

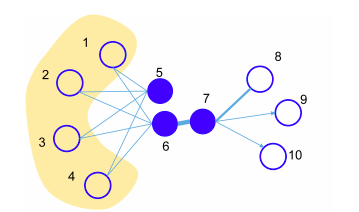
- là thuộc tính của nút

- Độ thưa thớt chỉ ra rằng bộ nhớ tập cạnh bằng bộ nhớ tập nút trong mạng:



* ***Đưa ra các định nghĩa về vấn đề:***

***Định nghĩa 1:*** Cấu trúc lân cận (Structure Proximity) biểu thị sự gần gũi có thể được quan sát bằng các liên kết. Nếu tồn tại một cạnh giữa nút và điều đó chỉ ra lân cận bậc nhất (first-order proximity). Nếu nút nằm trong ngữ cảnh (context) của , điều đó chỉ ra lân cận bậc cao (high-order proximity). Mức độ lân cận bậc nhất và bậc cao đều là cấu trúc lân cận.



Ví dụ:

Khi 2 user A, B hoàn toàn không liên quan đến nhau, user A thích các sản phẩm P1, P2, P3, P4; user B thích các sản phẩm P1, P3, P4. Vậy nhiều khả năng rằng thói quen mua đồ của 2 user là khá giống nhau và hệ thống tiến hành gợi ý thêm cho user B sản phẩm P2, ... Hay ví dụ như hình trên, nút 5 và 6 không connect trực tiếp với nhau nhưng có chung nhiều nút "làng giềng" nên giả định rằng 2 nút này sẽ khá giống nhau, tương đồng với nhau về mặt context

Vùng lân cận bậc nhất cho biết vùng lân cận trực tiếp trong mạng, hay còn có thể được xem như là lân cận cục bộ. Các mức độ lân cận bậc cao cho biết mức độ giống nhau của vùng lân cận, hay còn có thể được xem là vùng lân cận toàn cầu. Các nút chia sẻ những lân cận giống nhau cũng giống nhau, mặc dù chúng không kết nối trực tiếp. Một cách phổ biến để nắm bắt vùng lân cận bậc cao là thực hiện các bước đi ngẫu nhiên trên mạng để tạo chuỗi nút. Nếu hai nút xuất hiện trong cùng một chuỗi nút, chúng được coi là xuất hiện cùng một bối cảnh. Trong phần còn lại của bài báo này, để đơn giản hóa chúng ta sử dụng thuật ngữ "Nút ngữ cảnh" (context nodes) để biểu thị các nút được kết nối trực tiếp và các nút trong cùng ngữ cảnh.

***Định nghĩa 2:*** Thuộc tính lân cận (Attribute Proximity) biểu thị sự giống nhau của các thuộc tính giữa nút và nút . Mối tương quan giữa thuộc tính được mã hóa và thể hiện sự lân cận của thuộc tính.

***Định nghĩa 3:*** Nhúng mạng thuộc tính thưa thớt

Cho một mạng phân bổ thưa thớt:

,

Trong đó:



Chúng ta đặt mục tiêu tìm hiểu mỗi nút như là vector chiều thấp trong đó . Ánh xạ hàm có thể duy trì cấu trúc lân cận và thuộc tính lân cận đồng thời

1. **MÔ HÌNH ĐỀ XUẤT**

Trong phần này, chúng ta giới thiệu cách giải quyết thông tin của những nút lân cận bằng cách làm mịn. Rồi sau đó, chúng ta sẽ giới thiệu mô hình đề xuất.

1. **Việc lấy thông tin của những nút lân cận**

Trong mạng thưa thớt có thuộc tính, nó khó để bắt được structure proximity bởi vì thiếu cấu trúc thông tin ở trong mạng. Thuộc tính nút có thể làm giảm vấn đề gây ra bởi sự thưa thớt. Tuy nhiên, những phương pháp hiện tại chỉ sử dụng thuộc tính của nút hiện tại nhưng bỏ qua những thuộc tính của những nút lân cận của nó. AANI cải thiện chất lượng của nút đại diện thông qua khai thác thuộc tính - thông tin của nút hiện tại cũng như của những nút lân cận của nó. Ngoài ra, cấu trúc thông tin có thể ngầm lấy từ quá trình tìm kiếm những nút lân cận.

