

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
ĐẠI HỌC CÔNG NGHỆ THÔNG TIN  
KHOA KHOA HỌC MÁY TÍNH**

**LÊ XUÂN TÙNG  
PHẠM NGUYỄN XUÂN TRƯỜNG**

**KHÓA LUẬN TỐT NGHIỆP**

**NGHIÊN CỨU PHƯƠNG PHÁP NHẬN DẠNG VĂN BẢN NGHỆ  
THUẬT TRONG ẢNH**

**A STUDY ON METHOD FOR RECOGNIZING ARTISTIC TEXT**

**CỦ NHÂN NGÀNH KHOA HỌC MÁY TÍNH**

**TP. HỒ CHÍ MINH, 2023**

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
ĐẠI HỌC CÔNG NGHỆ THÔNG TIN  
KHOA KHOA HỌC MÁY TÍNH**

**LÊ XUÂN TÙNG – 20520347  
PHẠM NGUYỄN XUÂN TRƯỜNG - 20520835**

**KHÓA LUẬN TỐT NGHIỆP**

**NGHIÊN CỨU PHƯƠNG PHÁP NHẬN DẠNG VĂN BẢN NGHỆ  
THUẬT TRONG ẢNH**

**A STUDY ON METHOD FOR RECOGNIZING ARTISTIC TEXT**

**CỬ NHÂN NGÀNH KHOA HỌC MÁY TÍNH**

**GIẢNG VIÊN HƯỚNG DẪN**

**TS. NGÔ ĐỨC THÀNH  
ThS. ĐỖ VĂN TIẾN**

**TP. HỒ CHÍ MINH, 2023**

## **DANH SÁCH HỘI ĐỒNG BẢO VỆ KHÓA LUẬN**

Hội đồng chấm khóa luận tốt nghiệp, thành lập theo quyết định số 154/QĐ-DHCNTT ngày 01/03/2023 của Hiệu trưởng Trường Đại học Công nghệ Thông tin.

1. ..... - Chủ tịch.
2. ..... - Thư ký.
3. ..... - Uỷ viên.

## **LỜI CẢM ƠN**

Lời đầu tiên, em xin chân thành cảm ơn thầy Ngô Đức Thành và thầy Đỗ Văn Tiến, hai người thầy đã tận tâm hướng dẫn em trong suốt quá trình thực hiện khoá luận. Không chỉ giúp em vượt qua khó khăn gặp phải khi thực hiện khoá luận, khi làm việc với thầy, em còn được học những kiến thức, kỹ năng quan trọng giúp ích cho công việc của mình sau này.

Em cũng xin gửi lời cảm ơn đến các bạn thuộc CLB AI Club - Trường ĐH Công nghệ Thông tin - ĐHQG TP.HCM đã hỗ trợ em trong việc xây dựng và hoàn thiện tập dữ liệu.

Cuối cùng, em xin gửi lời cảm ơn đến tất cả các cô thày, cô là giảng viên của trường nói chung và khoa Khoa học máy tính nói riêng đã giảng dạy, chỉ bảo tận tâm trong suốt thời gian em học tập và rèn luyện tại trường Đại học Công nghệ Thông tin.

TP. Hồ Chí Minh, tháng 12 năm 2022

Sinh viên thực hiện

## TÓM TẮT KHÓA LUẬN

Nhận dạng văn bản trong ảnh (Text recognition) là bài toán đang được cộng đồng nghiên cứu trong lĩnh vực thị giác máy tính quan tâm do có tính ứng dụng cao. Với đầu vào là ảnh chứa văn bản cần nhận dạng và đầu ra là nội dung của văn bản đó, kết quả của bài toán đóng vai trò quan trọng trong việc giúp cho máy tính có thể hiểu được nội dung của các ảnh phức tạp như biển hiệu, ảnh chữ viết tay, ảnh chụp đời thường. Đây cũng là bài toán có nhiều thách thức như có kiểu chữ đa dạng, phông chữ, màu sắc, hiệu ứng khác nhau, các chữ có liên kết phức tạp, bị chồng chéo lên nhau, chữ lẩn trong phông nền. Trong đó, vấn đề liên quan đến chữ có phong cách nghệ thuật (artistic text) đang là thách thức mới nhất mà cộng đồng nghiên cứu đang tìm hướng giải quyết. Theo tìm hiểu, hiện tại chỉ có nghiên cứu [1] tập trung giải quyết bài toán nhận dạng chữ nghệ thuật. Trong đó nhóm nghiên cứu cung cấp bộ dữ liệu WordArt cho ngôn ngữ tiếng Anh gồm các ảnh chứa chữ nghệ thuật từ các áp phích, thiệp chúc mừng, bìa, biển quảng cáo, chữ viết tay. Đồng thời tác giả cũng đề xuất phương pháp sử dụng chỉ dẫn về góc cho mô hình Transformer. Tuy có kết quả vượt trội tập hơn các phương pháp khác trên tập dữ liệu WordArt, nhưng phương pháp mà tác giả đề xuất vẫn chưa giải quyết được trường hợp khi các hoa văn trang trí trên nền có hình dáng và hình dạng giống hệt như văn bản, điều này dẫn đến khó khăn trong việc phân biệt những hoa văn này có thuộc về văn bản hay không.

Do đó trong phạm vi khóa luận lần này, tôi tập trung vào nghiên cứu hướng tiếp cận áp dụng chỉ dẫn về nét để giải quyết những hạn chế của các phương pháp hiện tại. Bên cạnh đó tôi cũng mở rộng bộ dữ liệu phục vụ cho bài toán, cụ thể là xây dựng bộ dữ liệu thư pháp tiếng Việt. Sau cùng, tôi sẽ tiến hành thực nghiệm hướng tiếp cận đề xuất trên các bộ dữ liệu dành riêng cho việc nhận dạng chữ nghệ thuật và so sánh với các phương pháp nhận dạng văn bản tiên tiến hiện nay.

**Từ khóa:** nhận dạng chữ trong ảnh, chữ nghệ thuật, artistic text, scene text recognition.

# Mục lục

<b>Mục lục</b>	<b>v</b>
<b>Danh sách hình vẽ</b>	<b>ix</b>
<b>Danh sách bảng</b>	<b>xiv</b>
<b>Danh mục từ viết tắt</b>	<b>xvi</b>
<b>1 TỔNG QUAN</b>	<b>1</b>
1.1 Đặt vấn đề . . . . .	1
1.2 Mục tiêu và phạm vi . . . . .	4
1.2.1 Mục tiêu . . . . .	4
1.2.2 Phạm vi . . . . .	4
1.3 Đóng góp của khóa luận . . . . .	5
1.4 Cấu trúc khóa luận . . . . .	5
<b>2 CƠ SỞ LÝ THUYẾT VÀ CÁC NGHIÊN CỨU LIÊN QUAN</b>	<b>7</b>
2.1 Định nghĩa về văn bản trong ảnh . . . . .	7
2.2 Nhận dạng văn bản nghệ thuật trong ảnh . . . . .	9
2.3 Các nghiên cứu liên quan . . . . .	11
2.3.1 Các bộ dữ liệu thực tế . . . . .	11
2.3.2 Các phương pháp liên quan . . . . .	12

## **MỤC LỤC**

---

2.4	Cơ sở lý thuyết . . . . .	18
2.4.1	Mô hình Sequence-to-sequence . . . . .	18
2.4.1.1	Kiến trúc tổng quan . . . . .	18
2.4.1.2	Cơ chế giải mã với Greedy Search . . . . .	21
2.4.1.3	Cơ chế giải mã với Beam Search . . . . .	21
2.4.2	Mô hình Attention . . . . .	22
2.4.3	Mô hình Transformer . . . . .	24
2.4.3.1	Transformer - Cơ chế Self-attention . . . . .	27
2.4.3.2	Transformer - Multi-Head Attention . . . . .	30
2.4.3.3	Transformer - Positional Encoding . . . . .	32
2.4.4	Mô hình Vision Transformer . . . . .	33
<b>3</b>	<b>MỘT SỐ PHƯƠNG PHÁP NHẬN DẠNG VĂN BẢN ÁP DỤNG CHO CHỮ NGHỆ THUẬT TIẾNG VIỆT</b>	<b>37</b>
3.1	Mô hình SATRN . . . . .	37
3.1.1	Kiến trúc mô hình . . . . .	38
3.1.1.1	Khối Encoder . . . . .	39
3.1.1.2	Khối Decoder . . . . .	41
3.2	Mô hình Corner Transformer . . . . .	42
3.2.1	Kiến trúc mô hình . . . . .	43
3.2.2	Khối Corner-Guided Encoder . . . . .	43
3.2.3	Hàm mất mát Character contrastive loss . . . . .	45
3.3	Mô hình ViTSTR . . . . .	46
3.4	Mô hình ABINet . . . . .	48
3.4.1	Cách nhìn tổng quan về mô hình ABINet . . . . .	50
3.4.1.1	Vision Model . . . . .	51
3.4.1.2	Language Model . . . . .	53
3.5	Mô hình PARSeq . . . . .	56

## **MỤC LỤC**

---

3.5.1	Permutation Language Modeling (PLM) - thiết lập tổ hợp	57
3.5.2	Tổng quát mô hình . . . . .	58
3.5.2.1	Encoder . . . . .	58
3.5.2.2	Decoder - Visio-lingual decoder . . . . .	58
3.5.2.3	Cách thức decoding . . . . .	60
<b>4</b>	<b>ÁP DỤNG CHỈ DẪN VỀ NÉT CHO BÀI TOÁN NHẬN DẠNG VĂN BẢN NGHỆ THUẬT</b>	<b>62</b>
4.1	Ý tưởng . . . . .	62
4.2	Thuật toán dò tìm khung xương . . . . .	63
4.3	Thuật toán phân cụm . . . . .	66
4.4	Áp dụng vào mô hình Transformer . . . . .	67
<b>5</b>	<b>THỰC NGHIỆM VÀ ĐÁNH GIÁ</b>	<b>70</b>
5.1	Các bộ dữ liệu cho bài toán nhận dạng văn bản nghệ thuật . . . . .	70
5.1.1	Bộ dữ liệu chữ nghệ thuật cho tiếng Anh và tiếng Việt .	70
5.1.2	Bộ dữ liệu thư pháp tiếng Việt . . . . .	71
5.1.2.1	Thu thập và gán nhãn dữ liệu . . . . .	71
5.1.2.2	Một số đánh giá trên tập dữ liệu . . . . .	72
5.2	Các độ đo để thực hiện đánh giá . . . . .	74
5.2.1	Độ chính xác (Accuracy) . . . . .	74
5.2.2	Tỉ lệ lỗi (CER) . . . . .	75
5.2.3	Số khung hình trên giây (FPS) . . . . .	75
5.3	Kết quả thực nghiệm và đánh giá . . . . .	76
5.3.1	Đánh giá hướng tiếp cận mới so với phương pháp nền tảng	76
5.3.2	Đánh giá dựa trên việc điều chỉnh tham số có sẵn . . . . .	76
5.3.3	Đánh giá hướng tiếp cận mới trên toàn bộ dữ liệu chữ nghệ thuật . . . . .	78

## **MỤC LỤC**

---

<b>6 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN</b>	<b>82</b>
6.1 Kết luận . . . . .	82
6.2 Hướng phát triển . . . . .	82
<b>Tài liệu tham khảo</b>	<b>84</b>

# Danh sách hình vẽ

1.1	Ảnh minh họa đầu vào và đầu ra của bài toán nhận dạng văn bản	2
1.2	Một số mẫu chữ nghệ thuật từ tập dữ liệu WordArt - (Nguồn: Bài báo [1]) . . . . .	2
1.3	Hình ảnh minh họa văn bản trên biển tên cửa hàng . . . . .	3
1.4	Ảnh minh họa độ khó của chữ nghệ thuật. (a) Hoa văn trên nền có hình dạng giống hệt văn bản, (b) Nét của chữ 'g' kéo dài và cong hơn nét bình thường . . . . .	3
2.1	Hình minh họa văn bản ngoại cảnh (a) và văn bản đồ họa (b). <i>Văn bản cảnh</i> là văn bản xuất hiện trong lúc chụp ảnh; còn ngược lại, <i>văn bản đồ họa</i> là văn bản được thêm vào bởi con người sau khi chụp (thường dành cho mục đích tiêu đề, phụ đề, ...). . . . .	8
2.2	Hình minh họa về chữ nghệ thuật. Có thể thấy, chữ cách điệu được thiết kế đa dạng về phông chữ, hiệu ứng chìm, nổi, các kí tự bị gối lên nhau, . . . . . Và ngoài ra, chữ với nền phức tạp cũng được tính là chữ cách điệu. (Nguồn: Bài báo [1]) . . . . .	10
2.3	Hình ảnh một số mẫu dữ liệu trong các tập dữ liệu nhận dạng văn bản ngôn ngữ tiếng Anh . . . . .	12
2.4	Hình ảnh một số mẫu dữ liệu trong các tập dữ liệu nhận dạng văn bản ngôn ngữ tiếng Việt . . . . .	13

