**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH**

**KHOA ĐÀO TẠO CHẤT LƯỢNG CAO**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

◈◇◈◇◈


# TIỂU LUẬN CHUYÊN NGÀNH

## Đề tài:

## TÌM HIỂU BÀI TOÁN PHỐI TRANG PHỤC THEO SỰ KIỆN

**GVHD: TS. Nguyễn Thiên Bảo**

**SVTH: Phạm Ngọc Minh**

**MSSV: 16110156**

**SVTH: Nguyễn Phi Long**

**MSSV: 16110142**

**TP. Hồ Chí Minh, ngày 7 tháng 12 năm 2019**

# LỜI CẢM ƠN

Với một lĩnh vực phổ biến và đang phát triển một cách mạnh mẽ như AI thì việc tiếp cận và nghiên cứu đòi hỏi trước tiên phải có một nền tảng kiến thức cơ bản. Tiểu Luận Chuyên Ngành là khá quan trọng và là bước đà vững chắc cho việc nghiên cứu học hỏi ở các mức cao hơn. Tuy nhiên, Bài toán phối trang phục theo sự kiện khá mới mẽ và lượng kiến thức tương đối nhiều, việc tiếp thu thật sự khó khăn. Nhóm em xin chân thành cảm ơn thầy Nguyễn Thiên Bảo đã hỗ trợ, cung cấp kiến thức một cách tận tình và dễ hiểu nhất. Giúp nhóm hoàn thành tốt cùng với một lượng kiến thức để nhóm có kĩ năng cho việc nghiên cứu cao hơn trong tương lai như deep learning.

Nhóm em xin cảm ơn!

# Mục lục

# Mục lục hình ảnh

# 1. MỞ ĐẦU

## *1.1 Sự cần thiết của đề tài.*

Thị lực máy tính (Computer vision) là một nhánh trong lĩnh vực trí tuệ nhân tạo (Artificial Intelligence) và Khoa học máy tính (Computer science). Lĩnh vực này giúp máy tính có khả năng thị giác như con người, giúp máy tính có thể nhận diện và hiểu biết một hình ảnh mang tính điện tử.

Thị lực máy tính được áp dụng mạnh mẽ trong các tác vụ nhận diện hình ảnh và cụ thể trong đề tài này là nhận diện và mô tả hình ảnh thời trang.

Mô tả hình ảnh thời trang là bài toán có tính ứng dụng cao trong thực tiễn và đang được nhiều người quan tâm. Trong thời đại công nghệ số đang phát triển mạnh mẽ, dẫn đến việc thương mại điện tử cũng phát triển không ngừng, số lượng mặt hang sản phẩm ngày càng tăng và thay đổi nhanh chóng, điều này dẫn đến việc mô tả cho sản phẩm ngày càng khó khăn, mất nhiều thời gian và chi phí. Giải pháp cho vấn đề trên là áp dụng thị lực máy tính vào tác vụ mô tả ảnh, máy tính sẽ giúp con người tạo ta câu mô tả cho sản phẩm một cách trực quan, đầy đủ ý nghĩa. Giúp giảm thiểu chi phí và thời gian cho con người. Nhưng để máy tính có thể nhìn vào hình ảnh và hiểu rồi sau đó đưa ra câu mô tả bằng ngôn ngữ tự nhiên người là một điều không hề dễ dàng. Nó đòi hỏi các thuật toán phù hợp và nguồn dữ liệu đầu vào phong phú và chính xác, đặc biệt là các nhãn và câu mô tả mẫu.

Thấy được tầm quan trọng của bài toán mô tả hình ảnh thời trang. Nhóm em xin chọn đề tài "Mô tả hình ảnh thời trang ứng dụng Học sâu" để làm tiểu luận chuyên ngành.

## *1.2 Mục đích của đề tài*

Tìm hiểu về bài toán mô tả ảnh thời trang, các kiến thức về Học sâu, Mạng thần kinh nhân tạo (Neural network), các mô hình mạng nơ-ron như mạng nơ-ron tích chập (Convolution Neural Network - CNN), mạng nơ-ron hồi quy (Recurrent Neural Network – RNN).

## *1.3 Cách tiếp cận và phương pháp nghiên cứu*

**Cách tiếp cận đề tài:** Sử dụng các mạng nơ-ron nhân tạo như mạng nơ-ron tích chập, mạng nơ-ron hồi quy, và áp dụng các cơ chế attention (spatial attention, chanel-wise attention) cho hình ảnh thời trang.

**Phương pháp nghiên cứu:** Nghiên cứu tài liệu, các bài báo, bài báo cáo liên quan đến mô tả ảnh, các lý thuyết về học sâu, mạng nơ-ron và cơ chế attention.

# 2. BÀI TOÁN PHỐI TRANG PHỤC SỰ KIỆN

## *2.1 Giới thiệu bài toán*

Phối đồ theo sự kiện là một ứng dụng của bài toán mô tả hình ảnh nói chung. Mô tả nội dung ảnh là một lĩnh vực trong ngành Trí tuệ nhân tao, yêu cầu máy tính có thể hiểu được nội dung của hình ảnh và mô tải lại bằng ngôn ngữ tự nhiên. Mô hình cũng phải hiểu được ngữ cảnh, vị trị và các đối tượng chính của bức ảnh để tạo được câu mô tả hợp lý, đầy đủ ngữ nghĩa. Không những thế, để tạo ra câu mô tả tối ưu, mô hình cần phải có sự hiểu biết về ngữ pháp của ngôn ngữ đích.

Một trong những nhiệm vụ chính quyết định độ hiệu quả của mô hình là việc trích xuất thông tin từ hình ảnh đầu vào ở định dang thô sang đinh dạng mà mô hình có thể học và sử dụng được, các thông tin được trích xuất từ hình ảnh được gọi là *features,* và chúng thường được biểu diễn dưới dạng vector. Để trích xuất dữ liệu ta sẽ sử dụng các kỹ thuật dựa trên Học sâu, các feature được học từ dữ liệu huấn luyện và chúng có thể xử lý được số lượng lớn hình ảnh. Mạng nơ-ron tích chập (CNN) được sử dụng phổ biết trong tác vụ trích xuất và học các feature. Sau đó mô hình sẽ sử dụng mạng nơ-ron hồi quy (RNN) để tạo ra sự lựa chọn phù hợp.

Trong bài viết này, chúng tôi mô tả một giải pháp để giải quyết một loạt thách thức chung trong thương mại điện tử, phát sinh từ thực tế là các sản phẩm mới liên tục được thêm vào danh mục. Những thách thức liên quan đến việc cá nhân hóa trải nghiệm của khách hàng, dự báo nhu cầu và lập kế hoạch phạm vi sản phẩm. Lập luận rằng phần nền tảng để giải quyết tất cả những vấn đề này là có sự nhất quán và thông tin chi tiết về từng sản phẩm, thông tin hiếm khi có sẵn hoặc nhất quán với vô số nhà cung cấp và chủng loại của sản phẩm. Góp phần mô tả chi tiết kiến trúc và phương pháp được triển khai tại, áp dụng trong những nhà phát triển thời trang lớn hoặc nhà bán lẻ thương mại điện tử. Sau đó dựa trên thông tin mà hệ thống học được, sẽ góp phần cho sự chọn lựa trang phục của khách hàng cũng như tăng sự thu hút đối với các hãng thời trang.

## 2.2 Giải pháp cho bài toán

Để đạt được những đặc tính  lạc của tất cả các sản phẩm, Nhóm đã tận dụng các thuộc tính được chú thích thủ công chỉ bao gồm một phần của danh mục sản phẩm như được minh họa trong Hình 1. Như dữ liệu luôn nhất quán, có sẵn cho tất cả các sản phẩm, chúng tôi có một bộ hình ảnh, mô tả văn bản cũng như loại sản phẩm và nhãn hiệu, như trong Hình 2.

Nhằm mục đích cho thấy trải nghiệm của khách hàng có thể được cải thiện bằng cách làm phong phú dữ liệu sản phẩm / nội dung, bài viết này đóng góp:

(1) Mô tả trường hợp sử dụng thực tế trong đó sản phẩm tăng thông tin cho phép cá nhân hóa tốt hơn;

(2) Một hệ thống hợp nhất các thuộc tính sản phẩm liên quan đến thiếu nhãn trong cài đặt đa tác vụ và ở quy mô;

(3) Một cách tiếp cận hệ thống đề xuất sử dụng thành phần vectơ của các thành phần phối hợp và dựa trên nội dung.

Các khía cạnh chính của hệ thống đề xuất là:

(1) model composition – the hybrid model là một thành phần vectơ của mô hình hợp tác và dựa trên nội dung.

(2) simultaneous optimisation – tối ưu hóa đồng thời hai thành phần của mô hình.

| product | type | segment | pattern | ... |
|---|---|---|---|---|
| A | dress | ? | floral | ? |
| B | dress | girly girl | ? | ... |
| C | skirt | ? | check | ? |
| ... | ... | ... | ... | ... |

*Figure 1.Minh họa dữ liệu sản phẩm gốc.*

Hầu hết các sản phẩm đều thiếu một phần thuộc tính và hầu như không có sản phẩm nào có tất cả các thuộc tính khả dụng.

## 2.3 Design

Minh họa các lựa chọn thực hiện để có thể dự đoán.