* **Neighborhood attributes smoothing(Làm mịn các thuộc tính của các nút lân cận)**

Để khai thác thông tin của thuộc tính, chúng ta cần tìm kiếm những nút lân cận nằm trong vòng phạm vi k nút tính từ nút ở giữa (hop), kí hiệu là . Sau đó, chúng ta tính trung bình của những thuộc tính của những nút trong tới nút thứ k có thuộc tính được làm mịn, kí hiệu là .

Như ví dụ như hình 1, nút lân cận mà cách 1 nút từ nút v1 là (v2, v3) (hop=1). Sau đó, chúng ta tính trung bình những thuộc tính của (v2,v3) để đạt được thuộc tính đã được làm mịn .

Để khai thác thêm thông tin của thuộc tính, chúng ta tìm kiếm những vùng nút lân cận lớn hơn (v2,v3,v4,v5,v6) – tìm trong phạm vị của những nút cách 2 hop từ nút v1 và tính trung bình những thuộc tính, ta được . Thu thập và làm mịn thông tin thuộc tính của những nút trong có thể mang lại hai lợi ích. Thứ nhất, vì có nhiều thông tin thuộc tính hơn có thể được sử dụng, AANI đạt được nhiều thông tin của nút đại điện trong mạng thưa thớt có thuộc tính. Thứ hai, vì những loại của thuộc tính thì giống nhau ở các nút lân cận nên nhiễu của những thuộc tính phát sinh từ nút trung tâm có thể được làm mịn bằng cách tính trung bình của các những thuộc tính của các nút trong .

Thông qua việc mở rộng phạm tìm kiếm, chúng ta có thể khai thác được nhiều thông tin của thuộc tính. Nếu một nút đứng độc lập như nút v7 trong hình 1, chúng ta cho

, với mọi giá trị k.

Để thuận tiện cho việc kí hiệu, chúng ta sử dụng thay vì , lúc đó thuộc tính của nút hiện tại được xem như thuộc tính được làm mịn thứ 0. Vì vậy, chúng ta tổng quát hóa lên của như sau:

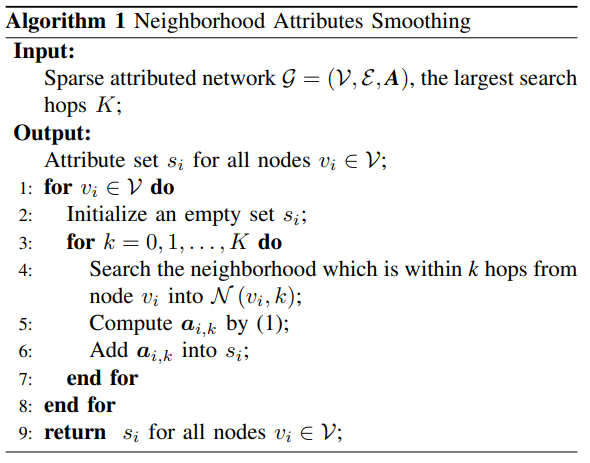
Trong đó:

- vi ∈ V

- k ∈ {0, 1, … K} với K là những hop lớn nhất

Sau đó chúng ta có thể đặt một thuộc tính để chuẩn bị cho bước training tiếp theo.

Cách làm mịn những thuộc tính của các nút lân cận bằng thuật toán như sau:



* **Neighborhood structure capturing (Lấy được cấu trúc của những nút lân cận)**

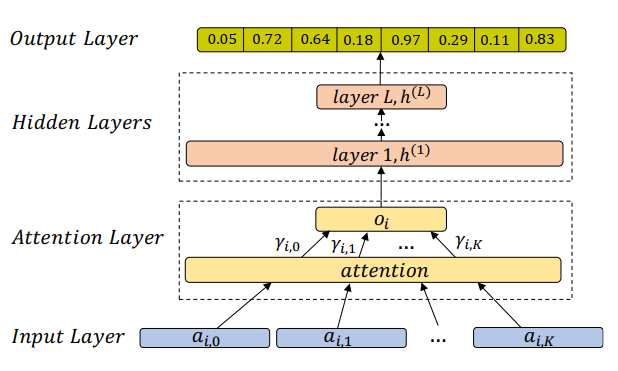
Việc tìm kiếm k hop – số bước nhảy đến các nút lân cận từ nút trung tâm cũng ngầm lấy được cấu trúc thông tin.

