du

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**-----🙞🙜🕮🙞🙜-----**



Bộ môn: **Một số thuật toán thông minh**

GVHD: **Nguyễn Công Hoan**

Sinh viên thực hiện:

**Vi Chí Thiện – 16521169**

**Nguyễn Duy Cương – 16520147**

**Trần Thị Kim The – 16521152**

**Trần Xuân Bắc – 16520069**

**TOPIC DETECTION MODEL**

Hồ Chí Minh, ngày 8 tháng 3 năm 2019

Mục lục

[*LỜI CẢM ƠN* 4](#_Toc11011633)

[**I** **Mô hình phát hiện chủ đề** 5](#_Toc11011634)

[**1.** **Word embeddings** 5](#_Toc11011635)

[1.1 Word embeddings là gì? 5](#_Toc11011636)

[1.2 One-hot Encoding 5](#_Toc11011637)

[1.3 Co-occurrence matrix 6](#_Toc11011638)

[1.4 Word embeddings 7](#_Toc11011639)

[1.5 Skip-gram Model 9](#_Toc11011640)

[1.6 CBOW 10](#_Toc11011641)

[1.7 FastText 11](#_Toc11011642)

[1.8 FastTextvới Word representation 12](#_Toc11011643)

[**2.** **Topic modelling** 15](#_Toc11011644)

[2.1 Mô hình phát hiện chủ đề là gì? 15](#_Toc11011645)

[2.2 Các loại phát hiện chủ đề 15](#_Toc11011646)

[2.3 Latent Dirichlet Allocation (LDA) 15](#_Toc11011647)

[**II** **Machine Learning** 15](#_Toc11011648)

[**1.** **Máy học** 15](#_Toc11011649)

[1.1 Machine Learning 15](#_Toc11011650)

[1.2 Machine Learning hoạt động như thế nào? 16](#_Toc11011651)

[1.3 Các vấn đề về trong việc sử dụng Machine Learning 16](#_Toc11011652)

[**III** **Xây dựng mô hình phân loại tin tức đơn giản** 17](#_Toc11011653)

[**1.** **Tiếp cận vấn đề** 17](#_Toc11011654)

[**2.** **Nguồn dữ liệu** 17](#_Toc11011655)

[**3.** **Xây dựng một Web Crawler bằng Scrapy** 17](#_Toc11011656)

[3.1 Scrapy là gì? 17](#_Toc11011657)

[3.2 Tạo Project 17](#_Toc11011658)

[*3.3* *Lấy thông tin động bằng Crawler* 18](#_Toc11011659)

[**4.** **Xử lý dữ liệu** 18](#_Toc11011660)

[4.1 Dữ liệu thu được 18](#_Toc11011661)

[4.2 Xử lý 18](#_Toc11011662)

[**5.** **Biểu diễn từ với word2vec** 20](#_Toc11011663)

[**6.** **Biểu diễn từ với FastText** 21](#_Toc11011664)

[**IV** **Đánh giá mô hình** 22](#_Toc11011665)

[**1.** **Kết quả đạt được** 22](#_Toc11011666)

[**2.** **Nhận xét** 22](#_Toc11011667)

# *LỜI CẢM ƠN*

Để hoàn thành đồ án này, nhóm thực hiện đề tài xin chân thành cảm ơn quý thầy, cô trong khoa Công nghệ Phần mềm, Trường Đại học Công nghệ Thông tin đã quan tâm, giúp đỡ và chỉ đạo nhiệt tình.

Đặc biệt nhóm thực hiện đề tài xin gửi lời cảm ơn chân thành nhất tới thầy Nguyễn Công Hoan. Thầy là người đã tận tình chỉ bảo, chia sẻ cho nhóm những kiến thức bổ ích hướng dẫn động viên trong suốt thời gian qua, tạo mọi điều kiện tốt nhất cho nhóm trong quá trình thực hiện.

Mặc dù nhóm rất cố gắng trong quá trình làm đề tài song không thể tránh khỏi những thiếu sót, hạn chế. Nhóm rất mong nhận được sự đóng góp chân thành từ các thầy, cô và các bạn để đề tài của nhóm được hoàn thiện hơn.

Nhóm thực hiện đề tài xin chân thành cảm ơn!

# **Mô hình phát hiện chủ đề**

## **Word embeddings**

### Word embeddings là gì?

Tập nhúng từ là tên chung cho một tập hợp các mô hình ngôn ngữ và các phương pháp học đặc trưng trong xử lý ngôn ngữ tự nhiên (NLP), nơi các từ hoặc cụm từ từ vựng được ánh xạ tới vector số thực. Về mặt khái niệm, nó liên quan đến việc nhúng toán học từ một không gian với một chiều cho mỗi từ vào một không gian vector liên tục với kích thước thấp hơn nhiều.

Định nghĩa thì rườm rà nhưng ngắn gọn lại word embeddings (tập nhúng từ) là phương pháp ánh xạ mỗi từ vào một không gian số thực nhiều chiều nhưng có kích thước nhỏ hơn nhiều so với kích thước từ điển. Sau đây chúng ta cùng duyệt qua các phương pháp biểu diễn phổ biến trước thời word embeddings:

### One-hot Encoding

Đây là cách đơn giản để biểu diễn ngôn ngữ sang dạng vector với số chiều là kích thước từ điển. Giống như tên của nó, chỉ ở chiều mà vị trí một từ xuất hiện trong từ điển có giá trị là 1. Các chiều khác đều có giá trị là 0.

Ví dụ đơn giản khi tập dữ liệu của ta có 3 câu: Câu 1: tôi đang đi tìm một\_nửa của mình.

Câu 2: tôi đã ăn một\_nửa quả táo

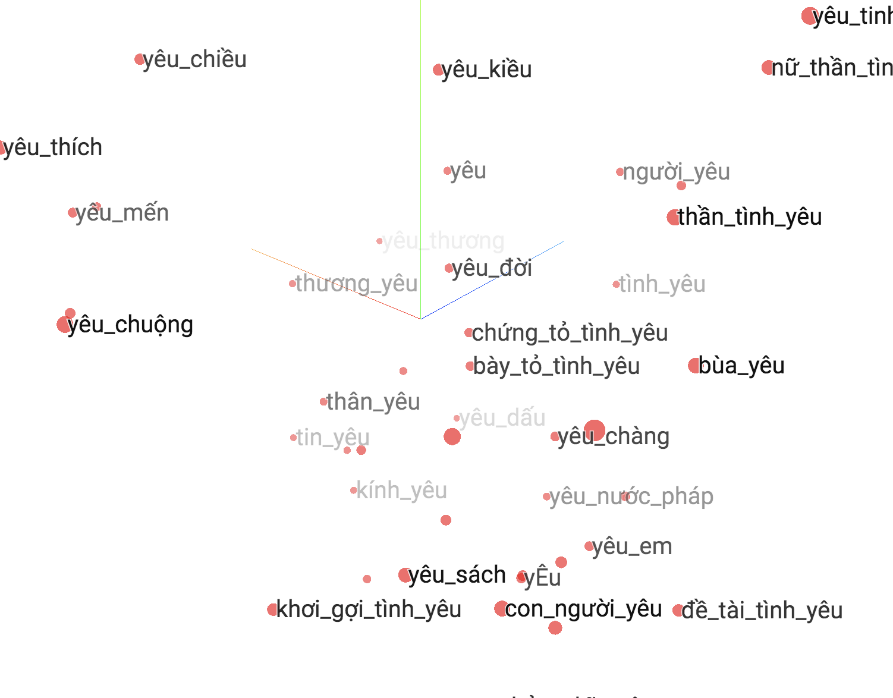
Câu 3: tôi đã đi tìm một\_nửa quả táo

Như vậy từ điển V = {tôi1, đang2, đi3, tìm4, một−nửa5, của6, mình7, đã8, ăn9, quả10, táo11} có kích thước S=11. Biểu diễn theo one-hot thì:

tôi = [1 0 0 0 0 0 0 0 0 0 0]   
đang = [0 1 0 0 0 0 0 0 0 0 0]   
…   
mình = [0 0 0 0 0 0 1 0 0 0 0]   
táo = [0 0 0 0 0 0 0 0 0 0 1]