### 2.3.1 Image Classification

Thời trang là một lĩnh vực rất được quan tâm và với sự phổ biến của Deep Learning, cho phép thu tập thông tin tiếp cận để phân loại và dự đoán các thuộc tính quần áo từ hình ảnh, cả từ các mô hình được đào tạo từ đầu đến cuối hoặc từ biểu diễn trung gian: ví dụ: [7, 18, 23]. Cùng với hình ảnh, chúng ta có thông tin văn bản để tận dụng và chúng ta và cần áp dụng nó với quy mô lớn. Do đó, xử lý hình ảnh trở thành một phần của một đường ống lớn hơn nơi các tính năng hình ảnh được trích xuất từ các sản phẩm, ảnh chụp được sắp xếp sẵn và được lưu trữ để tăng tốc độ đào tạo mô hình dự đoán thuộc tính, nhưng cũng theo thứ tự cho các tính năng đã sẵn sàng cho các ứng dụng khác.

Bước tạo feature là một ví dụ về biểu diễn Machine Learning, đểthực hiện, chúng ta áp dụng Neural Network. Neural Network được đào tạo trước trên dataset . Mặc dù không phải thuật toán mới, nhưng kiến trúc này vẫn còn rộng rãi áp dụng cho các nhiệm vụ liên quan của phát hiện đối tượng, phân đoạn và truy xuất hình ảnh do khả năng chuyển đổi của của nó. Đối với mỗi lần chụp sản phẩm, chúng ta trích xuất $7 \times 7 \times 512$ các tính năng từ lớp cuối cùng trước khi được kết nối đầy đủ các funtion và thực hiện một hoạt động tổng hợp trên những funtion đó. Điều này kết quả trong 512 tính năng hình ảnh cho mỗi lần chụp sản phẩm, mỗi hình ảnh do đó được chiếu lên so-called embedding space.

### 2.3.2 Text Classification

Neural networks (CNN) đã được chứng minh là có hiệu quả trong việc phân loại không chỉ hình ảnh mà còn văn bản. Thật vậy, một câu có thể được coi là chuỗi từ, trong đó mỗi từ lần lượt có thể được biểu diễn dưới dạng một vectơ trong không gian nhúng từ nhiều chiều.

Tích hợp 1-D trên biểu diễn này, với các bộ lọc bao gồm nhiều từ nhúng cùng một lúc, là một cách hiệu quả để xem xét trật tự từ [8]. Theo [10], chúng tôi đã thử nghiệm cả hai phần nhúng cố định, tức là đã được đào tạo trước và miễn phí, được khởi tạo ngẫu

nhiên; thứ hai thực hiện tốt hơn đáng kể trong trường hợp của chúng tôi. Chúng tôi cũng đảm bảo rằng khi chọn kiến trúc này có một lợi thế so với linear classifier được điều chỉnh tốt trên bag-of-words (TF-IDF), cung cấp một baseline vì các từ trong phần mô tả sản phẩm thường có liên quan trực tiếp đến các giá trị thuộc tính.

Nhánh trung tâm của sơ đồ trong Future 3 mô tả văn bản của chúng ta. processing pipeline một 1D đơn giản trên lớp embedding layer, theo sau là một nhóm tích chập tối đa theo thời gian

Trong thực tế, kể từ khi sản phẩm của chúng ta chú thích thường ngắn (ví dụ: 50 từ hoặc ít hơn), chúng tôi không cần bất kỳ lớp gộp tùy chỉnh nào mặc dù là bản chất đa nhãn. Để xây dựng đủ năng lực để có thể học nhiều thuộc tính cùng một lúc, chúng tôi chỉ tăng số lượng bộ lọc và số lượng neron network dày, tương ứng.

### 2.3.3 Multi-modal Fusion

Hợp nhất các tính năng hình ảnh của một hình ảnh với thông tin từ ngữ được trích xuất từ văn bản kèm theo hình ảnh đã nói, hữu ích trong việc nhận dạng đối tượng nhiệm vụ, đặc biệt khi số lượng lớp đối tượng lớn hoặc thông tin chi tiết rất khó thu thập từ hình ảnh đơn lẻ. Liên quan đến các ứng dụng cho thời trang nói riêng, tồn tại các phương pháp tiếp cận đa phương thức để dự báo bán hàng, phát hiện sản phẩm, và tìm kiếm.

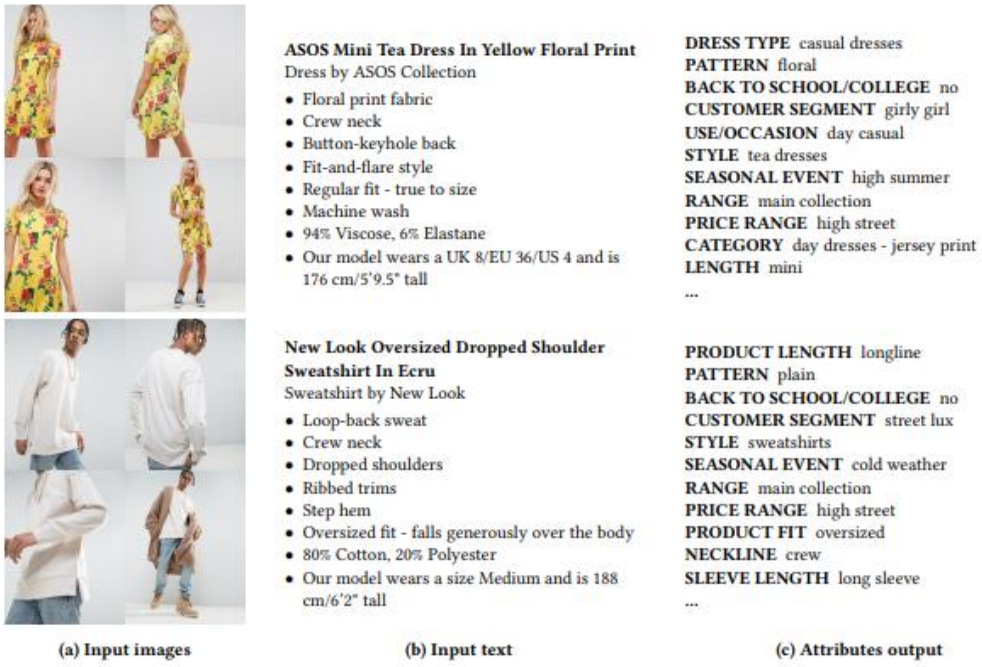



*Figure 2.ví dụ với (a) hình ảnh sản phẩm được liên kết và (b) chi tiết sản phẩm*

Giải quyết một vấn đề liên quan chặt chẽ, phân loại sản phẩm trong thương mại điện tử, họ sử dụng một image CNN, một CNN word.

Giải pháp của chúng tôi, ngoài việc phải xử lý các nhãn được chú thích một phần, như sẽ được thảo luận trong phần tiếp theo, cũng phải giải quyết một lượng lớn dữ liệu sản phẩm bổ sung (loại sản phẩm, nhãn hiệu và thành phần), và hợp nhất các bảng mã từ tất cả các đầu vào này thông qua kiến trúc được mô tả trong Hình 3. Cụ thể, thông tin loại sản phẩm là được sử dụng cả khi hợp nhất các hình ảnh được tính toán trước, kể từ khi chứa nhiều sản phẩm và khi hợp nhất tính năng hình ảnh với thông tin từ ngữ (tiêu đề sản phẩm và mô tả) được xử lý bởi văn bản CNN, cùng với các nhúng nhúng thương hiệu và phân chia được đào tạo từ đầu đến cuối. Đáng chú ý là cấu trúc của các lớp đầu ra như được thảo luận tiếp theo.

### 2.3.4 Multi-Task learning with Missing Labels

Giải pháp đơn giản nhất cho dữ liệu đa nhãn sẽ là one-hot mã hóa và nối tất cả các nhãn thành một đầu ra duy nhất, sau đó train một netwwork với entropy nhị phân như là

một loss và kích hoạt sigmoid trong lớp cuối cùng của nó. Tuy nhiên, cách tiếp cận này sẽ coi tất cả các lớp trên tất cả các nhãn là độc lập lẫn nhau, trong khi thực tế, mỗi thuộc tính chỉ có một tập hợp con là tất cả các giá trị mục tiêu. Trong việc liên quan để phân loại và chú thích hình ảnh đa nhãn, các phương pháp tiếp cận hiện đại cũng hướng đến mô hình hóa các phụ thuộc nhãn này, ví dụ bằng cách xếp chồng CNN với lớp lặp lại (RNN) xử lý các nhãn được dự đoán bởi CNN. Tuy nhiên, sự các điểm nổi bật của các nhãn bị thiếu trong khi train dữ liệu sẽ yêu cầu thực hiện một chức năng loại bỏ tùy chỉnh hoặc che các lớp để tránh lỗi lan truyền khi nhãn không có sẵn. Vì lý do này, chúng tôi đã chọn một đa tác vụ đơn giản hơn là cách tiếp cận, vẫn cho phép một mức độ phân cấp nhãn bởi vì nó khớp với từng thuộc tính với các mất mát entropy, theo đó lớp dự đoán được chọn trong số các giá trị có thể chỉ cho thuộc tính đó. Đó là, mỗi mục tiêu trở thành một vấn đề riêng biệt, mất cân bằng, đa lớp. Do đó, chúng ta có thể thực hiện tùy chỉnh kế hoạch trọng số cho việc mất các mục tiêu khác nhau tùy thuộc trên các tần số nhãn của thuộc tính cụ thể.