Ví dụ, khi chúng ta tìm ra vùng lân cận (v2,v3) – 1 hop từ v1, chúng ta có thế lấy first-order proximity theo định nghĩa 1 trong phần II, đã được chỉ cạnh liên kết trực tiếp giữa các nút.

Với giá trị k lớn hơn, phạm vị rộng hơn sẽ giúp tìm kiếm và bắt lấy được nhiều cấu trúc thông tin.

1. **Framework**

Tổng quan về AANI được thể hiện như sau:



*Hình 2: Tổng quan mô hình đề xuất*

Trong lớp Input, đại diện thuộc tính bậc k đã được làm mịn, đại diện thuộc tính của nút hiện tại.

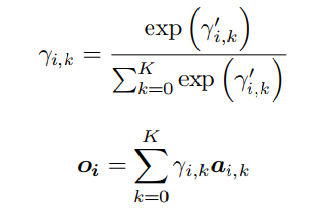
**Giới thiệu từng lớp mô hình được đề xuất:**

*Lớp Input*: là tập hợp với tất cả các nút . Tập hợp chứa thông tin thuộc tính của nút hiện tại và vùng lân cận của nó, và đại diện cho thuộc tính làm mịn bậc thứ *k*.

*Lớp Attention:* Thông tin thuộc tính được làm mịn trong các phạm vi tìm kiếm khác nhau có ảnh hưởng khác nhau đến nút đại diện. Trên thực tế, rất khó để biết thuộc tính nào trong số thuộc tính bậc k được làm mịn là quan trọng hơn. Với mỗi nút trong mạng, AANI sử dụng cơ chế chú ý để gán trọng số một cách thích ứng cho . Nút đại diện càng quan trọng, trọng số được ấn định sẽ càng lớn. Tương tự đối với mỗi nút , một trọng số dương được đặt trên mỗi thuộc tính để chỉ ra tầm quan trọng tương đối trên nút đại diện. Về mặt hình thức, mạng lưới chú ý có thể được biểu diễn như sau:

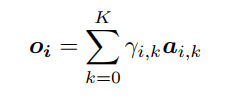


Trong đó là một vectơ mức độ thuộc tính ẩn được học trong quá trình training. Chúng ta sử dụng hàm kích hoạt ReLU = max (0; ), ReLU đơn giản lọc các giá trị < 0, để thu thập thêm thông tin phi tuyến. Khi đó, điểm chú ý trong đó được chuẩn hóa bằng hàm softmax:



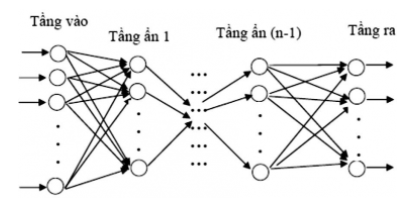
Softmax là hàm trung bình mũ. Hàm softmax tính toán xác suất xảy ra của một sự kiện, có nghĩa là nó sẽ tính khả năng xuất hiện của một class trong tổng số tất cả các class có thể xuất hiện. Sau đó, xác suất này sẽ được sử dụng để xác định class mục tiêu cho các input:

* Xác suất sẽ luôn nằm trong khoảng (0:1].
* Tổng tất cả các xác suất bằng 1.



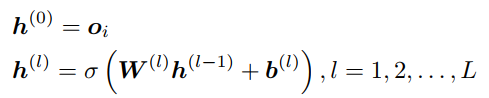
Thông tin thuộc tính được tổng hợp cuối cùng được tổng hợp bằng tổng của tất cả thuộc tính được làm mịn bậc k được tính theo trọng số chú ý tương ứng của chúng. Đặc biệt, kết hợp thông tin thuộc tính của nút hiện tại và khu vực lân cận của nó cũng như đóng góp tương ứng của chúng.