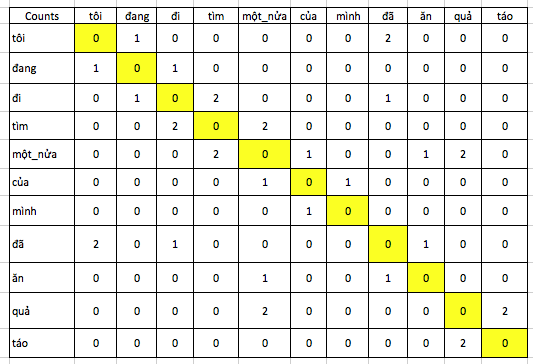
### Co-occurrence matrix

Từ năm 1957, nhà ngôn ngữ học J.R. Firth [1] đã định nghĩa “You shall know a word by the company it keeps” (tạm dịch: bạn sẽ hiểu một từ qua các từ đi cùng với nó). Chẳng hạn nói đến Hà\_Nội, nếu lúc buồn buồn thì bạn nghĩ đến Hà Lội, rồi khói bụi, hay lúc vui bạn nghĩ tới mùa thu Hà Nội. Hoặc nói đến yêu bạn nghĩ đến người yêu, yêu thích. Ví dụ như hình dưới biểu diễn mối tương quan của từ yêu và các đồng xuất hiện gần nhất của nó:



Hình 1: Biểu diễn từ **yêu** bằng các từ lân cận nó. Nguồn hình ảnh: Xuan-Son Vu.

Ma trận đồng xuất hiện: được đề xuất ở hai mức là mức document (văn bản) và mức windows (cửa sổ từ). Mức văn bản cho thông tin chung về các chủ đề hướng tới các phương pháp LSA (latent semantic analysis). Mức cửa sổ từ cho thông tin về cả chức năng cú pháp của từ và ngữ nghĩa.



Hình 2: Ma trận đồng xuất hiện với 2 từ liền kề nhau.

Vấn đề của phương pháp: phương pháp này tuy lưu trữ được thông tin nhiều hơn one-hot encoding nhưng vẫn tồn tại ít nhất 2 vấn đề lớn:

* Chiều của vector tăng theo kích thước từ điển.
* Cần không gian nhớ lớn để lưu thông tin.
* Các mô hình phân loại sau đó dựa trên cách biểu diễn này sẽ gặp phải vấn đề biểu diễn thưa (sparsity issues).

### Word embeddings

Do một đống vấn đề của ma trận đồng xuất hiện mà đã có nhiều nghiên cứu hướng theo giải pháp học biểu diễn ở số chiều thấp hơn. Trước thời word2vec đã có các nghiên cứu như bài của nhóm bác Bengio năm 2003 [3]. Nhưng những giải pháp này vẫn gặp vđề về chi phí tính toán. Cho đến năm 2013, nhóm Mikolov giúp giới NLP thở phào với giải pháp mới mang tên word2Vec. Từ thời điểm này hàng loạt bài toán NLP được giải quyết với độ chính xác cao hơn nhiều so với trước.

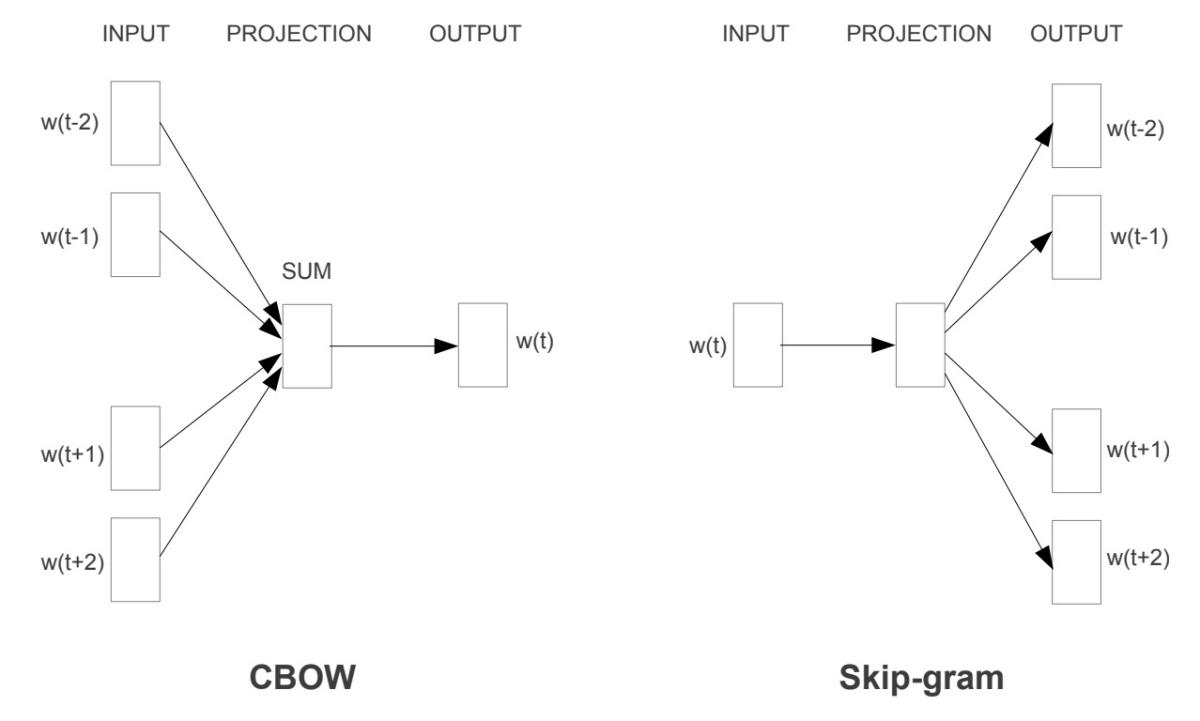
Ý tưởng chính của word2vec là:

* Thay vì lưu thông tin xuất hiện của các từ bằng cách đếm trực tiếp như ma trận đồng xuất hiện, word2vec học để đoán từ lân cận của tất cả các từ.
* Các giải pháp sau đó như Glove cũng tương tự word2vec được đề xuất bởi nhóm Pennington năm 2014.
* Tính toán nhanh hơn và dễ dàng thêm dữ liệu mới vào trong mô hình

Phương pháp:

* Đoán các từ lân cận trong cửa số m của mỗi từ 1.1. Với mỗi từ t = 1 … T 1.2. Đoán các từ trong cửa sổ bán kính m của tất cả các từ
* Hàm mục tiêu (objective function): tối ưu hợp lý hoá cực đại của bất kỳ **từ** ngữ cảnh (context word) đối với một **từ** đang xét hiện tại (center word):

J(θ)=−1T∏Tt=1∏−m≤j≤m,j≠0p(wt+j|wt;θ)



Hình 3: hai mô hình cơ bản của word2vec là CBoW và Skip-gram.