A multi-task network cung cấp cho chúng ta cách học hiệu quả từ tất cả các sản phẩm có ít nhất một trong các thuộc tính đã được dán nhãn, do đó thực hiện tăng dữ liệu ngầm, và cũng ngầm định khi đồng thời phù hợp nhiều mục tiêu. Ngược lại, thậm chí không quan tâm đến sự phức tạp thêm vào của việc phải duy trì nhiều mô hình trong sản xuất, single-task networks cho các thuộc tính riêng lẻ chỉ có thể tận dụng một phần dữ liệu (xem Bảng 1, cột thứ hai), có nguy cơ quá cao. Cũng tệ hơn, như đã thảo luận trước đây, một mô hình đa nhãn đơn nhiệm vụ sẽ không có đủ dữ liệu đào tạo nếu không có cách thông minh để đối phó với các nhãn bị thiếu.

Trong thực tế, như được minh họa trong Hình 3, chúng tôi xây dựng một mô hình cho mỗi thuộc tính, nhưng tất cả chúng đều chia cùng một tham số cho đến đầu ra lớp, theo ‘hard’ parameter-sharing paradigm [19, 22]. Chúng tôi cũng chuẩn bị một tập dữ liệu cho mỗi thuộc tính. Trong quá trình đào tạo, một cách ngẫu nhiên thứ tự tuần tự, chúng tôi cập nhật từng mô hình với một mật độ giảm dần.

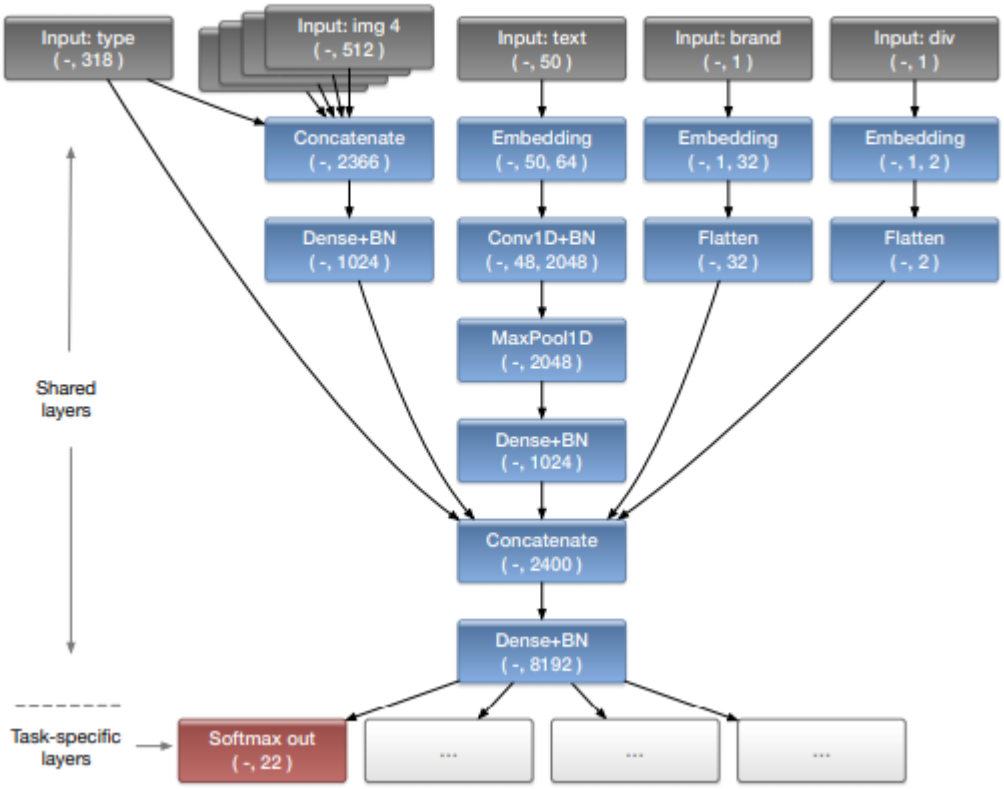



*Figure 3. Multi-modal multi-task architecture.*

Trên thực tế, các thí nghiệm sơ bộ đã tạo ra sự bất ổn khi sử dụng tối ưu hóa thích ứng với động lượng [11]; cụ thể là độ chính xác của bộ xác nhận dường như bị đình trệ nhanh hơn và dao động nhiều hơn trên các mô hình khác nhau. Có bằng chứng thực nghiệm cho thấy thích nghi phương pháp có thể mang lại giải pháp tổng quát hóa tồi tệ hơn phương pháp từ SGD đơn giản với động lượng. Tuy nhiên, nó không rõ ràng cho dù điều đó áp dụng cho một cài đặt đa tác vụ với chia tham số. Trong trường hợp của chúng tôi, việc bổ sung động lượng đơn giản cho SGD dường như đã làm tổn thất hiệu suất đào tạo. Do đó, cuối cùng chúng tôi đã dùng đến sử dụng vanilla SGD với tốc độ học tập không đổi và không có momentor.

# 3. Deep Learning

## *3.1 Giới thiệu về Deep learning*

Để hiểu được Deep Learning là gì, trước hết chúng ta cần phải hiểu được mối quan hệ giữa Deep Learning và machine learning, mạng neuron, và trí tuệ nhân tạo. Cách tốt nhất để hiểu về mối quan hệ này là mường tượng chúng thành những vòng tròn đồng tâm:

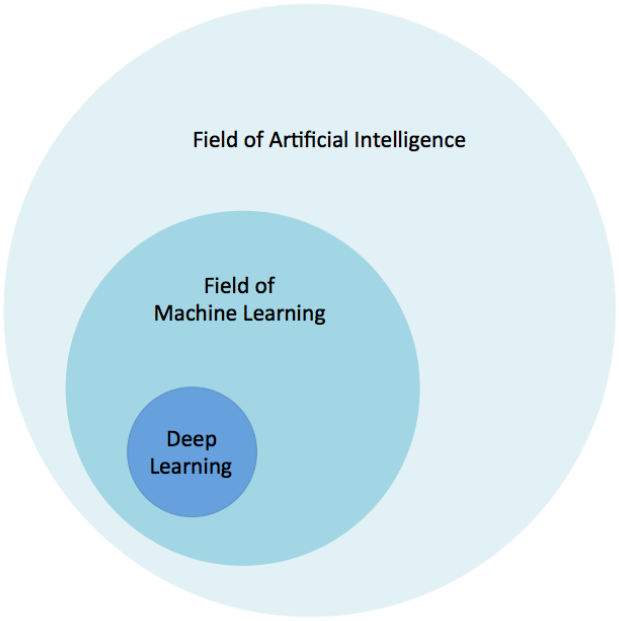



*Figure 4. Mô phỏng cho deep learning*

Ở vòng ngoài cùng, bạn có trí thông minh nhân tạo (sử dụng máy tính và lập luận). Bên trong lớp này là machine learning. Với mạng neuron nhân tạo và deep learning tại trung tâm.

Học sâu là một phân nhánh của ngành học máy dựa trên một tập hợp các thuật toán để cố gắng mô hình dữ liệu trừu tượng hóa ở mức cao bằng cách sử dụng nhiều lớp xử lý với cấu trúc phức tạp.

Học sâu là một kỹ thuật trong học máy, có liên quan đến các thuật toán lấy cảm hứng dựa trên cấu trúc và cơ chế hoạt động của bộ não và được gọi là mạng thần kinh nhân tạo

(Artificial Neural Network - ANN). Một quan sát có thể được biểu diễn bằng nhiều cách như một vector của các giá trị cường độ cho mỗi điểm ảnh, hoặc một cách trừu tượng hơn như là một tập hợp các cạnh, các khu vực có hình dạng cụ thể ( nhận dạng khuôn mặt, nhận dạng các món quần áo đang mặc…)

Có nhiều kiến trúc Học sâu khác nhau như mạng nơ-ron tích chập (Convolution Neural Network - CNN), mạng nơ ron hồi quy (Recurrent Neural Network - RNN), Deep belief network (DBN), được áp dụng vào các lĩnh vực như thị giác máy tính (Computer vision), nhận diện giọng nói (Speech recognition), xử lý ngôn ngữ tự nhiên (Natural language processing). Deep learning được chứng mình là đã tạo ra các kết quả rất tốt đối với nhiều nhiệm vụ khác nhau.

## *3.2 Mạng nơ-ron nhân tạo (Neural Network NN)*

Mạng nơ-ron nhân tạo (Neural Network - NN) là một mô hình lập trình rất đẹp lấy cảm hứng từ mạng nơ-ron thần kinh. Kết hợp với các kĩ thuật học sâu (Deep Learning - DL), NN đang trở thành một công cụ rất mạnh mẽ mang lại hiệu quả tốt nhất cho nhiều bài toán khó như nhận dạng ảnh, giọng nói hay xử lý ngôn ngữ tự nhiên.

Theo nghĩa sinh học, mạng Neural là một tập hợp các dây thần kinh kết nối với nhau. Trong Deep learning, Neural networks để chỉ mạng Neural nhân tạo, cấu thành từ các lớp Neural.