*Lớp Hidden:* Xếp chồng nhiều lớp phi tuyến tính có thể dễ nắm bắt mối quan hệ phi tuyến tính hơn và nâng cao chất lượng nhúng. Nhiều phương pháp hiện có như DeepWalk và node2vec bị giới hạn ở khả năng của mạng shallow (sử dụng một số lượng nhỏ lớp hidden) và không nắm bắt được thông tin phi tuyến tính. Do đó, chúng ta cung cấp cho thông tin thuộc tính được tổng hợp một mạng nhiều tầng truyền thẳng (MLP: Multilayer perceptron) để nắm bắt mối quan hệ phức tạp giữa cấu trúc mạng và các thuộc tính của nút. Một mạng MLP tổng quát là mạng có n (n≥2) lớp (thông thường lớp Input không được tính đến): trong đó gồm một lớp đầu ra (lớp thứ n) và (n-1) lớp ẩn.



Hình 6:Mạng MLP

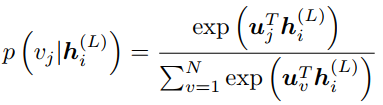
Biểu diễn của các lớp hidden được định nghĩa là , có thể được ký hiệu như sau:



Trong đó:

* σ (·) đại diện cho các hàm có thể kích hoạt có như ReLU, sigmod và tanh.
* và là những tham số có thể đào tạo được của lớp thứ *l* và *L* là tổng số các lớp ẩn.
* Sử dụng làm nút đại diện cuối cùng.

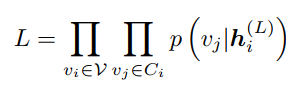
*Lớp Output:* Lấy cảm hứng từ mô hình skip-gam được sử dụng trong node2vec, chúng ta thực hiện các bước đi ngẫu nhiên trên mạng để tạo ngữ cảnh corpus (là 1 dữ liệu tập hợp các văn bản, ngôn ngữ đã được số hoá) với kích thước cửa sổ *t* cho mỗi nút . Để nắm bắt cấu trúc lân cận, chúng ta sử dụng đại diện cuối ***h****i(L)* của nút để dự đoán xác suất xuất hiện của các nút ngữ cảnh của nó , có thể được ký hiệu là:

 (6)

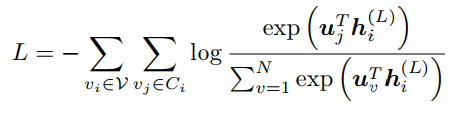
Trong đó:

* ***U*** là ma trận trọng số để dự đoán ngữ cảnh
* là biểu diễn trừu tượng khi nút được coi là nút ngữ cảnh. tương ứng với cột thứ *j* trong ***U***.

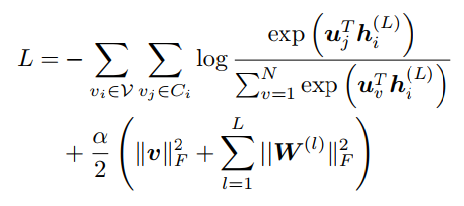
Do đó, hàm mục tiêu nhằm mục đích tối đa hóa sự xuất hiện xác suất của nút và các nút ngữ cảnh của nó trong tập , có thể được ký hiệu là:

(7)

Để đào tạo mạng nơ-ron hiệu quả, chúng ta có thể khiến việc tính toán dễ dàng hơn bằng cách biến đổi về bài toán Maximum Log-likelihood, theo (6), (7) có thể được viết lại thành:



Hàm mất mát cuối cùng của mô hình là hàm để đánh giá độ tốt của model (tức làm dự đoán) được ký hiệu như sau:

(9)

Trong đó:

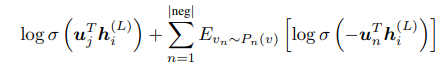
α là hệ số điều chỉnh định mức

và ***W****(l)* là các ma trận trọng số cho lớp attention và các lớp hidden.

1. **Training**

Việc dùng công thức số (9) thì tốt hơn bởi vì nó yêu cầu đi qua tất các nút trong mạng khi tính toán xác suất xảy ra của .

Để giảm việc tính toán phức tạp, chúng ta áp dụng chiến lược lấy mẫu phủ định được đề xuất bởi những mẫu phủ định trước đó theo như một số phân bố nhiễu. Chi tiết như sau, cho cặp nút ngữ cảnh xác định (vi, vj), chúng ta có:



Trong đó:

- là hàm sigmod[[2]](#footnote-3)

- |neg| : số lượng mẫu âm (negative samples).