## DANH SÁCH HÌNH VẼ

---

- 2.5 Hình mô tả cách hoạt động của một mạng STN. Cụ thể, mạng STN bao gồm 3 thành phần chính. Bước đầu (1), một mạng giúp dự đoán một ma trận  $\theta$  (từ đầu vào là ảnh/đặc trưng đầu vào được rút trích bởi 1 mạng CNN). Ma trận  $\theta$  sẽ trở thành tham số đầu vào cho phép biến đổi affine  $T_\theta$ . Bước tiếp theo (2), ta cần tạo được ánh xạ được tọa độ từ ảnh gốc U sang tọa độ của ảnh V. Bước cuối cùng (3), ta cần nội suy được giá trị của pixel trong ảnh V dựa trên giá trị pixel từ ảnh gốc và cắp tọa độ suy từ bước vừa rồi. (Nguồn: Bài báo [2]) . . . . . 15
- 2.6 Hình minh họa 2 ví dụ về hai ma trận affine  $T_\theta$ . Ví dụ a) (Trivial examples), ma trận biến đổi  $\theta$  là một ma trận đơn vị  $I$ , nghĩa là phép biến đổi  $T_\theta$  hoàn toàn không thay đổi gì ảnh gốc U. Ngược lại, trường hợp  $\theta$  là ma trận khác ma trận đơn vị nêu trên, sẽ cho ta kết quả là hình ảnh V từ ảnh gốc U. (Nguồn: Bài báo [2]) . . . 15
- 2.7 Hình minh họa đơn giản chiến lược thiết kế các thuật toán STR. Trong đó, hình a) ví dụ cho thuật toán sử dụng một mạng CNN, một khối encoder, và một khối decoder. Ngược lại, ViTSTR loại bỏ hoàn toàn khối CNN và khối decoder. Lưu ý: Hình chỉ minh họa sự khác biệt giữa 2 chiến lược thiết kế thuật toán, không mô tả hoàn toàn các bước của 2 thuật toán được nêu. . . . . 17
- 2.8 Kiến trúc mô hình sequence-to-sequence (Nguồn: Bài báo [3]) . 19
- 2.9 Kiến trúc tổng quan của mô hình seq2seq có sử dụng cơ chế Attention (Nguồn: Internet<sup>1</sup>) . . . . . 22
- 2.10 Vector ngữ nghĩa trong kiến trúc attention (Nguồn: Internet<sup>1</sup>) . 23
- 2.11 Cách mô hình attention sử dụng vector ngữ nghĩa để tạo ra vector trạng thái ẩn attention (Nguồn: Internet<sup>1</sup>) . . . . . 23
- 2.12 Tính điểm số attention (Nguồn: Internet<sup>1</sup>) . . . . . 24
- 2.13 Kiến trúc của mô hình Transformer (Nguồn: Bài báo [4]) . . . . . 26

## DANH SÁCH HÌNH VẼ

---

2.14	Hình ảnh trực quan hóa quá cách hoạt động của Self-attention (Nguồn: Internet <sup>2</sup> ) . . . . .	28
2.15	Cách hoạt động của self-attention (Nguồn: Internet <sup>2</sup> ) . . . . .	29
2.16	Hình ảnh minh họa cơ chế attention và kiến trúc Multi-Head Attention (Nguồn: Bài báo [4]) . . . . .	30
2.17	Cách hoạt động của Muli-Head attention (Nguồn: Internet <sup>2</sup> ) . .	31
2.18	Kiến trúc mô hình ViT (Nguồn: Bài báo [5]) . . . . .	34
2.19	Ảnh gốc (bên trái) và bản đồ Attention tương ứng với ảnh đầu vào của mô hình ViT đã được huấn luyện (bên phải) (Nguồn: Internet <sup>3</sup> ) . . . . .	35
3.1	Kiến trúc mô hình SATRN (Nguồn: Bài báo [6]) . . . . .	38
3.2	Sự thay đổi so với các lớp feedforward trong Encoder của Trans- former (Nguồn: Bài báo [6]) . . . . .	41
3.3	Kiến trúc mô hình Corner Transformer (Nguồn: Bài báo [1]) . .	43
3.4	Hình ảnh trực quan hóa kết quả của bản đồ key point (Nguồn: Bài báo [1]) . . . . .	44
3.5	Hình ảnh bản đồ nhiệt của attention không có và có sử dụng corner-query (Nguồn: Bài báo [1]) . . . . .	45
3.6	Trực quan hóa các điểm dữ liệu trong không gian khi huấn luyện mô hình chỉ dùng CE loss và dùng CE loss kết hợp với CC loss (Nguồn: Bài báo [1]) . . . . .	46
3.7	Kiến trúc của ViTSTR so với các kiến trúc STR khác (Nguồn: Bài báo [7]) . . . . .	47
3.8	Kiến trúc của mô hình ViTSTR (Nguồn: Bài báo [7]) . . . . .	48

## DANH SÁCH HÌNH VẼ

---

3.9	Hình cắt ra từ bài báo ABINet. a) Mô tả mô hình ngôn ngữ phụ thuộc vào kết quả đầu ra mô hình thị giác. b) Cải thiện của tác giả, không cho mô hình ngôn ngữ lệ thuộc vào mô hình thị giác nữa. Và ngoài ra, quá trình dự đoán có sự kết hợp của cả hai. c) Cải thiện tiếp theo của tác giả: Áp dụng đặc trưng hai chiều <i>bidirectional</i> - đặt tên là <b>BCN</b> - trong mô hình ngôn ngữ - dựa trên ý tưởng của BERT. d) Mô hình <i>tự hồi qui</i> (autoregressive) - dự đoán tuần tự (nghĩa là kết quả đầu ra $y_t$ có phụ thuộc vào $y_{t-1}$ ). e) Áp dụng <b>transformer</b> theo cả 2 hướng <b>unidirectional</b> - để dự đoán cùng một lúc tất cả kí tự. (Nguồn: Bài báo [8]) . . . . .	49
3.10	Hình tổng quan thuật toán ABINet (Nguồn: Bài báo [8]) . . . . .	51
3.11	Tổng quan phần Vision Model (VM) của ABINet (Nguồn: Bài báo [8]) . . . . .	52
3.12	Tổng quan phần Language Model (LM) của ABINet - Kiến trúc BCN (Nguồn: Bài báo [8]) . . . . .	54
3.13	Hình so sánh mô hình a) ABINet với b) PARSeq. Với cách hoạt động của ABINet, ta tách rời VM (không ngữ cảnh - <b>context-free</b> ) với LM (tinh chỉnh - <b>iterative-refinement</b> ), hay nói cách khác, ta sử dụng LM như một bộ từ điển - để chỉnh chính tả. Điều này đôi lúc dẫn tới LM sẽ dự đoán sai nếu như từ sai đó lại có xác suất cao hơn (do quá trình huấn luyện pretrainng LM). Ngược lại, mô hình LM mà nhóm của PARSeq đề xuất kết hợp <b>context-free</b> và <b>iterative-refinement</b> , một cách linh hoạt thông qua mô hình <b>Permutation Language Modeling</b> - gọi tắt là (PLM). (Nguồn: Bài báo [9]) . . . . .	56
3.14	Tổng quan mô hình và cách huấn luyện của PARSeq (Nguồn: Bài báo [9]) . . . . .	59

## DANH SÁCH HÌNH VẼ

---

3.15 Khối <b>decoder</b> của PARSeq - được đặt tên là Visio-lingual decoder. Mỗi lớp của khối sẽ có điểm khác biệt là có đến 2 lớp attention (Nguồn: Bài báo [9]) . . . . .	59
4.1 Hình minh họa bản đồ góc (corner map) và bản đồ khung xương (skeleton-map) . . . . .	63
4.2 Hình ảnh minh họa thuật toán dò tìm khung xương (Nguồn: Thư viện Scikit-image) . . . . .	64
4.3 Hình minh họa cho ba giả thuyết (a), (b), (c) . . . . .	65
4.4 Hình ảnh minh họa kết quả của thuật toán dò tìm khung xương .	66
4.5 Ảnh so sánh kết quả phân cụm của thuật toán K-means gốc và thuật toán K-means sau điều chỉnh . . . . .	68
4.6 Mô hình Transformer sử dụng hướng tiếp cận chỉ dẫn về nét (Skeleton-Guided) . . . . .	69
5.1 Ảnh chứa chữ thư pháp trên các phương tiện truyền thông (a) và ảnh được cắt bởi công cụ gán nhãn (b) . . . . .	72
5.2 Giao diện của công cụ gán nhãn dữ liệu LabelMe . . . . .	73
5.3 Một số hình ảnh trong bộ dữ liệu chữ thư pháp tiếng Việt . . . . .	74
5.4 Một số kết quả dự đoán mà hướng tiếp cận Skeleton-Guided đúng và các phương pháp khác sai . . . . .	80
5.5 Một số kết quả dự đoán mà hướng tiếp cận Skeleton-Guided sai và các phương pháp khác đúng . . . . .	81

# Danh sách bảng

2.1	Các phiên bản của mô hình ViT . . . . .	35
3.1	Các phiên bản của mô hình ViTSTR . . . . .	48
3.2	Bảng biểu diễn cách để hoạt động của attention mask - $\mathbf{m}$ với các hoán vị đã như nêu ra ở 3.23. Với [B] và [E] lần lượt là token biểu diễn khởi đầu và kết thúc từng chuỗi. Và cũng như đã nêu ở phương trình 3.24 - dễ dàng thấy, ngoài việc một xác suất có điều kiện sẽ của vị trí theo cột (có [E]) phụ thuộc vào vị trí theo hàng (có [B]); thì mọi vị trí đương nhiên đều phụ thuộc vào token khởi đầu [B]; và token [E] sẽ phụ thuộc vào mọi vị trí.	60
3.3	Bảng minh họa cho các biểu diễn ba chiến lược masking của mô hình AR, NAR và cloze (mà ABINet sử dụng). Cả ba chiến lược này đều có thể được học thông qua một trong các tổ hợp liệt kê từ 3.23 . . . . .	61
5.1	Bảng thống kê về số lượng ảnh chữ nghệ thuật, ảnh không phải chữ nghệ thuật và số lượng đóng góp của các bộ dữ liệu . . . . .	71
5.2	Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của hai mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText . . . . .	76

## **DANH SÁCH BẢNG**

---

5.3	Bảng kết quả độ chính xác của các mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText . . . . .	77
5.4	Bảng kết quả tỉ lệ lỗi của các mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText . . . . .	77
5.5	Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của các mô hình trên dữ liệu đánh giá 1 . . . . .	79
5.6	Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của các mô hình trên dữ liệu đánh giá 2 . . . . .	79
5.7	Bảng kết quả tốc độ dự đoán của các mô hình trên cả hai dữ liệu đánh giá . . . . .	79

# Danh mục từ viết tắt

Từ viết tắt	Nội dung
DL	Deep Learning
CNN(s)	Convolutional Neural Networks
RNN	Recurrent Neural Network
LSTM	Long Short Term Memory
Conv	Convolution

# **Chương 1**

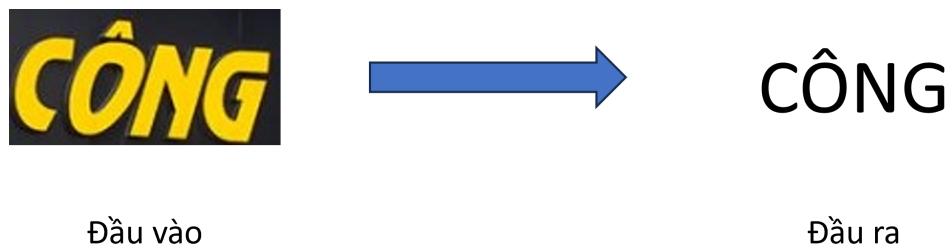
## **TỔNG QUAN**

### **1.1 Đặt vấn đề**

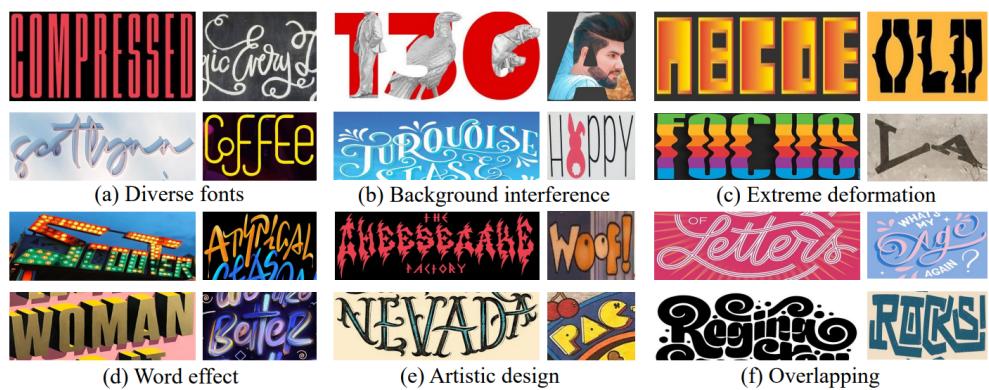
Các chữ viết đầu tiên đã xuất hiện cách đây hàng ngàn năm, ở trên các bức vách trong hang động của người tiền sử hay trong các bức vẽ. Chữ viết có vai trò to lớn trong lịch sử phát triển của xã hội loài người. Nó cũng là phương tiện hữu ích để con người có thể truyền đạt lại thông tin, kiến thức, kinh nghiệm để các thế hệ sau có thể sử dụng lại. Trong thời đại phát triển của kĩ thuật số, văn bản chữ viết xuất hiện rất nhiều trong các phương tiện truyền thông như ảnh và video. Từ đó phát sinh nhu cầu phát hiện và nhận dạng văn bản trong ảnh hay video. Bài toán nhận dạng văn bản trong ảnh - scene text recognition (STR) ra đời để phục vụ nhu cầu đó, và thu hút được rất nhiều sự quan tâm trong cộng đồng nghiên cứu.

