**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

**NGUYỄN ANH ĐẮC : 19133020**

**NGUYỄN THANH TÂN KỶ : 19133031**

**Đề Tài:**

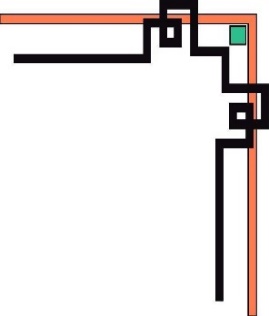
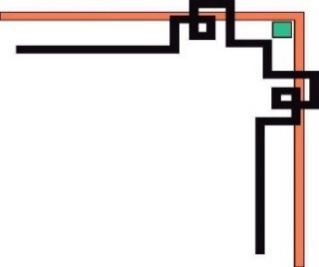
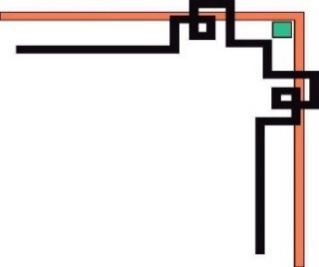
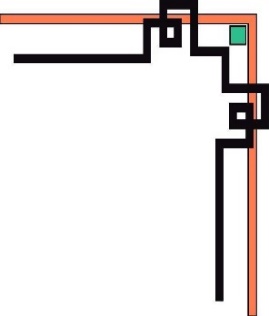
**TÌM HIỂU VỀ FEW – SHOT LEARNING VÀ ỨNG DỤNG**

**KHÓA LUẬN TỐT NGHIỆP**

GIÁO VIÊN HƯỚNG DẪN

**ThS. QUÁCH ĐÌNH HOÀNG**

**TP. HCM, ngày tháng năm 2023**



Logo, company name

Description automatically generated

**TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP. HCM**

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**NGUYỄN ANH ĐẮC: 19133020**

**NGUYỄN THANH TÂN KỶ: 19133031**

**Đề Tài:**

**TÌM HIỂU VỀ FEW – SHOT LEARNING VÀ ỨNG DỤNG**

**KHÓA LUẬN TỐT NGHIỆP**

GIÁO VIÊN HƯỚNG DẪN

**ThS. QUÁCH ĐÌNH HOÀNG**

**TP. HCM, ngày tháng năm 2023**

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

Họ và tên sinh viên 1: Nguyễn Anh Đắc MSSV: 19133020

Họ và tên sinh viên 2: Nguyễn Thanh Tân Kỷ MSSV: 19133031

Ngành: Kỹ thuật dữ liệu

Tên đề tài: Tìm hiểu về few – shot learning và ứng dụng.

Họ và tên giảng viên hướng dẫn: ThS.Quách Đình Hoàng

NHẬN XÉT:

1. Về nội dung và đề tài khối lượng thực hiện:
2. Ưu điểm:
3. Khuyết điểm:
4. Đề nghị cho bảo vệ hay không?
5. Đánh giá loại:
6. Điểm:

*Tp. Hồ Chí Minh, ngày...tháng...năm 2023*

Giảng viên hướng dẫn

*Ký & ghi rõ họ tên*

**PHIẾU NHẬN XÉT CỦA GIẢNG VIÊN PHẢN BIỆN**

Họ và tên sinh viên 1: Nguyễn Anh Đắc MSSV: 19133020

Họ và tên sinh viên 2: Nguyễn Thanh Tân Kỷ MSSV: 19133031

Ngành: Kỹ thuật dữ liệu

Tên đề tài: Tìm hiểu về few – shot learning và ứng dụng.

Họ và tên giảng viên phản biện:

NHẬN XÉT:

1. Về nội dung và đề tài khối lượng thực hiện:
2. Ưu điểm:
3. Khuyết điểm:
4. Đề nghị cho bảo vệ hay không?
5. Đánh giá loại:
6. Điểm:

*Tp. Hồ Chí Minh, ngày...tháng...năm 2023*

Giảng viên phản biện

*Ký & ghi rõ họ tên*

# LỜI CẢM ƠN

Trong quá trình nghiên cứu đề tài, giảng viên đã luôn hỗ trợ, hướng dẫn sinh viên một cách hết sức là nhiệt tình và chu đáo. Với tất cả sự kính trọng, chúng tôi xin được bày tỏ lòng biết ơn sâu sắc đến thầy, cô vì đã luôn theo dõi và hướng dẫn trong suốt thời gian thực hiện đề tài.

Đầu tiên, chúng tôi xin gửi lời cảm ơn sâu sắc nhất đến Ban giám hiệu trường Đại học Sư phạm Kỹ Thuật Thành phố Hồ Chí Minh đã tạo điều kiện, môi trường học tập, cơ sở vật chất chất lượng và hiệu quả để chúng tôi có thể phát huy một cách tốt nhất việc nghiên cứu đề tài.

Đồng thời, chúng tôi xin gửi lời cảm ơn đến Ban chủ nhiệm khoa Công nghệ Thông tin và các Thầy, Cô khoa Công nghệ Thông tin - Trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh đã tạo ra một môi trường học tập và làm việc chuyên nghiệp, nhiệt tình trong phương pháp giảng dạy để chúng tôi có thể thực hiện tốt đề tài nói riêng và sinh viên trong khoa Công nghệ Thông tin nói chung trong quá trình học tập và làm việc tại trường.

Đặc biệt, chúng tôi xin gửi lời cảm ơn chân thành nhất đến Thầy Quách Đình Hoàng – Giáo viên hướng dẫn khóa luận tốt nghiệp – Khoa Công nghệ Thông tin – Trường Đại học Sư phạm Kỹ thuật Thành phố Hồ Chí Minh đã hướng dẫn, quan tâm, góp ý và luôn đồng hành cùng chúng tôi trong những giai đoạn khó khăn nhất của việc nghiên cứu và thực hiện đề tài.

Cũng khó lòng tránh khỏi những sai sót và hạn chế nhất định trong việc thực hiện đề tài. Kính mong nhận được những phản hồi, đóng góp ý kiến và chỉ bảo thêm từ Quý Thầy Cô, để chúng tôi có thể tiếp thu những điều đó và từ đó đạt được những kiến thức hữu ích, nâng cao trình độ để phục vụ cho sự nghiệp sau này.

*Xin chân thành cảm ơn!*

# KẾ HOẠCH THỰC HIỆN

|  |  |  |  |
| --- | --- | --- | --- |
| Tuần | Thời gian | Nội dung công việc | Ghi chú |
| Tuần 1 – 2 | 30/1 – 12/2 | Lựa chọn và xác định đề tài khóa luận tốt nghiệp |  |
| Tuần 3 – 4 | 13/2 – 26/2 | Nghiên cứu sơ lược về đề tài và xây dựng khung sườn cho việc tìm hiểu. |  |
| Tuần 5 – 6 | 27/2 – 12/3 | Tìm hiểu khái niệm, cách tiếp cận cho bài toán few – shot learning |  |
| Tuần 7 | 13/3 – 19/3 | Tìm hiểu lý thuyết, các loại few – shot learning. |  |
| Tuần 8 | 20/3 – 26/3 | Tìm hiểu về thuật toán few – shot learning. |  |
| Tuần 9 – 11 | 27/3 – 16/4 | Tìm hiểu về thuật toán prototypical networks. |  |
| Tuần 12 | 17/4 – 23/4 | Tìm hiểu deep learning – CNN, Resnet – 18 |  |
| Tuần 13 | 24/4 – 30/4 | Tìm hiểu về các ứng dụng của few – shot learning trong lĩnh vực. |  |
| Tuần 14 | 1/5 – 7/5 | Nghỉ lễ |  |
| Tuần 15 | 8/5 – 14/5 | Tìm hiểu và nghiên một số triển khai ứng dụng của few – shot learning |  |
| Tuần 16 | 15/5 – 21/5 | Thực hiện xây dựng mô hình demo, hoàn thiện báo cáo |  |
| Tuần 17 – 18 | 22/5 – 4/6 | Thực hiện xây dựng mô hình demo, hoàn thiện báo cáo |  |
| Tuần 19 | 5/6 – 11/6 | Chỉnh sửa báo cáo, demo theo góp ý của giảng viên hướng dẫn |  |

# MỤC LỤC

[CHƯƠNG 1: MỞ ĐẦU 2](#_Toc139822679)

[1.1. TÍNH CẤP THIẾT CỦA ĐỀ TÀI 2](#_Toc139822680)

[1.2. MỤC TIÊU VÀ NHIỆM VỤ NGHIÊN CỨU 3](#_Toc139822681)

[1.3. CÁCH TIẾP CẬN VÀ PHƯƠNG PHÁP NGHIÊN CỨU 4](#_Toc139822682)

[1.4. KẾT QUẢ DỰ KIẾN ĐẠT ĐƯỢC 4](#_Toc139822683)

[CHƯƠNG 2: NỘI DUNG 5](#_Toc139822684)

[2.1. TỔNG QUAN VỀ FEW – SHOT LEARNING 5](#_Toc139822686)

[2.1.1. ĐỊNH NGHĨA 5](#_Toc139822687)

[2.1.2. SO SÁNH FSL VỚI CÁC PHƯƠNG PHÁP HỌC MÁY KHÁC 5](#_Toc139822688)

[2.1.3. TẦM QUAN TRỌNG CỦA FSL 6](#_Toc139822689)

[2.1.4. PHÂN LOẠI FSL 8](#_Toc139822690)

[2.1.5. ƯU, NHƯỢC ĐIỂM CỦA FSL 9](#_Toc139822691)

[2.2. TỔNG QUAN VỀ DEEP LEARNING 11](#_Toc139822692)

[2.2.1. DEEP LEARNING 11](#_Toc139822693)

[2.2.2. THUẬT TOÁN TRONG DEEP LEARNING 12](#_Toc139822694)

[2.3. CÁC PHƯƠNG PHÁP HỌC MÁY CỦA FSL 24](#_Toc139822695)

[2.3.1. ĐẶT VẤN ĐỀ 25](#_Toc139822696)

[2.3.2. META LEARNING 27](#_Toc139822697)

[CHƯƠNG 3: ỨNG DỤNG 52](#_Toc139822698)

[3.1. LĨNH VỰC ỨNG DỤNG 52](#_Toc139822702)

[3.1.1. THỊ GIÁC MÁY TÍNH 52](#_Toc139822703)

[3.1.2. XỬ LÝ NGÔN NGỮ TỰ NHIÊN 53](#_Toc139822704)

[3.1.3. PHÂN LOẠI ÂM THANH 54](#_Toc139822705)

[3.1.4. ROBOTICS 55](#_Toc139822706)

[3.1.5. CHĂM SÓC SỨC KHỎE 56](#_Toc139822707)

[3.2. TÌNH HÌNH PHÁT TRIỂN 58](#_Toc139822708)

[CHƯƠNG 4: TRIỂN KHAI ỨNG DỤNG 60](#_Toc139822709)

[4.1. MỤC TIÊU HƯỚNG TỚI 60](#_Toc139822711)

[4.2. DỮ LIỆU 60](#_Toc139822712)

[4.3. MÔ HÌNH VÀ THUẬT TOÁN 62](#_Toc139822713)

[4.4. TRIỂN KHAI ỨNG DỤNG 62](#_Toc139822714)

[4.5. KẾT QUẢ 64](#_Toc139822715)

[4.5.1. THỬ NGHIỆM MÔ HÌNH VỚI CÁC GIÁ TRỊ KHÁC NHAU 64](#_Toc139822716)

[4.5.2. ỨNG DỤNG WEB 66](#_Toc139822717)

[CHƯƠNG 5: KẾT LUẬN 70](#_Toc139822718)

[5.1. KẾT QUẢ ĐẠT ĐƯỢC 70](#_Toc139822720)

[5.1.1. Ý NGHĨA KHOA HỌC 70](#_Toc139822721)

[5.1.2. Ý NGHĨA THỰC TIỄN 70](#_Toc139822722)

[5.2. HẠN CHẾ 70](#_Toc139822723)

[5.3. HƯỚNG PHÁT TRIỂN 71](#_Toc139822724)

[TÀI LIỆU THAM KHẢO 72](#_Toc139822725)

# 

# DANH SÁCH HÌNH ẢNH

[Hình 1: Kiến trúc cơ bản của CNN [8] 13](#_Toc139821547)

[Hình 2: Lớp tích chập Conv [10] 15](#_Toc139821548)

[Hình 3: Phép biến đổi ReLU [11] 15](#_Toc139821549)

[Hình 4: Max pooling & average pooling trong lớp tổng hợp [10] 17](#_Toc139821550)

[Hình 5: Lớp liên kết đầy đủ. [11] 17](#_Toc139821551)

[Hình 6: Lớp hàm tổng quát và lớp hàm chồng nhau [13] 19](#_Toc139821552)

[Hình 7: Sự khác biệt giữa một khối thông thường (trái) và một khối phần dư (phải). [13] 20](#_Toc139821553)

[Hình 8: Khối ResNet thông thường (trái). Khối ResNet với tầng tích chập . [13] 21](#_Toc139821554)

[Hình 9: Kiến trúc GoogLeNet đầy đủ [13] 22](#_Toc139821555)

[Hình 10: Kiến trúc ResNet – 18. [13] 23](#_Toc139821556)

[Hình 11: Kiến trúc cơ bản của FSL [4] 25](#_Toc139821557)

[Hình 12: Kiến trúc của meta - learning [14] 28](#_Toc139821558)

[Hình 13: Kiến trúc cơ bản của gradient – based meta learning [4] 30](#_Toc139821559)

[Hình 14: Kiến trúc của siamese networks [15] 33](#_Toc139821560)

[Hình 15: Dữ liệu sau khi được biến dạng Affine [15] 34](#_Toc139821561)

[Hình 16: Kiến trúc tổng quan triplet networks 35](#_Toc139821562)

[Hình 17: Kiến trúc chi tiết của triplet networks [16] 36](#_Toc139821563)

[Hình 18: Kiến trúc của matching networks 38](#_Toc139821564)

[Hình 19: K – means clustering [18] 43](#_Toc139821565)

[Hình 20: Kiến trúc của prototypical networks [19] 44](#_Toc139821566)

[Hình 21: Kiến trúc relation networks 50](#_Toc139821567)

[Hình 22: Số lượng bài báo liên quan đến FSL được công bố trên các tạp chí uy tín từ năm 2010 đến nửa đầu năm 2021, không bao gồm các trích dẫn [3] 59](#_Toc139821568)

[Hình 23: Một số ký tự thuộc bộ ký tự Japanese\_(katakana) 61](#_Toc139821569)

[Hình 24: Một số ký tự thuộc bộ ký tự Latin 61](#_Toc139821570)

[Hình 25: Xử lý dữ liệu đầu vào [27] 62](#_Toc139821571)

[Hình 26: Mô hình prototypical networks[27] 63](#_Toc139821572)

[Hình 27: Thực hiện đào tạo mô hình [27] 64](#_Toc139821573)

[Hình 28: Lưu mô hình 64](#_Toc139821574)

[Hình 29: 5 – way 1 – shot. 64](#_Toc139821575)

[Hình 30: 5 – way 5 – shot. 65](#_Toc139821576)

[Hình 31: 5 – way 10 – shot. 65](#_Toc139821577)

[Hình 32: 10 – way 10 – shot. 65](#_Toc139821578)

[Hình 33: Biểu đồ độ chính xác theo số lần training 66](#_Toc139821579)

[Hình 34: Giao diện web 67](#_Toc139821580)

[Hình 35: Chọn một ký tự để nhận diện 67](#_Toc139821581)

[Hình 36: Nhận diện thành công 68](#_Toc139821582)

[Hình 37: Giao diện web cho phép người dùng vẽ ký tự. 68](#_Toc139821583)

[Hình 38: Kết quả phân loại ảnh từ ký tự người dùng vẽ. 69](#_Toc139821584)

# CHƯƠNG 1: MỞ ĐẦU

## TÍNH CẤP THIẾT CỦA ĐỀ TÀI

“Máy móc có thể suy nghĩ không?” Đây là câu hỏi được đặt ra trong bài báo quan trọng của Alan Turing có tựa đề “Máy tính và trí thông minh” vào năm 1950. Ông đã tuyên bố rằng “Ý tưởng đằng sau máy tính kỹ thuật số có thể được giải thích bằng cách nói rằng những chiếc máy này nhằm thực hiện bất kỳ hoạt động nào có thể xảy ra. được thực hiện bởi một máy tính của con người”. Nói cách khác, mục tiêu cuối cùng của máy móc là trở nên thông minh như con người. Trong những năm gần đây, do sự xuất hiện của các thiết bị điện toán mạnh mẽ (ví dụ: GPU và nền tảng phân tán), bộ dữ liệu lớn (ví dụ: dữ liệu ImageNet với 1000 lớp), các mô hình và thuật toán tiên tiến (ví dụ: mạng thần kinh tích chập (CNN)) và bộ nhớ ngắn hạn kéo dài (LSTM)), AI tăng tốc độ để giống con người và đánh bại con người trong nhiều lĩnh vực. Có thể kể đến một vài ví dụ, AlphaGo đánh bại các nhà vô địch loài người trong trò chơi cờ vây cổ đại; và kiến trúc mạng ResNet thu được hiệu suất phân loại tốt hơn so với con người trên ImageNet. AI cũng hỗ trợ phát triển các công cụ thông minh trong nhiều khía cạnh của cuộc sống hàng ngày, chẳng hạn như trợ lý giọng nói, công cụ tìm kiếm, ô tô lái tự động và rô-bốt công nghiệp.

Dù rất thành công, các kỹ thuật AI hiện tại không thể khái quát hóa nhanh chóng từ một vài ví dụ. Các ứng dụng AI thành công nói trên dựa vào việc học hỏi từ dữ liệu quy mô lớn. Ngược lại, con người có khả năng học các nhiệm vụ mới một cách nhanh chóng bằng cách sử dụng những gì họ đã học được trong quá khứ. Ví dụ, một đứa trẻ đã học cách cộng có thể nhanh chóng chuyển kiến thức của mình sang học phép nhân với một vài ví dụ (ví dụ: 2 × 3 = 2 + 2 + 2 và 1 × 3 = 1 + 1 + 1). Một ví dụ khác là khi được cung cấp một vài bức ảnh của một người lạ, một đứa trẻ có thể dễ dàng nhận ra cùng một người từ một số lượng lớn ảnh.

Thu hẹp khoảng cách này giữa AI và con người là một hướng quan trọng. Nó có thể được giải quyết bằng học máy, liên quan đến câu hỏi làm thế nào để xây dựng các chương trình máy tính tự động cải thiện theo kinh nghiệm. Để học hỏi từ một số ví dụ hạn chế với thông tin được giám sát, một mô hình học máy mới có tên là few – shot learning (FSL) được đề xuất. Một ví dụ điển hình là tạo ký tự, trong đó các chương trình máy tính được yêu cầu phân tích cú pháp và tạo ra các ký tự viết tay mới từ một vài ví dụ. Để xử lý tác vụ này, người ta có thể phân tách các ký tự thành các phần nhỏ hơn có thể chuyển qua các ký tự, sau đó tổng hợp các thành phần nhỏ hơn này thành các ký tự mới. Đây là cách học như người. Đương nhiên, FSL cũng có thể cải tiến công nghệ người máy, công ty phát triển các loại máy có thể tái tạo hành động của con người. Các ví dụ bao gồm bắt chước hành động bằng một lần cử chỉ, điều hướng bằng hình ảnh và điều khiển liên tục.

Một kịch bản FSL cổ điển khác khi mà bây giờ việc tìm kiếm dữ liệu sẽ càng khó khăn do các vấn đề về quyền riêng tư, an toàn hoặc đạo đức. Một ví dụ điển hình là khám phá về lĩnh vực thuốc, cố gắng khám phá các đặc tính của các phân tử mới để xác định những phân tử hữu ích làm thuốc mới. Việc các nhà nghiên cứu muốn tạo ra một loại thuốc mới có thể sẽ rất không an toàn vì có quá ít dữ liệu. Do đó, điều quan trọng là học hiệu quả từ một số ít mẫu. Thông qua FSL, việc học các mô hình phù hợp cho những trường hợp hiếm gặp này có thể trở nên khả thi.

FSL cũng có thể giúp giảm bớt gánh nặng thu thập dữ liệu được giám sát quy mô lớn. Ví dụ: mặc dù ResNet hoạt động tốt hơn con người trên ImageNet, nhưng mỗi lớp cần có đủ hình ảnh được gắn nhãn, điều này có thể tốn nhiều công sức để thu thập. FSL có thể giảm nỗ lực thu thập dữ liệu cho các ứng dụng sử dụng nhiều dữ liệu. Các ví dụ bao gồm phân loại hình ảnh, truy xuất hình ảnh, theo dõi đối tượng, nhận dạng cử chỉ, chú thích hình ảnh, trả lời câu hỏi trực quan, phát hiện sự kiện video, lập mô hình ngôn ngữ và tìm kiếm kiến trúc thần kinh.