Ta đặt và là degree của nút .

Bằng cách này, mô hình của ta đã bảo toàn được structure proximity và attribute proximity trong một framework nhất định. Hơn nữa, chúng ta khai thác thông tin của các nút lân cận trong mạng thưa thớt có thuộc tính để cải thiện nút đại điện. Để giảm thiểu hàm đối tượng, chúng ta sử dụng thuật toán gradient ngẫu nhiên (stochastic gradient algorithm) để tối ưu hóa cho đến khi mô hình hội tụ. Tất cả các tham số của mô hình được ký hiệu là Θ và quá trình training tiếp theo được tóm tắt trong thuật toán 2.

*Thuật toán 2: Training*

* + **Đầu vào**: Mạng thưa thớt có thuộc tính G=(V,E,A), window size b, những bước đi mỗi đỉnh là γ, độ dài bước đi t, hệ số điều chỉnh (regularizer coefficient)[[3]](#footnote-4) α, kích thước nhúng (embedding size) d;
  + **Đầu ra**: Nút đại diện Y thuộc ma trận |V| x d

1. Xây dựng tập nút ngữ cảnh C bằng cách bắt đầu thực hiện những bước đi ngẫu nhiên γ lần với chiều dài t cho mỗi nút.
2. Xây dựng tập những thông tin thuộc tính cho tất cả các nút bằng thuật toán 1
3. Khởi tạo ngẫu nhiên cho tất cả biến Θ
4. Lặp nếu chưa hội tụ:
5. Lấy mẫu một tập các nút nhỏ với ngữ cảnh của nó và tập những thông tin thuộc tính đã được làm mịn.
6. Tính gradient của hàm mất mát (loss function) dựa vào (9)
7. Nâng cấp những tham số trong neural network
8. Kết thúc lặp
9. Trả về nút đại diện Y =
10. **Thảo luận**

*Các nút mới:* Vì nút đi đến liên tục là một tình huống quan trọng đối với các mạng đang phát triển, phương pháp của chúng ta cung cấp một cách khả thi để đối phó với các nút mới. Đối với nút mới , chúng ta có thể nhận được tập tương ứng theo *Thuật toán 1*. Sau đó, chúng ta có thể đưa vào mô hình đã được huấn luyện để có được sự biểu diễn của nút . Đặc biệt, vì chúng ta chỉ lấy thông tin thuộc tính làm đầu vào, phương pháp của chúng ta sẽ vẫn hoạt động khi nút mới bị cô lập không có cạnh.

*Thuộc tính không đầy đủ:* Nút mới có thuộc tính không hoàn chỉnh là một tình huống có thể xảy ra trong các mạng được phân bổ. Thuộc tính của vùng lân cận có thông tin bổ sung có thể làm giảm ảnh hưởng của vấn đề này. Hơn nữa, cơ chế chú ý tổng hợp thông tin thuộc tính từ nút hiện tại và vùng lân cận của nó một cách thích hợp sẽ rất hữu ích cho việc biểu diễn nút được mịn hơn trong trường hợp này.

# **THÍ NGHIỆM**

Trong phần này, chúng ta tiến hành các thử nghiệm mở rộng với 3 datasets để thấy được hiệu quả của mô hình đề xuất trong các mạng phân bổ thưa thớt.

## **A. Tập dữ liệu và đường cơ sở**

Tập dữ liệu: Chúng ta tóm tắt số liệu thống kê của 3 datasets trong Bảng I. Mật độ biểu thị mức độ thưa thớt của mạng. Thông tin chi tiết như sau:

* Cora1 chứa 2.708 bài báo dưới dạng nút và 5.249 liên kết trích dẫn dưới dạng cạnh. Những bài báo này được chia thành 7 thể loại. Thuộc tính của mỗi nút là một vectơ nhị phân có kích thước là 1433 cho biết sự xuất hiện của các từ khóa quan trọng trong các bài báo.
* Citeseer chứa 3,312 ấn phẩm của loại như: Agents, ML, AI, DB, IR và HCI. Có 4,732 liên kết trong mạng và phân bổ của mỗi nút là vector có kích thước 3.703.
* Pubmed chứa 19.717 bài báo sinh học được chia thành ba loại như “Bệnh đái tháo đường có kinh nghiệm”, “Bệnh đái tháo đường týp 1” và “Bệnh đái tháo đường týp 2”. Thuộc tính trong Pubmed được biểu diễn bằng một vectơ có kích thước là 500.