Bài toán được mô tả là với ảnh đầu vào chứa vùng ảnh cần nhận dạng và đầu ra là nội dung của văn bản trong ảnh (ảnh minh họa [1.1](#))

- Đầu vào (input): Ảnh chứa văn bản cần nhận dạng
- Đầu ra (output): Nội dung văn bản trong ảnh



Hình 1.1: Ảnh minh họa đầu vào và đầu ra của bài toán nhận dạng văn bản



Hình 1.2: Một số mẫu chữ nghệ thuật từ tập dữ liệu WordArt - (Nguồn: Bài báo [1])

Nhận dạng văn bản nghệ thuật trong ảnh có nhiều khó khăn do sự đa dạng của kiểu chữ, phông chữ, màu sắc, và các hiệu ứng khác nhau trong ảnh. Các chữ có thể bị chồng chéo lên nhau, hoặc lẫn lộn với nền. Đặc biệt, việc nhận dạng văn bản có phong cách nghệ thuật, được thiết kế đặc biệt (gọi là "Artistic Text" như Hình 1.3), đang là một thách thức gần đây mà cộng đồng nghiên cứu đang nỗ lực tìm hướng giải quyết. Cụ thể trong trường hợp khi các hoa văn trang trí trên nền có hình dáng và hình dạng giống hệt như văn bản thì rất khó để phân biệt những hoa văn này có thuộc về văn bản hay không (1.4). Hay với chữ thư pháp ở hình (1.4), nét của chữ 'g' kéo dài và bao gồm các chữ cái khác. Đây thực sự là những ví dụ đầy thách thức đối với bất kỳ phương pháp nhận dạng văn bản nào.

Trong phạm vi khóa luận lần này, tôi tập trung vào nghiên cứu hướng tiếp

## 1. Tổng quan



Hình 1.3: Hình ảnh minh họa văn bản trên biển tên cửa hàng



a



b

Hình 1.4: Ảnh minh họa độ khó của chữ nghệ thuật. (a) Hoa văn trên nền có hình dạng giống hệt văn bản, (b) Nét của chữ 'g' kéo dài và cong hơn nét bình thường

cận áp dụng chỉ dẫn về nét để giải quyết những hạn chế của các phương pháp hiện tại. Bên cạnh đó tôi cũng mở rộng bộ dữ liệu phục vụ cho bài toán, cụ thể là xây dựng bộ dữ liệu thư pháp tiếng Việt. Sau cùng, tôi sẽ tiến hành thực nghiệm hướng tiếp cận đề xuất trên các bộ dữ liệu dành riêng cho việc nhận dạng chữ nghệ thuật và so sánh với các phương pháp nhận dạng văn bản tiên tiến hiện nay.

## 1.2 Mục tiêu và phạm vi

### 1.2.1 Mục tiêu

Những mục tiêu cụ thể để hoàn thành đề tài khóa luận "*Nghiên Cứu Phương Pháp Nhận Dạng Văn Bản Nghệ Thuật Trong Ảnh*" được đề ra cụ thể như sau:

- Tìm hiểu tổng quan về bài toán và các hướng tiếp cận tiên tiến hiện nay đối với bài toán.
- Xây dựng và phân tích tập dữ liệu chữ thư pháp tiếng Việt.
- Tìm hiểu và cài đặt hướng tiếp cận sử dụng chỉ dẫn về nét trong bài toán nhận dạng văn bản nghệ thuật.
- Đánh giá và so sánh một số phương pháp tiên tiến hiện nay với hướng tiếp cận đề xuất trên các bộ dữ liệu chữ nghệ thuật.

### 1.2.2 Phạm vi

Dựa trên những mục đã đề xuất tại nội dung 1.2.1, trong khuôn khổ của khóa luận, tôi sẽ tập trung hoàn thành các nội dung sau đây:

- Xây dựng tập dữ liệu thư pháp tiếng Việt bao gồm 15441 ảnh.
- Xây dựng tập dữ liệu chữ nghệ thuật cho tiếng Anh và tiếng Việt từ các tập dữ liệu chuẩn như VinText [10], BKAI-TEXT, VietSignBoard, CUTE80 [11], Total-Text [12], ICDAR13 [13].
- Tìm hiểu tổng quan về bài toán, các thách thức và cơ sở lý thuyết của một số phương pháp phổ biến: ViTSTR [7], SATRN [6], Corner Transformer [1], ABINet [8], PARSeq [9].

- Nghiên cứu và cài đặt hướng tiếp cận "Áp dụng chỉ dẫn về nét trong bài toán nhận dạng văn bản".
- Đánh giá và so sánh các phương pháp tiên tiến với hướng tiếp cận đề xuất trên các tập dữ liệu chữ nghệ thuật.

### **1.3 Đóng góp của khóa luận**

Sau đây là một số kết quả đóng góp mà tôi đã đạt được trong quá trình thực hiện khóa luận:

- Xây dựng bộ dữ liệu thư pháp tiếng Việt gồm 15441 ảnh với 2980 từ.
- Đánh giá một số phương pháp tiên tiến hiện nay trên các tập dữ liệu chữ nghệ thuật: SATRN, ViTSTR, CornerTransformer, ABINet và PARSeq.
- Đưa ra hướng tiếp cận "Áp dụng chỉ dẫn về nét" cho bài toán nhận dạng văn bản nghệ thuật.
- Tổng kết các kết quả thu thập và đánh giá trên bộ dữ liệu chữ thư pháp tiếng Việt thành một bài báo khoa học (được đính kèm ở sau cuốn khóa luận).

### **1.4 Cấu trúc khóa luận**

**Chương 1:** Giới thiệu tổng quan về bài toán nhận dạng văn bản.

**Chương 2:** Trình bày về cơ sở lý thuyết và các nghiên cứu liên quan đến bài toán nhận dạng văn bản.

**Chương 3:** Trình bày về các phương pháp nhận dạng văn bản sử dụng để thực nghiệm.

## **1. Tổng quan**

---

**Chương 4:** Trình bày về hướng tiếp cận "Áp dụng chỉ dẫn về nét" trong bài toán nhận dạng văn bản nghệ thuật

**Chương 5:** Trình bày cách xây dựng bộ dữ liệu, phân tích dữ liệu, cách thức đánh giá và so sánh kết quả của các mô hình trên bộ dữ liệu đã xây dựng.

**Chương 6:** Kết luận về khóa luận tốt nghiệp và hướng phát triển trong tương lai.

## **Chương 2**

# **CƠ SỞ LÝ THUYẾT VÀ CÁC NGHIÊN CỨU LIÊN QUAN**

Trước khi đi vào nội dung chính của khóa luận, tôi xin được trình bày các khái niệm, cơ sở lý thuyết của bài toán nhận dạng văn bản và một số hướng tiếp cận phổ biến hiện nay.

### **2.1 Định nghĩa về văn bản trong ảnh**

Trong thực tế, văn bản trong ảnh có thể xuất hiện dưới nhiều hình dạng khác nhau. Ví dụ, dựa vào cách tạo ra có thể chia làm văn bản in ấn và văn bản viết tay [14]. Văn bản in ấn được tạo ra bằng cách sử dụng máy in và có thể dễ dàng nhận biết hơn so với văn bản viết tay, do văn bản viết tay thường phức tạp, chồng chéo, đan xen nhau, kể cả cùng một người, nét chữ giữa các lần viết cũng có sự khác nhau. Ngoài ra, cũng có thể phân loại văn bản thành hai loại là văn bản đồ họa (graphic text) và văn bản ngoại cảnh (scene text) như trong [15] - xem hình minh họa 2.1. Trong đó có thể thấy văn bản ngoại cảnh đa dạng và khó để nhận dạng hơn rất nhiều so với văn bản đồ họa.

Theo như trong bài báo [14], văn bản đồ họa là văn bản được chèn thêm vào

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



(a) Văn bản ngoại cảnh



(b) Văn bản đồ họa

Hình 2.1: Hình minh họa văn bản ngoại cảnh (a) và văn bản đồ họa (b). *Văn bản cảnh* là văn bản xuất hiện trong lúc chụp ảnh; còn ngược lại, *văn bản đồ họa* là văn bản được thêm vào bởi con người sau khi chụp (thường dành cho mục đích tiêu đề, phụ đề, ...).

ảnh, video sau khi chụp, quay. Văn bản ngoại cảnh là văn bản được chụp trong môi trường tự nhiên, trong đời sống hàng ngày. Hơn thế nữa, tùy vào các ngôn ngữ khác nhau, ví dụ ngôn ngữ dựa trên tiếng Latin như tiếng Anh, tiếng Việt, ngôn ngữ dựa trên chữ tượng hình như tiếng Trung, tiếng Nhật... thì số lượng ký tự trong bảng chữ cái, thứ tự đọc cũng sẽ khác nhau, và mỗi ngôn ngữ sẽ có thách thức riêng. Trong khóa luận lần này, tôi sẽ tập trung vào cả văn bản tiếng Việt và tiếng Anh cũng như ảnh đồ họa lẫn ảnh ngoại cảnh. Đối với văn bản ngoại cảnh, vì được chụp trong môi trường tự nhiên nên sẽ bị đa dạng phông chữ, nhiều do điều kiện chụp như ngược sáng, không đủ sáng, bị mờ, độ phân giải thấp... Riêng đối với tiếng Việt, số lượng ký tự trong bảng chữ cái nhiều hơn tiếng Anh do các dấu thanh (‘, ‘, ?, ..) trên nguyên âm (a, o, e, u, i) và các biến thể (ví dụ: ô, ể, â, ă, ơ, ư). Vì thế dễ gây ra sự nhầm lẫn giữa các ký tự gốc và các ký tự có nguyên âm hay biến thể cao hơn so với tiếng Anh.

### 2.2 Nhận dạng văn bản nghệ thuật trong ảnh

**Nhận dạng văn bản trong ảnh** là một trong những bài toán được cộng đồng thế giới quan tâm nghiên cứu vì tính ứng dụng cao của nó. Điều này có thể dễ dàng nhận thấy thông qua một vài ứng dụng của nó trong các hệ thống thông minh ví dụ như hỗ trợ định vị, nhận dạng tên đường, hay số hóa văn bản, tài liệu, giấy tờ tùy thân . . . Vì thế, bài toán nhận dạng văn bản trong ảnh ngoại cảnh là một trong những chủ đề rất thu hút đối với cộng đồng nghiên cứu về thị giác máy tính trên thế giới. Hơn thế nữa, trong những năm gần đây, bài toán được đặc biệt đầu tư nhờ có sự tổ chức của cuộc thi "*ICDAR Robust Reading Competitions*" với lần đầu tiên được tổ chức là vào năm 2013, từ đó đến nay cuộc thi đã cung cấp nhiều bộ dữ liệu chuẩn đáng tin cậy phục vụ cho bài toán nhận dạng văn bản. Ở Việt Nam cũng đã có nhiều cuộc thi về chủ đề nhận dạng văn bản trong ảnh được tổ chức như "*Ho Chi Minh AI Challenge 2021*", "*BKAI-NAVER Challenge 2022*" và nhận được sự quan tâm rất lớn của các bạn sinh viên cũng như của cộng đồng nghiên cứu trong nước.