Các loại Word2vec:

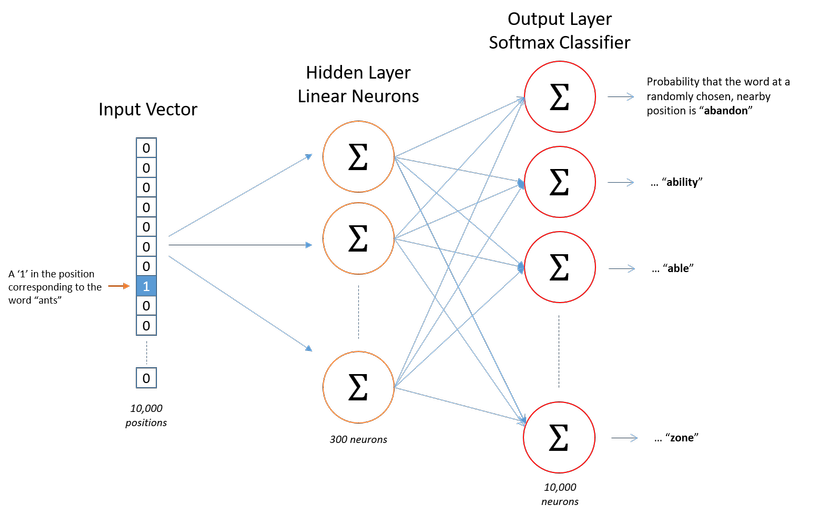
* Cbow:
  + Cho các từ ngữ cảnh
  + Đoán xác suất của một từ đích
* Skip-gram model :
  + Cho từ đích
  + Đoán xác suất của các từ ngữ cảnh

### Skip-gram Model

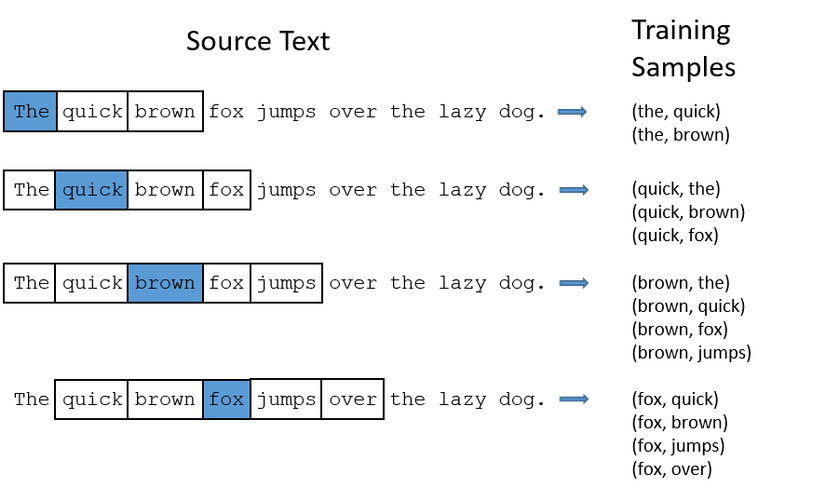
**Mục tiêu:** Học trọng số các lớp ẩn, các trọng số này là các words vector

**Cách thức:** Cho một từ cụ thể ở giữa câu(input word), nhìn vào những từ ở gần và chọn ngẫu nhiên. Mạng neural sẽ cho chúng ta biết xác suất của mỗi từ trong từ vựng về việc trở thành từ gần đó mà chúng ta chọn.

Dưới đây là mô hình kiến trúc của mạng Skip-gram và cách xây dựng training data.



**Ví dụ:** Xây dựng training data với windows size = 2. Ở đây windows được hiểu như một cửa sổ trượt qua mỗi từ. Windows size = 2 tức là lấy 2 từ bên trái và bên phải mỗi từ trung tâm.



Chi tiết về model:

* Xây dựng bộ từ vựng
* Biểu diễn mỗi từ thành các one-hot-vector
* Đầu ra là một vector duy nhất, có kích thước bằng kích thước của bộ từ vựng, thể hiện xác suất của mỗi từ được là lân cận của từ đầu vào.
* Không có hàm kích hoạt trên tầng ẩn
* Hàm kích hoạt trên tầng output là softmax
* Trong quá trình training, input là 1 one-hotvector, ouput cũng là 1 one-hot-vector
* Trong quá trình đánh giá sau khi training, đầu ra phải là 1 phân bố xác suất.

### CBOW

Cách thức hoạt động của CBOW là nó có xu hướng dự đoán xác suất của một từ được đưa ra theo ngữ cảnh. Một bối cảnh có thể là một từ đơn hoặc một nhóm từ. Nhưng để đơn giản, tôi sẽ lấy một từ ngữ cảnh duy nhất và cố gắng dự đoán một từ mục tiêu duy nhất.

**Ưu điểm của CBOW:**

* **Có tính xác suất là bản chất, nó được cho là thực hiện vượt trội so với các phương pháp xác định (nói chung).**
* **Đó là bộ nhớ thấp. Nó không cần phải có các yêu cầu RAM lớn như ma trận xuất hiện, nơi nó cần lưu trữ ba ma trận lớn.**

Nhược điểm của CBOW:

* CBOW lấy mức trung bình của ngữ cảnh của một từ (như đã thấy ở trên trong tính toán kích hoạt ẩn). Ví dụ: Apple có thể vừa là trái cây vừa là công ty nhưng CBOW lấy trung bình cả hai bối cảnh và đặt nó ở giữa một cụm cho trái cây và công ty.
* Đào tạo một CBOW từ đầu có thể mất mãi mãi nếu không được tối ưu hóa đúng cách.

### FastText

FastTextlà một thư viện nhẹ, miễn phí, mã nguồn mở cho phép người dùng tìm hiểu các biểu diễn văn bản và phân loại văn bản được tạo bởi phòng thí nghiệm AI Resrarch (FAIR) của Facebook. Nó hoạt động trên phần cứng tiêu chuẩn, chung chung. Các mô hình sau đó có thể được giảm kích thước để thậm chí phù hợp với các thiết bị di động.

Mô hình cho phép tạo ra một thuật toán học tập không giám sát hoặc học có giám sát để có được các biểu diễn vector cho các từ. Facebook cung cấp các mô hình được sàng lọc sẵn cho 294 ngôn ngữ. FastTextsử dụng mạng thần kinh để nhúng từ.

FastTextcó 2 chức năng chính:

* Text classification: quá trình gán thẻ hoặc danh mục cho văn bản theo nội dung của nó
* Word representation: biểu diễn từ dưới dạng các vector đặc trưng

Điểm mạnh của FastTextso với các công cụ tương tự trong việc tạo ra các word embedding là FastTextcho phép sử dụng subword information. Theo đó, mỗi một word sẽ được biểu diễn bằng vector đã được học, nhưng vector này lại là tổng các các vector con, mà mỗi vector con đó là biểu diễn của các subword của từ đó.

Điểm đặc biệt của FastTextso với các word embedding khác là FastTextcho phép sử dụng subword information. Theo đó, mỗi một word sẽ được biểu diễn bằng vector đã được học, nhưng vector này lại là tổng các các vector con, mà mỗi vector con đó là biểu diễn của các subword của từ đó.