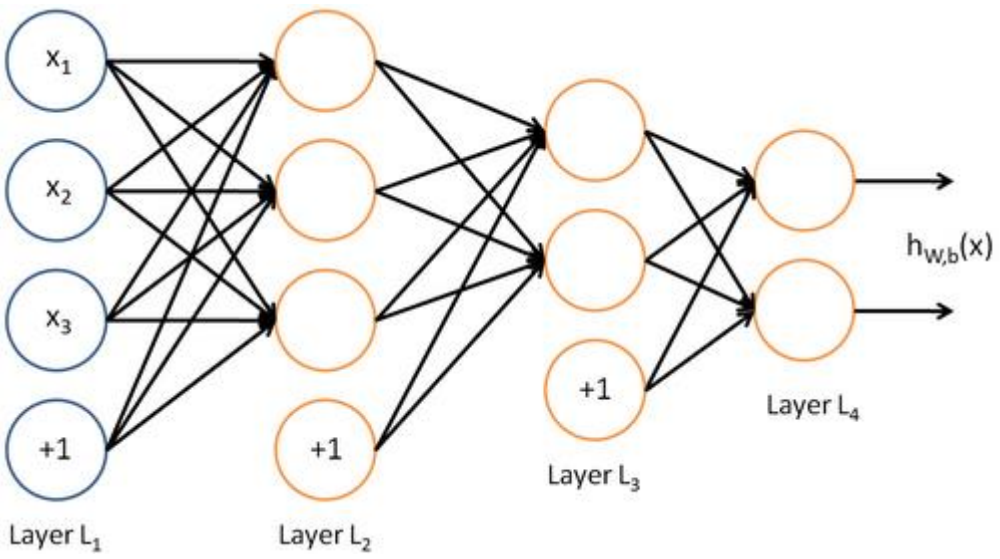



*Figure 5. Mô hình Neura Network.l*

## 3.2.1 Cấu trúc

### 3.2.1.1 Mạng nơ-ron sinh học (Biological Neural Networks)

Não bộ con người là một mạng lưới khoảng 1011 tế bào thần kinh hay còn gọi là nơron. Chúng có cấu trúc và chức năng tương đối đồng nhất. Các nhà nghiên cứu sinh học về bộ não con người đã đưa ra kết luận rằng các nơron là đơn vị đảm nhiệm những chức năng nhất định trong hệ thần kinh bao gồm não, tuỷ sống và các dây thần kinh. Hình dưới đây chỉ ra cấu tạo của hệ thống tế bào sinh học này.

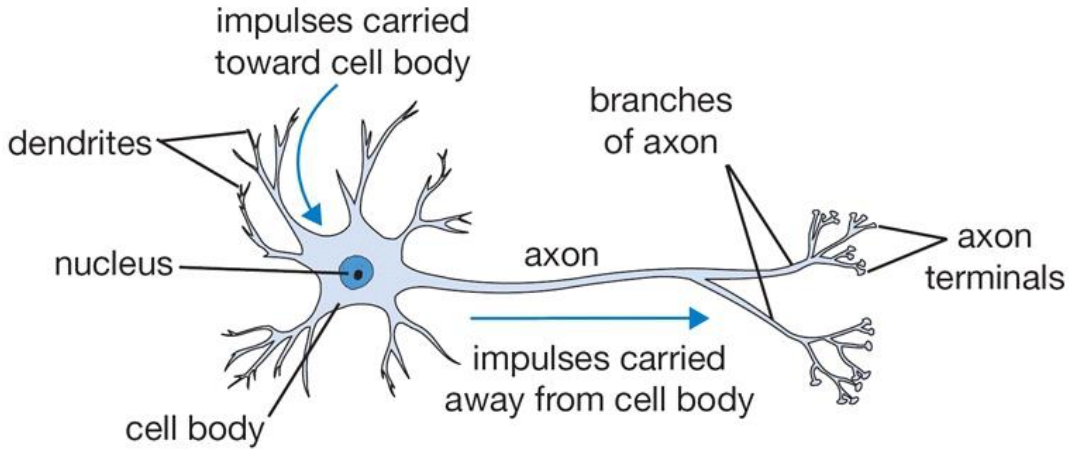



*Figure 6. Mô hình cấu tạo neural thần kinh*

Chính cấu trúc mạng nơron và mức độ liên kết của các khớp nối đã tạo nên chức năng của hệ thần kinh con người. Quá trình phát triển của hệ thần kinh là một quá trình "học" liên tục. Ngay từ khi chúng ta sinh ra, một số cấu trúc thần kinh đơn giản đã được hình thành. Sau đó các cấu trúc khác lần lượt được xây dựng thêm nhờ quá trình học. Do đó cấu trúc mạng nơron liên tục biến đổi để ngày càng phát triển hoàn thiện.

Một vấn đề đặt ra là dựa trên những kết quả nghiên cứu về hệ thần kinh con người chúng ta có thể mô phỏng, xây dựng lên các hệ thần kinh nhân tạo nhằm phục vụ cho một chức năng nào đó không. Nghiên cứu trả lời câu hỏi này đã đưa ra một hướng phát triển mới: Mạng nơron nhân tạo.

### 3.2.1.2 Mạng nơ-ron nhân tạo

#### *3.2.1.2.1 Perceptrons*

##### 3.2.1.2.1.1 Perceptrons cơ bản.

Một nơ-ron có thể nhận nhiều đầu vào và cho ra một kết quả duy nhất. Mô hình của perceptron cũng tương tự như vậy:

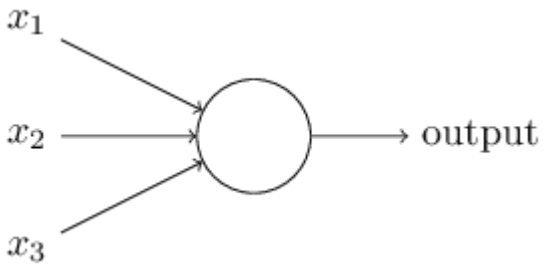



*Figure 7. Perceptrons cơ bản.*

Một perceptron sẽ nhận một hoặc nhiều X vào dạng nhị phân và cho ra một kết quả O dạng nhị phân duy nhất. Các đầu vào được điều phối tầm ảnh hưởng bởi các tham số trọng lượng tương ứng W của nó, còn kết quả đầu ra được quyết định dựa vào một ngưỡng quyết định $b$ nào đó:

$$y = \begin{cases} 0 \; if \; \sum x_i w_i \; \leq threshold \\ 1 \; if \; \sum x_i w_i \; > threshold \end{cases}$$

đặt $b = -threshold$ và (1) có thể viết lại thành

$$y = \begin{cases} 0 \; if \; \sum x_i w_i + b \leq 0 \\ 1 \; if \; \sum x_i w_i + b > 0 \end{cases}$$

Ví dụ: Việc có nên đi chơi chơi không dựa vào 4 yêu tố:

- 1. Trời có nắng hay không?
- 2. Có hẹn trước hay không?
- 3. Có kế hoạch hay không?
- 4. Bạn có ít khi gặp được hay không?

Thì ta coi 4 yếu tố đầu vào là $x_1, x_2, x_3, x_4$. và nếu o=0 thì ta không đi chơi còn o=1 thì ta đi Type equation here.chơi. Giả sử mức độ quan trọng của 4 yếu tố trên lần lượt là $W_1 = 0{,}05, W_2 = 0{,}5, W_3 = 0{,}2, W_4 = 0{,}25$ và chọn ngưỡng $b = -0{,}5$ thì ta có thể thấy rằng việc trời nắng có ảnh hưởng chỉ 5% tới quyết định đi nhậu và việc có hẹn từ trước ảnh hưởng tới 50% quyết định đi nhậu của ta.

Nếu gắn $x_0 = 1, w_o = b$ ta còn có thể viết gọn lại thành:

$$\text{o} = \begin{cases} 0 \; if \; w^T x \leq 0 \\ 1 \; if \; w^T x > 0 \end{cases}$$

### 3.2.1.2.1.2 Sigmoid Neurons

Với đầu vào và đầu ra dạng nhị phân, ta rất khó có thể điều chỉnh một lượng nhỏ đầu vào để đầu ra thay đổi chút ít, nên để linh động, ta có thể mở rộng chúng ra cả khoảng [0,1]. Lúc này đầu ra được quyết định bởi một hàm sigmoid:

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Đồ thị của hàm này cũng cân xứng rất đẹp thể hiện được mức độ công bằng của các tham số:

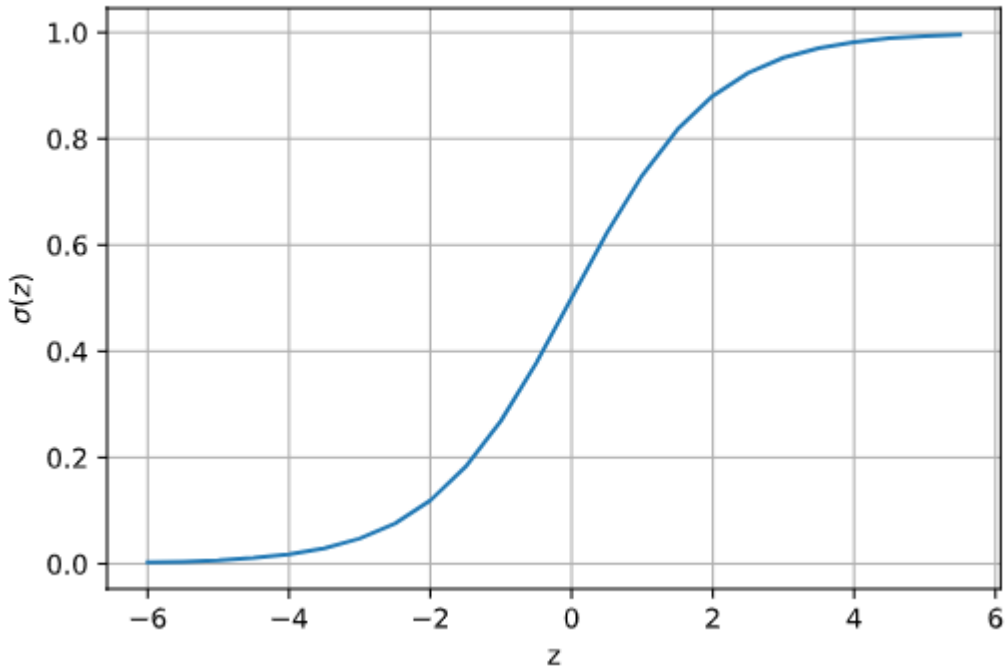



*Figure 8. Sigmoid funtion*

Đặt  thì công thức của perceptron lúc này sẽ có dạng:

$$o = \sigma(z) = \frac{1}{1 - \exp(-\boldsymbol{w}^T x)}$$

Với kết quả đầu ra dạng nhị phân 0 và 1, nhược điểm ở đây là với một sự thay đổi nhỏ ở dữ liệu đầu vào cũng có thể làm đảo ngược kết quả nhị phân ở đầu ra dẫn đến sự thay đổi lớn ảnh hưởng tới toàn hệ thống mạng. Để khắc phục nhược điểm trên kết quả đầu ra được quyết định bằng một hàm sigmoid $\sigma(x_i w_i + b)$.