Trong quá trình quan sát về dữ liệu liên quan đến bài toán, tôi nhận thấy một số văn bản xuất hiện trong các tập dữ liệu kể trên được thiết kế với tính thẩm mỹ cao, ví dụ như hiệu ứng đổ bóng, uốn lượn, phông chữ đa dạng, phông nền rực rỡ [2.2](#)). Phần lớn các văn bản này thường xuất hiện trên các áp phích, thiệp chúc mừng, bìa, biển quảng cáo và chúng còn được chụp trong điều kiện ngoại cảnh, điều này gây ra khó khăn rất lớn cho các thuật toán nhận dạng văn bản tiên tiến hiện nay. Đây sẽ là một bài toán đem lại nhiều thách thức cho cộng đồng nghiên cứu - gọi là thách thức chữ nghệ thuật - *artistic text*.

Tuy nhiên, một hạn chế lớn của các tập dữ liệu vừa nêu trên đó chính là không phân biệt chữ với những tiêu chí nêu trên với chữ với phông thường gấp trên máy tính. Điều này dẫn tới không có nhiều nghiên cứu cụ thể trên chữ nghệ thuật - tính tới thời điểm hiện tại, chỉ có công trình [\[1\]](#) là tập trung vào thách

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.2: Hình minh họa về chữ nghệ thuật. Có thể thấy, chữ cách điệu được thiết kế đa dạng về phông chữ, hiệu ứng chìm, nổi, các kí tự bị gối lên nhau, ... . Và ngoài ra, chữ với nền phức tạp cũng được tính là chữ cách điệu. (Nguồn: Bài báo [1])

thức đề cập ở trên.

Tuy nhiên, hiện nay chưa có tập dữ liệu nào phân biệt chữ nghệ thuật với những đặc điểm nêu trên. Điều này dẫn đến không có nhiều nghiên cứu cụ thể trên chữ nghệ thuật - tính đến thời điểm hiện tại, chỉ có công trình [1] là tập trung vào giải quyết thách thức đề cập ở trên.

Tôi nhận thấy rằng tập dữ liệu do tác giả [1] đề xuất chỉ bao gồm các ảnh áp phích, thiệp chúc mừng, bìa, biển quảng cáo. Trong khi đó, có một số thể loại văn bản khác cũng được tính là chữ nghệ thuật như chữ thư pháp, chữ graffiti. Ngoài ra, tuy có kết quả vượt trội tập hơn các phương pháp khác trên tập dữ liệu WordArt, nhưng phương pháp mà tác giả đề xuất vẫn chưa giải quyết được trường hợp khi các hoa văn trang trí trên nền có hình dáng và hình dạng giống hệt như văn bản, điều này dẫn đến khó khăn trong việc phân biệt những hoa văn này có thuộc về văn bản hay không.

Theo đó trong khuôn khổ của khóa luận, tôi sẽ tiến hành xây dựng một bộ dữ liệu về chữ thư pháp tiếng Việt nhằm góp phần tăng cường dữ liệu cho bài toán nhận dạng văn bản nghệ thuật. Đồng thời, tôi cũng sẽ tìm hiểu, nghiên cứu

## **2. Cơ sở lý thuyết và các nghiên cứu liên quan**

---

và cài đặt hướng tiếp cận "áp dụng chỉ dẫn về nét". Từ đó sẽ đưa ra so sánh giữa hướng tiếp cận này với các phương pháp tiên tiến hiện nay.

### **2.3 Các nghiên cứu liên quan**

#### **2.3.1 Các bộ dữ liệu thực tế**

Nhận dạng văn bản là một hướng nghiên cứu đã có từ lâu và thu hút được nhiều nhà nghiên cứu quan tâm đóng góp, vì thế dữ liệu cho bài toán rất đa dạng. Một số bộ dữ liệu nổi tiếng có thể kể đến như ICDAR13, TotalText, CUTE80 ([2.3](#)). Đối với tiếng Việt, hiện nay có một số bộ dữ liệu như BKAI, VinText hay VietSignBoard ([2.4](#)).

ICDAR 2013, gọi tắt là ICDAR13 hay IC13, là tập dữ liệu phục vụ cho cuộc thi "ICDAR Robust Reading Competitions" vào năm 2013, từ đó đến nay cuộc thi được tổ chức thường niên với nhiều nhiệm vụ mới tập trung vào các bài toán phân tích và đọc hiểu văn bản. Tập dữ liệu ICDAR13 bao gồm 229 ảnh huấn luyện và 233 ảnh kiểm thử, ảnh được gán nhãn theo mức từ (word level), văn bản trong tập dữ liệu này có đặc điểm là thẳng và hơi cong nhẹ, ảnh có chất lượng tốt hơn so với các tập dữ liệu khác.

TotalText là tập dữ liệu tập trung vào các văn bản chữ cong. Khác với ICDAR13, tập TotalText được gán nhãn theo hình đa giác (polygon) sử dụng 8, 10 và 12 điểm thay vì 4 điểm như tập ICDAR13. Tập TotalText có 1255 ảnh huấn luyện và 1300 ảnh kiểm thử.

CUTE80 cũng là tập dữ liệu tập trung vào chữ cong, nhưng các ảnh dữ liệu lại chủ yếu được chụp từ áo thi đấu của các cầu thủ hay trong các ảnh logo. Đặc điểm của bộ dữ liệu này là đa phần các văn bản trong ảnh đều là chữ nghiêng, cong và bị nhăn do chữ trên áo. Bộ dữ liệu này có tổng cộng 80 ảnh dữ liệu.

Đối với ngôn ngữ tiếng Việt, có tập dữ liệu VinText [[10](#)]. Với số lượng ảnh là

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



(a) ICDAR13



(b) TotalText



(c) CUTE80

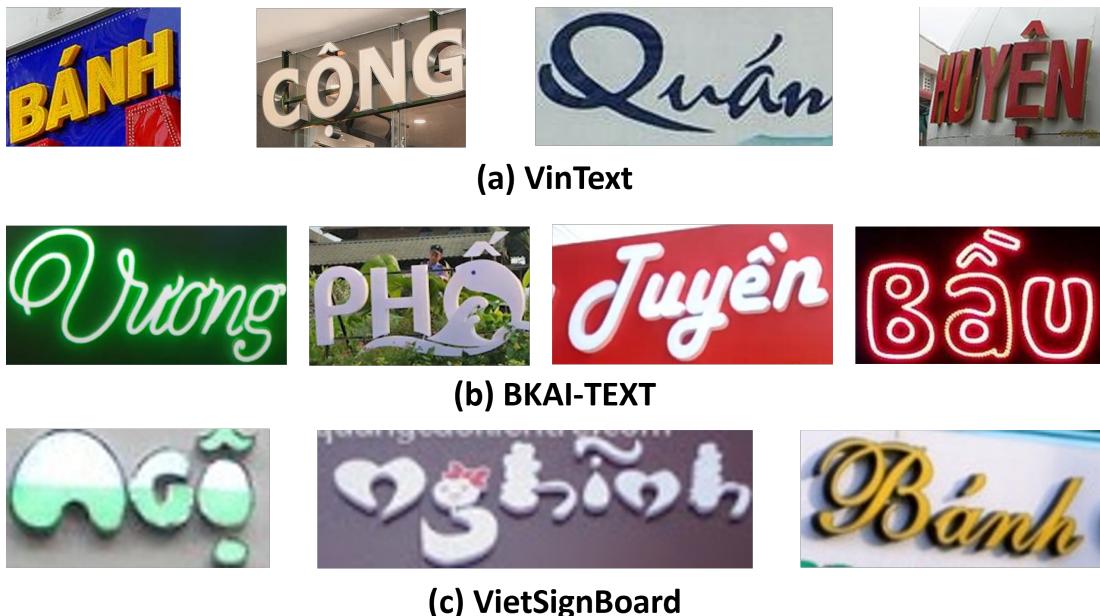
Hình 2.3: Hình ảnh một số mẫu dữ liệu trong các tập dữ liệu nhận dạng văn bản ngôn ngữ tiếng Anh

2000, được thu thập ảnh chứa cảnh đời sống sinh hoạt hằng ngày của con người Việt Nam. Tính đến hiện tại VinText là tập dữ liệu chuẩn đầu tiên bằng tiếng Việt. BKAI-Text là tập dữ liệu được cung cấp bởi ban tổ chức cuộc thi "BKAI-NAVER Challenge 2021" dành cho nhiệm vụ phát hiện và nhận dạng văn bản trong ảnh. VietSignBoard là bộ dữ liệu tập trung vào các ảnh biển hiệu, biển quảng cáo ở Việt Nam. Một số điểm dữ liệu của các tập dữ liệu ngôn ngữ tiếng Việt [2.4](#)

### **2.3.2 Các phương pháp liên quan**

Trong thực tế, bài toán nhận dạng đối mặt với nhiều thách thức khác nhau như đã đề cập tại phần dữ liệu [2.3.1](#) - như chữ với nền, phông chữ phức tạp, ... hay chữ bị quá cong, quá mờ, ... . Như vậy, để dễ dàng giải quyết từng thách thức

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.4: Hình ảnh một số mẫu dữ liệu trong các tập dữ liệu nhận dạng văn bản ngôn ngữ tiếng Việt

một, bài khảo sát của [14], cũng như nghiên cứu của [16] đã đề ra tiêu chuẩn cấu trúc của một thuật toán STR như sau:

- (1) Bước tiền xử lý ảnh **Transformation**. Mục đích của bước này là góp phần nâng cao hiệu quả của bước tiếp theo. Có một vài hướng tiếp cận như: sử dụng một mạng nắn thẳng (thường gấp trong trường hợp chữ cong) hoặc loại bỏ đi nhiễu nền;
- (2) Bước rút trích đặc trưng **Feature extraction**. - một mạng học sâu dùng để rút trích đặc trưng thị giác (thông thường đã được huấn luyện trước). Thường dùng nhất là các mạng tích chập CNN như VGG[17] hay ResNet[18]; tuy nhiên, trong những năm gần đây những mô hình Transformer được ứng dụng vào các nhiệm vụ thị giác máy tính như ViT [5], DeiT [19] đã được sử dụng để rút trích đặc trưng;

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

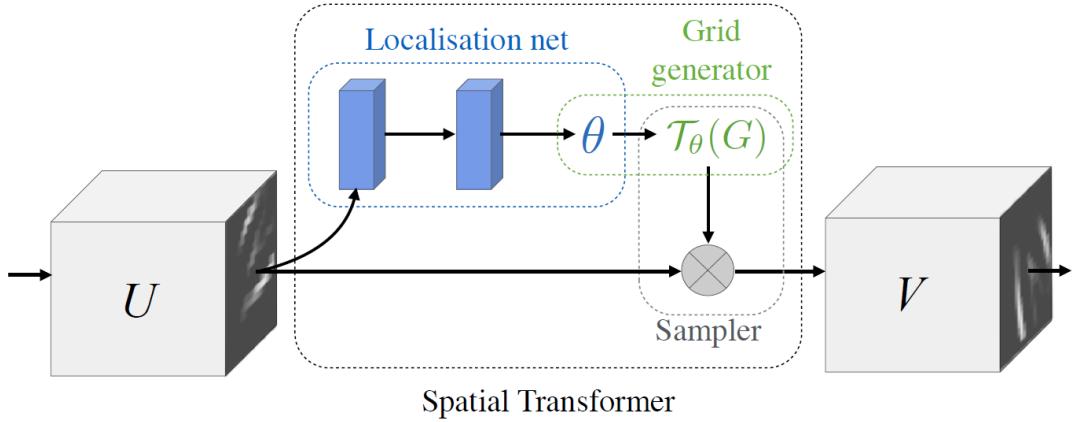
- (3) Mô hình tuần tự **Sequence modeling**. Kí tự trong một từ của một ngôn ngữ thường liên hệ mật thiết với nhau. Cho nên bước này sử dụng mạng hồi quy như RNN/LSTM giúp tạo ra mối liên hệ giữa các đặc trưng thị giác xa/gần (cũng chính là đặc trưng của từng kí tự). Vì thế, giai đoạn này giúp cải thiện độ ổn định cho bước dự đoán kết quả.
- và cuối cùng là (4) Giai đoạn dự đoán kết quả **Pred.**, và theo như một vài SOTA hiện nay, bước này thường sử dụng mô hình CTC [20] hoặc cơ chế Attention[21] nổi bật nhất trong cả lĩnh vực Thị giác máy tính lẫn Ngôn ngữ tự nhiên.

**Bước tiền xử lý ảnh:** Để giải quyết bài toán nhận dạng văn bản theo hướng cong, đã có các công trình như **STAR-Net** của Liu và cộng sự [22], RARE [23] ASTER,[24] áp dụng STN *Spatial Transformer Network*[2]. Cụ thể là module STN được dùng để chuyển đổi ảnh gốc  $I$  thành ảnh  $I'$ . Cụ thể, bằng cách dự đoán một số điểm neo *fiducial points* sử dụng 1 mạng CNN (Spatial Transform Networks) nắn thẳng văn bản cong thành dạng thẳng. Việc này giúp ta loại bỏ được phần thông tin không quan trọng từ nền như nhiễu và tạo điều kiện cho việc huấn luyện trở nên dễ dàng hơn.