Đường cơ sở: Để chứng minh tính hiệu quả của AANI, ta so sánh nó với một số phương pháp hiện đại. Vì AANI kết hợp cả cấu trúc mạng và thuộc tính nút, ta chọn đường cơ sở từ hai khía cạnh. Phương pháp đầu tiên chỉ sử dụng cấu trúc mạng như DeepWalk, LINE và SDNE. Phương pháp thứ 2 là các phương pháp sử dụng cả cấu trúc mạng lưới và các thuộc tính nút như AANE và ASNE. Chúng tối cũng tiến hành hai thử nghiệm có tên AANI / cur và AANI / noAtt để cho thấy hiệu quả của việc tổng hợp thông tin vùng lân cận và cơ chế chú ý. Chi tiết được minh họa như sau:

* DeepWalk: DeepWalk thiết kế một lối đi ngẫu nhiên được cắt ngắn để tạo chuỗi nút và đưa chúng vào mô hình skip-gram để tìm hiểu các cách biểu diễn nút.
* LINE: duy trì sự gần gũi bậc 1 hoặc bậc 2 trong mạng bằng cách tối ưu hóa hàm mục tiêu.
* SDNE : SDNE đề xuất một mô hình sâu có thể thu được cấu trúc mạng phi tuyến tính cao.
* AANE : AANE là một phương pháp nhúng hiệu quả kết hợp sự gần gũi về cấu trúc và sự tương đồng của thuộc tính theo cách phân phối.
* ASNE : ASNE áp dụng một mô hình sâu để nắm bắt được sự gần nhau về cấu trúc và sự gần gũi thuộc tính một cách đơn giản.
* AANI / cur : AANI / cur (current) là một trong những biến thể của AANI. AANI / cur chỉ sử dụng thuộc tính của nút hiện tại với việc đặt bước nhảy tìm kiếm lớn nhất là 0.
* AANI / noAtt (no attention) là một biến thể khác ​​mà không có lớp chú ý được thể hiện trong Hình 2.

Đối với LINE và SDNE, ta sử dụng O1penNE2, một nền tảng mã nguồn mở dành cho mạng em-bedding. Các tham số cho đường cơ sở được điều chỉnh để tối ưu. Chúng ta đặt kích thước d là 128, kích thước cửa sổ b là 10, chiều dài bước đi t là 80, số lần đi trên mỗi nút la 10, mẫu âm là 10. Đối với AANI, số lượng nơ ron trong mỗi lớp được thể hiện trong Bảng II. Chúng ta đặt bước nhảy tìm kiếm lớn nhất K (2) trong AANI và AANI / noAtt để cân bằng hiệu suất với độ phức tạp tính toán. Ở phần cuối của phần này, chúng ta sẽ chỉ ra ảnh hưởng của giá trị của K.

## **B. Dự đoán liên kết**

Dự đoán liên kết đánh giá khả năng của các biểu diễn nút trong việc tái tạo lại cấu trúc mạng dựa trên thông tin hiện có. Chúng ta tạo tập dữ liệu được gắn nhãn của các cạnh. Lấy mẫu ngẫu nhiên 50% các liên kết hiện có trong một số lượng tương đương các liên kết không tồn tại dưới dạng các trường hợp tích cực và các trường hợp tiêu cực. Tập dữ liệu được gắn nhãn của các cạnh được hình thành bởi cả thể hiện tích cực và tiêu cực. Sau đó, sử dụng mạng còn lại để đào tạo các phương pháp nhúng. Sau khi có được các biểu diễn nút cho mỗi nút, ta sử dụng các biểu diễn này để thực hiện nhiệm vụ dự đoán liên kết trong tập dữ liệu được gắn nhãn. Chúng ta xếp hạng cả hai trường hợp tích cực và tiêu cực theo hàm tương tự cosine. Khu vực dưới ROC Curve (AUC) được sử dụng làm thước đo đánh giá.