Được thúc đẩy bởi mục tiêu học thuật để AI tiếp cận con người và nhu cầu công nghiệp về học tập không tốn kém, FSL gần đây đã thu hút nhiều sự chú ý và hiện là một chủ đề nóng. Nhiều phương pháp học máy liên quan đã được đề xuất, chẳng hạn như meta – learning, embedding learning và generative modeling. [1]

## MỤC TIÊU VÀ NHIỆM VỤ NGHIÊN CỨU

Mục tiêu của đề tài là tập trung nghiên cứu về cơ sở lý thuyết của bài toán FSL các ứng dụng liên quan đến thuật toán và từ đó khai thác chiều sâu của bài toán cũng như những thuật toán được sử dụng trong bài toán đó. Trong đề tài này, chúng tôi muốn xây dựng một mô hình phân tích để có thể phân tích và nhận dạng được các chữ viết bằng phương pháp meta – learning, prototypical networks. Để đạt được những điều đó, điều đầu tiên chúng tôi cần tìm hiểu một số vấn đề liên quan đến bài toán đã đặt ra như sau:

* Tìm hiểu cơ sở lý thuyết của bài toán FSL.
* Tìm hiểu các phương pháp tiếp cận cho những bài toán cụ thể.
* Ứng dụng bài toán vào một tập dữ liệu cụ thể để trực quan hóa bài toán.
* Đánh giá và giải thích kết quả đạt được.

## CÁCH TIẾP CẬN VÀ PHƯƠNG PHÁP NGHIÊN CỨU

Bài toán phân tích và nhận dạng chữ viết của 50 loại ngôn ngữ khác nhau trong tập dữ liệu Omniglot thì cách tiếp cận phổ biến nhất là sử dụng mô hình học sâu. Với các thuật toán học máy có giảm sát như decision trees, logistic regression, CNN, ... ở hướng tiếp cận các chữ viết, thì đây chính là một bài toán dạng phân loại và thuật toán prototypical networks mà chúng tôi tiếp cận sử dụng không quá nhiều dữ liệu nên việc có thể ra một độ chính xác cao với từng đó dữ liệu chính là vấn đề khá quan trọng.

## KẾT QUẢ DỰ KIẾN ĐẠT ĐƯỢC

Nhóm chúng tôi mong muốn sau khi thực hiện quá trình nghiên cứu nhiều công trình cũng như các ứng dụng từ các tác giả đi trước, nhóm có thể học hỏi và đúc kết thành một bài báo cáo chi tiết về nội dung lý thuyết của FSL và triển khai vào bài toán nhận dạng chữ viết, các cách phân tích, xử lý cho bài toán.

Về phần ứng dụng, để trực quan hóa bài toán nhóm chúng tôi sẽ xây dựng một mô hình đơn giản để trực quan hóa kết quả sau khi phân tích từ tập dữ liệu nhằm có cái nhìn cụ thể hơn cũng như thấy được sự hữu ích khi áp dụng FSL vào thực tế và từ đó đưa một cách kết luận khả quan nhất về thuật toán này.

# CHƯƠNG 2: NỘI DUNG



## TỔNG QUAN VỀ FEW – SHOT LEARNING

### ĐỊNH NGHĨA

Few – shot learning (FSL – học từ một vài ví dụ) là một phương pháp học máy cho phép mô hình học cách nhận diện và phân loại các đối tượng mới chỉ với một số ít dữ liệu huấn luyện. Điều này khác biệt với học máy truyền thống, nơi cần có một lượng lớn dữ liệu huấn luyện để mô hình có thể đạt được độ chính xác cao.

Các phương pháp FSL thường sử dụng các mô hình học sâu, như mạng nơ-ron tích chập (CNN) hoặc các mạng dựa trên kiến trúc transformer (transformer – based networks), để học cách tạo ra các đặc trưng chung từ các bộ dữ liệu huấn luyện khác nhau. Sau đó, các đặc trưng này được sử dụng để huấn luyện mô hình phân loại trên các tác vụ mới, mà mô hình chưa bao giờ được huấn luyện trước đó.

 FSL có thể được áp dụng trong nhiều lĩnh vực, chẳng hạn như phân loại ảnh, xử lý ngôn ngữ tự nhiên và robot học. Với sự phát triển của FSL, các ứng dụng như nhận dạng khuôn mặt và nhận diện đồ vật trên các thiết bị nhỏ gọn đã trở nên dễ dàng hơn.

Một trong những phương pháp tiếp cận phổ biến nhất của FSL được gọi là meta – learning hoặc learn – to – learn (học cách học). Trong phương pháp này, mô hình được huấn luyện trên một tập các tác vụ FSL, và học cách thích nghi nhanh với các tác vụ mới với một số ít ví dụ bằng cách học một tập các mẫu có thể tổng quát hóa trên các tác vụ khác nhau. Một phương pháp khác được gọi là học dựa trên độ đo, trong đó mô hình học một độ đo khoảng cách giữa các ví dụ và sử dụng độ đo này để phân loại các ví dụ mới với số lượng ít ví dụ. Tổng quát lại, FSL là một phương pháp hứa hẹn cho phép máy học với ít dữ liệu hơn, và có tiềm năng làm cho học máy trở nên dễ tiếp cận hơn trong các lĩnh vực nơi thu thập lượng lớn dữ liệu huấn luyện là khó khăn và tốn kém.

Một ví dụ về FSL, giả sử chúng ta làm việc trong lĩnh vực chăm sóc sức khỏe và gặp khó khăn trong việc phân loại các bệnh về xương thông qua ảnh chụp X – quang. Một số bệnh lý hiếm gặp có thể thiếu hình ảnh để sử dụng trong tập huấn luyện. Đây là loại vấn đề có thể được giải quyết bằng cách xây dựng bộ phân loại FSL. [2]

### SO SÁNH FSL VỚI CÁC PHƯƠNG PHÁP HỌC MÁY KHÁC

Theo truyền thống, việc phát triển các hệ thống máy học liên quan đến việc thu thập một lượng lớn dữ liệu và đào tạo các thuật toán machine learning (ML) trên đó để tạo ra kết quả. Để huấn luyện một mô hình để phân loại các đối tượng, ta cần một lượng lớn dữ liệu huấn luyện. ML sử dụng bộ dữ liệu quy mô lớn làm đầu vào. Đánh giá của nó về một mẫu mới dựa trên kết quả thống kê được trích xuất bởi dữ liệu lịch sử. Tuy nhiên, việc thu thập, dán nhãn và xác thực dữ liệu lớn rất tốn kém. Có nhiều trường hợp, ta không có quyền truy cập vào bộ dữ liệu lớn và phải dựa vào một vài ví dụ để đưa ra kết quả. Hay sẽ thật khó chịu nếu smartphone cần phải có hàng nghìn bức ảnh của người dùng để nhận diện và mở khóa. Giờ đây, sự bùng nổ của Internet (gần nhất là 5G) cung cấp khả năng kết nối khổng lồ cho hàng triệu thiết bị đầu cuối, cho phép kết nối mọi thứ với nhau. Tổng lượng dữ liệu do các thiết bị đầu cuối tạo ra là rất lớn, nhưng số lượng của một tập dữ liệu đơn lẻ lại cực kỳ nhỏ. Do đó, ML truyền thống, có hiệu suất phụ thuộc nhiều vào tập dữ liệu lớn, không thể hoạt động tốt trong cài đặt này với ít mẫu. Cuối cùng, FSL xuất hiện và cung cấp một cách đầy hứa hẹn để xử lý tình huống khan hiếm dữ liệu. [3]

FSL giải quyết vấn đề này bằng cách sử dụng một số lượng nhỏ các mẫu huấn luyện để xây dựng một mô hình phân loại tốt. Ý tưởng của FSL là learn – to – learn, tức là dùng một mô hình máy học để học cách áp dụng những kiến thức và kinh nghiệm đã học được từ các bài toán tương tự để giải quyết các bài toán mới. Các mô hình FSL thường sử dụng các kỹ thuật học tăng cường (reinforcement learning), học không giám sát (unsupervised learning) hoặc học chuyển tiếp (transfer learning) để giúp mô hình học được các khái niệm và tính chất mới một cách hiệu quả. Sự khác biệt đáng chú ý nhất giữa FSL và học máy truyền thống là tập hợp các lớp của tập hỗ trợ (support set) và tập truy vấn (query set) không liên kết với nhau. Trong máy học, các lớp của tập kiểm tra được đưa vào đào tạo trước. FSL kết hợp thông tin giám sát hạn chế với kiến thức có sẵn để đào tạo mô hình huấn luyện. Đầu vào của mô hình thường được đưa ra dưới dạng các nhiệm vụ. Thông qua công việc thu thập liên tục các tác vụ, mô hình có thể nhận ra sự giống nhau và khác nhau giữa dữ liệu cũng như tác vụ. Khi mô hình gặp phải một nhiệm vụ không nhìn thấy được, việc chuyển giao kiến thức thức có thể được thực hiện nhanh chóng chỉ với một vài bước đào tạo lặp đi lặp lại với các tham số khởi động phù hợp. Ngược lại, học máy truyền thống yêu cầu tối ưu hóa thông số thông qua hàm mất mát được tạo ra bởi một bộ dữ liệu quy định lớn trong mô hình. Tóm lại, FSL chỉ là một nhánh rất non trẻ của học máy, chủ yếu giải quyết vấn đề khó truy cập vào tài liệu chất lượng trong các vấn đề về học máy.

### TẦM QUAN TRỌNG CỦA FSL

Năm 1980, Kunihiko Fukushima đã phát triển mạng lưới thần kinh tích chập (CNN) đầu tiên. Kể từ đó, nhờ khả năng tính toán ngày càng tăng và những nỗ lực to lớn từ cộng đồng máy học, các thuật toán học sâu đã không ngừng cải thiện hiệu suất của chúng đối với các nhiệm vụ liên quan đến thị giác máy tính. Vào năm 2015, Kaiming He và nhóm của anh ấy tại Microsoft đã báo cáo rằng mô hình của họ hoạt động tốt hơn con người trong việc phân loại hình ảnh từ ImageNet. Vào thời điểm đó, người ta có thể lập luận rằng máy tính đã trở nên giỏi hơn chúng ta trong việc khai thác hàng tỷ hình ảnh để giải quyết một nhiệm vụ cụ thể.

Tuy nhiên, không phải lúc nào chúng ta cũng có thể xây dựng bộ dữ liệu với nhiều hình ảnh như vậy. Khi làm việc trong lĩnh vực computer vision (thị giác máy tính), đôi khi chúng ta phải phân loại hình ảnh chỉ với một hoặc hai mẫu cho mỗi nhãn. Cho trẻ em xem hình ảnh con voi và từ giờ trở đi chúng sẽ không bao giờ quên nhận ra con voi. Nếu chúng ta làm điều tương tự với resnet – 50, chúng ta có thể thất vọng với kết quả của nó. Vấn đề như thế này, học từ một vài ví dụ được gọi là FSL.

Với tầm quan trọng của FSL như vậy, chúng ta có thể thấy được một vài ưu điểm của nó như sau:

Cơ sở thử nghiệm để học như con người: Con người có thể nhận ra sự khác biệt giữa các ký tự viết tay sau khi xem một vài ví dụ. Tuy nhiên, máy tính cần một lượng lớn dữ liệu để phân loại những gì chúng “thấy” và phát hiện sự khác biệt giữa các ký tự viết tay. FSL là một cơ sở thử nghiệm trong đó máy tính được mong đợi sẽ học từ một vài ví dụ như con người.

Học cho các trường hợp hiếm gặp: Bằng cách sử dụng phương pháp học ít lần, máy có thể học các trường hợp hiếm gặp. Ví dụ: khi phân loại hình ảnh về động vật, một mô hình máy học được đào tạo bằng kỹ thuật học ít ảnh có thể phân loại chính xác hình ảnh của một loài quý hiếm sau khi tiếp xúc với một lượng nhỏ thông tin trước đó.

Giảm nỗ lực thu thập dữ liệu và chi phí tính toán: Vì phương pháp FSL yêu cầu ít dữ liệu hơn để đào tạo một mô hình, chi phí cao liên quan đến việc thu thập và ghi nhãn dữ liệu sẽ bị loại bỏ. Lượng dữ liệu đào tạo thấp có nghĩa là số chiều thấp trong tập dữ liệu đào tạo, điều này có thể làm giảm đáng kể chi phí tính toán.

Trong vài năm nay, vấn đề học vài lần đã thu hút rất nhiều sự chú ý trong cộng đồng nghiên cứu và rất nhiều giải pháp hay đã được phát triển. Các giải pháp phổ biến nhất hiện nay sử dụng phương pháp meta – learning, hay nói một cách ngắn gọn: học cách học (learn to learn).

### PHÂN LOẠI FSL

Các nhà nghiên cứu đã phân FSL thành các loại: N – shot learning (NSL), One – shot learning (OSL), zero – shot learning (ZSL).

Khi nói về FSL, chúng ta thường nghĩ đến N – way K – shot: N là số class, K là số lượng mẫu từ mỗi class dùng để training.

N – shot learning (NSL) được xem như là một khái niệm tổng quát hóa. Few – shot, one – shot, zero – shot learning là các trường hợp đặc biệt của NSL. [4]

#### N – SHOT LEARNING (NSL)

N – shot learning là một phương pháp học máy được sử dụng để giải quyết vấn đề học có giám sát với số lượng dữ liệu huấn luyện nhỏ. Trong NSL, mô hình được huấn luyện với một tập hợp nhỏ các mẫu huấn luyện và sau đó được đánh giá với một tập hợp lớn các mẫu kiểm tra.

Trong NSL, "N" thường được đặt là một số nguyên dương, đại diện cho số lượng các mẫu huấn luyện được sử dụng để huấn luyện mô hình. Ví dụ, nếu N = 5, thì mô hình sẽ được huấn luyện với 5 mẫu huấn luyện và sau đó được đánh giá trên tập kiểm tra. NSL được sử dụng trong các ứng dụng nhận dạng đối tượng, nhận dạng khuôn mặt và xử lý ngôn ngữ tự nhiên. Các mô hình NSL thường sử dụng các kỹ thuật transfer learning và meta – learning để tăng cường khả năng tổng quát hóa của mô hình. [4]

#### ONE – SHOT LEARNING (OSL)

OSL chỉ có một nhãn chính xác cho mỗi mẫu trong tập dữ liệu hỗ trợ, nhằm mục đích tìm ra lớp tương tự nhất dưới dạng đối sánh giữa các lớp đã nhìn thấy. Ví dụ trong quá trình thẩm vấn của cảnh sát, hai quá trình này giống nhau đến khó tin. Nhân chứng chỉ nhìn nghi phạm một lần và những bức ảnh do cảnh sát đưa ra có thể được coi là hình ảnh truy vấn. Nhân chứng chỉ cần trả lời 'có' hoặc 'không' đối với những bức ảnh đó. Tương tự, OSL không phân loại dữ liệu một cách cụ thể mà chỉ đơn giản là tạo một cụm theo thứ tự chức năng tương tự. Theo công việc hiện có, OSL có thể được chia thành hai cách tiếp cận chính. Một là sử dụng các mô hình tổng quát để loại bỏ kiến thức có sẵn, trong đó việc học lập trình bayesian là khuôn khổ tiêu biểu nhất trong lĩnh vực này. Một phương pháp khác là chuyển đổi tác vụ phân loại OSL thành tác vụ xác minh.

Trong OSL, chúng ta chỉ có một ví dụ duy nhất cho mỗi lớp. Bây giờ, nhiệm vụ là phân loại bất kỳ hình ảnh thử nghiệm nào thành một lớp bằng cách sử dụng ràng buộc đó. Có nhiều kiến ​​trúc khác nhau được phát triển để đạt được mục tiêu này, chẳng hạn như siamese neural networks, đã mang lại tiến bộ lớn và dẫn đến kết quả đặc biệt, sau đó là matching networks.

Nhiều vấn đề quan tâm yêu cầu suy luận nhanh chóng từ một lượng nhỏ dữ liệu. Trong giới hạn của “học từ một ví dụ”, các quan sát đơn lẻ sẽ dẫn đến những thay đổi đột ngột trong hành vi. [4]

#### ZERO – SHOT LEARNING (ZSL)

Mục tiêu của ZSL là khả năng học và phân loại các đối tượng mới mà mô hình chưa được huấn luyện trước đó. Chúng ta có thể tưởng tượng như là có thể phát hiện một object mà thậm chí không cần nhìn thấy nó không? Nếu chúng ta có những khái niệm về một object như là diện mạo, thuộc tính và chức năng của nó, thì đó không phải là vấn đề với con người. ZSL lần đầu tiên được đề xuất bởi Lampert và các cộng sự, xem xét một trường hợp cực đoan hơn trong FSL. Trong trường hợp không có bất kỳ mẫu truy vấn nào, cơ chế suy luận chỉ dựa vào việc xác định các mẫu chưa từng thấy trước đây. ZSL về cơ bản được thực hiện bằng cách sử dụng các thuộc tính đặc trưng high – dimensional để thay thế dữ liệu thô low – dimensional. Nhúng các biểu diễn và bộ mã hóa tự động là những cách hiệu quả nhất để xây dựng các không gian đặc trưng trung gian, chứa các thuộc tính xác định phân loại một cách toàn diện hơn. Cho đến nay, ZSL là một trong những phương pháp gần gũi nhất với trí thông minh của con người giúp phân biệt các phân loại trước đây không được quan sát. OSL và FSL về cơ bản có thể được coi là ZSL đặc biệt. [4]

### ƯU, NHƯỢC ĐIỂM CỦA FSL

#### ƯU ĐIỂM

Few-shot learning có một số ưu điểm quan trọng, bao gồm:

**Tính linh hoạt:** FSL cho phép mô hình học máy tổng quát hóa và áp dụng kiến thức đã học cho các nhiệm vụ mới chỉ với một số lượng nhỏ các mẫu huấn luyện. Điều này tạo ra tính linh hoạt cao, cho phép hệ thống thích ứng nhanh chóng với các tình huống mới và không cần phải thu thập một tập dữ liệu huấn luyện lớn.

**Tiết kiệm dữ liệu:** FSL giúp tiết kiệm công sức và thời gian cần thiết để thu thập và gán nhãn dữ liệu huấn luyện lớn. Thay vì yêu cầu một số lượng lớn các mẫu huấn luyện, FSL chỉ đòi hỏi một số lượng nhỏ các mẫu để học và tổng quát hóa.

**Khả năng học từ ít dữ liệu:** Với FSL, mô hình học máy có khả năng học từ ít dữ liệu. Thậm chí khi chỉ có một vài mẫu huấn luyện, mô hình có thể học được các đặc trưng quan trọng và áp dụng chúng cho các mẫu mới.

**Áp dụng trong các tình huống thực tế:** FSL có ý nghĩa đặc biệt trong các tình huống thực tế khi không có sẵn một tập dữ liệu huấn luyện lớn hoặc khi cần phải xử lý nhiều nhiệm vụ khác nhau chỉ với một số lượng nhỏ các mẫu huấn luyện.

**Giải quyết vấn đề đa dạng và thay đổi:** FSL cung cấp khả năng tổng quát hóa tốt và khả năng thích ứng trong môi trường đa dạng và thay đổi, nơi mô hình cần phải nhận biết và phân loại các đối tượng mới mà không có sẵn dữ liệu huấn luyện đầy đủ.

#### NHƯỢC ĐIỂM

Giai đoạn này, vẫn còn nhiều thách thức trong FSL, được tạo ra từ nhiều khía cạnh khác nhau. Trong bối cảnh này, các thách thức có thể được tóm tắt chung theo mức độ tích hợp kiến thức như sau:

**Đánh giá phân phối dữ liệu không chính xác:** FSL không có quyền truy cập vào các bộ dữ liệu lớn do chi phí, đạo đức, pháp lý hoặc các lý do khác. Do đó, việc dựa vào một vài mẫu để học sẽ tạo ra sai lệch trong việc ước tính phân phối dữ liệu thực tế, điều này có thể gây ảnh hưởng xấu cho một số tác vụ. Cuối cùng, sự tối đa hóa việc khám phá các phân phối dữ liệu với thông tin hạn chế trở thành thách thức quan trọng nhất đối với FSL. Tăng cường dữ liệu là cách trực tiếp để giải quyết ước tính không chính xác của FSL. Những nỗ lực chính hiện tập trung vào việc khám phá các đặc trưng (feature) có thể di chuyển giữa các lớp và tùy chỉnh các hình ảnh cụ thể bằng trình tạo.

**Khả năng tái sử dụng lại đặc trưng (feature reusability):** Tích lũy liên tục kiến thức tiên nghiệm bằng cách lấy mẫu các bộ dữ liệu hỗ trợ quy mô lớn. Học chuyển giao có thể dễ dàng sử dụng nó từ miền nguồn sang miền đích tương tự. Đào tạo trước nhằm mục đích trích xuất các vectơ đặc trưng cao thông qua trình trích xuất đặc trưng, trong khi mục tiêu của tinh chỉnh là thực hiện các điều chỉnh nhỏ đối với các tham số ban đầu của đào tạo trước. Học chuyển giao tập trung vào cấp độ dữ liệu và thu được nhiều đặc trưng có giá trị hơn độc lập với tác vụ bằng cách ánh xạ dữ liệu tới các nhãn. Nó có hiệu suất vượt trội trong việc tối ưu hóa các nhiệm vụ cụ thể, nhưng nó thường bị hạn chế bởi các đặc điểm của các nhiệm vụ hiện tại và có khả năng khái quát hóa kém đối với các nhiệm vụ trong tương lai. Đặc biệt khi có sự thay đổi lớn trong miền, nếu không lọc và căn chỉnh các đặc trưng có thể dẫn đến việc chuyển giao tri thức xảy ra không được chính xác.