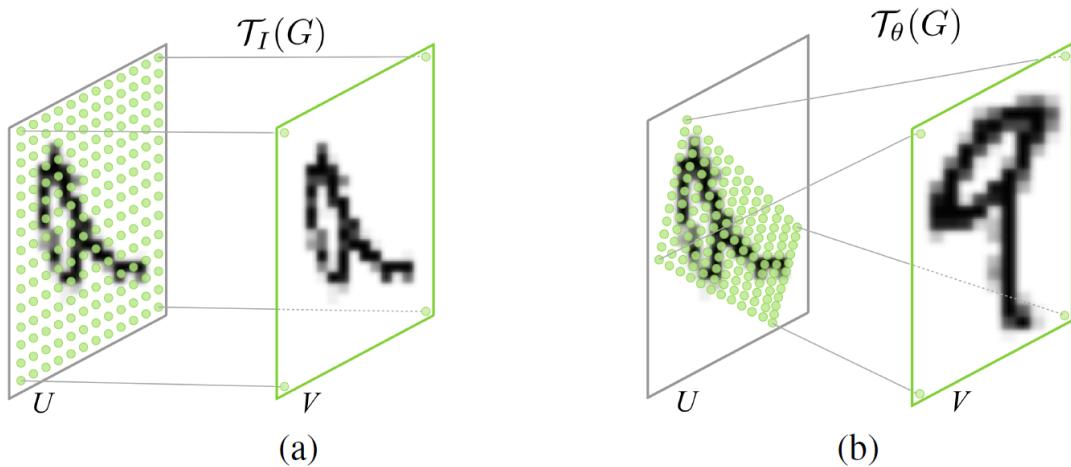
$$\begin{pmatrix} a & b \\ c & d \end{pmatrix} = \mathcal{T}_\theta(G_i) = A_\theta \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} = \begin{vmatrix} \theta_{11} & \theta_{12} & \theta_{13} \\ \theta_{21} & \theta_{22} & \theta_{23} \end{vmatrix} \begin{pmatrix} x_i^t \\ y_i^t \\ 1 \end{pmatrix} \quad (2.1)$$

**Giai đoạn rút trích đặc trưng thị giác:** Tương tự các bài toán khác trong cùng lĩnh vực thị giác máy tính, như nhận dạng đối tượng, phát hiện đối tượng, ..., bước này sẽ giúp mô hình học được đặc trưng bất biến của từng kí tự *invariant features* mà không phải lệ thuộc vào bất kì một phông chữ, yếu tố màu sắc nào; ngoài ra, mục đích của bước này nhằm hạn chế đi tác động của nhiễu nền hay bất kì thông tin không liên quan đến kí tự của văn bản.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.5: Hình mô tả cách hoạt động của một mạng STN. Cụ thể, mạng STN bao gồm 3 thành phần chính. Bước đầu (1), một mạng giúp dự đoán một ma trận  $\theta$  (từ đầu vào là ảnh/đặc trưng đầu vào được rút trích bởi 1 mạng CNN). Ma trận  $\theta$  sẽ trở thành tham số đầu vào cho phép biến đổi affine  $\mathcal{T}_\theta$ . Bước tiếp theo (2), ta cần tạo được ánh xạ được tọa độ từ ảnh gốc  $U$  sang tọa độ của ảnh  $V$ . Bước cuối cùng (3), ta cần nội suy được giá trị của pixel trong ảnh  $V$  dựa trên giá trị pixel từ ảnh gốc và cặp tọa độ suy từ bước vừa rồi. (Nguồn: Bài báo [2])



Hình 2.6: Hình minh họa 2 ví dụ về hai ma trận affine  $\mathcal{T}_\theta$ . Ví dụ a) (Trivial examples), ma trận biến đổi  $\theta$  là một ma trận đơn vị  $I$ , nghĩa là phép biến đổi  $\mathcal{T}_\theta$  hoàn toàn không thay đổi gì ảnh gốc  $U$ . Ngược lại, trường hợp  $\theta$  là ma trận khác ma trận đơn vị nêu trên, sẽ cho ta kết quả là hình ảnh  $V$  từ ảnh gốc  $U$ . (Nguồn: Bài báo [2])

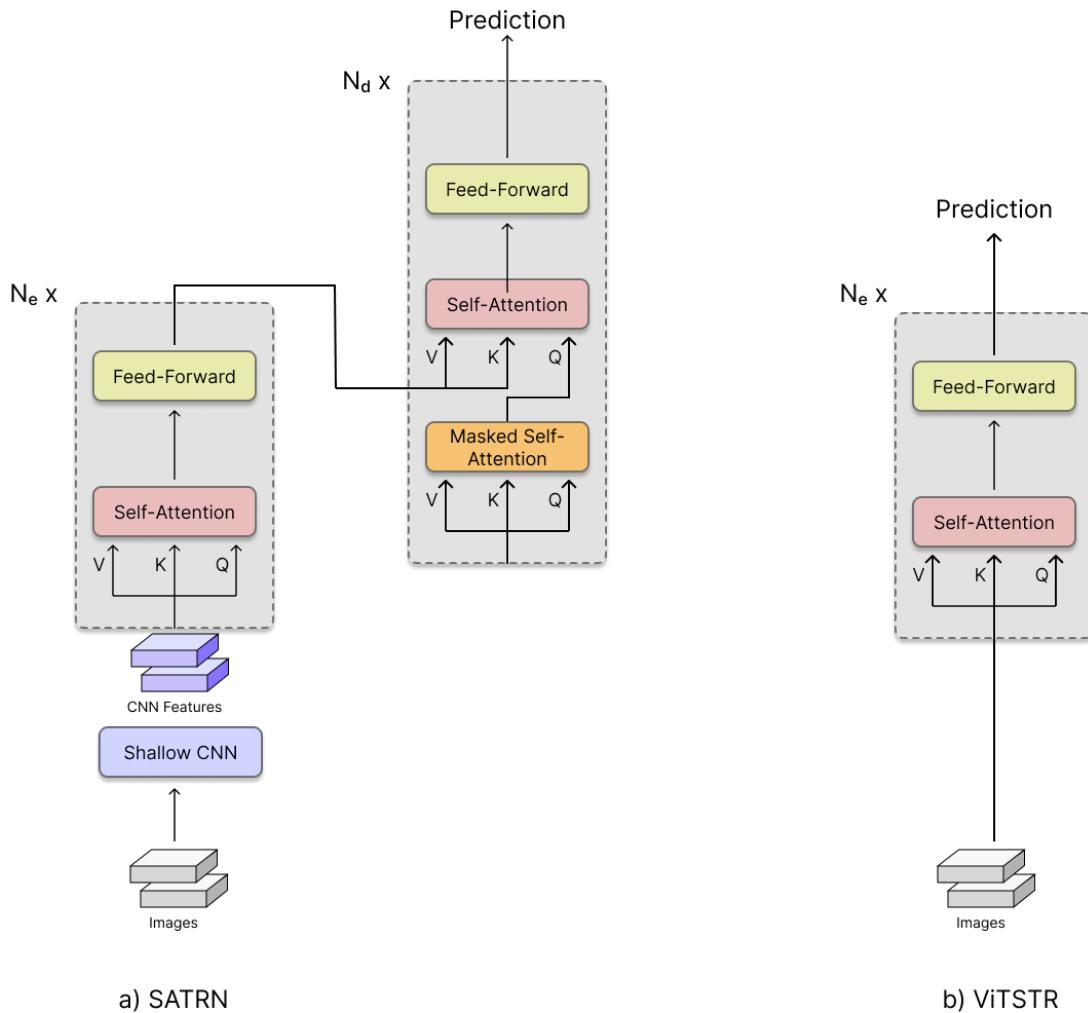
## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

Một vài phương pháp sử dụng mạng VGG[17] như RARE[23], ... ; tuy nhiên, các phương pháp [22],[24] sử dụng ResNet[18] để thu lại đặc trưng học sâu tốt hơn. Đặc biệt hơn nữa, là trong những năm gần đây, nhờ có sự phát triển của Vision Transformer, các phương pháp như SATRN[6] ViTSTR[7], CornerTransformer[1], đã không còn (hay hoàn toàn bỏ hẳn mạng CNN), mà thay vào đó là sử dụng backbone là mạng Vision Transformer để rút trích đặc trưng. Ví dụ như SATRN hay CornerTransformer vẫn còn giữ lại một lớp mạng CNN - nhưng không quá sâu (chỉ khoảng 1 - 2 lớp tích chập, dùng để rút trích một vài đặc trưng cấp thấp), các đặc trưng này sẽ được chú ý bởi khối encoder (gồm 12 lớp self attention), theo đó, các khối này sẽ giúp mô hình tập trung vào được kí tự trong không gian 2D mà không nhất thiết cần đến đến mạng nắn thẳng. Tuy nhiên, với phương pháp đề xuất bởi công trình của Atienza et al. [7], toàn bộ phần rút trích đặc trưng đã được thay thế bằng mạng Vision Transformer, điều này góp phần thay, và so với SATRN, ViTSTR hoàn toàn loại bỏ khối decoder, những cải tiến này khiến ViTSTR tăng tốc độ xử lý (xem hình 2.7)

Trong những năm gần đây, các mô hình rút trích đặc trưng ngày càng tân tiến, kết quả của quá trình này là khả năng biểu diễn thị giác ngày càng giàu ngữ nghĩa hơn. Tận dụng điều này, các thuật toán STR ngày càng hiệu quả hơn, kể cả khi xử lý với môi trường phức tạp. Điều hiển nhiên là việc sử dụng backbone

**Mô hình tuần tự:** Mô hình tuần tự có nhiệm vụ làm cầu nối giữa bước rút trích đặc trưng thị giác và bước dự đoán kết quả. Thông qua bước này, mô hình có thể nắm bắt thông tin theo ngữ cảnh trong một chuỗi kí tự, phục vụ cho bước dự đoán từng kí tự. Theo như các nghiên cứu trước đã chỉ ra, mô hình tuần tự giúp cải thiện độ ổn định của thuật toán, và theo lý thuyết, bước này phù hợp cho với hơn so với hướng dự đoán độc lập từng kí tự.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.7: Hình minh họa đơn giản chiến lược thiết kế các thuật toán STR. Trong đó, hình a) ví dụ cho thuật toán sử dụng một mạng CNN, một khối encoder, và một khối decoder. Ngược lại, ViTSTR loại bỏ hoàn toàn khối CNN và khối decoder. Lưu ý: Hình chỉ minh họa x sự khác biệt giữa 2 chiến lược thiết kế thuật toán, không mô tả hoàn toàn các bước của 2 thuật toán được nêu.

## 2.4 Cơ sở lý thuyết

### 2.4.1 Mô hình Sequence-to-sequence

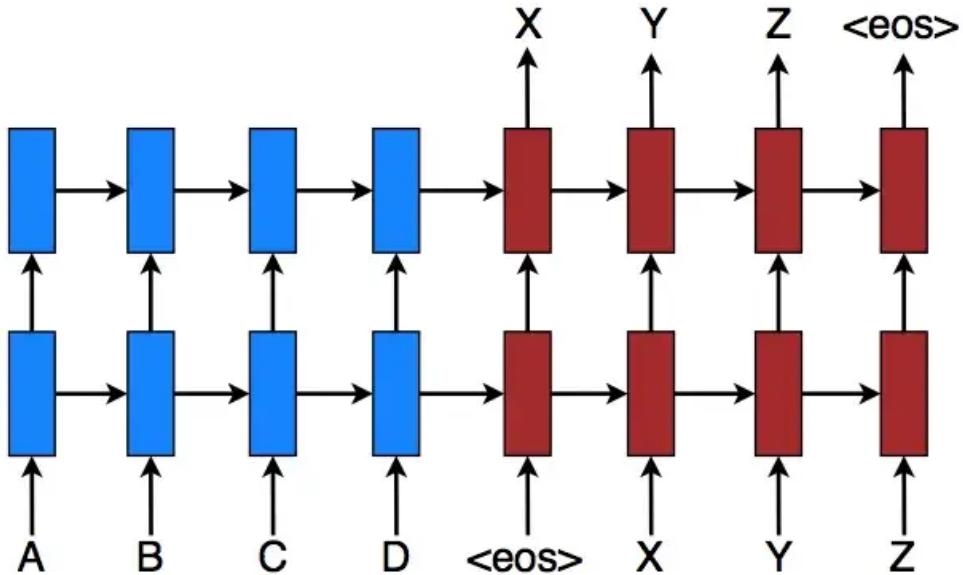
#### 2.4.1.1 Kiến trúc tổng quan

Mô hình Sequence-to-sequence (Seq2seq) được giới thiệu năm 2014 bởi các nhà nghiên cứu đến từ Google [25]. Mô hình seq2seq được sinh ra với mục đích dịch một chuỗi các token ở ngôn ngữ gốc sang ngôn ngữ khác. Với một chuỗi các token  $x = \{x_1, \dots, x_n\}$ , seq2seq sẽ ánh xạ  $x$  sang một chuỗi token  $y = \{y_1, \dots, y_m\}$ , độ dài chuỗi  $x$  và  $y$  có thể khác nhau. Trong quá trình huấn luyện, mục tiêu của mô hình là tối đa hóa xác suất có điều kiện  $p(y_1, \dots, y_m | x_1, \dots, x_n)$ . Mô hình seq2seq sử dụng kiến trúc Encoder-Decoder để có thể biến chuỗi token có độ dài  $n$  thành câu có độ dài  $m$ . Bộ Encoder và Decoder là các mô hình RNN, thông thường, các mô hình cải tiến của RNN như GRU và LSTM sẽ được sử dụng cho cả Encoder và Decoder. Trong bài báo [25], nhóm tác giả sử dụng LSTM cho cả 2 module vì nó cho kết quả tốt nhất trong thực nghiệm của họ. Điều này có thể được giải thích được là vì mô hình LSTM có khả năng giải quyết được vấn đề phụ thuộc dài.