Giá trị AUC cao hơn cho thấy hiệu suất tốt hơn. Kết quả dự đoán liên kết được trình bày trong Bảng III. Sử dụng màu xanh lam để làm nổi bật các chiến thắng và tóm tắt các đặc điểm sau:

* AANI Đạt được hiệu suất tốt nhất trong tất cả các bộ dữ liệu. So với kết quả tốt nhất trong đường cơ sở, phương pháp của chúng ta tạo ra một sự cải thiện là 9.08% trên Cora và 10.16% trên Pubmed. Kết quả cho thấy hiệu quả của AANI trong việc giải mã cấu trúc mạng.
* Vì DeepWalk, LINE và SDNE chỉ sử dụng thông tin struc-tural nên hiệu suất kém. Thú vị hơn, Chúng ta nhận thấy rằng Deepwalkperforms tốt hơn, chủ yếu Bởi vì DeepWalk đó có thể làm cấu trúc mạng tốt hơn thông qua đi bộ ngẫu nhiên bị cắt ngắn.
* Cả AANE và ASNE có được sự cải thiện so vớiwith Các phương pháp chỉ sử dụng thông tin cấu trúc.. Tuy nhiên, hiệu suất vẫn kém hơn AANI vì bỏ qua thông tin của các nút lân cận.

## **C. Phân loại nút**

Phân loại nút là một nhiệm vụ quan trọng trong phân tích mạng.Tương tự như các phương pháp trước đây [7] và [8], chúng ta sử dụng Micro-F1 và Macro-F1 làm số liệu để đo lường phân loại nút hiệu suất. Các giá trị cao hơn của cả hai số liệu minh chứng cho hiệu suất tốt hơn. Sau khi có được các biểu diễn nút, chúng ta ngẫu nhiên lấy mẫu 30% nút được dán nhãn để đào tạo một bộ phân loại SVM và các nút còn lại được sử dụng để kiểm tra hiệu suất. Ta lặp lại quá trình này 10 lần và báo cáo kết quả trung bình trong Bảng IV. Tóm lại, ta có các quan sát theo dõi:

* AANI đạt được hiệu suất tốt nhất đánh bại tất cả các dòng cơ sở. So với kết quả tốt nhất, phương pháp của chúng ta được cải thiện 17,6%, 7.30% và 4,31% trên Micro-F1 tương ứng với Citeseer, Cora và Pubmed. Kết quả cho thấy việc khai thác thông tin của nút lân cận có thể nâng cao hiệu suất của các đại diện nút.
* Các phương pháp chỉ sử dụng thông tin cấu trúc kém như LINE và SDNE. Những phương pháp này không thể có đủ thông tin cấu trúc khi mạng thưa thớt. DeepWalk hoạt động tốt hơn. Tuy nhiên, DeepWalk không thể đạt được kết quả thỏa đáng trên Citeseer, khi chứa nhiều nút bị cô lập.
* Các phương pháp sử dụng cả cấu trúc mạng và thuộc tính nút thực Hiện tốt hơn LINE và SDNE. Kết quả cho thấy lợi ích của các thuộc tính. Tuy nhiên, AANE và ASNE không thể đạt được kết quả đầy hứa hẹn so với DeepWalk trên Cora và Pubmed. Kết quả cho thấy chỉ sử dụng thuộc tính của nút hiện tại là không đủ trong các mạng được gán thưa thớt.
* Cuối cùng, AANI / cur hoạt động kém vì nó chỉ sử dụng thuộc tính của nút hiện tại nhưng bỏ qua các thuộc tính của khu phố của nó.. AANI / noAtt hoạt động kém vì nó trung bình theo thứ tự k-th làm cho thuộc tính Si tốt hơn nhưng bỏ qua tầm quan trọng tương ứng của chúng đối với sự phẫn nộ của nút. Chúng cho thấy sự cần thiết của tất cả các các thành phần thiết yếu của AANI.

## **D. Trực quan hóa Mạng**

Để hiển thị thêm kết quả nhúng của AANI, chúng ta hình dung các biểu diễn nút bằng cách sử dụng t-SNE. Bị giới hạn bởi không gian, chúng ta chỉ đăng kết quả của ba đại diện baselineson Cora. Kết quả trực quan hóa được hiển thị trong Hình 3.