**Tính tổng quát của các nhiệm vụ trong tương lai:** Khác với học chuyển đổi, meta – learning học cách nhanh chóng xây dựng ánh xạ từ các nhiệm vụ đã biết đến các mô hình đích trong các nhiệm vụ chưa từng thấy trước đây bằng cách lấy mẫu kép nhiệm vụ và dữ liệu. Trong FSL, bằng cách khám phá không gian nhiệm vụ, tóm tắt siêu kiến thức trong các nhiệm vụ khác nhau có thể dẫn đến việc tổng hợp nhanh các nhiệm vụ chưa từng thấy với chi phí thấp hơn. Là một khuôn khổ học tập chung, meta – learning độc lập với các vấn đề cụ thể và hướng nhiều hơn đến các nhiệm vụ trong tương lai thay vì tối ưu hóa nhiệm vụ hiện tại. Tuy nhiên, meta – learning chỉ được chứng minh là có hiệu quả khi các nhiệm vụ kiểm tra và đào tạo tương đối giống nhau và nó phụ thuộc nhiều vào cấu trúc mạng và thiếu tính linh hoạt. Khi đào tạo meta – learners với một tập hợp các nhiệm vụ cùng một lúc, thậm chí rất khó thích ứng với việc phân phối các nhiệm vụ, đòi hỏi phải thiết kế lại cấu trúc mạng.

**Khiếm khuyết của thông tin đơn phương thức:** Rất khó để tìm hiểu các đặc trưng một cách hiệu quả vì FSL vốn đã bị hạn chế về thông tin. Tình trạng này được cải thiện rất nhiều khi được hỗ trợ bằng cách lấy thông tin từ các phương thức khác. Về mặt này, hỗ trợ ngữ nghĩa (semantic assistance) là một phương pháp tuyệt vời để cung cấp kiến thức bên ngoài, trong đó thông qua việc giới thiệu hoặc tạo thông tin ngữ nghĩa dưới dạng giám sát yếu, phân loại thích ứng có thể được thực hiện cùng với mô hình ban đầu.

## TỔNG QUAN VỀ DEEP LEARNING

### DEEP LEARNING

#### ĐỊNH NGHĨA

Deep learning (DL) là dựa trên nhánh của ML, đó là một phần con của trí tuệ nhân tạo. Vì mạng nơ – ron mô phỏng não người và deep learning cũng vậy. Trong DL, không có gì được lập trình một cách rõ ràng. Đơn giản, đó là một lớp học máy sử dụng nhiều đơn vị xử lý phi tuyến để thực hiện trích xuất và biến đổi đặc trưng. Đầu ra từ mỗi lớp trước đó được sử dụng làm đầu vào cho mỗi lớp tiếp theo.

Các mô hình DL có khả năng tập trung vào các đặc trưng chính xác mà không cần hướng dẫn nhiều từ người lập trình và rất hữu ích trong việc giải quyết vấn đề chiều dữ liệu. Các thuật toán DL được sử dụng đặc biệt khi chúng ta có một lượng lớn đầu vào và đầu ra.

Vì DL đã được phát triển bởi máy học, mà chính nó là một phần con của trí tuệ nhân tạo và ý tưởng đằng sau trí tuệ nhân tạo là mô phỏng hành vi của con người, vì vậy "ý tưởng của DL là xây dựng một thuật toán có thể mô phỏng não".

DL được thực hiện nhờ mạng nơ – ron, và ý tưởng sau sự truyền cảm hứng của mạng nơ-ron là các tế bào sinh học, đó là một tế bào não. [5]

#### CÁCH THỨC HOẠT ĐỘNG

Các mạng DL học bằng cách khám phá các cấu trúc phức tạp trong dữ liệu mà chúng tiếp xúc. Bằng cách xây dựng các mô hình tính toán được tạo thành từ nhiều lớp xử lý, các mạng có thể tạo ra nhiều cấp độ trừu tượng để đại diện cho dữ liệu.

Ví dụ, một mô hình DL được biết đến là mạng nơ – ron tích chập có thể được huấn luyện bằng cách sử dụng một số lượng lớn (ví dụ như hàng triệu) hình ảnh, chẳng hạn như những hình ảnh chứa hình mèo. Loại mạng nơ-ron này thường học từ các pixel có trong các hình ảnh mà nó nhận được. Nó có thể phân loại các nhóm pixel đại diện cho các đặc trưng của một con mèo, với các nhóm đặc trưng như móng vuốt, tai và mắt cho thấy sự hiện diện của một con mèo trong một hình ảnh.

DL khác biệt cơ bản so với ML truyền thống. Trong ví dụ này, một chuyên gia trong lĩnh vực cần phải dành thời gian đáng kể để thiết kế một hệ thống ML truyền thống để phát hiện các đặc trưng đại diện cho một con mèo. Với DL, chỉ cần cung cấp hệ thống một số lượng rất lớn các hình ảnh mèo, và hệ thống có thể tự động học các đặc trưng đại diện cho một con mèo.

Đối với nhiều nhiệm vụ như thị giác máy tính, nhận dạng giọng nói (còn được gọi là xử lý ngôn ngữ tự nhiên), dịch máy và robot, hiệu suất của các hệ thống DL vượt xa so với các hệ thống ML truyền thống. Điều này không có nghĩa là xây dựng các hệ thống DL dễ dàng hơn so với các hệ thống ML truyền thống. Mặc dù việc nhận dạng đặc trưng là tự động trong DL, hàng nghìn siêu tham số (núm điều chỉnh) cần được điều chỉnh để mô hình DL trở nên hiệu quả. [6]

### THUẬT TOÁN TRONG DEEP LEARNING

#### MẠNG NƠ - RON TÍCH CHẬP – CNN

##### ĐỊNH NGHĨA

Mạng nơ-ron tích chập (CNN) là một loại kiến ​​trúc mạng thần kinh Deep Learning thường được sử dụng trong thị giác máy tính. Thị giác máy tính là một lĩnh vực trí tuệ nhân tạo cho phép máy tính hiểu và diễn giải hình ảnh hoặc dữ liệu trực quan.

Khi nói đến ML, mạng nơ-ron tích chập hoạt động rất tốt. Mạng nơ – ron được sử dụng trong các bộ dữ liệu khác nhau như hình ảnh, âm thanh và văn bản. Các loại mạng nơ-ron khác nhau được sử dụng cho các mục đích khác nhau, ví dụ: để dự đoán chuỗi từ, ta có thể sử dụng mạng nơ-ron hồi tiếp (RNN – [recurrent neural networks](https://www.geeksforgeeks.org/introduction-to-recurrent-neural-network/)) chính xác hơn là bộ nhớ ngắn hạn kéo dài (LSTM – long short - term memory); tương tự, để phân loại hình ảnh thì chúng ta sẽ sử dụng mạng nơ-ron tích chập.

Trong mạng nơ-ron thông thường, có ba loại lớp:

Lớp đầu vào (input layer): Đó là lớp mà chúng tôi cung cấp đầu vào cho mô hình của mình. Số lượng tế bào thần kinh trong lớp này bằng với tổng số tính năng trong dữ liệu của chúng tôi (số pixel trong trường hợp hình ảnh).

Lớp ẩn (hidden layer): Đầu vào từ lớp đầu vào sau đó được đưa vào lớp ẩn. Có thể có nhiều lớp ẩn tùy thuộc vào mô hình và kích thước dữ liệu của chúng ta. Mỗi lớp ẩn có thể có số lượng tế bào thần kinh khác nhau, thường lớn hơn số lượng đặc trưng. Đầu ra từ mỗi lớp được tính bằng phép nhân ma trận của đầu ra của lớp trước đó với các trọng số có thể học được của lớp đó và sau đó bằng phép cộng các độ lệch có thể học được, sau đó là hàm kích hoạt làm cho mạng trở nên phi tuyến tính.

Lớp đầu ra (output layer): Đầu ra từ lớp ẩn sau đó được đưa vào một hàm logistic như sigmoid hoặc softmax để chuyển đổi đầu ra của mỗi lớp thành điểm xác suất của mỗi lớp. [7]

##### KIẾN TRÚC CỦA CNN

Mạng nơ-ron tích chập (CNN) bao gồm 3 lớp chính như: lớp tích chập (convolutional layer), lớp tổng hợp (pooling layer) và lớp kết nối đầy đủ (fully connected).

A picture containing sketch, diagram, line, drawing

Description automatically generated

Hình : Kiến trúc cơ bản của CNN [8]

Lớp tích chập là lớp đầu tiên của mạng tích chập. Mặc dù các lớp tích chập có thể được theo sau bởi các lớp tích chập bổ sung hoặc các lớp tổng hợp, nhưng lớp được kết nối đầy đủ là lớp cuối cùng. Với mỗi lớp, CNN tăng độ phức tạp của nó, xác định các phần lớn hơn của hình ảnh. Các lớp trước đó tập trung vào các tính năng đơn giản, chẳng hạn như màu sắc và các cạnh. Khi dữ liệu hình ảnh tiến triển qua các lớp của CNN, nó bắt đầu nhận ra các yếu tố hoặc hình dạng lớn hơn của đối tượng cho đến khi cuối cùng nó xác định được đối tượng dự định.

1. Lớp tích chập (Conv)

Lớp tích chập là thành phần chính của một mạng CNN và là nơi diễn ra hầu hết các phép tính toán. Nó yêu cầu một số thành phần như dữ liệu đầu vào, một bộ lọc và một feature map. Giả sử đầu vào là một hình ảnh màu, được tạo thành từ một ma trận các pixel trong không gian ba chiều. Điều này có nghĩa rằng đầu vào sẽ có ba chiều - chiều cao, chiều rộng và chiều sâu - tương ứng với RGB trong hình ảnh. Chúng ta cũng có một bộ phát hiện đặc trưng, còn được gọi là kernel hoặc bộ lọc, sẽ di chuyển qua các vùng nhận thức (receptive field) của hình ảnh để kiểm tra xem đặc trưng có tồn tại không. Quá trình này được gọi là phép tích chập. [9]

Bộ phát hiện đặc trưng là một mảng hai chiều (2 – D) chứa các trọng số, đại diện cho một phần của hình ảnh. Mặc dù kích thước có thể thay đổi, kích thước bộ lọc thường là một ma trận ; điều này cũng xác định kích thước của vùng nhận thức. Sau đó, bộ lọc được áp dụng vào một vùng của hình ảnh và tính toán tích vô hướng giữa các pixel đầu vào và bộ lọc. Tích vô hướng này được đưa vào một mảng đầu ra. Sau đó, bộ lọc dịch chuyển theo một bước, lặp lại quá trình cho đến khi bộ lọc đã quét qua toàn bộ hình ảnh. Kết quả cuối cùng từ chuỗi tích vô hướng giữa đầu vào và bộ lọc được gọi là feature map, activation map hoặc convolved feature. Chúng ta có thể thấy cách thức biến đổi của lớp tích chập qua hình 2. [9]

A picture containing text, diagram, line, plan

Description automatically generated

Hình : Lớp tích chập Conv [10]

Sau mỗi phép tích chập, một CNN áp dụng phép biến đổi ReLU cho feature map, nó sẽ làm tăng tính phi tuyến tính vào mô hình. Có những công thức ReLu được tổng hợp ở hình 3 như sau:

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình : Phép biến đổi ReLU [11]

Chúng ta cũng có thể thiết lập những thông số trước khi xây dựng các phép tích chập:

* **Deepth:** Độ sâu, tương ứng với số lượng bộ lọc hoặc kernel. Số lượng bộ lọc sẽ quyết định số lượng feature map.
* **Filter size:** Kích thước bộ lọc, được định nghĩa là chiều cao hay chiều rộng của bộ lọc.
* **Stride:** Đại diện cho số lượng pixel mà cửa sổ di chuyển sau mỗi phép tính. Nó sẽ quy định khoảng cách di chuyển của cửa sổ tích chập khi quét qua dữ liệu đầu vào. Nếu stride = 1 có nghĩa là cửa số tích chập sẽ di chuyển một pixel mỗi lần 🡪 sự chồng chéo mật độ các vùng tiếp nhận. Điều này cho phép phân tích chi tiết hơn của dữ liệu đầu vào nhưng tăng độ phức tạp tính toán. Trong khi đó, stride > 1, như 2 hoặc 3, có nghĩa là cửa sổ tích chập hoặc cửa sổ pooling bỏ qua một số pixel, dẫn đến việc giảm độ chồng chéo. Điều này dẫn đến việc giảm kích thước của feature map và có thể giúp giảm độ phức tạp tính toán trong khi vẫn giữ được thông tin quan trọng.
* **Zero – padding:** Đệm bằng số 0, mục đích là bảo tồn kích thước không gian của khối dữ liệu đầu vào và điều khiển kích thước không gian của khối dữ liệu đầu ra sau khi tích chập. Nó cho phép mạng áp dụng các phép tích chập cho các pixel biên của khối dữ liệu đầu vào, giúp giữ lại thông tin quan trọng từ cạnh của dữ liệu đầu vào. Zero – padding thường được áp dụng đối xứng vào khối dữ liệu đầu vào, thêm cùng một số hàng và cột số không vào mỗi phía. Số lượng đệm được xác định bởi kích thước đầu ra mong muốn và kích thước của kernel tích chập.

Như đã đề cập trước đó, một lớp tích chập khác có thể tiếp theo lớp tích chập ban đầu. Khi điều này xảy ra, cấu trúc của CNN có thể trở thành phân cấp vì các lớp sau có thể xem các pixel trong vùng nhận thức của các lớp trước. Ví dụ, giả sử chúng ta đang cố gắng xác định xem một hình ảnh có chứa một chiếc xe đạp hay không. Chúng ta có thể coi chiếc xe đạp là tổng của các bộ phận. Nó bao gồm khung, cốt yên, bánh xe, bàn đạp và nhiều phần khác. Mỗi phần riêng lẻ của xe đạp tạo thành một mẫu cấp thấp trong mạng nơ-ron và sự kết hợp của các phần này tạo thành một mẫu cấp cao, tạo ra một hệ thống phân cấp đặc trưng trong CNN.

1. Lớp tổng hợp (pooling layer)

Lớp pooling, còn được gọi là lớp giảm kích thước, thực hiện việc giảm số chiều của đầu vào, từ đó giảm số lượng tham số. Tương tự như lớp tích chập, phép toán pooling dịch một bộ lọc qua toàn bộ đầu vào, nhưng khác biệt là bộ lọc này không có trọng số. Thay vào đó, lõi pooling áp dụng một hàm tổng hợp cho các giá trị trong phạm vi nhận thức từ đó tạo ra mảng kết quả. Có hai loại chính của pooling:

* Max pooling: Khi bộ lọc di chuyển qua đầu vào, nó chọn pixel có giá trị lớn nhất để gửi đến mảng kết quả. Đây là phương pháp được sử dụng nhiều hơn so với average pooling.
* Average pooling: Khi bộ lọc di chuyển qua đầu vào, nó tính giá trị trung bình trong phạm vi nhận thức để gửi đến mảng kết quả.

Mặc dù rất nhiều thông tin bị mất trong lớp pooling, nhưng nó cũng mang lại nhiều lợi ích cho CNN. Chúng giúp giảm độ phức tạp, cải thiện hiệu suất và giới hạn nguy cơ quá khớp. [9]

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình : Max pooling & average pooling trong lớp tổng hợp [10]

1. Lớp liên kết đầy đủ (fully connected layer)

Lớp fully connected (fully connected layer) được mô tả rõ ràng bởi tên gọi của nó. Như đã đề cập trước đó, các giá trị pixel của ảnh đầu vào không được kết nối trực tiếp với lớp đầu ra trong các lớp kết nối một phần (partially connected layers). Tuy nhiên, trong lớp fully connected, mỗi nút trong lớp đầu ra kết nối trực tiếp với một nút trong lớp trước đó.

A picture containing sketch, diagram, line, screenshot

Description automatically generated

Hình : Lớp liên kết đầy đủ. [11]

Lớp này thực hiện công việc phân loại dựa trên các đặc trưng được trích xuất thông qua các lớp trước và các bộ lọc khác nhau. Trong khi các lớp tích chập (convolutional) và lớp gộp (pooling) thường sử dụng các hàm ReLu, lớp fully connected thường sử dụng hàm kích hoạt softmax để phân loại đầu vào một cách thích hợp, tạo ra xác suất từ 0 đến 1. [9]

#### MẠNG PHẦN DƯ – RESNET

##### ĐỊNH NGHĨA

ResNet (residual network) được giới thiệu đến công chúng vào năm 2015 và thậm chí đã giành được vị trí thứ 1 trong cuộc thi ILSVRC 2015 với tỉ lệ lỗi top 5 chỉ 3.57%. Không những thế nó còn đứng vị trí đầu tiên trong cuộc thi ILSVRC and COCO 2015 với ImageNet detection, ImageNet localization, Coco detection và Coco segmentation. Hiện tại thì có rất nhiều biến thể của kiến trúc ResNet với số lớp khác nhau như ResNet – 18, ResNet – 34, ResNet – 50, ResNet – 101, ResNet – 152, ...Với chỉ số theo sau chữ ResNet sẽ là chỉ số lớp nhất định của mô hình đó. [12]

Mạng ResNet là một mạng CNN được thiết kế để làm việc với hàng trăm lớp. Một vấn đề xảy ra khi xây dựng mạng CNN với nhiều lớp tích chập sẽ xảy ra hiện tượng vanishing gradient dẫn tới quá trình học tập không tốt. [12]

##### KIẾN TRÚC CỦA RESNET

###### CÁC LỚP HÀM SỐ

Coi *Ƒ* là một lớp các hàm mà một kiến trúc mạng cụ thể (cùng với tốc độ học và các siêu tham số khác) có thể đạt được. Nói cách khác, với mọi hàm số *f* ∈ *Ƒ*, luôn tồn tại một số tập tham số *W* có thể tìm được bằng việc huấn luyện trên một tập dữ liệu phù hợp. Giả sử  là hàm cần tìm. Sẽ rất thuận lợi nếu hàm này thuộc tập F, nhưng thường không may mắn như vậy. Thay vào đó, chúng ta sẽ cố gắng tìm các hàm số  tốt nhất có thể trong tập *Ƒ*.

Ví dụ, có thể thử tìm  bằng cách giải bài toán tối ưu sau:

Khá hợp lý khi giả sử rằng nếu thiết kế một kiến trúc khác *Ƒ’* mạnh mẽ hơn thì sẽ đạt được kết quả tốt hơn. Nói cách khác, chúng ta kỳ vọng hàm số  sẽ “tốt hơn” . Tuy nhiên, nếu , thì không khẳng định được  “tốt hơn” . Trên thực tế,  có thể còn tệ hơn. Và đây là trường hợp thường xuyên xảy ra — việc thêm các tầng không phải lúc nào cũng tăng tính biểu diễn của mạng mà đôi khi còn tạo ra những thay đổi rất khó lường. Hình 6 minh hoạ rõ hơn về sự khác biệt của hàm tổng quát và hàm chồng nhau như sau.

A picture containing sketch, circle, design

Description automatically generated

Hình : Lớp hàm tổng quát và lớp hàm chồng nhau [13]

Chỉ khi các lớp hàm lớn hơn chứa các lớp nhỏ hơn, thì mới đảm bảo rằng việc tăng thêm các tầng sẽ tăng khả năng biểu diễn của mạng. Đây là câu hỏi mà He và các cộng sự đã suy nghĩ khi nghiên cứu các mô hình thị giác sâu năm 2016. Ý tưởng trọng tâm của ResNet là mỗi tầng được thêm vào nên có một thành phần là hàm số đồng nhất. Điều này có nghĩa rằng, nếu ta huấn luyện tầng mới được thêm vào thành một ánh xạ đồng nhất  , thì mô hình mới sẽ hiệu quả ít nhất bằng mô hình ban đầu. Vì tầng được thêm vào có thể khớp dữ liệu huấn luyện tốt hơn, dẫn đến sai số huấn luyện cũng nhỏ hơn. Tốt hơn nữa, hàm số đồng nhất nên là hàm đơn giản nhất trong một tầng thay vì hàm null . [13]

###### CÁC KHỐI PHẦN DƯ

Bây giờ, hãy tập trung vào mạng nơ-ron dưới đây. Ký hiệu đầu vào là *x*. Giả sử ánh xạ lý tưởng muốn học được là , và được dùng làm đầu vào của hàm kích hoạt. Phần nằm trong viền nét đứt bên trái phải khớp trực tiếp với ánh xạ . Điều này có thể không đơn giản nếu chúng ta không cần khối đó và muốn giữ lại đầu vào x. Khi đó, phần nằm trong viền nét đứt bên phải chỉ cần tham số hoá độ lệch khỏi giá trị *x*, bởi vì chúng ta đã trả về . Trên thực tế, ánh xạ phần dư thường dễ tối ưu hơn, vì chỉ cần đặt . Nửa bên phải hình 7 mô tả khối phần dư (residual block) cơ bản của ResNet.

A picture containing text, screenshot, diagram, line

Description automatically generated

Hình : Sự khác biệt giữa một khối thông thường (trái) và một khối phần dư (phải). [13]

ResNet có thiết kế tầng tích chập  giống VGG (mạng sử dụng khối). Khối phần dư có hai tầng tích chập với cùng số kênh đầu ra. Mỗi tầng tích chập được theo sau bởi một tầng chuẩn hóa theo batch và một hàm kích hoạt ReLU. Ta đưa đầu vào qua khối phần dư rồi cộng với chính nó trước hàm kích hoạt ReLU cuối cùng. Thiết kế này đòi hỏi đầu ra của hai tầng tích chập phải có cùng kích thước với đầu vào, để có thể cộng lại với nhau. Nếu muốn thay đổi số lượng kênh hoặc stride trong khối phần dư, cần thêm một tầng tích chập  để thay đổi kích thước đầu vào tương ứng ở nhánh ngoài.