Trong mô hình seq2seq gồm 2 thành phần chính là Encoder và Decoder có chức năng:

- Encoder: bộ mã hóa có nhiệm vụ ánh xạ một chuỗi token đầu vào vào một không gian vector có chiều cố định, vector đó được gọi là vector trạng thái ẩn và nó đại diện cho chuỗi đầu vào tại bước mã hóa cuối cùng. Ở mỗi bước mã hóa, đầu vào của Encoder sẽ là token ở bước hiện tại và vector trạng thái ẩn để tạo ra vector trạng thái ẩn biểu diễn chuỗi đầu vào ở bước mã hóa đó.
- Decoder: bộ giải mã được sử dụng để tạo ra chuỗi token ở ngôn ngữ đích.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.8: Kiến trúc mô hình sequence-to-sequence (Nguồn: Bài báo [3])

Bộ giải mã nhận đầu vào là vector trạng thái ẩn, tại mỗi bước mã hóa, một kí tự sẽ được tạo ra, bước giải mã sẽ lặp lại cho đến khi gặp kí tự kết thúc hoặc chuỗi đầu ra đạt đến độ dài quy định trước. Hàm xác xuất có điều kiện được tính theo công thức sau:

$$p(y_1, \dots, y_m | x_1, \dots, x_n) = \prod_{t=1}^m p(y_t | s, y_1, \dots, y_{t-1}) \quad (2.2)$$

Trong công thức trên, vế phải  $p(y_t | s, y_1, \dots, y_{t-1})$  biểu diễn sự xuất hiện của  $y_t$  với  $s$  đại diện cho câu đầu vào và các token xuất hiện trước thời gian  $y_{t-1}$ . Phân phối này được tính bởi hàm softmax trên tất cả các token của tập từ điển ở ngôn ngữ gốc.

Công thức trên có thể viết lại như sau:

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

---

$$\log p(y|x) = \sum_{t=1}^m \log p(y_t|y_{<t}, s) \quad (2.3)$$

Mỗi token  $y_t$  có xác suất được tính theo công thức:

$$p(y_t|y_{t < s}, s) = \text{softmax}(g(h_t)) \quad (2.4)$$

Trong đó, hàm  $g$  là hàm biến đổi trạng thái ẩn  $h_t$  của Decoder tại thời gian tương ứng  $t$ , kết quả của hàm  $g$  là một vector có số chiều bằng với số lượng token trong tập từ điển đích. Cuối cùng kết quả sẽ được đi qua hàm Softmax để tính xác suất dự đoán token  $y_t$ . Hàm  $g(h_t)$  được tính như sau:

$$g(h_t) = f(h_{t-1}, s) \quad (2.5)$$

Trong đó thì hàm  $f$  là hàm tính giá trị chung của các mô hình RNN được sử dụng, hàm này được tính dựa trên trạng thái ẩn ở bước thời gian trước đó  $h_{t-1}$  và token đầu vào hiện tại  $s$ . Trong mô hình của [25], vector  $s$  biểu diễn cho câu nguồn chỉ được sử dụng ở bước giải mã đầu tiên của bộ Decoder.

Hàm mất mát được sử dụng trong quá trình huấn luyện có công thức như sau:

$$L = - \sum_{i=1}^m \sum_{j=1}^V q_{i,j} \log p_{i,j} \quad (2.6)$$

Trong đó,  $q_{i,j}$  là phần tử thứ  $j$  của one-hot vector  $q_i$  có số chiều là  $V$  biểu diễn cho token thứ  $i$  trong tập dữ liệu huấn luyện. Vector  $p_{i,j}$  là phần tử thứ  $j$  của vector  $p_i$  là dự đoán của mô hình được tính từ hàm  $p_i = \text{softmax}(g(h_i))$  và vector  $p_i$  cùng có số chiều là  $V$ .

Có thể nói rằng, trong quá trình dự đoán sinh ra chuỗi dịch  $\hat{y}$ , cho trước chuỗi đầu vào  $x$ , bằng cách dự đoán kí tự có xác suất xuất hiện cao nhất trong từ điển cho trước:

$$\hat{y} = \operatorname{argmax}_y (p(y|x)) \quad (2.7)$$

### 2.4.1.2 Cơ chế giải mã với Greedy Search

Trong quá trình giải mã ở Decoder của mô hình seq2seq, Greedy Search được sử dụng để dự đoán token đầu ra ở mỗi bước giải mã. Ở mỗi bước giải mã, đầu ra của Decoder sẽ là một vector trạng thái ẩn và một vector  $p$ , vector  $p$  sẽ được dùng để dự đoán token ở bước giải mã hiện tại. Vector  $p$  có số chiều tương ứng với số lượng từ trong tập từ điển ở ngôn ngữ đích. Vector  $p$  sẽ được đi qua hàm softmax để tạo thành một vector có cùng số chiều với các giá trị của nó biểu diễn xác suất xuất hiện của token ở vị trí tương ứng trong vector đó. Sau cùng, chỉ cần dùng hàm argmax để có thể biết được vị trí của token có xác suất xuất hiện cao nhất trong tập từ điển. Các bước giải mã này sẽ ngừng lại khi gặp kí tự kết thúc  $<EOS>$  hoặc số từ trong câu sinh ra đạt độ dài tối đa quy định.

### 2.4.1.3 Cơ chế giải mã với Beam Search

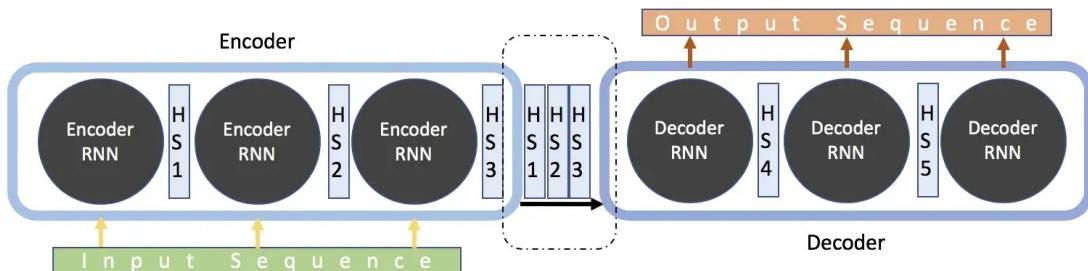
Cơ chế giải mã Greedy Search có tốc độ nhanh, dễ dàng cài đặt, tuy nhiên nó lại có nhược điểm. Với Greedy Search, nếu các kí tự đầu tiên của chuỗi dự đoán không chính xác có thể dẫn đến kết quả của câu sinh ra. Do đó, thuật toán Beam Search được sử dụng để khắc phục nhược điểm của thuật toán Greedy Search. Về cơ bản, thay vì ở mỗi bước giải mã chỉ lấy token có xác suất cao nhất, thuật toán Beam Search sẽ giữ lại  $k$  token có xác suất xuất hiện cao nhất (với  $k$  là beam width). Sau khi kết thúc việc giải mã (khi gặp kí tự kết thúc  $<EOS>$  hoặc câu sinh ra đạt độ dài quy định), thuật toán Beam Search sẽ chọn ra câu có xác suất  $p(y_1, y_2, \dots, y_{<EOS>} | x_1, x_2, \dots, x_n)$ . Với ý tưởng này, thuật toán Beam Search có thể tạo ra câu có kết quả tốt hơn Greedy Search trong những trường hợp những từ đầu tiên của câu bị dự đoán không chính xác.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

### 2.4.2 Mô hình Attention

Một vấn đề dễ gặp phải khi sử dụng mô hình seq2seq là nó phụ thuộc rất nhiều vào ngữ cảnh được biểu diễn bởi vector ẩn, đầu ra của encoder, do đó rất khó để giữ được ngữ nghĩa của cả chuỗi khi nó quá dài. Khi câu quá dài, khả năng cao là vector ngữ nghĩa ở đầu chuỗi đầu vào sẽ bị mất ở cuối câu. Do đó, nhóm nghiên cứu [21] và [3] đề xuất sử dụng mô hình Attention (thường được gọi là cơ chế attention) để giải quyết được vấn đề phụ thuộc dài của vector trạng thái ẩn.

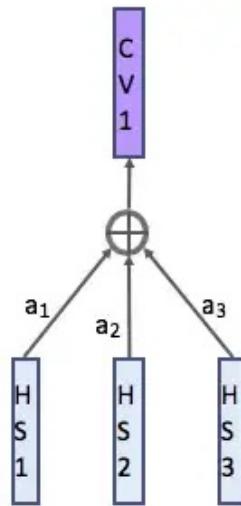
Kiến trúc của mô hình sử dụng cơ chế Attention cũng tương tự như mô hình seq2seq, tuy nhiên có điều chỉnh để tận dụng được hết các vector trạng thái ẩn  $h_i$  nhằm giải quyết vấn đề phụ thuộc dài. Cụ thể, ở module Decoder, thay vì chỉ sử dụng vector trạng thái ẩn đầu ra của Encoder, Decoder sử dụng toàn bộ những vector trạng thái ẩn ở mỗi bước mã hóa của Encoder 2.9, điều này giúp cho mô hình tận dụng được toàn bộ thông tin ở tất cả các bước mã hóa.



Hình 2.9: Kiến trúc tổng quan của mô hình seq2seq có sử dụng cơ chế Attention (Nguồn: Internet<sup>1</sup>)

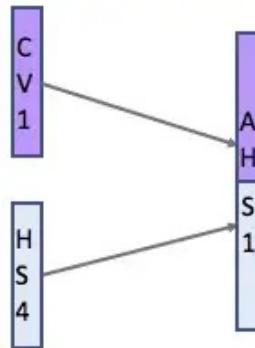
Ở module Decoder, để tận dụng được các thông tin của tất cả các vector trạng thái ẩn, một vector ngữ nghĩa (context vector) được tạo ra bằng cách tính tổng của các tích giữa trọng số alignment  $a_i$  và vector trạng thái ẩn tương ứng như hình bên dưới 2.10.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.10: Vector ngữ nghĩa trong kiến trúc attention (Nguồn: Internet<sup>1</sup>)

Để sử dụng vector ngữ nghĩa trong quá trình dự đoán, ở mỗi bước dự đoán, vector trạng thái ẩn attention được tạo ra bằng cách nối vector ngữ nghĩa và vector trạng thái ẩn ở bước thời gian tương ứng lại với nhau [2.11](#).

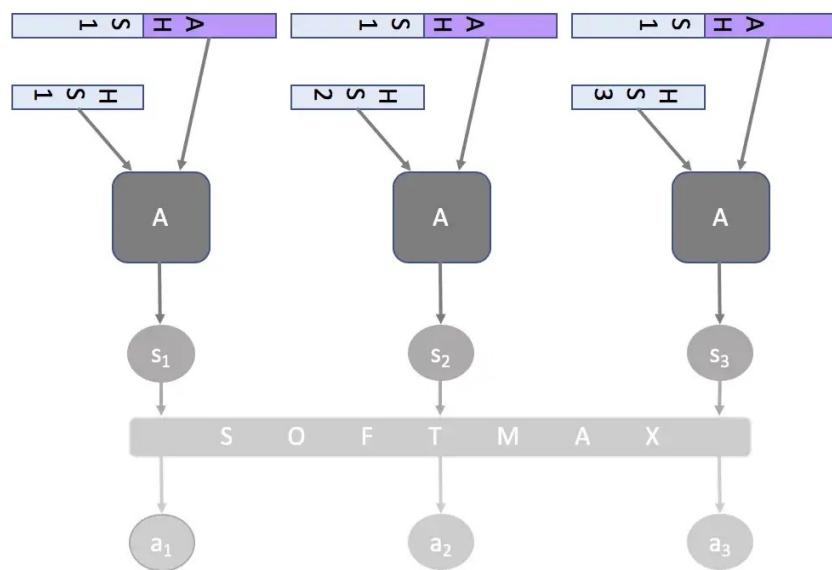


Hình 2.11: Cách mô hình attention sử dụng vector ngữ nghĩa để tạo ra vector trạng thái ẩn attention (Nguồn: Internet<sup>1</sup>)

Sau cùng để biết được đầu vào nào quan trọng trong bước dự đoán, mô hình cần phải dự đoán xem đầu vào nào là quan trọng, để làm được điều này, chúng ta cần phải có attention score hay điểm số attention. Điểm số attention này được một mô hình dự đoán riêng, gọi là mô hình alignment, mô hình này được huấn

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

luyện chung với mô hình seq2seq. Với mô hình alignment, điểm số attention sẽ đánh giá mỗi đầu vào (được biểu diễn bởi vector trạng thái ẩn) sẽ nêu chú ý vào đầu ra nào trước đó (được biểu diễn bởi vector trạng thái ẩn attention) và nên chú ý vào input đầu vào nào với mỗi kết quả đầu ra trước đó. Sau cùng, hàm softmax sẽ được tính trên tất cả các đầu ra từ mô hình alignment để có được điểm số attention.