Tới đây thì ta có thể thấy rằng mỗi sigmoid neuron cũng tương tự như một bộ phân loại tuyến tính (logistic regression) bởi xác suất

$$P(y_i = 1 | x_i; \boldsymbol{w}) = \sigma(\boldsymbol{w}^T x)$$

Bằng cách biểu diễn như vậy, ta có thể coi neuron sinh học được thể hiện như sau:

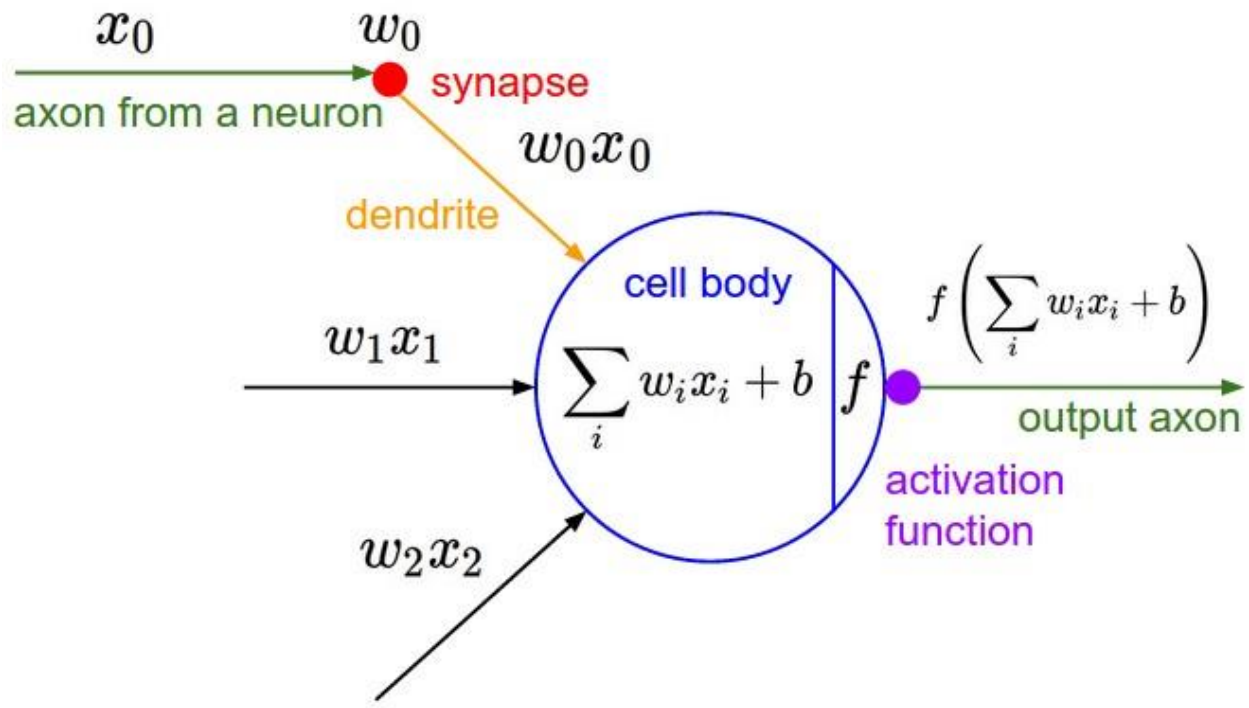



*Figure 9.Mô hình Nơ-ron*

Một điểm cần lưu ý là các hàm kích hoạt buộc phải là hàm phi tuyến. Vì nếu nó là tuyến tính thì khi kết hợp với phép toán tuyến tính $\boldsymbol{w}^T x$ thì kết quả thu được cũng sẽ là một thao tác tuyến tính dẫn tới chuyện nó trở nên vô nghĩa.

## *3.3* Kiến Trúc Neural Network

Mạng NN là sự kết hợp của của các tầng perceptron hay còn được gọi là perceptron đa tầng (multilayer perceptron) như hình vẽ bên dưới:

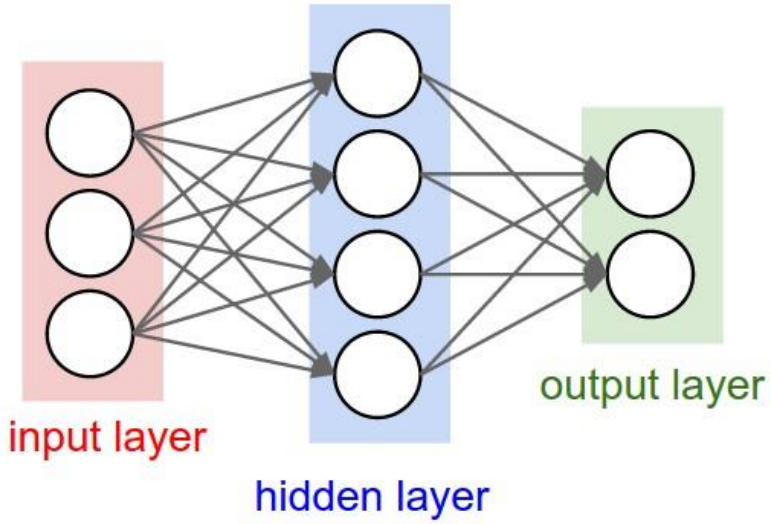



*Figure 10. mô hình multilayer.*

Một mạng NN sẽ có 3 kiểu tầng:

**Tầng vào (input layer)**: Là tầng bên trái cùng của mạng thể hiện cho các đầu vào của mạng.

**Tầng ra (output layer)**: Là tầng bên phải cùng của mạng thể hiện cho các đầu ra của mạng.

**Tầng ẩn (hidden layer)**: Là tầng nằm giữa tầng vào và tầng ra thể hiện cho việc suy luận logic của mạng.

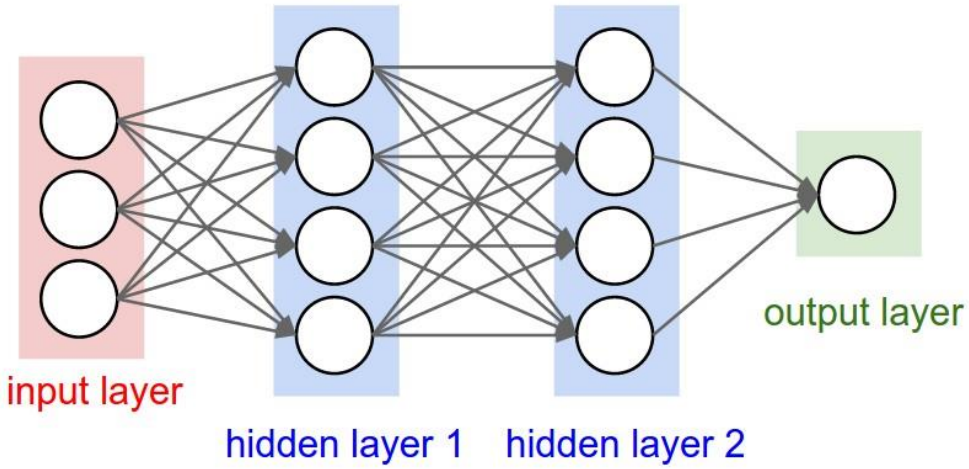



*Figure 11.mô hình multilayer 2*

Trong mạng NN, mỗi nút mạng là một sigmoid nơ-ron nhưng hàm kích hoạt của chúng có thể khác nhau. Tuy nhiên trong thực tế người ta thường để chúng cùng dạng với nhau để tính toán cho thuận lợi.

Ở mỗi tầng, số lượng các nút mạng (nơ-ron) có thể khác nhau tuỳ thuộc vào bài toán và cách giải quyết. Nhưng thường khi làm việc người ta để các tầng ẩn có số lượng nơ-ron bằng nhau. Ngoài ra, các nơ-ron ở các tầng thường được liên kết đôi một với nhau tạo thành mạng kết nối đầy đủ (full-connected network). Khi đó ta có thể tính được kích cỡ của mạng dựa vào số tầng và số nơ-ron. Ví dụ ở hình trên ta có:

- tầng mạng, trong đó có 2 tầng ẩn:
- 3+4*2+1=12 nút mạng
- (3*4+4*4+4*1) + (4+4+1)= 41 tham số.