Ví dụ như minh họa ở hình 8: ta có một loại đầu tiên (trái) cộng “đầu vào” vào đầu ra trước khi áp dụng hàm phi tuyến ReLU, còn ở loại thứ hai (phải) chúng ta thay đổi số kênh và độ phân giải bằng một tầng tích chập  trước khi thực hiện phép cộng. [13]

A picture containing text, screenshot, diagram, parallel

Description automatically generated

Hình : Khối ResNet thông thường (trái). Khối ResNet với tầng tích chập . [13]

###### MÔ HÌNH CỦA RESNET

Hai tầng đầu tiên của ResNet giống hai tầng đầu tiên của GoogLeNet: tầng tích chập  với 64 kênh đầu ra và stride = 2, theo sau bởi tầng gộp cực đại  với stride = 2. Sự khác biệt là trong ResNet, mỗi tầng tích chập theo sau bởi tầng chuẩn hóa theo batch.

A picture containing text, screenshot, font, diagram

Description automatically generated

Hình : Kiến trúc GoogLeNet đầy đủ [13]

GoogLeNet (mạng nối song song) sử dụng bốn mô-đun được tạo thành từ các khối Inception (như hình 9). ResNet sử dụng bốn mô-đun được tạo thành từ các khối phần dư có cùng số kênh đầu ra. Mô-đun đầu tiên có số kênh bằng số kênh đầu vào. Vì trước đó đã sử dụng tầng gộp cực đại với stride = 2, nên không cần phải giảm chiều cao và chiều rộng ở mô-đun này. Trong các mô-đun sau, khối phần dư đầu tiên nhân đôi số kênh, đồng thời giảm một nửa chiều cao và chiều rộng.

A picture containing diagram, screenshot, text, line

Description automatically generated

Hình : Kiến trúc ResNet – 18. [13]

Với ResNet – 18 (hình 10): có 4 tầng tích chập trong mỗi mô-đun (không tính tầng tích chập  ). Cộng thêm tầng tích chập đầu tiên và tầng kết nối đầy đủ cuối cùng, mô hình có tổng cộng 18 tầng. Do đó, mô hình này thường được gọi là ResNet – 18. Có thể thay đổi số kênh và các khối phần dư trong mô-đun để tạo ra các mô hình ResNet khác nhau, ví dụ mô hình 152 tầng của ResNet – 152. Mặc dù có kiến trúc lõi tương tự như GoogLeNet, cấu trúc của ResNet đơn giản và dễ sửa đổi hơn. Tất cả các yếu tố này dẫn đến sự phổ cập nhanh chóng và rộng rãi của ResNet. [13]

## CÁC PHƯƠNG PHÁP HỌC MÁY CỦA FSL

Cùng hiểu rõ hơn N – way K – shot – classification. Hãy tưởng tượng rằng chúng ta có một bộ đào tạo (support set) bao gồm: N số lượng class, K số lượng labels cho từng lớp (số lượng ít, dưới mười mẫu / lớp), Q ảnh cần chạy detect.

Nhiệm vụ cần thực hiện là phân loại Q hình ảnh trong N lớp. Với số lượng data rất ít (N \* K) cho tập train, thì vấn đề ở đây là không đủ dữ liệu để training model.

Bước đầu tiên trong FSL là tích lũy kinh nghiệm từ các cách xử lý vấn đề tương tự. Đây là lý do tại sao FSL được mô tả như là meta – learning. Trong một bài toán truyền thống, chúng ta cố gắng học cách phân loại từ training dataset và đánh giá bằng cách sử dụng testing dataset. Trong **meta – learning**, chúng ta “learn how to learn” từ một training dataset, dataset này là tập hợp các class không phải là các class trong bài toán mình đang giải quyết.

Có hai cách tiếp cận để giải quyết bài toán bằng cách sử dụng FSL

* **Data-level approach (DLA)**

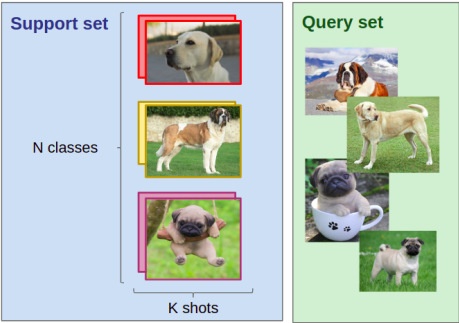
Cách tiếp cận này dựa trên khái niệm: nếu chúng ta không có đủ dữ liệu để xây dựng một model đáng tin cậy và để tránh overfitting/underfitting thì chúng ta chỉ cần thêm nhiều dữ liệu hơn. Chúng ta cũng có thể tự tạo ra nhiều dữ liệu hơn bằng cách sử dụng data augmentation (DA) hoặc generative adversarial networks (GANs).

* **Parameter-level approach (PLA)**

Với data khá ít nên việc overfit khi train model rất dễ xảy ra. Để khắc phục vấn đề này, chúng ta cần giới hạn parameter space hoặc regularization hoặc sử dụng các hàm loss một cách cẩn thận hoặc kết hợp các phương pháp này. Kết quả là chúng ta có một model tổng quát hơn, không bị overfitting trên tập data nhỏ.

Ngoài ra, chúng ta cũng có thể nâng cao hiệu suất của model bằng cách sử dụng standard optimization, điều này làm cho model bớt đáng tin hơn, nhưng không bị quá overfit. Tóm lại, với parameter – level là cách tìm đường đi tốt nhất để model có thể học tốt trên tập data hiện tại. Và như được đề cập ở trên, kỹ thuật này được gọi là Meta – learning. [4]

### ĐẶT VẤN ĐỀ



Hình : Kiến trúc cơ bản của FSL [4]

Chúng ta định nghĩa N – way – K – shot cho vấn đề phân loại hình ảnh như sau:

* Một bộ hỗ trợ bao gồm:
  + N: số nhãn lớp phân loại
  + K: với mỗi nhãn lớp, sẽ có K hình ảnh được gắn nhãn
* Hình ảnh truy vấn (Q query)

Chúng ta muốn phân loại các hình ảnh truy vấn trong N lớp. Các hình ảnh trong bộ hỗ trợ là những ví dụ duy nhất có sẵn cho các lớp này. Khi K nhỏ (thường là K < 10), chúng ta nói về phân loại ảnh ít ảnh (few – shot) (hoặc một ảnh (one – shot) trong trường hợp K = 1). Vấn đề trong trường hợp này là chúng ta không cung cấp đủ hình ảnh của từng lớp để giải quyết vấn đề phân loại với deep neural network tiêu chuẩn, thường yêu cầu hàng nghìn hình ảnh. Lưu ý rằng vấn đề này khác với học có giám sát bán hoặc hàng tuần, vì dữ liệu được dán nhãn đầy đủ. Vấn đề ở đây không phải là khan hiếm nhãn, mà là khan hiếm dữ liệu huấn luyện.

Một ví dụ trực quan về vấn đề phân loại một vài bức ảnh được hiển thị trong hình ở trên. Bài toán FSL (bao gồm phân loại một vài bức ảnh) đã thu hút rất nhiều sự chú ý trong vài năm qua. Nhiều cách khác nhau để giải quyết vấn đề này đã được tưởng tượng. Tất cả đều có điểm chung là sử dụng thông tin bổ sung từ tập dữ liệu cơ sở lớn. Chẳng hạn, nếu nhiệm vụ mục tiêu đang phân loại hình ảnh là Labrador, Saint – Bernard hoặc Pug (hình 11), bộ dữ liệu cơ sở có thể bao gồm nhiều giống chó khác. Ở đây sẽ cung cấp một cái nhìn tổng quan về các giải pháp cho bài toán này.

**Memory-augmented networks:** Santoro và cộng sự (2016) đã có ý tưởng rằng các hình ảnh mới từ các lớp chưa từng thấy trước đây có thể được phân loại bằng cách sử dụng thông tin được lưu trữ về phân loại hình ảnh trước đó. Mô hình của họ sử dụng Recurrent Neural Networks học cả cách lưu trữ và cách truy xuất thông tin liên quan từ dữ liệu trong quá khứ. Các phương pháp khác khai thác ý tưởng mở rộng mạng thần kinh bằng bộ nhớ ngoài.

**Metric learning:** Koch và các cộng sự (2015) đã đề xuất Siamese Neural Networks để giải quyết phân loại hình ảnh vài lần chụp. Mô hình của họ bao gồm hai mạng thần kinh tích chập có trọng số chung, tính toán các phần nhúng (tức là các vectơ đặc trưng) cho hình ảnh đầu vào của chúng và một đầu so sánh đầu ra tương ứng. Tại thời điểm đào tạo (trên tập dữ liệu cơ sở lớn), mạng nhận các cặp hình ảnh làm đầu vào, dự đoán xem chúng có thuộc cùng một lớp hay không và được đào tạo dựa trên độ chính xác của dự đoán này. Cuối cùng, khi được đánh giá trên một lớp phân loại few – shot, mỗi hình ảnh truy vấn được so sánh với mọi hình ảnh trong bộ hỗ trợ và được gán cho lớp được coi là gần nhất (ví dụ: sử dụng k – nearest neighbours). Thuật toán này đã đạt được kết quả thú vị khi phân loại ảnh few – shot. Tuy nhiên, nhiệm vụ mà nó được đào tạo (so sánh hai hình ảnh) khác với nhiệm vụ mà nó được đánh giá (phân loại hình ảnh). Vinals và các cộng sự (2016) cho rằng đây là một nhược điểm và đã đề xuất một phiên bản hơi khác của thuật toán này, bên trong framework meta-learning. Matching Networks của họ cũng phân loại hình ảnh truy vấn bằng cách so sánh nhúng của chúng với nhúng được tính toán từ hình ảnh bộ hỗ trợ, nhưng điểm khác biệt là mục tiêu đào tạo của họ cũng là phân loại hình ảnh. Họ vượt trội so với Siamese Networks. Các công trình sau này hầu như đều nhằm mục đích cải thiện cho thuật toán này.

**Gradient – based meta – learners:** Các thuật toán khác bên trong framework meta – learning tìm hiểu một cách hiệu quả để tinh chỉnh mạng thần kinh tích chập (CNN) trên bộ hỗ trợ nhằm phân loại chính xác bộ truy vấn. Finn và các cộng sự. (2017) đã phát triển model – agnostic meta – learner (MAML) cố gắng tìm hiểu các tham số tốt nhất để khởi tạo CNN nhằm đạt được độ chính xác cao trên tập truy vấn chỉ sau một vài lần giảm độ dốc trên tập hỗ trợ. Meta - SGD được phát triển bởi Li và các cộng sự (2017) còn đi xa hơn: ngoài các tham số khởi tạo, thuật toán này học cho mỗi tham số một tốc độ học và hướng cập nhật. Ravi & Larochelle (2016) đã đề xuất một LSTM network trong đó trạng thái ô là các tham số của CNN. Điều này cho phép thực hiện giảm dần độ dốc đã học, trong đó tất cả các siêu tham số đào tạo của CNN thực sự là các tham số được đào tạo của LSTM.

**Data generation** Một tùy chọn khác để giải quyết vấn đề có quá ít ví dụ để học hỏi là tạo thêm dữ liệu. Hariharan & Girshick (2017) thuật toán học số liệu tăng cường (augmented metric learning) với các vectơ đặc trưng ảo đã được thêm vào các vectơ đặc trưng được trích xuất từ ​​hình ảnh thực. Antoniou et al. (2017) đã áp dụng Generative Adversarial Networks để tăng cường dữ liệu cho việc đào tạo: GAN của họ có thể lấy hình ảnh đầu vào từ một lớp chưa từng thấy trước đó để tạo ra các hình ảnh mới thuộc cùng một lớp. Wang và cộng sự. (2018) đã đề xuất một trình tạo dữ liệu tưởng tượng siêu học có thể được đào tạo theo kiểu đầu cuối với thuật toán phân loại siêu học.[4]

*Trong số rất nhiều giải pháp này, chúng tôi quyết định tập trung vào các thuật toán siêu học (Meta – learning), hiện đang đạt được kết quả hiện đại trong việc phân loại hình ảnh trong vài lần chụp, bên cạnh việc khai thác một mô hình hấp dẫn về mặt khái niệm. Phần tiếp theo đề xuất một công thức của mô hình này.*

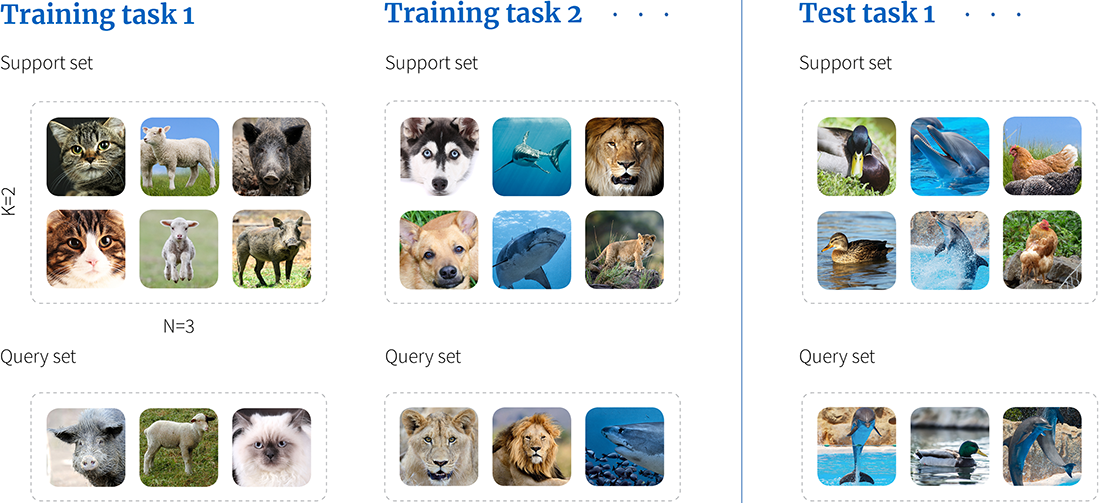
### META LEARNING

#### META LEARNING LÀ GÌ?

Meta – learning (learning to learn) thực hiện việc học thông qua nhiều đợt huấn luyện. Trong quá trình này, nó học cách tự cải thiện thuật toán học tập. Do đó, nó đã thể hiện hiệu suất tốt hơn khi khái quát hóa, đặc biệt là khi một lượng dữ liệu hạn chế được cung cấp.

#### META LEARNING DÀNH CHO FSL

Trong khung học cổ điển (classical learning framework), chúng tôi học cách phân loại từ dữ liệu huấn luyện và đánh giá kết quả bằng dữ liệu thử nghiệm. Trong khung siêu học tập (meta learning framework), chúng tôi học cách phân loại một tập hợp các nhiệm vụ đào tạo nhất định và đánh giá bằng cách sử dụng một tập hợp các nhiệm vụ kiểm tra (hình 12); Nói cách khác, chúng tôi sử dụng một tập hợp các vấn đề phân loại để giúp giải quyết các tập hợp không liên quan khác.



Hình : Kiến trúc của meta - learning [14]

Ở hình 12, một thuật toán được đào tạo bằng cách sử dụng một loạt các nhiệm vụ đào tạo. Ở đây, mỗi nhiệm vụ là một vấn đề phân loại 3 – way 2 – shot vì mỗi nhiệm vụ đào tạo chứa một bộ hỗ trợ với ba loại khác nhau và hai ví dụ cho mỗi loại. Trong quá trình đào tạo, hàm mất mát lần lượt đánh giá hiệu suất trên tập truy vấn cho từng tác vụ với tập hỗ trợ tương ứng. Tại thời điểm thử nghiệm, chúng tôi sử dụng một nhóm tác vụ hoàn toàn khác và đánh giá hiệu suất trên nhóm truy vấn, dựa trên nhóm hỗ trợ. Lưu ý rằng không có sự trùng lặp giữa các lớp trong hai nhiệm vụ huấn luyện {mèo, cừu, lợn}, {chó, cá mập, sư tử} và giữa các lớp trong nhiệm vụ kiểm tra {vịt, cá heo, gà mái}, vì vậy thuật toán phải học cách phân loại các lớp ảnh nói chung chứ không phải một tập hợp cụ thể nào.

Ở đây, mỗi tác vụ mô phỏng kịch bản few – shot, vì vậy đối với phân loại N – way K – shot, mỗi tác vụ bao gồm các lớp có ví dụ của mỗi lớp. Chúng được gọi là tập hỗ trợ cho tác vụ và được sử dụng để học cách giải quyết tác vụ này. Ngoài ra, còn có các ví dụ khác của cùng các lớp, được gọi là tập truy vấn, được sử dụng để đánh giá hiệu suất trên tác vụ này. Mỗi tác vụ có thể hoàn toàn không chồng chéo; chúng ta có thể không bao giờ thấy các lớp từ một tác vụ trong bất kỳ tác vụ nào khác. Ý tưởng là hệ thống liên tục nhìn thấy các trường hợp (tác vụ) trong quá trình huấn luyện phù hợp với cấu trúc của tác vụ few – shot cuối cùng nhưng chứa các lớp khác nhau.

Tại mỗi bước của quá trình meta – learning, chúng ta cập nhật các tham số của mô hình dựa trên một tác vụ huấn luyện được chọn ngẫu nhiên. Hàm mất mát được xác định bằng hiệu suất phân loại trên tập truy vấn của tác vụ huấn luyện này, dựa trên kiến thức thu được từ tập hỗ trợ của nó. Vì mạng được trình bày với một tác vụ khác nhau ở mỗi bước thời gian, nó phải học cách phân biệt các lớp dữ liệu chung chung hơn là một tập con cụ thể của các lớp.

Để đánh giá hiệu suất few – shot, chúng ta sử dụng một tập các tác vụ kiểm tra. Mỗi tác vụ chứa chỉ các lớp chưa được nhìn thấy trong bất kỳ tác vụ huấn luyện nào. Đối với mỗi tác vụ, chúng ta đo hiệu suất trên tập truy vấn dựa trên kiến thức từ tập hỗ trợ của chúng. [14]

#### THUẬT TOÁN CỦA META LEARNING

##### GRADIENT – BASED META LEARNING

Trong phần này, chúng ta phân biệt meta – learner, là mô hình học qua các tập episodes, mô hình thứ hai là base – learner, được khởi tạo và đào tạo bên trong một episode bởi meta – learner.

Chúng tôi hãy xem xét một giai đoạn meta – training, với nhiệm vụ phân loại được xác định bởi một tập hỗ trợ gồm N ∗ K hình ảnh được dán nhãn và một tập truy vấn gồm Q hình ảnh. Mô hình base – learner, điển hình là bộ phân loại CNN, sẽ được khởi tạo, sau đó được đào tạo trên tập hỗ trợ (ví dụ: tập huấn luyện cơ sở). Thuật toán được sử dụng để đào tạo base – learner được xác định bởi mô hình meta-learner. Sau đó, mô hình base-learner được áp dụng để dự đoán việc phân loại các hình ảnh truy vấn Q. Các tham số của meta – learner được đào tạo ở cuối episode tránh khỏi sự mất mát do lỗi phân loại. Từ thời điểm này, các thuật toán khác nhau về sự lựa chọn meta – model của chúng.

A diagram of a bird

Description automatically generated with medium confidence

Hình : Kiến trúc cơ bản của gradient – based meta learning [4]

Hình 8: episode meta – training thứ d, tuân theo quy trình này:

(1) Tập hỗ trợ và tập truy vấn được lấy mẫu từ tập huấn luyện.

(2) Mô hình cơ sở (base – model) được khởi tạo bởi siêu mô hình (meta – model) .

(3) Các tham số của base – model được tinh chỉnh trên tập hỗ trợ (quá trình tinh chỉnh phụ thuộc vào ).

(4) Sau khi cập nhật T, base – model được đánh giá trên bộ truy vấn.

(5) Các tham số Θ của siêu mô hình được cập nhật bằng cách lan truyền ngược tổn thất do các dự đoán của base – model trên tập truy vấn.

**Meta – LSTM (2016)**: Ravi & Larochelle đã quyết định sử dụng long – short – term – memory network: các tham số θ của base – learner được biểu thị bằng trạng thái ô của LSTM, dẫn đến quy tắc cập nhật trong đóvà  tương ứng là forget gate và input gate vào của LSTM, và là một đầu vào. Chúng ta có thể xem quy tắc cập nhật là một phần mở rộng của lan truyền ngược, vì với , là tốc độ học và , chúng ta có được lan truyền ngược tiêu chuẩn. Do đó, mô hình này học cách vận hành giảm độ dốc một cách hiệu quả trên Base – Model từ tập hỗ trợ, để làm cho Base – Model này chính xác hơn trên tập truy vấn.

**Model – Agnostic Meta – Learning (MAML** ,**2017):** Finn và các cộng sự đã đề xuất một thuật toán học cách khởi tạo các tham số của base – model, nhưng không can thiệp vào việc cập nhật các tham số của base – model. Ở đây, meta – learner tạo một bản sao của chính nó ở đầu mỗi episode và bản sao này (base – model) được tinh chỉnh trên tập hỗ trợ, sau đó đưa ra dự đoán trên tập truy vấn. Tổn thất được tính toán từ những dự đoán này được sử dụng để cập nhật các tham số của meta – model (do đó, các tham số khởi tạo cho các tập tiếp theo sẽ khác). Đặc điểm chính của phương pháp này là nó được coi là bất khả tri của base – model, có nghĩa là nó hầu như có thể được áp dụng cho bất kỳ thuật toán học máy nào. Finn và các cộng sự đã thử nghiệm nó trên hồi quy và phân loại có giám sát, cũng như trên các nhiệm vụ học tăng cường, nhưng nó có thể được sử dụng để giải quyết nhiều vấn đề khác đòi hỏi sự thích ứng nhanh của deep neural network, chẳng hạn như để phát hiện đối tượng trong FSL. [2]

Thuật toán MAML với một độ dốc cập nhật trên tập hỗ trợ.