Hình 2.12: Tính điểm số attention (Nguồn: Internet<sup>1</sup>)

### 2.4.3 Mô hình Transformer

Vào năm 2017, cộng đồng nghiên cứu xử lý ngôn ngữ tự nhiên có nhiều đột phá mới với sự ra mắt của mô hình Transformer [4]. Kể từ khi ra mắt, Transformer đã phá vỡ được nhiều kỷ lục trong các tác vụ xử lý ngôn ngữ tự nhiên như dịch máy. Cũng giống như nhiều mô hình dịch máy khác ra mắt trước đó, mô hình Transformer cũng sử dụng kiến trúc Encoder-Decoder, Encoder chịu trách

<sup>1</sup><https://towardsdatascience.com/day-1-2-attention-seq2seq-models-65df3f49e263>

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

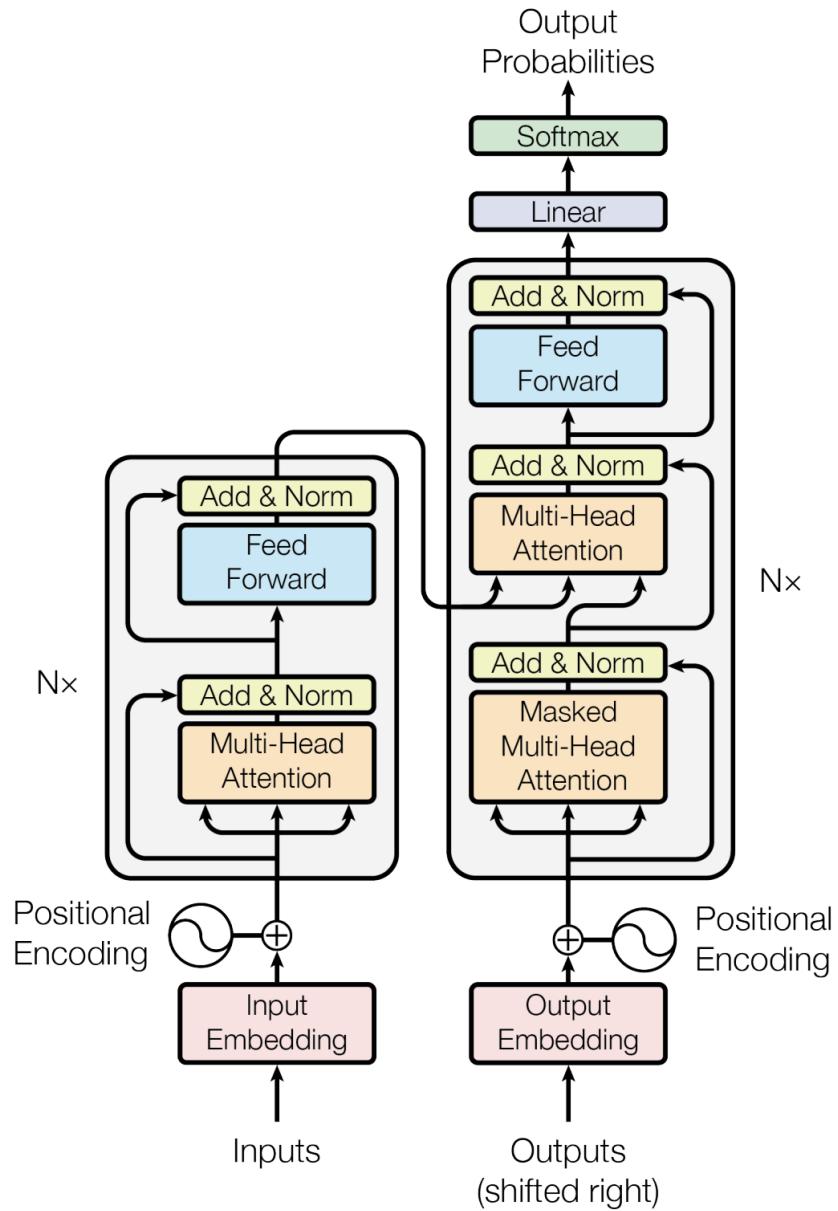
nhiệm trích xuất thông tin ngữ nghĩa ở câu gốc, sau đó Decoder có nhiệm vụ sinh ra câu ở ngôn ngữ đích sử dụng vector ngữ nghĩa được tạo ra bởi Encoder. Tuy nhiên, khác so với các mô hình trước đó, Transformer chủ yếu sử dụng các khối Encoder, Decoder dựa trên cơ chế self-attention thay vì các mô hình RNNs như LSTM hay GRU, tuy nhiên các kiến trúc đó có nhiều nhược điểm có thể kể đến như:

- Đối với những dòng văn bản quá dài, chẩn hạn như đầu vào là cả một bài viết thay vì một câu hay một đoạn văn, do phần decoder lấy vector đầu ra của encoder, khi RNNs trong encoder duyệt qua chuỗi quá nhiều lần, nó sẽ bị mất thông tin ở của đầu vào, đặc biệt là thông tin ở đầu
- Do một chuỗi các từ phải bảo toàn thứ tự thì mới có nghĩa, RNNs chỉ có thể đưa lần lượt các từ vào huấn luyện lần lượt theo thứ tự, ngoài ra quá trình tính lan truyền ngược cũng do đó mà phức tạp hơn, lâu hơn, do đó việc huấn luyện mô hình seq2seq hay RNNs nói chung rất chậm.
- Kiến trúc LSTM không phù hợp cho việc transfer learning, khi thực hiện một tác vụ khác trên một tập dữ liệu khác, ta sẽ phải huấn luyện lại mô hình từ đầu, tốn rất nhiều thời gian.

Để giải quyết những vấn đề trên, Google đề xuất ra các giải pháp như sử dụng cơ chế attention để tránh việc mất thông tin khi đầu vào quá dài, sử dụng positional encoding, masking token để có thể huấn luyện song song mà không cần tuần tự, tăng tốc quá trình huấn luyện. Ngoài ra, Transformer còn cho phép đọc dữ liệu đầu vào theo cả hai hướng như trong Bidirection LSTM (BiLSTM) mà không cần phải xếp chồng thêm một mô hình ngược chiều như LSTM.

Cũng tương tự như mô hình seq2seq, Transformer cũng có hai thành phần chính là encoder và decoder, tuy nhiên cấu tạo và cơ chế hoạt động hoàn toàn khác nhau.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.13: Kiến trúc của mô hình Transformer (Nguồn: Bài báo [4])

- **Encoder:** trong Transformer, mỗi khối encoder bao gồm  $N$  layer giống nhau ( $N=6$  trong bài báo), mỗi layer bao gồm 2 layer con: Multi-Head Attention và Feed forward network. Và ở mỗi layer con đều có sử dụng residual connection, việc sử dụng residual connection sẽ giúp cho mô hình

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

có thể mở rộng hơn theo chiều sâu, nhờ việc tránh được hiện tượng mất mát đạo hàm (gradient vanishing).

- **Decoder:** tương tự như encoder, decoder cũng được tạo thành từ 6 layer, với mỗi layer gồm 2 layer con, tuy nhiên, có thêm một layer con khác được thêm vào là Masked Multi-Head Attention, layer con này sẽ nhận đầu vào là đầu ra của encoder, và layer con này cũng có residual connection.

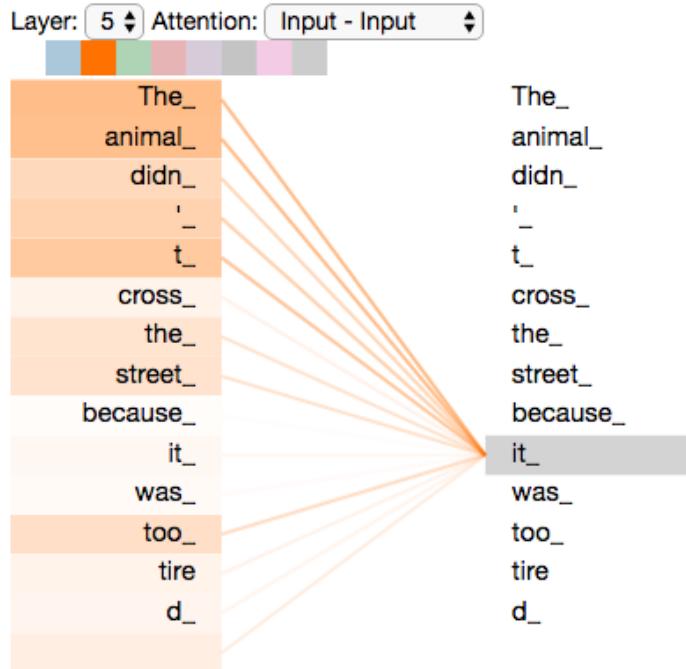
### 2.4.3.1 Transformer - Cơ chế Self-attention

Trước khi đi vào chi tiết cách hoạt động của mô hình Transformer, chúng ta cần phải tìm hiểu chi tiết về cơ chế Self-attention, trái tim của mô hình Transformer *Attention is all you need*. Chúng ta có thể xem như là một thuật toán tìm kiếm, với một câu đầu vào, khi xử lý từng từ trong câu, cơ chế self-attention sẽ tập trung chú ý vào các từ liên quan nhất với từ đang xử lý. Giả sử với câu đầu vào "The animal didn't cross the street because it was too tired", khi xử lý đến từ "it" thì mô hình cần chú ý đến từ gì? là "street" hay "animal", câu hỏi này có thể đơn giản với con người nhưng với máy tính thì không. Cơ chế self-attention sẽ cho phép mô hình tập trung vào chữ "animal" thay vì "street" khi nó đang xử lý từ "it"<sup>2.14</sup>, điều này giúp cho mô hình có thể mã hóa được từ hiện tại tốt hơn khi nó quan sát toàn bộ câu đầu vào để biết nó cần phải chú ý vào đâu. Trong các mô hình RNNs, mỗi "cell" trong mô hình phải cố gắng duy trì được trạng thái ẩn và ở "cell" giải mã, kết hợp trạng thái ẩn và các token được sinh ra trước đó để dự đoán. Với cơ chế self-attention, bộ mã hóa sẽ được bảo là nó có thể "hiểu" được mức độ liên quan giữa các từ với nhau, từ đó có thể dự đoán chính xác hơn.

Đầu vào của Self-attention là 3 vector query, key và value, các vector này được tính bằng cách nhân đầu vào với các ma trận trọng số tương ứng với query,

<sup>2</sup><https://jalammar.github.io/illustrated-transformer/>

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.14: Hình ảnh trực quan hóa quá cách hoạt động của Self-attention (Nguồn: Internet<sup>2</sup>)

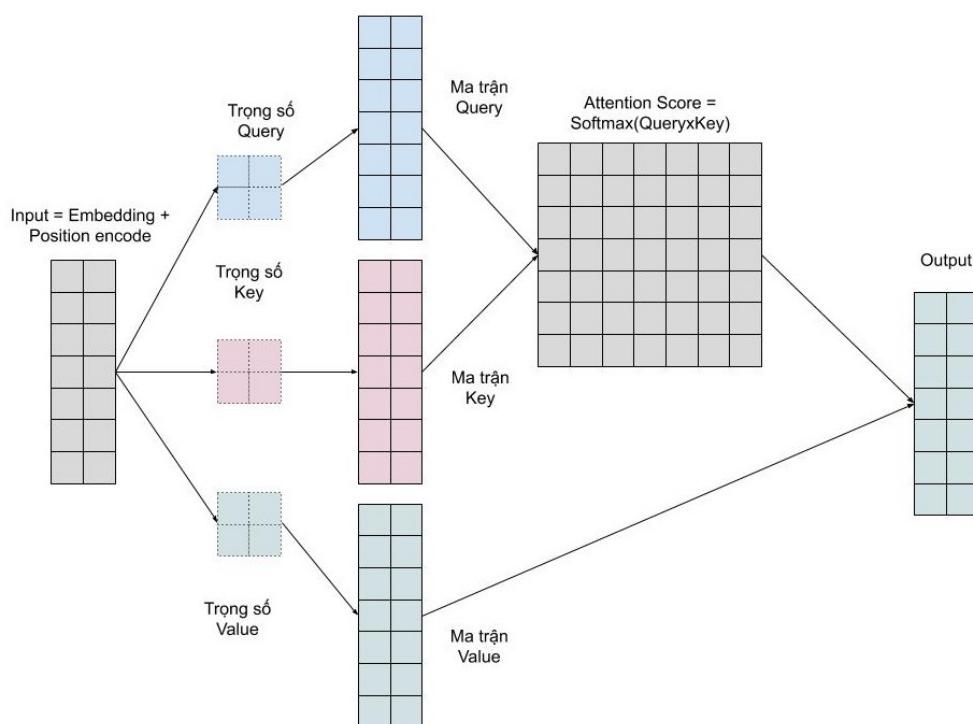
key và value.