## *3.4 Phương pháp huấn luyện*

### 3.4.1 Gradient descent

Trong các bài toán tối ưu, chúng ta thường xuyên phải tìm giá trị cực tiểu (hoặc cực đại) của một hàm số nào đó. Tuy nhiên việc tìm giá trị mà tại đó hàm số đạt giá trị nhỏ nhất (global minimum) của các hàm mất mát đôi khi là rất phức tạp. Thay vì đi tìm global minimum, người ta sẽ đi tìm các điểm cực tiêu (local minimum), và xét ở một mức độ nào đó, đây có thể coi là nghiệm của hàm mất mát cần tìm. Các điểm local minimum là nghiệm của phương trình đạo hàm bằng 0, theo lý thuyết ta có thể tìm toàn bộ các điểm local minimum sau đó thế lần lượt vào hàm số để tìm ra global minimum. Tuy nhiên, trong thực tế việc giải phương trình đạo hàm bằng 0 đôi khi là bất khả thi, nguyên nhân có thể đến từ sự phức tạp của hàm số, số lượng chiều dữ liệu.

Giải pháp ở đây là chúng ta sẽ khởi tạo một điểm trong đồ hàm số và sử dụng phép toán lặp để tiến dần về điểm cực tiểu. Gradient descent là một trong những phương pháp được sử dụng nhiều nhất.

Thuật toán Gradient descent:

1. Khởi tạo giá trị $x = x_t$ bất kỳ.
2. Gán $x_t = x_{t-1} - \eta \cdot f'(x_{t-1})$ với $\eta(learning\ rate)$ là hằng số không âm.
3. Tính $f(x_t)$, nếu $f(x_t)$ đủ nhỏ thì ngừng vòng lặp, người lại tiếp tục thực hiện bước 2.

Tốc độ hội tụ của Gradient Descent không chỉ phụ thuộc và điểm khởi tạo ban đầu mà còn phụ thuộc vào tham số learning rate, do đó việc lựa chọn learning rate rất quan trọng, sẽ có ba trường hợp xảy ra:

- Learning rate nhỏ: Tốc độ hội tụ chậm, ảnh hưởng nhiều đến tốc độ của thuật toán.
- Learning rate quá lớn: Dẫn đến việc thuật toán không tìm được giá trị nhỏ nhất do bước nhảy quá lớn.

- Learning rat hợp lý: Tìm được giá trị nhỏ nhất phù hợp sau một số lần lặp vừa phải.

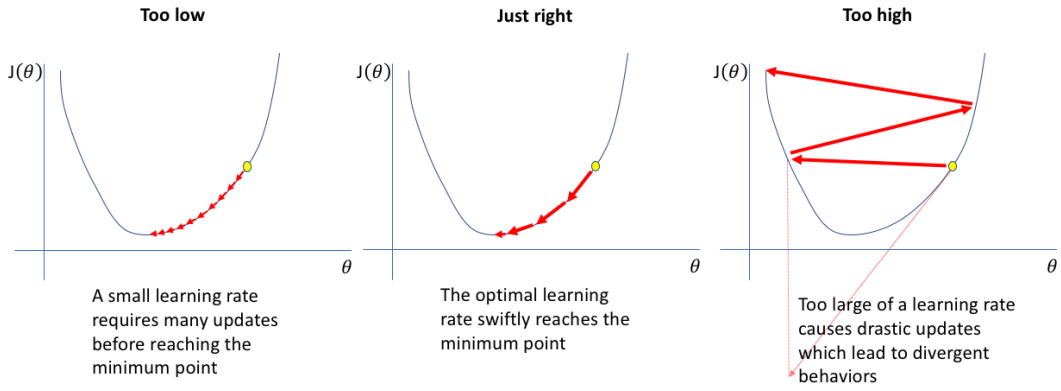



*Figure 12. biểu đồ gradient Descent*

Thuật toán Gradient Descent đề cập ở trên còn còn được gọi với tên khác là Batch Gradient Descent. "Batch" ở đây được hiểu là tất cả, nghĩa là ta sẽ sử dụng toàn bộ dữ liệu để tính toán. Phương pháp này sẽ gặp một vài hạn chế đối với tập dữ liệu có kích thước lớn. Việc tính toán lại đạo hàm cho tất cả các điểm sau mỗi vòng lặp trở nên tốn kém và không hiệu quả.

Để giải quyết khó khăn nêu trên, Stochastic Gradient Descent và Mini-Batch Gradient Descent là cải tiến của Batch Gradient Descent được sinh ra. Hai thuật toán này đơn giản hơn Batch Gradient Descent tuy nhiên lại rất hiệu quả.

**Stochastic Gradient Descent:** Chỉ dùng một điểm dữ liệu cho mỗi lần thực hiện bước tính đạo hàm. Mỗi lần duyệt qua toàn bộ dữ liệu được gọi là một *epoch*. Đối với BGD thì mỗi epoch tương ứng với 1 lần cập nhật $x$, đối với SGD thì mỗi epoch tương ứng với N lần cập nhật $x$ với N là số điểm dữ liệu

**Mini-Batch Gradient Descent:** Dùng một phần dữ liệu cho mỗi lần thực hiện bước tính đạo hàm. MGD sử dụng một lượng $n$ lớn hơn 1 và nhỏ hơn $N$. MGD sẽ chia toàn bộ dự liêu ra làm các *mini-batch* mỗi *mini-batch* có *n* dữ liệu (ngoại trừ *mini-batch* cuối

cùng có thể có số dữ liệu nhỏ hơn *n* do *N* không chia hết cho *n*. Mỗi lần cập nhật, thuật toán này lấy ra một *mini-batch* để tính đạo hàm và cập nhật giá trị hàm mất mát.

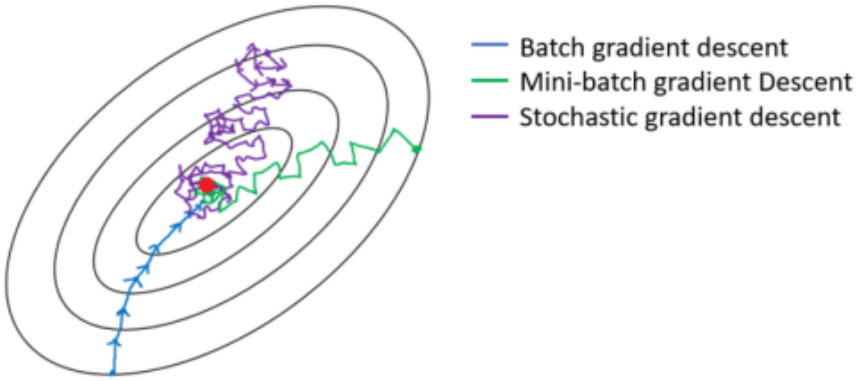



*Figure 13. Biểu đồ giao động gradient desent*

Hình trên mô tả giao động giá trị của hàm mất mát đối với 3 thuật toán Gradient Descent. Ta thấy BGD giảm đều sau mỗi epoch và đạt được giá trị hàm mất mát tối ưu. MGD cũng có xu hướng giảm nhưng xuất hiện giao động tuy nhiên vẫn đạt được giá trị hàm mất mát tối ưu. SGD thì có độ giao động lớn và sau rất nhiều bước vẫn chưa đạt được giá trị tối ưu cho hàm mất mát.

### 3.4.2 Lan truyền tiến

Như bạn thấy thì tất cả các nốt mạng (nơ-ron) được kết hợp đôi một với nhau theo một chiều duy nhất từ tầng vào tới tầng ra. Tức là mỗi nốt ở một tầng nào đó sẽ nhận đầu vào là tất cả các nốt ở tầng trước đó mà không suy luận ngược lại. Hay nói cách khác, việc suy luận trong mạng NN là suy luận tiến (feedforward):

$$z_i^{(l+1)} = \sum_{i=1}^{n^{(l)}} w_{ij}^{(l+1)} + a_i^{(l)} + b_i^{(l+1)}$$

$$a_i^{(l+1)} = f(z_i^{(l+1)})$$

Trong đó, $n^{(l)}$ số lượng nút ở tầng $l$ tương ứng và là nút mạng thứ $a_i^{(l)}$của tầng *l*. Còn $w_{ij}^{(l+1)}$ là tham số trọng lượng của đầu vào $a_i^{(l)}$ đối với nút mạng thứ *i* của

tầng *l*+1 và $b_i^{(l+1)}$ là độ lệch (*bias*) của nút mạng thứ *i* của tầng *l*+1. Đầu ra của nút mạng này được biểu diễn bằng $a_i^{(l+1)}$ ứng với hàm kích hoạt $f(z_i^{(l+1)})$ tương ứng.

Để tiện tính toán, ta coi $a_0^{(l)}$ là một đầu vào và $w_{i0}^{(l+1)} = b_i^{(l+1)}$ là tham số trọng lượng của đầu vào này. Lúc đó ta có thể viết lại công thức trên dưới dạng véc-tơ:

$$z_i^{(l+1)} = w_i^{(l+1)}.a^l$$

$$a_i^{(l+1)} = f(z_i^{(l+1)})$$

### 3.4.3 Lan truyền ngược và đạo hàm

Để tính đạo hàm của hàm lỗi $\Delta j(w)$ trong mạng NN, ta sử dụng một giải thuật đặc biệt là giải thuật lan truyền ngược (backpropagation). Nhờ có giải thuật được sáng tạo vào năm 1986 này mà mạng NN thực thi hiệu quả được và ứng dụng ngày một nhiều cho tới tận ngày này.

Về cơ bản phương pháp này được dựa theo quy tắc chuỗi đạo hàm của hàm hợp và phép tính ngược đạo hàm để thu được đạo hàm theo tất cả các tham số cùng lúc chỉ với 2 lần duyệt mạng.