**Require:** p(*T*): *phân phối trên các nhiệm vụ*

**Require:** : *tham số step size*

*Khởi tạo ngẫu nhiên*

**while** not done **do**

*Lấy mẫu một lô các nhiệm vụ*

**for all do**

*Đánh giá* *đối với K ví dụ*

*Tính toán các tham số đã thích nghi thông qua việc giảm*

*gradient:*

**end for**

*Cập nhật*

**end while**

##### METRIC LEARNING

###### BỘ SO SÁNH CẶP

Các bộ so sánh theo cặp lấy hai ví dụ và phân loại chúng thuộc cùng một lớp hoặc các lớp khác nhau. Điều này khác với cấu hình N – way K – shot tiêu chuẩn và rõ ràng không ánh xạ vào mô tả meta learning ở trên mặc dù như chúng ta sẽ thấy sau này, trên thực tế, có một mối quan hệ chặt chẽ.

SIAMESE NETWORKS

1. Kiến trúc của Siamese Networks

Mô hình tiêu chuẩn của chúng ta là một mạng thần kinh tích chập siamese với *L* lớp, mỗi lớp có đơn vị, trong đó đại diện gửi vectơ ẩn trong lớp l cho cặp song sinh đầu tiên và biểu thị tương tự cho cặp song sinh thứ hai. Chúng ta sử dụng các đơn vị tuyến tính được chỉnh lưu riêng (ReLU) trong các lớp *L* - 2 đầu tiên và các đơn vị sigmoidal trong các lớp còn lại.

Mô hình bao gồm một chuỗi các lớp tích chập, mỗi lớp sử dụng một kênh duy nhất với các bộ lọc có kích thước khác nhau và stride cố định là 1. Số lượng bộ lọc tích chập được chỉ định là bội số của 16 để tối ưu hóa hiệu suất. Mạng áp dụng chức năng kích hoạt ReLU cho các feature maps đầu ra, tùy chọn theo sau là tổng hợp tối đa với kích thước bộ lọc và stride là 2. Do đó, filter map thứ *k* trong mỗi lớp có dạng sau:

Trong đó là tensor 3 chiều biểu thị các feature maps cho lớp *l* và là phép toán tích chập hợp lệ tương ứng cho việc chỉ trả về các đơn vị đầu ra là kết quả của sự chồng lấp hoàn toàn giữa mỗi bộ lọc tích chập và feature maps đầu vào.

Các đơn vị trong lớp tích chập cuối cùng được làm phẳng thành một vectơ duy nhất. Lớp tích chập này được theo sau bởi một lớp được kết nối đầy đủ, và sau đó một lớp nữa sẽ đặt thước đo khoảng cách cảm ứng giữa mỗi cặp song sinh siamese, được cấp cho một đơn vị đầu ra sigmoidal duy nhất. Chính xác hơn, vectơ dự đoán được đưa ra dưới dạng , trong đó σ là hàm kích hoạt sigmoidal. Lớp cuối cùng này tạo ra một số liệu trên không gian đặc trưng đã học của lớp ẩn thứ ) và chấm điểm sự giống nhau giữa hai vectơ đặc trưng. là các tham số bổ sung được mô hình học trong quá trình huấn luyện, đánh giá tầm quan trọng của khoảng cách giữa các thành phần. Điều này xác định lớp được kết nối đầy đủ lớp thứ *L* cuối cùng cho mạng kết hợp hai cặp song sinh siamese. [15]

Diagram

Description automatically generated

Hình : Kiến trúc của siamese networks [15]

1. Quá trình học tập

**Hàm mất mát:** Đặt *M* đại diện cho kích thước minibatch, trong đó *i* lập chỉ mục cho minibatch thứ i. Bây giờ hãy để là một vectơ độ dài M chứa các nhãn cho lô nhỏ, trong đó chúng ta giả sử bất cứ khi nào và thuộc cùng một lớp ký tự và nếu không. Chúng ta áp đặt một mục tiêu entropy chéo được chuẩn hóa trên bộ phân loại nhị phân của ở dạng sau:

**Tối ưu hóa:** Mục tiêu này được kết hợp với thuật toán lan truyền ngược tiêu chuẩn, trong đó độ dốc được bổ sung trên các mạng kép do các trọng số bị ràng buộc. Chúng ta cố định kích thước minibatch là 128 với tốc độ học , xung lượng và trọng số chuẩn hóa , được xác định theo từng lớp, do đó quy tắc cập nhật của chúng ta tại epoch T như sau:

Trong đó là đạo hàm riêng theo trọng số giữa nơ-ron thứ *j* trong lớp nào đó và nơ-ron thứ *k* trong lớp kế tiếp

**Khởi tạo trọng số:** Khởi tạo tất cả các trọng số mạng trong các lớp tích chập từ phân phối chuẩn với giá trị trung bình bằng 0 và độ lệch chuẩn là . Các độ lệch cũng được khởi tạo từ một phân phối chuẩn, nhưng với giá trị trung bình là 0.5 và độ lệch chuẩn . Trong các lớp được kết nối đầy đủ, các độ lệch được khởi tạo theo cách giống như các lớp tích chập, nhưng các trọng số được rút ra từ phân phối chuẩn rộng hơn nhiều với độ lệch trung bình bằng không và độ lệch chuẩn 2 × .

**Kế hoạch học tập:** Mặc dù cho phép tốc độ học tập khác nhau cho mỗi lớp, tốc độ học tập được phân rã đồng đều trên mạng 1 phần trăm mỗi epoch, do đó . Chúng ta nhận thấy rằng bằng cách tăng tốc độ học tập, mạng có thể hội tụ về điểm cực tiểu cục bộ dễ dàng hơn mà không gặp phải lỗi trên bề mặt. Sau đó chúng ta cần cố định động lượng để bắt đầu ở mức 0.5 trong mỗi lớp, tăng tuyến tính mỗi kỷ nguyên cho đến khi đạt đến giá trị , thuật ngữ động lượng riêng cho lớp thứ *j*.

**Tối ưu hóa siêu tham số:** Sử dụng phiên bản beta của Whetlab, một khung tối ưu hóa Bayesian, để thực hiện lựa chọn siêu tham số. Đối với siêu tham số lịch trình học và chính quy hóa, chúng ta đặt tốc độ học theo từng lớp ∈ [, ], động lượng theo từng lớp ∈ [0, 1] và layer – wise theo từng lớp ∈ [0, 0,1]. Đối với siêu tham số mạng, chúng ta để kích thước của các bộ lọc tích chập thay đổi từ đến , trong khi số lượng bộ lọc tích chập trong mỗi lớp thay đổi từ 16 đến 256 bằng cách sử dụng bội số của 16. Các lớp được kết nối đầy đủ nằm trong khoảng từ 128 đến 4096 đơn vị, cũng theo bội số trên 16. Chúng ta đặt trình tối ưu hóa để tối đa hóa độ chính xác của bộ xác thực một lần. Điểm số được chỉ định cho một lần lặp lại Whetlab là giá trị cao nhất của chỉ số này được tìm thấy trong bất kỳ kỷ nguyên nào.

**Biến dạng Affine**: Ngoài ra, chúng ta có thể tăng cường tập huấn luyện với các biến dạng Affine nhỏ (Hình 15). Đối với mỗi cặp ảnh , chúng ta tạo ra một cặp phép biến hình affine để tạo ra , , trong đó , được xác định ngẫu nhiên bởi một đa chiều phân phối đồng đều. Vì vậy, đối với dạng chuyển đổi tùy ý *T*, chúng ta có *T* = (*θ*, , , , , , ), với *θ* ∈ [−10.0, 10.0], , ∈ [−0.3, 0.3], , ∈ [0.8, 1.2] và, ∈ [−2, 2]. Mỗi thành phần này của quá trình chuyển đổi được bao gồm với xác suất 0.5. [15]

Text, letter

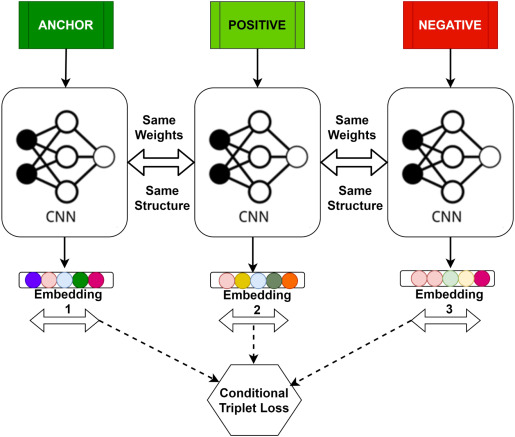
Description automatically generated

Hình : Dữ liệu sau khi được biến dạng Affine [15]

TRIPLET NETWORKS

Triplet networks là một kiến trúc mạng nơ-ron được sử dụng trong bài toán học có giám sát với số lượng dữ liệu huấn luyện rất nhỏ, gọi là FSL. Triplet networks giúp xác định sự tương đồng giữa các đối tượng trong không gian vectơ, thông qua việc học cách phân biệt các đối tượng. Trong triplet networks, một bộ ba ảnh được đưa vào mạng gồm một ảnh anchor, một ảnh positive và một ảnh negative. Mục tiêu của mô hình là học cách đưa ảnh anchor và positive gần nhau hơn trong không gian vectơ so với ảnh anchor và negative. Việc học này giúp mô hình có khả năng phân biệt tốt hơn giữa các đối tượng và từ đó dễ dàng phân loại các đối tượng mới chỉ bằng một số lượng nhỏ các ảnh huấn luyện. Triplet networks là một trong những kiến trúc mạng phổ biến trong lĩnh vực nhận dạng ảnh và xử lý ngôn ngữ tự nhiên.

* Kiến trúc của triplet networks:



Hình : Kiến trúc tổng quan triplet networks

Kiến trúc mô hình của triplet networks bao gồm ba mạng nơ-ron với cùng một kiến trúc được chia sẻ trọng số. Ba mạng nơ – ron này được đặt tên là anchor network, positive network và negative network.

Mỗi một mạng nơ – ron này nhận đầu vào là một ảnh và trả về một embedding vectơ tương ứng. Một số kiến trúc mạng được sử dụng trong base networks là convolution neural networks (CNN), residual networks (ResNet) hoặc inception networks. Embedding vectors này có số chiều bằng nhau và được giảm chiều trước khi đưa vào hàm tính toán khoảng cách. Ta có thể sử dụng công thức euclid hoặc cosine distance.

Nếu chúng ta biểu thị đầu vào là tương ứng với anchor, positive và negative. Sau đó biểu diễn nhúng của mạng là , thì đầu vào trước lớp cuối cùng sẽ là vectơ:

Nói cách khác, điều này mã hóa cặp khoảng cách giữa mỗi và so với tham chiếu x.

A picture containing diagram, plan, sketch, technical drawing

Description automatically generated

Hình : Kiến trúc chi tiết của triplet networks [16]

* Quá trình huấn luyện

Quá trình huấn luyện được hình thành trước bằng cách cung cấp cho mạng các mẫu trong đó, anchor (x) và positive thuộc cùng một lớp và negative thuộc lớp khác. Kiến trúc mạng cho phép nhiệm vụ được thể hiện dưới dạng bài toán phân loại 2 lớp, trong đó mục tiêu là phân loại chính xác và thuộc cùng một lớp với x. Và nhấn mạnh rằng trong một cài đặt tổng quát hơn, trong đó mục tiêu có thể là tìm hiểu cách nhúng số liệu, nhãn sẽ xác định ví dụ nào gần với x hơn. Ở đây chúng ta chỉ đơn giản giải thích “sự gần gũi” là “chia sẻ cùng một nhãn”. Để xuất toán tử so sánh từ mô hình, hàm softmax được áp dụng trên cả hai đầu ra để tạo ra thước đo tỷ lệ một cách hiệu quả. Tương tự như các mạng tích chập truyền thống, quá trình huấn luyện được thực hiện bằng phương pháp SGD đơn giản trên hàm mất mát negative-log-likelihood đối với bài toán 2 lớp. Sau đó, chúng ta có thể thử nghiệm và thấy kết quả tốt hơn khi hàm mất mát được thay thế bằng một MSE (mean squared error) trên kết quả softmax, so sánh với vectơ (0, 1), do đó phép tính hàm mất mát sẽ là:

Với

Và

.

Lưu ý rằng: , đó là mục tiêu cần thiết. Bằng cách sử dụng cùng một mạng tham số được chia sẻ, sẽ cho phép thuật toán lan truyền ngược cập nhật mô hình liên quan đến cả ba mẫu cùng một lúc. [16]

###### BỘ SO SÁNH NHIỀU LỚP

MATCHING NETWORKS

Con người học các khái niệm mới với rất ít sự giám sát – ví dụ: một đứa trẻ có thể khái quát hóa khái niệm “hươu cao cổ” từ một bức tranh trong sách – nhưng hệ thống học sâu tốt nhất của chúng ta cần hàng trăm hoặc hàng nghìn ví dụ. Điều này thúc đẩy sự quan tâm của những người nghiên cứu: học tập “với một ví dụ”, bao gồm việc học một lớp từ một ví dụ được dán nhãn duy nhất.

Học sâu đã đạt được những tiến bộ lớn trong các lĩnh vực như lời nói, tầm nhìn và ngôn ngữ, nhưng hiển nhiên là đòi hỏi bộ dữ liệu lớn. Các kỹ thuật tăng cường và chính quy hóa dữ liệu làm giảm bớt tình trạng thừa trong các chế độ dữ liệu thấp, nhưng không giải quyết được vấn đề đó. Hơn nữa, quá trình học vẫn còn chậm và dựa trên các bộ dữ liệu lớn, yêu cầu nhiều cập nhật trọng số bằng cách sử dụng phương pháp giảm độ dốc ngẫu nhiên. Theo quan điểm của nhiều người, điều này chủ yếu là do khía cạnh tham số của mô hình, trong đó các ví dụ đào tạo cần được mô hình học từ từ thành các tham số của nó.

Diagram

Description automatically generated

Hình : Kiến trúc của matching networks

Ngược lại, nhiều mô hình phi tham số cho phép các ví dụ mới lạ được đồng hóa nhanh chóng, trong khi không bị lãng quên một cách quá nghiêm trọng. Một số mô hình không yêu cầu bất kỳ khóa đào tạo nào nhưng hiệu suất phụ thuộc vào số liệu đã chọn. Công việc trước đây về học số liệu trong các thiết lập không tham số đã có ảnh hưởng đến mô hình và mong muốn kết hợp các đặc điểm tốt nhất từ cả mô hình tham số và phi tham số - cụ thể là, thu thập nhanh các ví dụ mới trong khi cung cấp khả năng khái quát hóa xuất sắc từ các mô hình ví dụ phổ biến. [17]

1. Kiến trúc của Matching Networks

Cách tiếp cận phi tham số để giải quyết vấn đề one – shot learning dựa trên hai thành phần. Đầu tiên, kiến trúc mô hình tuân theo những tiến bộ gần đây trong mạng thần kinh được tăng cường bộ nhớ (neural networks). Với một tập hỗ trợ (nhỏ) S, mô hình được xác định bởi một hàm (hoặc bộ phân loại) cho mỗi S, tức là một ánh xạ *S* →. Thứ hai, sử dụng một chiến lược đào tạo được thiết kế để học một lần từ bộ hỗ trợ S.

1. Kiến trúc mô hình:

Trong những năm gần đây, nhiều nhóm đã nghiên cứu các cách để tăng cường kiến trúc mạng thần kinh bằng bộ nhớ bên ngoài và các thành phần khác khiến chúng trở nên “giống máy tính” hơn. Kiến trúc này lấy cảm hứng từ các mô hình như trình tự nối tiếp trình tự (seq2seq) với sự chú ý (attention), mạng bộ nhớ (memory networks) và mạng con trỏ (pointer networks).

Trong tất cả các mô hình này, một cơ chế chú ý thần kinh (neural attention mechanism), thường có thể phân biệt hoàn toàn, được xác định để truy cập (hoặc đọc) ma trận bộ nhớ lưu trữ thông tin hữu ích để giải quyết nhiệm vụ hiện tại. Các ứng dụng điển hình của điều này bao gồm dịch máy, nhận dạng giọng nói hoặc trả lời câu hỏi. Tổng quát hơn, các kiến trúc này mô hình trong đó A và/hoặc B có thể là một chuỗi (như trong các mô hình seq2seq), hoặc có thể là một tập hợp.

Đặt vấn đề học một lần trong khuôn khổ set-to-set. Điểm mấu chốt là khi được đào tạo, Matching networks có thể tạo nhãn kiểm tra hợp lý cho các lớp không được quan sát mà không có bất kỳ thay đổi nào đối với mạng. Chính xác hơn, chúng tôi muốn ánh xạ từ một bộ hỗ trợ (nhỏ) gồm k ví dụ về các cặp nhãn hình ảnh tới một bộ phân loại mà, với một ví dụ thử nghiệm , xác định phân phối xác suất trên đầu ra . Chúng tôi xác định ánh xạ *S* → là trong đó *P* được tham số hóa bởi một mạng thần kinh. Do đó, khi được cung cấp một bộ hỗ trợ mới gồm các ví dụ để học một lần, thì ta chỉ cần sử dụng mạng thần kinh tham số được xác định bởi *P* để đưa ra dự đoán về nhãn thích hợp cho từng ví dụ thử nghiệm : . Nói chung, lớp đầu ra dự đoán của chúng tôi cho một ví dụ đầu vào chưa biết đã cho và tập hỗ trợ S trở thành.

Từ đó được tính toán một cách đơn giản như sau:

Với là các mẫu và nhãn từ tập hỗ trợ và a là một cơ chế chú ý qua các ví dụ. Lưu ý rằng phương trình (1) về cơ bản mô tả đầu ra cho một lớp mới dưới dạng kết hợp tuyến tính của các nhãn trong bộ hỗ trợ. Trong đó cơ chế chú ý *a* là kernel trên, thì (1) gần giống với công cụ ước tính mật độ kernel. Khi cơ chế chú ý bằng không đối với *b* xa nhất tính từ theo một số liệu khoảng cách và một hằng số thích hợp nếu không thì (1) tương đương với ‘’ – đối tượng gần nhất. Do đó (1) gộp cả hai phương thức KDE và kNN. Một cách nhìn khác về (1) là nơi *a* hoạt động như một cơ chế chú ý và đóng vai trò là bộ nhớ liên kết với tương ứng. Trong trường hợp này, chúng ta có thể hiểu đây là một loại bộ nhớ kết hợp cụ thể trong đó, được cung cấp một đầu vào, chúng ta “trỏ” đến ví dụ tương ứng trong bộ hỗ trợ, truy xuất nhãn của nó. Tuy nhiên, không giống như các cơ chế bộ nhớ chú ý khác, (1) về bản chất là không tham số: khi kích thước bộ hỗ trợ tăng lên, thì bộ nhớ được sử dụng cũng tăng theo. Do đó, dạng hàm được xác định bởi bộ phân loại rất linh hoạt và có thể dễ dàng thích ứng với bất kỳ bộ hỗ trợ mới nào. [17]

* Cơ chế chú ý kernel (attention kernel)

Phương trình (1) dựa vào việc chọn *a* (.,.), cơ chế chú ý (attention mechanism), chỉ định đầy đủ bộ phân loại. Hình thức đơn giản nhất mà điều này sử dụng (và có mối quan hệ rất chặt chẽ với các mô hình chú ý phổ biến và các hàm nhân) là sử dụng softmax trên khoảng cách cosin c, tức là:

Với các hàm nhúng *f* và *g* là các mạng thần kinh thích hợp (có khả năng với *f* = *g*) để nhúng và . Trong các thử nghiệm , chúng ta sẽ thấy các ví dụ trong đó *f* và *g* được tham số hóa khác nhau như các mạng tích chập sâu cho các tác vụ hình ảnh (như trong VGG hoặc Inception) hoặc nhúng từ dạng đơn giản cho các tác vụ ngôn ngữ.

Lưu ý rằng, mặc dù liên quan đến việc học số liệu, trình phân loại được xác định bởi Phương trình 1 là có tính phân biệt. Đối với tập hỗ trợ S và mẫu cho trước để phân loại , điều kiện đủ để được căn chỉnh đủ mạnh với các cặp sao cho và không căn chỉnh phù hợp với phần còn lại. Loại mất mát này cũng có liên quan đến các phương pháp như neighborhood component analysis (NCA), triplet loss hoặc large margin nearest neighbor.

Tuy nhiên, mục tiêu mà chúng ta đang cố gắng tối ưu hóa chính xác phù hợp với việc phân loại đa hướng, một lần nhìn (multi-way, one-shot classification), và do đó chúng ta kỳ vọng nó sẽ hoạt động tốt hơn so với các phương pháp tương đương khác. Ngoài ra, mất mát này đơn giản và có khả năng khác biệt, cho phép tìm ra các tham số tối ưu theo cách "end-to-end" (từ đầu đến cuối) một cách dễ dàng. [17]

* Hàm nhúng toàn bộ ngữ cảnh (full context embedding)

Sự độc đáo chính của mô hình là nằm trong việc hiểu lại một khung hình đã được nghiên cứu kỹ (mạng nơ – ron với bộ nhớ bên ngoài) để thực hiện việc học one-shot. Liên quan mật thiết đến việc học đo lường, các hàm nhúng *f* và *g* hoạt động như một quá trình tăng cường không gian đặc trưng *X* để đạt độ chính xác tối đa thông qua hàm phân loại được mô tả trong phương trình 1.