### FastTextvới Word representation

Làm việc tương tự word2vec những huấn luyện ở mức ký tự (**character**) thay vì từ (**word**)

Ví dụ: Từ **where** và với n-grams size = 3:

<wh, whe, her, ere, re>, <where>

Ưu điểm:

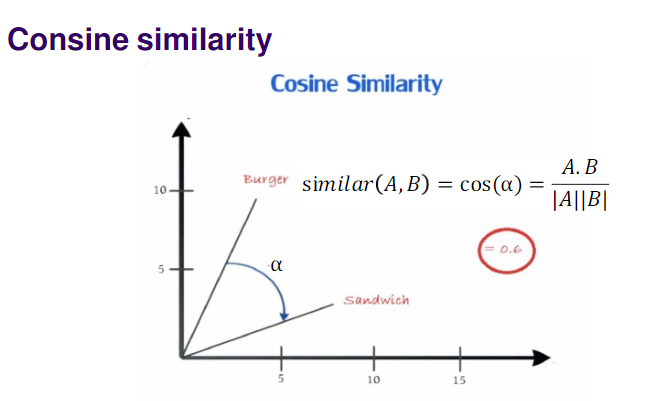
Có thể tạo ra vector **“rất gần”** với vector của từ đúng dù nó có sai một chút (vd: **behave** sẽ gần với **behave**).

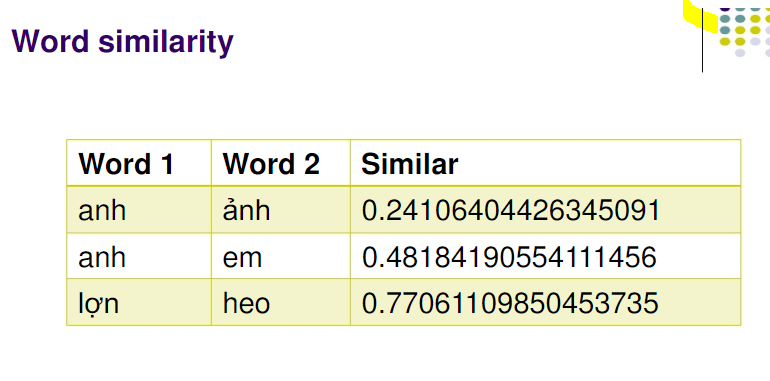
Có hiệu quả trong bài toán word similarity cho dù tập huấn luyện có **“sai chính tả”**

Có thể tạo ra vector của từ chưa biết

Phương pháp tiếp cận:

* Sử dụng FastText để biểu diễn từ sang vector
* Sử dụng bài toán word similarity để phát triển từ điển





Cách hoạt động:

Giả sử ta đang muốn tính word embedding của từ “nature”. Từ này có độ dài là 6. Khi xử lý, có 2 kí tự prefix và subfix được thêm vào, nên nature sẽ thành và có độ dài 8.

FastTextsẽ xem vector biểu diễn của từ nature bằng tổng các vector con của nó có độ dài từ 3 đến 6 (tham số này là mặc định và có thể thay đổi bằng các quy định lại tham số). Như vậy vector biểu diễn của nature sẽ bằng tổng vector biểu diễn của:

* Nhóm subword có độ dài 3: <na, nat, atu, tur, ure, re> và <nature>
* Nhóm subword có độ dài 4: <nat, natu, atur, ture, ure>
* Nhóm subword có độ dài 5: <natu, natur, ature, ture>
* Nhóm subword có độ dài 6: <natur, nature, ature> Và bản thân từ đó.

Tại sao chọn Fasttext:

* Thư viện nguồn mở, miễn phí từ Facebook AI Research công bố 08/2016.
* FastTextcó tốc độ training nhanh và cho độ chính xác cao.
* FastTextcó thể training với khối lượng dữ liệu lớn.

## **Topic modelling**

### Mô hình phát hiện chủ đề là gì?

Mô hình chủ đề là một loại mô hình thống kê để khám phá các "chủ đề" trừu tượng xảy ra trong một bộ sưu tập tài liệu và là một công cụ khai thác văn bản được sử dụng thường xuyên để khám phá các cấu trúc ngữ nghĩa ẩn trong thân văn bản. LDA là một ví dụ về mô hình chủ đề.

### Các loại phát hiện chủ đề

Phân tích ngữ nghĩa rõ ràng .

Phân tích ngữ nghĩa tiềm ẩn.

Phân bổ Dirichlet tiềm ẩn.

Quy trình Dirichlet phân cấp Hệ số ma trận không âm.

### Latent Dirichlet Allocation (LDA)

Ý tưởng của LDA dựa trên nguyên lý mỗi topic là phân bố của các từ, mỗi văn bản là sự trộn lẫn giữa nhiều topic, và mỗi từ phân bố vào một trong những topic này.

Mô hình giải quyết 2 vấn đề:

* Vấn đề “Sinh văn bản” (Bài toán thuận): Khi tạo lập thông điệp, người tạo lập (người viết văn bản) xác định trước chủ đề, sau đó xây dựng văn bản bằng cách chọn các từ xoay quanh chủ đề đã xác định
* Vấn đề “Khám phá chủ đề” (Bài toán ngược): Có văn bản, cần tìm các chủ đề mà người viết đã dựa trên đó để hình thành văn bản. Nghĩa là cần khám phá chủ đề tiềm ẩn trong nội dung thông điệp được người dùng trao đổi.

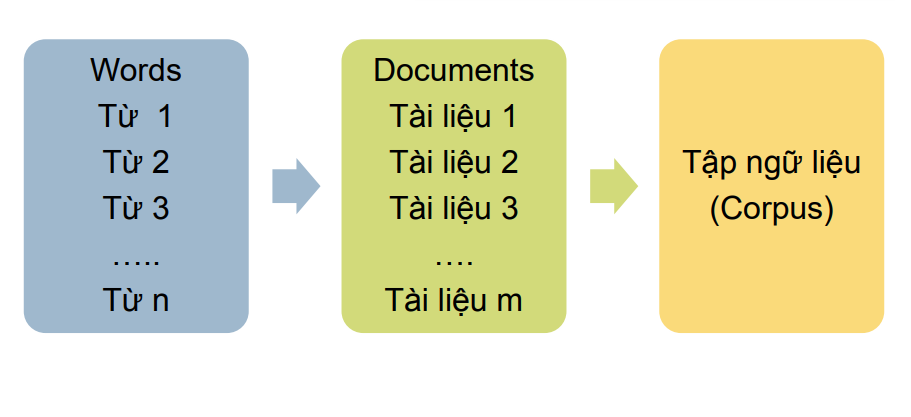
Cách chia bayes:

Mô hình xác suất theo mạng Bayes 3 cấp: tài liệu, chủ đề và từ

* Mỗi tài liệu (document) được mô tả dưới dạng kết hợp ngẫu nhiên của một tập các chủ đề.
* Mỗi chủ đề (topic) là một phân bố rời rạc của một tập các từ vựng (words).