Kết quả của AANE là kém, có thể liên quan đến hoạt động phân hủy của ma trận thuộc tính. Các hình ảnh trực quan của SDNE và ASNE không được tách biệt tốt, trong đó các nút với các nhãn khác nhau được trộn lẫn với nhau. So sánh với các phương pháp này, AANI có thể đạt được các cụm và sơn nhỏ gọn hơn. Do đó, AANI có thể đạt được hiệu suất tốt hơn ba ứng dụng mạng thực tế.

## **E. Nhạy cảm với Tham số**

Chúng ta điều tra độ nhạy cảm của mô hình được đề xuất trong phần này. Chúng ta tiến hành thí nghiệm này trên ba bộ dữ liệu và báo cáo kết quả phân loại khi tỷ lệ đào tạo là 30%.

Kích thước. Ảnh hưởng của kích thước kích thước đối với hiệu suất classifi-cation được thể hiện trong Hình 4 (a). Khi kích thước nhỏ, có thể kết hợp nhiều thông tin hữu ích hơn và hiệu suất trở nên tốt hơn. Tuy nhiên, quá lớn giá trị kích thước kích thước sẽ dẫn tới tiếng ồn và thông tin dư thừa thậm chí dẫn đến kết quả tồi tệ hơn.

Bước nhảy tìm kiếm lớn nhất. Hiệu quả của hiệu suất phân loại bước nhảy tìm kiếm lớn nhất được hiển thị trong Fig. 4(b). Khi các bước nhảy tìm kiếm lớn nhất là nhỏ, thông tin hữu ích hơn có thể được khai thác và hiệu suất cũng trở nên tốt hơn.Tuy nhiên, khoảng cách giữa hai nút càng xa và các thuộc tính của chúng càng ít giống nhau. Trên thực tế, AANI có thể giảm bớt tác động của thuộc tính thông qua việc cân khớp lệnh k-th làm mượt một cách thích ứng. Tuy nhiên, khi các bước nhảy tìm kiếm lớn nhất tiếp tục phát triển, thông tin thuộc tính vô dụng hơn của nốt bên cạnh sẽ bị khai thác dẫn đến kết quả tồi tệ hơn. Hơn nữa, quá giàu giá trị của các bước tìm kiếm lớn nhất sẽ dẫn đến sự phức tạp về gia tăng thời gian.

# **KẾT LUẬN**

Trong bài viết này, chúng ta đề xuất một phương pháp nhúng có tên là AANI. AANI cải thiện đáng kể perfor-mance của các đại diện nút thông qua việc khai thác thông tin của nút hiện tại và khu vực lân cận của nó một cách trơn tru.Ngoài ra, chúng ta giới thiệu cơ chế chú ý để gán trọng lượng thích ứng cho thông tin được làm mịn theo tầm quan trọng của chúng đối với đại diện nút. Cuối cùng, các bước đi ngẫu nhiên bị cắt ngắn trên mạng hoạt động tốt hơn và tối đa hóa xác suất xuất hiện của nút hiện tại và các nút ngữ cảnh của nó. Các thí nghiệm sâu rộng đã chứng minh rằng AANI ra ngoài thực hiện các phương pháp nhúng hiện đại. Đối với công việc truyền tải, chúng ta có kế hoạch mở rộng mô hình được đề xuất của chúng ta để thực hiện mạng lưới nhúng vào các mạng được gán động.

1. *Perceptron là một thuật toán Classification cho trường hợp đơn giản nhất: chỉ có hai class (lớp)* (bài toán với chỉ hai class được gọi là binary classification)*, là nền tảng cho một mảng lớn quan trọng của Machine Learning là Neural Networks và sau này là Deep Learning.* [↑](#footnote-ref-2)
2. Hàm sigmod là một hàm kích hoạt - những hàm phi tuyến được áp dụng vào đầu ra của các nơ-ron trong tầng ẩn của một mô hình mạng, và được sử dụng làm input data cho tầng tiếp theo. [↑](#footnote-ref-3)
3. Hệ số điều chỉnh: là hệ số ước tính bị giảm về 0 trong hồi qui regularization, tránh tình trạng overfitting. [↑](#footnote-ref-4)