Mặc dù chiến lược phân loại hoàn toàn dựa trên toàn bộ tập hỗ trợ thông qua nhưng việc nhúng các phần tử riêng lẻ bằng cách sử dụng hàm để tính toán sự tương đồng cosine là hẹp hơn trong ý nghĩa rằng mỗi phần tử được nhúng độc lập với các phần tử khác trong tập hỗ trợ *S*. Hơn nữa, *S* cần có khả năng thay đổi cách chúng ta nhúng hình ảnh kiểm tra thông qua hàm *f*.

Chúng tôi đề xuất nhúng các phần tử của tập hợp thông qua một hàm nhận đầu vào là toàn bộ tập hỗ trợ *S* cùng với , tức là *g* trở thành . Do đó, như một hàm của toàn bộ tập hỗ trợ *S*, *g* có thể thay đổi cách nhúng . Điều này có thể hữu ích khi một phần tử gần với , trong trường hợp này việc thay đổi hàm nhúng có thể mang lại lợi ích. Chúng tôi sử dụng một mạng LSTM hai chiều (Bidirectional Long-Short Term Memory) để mã hóa trong ngữ cảnh của tập hỗ trợ *S*, được coi là một chuỗi.

Vấn đề thứ hai có thể được giải quyết thông qua một LSTM với khả năng chú ý đọc qua toàn bộ tập *S*, trong đó đầu vào của LSTM là *x* và tập *S*:

Trong đó là mạng thần kinh (ví dụ bắt nguồn từ CNN) được nhập vào LSTM (không đổi tại mỗi bước thời gian), K là số bước không kiểm soát cố định của LSTM và là tập hợp mà chúng tham dự, được nhúng với được áp dụng cho từng phần từ từ tập hợp *S*. Điều này cho phép mô hình có khả năng bỏ qua một số phần tử trong Support Set S và thêm “độ sâu” vào tính toán của hàm Attention.

Như vậy, trạng thái sau *k* bước xử lý như sau:

Lưu ý rằng: LSTM (*x, h, c*) tuân theo cùng cách triển khai LSTM với *x* đầu vào, *h* đầu ra (nghĩa là ô sau cổng đầu ra) và c ô. a thường được gọi là sự chú ý dựa trên “nội dung” và softmax trong (5) bình thường hóa w.r.t. . Giá trị đọc ra từ *g(S)* được nối với . Vì chúng ta thực hiện K bước “đọc”, nên attLSTM trong đó được mô tả trong công thức số (2). [17]

1. Chiến lược đào tạo

Trong phần trước, chúng tôi đã mô tả Matching Networks, một mô hình ánh xạ từ tập hỗ trợ (support set) đến hàm phân loại, *S* →. Chúng tôi đạt được điều này thông qua một sự thay đổi của mô hình set – to – set hợp với chú ý (attention), với ánh xạ kết quả có dạng , lưu ý rằng *θ* là các tham số của mô hình (tức là của các hàm nhúng *f* và *g* đã được mô tả trước đó).

Quá trình huấn luyện phải được chọn cẩn thận để phù hợp với quá trình suy luận trong thời gian kiểm tra. Mô hình của chúng tôi phải hoạt động tốt với các tập hỗ trợ chứa các lớp chưa từng được thấy trong quá trình huấn luyện.

Cụ thể hơn, hãy định nghĩa một nhiệm vụ *T* là phân phối các tập nhãn khả thi *L*. Thông thường, chúng tôi coi T là phân phối đồng đều các tập dữ liệu chứ một số lớp duy nhất , với vài ví dụ cho mỗi lớp (ví dụ: tối đa 5). Trong trường hợp này, một tập nhãn *L* được lấy mẫu từ một nhiệm vụ *T*, *L ∼ T*, thường sẽ có từ 5 đến 25 ví dụ.

Để tạo ra một " episode " để tính gradient và cập nhật mô hình, chúng tôi trước tiên lấy mẫu *L* từ *T* (ví dụ: *L* có thể là tập nhãn *{mèo, chó}*). Sau đó, chúng tôi sử dụng L để lấy mẫu tập hỗ trợ S và một lô dữ liệu B (tức là cả S và B đều là các ví dụ được gán nhãn về mèo và chó). Mạng Matching Net sau đó được huấn luyện để giảm thiểu sai số trong dự đoán các nhãn trong lô B dựa trên tập hỗ trợ S. Đây là một hình thức của meta-learning vì quá trình huấn luyện rõ ràng học cách học từ một tập hỗ trợ đã cho để giảm thiểu mất mát trên một lô dữ liệu. Cụ thể hơn, mục tiêu huấn luyện của Matching Nets như sau:

Huấn luyện *θ* bằng phương trình (2) tạo ra một mô hình hoạt động tốt khi lấy mẫu support set từ một phân phối khác của nhãn mới. Đặc biệt, mô hình này không cần điều chỉnh trên các lớp mà nó chưa từng thấy nhờ tính phi tham số của nó. Tuy nhiên, rõ ràng, khi dịch chuyển xa so với *T* mà chúng tôi lấy mẫu để học , mô hình sẽ không hoạt động. [17]

PROTOTYPICAL NETWORKS

1. Kiến trúc của Prototypical Networks:

Cả hai phương pháp prototypical networks và k – mean clustering đều định vị các trung tâm hoặc các điểm trung tâm để đại diện cho các nhóm dữ liệu.

A picture containing diagram, colorfulness, screenshot, pixel

Description automatically generated

Hình : K – means clustering [18]

K – means clustering: Đầu tiên muốn phân vùng N điểm dữ liệu thành K cụm trong đó mỗi điểm thuộc về cụm có giá trị trung bình gần nhất, đóng vai trò là nguyên mẫu của cụm (prototype of the cluster):

(1) Khởi tạo: Chọn số lượng cụm K ban đầu và khởi tạo K điểm ngẫu nhiên làm các điểm trung tâm ban đầu (ví dụ trên hình là K = 3).

(2) Gán điểm dữ liệu: Gán từng điểm dữ liệu vào cụm gần nhất bằng cách tính khoảng cách từ điểm dữ liệu đó đến tất cả các điểm trung tâm, và chọn cụm có khoảng cách nhỏ nhất. Các phân vùng ở đây đại diện cho sơ đồ Voronoi được tạo bởi các means.

(3) Cập nhật lại trung tâm: Tính toán lại các điểm trung tâm của các cụm dựa trên các điểm dữ liệu được gán vào từng cụm. Các điểm trung tâm mới được tính bằng cách lấy trung bình của tất cả các điểm dữ liệu thuộc cùng một cụm.

(4) Lặp lại bước 2 và 3 cho đến khi không có sự thay đổi đáng kể trong việc gán dữ liệu và cập nhật trung tâm, hoặc đạt đến số lần lặp tối đa được xác định trước.

(5) Sau khi kết thúc, ta sẽ nhận được các cụm được phân chia dựa trên sự tương tự của các điểm dữ liệu và vị trí của các điểm trung tâm. [18]

Nhưng prototypical có một số điểm khác biệt so với k – means clustering:

A picture containing cartoon, diagram, design

Description automatically generated

Hình : Kiến trúc của prototypical networks [19]

Prototypical Networks tính toán một M – dimensional representation (biểu diễn chiều) , hoặc là prototype thông qua hàm nhúng với các tham số có thể học được. Mỗi prototype là vectơ trung bình của các điểm hỗ trợ nhúng thuộc về lớp của nó:

Với là tập hợp các ví dụ được gán nhãn với lớp *k*. Đối với một ví dụ mới *x*, xác suất của ví dụ này thuộc lớp *y = k* được tính bằng cách sử dụng hàm softmax.

Đưa ra một hàm khoảng cách , prototypical networks tạo ra phân phối trên các lớp cho một điểm truy vấn dựa trên softmax trên khoảng cách đến các nguyên mẫu trong không gian nhúng:

Trong đó *d* là thước đo khoảng cách. Khi khoảng cách euclide được sử dụng, đó là:

Quá trình học tiến hành bằng cách giảm thiểu xác suất log âm của lớp thực k thông qua SGD. Các giai đoạn huấn luyện được hình thành bằng cách chọn ngẫu nhiên một tập con các lớp từ tập huấn luyện, sau đó chọn một tập con các mẫu trong mỗi lớp để đóng vai trò là tập hỗ trợ và một tập con của phần còn lại để đóng vai trò là điểm truy vấn. Mã giả để tính hàm loss (hàm mất mát) cho một đợt huấn luyện được cung cấp trong phần b) của thuật toán bên dưới. [19]

1. Thuật toán của Prototypical Networks

Tính toán hàm loss (hàm mất mát) tập huấn luyện cho prototypical networks. N là số ví dụ trong tập huấn luyện, K là số lớp trong tập huấn luyện, ≤ K là số lớp trên mỗi tập, là số ví dụ hỗ trợ trên mỗi lớp, là số ví dụ truy vấn mỗi lớp. RANDOMSAMPLE (*S, N*) biểu thị một tập hợp gồm N phần tử được chọn ngẫu nhiên thống nhất từ tập hợp S, không thay thế.

**Input:** Tập huấn luyện , trong đó mỗi . biểu thị tập hợp con của *D* chứa tất cả các phần tử sao cho .

**Output**: Hàm loss *J* cho một tập huấn luyện được tạo ngẫu nhiên.

*V* 🡨 RANDOMSAMPLE 🡪 Chọn chỉ số lớp cho tập

**for** *k* in **do**

🡨 RANDOMSAMPLE 🡪 Chọn ví dụ hỗ trợ

🡨 RANDOMSAMPLE 🡪 Chọn ví dụ truy vấn

🡨 🡪 Tính toán Prototype từ các ví dụ hỗ trợ

**end for**

*J* 🡨 0 🡪 Khởi tạo hàm loss

**for** *k* **in** **do**

**for** (*x, y*) **in** **do**

*J* 🡨 🡪 Cập nhật hàm loss

**end for**

**end for**

1. Tối ưu hóa

Giả sử một miền được định nghĩa là phân phối chung của không gian ảnh *X* và không gian nhãn *Y*. Để (*x, y*) lưu ý một ví dụ có nhãn, trong đó *y* là nhãn của *x*. Trong phương pháp few – shot trên nhiều miền, miền nguồn trong đào tạo meta khác với miền đích trong thử nghiệm meta, trong khi ở bên trong miền, miền nguồn và miền đích là giống nhau. Trong siêu thử nghiệm, một bộ hỗ trợ, , được cung cấp sao cho mỗi một trong số *N* lớp cơ sở, có *K* ví dụ được dán nhãn. Mục tiêu là đào tạo một bộ phân loại giảm thiểu cho hàm mất mát , trong đó kỳ vọng được đánh giá bằng cách sử dụng bộ truy vấn trong thử nghiệm meta.

Cách tiếp cận của chúng tôi có ba thành phần chính: áp dụng hỗn hợp để tăng cường dữ liệu, tinh chỉnh EfficientNet có sẵn để trích xuất tính năng và sử dụng với xấp xỉ Gaussian cho việc học không gian số liệu:

**Mixup để tăng cường dữ liệu**: Tăng cường dữ liệu, đặc biệt là về các phát triển gần đây ngoài lật, cắt xén và chia tỷ lệ, là một cách hiệu quả để nâng cao hiệu quả dữ liệu bằng cách tăng số lượng và cải thiện tính đa dạng của dữ liệu. Ngoài việc tăng cường dữ liệu tiêu chuẩn, chúng ta còn có thể triển khai một kỹ thuật gọi là mixup sử dụng kết hợp tuyến tính của hai ví dụ huấn luyện bất kỳ và nhãn tương ứng của chúng để tạo ví dụ mới có nhãn. Cụ thể, một ví dụ đào tạo mới được tạo theo và , trong đó và là các ví dụ huấn luyện, và là các nhãn trong mã hóa one – hot và phân phối của λ ∈ [0,1] được rút ra từ Beta(α, α) cho α ∈ (0, ∞). Chúng ta có thể chọn giá trị của α ∈ {0,5, 0,8, 1, 1,5} khi nó mang lại hiệu suất tốt nhất.

**Kết hợp học tập siêu chuyển giao:** Ý tưởng cốt lõi của học siêu chuyển giao là các mạng được đào tạo trên bộ dữ liệu lớn có thể học cách trích xuất các tính năng nổi bật như đốm màu và cạnh, phổ biến trên các lớp hình ảnh khác nhau. Chúng ta có thể tận dụng quá trình học siêu chuyển giao bằng cách sử dụng EfficientNet làm xương sống. Trong cài đặt tên miền chéo, chúng ta có thể thay thế các trọng số bằng các trọng số được đào tạo trước tốt nhất thu được từ quá trình đào tạo trước đó. Chúng ta cũng có thể thử nghiệm các cấu hình khác nhau để tinh chỉnh EfficientNet bằng cách đóng băng các lớp cụ thể.

**Khám phá số liệu khoảng cách:** Khi EfficientNet trích xuất các tính năng nhúng, từ đó sử dụng một lớp tuyến tính để mang lại giá trị trung bình và phương sai cho mỗi mẫu, tức là . Sau đó, một prototype của lớp *k* được biểu thị dưới dạng trung bình của giá trị trung bình và phương sai của nó.

Áp dụng khoảng cách KL **(Kullback & Leibler (1951))** để đo mức độ gần nhau giữa các mẫu truy vấn và nguyên mẫu vì KL có dạng đóng giữa hai phân bố Gaussian:

Trong đó *d* là kích thước của các nguyên mẫu (prototype). Chọn thứ nguyên tiềm ẩn *d* = 64 để thử nghiệm phù hợp với cài đặt được đề xuất trong protonet. Sau đó kết hợp với khoảng cách euclide và cosine như sau:

**Hàm Softmax:**

Trong đó *d* là thước đo khoảng cách. Khi khoảng cách euclide được sử dụng, đó là:

**Biểu diễn tuyến tính:**

Trong đó và . Vì nó tối ưu hóa nguyên mẫu , nên chúng ta có thể bỏ qua .

Ngoài khoảng cách euclide, thì cũng có một dạng khoảng cách cosin đối với các giá trị nhỏ để tránh các vấn đề về số,

Phân kỳ KL là thước đo khoảng cách tự nhiên để sử dụng giữa các bản phân phối. Chúng tôi đưa ra một dẫn xuất ngắn của một dạng tổng quát của phương trình KL. Giả sử có hai phân phối chuẩn: là và là . Chúng ta có:

Khi đẳng thức cuối cùng đúng khi chúng ta thay thế biến ngẫu nhiên bằng giá trị trung bình và bằng hiệp phương sai vì nó lấy kỳ vọng của . Để đơn giản, chúng tôi chỉ xem xét hiệp phương sai đường chéo trong quá trình thực hiện của chúng tôi.

Khi chúng ta xem xét kỹ hơn khoảng cách KL, tức là , thuật ngữ cuối cùng của phương trình trên có thể được hiểu là một dạng của khoảng cách Mahalanobis. Nếu chúng ta chỉ xem xét phần này, phương trình (2) có thể được xem như một mô hình hỗn hợp Gaussian, tức là

Trong đó và là xác suất trước của lớp *k* và *k’*. Đó là một phân loại dựa trên mức độ tương đồng mềm (soft – membership) so với phân loại dựa trên mức độ tương đồng cứng (hard – membership) sử dụng khoảng cách euclidean. Trong phân loại mềm, chúng ta đo lường mức độ tương đồng giữa một mẫu và các lớp khác nhau bằng cách sử dụng khoảng cách euclidean. Kết quả của phân loại mềm là một vectơ các giá trị dự đoán, thể hiện mức độ tương đồng giữa mẫu và các lớp. Trái lại, trong phân loại cứng, mẫu được phân loại vào một lớp duy nhất dựa trên khoảng cách euclidean với lớp gần nhất. Sự chuyển đổi từ phân loại cứng sang phân loại mềm giúp chúng ta có được thông tin chi tiết hơn về mức độ tương đồng giữa mẫu và các lớp, mang lại kết quả phân loại tốt hơn và chính xác hơn. [20]

RELATION NETWORKS

Về cơ bản, chúng ta có ba tập dữ liệu: tập huấn luyện (training test), tập hỗ trợ (support set) và tập kiểm tra (testing set). Tập hỗ trợ và tập kiểm tra chia sẻ cùng một không gian nhãn, nhưng tập huấn luyện có không gian nhãn riêng biệt với tập hỗ trợ/kiểm tra. Nếu bộ hỗ trợ chứa K ví dụ được dán nhãn cho mỗi lớp duy nhất trong số C, được gọi là C – way K – shot.

Chỉ với bộ hỗ trợ, về nguyên tắc có thể đào tạo một bộ phân loại để gán nhãn lớp cho mỗi mẫu trong bộ kiểm tra. Tuy nhiên, do thiếu các mẫu được dán nhãn trong bộ hỗ trợ, hiệu suất của bộ phân loại như vậy thường không đạt yêu cầu. Do đó, chúng tôi đặt mục tiêu thực hiện siêu học tập (meta learning) trên tập huấn luyện, để trích xuất kiến thức có thể chuyển nhượng, cho phép chúng tôi thực hiện FSL tốt hơn trên tập hỗ trợ và do đó phân loại tập kiểm tra thành công hơn.

Một cách hiệu quả để khai thác tập huấn luyện là bắt chước cài đặt FSL thông qua huấn luyện dựa trên tập. Trong mỗi lần lặp huấn luyện, một tập được hình thành bằng cách chọn ngẫu nhiên các lớp C từ tập huấn luyện với K mẫu được gắn nhãn từ mỗi lớp C để đóng vai trò là tập mẫu , cũng như một phần còn lại của các mẫu của các lớp C đó để dùng làm bộ truy vấn. Sự phân tách bộ truy vấn/bộ mẫu này được thiết kế để mô phỏng bộ hỗ trợ/kiểm tra sẽ gặp phải tại thời điểm thử nghiệm. Một mô hình được đào tạo từ bộ mẫu/truy vấn có thể được tinh chỉnh thêm bằng cách sử dụng bộ hỗ trợ, nếu muốn. Trong phần này, chúng tôi áp dụng chiến lược đào tạo dựa trên tập như vậy. Trong các thử nghiệm few – shot, chúng ta xem xét cài đặt một lần (K = 1, hình 21) và năm lần (K = 5). Chúng ta cũng giải quyết trường hợp zero – shot với K = 0 như được giải thích trong phần dưới. [21]

* Kiến trúc của Relation Networks

**One – shot**: Relation Networks (RN) bao gồm hai modules: module nhúng (Embedding module) và module quan hệ (Relation module) , như được minh họa trong hình 21. Các mẫu trong tập truy vấn Q và các mẫu trong tập mẫu S được cung cấp thông qua module nhúng , tạo ra các feature map và . Các feature map và được kết hợp với toán tử . Trong công việc này, chúng ta giả sử là sự kết hợp sâu của các feature map, mặc dù có thể có các lựa chọn khác. Feature maps kết hợp của mẫu và truy vấn được đưa vào module quan hệ , module này cuối cùng tạo ra một đại lượng vô hướng trong phạm vi từ 0 đến 1 biểu thị mức độ tương tự giữa và , được gọi là điểm quan hệ (relation score). Vì vậy, trong cài đặt C – way 1 – shot, chúng ta tạo ra các điểm quan hệ C cho mối quan hệ giữa một đầu vào truy vấn và các ví dụ về tập mẫu huấn luyện ,

A diagram of a dog

Description automatically generated with low confidence

Hình : Kiến trúc relation networks

**K – shot**: Đối với K – shot trong đó K > 1, chúng ta tính tổng các phần tử thông minh trên các đầu ra module nhúng của tất cả các mẫu từ mỗi lớp đào tạo để tạo thành feature map của lớp này. Feature map cấp lớp tổng hợp này được kết hợp với feature map hình ảnh truy vấn như trên. Do đó, điểm quan hệ cho một truy vấn luôn là C trong cả cài đặt one – shot hoặc few – shot.

**Objective function (Hàm mục tiêu):** Sử dụng hàm mất mát sai số trung bình (MSE) (Phương trình (2)) để huấn luyện mô hình của mình, hồi quy điểm quan hệ với sự thật cơ bản: các cặp phù hợp có độ tương tự 1 và cặp không khớp có độ tương tự 0.

Sự lựa chọn của MSE có phần không chuẩn. Bài toán là một vấn đề phân loại với không gian nhãn {0, 1}. Tuy nhiên, về mặt khái niệm, chúng tôi đang dự đoán điểm số của mối quan hệ, đây có thể được coi là một vấn đề hồi quy mặc dù đối với sự thật cơ bản, chúng tôi chỉ có thể tự động tạo các mục tiêu {0, 1}.