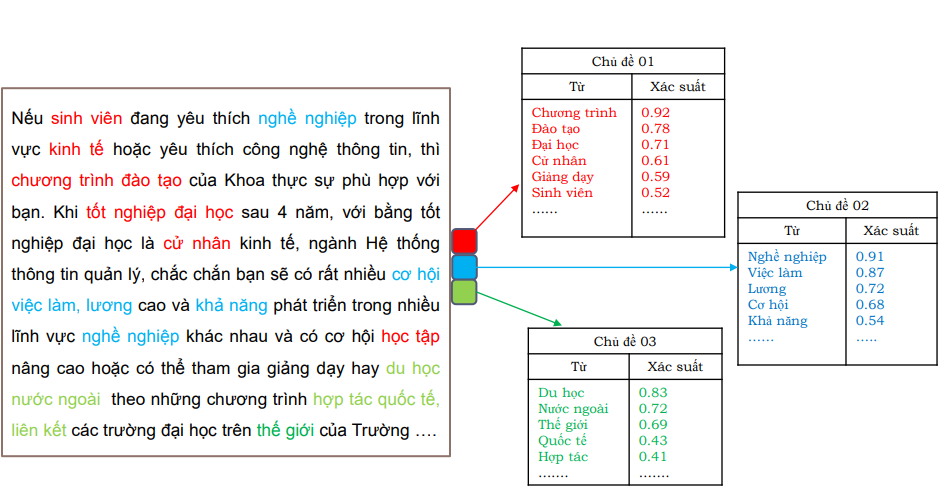
Mô hình sinh tài liệu (bài toán thuận)

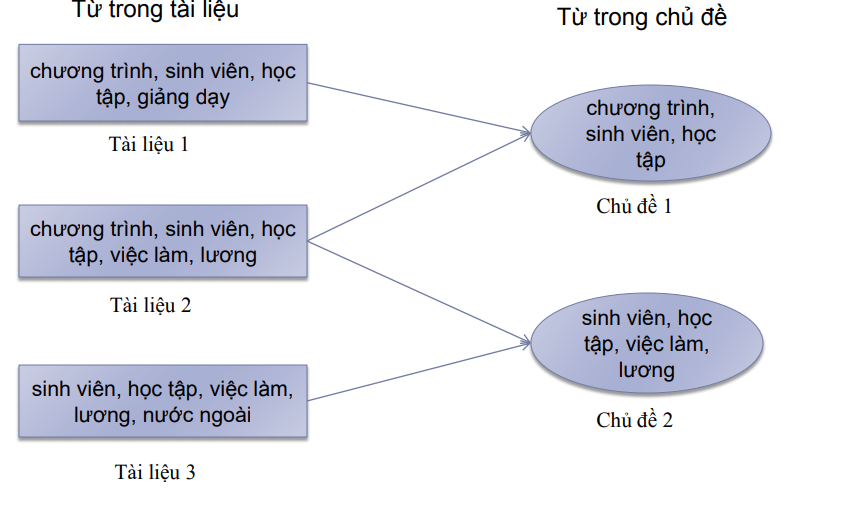
* Dựa trên việc rút trích tập từ đặc trưng để sinh tài liệu

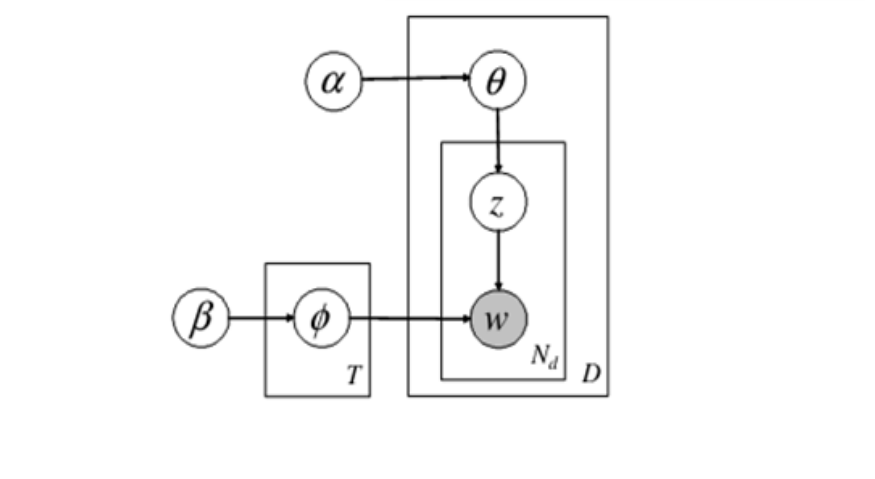


Mô hình chủ đề - LDA (khám phá chủ đề) (bài toán ngược)

* Khám phá chủ đề tiềm ẩn







# **Machine Learning**

## **Máy học**

### Machine Learning

Học máy (Machine Learning - ML) là một mảng thuộc AI và được coi là một bước nâng cấp của AI. Trước khi tìm hiểu ML là gì, ta nghĩ một chút xem thế nào là Học (Learning)? Học là một quá trình bao gồm: ghi nhớ (remembering), áp dụng (adapting), và khái quát (generalising).

Lấy ví dụ như việc ta học Toán ở cấp 3, bắt đầu bằng việc ghi nhớ các công thức, bài tập mẫu, sau đó tự giải lại các bài tập mẫu này, và cuối cùng là dùng kiến thức đó để giải các bài tập mới.

Việc học của máy tính cũng tương tự, mục tiêu của ML là muốn cho máy tính không chỉ ghi nhớ và thực hiện các lệnh có sẵn, mà còn có thể khái quát vấn đề, từ đó giải được các bài toán mới không được lập trình sẵn.

Vậy máy tính học từ đâu? Học từ data.

Vì thế có thể nói ML là kỹ thuật thuộc ngành Khoa học dữ liệu (Data Science). Quay trở lại ví dụ về con bot trong trò chơi điện tử ở trên. Nếu con bot được cài đặt sử dụng ML, giả sử rằng ban đầu ta chiến thắng con bot này dễ dàng, nhưng sau nhiều lần chơi, con bot học được cách chơi từ ta, từ đó nó phát kiến ra những cách chơi mới để thắng ta, và ta không thể thắng nó được nữa. Chưa hết, con bot này còn có thể đem cách chơi học được để thắng ta để đấu với các người chơi mới. Đó là một ví dụ của việc khái quát.

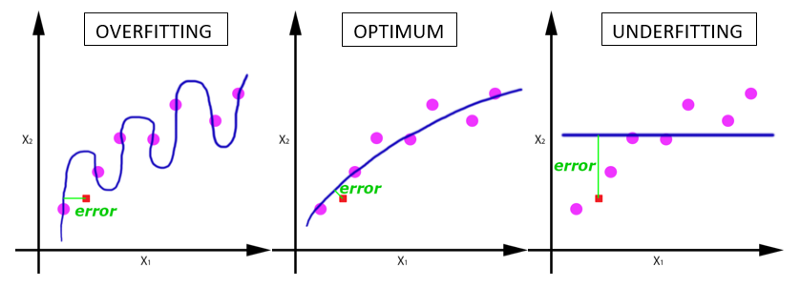
### Machine Learning hoạt động như thế nào?

Thực hiện Machine Learning bao gồm các bước như sau:

* Thu thập và chuẩn bị dữ liệu
* Chọn Feature
* Chuẩn hoá dữ liệu
* Chọn thuật toán
* Chọn parameter cho thuật toán
* Huấn luyện và Đánh giá

### Các vấn đề về trong việc sử dụng Machine Learning

Đi chung với các thuật toán về Machine Learning là các vấn đề mà ta cần lưu ý khi bắt đầu thực hành trên dữ liệu và xây dựng các mô hình dự đoán.