Giải thuật lan truyền ngược được thực hiện như sau:

1. **Lan truyền tiến**:

   Lần lượt tính các , $a^{(l)}$ từ $l$=2⟶$L$ theo công thức:

$$z^{(l)} = \boldsymbol{w}^{(l)}.\boldsymbol{a}^{(l-1)}$$

$$a^{(l)} = \boldsymbol{w}^{(l)}.\boldsymbol{a}^{(l-1)}$$

2. **Tính đạo hàm theo** z **ở tầng ra:**

$$\frac{\partial j}{\partial z^{(L)}} = \frac{\partial j}{\partial a^{(l)}}\frac{\partial a^{(l)}}{\partial z^{(L)}}$$

   với $z^{(l)}$, $a^{(l)}$ vừa tính được ở bước 1

3. **Lan truyền ngược**:

Tính đạo hàm theo $z$ ngược lại từ $l=(L-1) \rightarrow 2$ theo công thức:

$$\frac{\partial j}{\partial z^{(l)}} = \frac{\partial j}{\partial z^{(l+1)}} \frac{\partial z^{(l)}}{\partial a^{(l)}} \frac{\partial a^{(l)}}{\partial z^{(l)}}$$
$$= \left( \left( W^{(l+1)} \right)^T \frac{\partial j}{\partial z^{(l+1)}} \right)$$

với $z^{(l)}$ tính được ở bước 1 và $\frac{\partial j}{\partial z^{(l)}}$ tính được ở vòng lặp ngay trước.

4. **Tính đạo hàm**:

Tính đạo hàm theo tham số w bằng công thức:

$$\frac{\partial j}{\partial w^{(l)}} = \frac{\partial j}{\partial z^{(l)}} \frac{\partial z^{(l)}}{\partial w^{(L)}}$$
$$= \frac{\partial j}{\partial z^{(l)}} \left( a^{(l-1)} \right)^T$$

với $\left( a^{(l+1)} \right)^T$ tính được ở bước 1 và $\frac{\partial j}{\partial z^{(l)}}$ tính được ở bước 3.

## *3.5 Mạng nơ-ron hồi quy (Recurrent neural network-RNN)*

Trong mạng nơ-ron truyền thống, các đầu vào và đầu ra là độc lập với nhau, có nghĩa là chúng không liên kết với nhau tạo thành chuỗi. Điều này không phù hợp với một số bài toán có dữ liệu tuần tự (video, câu văn,..). Và mạng nơ-ron hồi quy được sinh ra để giải quyết vấn đề trên.

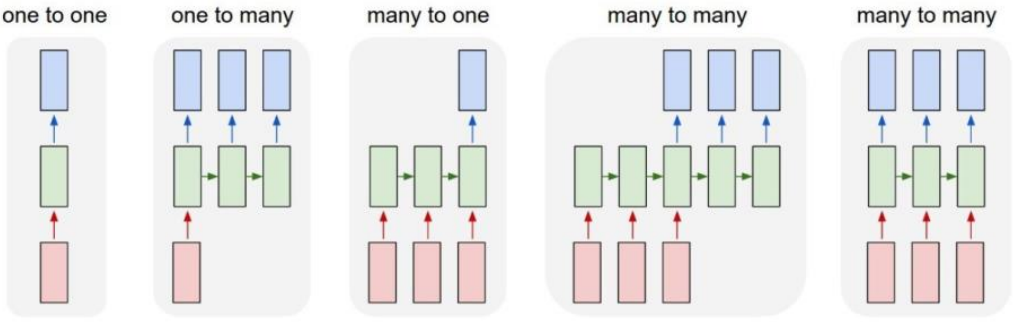



*Figure 14.Các dạng bài toán RNN*

**One to one:** Thường sử dụng cho mạng nơ-ron và mạng nơ-ron tích chập, có một đầu vào và một đầu ra. Ví dụ đối với mạng nơ-ron tích chập thì đầu vào là một ảnh và đầu ra là một phân đoạn hình ảnh.

**One to many:** Bài toán này có số lượng đầu vào là một và nhiều đầu ra. Ví dụ như bài toán mô tả ảnh, đầu vào là một ảnh và đầu ra là nhiều từ mô tả cho ảnh đấy.

**Many to one:** Bài toán này có nhiều đầu vào nhưng chỉ có một kết quả đầu ra. Ví dụ như bài toán phân loại hành động trong video, đầu vào sẽ là nhiều ảnh tách ra từ video và đầu ra là hành động trong video đó.

**Many to many:** Gồm có nhiều đầu vào và nhiều đầu ra. Ví dụ như bài toán dịch ngôn ngữ, giả sử đầu vào là câu gồm nhiều chữ: "I love Vietnam" thì đầu ra cũng là một câu gồm nhiều chữ là "Tôi yêu Việt Nam"

Mạng nơ-ron tích hồi quy được ứng dụng nhiều trong các lĩnh vực như: Chuyển giọng nói thành văn bản (Speech to text), dịch ngôn ngữ (Machine translation), nhận diện hành động, nội dung trong video (Video recognition), chẩn đoán bệnh tim trong y tế.

Ở các mạng nơ-ron truyền thống thì đầu vào và đầu ra là độc lập với nhau. Trong RNN thì các nơ-ron thực hiện cùng 1 tác vụ với đầu ra phụ thuộc vào các phép tính trước nó, hoặc có thể hiểu là RNN có khả năng lưu trữ các thông tin tính toán trước đó.

Ý tưởng chính của RNN là xử lý chuỗi các thông tin. Cấu trúc gồm nhiều nơ-ron kết nối với nhau tạo thành 1 chuỗi liên tục.

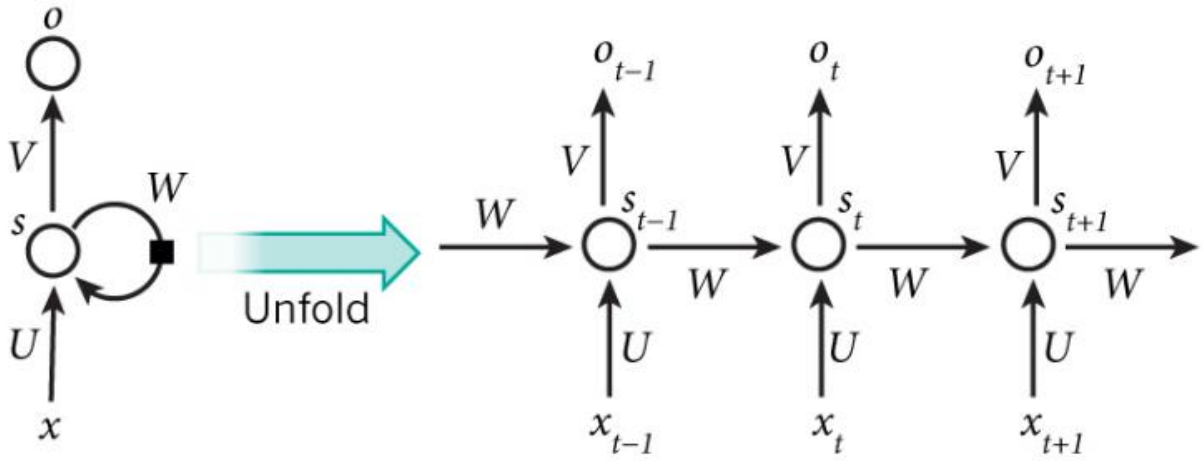



*Figure 15.Mạng nơ-ron hồi quy tổng quát và sau khi duỗi thẳng*

Mô hình trên mô tả phép triển khai nội dung của một RNN. Tương ứng với mỗi từ trong câu input thì sẽ có tương ứng số tầng nơ-ron. Giả sử ta có 1 câu gồm "Trời hôm nay đẹp quá" thì sẽ có 5 lớp tương ứng với mỗi chữ trong câu.

$X_t$ là đầu vào bước t. Giả sử $X_1$ là một véc-tơ tương ứng với từ thứ 2 trong câu (hôm)

$S_t$ là trạng thái ẩn tại bước t. Nó chính là bộ nhớ của mạng. $S_t$ được tính toán dựa trên các trạng thái ẩn phía trước và đầu vào tại bước đó.

Công thức: $S_t = f_{active}(W . s_{t-1} + U . x_t)$.Với hàm $f_{active}$ là hàm kích hoạt (activation function) có nhiệm vụ là chuẩn hoá đầu ra.

$O_t$ là đầu ra tại bước t. Ví dụ, ta muốn dự đoán từ tiếp theo có thể xuất hiện trong câu thì $O_t$ chính là một vecto xác xuất các từ trong danh sách từ vựng

Công thức: $O_t = f(V. S_t)$. Với W,U,V là những trọng số huấn luyện.

Để huấn luyện mạng nơ-ron hồi quy, ta phải thay đổi cách giải thuật lan truyền ngược (backpropagation) vì đạo hàm tại mỗi đầu ra không chỉ phụ thuộc vào bước tính toán trước đó mà còn phụ thuộc vào bước tính toán trước đó nữa, do các tham số trong mạng RNN được sử dụng chung cho tất cả các bước trong mạng.

Giả sử để tính đạo hàm ở bước thứ 4, ta cần phải tính đạo hàm của ba bước trước đó rồi cộng tổng đạo hàm của chúng lại với nhau. Việc tính toán kiểu này được gọi là lan truyền ngược liên hồi.

Có 3 tham số ta cần tìm là U,W,V tức là ta cần tính đạo hàm của $\frac{\partial L}{\partial U}$, $\frac{\partial L}{\partial W}$ và $\frac{\partial L}{\partial V}$.