**Zero – shot learning**: Tương tự như one-shot learning trong đó một mốc thời gian được đưa ra để xác định từng lớp cần nhận dạng. Tuy nhiên, thay vì được cung cấp một bộ hỗ trợ với hình ảnh chụp một lần cho mỗi lớp đào tạo C, nó chứa một embedding vector cho mỗi lớp semantic. Việc sửa đổi khung để xử lý trường hợp zero – shot rất đơn giản: vì một phương thức khác của sematic vector được sử dụng cho bộ hỗ trợ (ví dụ: vectơ thuộc tính thay vì hình ảnh), sau đó sử dụng module nhúng (Embedding module) không đồng nhất thứ hai bên cạnh module nhúng đã sử dụng cho bộ truy vấn hình ảnh. Sau đó, mạng quan hệ được áp dụng như trước đây. Do đó, điểm quan hệ cho mỗi đầu vào truy vấn sẽ là:

Hàm mục tiêu của phương pháp học zero – shot cũng giống như phương pháp few – shot. [21]

# CHƯƠNG 3: ỨNG DỤNG



## LĨNH VỰC ỨNG DỤNG

### THỊ GIÁC MÁY TÍNH

#### PHÂN LOẠI HÌNH ẢNH

Ngoại trừ các công ty như Google và Facebook, hầu hết các nhà nghiên cứu trong cuộc sống thực không có quyền truy cập vào một bộ dữ liệu lớn chất lượng tốt. Trong các nhiệm vụ phân loại ảnh FSL trong lĩnh vực thị giác máy tính, mỗi nhiệm vụ có thể chỉ chứa một hoặc một vài mẫu. Việc giải quyết các nhiệm vụ phân loại ảnh few – shot chủ yếu được thực hiện thông qua augmentation dữ liệu, học chuyển giao, học siêu, và học kết hợp đa phương thức. Hiện nay, ba phương pháp hàng đầu về độ chính xác đều dựa trên tăng cường đặc trưng và biến đổi đặc trưng của mô hình cốt lõi. Chúng ta có thể chỉ ra một vài ví dụ về phân loại hình ảnh như: nhận dạng người, vân tay, biển báo giao thông, … [22]

#### PHÁT HIỆN ĐỐI TƯỢNG

Few – shot object detection (FSOD) là nhiệm vụ nhằm phát hiện các đối tượng hiếm từ một số mẫu. Có rất nhiều tiến bộ trong FSL cho nhiệm vụ phân loại hình ảnh, nhưng ít khi có trong việc phát hiện đối tượng. Hiện nay, sự phát triển của FSOD có thể được chia thành ba phương pháp chính: tăng cường dữ liệu, học chuyển giao và học meta. Trong số đó, cơ chế Attention đóng vai trò quan trọng trong việc phát hiện mục tiêu từ mẫu nhỏ. Tuy nhiên, vấn đề về thời gian suy luận chậm để đáp ứng yêu cầu thời gian thực cho việc phát hiện đối tượng từ một số mẫu vẫn còn nghiêm trọng. [22]

#### PHÂN ĐOẠN Ý NGHĨA

Few – shot semantic segmentation, được đề xuất lần đầu năm 2017, đã được sử dụng rộng rãi trong các lĩnh vực như hình ảnh y học và xe tự lái. Khác với phân đoạn ý nghĩa truyền thống, phân đoạn ý nghĩa few – shot có ít thông tin chú thích pixel hơn trong tập dữ liệu hỗ trợ. Hiện nay, phân đoạn ý nghĩa few – shot có thể được phân loại rộng rãi thành ba loại chính: phân đoạn ý nghĩa có giám sát, phân đoạn ý nghĩa không có giám sát và phân đoạn ý nghĩa video. Trong giai đoạn học máy, phương pháp cổ điển phổ biến hơn là sử dụng ánh xạ xác suất làm kiến thức tiền đề cho việc rút ra kết quả. Trong giai đoạn học sâu, xuất hiện nhiều thuật toán hiệu quả cho công cụ phân đoạn, nhưng những mô hình này thường yêu cầu một số lượng lớn chú thích mẫu thủ công. Gần đây cũng đã đạt được những cải tiến đáng kể trong phân đoạn ý nghĩa few – shot bằng cách đề xuất một mô hình ngắn gọn hơn, trong đó chỉ có bộ phân loại được học meta và bộ giải mã mã hóa đặc trưng vẫn được huấn luyện bằng một mô hình phân đoạn thông thường. [22]

#### PHÂN ĐOẠN ĐỐI TƯỢNG

Khác với phân đoạn ý nghĩa, phân đoạn đối tượng riêng biệt (instance segmentation) liên quan đến việc xác định từng pixel trong một hình ảnh và gán nhãn cho nó một cách riêng lẻ. Gần đây, có một số nghiên cứu đang xử lý vấn đề phân đoạn một số mẫu về các đối tượng. Công trình hiện tại vẫn tập trung vào việc cải thiện R-CNNs bằng cách sử dụng một số công cụ hiệu quả. Gần đây một số nhà nghiên cứu đã đề xuất một thuật toán phân đoạn đối tượng riêng biệt few – shot tăng dần, làm cải thiện đáng kể hiệu suất trên các tập dữ liệu benchmark. Trong phần này, chúng tôi khảo sát các bài báo trong ba năm gần đây về phân đoạn đối tượng riêng biệt few – shot. [22]

### XỬ LÝ NGÔN NGỮ TỰ NHIÊN

Khả năng học hiệu quả từ dữ liệu hạn chế hoặc không có dữ liệu là điều cần thiết để áp dụng NLP vào các nhiệm vụ mà việc thu thập dữ liệu tốn kém hoặc khó khăn. Đây là một bối cảnh thách thức cả về mặt học thuật và thực tế, đặc biệt là vì việc huấn luyện mô hình thường đòi hỏi lượng lớn dữ liệu được gán nhãn. Gần đây, những tiến bộ trong việc tiền huấn luyện trên dữ liệu không được gán nhãn đã đem lại tiềm năng cho việc học không gian hoặc học từ ít dữ liệu. Đặc biệt, trong suốt năm qua, đã có rất nhiều nghiên cứu được tiến hành để tìm hiểu cách học tốt hơn từ dữ liệu hạn chế bằng cách sử dụng các mô hình ngôn ngữ quy mô lớn. Cụ thế nhất là áp dụng vào phân loại văn bản.

Few – shot learning (FSL) nhằm mục đích học các bộ phân loại từ một vài ví dụ cho mỗi lớp. Gần đây, học sâu đã được khai thác thành công cho FSL thông qua việc học các mô hình meta từ một số lượng lớn các nhiệm vụ đào tạo meta. Sau đó, các siêu mô hình này có thể được sử dụng để thích ứng nhanh cho các nhiệm vụ kiểm tra mục tiêu/siêu dữ liệu chỉ có một vài ví dụ đào tạo. Ví dụ về các mô hình meta như vậy bao gồm: (1) các mô hình dựa trên số liệu/độ tương tự, tìm hiểu các biện pháp tương tự theo ngữ cảnh và theo nhiệm vụ cụ thể và (2) các mô hình dựa trên tối ưu hóa, nhận đầu vào của độ dốc từ tác vụ FSL và dự đoán tham số mô hình hoặc cập nhật tham số.

Trước đây, FSL chủ yếu xem xét các miền hình ảnh, trong đó tất cả các tác vụ thường được lấy mẫu từ một bộ sưu tập dữ liệu khổng lồ, chẳng hạn như Omniglot và ImageNet, tạo ra các tác vụ đến từ một do chính duy nhất có liên quan. Do cài đặt được đơn giản hóa như vậy, hầu hết tất cả các tác phẩm trước đây đều sử dụng một siêu mô hình chung (dựa trên số liệu/tối ưu hóa) cho tất cả các tác vụ ít cảnh quay. Tuy nhiên, cài đặt này khác xa với các tình huống thực tế trong nhiều ứng dụng trong thế giới thực của phân loại văn bản bằng vài lần chụp. Ví dụ: trên dịch vụ đám mây AI dành cho doanh nghiệp, nhiều khách hàng gửi các tác vụ khác nhau để đào tạo các mô hình phân loại văn bản cho các mục đích dành riêng cho doanh nghiệp. Các nhiệm vụ có thể là phân loại nhận xét hoặc ý kiến của khách hàng về các sản phẩm/dịch vụ khác nhau, theo dõi phản ứng của công chúng đối với các thay đổi chính sách khác nhau hoặc xác định ý định của người dùng trong các loại dịch vụ trợ lý cá nhân khác nhau. Vì hầu hết các khách hàng không thể thu thập đủ dữ liệu, các nhiệm vụ đã gửi của họ sẽ tạo thành một cài đặt few – shot. Ngoài ra, các tác vụ này rất đa dạng, do đó, một số liệu chung không đủ để xử lý tất cả các tác vụ này. [23]

### PHÂN LOẠI ÂM THANH

Phân loại âm thanh là một lĩnh vực nghiên cứu được nghiên cứu kỹ lưỡng với nhiều ứng dụng như tìm kiếm và truy xuất đa phương tiện, giám sát âm thanh đô thị, giám sát âm thanh sinh học và phụ đề âm thanh. Hầu hết các phương pháp phân loại âm thanh gần đây sử dụng phương pháp học tập có giám sát tiêu chuẩn được áp dụng cho các mạng thần kinh sâu. Mặc dù thành công, cách tiếp cận này có hai nhược điểm đáng kể: (1) nó yêu cầu số lượng lớn dữ liệu được dán nhãn và (2) chỉ có thể phát hiện các lớp được bao gồm trong những dữ liệu này, nghĩa là nó áp đặt một từ vựng lớp cố định. Những yêu cầu này, mặc dù có vẻ vô hại, nhưng có thể làm cho phần lớn các phương pháp phân loại âm thanh không sử dụng được cho các ứng dụng mà từ vựng của lớp mục tiêu không được biết trước. Tức là, nhiều tình huống trong thế giới thực yêu cầu chúng ta tùy chỉnh từ vựng của lớp, chẳng hạn như thêm các lớp mới, chẳng hạn để cá nhân hóa các từ đánh thức trên thiết bị thông minh, để theo dõi các loài chim mới ở các địa điểm khác nhau hoặc để phiên âm các loài chim quý hiếm. nhạc cụ.

Thay vào đó, FSL đã được áp dụng để phân loại âm thanh và phát hiện sự kiện âm thanh, trong đó bộ phân loại phải học cách nhận ra một lớp mới từ rất ít ví dụ. Trong số các phương pháp học vài lần khác nhau, các mạng nguyên mẫu dựa trên số liệu đã được chứng minh là mang lại hiệu suất tuyệt vời cho âm thanh. Tuy nhiên, các phương pháp rút ngắn không duy trì từ vựng của lớp dữ liệu huấn luyện, yêu cầu ghi nhãn thủ công cho tất cả các lớp mới để triển khai, điều này có thể gây quá tải đối với các vấn đề về từ vựng lớn.

Ngược lại, continual learning (học không ngừng, học tăng dần, học suốt đời) là một chiến lược học trực tuyến trong đó một thuật toán tìm cách liên tục thích ứng với một chuỗi các nhiệm vụ và thực hiện tốt tất cả các nhiệm vụ mà không bị quên. Nó đã được đề xuất để phân loại âm thanh và phát hiện sự kiện âm thanh để học các sự kiện âm thanh mới mà không quên những sự kiện đã học trước đó. Tuy nhiên, các phương pháp học tập liên tục thường yêu cầu đào tạo lại khi giới thiệu các lớp mới, quy trình đào tạo phức tạp hoặc lượng lớn dữ liệu được gắn nhãn của các lớp mới, không lý tưởng cho ứng dụng thực tế với môi trường máy tính hoặc miền âm thanh bị hạn chế về tài nguyên.

Gần đây, lĩnh vực nghiên cứu mới về few – shot continual learning (học gia tăng với một vài ví dụ) kết hợp các điểm mạnh của các phương pháp đã nói ở trên và nhằm mục đích liên tục mở rộng khả năng của bộ phân loại chỉ dựa trên một vài dữ liệu tại thời điểm suy luận. Điều này cho phép người dùng cuối cập nhật mô hình nhanh chóng và tương tác. [24]

### ROBOTICS

Kể từ giữa những năm 1980, lập trình bằng trình diễn (PbD) đã nổi lên như một chủ đề nghiên cứu đầy hứa hẹn trong chế tạo người máy do những ưu điểm tương đối của nó so với các phương pháp truyền thống. Mối quan tâm ngày càng tăng đối với các thuật toán học sẽ trang bị cho rô-bốt học bằng cách bắt chước hành động của cả con người và các tác nhân rô-bốt khác đã dẫn đến nhiều hướng nghiên cứu trong lĩnh vực này. Ví dụ, một mô hình phân cấp đã được đề xuất trong lĩnh vực tương tác giữa người và rô-bốt (HRI) để thu hút sự chú ý và nhận thức, trong khi một số nghiên cứu khác tập trung vào các phương pháp giảng dạy kỹ năng sử dụng một loạt các kỹ thuật. Nhiều hướng nghiên cứu trong số này có thể được gói gọn trong 5 chữ “W” của bắt chước - cụ thể là **W**ho, **W**here, **W**hen, **W**hat và Ho**w**. Thông thường, rô-bốt phải mất nhiều thử nghiệm học tập để khái quát hóa từng khía cạnh trừu tượng này trong một hành động.

Trong nỗ lực giảm số lượng thử nghiệm liên quan đến việc học, chúng tôi đề xuất một thuật toán để giải quyết câu hỏi “làm thế nào để” bắt chước. Đó là FSL cụ thể là sử dụng OSL, một lĩnh vực thích hợp phổ biến trong học máy có thể giúp giải quyết vấn đề này. Các tác giả lập luận rằng các mô hình khái quát hóa các hành vi được quan sát thành một tập hợp các tham số mô hình phức tạp nội tại sẽ hạn chế khả năng tương tác của người dùng sau khi trình diễn. Tuy nhiên, nhiều mô hình tiên tiến nhất trong việc bắt chước con đường ủng hộ những mô hình như vậy. Chẳng hạn, trong các thuật toán sử dụng mô hình hỗn hợp Gaussian, các tham số tổng quát là các trọng số với các tham số Gaussian liên quan. Các thuật toán này ngăn cản sự tương tác/can thiệp trực tiếp của người dùng, vì ý nghĩa trừu tượng của các tham số này không dễ dàng thao túng bởi người dùng. Ví dụ, khi người trình diễn vô tình thực hiện sai động tác, thay vì sửa các thông số của mô hình, người dùng phải thực hiện lại tất cả các động tác hoặc thực hiện nhiều động tác đúng hơn. Để giải quyết các vấn đề trên và câu hỏi “làm thế nào”, một vấn đề bắt chước đường dẫn có thể được coi là một sự điều chỉnh kế hoạch trong đó đường dẫn được minh họa với một tập hợp các đặc điểm môi trường được chiếu vào một không gian cong vênh. Dựa trên việc duy trì mối quan hệ không gian giữa các tính năng bắt chước và ban đầu trong không gian đó, một đường dẫn bắt chước có thể được tạo ra. Cách biến một cảnh trong bối cảnh hiện tại thành một cảnh mới như vậy tồn tại trong cả lĩnh vực xử lý ảnh và chuyển thể kế hoạch chuyển động, trong đó năng lượng liên quan đến việc biến hình được giảm thiểu. [25]

### CHĂM SÓC SỨC KHỎE

Sự phát triển và áp dụng lợi ích của deep learning không chỉ giới hạn trong lĩnh vực khoa học máy tính mà còn đã phát triển mạnh mẽ trong các lĩnh vực khác, trong đó y sinh học đứng đầu. Các thuật toán deep learning được áp dụng trong nhiều phần khác nhau của y học như dự đoán hình ảnh, phát hiện bệnh nhanh chóng và phân tích hình ảnh y sinh học. Tiến bộ của deep learning đã được thúc đẩy theo cách mà nó mang lại khả năng dự đoán tốt hơn so với độ chính xác của con người đối với các nhiệm vụ phân loại. Phân tích hình ảnh y sinh học đòi hỏi hiểu biết phức tạp và nghiên cứu chi tiết về các đặc trưng được trích xuất từ hình ảnh. Ngay cả các chuyên gia đã được đào tạo cũng gặp khó khăn khi hiểu được những hình ảnh này. Quá trình thủ công như nhận dạng lớp của khối u não hoặc phân đoạn và chú thích các cơ quan của cơ thể con người đòi hỏi nhiều năm kinh nghiệm, kiến thức và thực hành trước khi ai đó có thể trở thành một chuyên gia trong lĩnh vực này. Tuy nhiên, tiến bộ hiện đại của hệ thống deep learning đã làm cho việc này dễ dàng hơn trước đây vì nó có thể trích xuất đặc trưng nhanh hơn và học từ chúng để tạo ra một biểu đồ nội bộ của các đặc trưng này và đưa ra dự đoán. Xử lý của mô hình, mục tiêu và độ chính xác cũng luôn được so sánh với độ chính xác của con người.

Rõ ràng rằng việc áp dụng deep learning đang tăng nhanh trong lĩnh vực y tế. Tuy nhiên, một số hạn chế trở thành rào cản đối với việc ứng dụng deep learning trong lĩnh vực xử lý hình ảnh y tế. Trong số đó, sự khan hiếm dữ liệu được gán nhãn và không nhất quán về lớp được coi là một trong những phụ thuộc lớn nhất của quá trình deep learning. Độ chính xác và độ chính xác của mô hình deep learning phụ thuộc lớn vào số lượng lớn dữ liệu huấn luyện. Vấn đề này có thể không tạo ra vấn đề lớn đối với nhiều trường hợp y tế, nhưng có một số trường hợp hiếm gặp mà số lượng dữ liệu có thể không có sẵn do tính bảo mật. Việc tìm kiếm dữ liệu có sẵn cho một số trường hợp hiếm gặp là khó khăn, điều này cuối cùng không đưa ra kết quả đáng tin cậy đối với phân tích hình ảnh deep learning trong lĩnh vực y tế. Ngay cả khi không có đảm bảo rằng việc thu thập dữ liệu sẽ có hiệu lực và bền vững cho mô hình trong quá trình huấn luyện, vấn đề phụ thuộc dữ liệu của deep learning phải được giải quyết theo cách xử lý của con người với ít dữ liệu.

FSL là phiên bản phát triển của deep learning với nguồn dữ liệu hạn chế để tổng quát hóa các lớp dữ liệu toàn bộ. Phương pháp này hoạt động bằng cách trích xuất các đặc trưng dữ liệu nguyên bản để phân biệt các lớp. Mạng nơ-ron tích chập hoàn toàn phụ thuộc vào việc miêu tả một tập dữ liệu huấn luyện lớn hơn. Vấn đề của việc miêu tả tổng quát mạng nơ-ron tích chập biến thành một bài toán cụ thể với sự giúp đỡ của FSL.

Các con đường FSL đang mở rộng do đáng tin cậy của dữ liệu huấn luyện hạn chế. Một thách thức đáng kể khác trong lĩnh vực y tế là theo dõi các vấn đề tiêu hóa, thận và gan thông qua Endoscopy. Tuy nhiên, các tập dữ liệu lớn từ các nguồn lâm sàng khác nhau có thể tạo ra một sự thiên lệ rộng lớn, làm cho mô hình không thể sử dụng được. Tuy nhiên, đã được giới thiệu một mô hình metric margin góc góp mới, là một khung cảnh được sửa đổi của mạng prototypical, để phân loại các mẫu kiểm tra Endoscopy từ một tập dữ liệu không nhìn thấy được huấn luyện với ít dữ liệu huấn luyện. Ngay cả các phương pháp FSL khác như Reptile dựa trên gradient và mạng prototypical dựa trên khoảng cách metric cũng được áp dụng để xác định bộ dữ liệu hình ảnh tổn thương da. Phương pháp được đề xuất được gọi là mạng meta – derm diagnosis. Yêu cầu của dữ liệu hạn chế khiến cho FSL trở nên nổi bật trong thời đại deep learning này. Các chiều mới của ứng dụng FSL như y học mắt đã được khởi đầu thông qua việc thực hành FSL. Mặc dù DL được áp dụng trong các trường hợp lâm sàng bệnh lý, có một số trường hợp hiếm gặp như viêm màng não hoặc bệnh thần kinh quá trình gây thiệt hại trước mắt không thể thực hiện các phương pháp học tập few-shot do sự khan hiếm của các tập dữ liệu huấn luyện. Mạng nơ-ron tích chập (CNN) kết hợp với công cụ t-SNE visualization được tích hợp trong đường cong ROC (AUC) lớn hơn 0 đã tạo ra kết quả tốt hơn so với các khung viền khác.

Trong việc cải thiện chẩn đoán lâm sàng và kế hoạch điều trị, việc phân đoạn hình ảnh là một nhiệm vụ không thể tránh được để chú thích một lượng lớn các tập dữ liệu huấn luyện. Một phân đoạn few – shot đã được đề xuất với một cấu trúc mới để giải quyết tình trạng khó khăn. Medical prior – based few – shot learning network (MPrNet) và interactive learning – based test time optimization algorithm (IL-TTOA) để hỗ trợ hình ảnh truy vấn và mục đích tương tác nhằm giảm thiểu sự phiền toái của các chuyên gia y tế và thử nghiệm kết quả của mô hình này với tập dữ liệu lâm sàng con người thực tế. Do đó, một mô hình phân đoạn khác được đề xuất là mạng tương quan toàn cục với nhúng phân biệt (GCN-DE) để nhanh chóng tổng quát hóa các tập dữ liệu không nhìn thấy với tính nhất quán không gian và quy luật. Phương pháp này có thể giảm độ phức tạp tính toán bằng cách tạo ra các lớp tích chập giữa hình ảnh hỗ trợ và hình ảnh truy vấn trong không gian đặc trưng. Hơn nữa, phân đoạn hình ảnh thông qua FSL được áp dụng một cách toàn diện trong các tập dữ liệu lâm sàng da và polyp với thuật toán học meta iMAML không xác định mô hình, môi trường chú ý và gradient conjugate (độ dốc liên hợp) để đạt được kết quả mong đợi. Đóng góp cách mạng trong phân đoạn hình ảnh tiếp tục thông qua nghiên cứu về phân đoạn hình ảnh y tế 3D, trong đó mạng U – Net, một mô hình để tính toán mối quan hệ giữa các hình ảnh không nhìn thấy dựa trên phân đoạn hình ảnh 2D. [26]

## TÌNH HÌNH PHÁT TRIỂN

Hiện nay, FSL đang là một trong những lĩnh vực nghiên cứu đang được quan tâm rất nhiều trong cộng đồng học máy và trí tuệ nhân tạo. Các nhà nghiên cứu đang nghiên cứu và phát triển nhiều phương pháp và kỹ thuật mới để cải thiện độ chính xác và khả năng áp dụng của FSL. Các phương pháp FSL được áp dụng rộng rãi trong các lĩnh vực như nhận dạng hình ảnh, xử lý ngôn ngữ tự nhiên, robot học, và nhiều lĩnh vực khác. Ví dụ, trong lĩnh vực nhận dạng hình ảnh, FSL được sử dụng để phân loại các đối tượng mới trong các tình huống thực tế, nơi mà việc có được số lượng dữ liệu huấn luyện lớn để huấn luyện một mô hình truyền thống là không khả thi hoặc rất tốn kém.