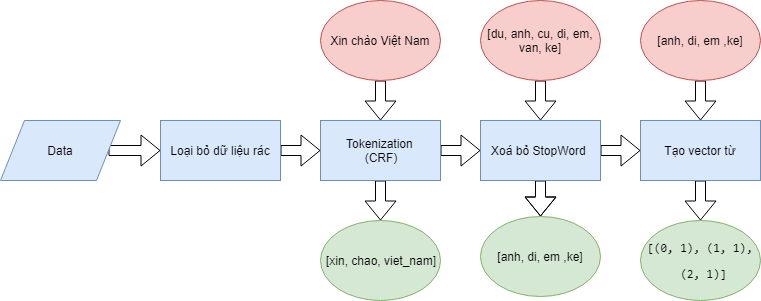


Overfitting: Là hiện tượng thuật toán cố gắng để thích ứng với từng điểm trong cơ sở dữ liệu để đưa ra kết quả chính xác nhất. Nhưng điều đó đồng nghĩa với việc các dự đoán với các dữ liệu mới sẽ bị sai lệch. Dựa vào hình ta thấy, đường màu xanh chính là mô hình của chúng ta, và mô hình này cố gắng đi qua hết tất cả các điểm trong cơ sở dữ liệu. Nhưng khi đưa vào các điểm dữ liệu mới, mô hình này sẽ cho ra kết quả rất xa so với thực tế do biên độ đường đi rất rộng

Underfitting: Ngược lại với overfitting, underfitting xảy ra do mô hình dự đoán quá đơn giản, làm cho biên độ khác biệt giữa thông số dự đoán và thông số thực thế không tối ưu nhất.

Lựa chọn cost function: Việc lựa chọn cost function ảnh hướng rất nhiều đến độ chính xác của mô hình dự đoán. Với các dữ liệu có biên độ lệch cao, khi chọn cost function là Mean Squared Error sẽ cho ra kết quả rất lớn, ảnh hướng đến quá trình backpropagation. Thay vào đó, Mean Absolute Error sẽ cho ra kết quả “dễ chịu” hơn, và có thể cho ra mô hình tốt hơn.

# **Xây dựng mô hình phân loại tin tức đơn giản**



## **Tiếp cận vấn đề**

Vấn đề tin tức hiện nay rất nhiều, với nhiều nội dung phong phú. Bên cạnh đó những tin giả mạo, tin lá cải cũng nhiều. Đó là một trong những nguyên nhân thách thứ với phân loại và phát hiện chủ đề tin tức. Trong phạm vi tìm hiểu của đề tài này, mô hình phân loại chủ đề tin tức chỉ nhắm tới xếp loại một tin tức nào đó vô những chủ đề có sẵn. Tự động đề xuất cho người sử dụng những chủ đề liên quan tới tin tức đó.

## **Nguồn dữ liệu**

Nguồn tin tức trên internet khá phong phú và không được xác thực. Một số trang báo đưa tin giật gân khiến cho bộ dữ liệu bị sai lệch. Nên nhóm quyết định lấy một số trang uy tín và có có vốn từ chuẩn để chuẩn bị cho máy học:

Nguồn thông tin được tổng hợp từ một trang tổng hợp báo trí do anh Vũ Hải Lộc tạo ra: <https://www.theodoibaochi.com/> và <https://vi.wikipedia.org/wiki>

Là nơi hợp các nguồn báo có uy tín và phân tích chủ đề, rút trích chủ đề có sẵn.

## **Xây dựng một Web Crawler bằng Scrapy**

### Scrapy là gì?

Scrapy là một khung công cụ thu thập dữ liệu web miễn phí và mã nguồn mở, được viết bằng Python. Ban đầu được thiết kế để cào web, nó cũng có thể được sử dụng để trích xuất dữ liệu bằng cách sử dụng API hoặc như một trình thu thập thông tin web có mục đích chung. Nó hiện đang được duy trì bởi Scrapinghub Ltd., một công ty phát triển và dịch vụ cào web.https

### Tạo Project

Scrapy giới thiệu ý tưởng về một dự án có nhiều trình thu thập thông tin hoặc trình thu thập thông tin trong một dự án duy nhất. Khái niệm này hữu ích đặc biệt nếu bạn đang viết nhiều trình thu thập dữ liệu của các phần khác nhau của trang web hoặc tên miền phụ của trang web. Vì vậy, trước tiên hãy tạo dự án

### *Lấy thông tin động bằng Crawler*

Bước đầu tiên chúng ta sẽ lấy tất cả những link từ vnexpress:



Từ những link ta lấy được từ hàm get\_urls\_new rồi ta lấy link:



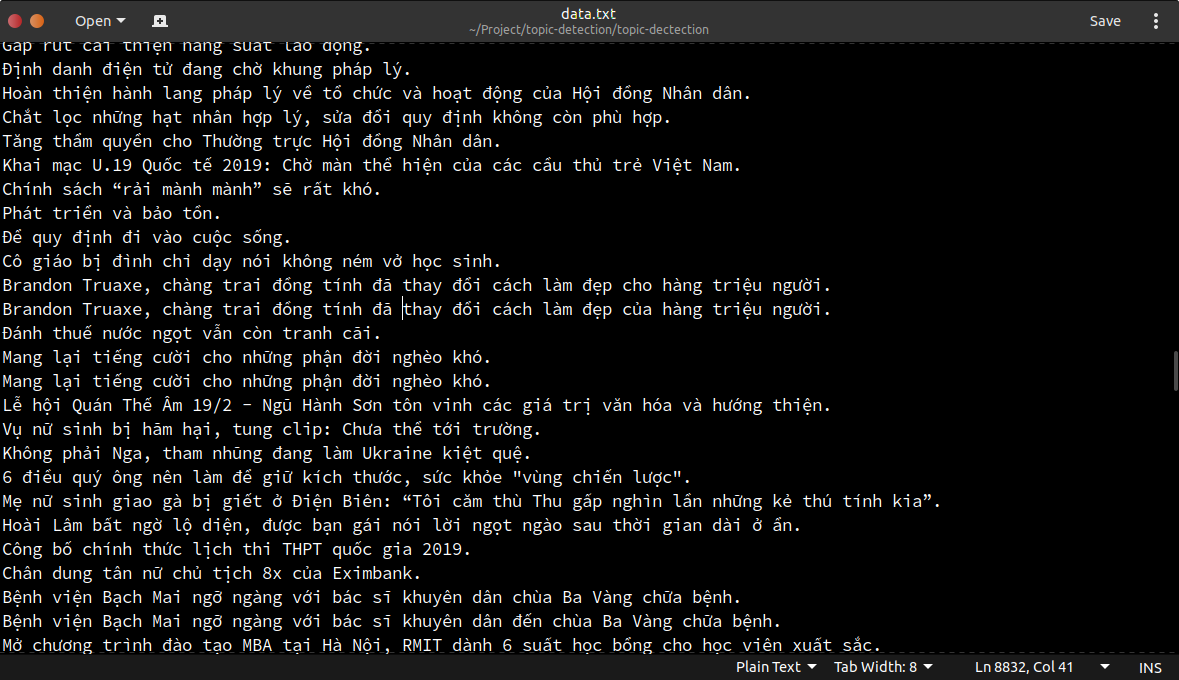
## **Xử lý dữ liệu**

### Dữ liệu thu được

Danh sách dứ liệu bao gồm:

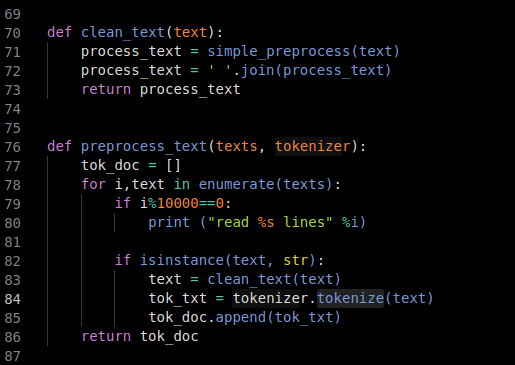
|  |  |  |
| --- | --- | --- |
| Topic | Số lượng | Ngày |
| Giáo dục | 7483 | 1/1/2019 - 21/5/2019 |
| Pháp luật | 9797 | 1/1/2019 - 21/5/2019 |
| Tâm sự | 8471 | 1/1/2019 - 21/5/2019 |
| Thể thao | 5499 | 1/1/2019 - 21/5/2019 |
| Thời sự | 6377 | 1/1/2019 - 21/5/2019 |

Dữ liệu ta thu được bao gồm khoảng 500 nghìn dòng, với khoảng 500mb. Với chủ yếu là văn bản, đoạn văn. Có bao gồm cả số chữ và các link.



### Xử lý

Dữ liệu thu được là những văn bản chưa được xử lý, với khá nhiều dấu chấm câu. Thêm vào đó là những từ vô nghĩa. Những từ viết tắt, sai chính tả. Một số kí hiệu đặc biệt cũng xuất hiện trong dữ liệu thu được. Kết luận dữ liệu có độ nhiễu khá cao. Do đó ta phải xử lý dữ liệu đầu vào loại bỏ các dấu và tách các từ:



Đoạn code clean\_text dùng để xoá những kí tự thừa, các kí tự không cần thiết. Để mang lại một câu. Tiếp theo đó tokenizer.tokenize(text) dùng để chia câu thành những từ có nghĩa.



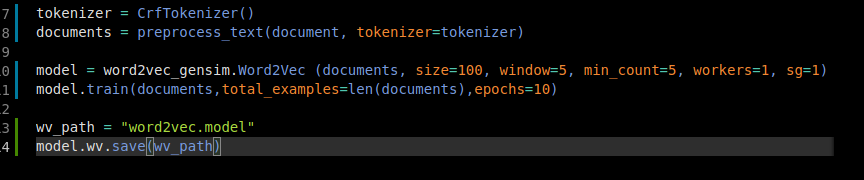
Đầu vào của hàm này là một câu và đầu ra là những từ có nghĩa:

Ví dụ câu:

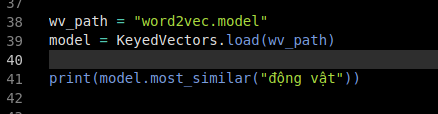
chó hay chó nhà là danh pháp khoa học.

Đầu ra sẽ là: [['chó', 'hay', 'chó', 'nhà', là', 'danh\_pháp', 'khoa\_học' ]]

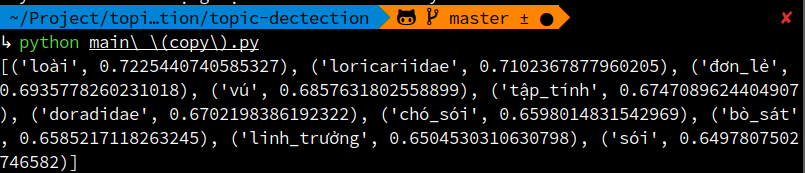
## **Biểu diễn từ với word2vec**



Sau khi ta chuẩn bị dữ liệu xong, ta tiến hành training cho tập dữ liệu vừa tìm được. Với một vector từ có 100 chiều, ta training trong khoảng 20 đến 30ph tuỳ vào số lượng từ và sức mạnh của phần cứng. Sau khi training xong ta thử độ chính xác bằng cách liệt kê ra những từ mang nghĩa giống với nó:



Đây là kết quả sau khi tra cứu một từ gần với động vật nhất:



Ta thấy **động vật** và **loài** có độ tương đồng về nghĩa là 0.722.

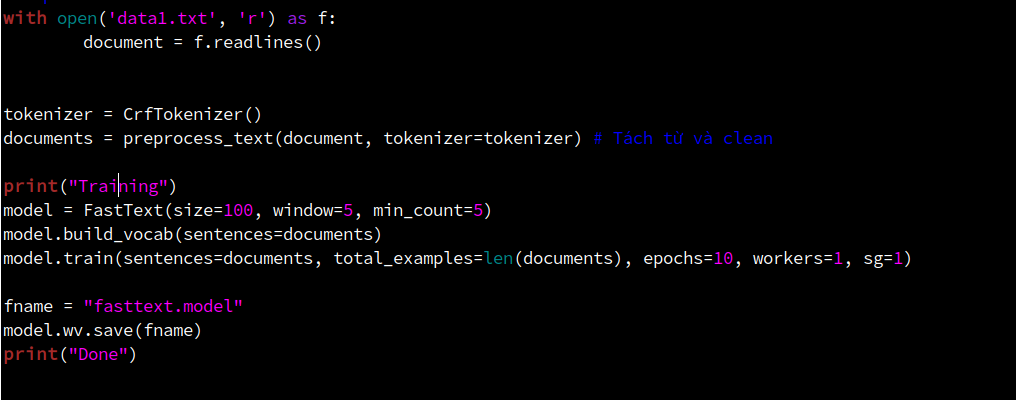
Tiếp tục ta thử so sánh **cổ động viên** với **khán giả**:

print(model.similarity("cổ\_động\_viên", "khán\_giả"))

Ta nhận được kết quả: 0.46766767

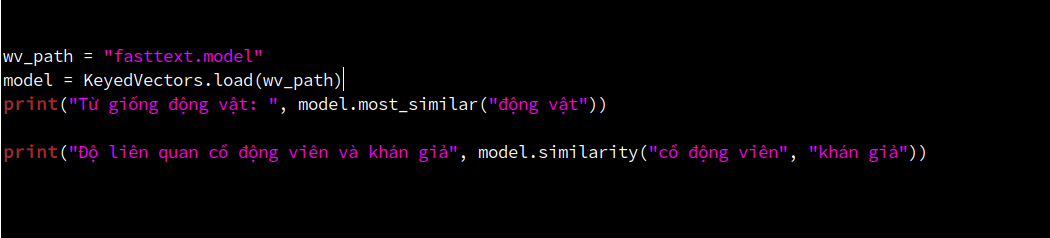
Nhược điểm của word2vec trên là cho độ chính xác khá thấp, cần số lượng từ lớn và không thể tính toán được từ không có trong từ điển. Đó là nhược điểm lớn nhất của word2vec.

## **Biểu diễn từ với FastText**

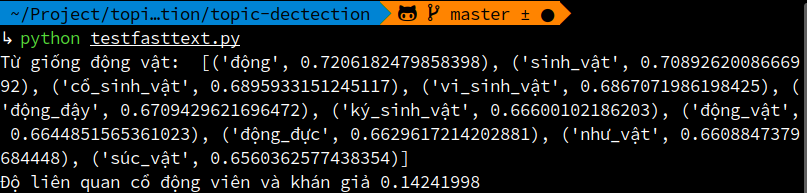


Trước tiên ta nạp tệp văn bản vào và làm sạch từ, tiền sử lý dữ liệu như làm với word2vec. Tiếp theo training dữ liệu với FastTextvới số chiều là 100, độ dài 1 từ là 5. sau khi training xong ta sẽ lưu lại thành ta sẽ được một tệp model.