Tính đạo hàm của L với W ở state thứ i:

$\frac{\partial L}{\partial W} = \frac{\partial L}{\partial \hat{y}} * \frac{\partial \hat{y}}{\partial S30} * \frac{\partial S30}{\partial Si} * \frac{\partial Si}{\partial W}$, trong đó $\frac{\partial S30}{\partial Si} = \prod_{j=i}^{29} \frac{\partial Si+1}{\partial Si}$

Ta có $S_t$ = tanh ($W*S_{29} + U*x_{t+1}$). Ta thấy có xuất hiện của $S_{29}$ nên có 29 phụ thuộc vào W.

$\frac{\partial S_t}{\partial S_{t-1}}$ = (1- $S_t^2$) * W => $\frac{\partial S_{30}}{\partial S_i}$ = $W^{30-I} * \prod_{j=i}^{29}(1 - S_t^2)$

Ta có $S_j$ và W đều nhỏ hơn 1 nên khi tính toán nhiều lần những giá trị nhỏ hơn 1 thì $\frac{\partial S30}{\partial Si}$ sẽ dần tiến về 0 hay $\frac{\partial L}{\partial W}$ sẽ dần tiến về 0, gây ra hiện tượng triệt tiêu đạo hàm (vanish gradient).

Dễ thấy rằng các trạng thái ở quá càng xa thì giá trị của các hệ số học không được cập nhật, tức là mạng nơ-ron hồi quy không học được các thông tin ở trước đó quá xa do hiện tượng triệt tiêu đạo hàm.

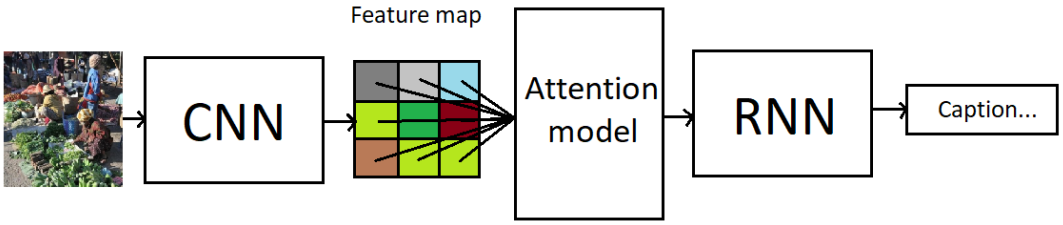
Để khắc phục điều này thì ta sẽ dùng mô hình cải tiến của RNN là Bộ nhớ dài – ngắn hạn(Long short-term memory).

# 4. Bài Toán Phối Trang Phục Theo Sự Kiện

## *4.1 Kiến trúc mô hình học sâu đối với bài toán*

Mô hình sử dụng cho bài toán phối đồ theo sự kiện nhận một bộ hình ảnh $I$ làm dữ liệu đầu vào và được huấn luyện để tối đa hóa xác suất $p(S|I)$ để nhận dang sự kiện thích hợp trong tập dữ liệu đã được huấn luyện.

Với cấu trúc mã hoá – giải mã, một mạng CNN được sử dụng làm bộ mã hoá – mã hoá hình ảnh thành các thông tin cần thiết, một mạng RNN làm bộ giải mã – giải mã các thông tin về hình ảnh thành và để thực hiện cơ chế attention, một mô hình attention sẽ đảm nhận giao tiếp trung gian giữa CNN và RNN.




## *4.2 Bộ mã hóa (Encoder)*

Mạng neural truy hồi (Recurrent Neural Network, viết tắt là RNN) được phát minh bởi John Hopfield năm 1982 [2]. Trong khoảng 5-6 năm gần đây, RNN được ứng dụng rộng rãi trong ngành NLP và thu được những thành tựu lớn. Mạng RNN mô hình hóa được bản chất của dữ liệu trong NLP (có đặc tính chuỗi và các thành phần như từ, cụm từ trong dữ liệu phụ thuộc lẫn nhau). Có thể nói việc áp dụng mạng RNN là một bước đột phá trong ngành NLP.

Mạng RNN nhận đầu vào là một chuỗi các vec-tơ x1, …, xn và trả ra đầu ra là một vector yn (số chiều của xi và yi không nhất thiết phải bằng nhau). Ví dụ một câu bao gồm nhiều từ, mỗi từ được biểu diễn bằng 1 vector (one-hot vec-tơ hoặc vec-tơ sinh ra bằng phương pháp word2vec). Quá trình sinh sử dụng một một đơn vị RNN, và đơn vị RNN này có bản chất là một hàm đệ quy để tính trạng thái đầu ra dựa vào trạng thái trước và đầu vào hiện tại. Mô tả toán học của mạng neural truy hồi như ở dưới đây:

$$y_n = RNN * (x_{(1;n)}; s_0)$$

$$y_i = 0(s_i)$$

$$s_i = R * (x_i; s_{i-1})$$

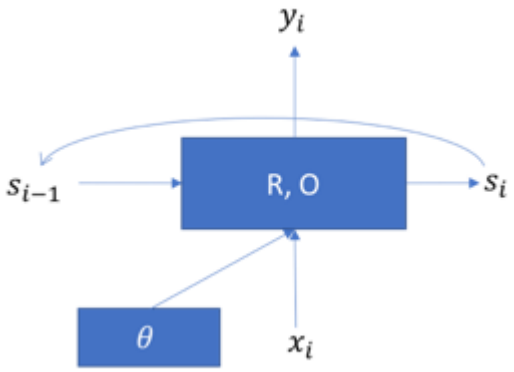



*Figure 16.Minh họa mạng RNN*

## 4.3 *Cơ chế Attention*

Cơ chế chú ý thị giác (Visual attention) được phát triển dựa trên ý tưởng từ thị giác con người. Cơ chế này sẽ không xử lý đồng thời toàn bộ thông tin của một bức ảnh, thay vào đó chỉ tập trung vào một phần được chọn nào đó của bức ảnh. Điều này giúp cho tác vụ tạo câu mô tả đạt được hiệu quả cao hơn, câu mô tả được chi tiết và dài hơn đối với những bức ảnh phức tạp về mặt nội dung.

$$V^l = CNN(X^{l-1}),$$

$$\gamma^l = \Phi(h_{t-1}, V^l),$$

$$X^l = f(V^l, \gamma^l)$$

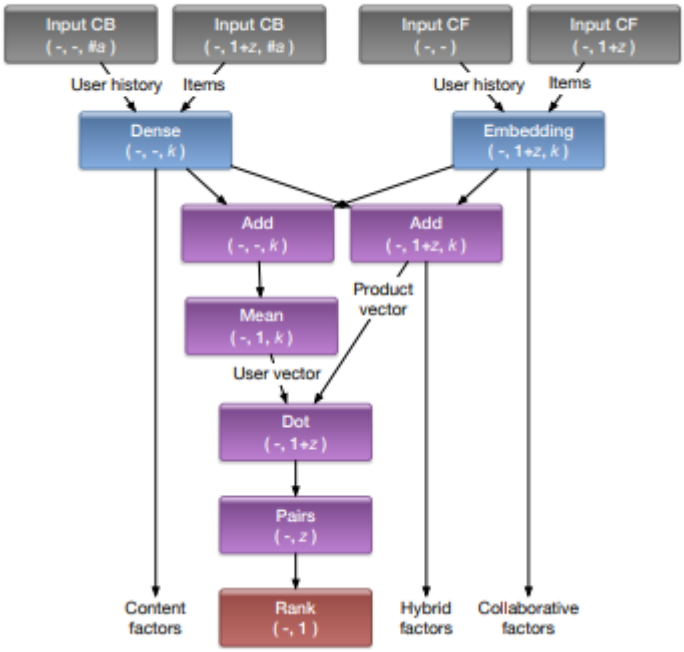



*Figure 17. Mô hình tổng thể cho bài toán phối trang phục sự kiện*

# 5. HIỆN THỰC VÀ ĐÁNH GIÁ MÔ HÌNH

## *5.1 Bộ dữ liệu Dataset*

Dataset là bộ gồm 28786 bộ ảnh. Mỗi bộ gồm 2 ảnh là áo và quần. Được dùng cho tập train và test trong quá trình máy học.


*Figure 18. Một bộ trang phục trong dataset*

## *5.2 Kết quả.*

Bài toán đưa ra được sự kiện mà bộ ảnh đã train được trong bộ dataset.

Một kết quả thu được cho trang phục dành cho ngày hè “summer daisy day”:

*Figure 19. Trang phục ngày hè*

# 6. Nguồn tham khảo

## *6.1 Sách online*


- Martín Abadi, Paul Barham, Jianmin Chen, Zhifeng Chen, Andy Davis, Jeffrey, Dean, Matthieu Devin, Sanjay Ghemawat, Geoffrey Irving, Michael Isard, et al. 2016. TensorFlow: A System for Large-Scale Machine Learning.. In OSDI, Vol. 16. 265–283.
- Xavier Amatriain. 2013. Mining Large Streams of User Data for Personalized Recommendations. SIGKDD Explor. Newsl. 14, 2 (April 2013), 37–48. https://doi.org/10.1145/2481244.2481250
- Christian Bracher, Sebastian Heinz, and Roland Vollgraf. 2016. Fashion DNA: Merging content and sales data for recommendation and article mapping. arXivpreprint arXiv:1609.02489 (2016)


## *6.2 Cách bài báo online*


[NN] Mạng nơ-ron nhân tạo - Neural Networks: https://dominhhai.github.io/vi/2018/04/nn-intro/

[ToMo] Giới Thiệu Về Deep Learning: https://ybox.vn/ky-nang/tomo-gioi-thieu-ve-deep-learning-ymh4xrcgrr