Trong thập niên gần đây, nghiên cứu về FSL đã được đại khám phá rộng và đã đạt được những tiến bộ nghiên cứu quan trọng, ví dụ: KGBer do Alibaba đề xuất lần đầu tiên thông qua con người trong lĩnh vực FSL.

Hình 22 có thể hiển thị số liệu thống kê của các ấn phẩm giấy liên quan đến FSL trong mười năm gần đây dựa trên kết quả thống kê của Web of Science. Theo tiết lộ, có khá ít tài liệu liên quan từ năm 2011 đến năm 2015 lý thuyết về FSL vẫn chưa hoàn chỉnh. Với sự gia tăng của nghiên cứu sâu kể từ năm 2015, số lượng tài liệu nghiên cứu liên quan đến FSL đã tăng tuyến tính gần như hàng năm. Trong năm 2020 vừa qua, số lượng bài báo liên quan đã lên tới 239 và số lần trích dẫn không trích dẫn lên tới 2731 lần. Các chủ đề nghiên cứu nóng và những phát triển tiên tiến trong lĩnh vực FSL trong những năm gần đây, bao gồm ở zero-shot learning, one-shot learning, transfer learning, multi-task learning, and meta-learning. Các nhiệm vụ liên quan đến FSL bao gồm biểu diễn thuộc tính (feature), trực quan hóa, robotics và chuyển miền. Trong số đó, thích ứng miền (domain adaptation) là một phương pháp được sử dụng rộng rãi cho việc học liên miền. Thị giác máy tính chiếm ưu thế là lĩnh vực nghiên cứu tích cực nhất, bao gồm phân loại hình ảnh, phát hiện đối tượng, phân đoạn ngữ nghĩa và phân đoạn thể hiện.

A picture containing plot, line, diagram, screenshot

Description automatically generated

Hình : Số lượng bài báo liên quan đến FSL được công bố trên các tạp chí uy tín từ năm 2010 đến nửa đầu năm 2021, không bao gồm các trích dẫn [3]

# CHƯƠNG 4: TRIỂN KHAI ỨNG DỤNG



## MỤC TIÊU HƯỚNG TỚI

Nghiên cứu và hiểu được thuật toán prototypical networks, một phương pháp hiệu quả để triển khai FSL dựa trên khái niệm của các vector trung tâm cho mỗi lớp.

Thử nghiệm và tinh chỉnh mô hình trên các tập dữ liệu phổ biến phù hợp với mô hình FSL. Cụ thể chúng ta sẽ sử dụng tập dữ liệu Omniglot để tinh chỉnh tham số, cải thiện mô hình. Đảm bảo mô hình có khả năng học và phân loại đối tượng từ ít thông tin huấn luyện và đánh giá hiệu suất của mô hình trên các tập dữ liệu thử nghiệm.

Sử dụng mô hình prototypical networks để xây dựng hệ thống phân loại ký tự trong trường hợp có ít dữ liệu huấn luyện. Ứng dụng gồm giao diện người dùng, cho phép người dùng tương tác cũng như nó sẽ tích hợp các chức năng xử lý ảnh và dữ liệu do người dùng đưa vào.

## DỮ LIỆU

Trong quá trình đào tạo với N – way K – shot thì tập hỗ trợ sẽ bao gồm: N lớp và mỗi lớp sẽ có K mẫu. Ví dụ: trong phân loại hình ảnh phổ biến là 5 – way 1 – shot, thì một bộ hỗ trợ sẽ có 5 loại và chỉ có 1 ví dụ cho mỗi loại tương ứng. Và trong phần triển khai ứng dụng này chúng ta sẽ sử dụng tập dữ liệu Omniglot để đào tạo cho mô hình. Tập dữ liệu Omniglot bao gồm hình ảnh chữ cái từ nhiều nguồn và ngôn ngữ khác nhau. Mỗi ký tự được viết tay và biểu diễn dưới dạng hình ảnh grayscale. Có tổng cộng hơn 1.600 ký tự khác nhau trong tập dữ liệu. Mỗi ký tự được lấy từ một nguồn khác nhau và có 20 biến thể khác nhau cho mỗi ký tự, các kí tự này được viết tay từ những người khác nhau. Kích thước của hình ảnh ký tự nhỏ, thường là 105 ×105 pixel.

Hình 23, 24 là hình ảnh của một số ký tự thuộc bộ ký tự Japanese (katakana) và Latin có trong tập dữ liệu được sử dụng:

A picture containing font, handwriting, typography, design

Description automatically generated

Hình : Một số ký tự thuộc bộ ký tự Japanese\_(katakana)

A picture containing font, white, typography, design

Description automatically generated

Hình : Một số ký tự thuộc bộ ký tự Latin

## MÔ HÌNH VÀ THUẬT TOÁN

Đầu tiên chúng ta sẽ tiến hành xử lý dữ liệu: Chuyển đổi hình ảnh ký tự sang định dạng phù hợp (tensor) để mô hình có thể xử lý được. Sau đó chuẩn hóa kích thước hình ảnh để đảm bảo tính nhất quán trong quá trình huấn luyện.

Xây dựng mô hình prototypical networks:

* Sử dụng mạng nơ-ron tích chập (CNN) để học các đặc trưng từ hình ảnh ký tự. Sau đó sử dụng lớp pooling để trích xuất đặc trưng quan trọng từ ảnh.
* Xây dựng mô hình prototypical networks bằng cách tính toán các vector trung tâm cho mỗi lớp ký tự dựa trên các mẫu huấn luyện. Sử dụng hàm khoảng cách euclidean để đo độ tương đồng giữa mỗi mẫu cần dự đoán và các vector trung tâm của mỗi lớp (hình 26).
* Phân loại mẫu dựa trên khoảng cách tối thiểu từ mẫu đến các vector trung tâm của các lớp ký tự.

Đào tạo mô hình và đánh giá:

* Chia tập dữ liệu Omniglot thành tập huấn luyện và tập kiểm tra
* Huấn luyện mô hình prototypical networks trên tập huấn luyện, trong một chu kỳ đào tạo sẽ chọn n lớp và k hình ảnh từ mỗi lớp này (k thường bé hơn hoặc bằng 10) để tính toán và cập nhật tham số của mô hình.
* Đánh giá hiệu suất của mô hình trên tập kiểm tra bằng cách tính toán độ chính xác (accuracy) trên tập kiểm tra.

## TRIỂN KHAI ỨNG DỤNG

Đầu tiên chúng tôi sẽ xử lý đầu vào dữ liệu, dữ liệu của chúng tôi sẽ trải qua nhiều chuyển đổi (hình 25), và đầu ra của các hình ảnh sẽ là tensor kiểu dữ liệu phù hợp cho việc đào tạo mô hình của chúng tôi.

A picture containing text, screenshot, font, line

Description automatically generated

Hình : Xử lý dữ liệu đầu vào [27]

A screenshot of a computer program

Description automatically generated with low confidence

Hình : Mô hình prototypical networks[27]

Mô hình sẽ có backbone là một mô hình khác (cụ thể trong ứng dụng này là *ResNet – 18* trong thư viện torch) dùng để trích xuất các đặc trưng của dữ liệu. Từ các đặc trưng này kết hợp với việc tính toán vectơ đặc trưng trung bình của các lớp cần phân loại thì chúng tôi sẽ tính toán khoảng cách euclidean để đưa ra dự đoán là lớp mà hình ảnh cần dự đoán có khoảng cách nhỏ nhất so với vectơ đặc trưng trung tâm.

Thực hiện đào tạo mô hình trên tập Omniglot để điều chỉnh các tham số cho phù hợp với dữ liệu ở hình 27.

A picture containing text, screenshot, font

Description automatically generated

Hình : Thực hiện đào tạo mô hình [27]

Lưu model chúng ta để tái sử dụng cho việc phân loại ký tự (hình 28).



Hình : Lưu mô hình

## KẾT QUẢ

### THỬ NGHIỆM MÔ HÌNH VỚI CÁC GIÁ TRỊ KHÁC NHAU

Thử nghiệm trên một số thay đổi N – way K – shot cho ra độ chính xác như sau:

5 – way 1 – shot (hình 29):

A picture containing text, screenshot, font, number

Description automatically generated

Hình : 5 – way 1 – shot.

5 – way 5 – shot (hình 30):

A picture containing text, font, screenshot

Description automatically generated

Hình : 5 – way 5 – shot.

5 – way 10 – shot (hình 31):

A close-up of a computer code

Description automatically generated

Hình : 5 – way 10 – shot.

10 – way 10 – shot (hình 32):

A screenshot of a computer

Description automatically generated

Hình : 10 – way 10 – shot.

A graph with blue dots and numbers

Description automatically generated

Hình : Biểu đồ độ chính xác theo số lần training

Hình 33: (biểu đồ) ở trên mô tả sự thay đổi độ chính xác qua số episode, qua quan sát thì mô mình ít cải thiện về độ chính xác với số episode khoảng 15000.

Như quan sát thì chúng ta càng cung cấp nhiều hình ảnh cho mỗi lớp thì độ chính xác sẽ càng được cải thiện. Ở kết quả bên trên thì độ chính xác khi chúng ta cung cấp 10 hình ảnh với 5 hình ảnh là không cải thiện nhiều, nên chúng ta sẽ chọn với 5 lớp thì sẽ cần 5 hình để mô hình đạt được kết quả tốt, với độ chính xác hơn 97%.

### ỨNG DỤNG WEB

Chúng tôi sẽ sử dụng mô hình đã lưu trước đó để nhận dạng cho các kí tự bằng một giao diện web sử dụng flask – một python module để phát triển các ứng dụng web.

Hình 32 là giao diện trang web của chúng tôi sau khi được triển khai:

A screenshot of a computer

Description automatically generated

Hình : Giao diện web

Nhấp vào nút “**SELECT IMAGE**” để chọn ảnh cho việc nhận dạng, các ảnh này phải có trong tập hỗ trợ và nên có tỷ lệ 1:1 để tránh ảnh bị biến dạng sau khi chuyển đổi ảnh.

A screenshot of a computer

Description automatically generated

Hình : Chọn một ký tự để nhận diện

Sau khi chọn ảnh đầu vào (hình 33) thì trang web sẽ trả về kết quả nhận dạng như hình 34:

A screenshot of a computer

Description automatically generated with medium confidence

Hình : Nhận diện thành công

A screenshot of a video game

Description automatically generated

Hình : Giao diện web cho phép người dùng vẽ ký tự.

Hình 35 là trang web nhóm đã phát triển thêm chức năng cho phép người dùng vẽ kỹ tự mình cần phân loại lên trang web. Ngoài ra trang web cũng cho phép người dùng thêm một ký tự mới để phân loại ảnh.

A screen shot of a computer

Description automatically generated

Hình : Kết quả phân loại ảnh từ ký tự người dùng vẽ.

Hình 36 là kết quả phân loại ảnh của mô hình sau khi nhận ký tự của người dùng vừa vẽ.

# CHƯƠNG 5: KẾT LUẬN



## KẾT QUẢ ĐẠT ĐƯỢC

### Ý NGHĨA KHOA HỌC

Báo cáo đã đưa ra một chủ đề quan trọng trong lĩnh vực học máy và trí tuệ nhân tạo đó là việc khi mà dữ liệu đang trở nên ngày càng quan trọng và việc thu thập dữ liệu đang ngày một khó khăn hơn. Vì lẽ đó FSL đã ra đời để khắc phục về việc thiếu hụt dữ liệu hoặc không có. Nội dung của báo cáo là đưa ra cơ sở lý thuyết và những giải pháp về việc muốn giải quyết một bài toán về học máy khi không có quá nhiều dữ liệu đi kèm mà không làm ảnh hưởng đến kết quả đào tạo của mô hình. Thông qua đề tài này, chúng tôi nằm bắt được khái niệm của FSL, cách hoạt động và triển khai mô hình nhận dạng chữ viết thông qua thuật toán prototypical networks. Thông qua việc triển khai mô hình, chúng tôi cũng tăng thêm hiểu biết, kỹ năng sử dụng python và các thư viện hỗ trợ cho việc triển khai mô hình. Bên cạnh đó, chúng tôi còn nâng cao thêm được khả năng đọc hiểu tài liệu, khả năng làm việc nhóm và khả năng trình bày báo cáo khoa học.

### Ý NGHĨA THỰC TIỄN

Chúng tôi hiểu được tầm quan trọng của FSL đối với trí tuệ nhân tạo ngày nay và cuộc sống. Chúng tôi biết được nhiều phương pháp, mô hình để diễn giải được các thuật toán học máy và ứng dụng được vào bài toán cụ thể. Thông qua việc thực hiện đề tài, chúng tôi biết được việc diễn giải học máy đang trở thành một trong những hướng nghiên cứu rất phát triển đặc biệt là tại các doanh nghiệp. Khả năng diễn giải của mô hình có nhiều ý nghĩa trong các lĩnh vực như thị giác máy tinh, xử lý ngôn ngữ tự nhiên, xử lý âm thanh, chăm sóc y tế, ... Trong các lĩnh vưc này, việc giải thích tại sao mô hình đưa ra những quyết định mà con người dùng phải mất vài năm nghiên cứu, hoặc các bác sỹ đưa ra các phương pháp điều trị bệnh kịp thời cho người bệnh qua việc dự đoán bệnh tình của FSL mà không cần quá nhiều dữ liệu về căn bệnh đó. Bên cạnh đó, sau khi thực hiện xây dựng mô hình phân loại chữ viết kết hợp FSL giúp chúng tôi biết nhiều hơn về thư viện pytorch, biết rõ hơn về cách thức hoạt động của mô hình CNN trong phân loại hình ảnh.

## HẠN CHẾ

Do sự hạn chế về nguồn lực và thời gian, chúng tôi tập trung vào nghiên cứu lý thuyết của FSL hơn. Về phần triển khai mô hình thực tế trên tập dữ liệu Omniglot, chúng tôi cũng gặp nhiều vấn đề khi triển khai mô hình vì chưa có nhiều tài liệu cho việc triển khai cụ thể cũng như xung đột giữa các phiên bản khi cài đặt các module trong python. Do giới hạn về cơ sở vật chất, không có một cấu hình máy đủ tốt nên nhóm chúng tôi chỉ thử nghiệm trên một vài siêu tham số. Chưa có kinh nghiệm về giao diện app bằng python nên giao diện chưa được tối ưu. Giới hạn kiến thức toán học và khả năng đọc hiểu tiếng anh cũng là một yếu tố cản trở việc nghiên cứu của chúng tôi.

## HƯỚNG PHÁT TRIỂN

Lý thuyết được trình bày trong báo cáo có thể được áp dụng cho một số lĩnh vực nhất định trong việc giải quyết các bài toán phân loại như nhận dạng khuôn mặt, xử lý ngôn ngữ tự nhiên, xử lý âm thanh, chăm sóc y tế, ...

Báo cáo cũng ra nhiều hướng nghiên cứu trong tương lai như:

* Nghiên cứu sâu hơn về các thuật toán khác được sử dụng trong FSL.
* Nghiên cứu về các thuật toán DL khác sử dụng cho việc xử lý và phân tích hình ảnh.

Về phần thực nghiệm, ta có thể cải thiện giao diện của ứng dụng nhận dạng hoặc là nâng cấp nó lên trở thành một ứng dụng game tập viết, nhận chữ cho những trẻ em nhỏ. Có thể thử nghiệm nhiều thuật toán khác hơn, dùng thêm các bộ siêu tham số hơn để cải thiện độ chính xác của mô hình. Ngoài ra có thể nghiên cứu áp dụng FSL cho một ứng dụng hữu ích, gần gũi hơn trong các lĩnh vực như y tế, đời sống, v.v.

# TÀI LIỆU THAM KHẢO

|  |  |
| --- | --- |
| [1] | Y. W. e. al., "Generalizing from a Few Examples: A Survey on Few-Shot," 2020, pp. 1 - 2. |
| [2] | V. Lyashenko, "Understanding Few-Shot Learning in Computer Vision: What You Need to Know," neptune.ai, 21 / 4 / 2023. [Online]. Available: https://neptune.ai/blog/understanding-few-shot-learning-in-computer-vision. [Accessed 12 / 6 / 2023]. |
| [3] | Y. S. e. al., "A Comprehensive Survey of Few-shot Learning: Evolution, Applications, Challenges, and Opportunities," 2022, pp. 4 - 5. |
| [4] | D. Pham, "Giới thiệu Few-shot Learning trong Computer Vision," FPT Telecom, 20 / 12 / 2022. [Online]. Available: https://blog.cads.live/gioi-thieu-few-shot-learning-trong-computer-vision/. [Accessed 12 / 5 / 2023]. |
| [5] | "Deep Learning Tutorial," javatpoint, [Online]. Available: https://www.javatpoint.com/deep-learning. [Accessed 11 / 6 / 2023]. |
| [6] | "What is Deep Learning?," NetApp, [Online]. Available: https://www.netapp.com/artificial-intelligence/what-is-deep-learning/. [Accessed 11 / 6 / 2023]. |
| [7] | "Introduction to Convolution Neural Network," geeksforgeeks, [Online]. Available: https://www.geeksforgeeks.org/introduction-convolution-neural-network/. [Accessed 12 / 6 / 2023]. |
| [8] | T. M. Cường, "Convolution - Tích chập giải thích bằng code thực tế," TechMaster, 2 / 9 / 2019. [Online]. Available: https://techmaster.vn/posts/35474/convolution-tich-chap-giai-thich-bang-code-thuc-te. [Accessed 12 / 6 / 2023]. |
| [9] | "What are convolutional neural networks?," IBM, [Online]. Available: https://www.ibm.com/topics/convolutional-neural-networks. [Accessed 12 / 6 / 2023]. |
| [10] | M. Mandal, "Introduction to Convolutional Neural Networks (CNN)," 1 / 5 / 2021. [Online]. Available: https://www.analyticsvidhya.com/blog/2021/05/convolutional-neural-networks-cnn/. [Accessed 14 / 6 / 2023]. |
| [11] | S. A. Afshine Amidi, "Convolutional Neural Networks cheatsheet," Stanford, [Online]. Available: https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-convolutional-neural-networks. [Accessed 12 / 6 / 2023]. |
| [12] | T. D. Thang, "Giới thiệu mạng ResNet," VIBLO, 14 / 1 / 2020. [Online]. Available: https://viblo.asia/p/gioi-thieu-mang-resnet-vyDZOa7R5wj. [Accessed 5 / 6 / 2023]. |
| [13] | "Đắm Mình Vào Học Sâu," [Online]. Available: https://d2l.aivivn.com/chapter\_convolutional-modern/resnet\_vn.html. [Accessed 5 / 6 / 2023]. |
| [14] | L. S. G. S. P. W. Zi, "Few-Shot Learning & Meta-Learning | Tutorial," BOREALIS AI, 30 3 2023. [Online]. Available: https://www.borealisai.com/research-blogs/tutorial-2-few-shot-learning-and-meta-learning-i/. [Accessed 12 6 2023]. |
| [15] | G. K. e. al., "Siamese Neural Networks for One-shot Image Recognition," pp. 3 - 5. |
| [16] | N. A. Elad Hoffer, "Deep Metric Learning Using Triplet Networks," 2018, pp. 2 - 3. |
| [17] | O. V. e. al., "Matching Networks for One Shot Learning," 2017, pp. 2 - 4. |
| [18] | "K - Means Clustering," Wikipedia, [Online]. Available: https://en.wikipedia.org/wiki/K-means\_clustering. [Accessed 20 / 5 / 2023]. |
| [19] | J. S. e. al., "Prototypical Networks for Few-shot Learning," 2017, pp. 2 - 3. |
| [20] | X. Chen, "Enhancing Prototypical Networks for Few-Shot," pp. 2 - 9. |
| [21] | F. S. e. al, "Learning to Compare: Relation Network for Few-Shot Learning," 2018, pp. 3 - 4. |
| [22] | Y. S. e. al., "A Comprehensive Survey of Few-shot Learning: Evolution, Applications, Challenges, and Opportunities," 2022, p. 15. |
| [23] | "Diverse Few-Shot Text Classification with Multiple Metrics," New York, 2018, pp. 1 - 2. |
| [24] | Y. W. e. al., "Few - Shot Continual Learning For Audio Classificaion," New York, 2021, p. 1. |
| [25] | "Towards One Shot Learning by Imitation for Humanoid Robots," 2010, p. 1. |
| [26] | J. N. e. al., "Few Shot Learning For Medical Imaging: A Comparative Analysis Of Methodologies And Formal Mathematical Framework," 2023, pp. 2 - 3. |
| [27] | E. Bennequin, "Your Own Few-Shot Classification Model Ready in 15mn with PyTorch," SICARA. [Online]. [Accessed 14 / 6 / 2023]. |