Tiếp theo đó ta nạp model lên rồi thực hiện kiểm tra độ chính xác:

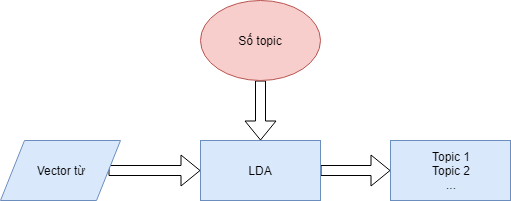


Kết quả ta nhận được kế quả:



Ta thấy việc so sánh giữa **cổ động viên** và **khán giả:** 0.1424. Kết quả khả quan hơn vì hai từ này hầu như không có tương đồng gì về nghĩa.

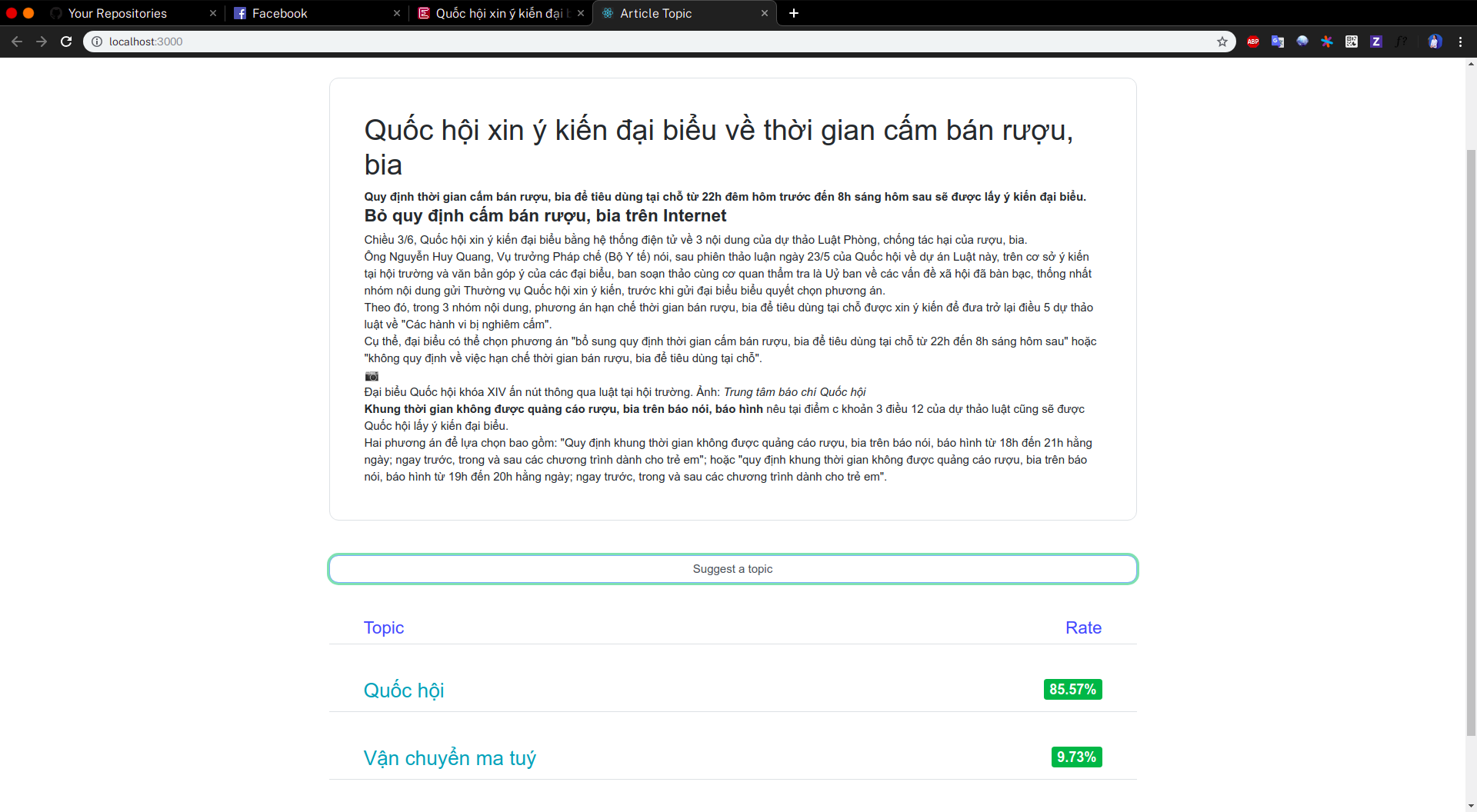
## **Training model**



Từ vector từ ở trên, sử dụng thuật toán lda được giới thiệu ở trên. Ta phân loại ra được những đặc trưng của mỗi loại chủ đề.

# **Đánh giá mô hình**

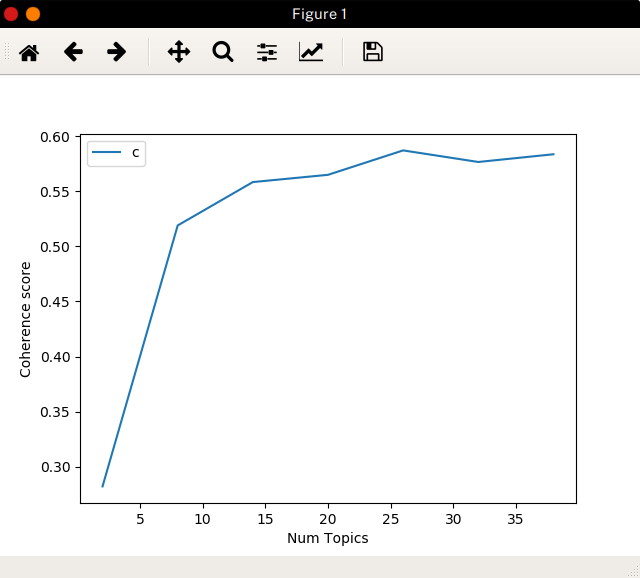
## **Kết quả đạt được**



Thử model vừa training với một bài báo từ vnexpress. Ta nhận được kết quả khả quan, kết quả đúng với chủ đề của bài báo

## **Nhận xét**

Với số lượng topic cho trước, ta có được một con số để đánh giá độ gắn kết chủ đề coherence:



Khi tăng số topic lên, độ gắn kết càng tăng đến một mức độ nào đó sẽ đạt được cực trị. Đó là số topic tối ưu cho số lượng bài báo mình đã training.

Ta thử độ chính xác so với bài báo mới hoàn toàn:

|  |  |  |  |
| --- | --- | --- | --- |
| Thể loại | Số bài báo | Số bài báo sai mục | Tỉ lệ (đúng) |
| Giáo dục | 50 | 15 | 70% |
| Pháp luật | 60 | 12 | 80% |
| Tâm sự | 50 | 20 | 60% |
| Thể thao | 55 | 14 | 74.5% |
| Thời sự | 66 | 13 | 80% |

Tổng tất cả kết quả lại thì độ chính xác của model khoảng **73%.**