- Query vector là vector dùng để chứa thông tin của từ đang xử lý ở bước thời gian hiện tại (là từ dùng để tìm kiếm, so sánh).
- Key vector là vector dùng để biểu diễn thông tin các từ được so sánh với từ đang xử lý ở trên.
- Value vector là vector biểu diễn nội dung, ý nghĩa của các từ.

Vector attention cho một từ thể hiện tính tương quan giữa các vector query, key và value. Vector này được tính bằng cách nhân tích vô hướng giữa vector query và vector key, sau đó dùng hàm softmax để chuẩn hóa kết quả, sau cùng chỉ cần nhân với vector value. Cụ thể từng bước như sau:

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

- Bước 1: Tính ma trận query, key và value bằng cách nhân ma trận đầu vào với ma trận trọng số tương ứng.
- Bước 2: Nhân ma trận query và key vừa tính được với nhau. Phép tính này thể hiện việc "học" sự tương quan giữa vector query và key. Sau đó, kết quả được huấn hóa về khoảng [0; 1] bằng hàm softmax, với giá trị càng cao thì sự tương quan giữa query và key càng nhiều và ngược lại.
- Bước 3: Kết quả sẽ được tạo ra bằng cách nhân ma trận có được ở bước 2 với vector value.



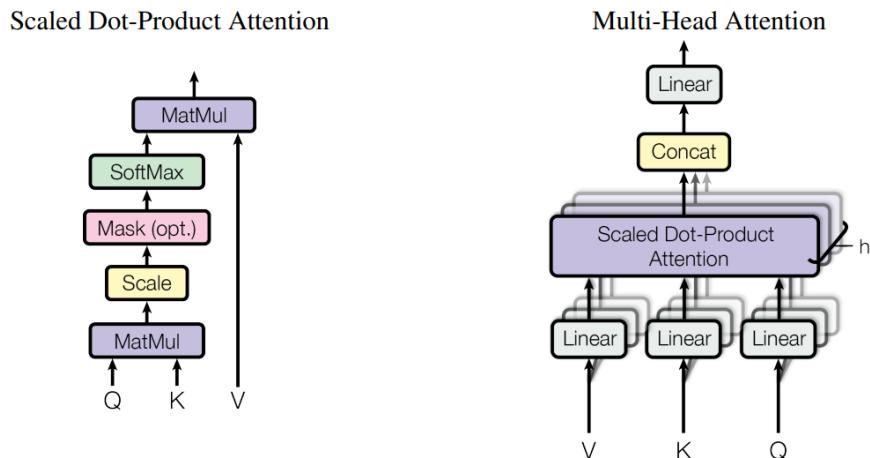
Hình 2.15: Cách hoạt động của self-attention (Nguồn: Internet<sup>2</sup>)

Hai hàm attention được sử dụng phổ biến là hàm additive attention và dot-product attention. Đặc điểm của hàm dot-product attention là sau khi nhân ma trận query và key với nhau, nó sẽ được chia cho  $\sqrt{d_k}$ , trong đó  $d_k$  là số chiều của vector key, trước khi đi qua hàm softmax.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

### 2.4.3.2 Transformer - Multi-Head Attention

Đối với các mô hình attention truyền thống, chuỗi đầu vào không thể được khái quát hết do chỉ có một attention head trong nó. Ví dụ câu "Trong buổi tiệc, Minh đã bỏ An về trước. Bạn ấy có việc bận", nếu chỉ có một cơ chế attention thì khó có thể xác định được từ "Bạn" trong câu muốn nói An hay Minh. Do đó, Transformer đã sử dụng Multi-Head Attention để khái quát được ý nghĩa của câu dưới nhiều góc độ. Multi-Head Attention là một trong những cơ chế quan trọng của mô hình Transformer. Attention có thể hiểu là một cơ chế ánh xạ một query và một cặp key-value sang một đầu ra, với query, key và value đều là vector. Đầu ra chính là tổng có trọng số giữa các giá trị trong values, với trọng số được tính từ query tương ứng với key. Hoặc nói cách khác, ta có thể xem query là câu truy vấn để tìm được mã câu tương ứng (key) và ta sẽ có được nội dung, hay ý nghĩa của các câu đó chính là value. Trong bài báo, nhóm tác giả ký hiệu query, key và value lần lượt là Q, K và V.

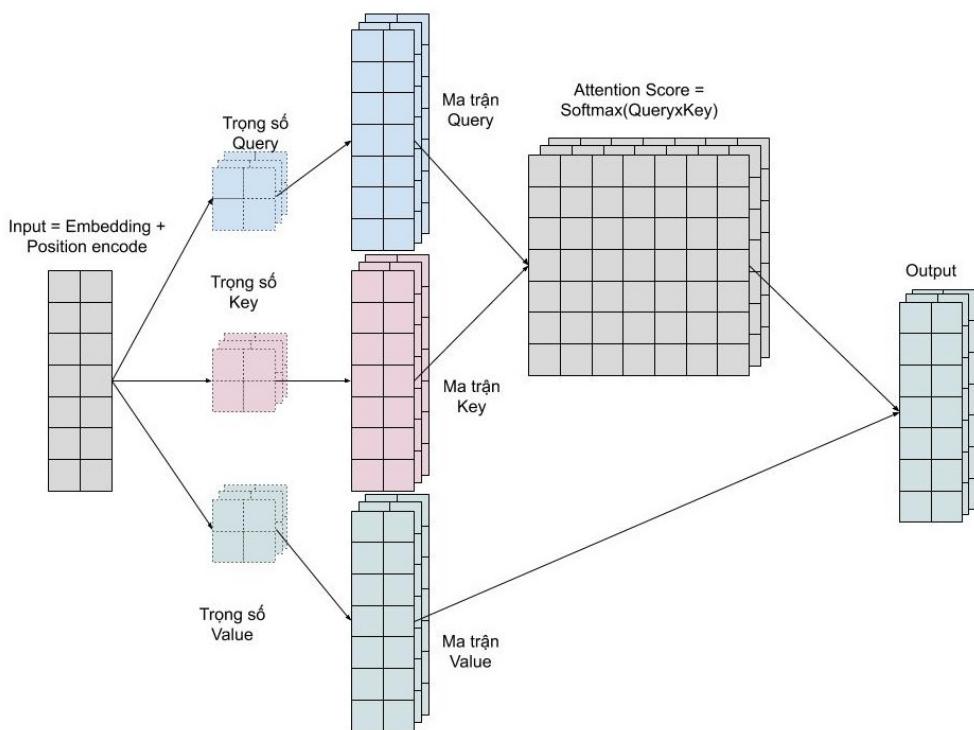


Hình 2.16: Hình ảnh minh họa cơ chế attention và kiến trúc Multi-Head Attention (Nguồn: Bài báo [4])

Trong kiến trúc Multi-Head Attention, với mỗi chuỗi đầu vào ta có tương ứng, Q, K và V sẽ được được biến đổi tuyến tính qua việc đi qua một lớp Linear

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

(mỗi Q, K và V đi qua lớp Linear tương ứng với nó như hình 2.16) để học được các trọng số attention. Và không chỉ đi qua một lớp tuyến tính như thế này, K, V và Q sẽ phải đi qua nhiều lớp tuyến tính chồng lên nhau, nhằm đạt được Multi-Head attention 2.17, chính việc này giúp cho mô hình có khả năng bao quát được nghĩa của câu dưới nhiều khía cạnh khác nhau. Sau khi đi qua lớp Linear, các trọng số attention được kết hợp với nhau thông qua phép Scaled Dot-Product Attention.



Hình 2.17: Cách hoạt động của Muli-Head attention (Nguồn: Internet<sup>2</sup>)

Scaled Dot-Product Attention, tuy nghe tên khá phức tạp tuy nhiên ý tưởng thì lại khá đơn giản nhưng hiệu quả. Mình chỉ cần lấy vector Q nhân với vector K để tìm ra các từ có liên quan đến nhau nhất, tuy nhiên, khi số chiều của vecotr K tăng lên, sẽ gây hiện tượng đạo hàm có giá trị cực nhỏ (do phép dot product giữa 2 vector  $q \cdot k = \sum_{i=1}^{d_k} q_i k_i$  sẽ có giá trị trung bình cực nhỏ và phương sai lớn

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

tương ứng với  $d_k$ ). Để giải quyết vấn đề này, kết quả phép dot product ở trên sẽ được đem chia cho  $d_k$ . Sau đó kết quả này sẽ được đi qua hàm softmax và nhân với  $V$  để giữ lại những thông tin quan trọng, từ quan trọng trong câu. Cuối cùng, các vector đầu ra của các lớp Scaled Dot Product Attention sẽ được nối lại qua phép concatenate và đi qua lớp Linear. Ta có công thức hoàn chỉnh như sau:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (2.8)$$

### 2.4.3.3 Transformer - Positional Encoding

Do mô hình Transformer không có kiến trúc hồi quy, do đó, để tận dụng được thông tin về vị trí tương đối hoặc tuyệt đối của các từ trong câu, cần phải cách để bổ sung thông tin về vị trí. Trong bài báo, nhóm tác giả sử dụng hàm sine và cosine ở các tầng số khác nhau:

$$\text{PE}_{(pos,2i)} = \sin(pos/10000^{2i/d_{model}}) \quad (2.9)$$

$$\text{PE}_{(pos,2i+1)} = \cos(pos/10000^{2i/d_{model}}) \quad (2.10)$$

Như vậy, có thể thấy rằng lớp Positional Encoding có các tính chất sau:

- Có thể biểu diễn được thông tin vị trí trong các câu dài
- Mỗi time-step trong câu đảm bảo có mã hoá duy nhất: vì một vị trí không thể nào có cùng lúc 2 từ, nên chắc chắn thông tin vị trí phải là duy nhất trong một câu
- Khoảng cách giữa 2 time-step liền kề trong embedding giữa 2 câu khác nhau phải bằng nhau

## **2. Cơ sở lý thuyết và các nghiên cứu liên quan**

---

### **2.4.4 Mô hình Vision Transformer**

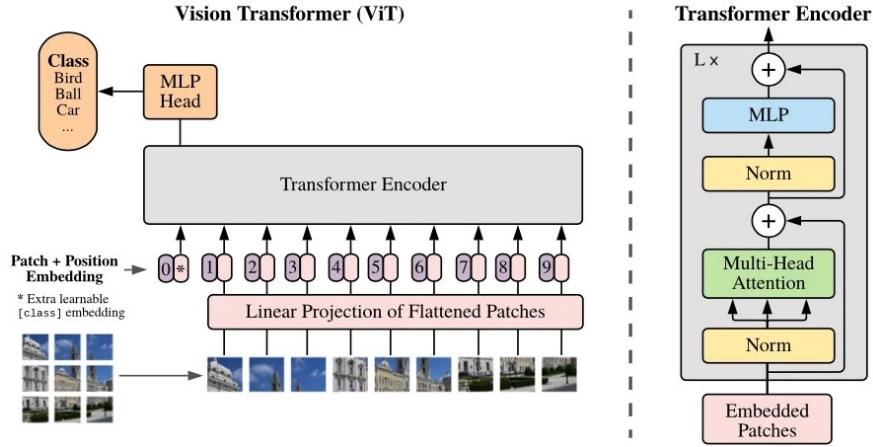
Kể từ khi mô hình Transformer [4] được giới thiệu vào năm 2017 và đạt được nhiều thành công đáng kể trong lĩnh vực xử lý ngôn ngữ tự nhiên, các nhà nghiên cứu đã tìm cách để có thể áp dụng Transformer vào trong các tác vụ thị giác máy tính. Kết quả là vào năm 2020, mô hình Vision Transformer (ViT) [5] ra đời. ViT kể từ khi ra mắt đã đánh bại được các mô hình tiên tiến nhất trên các tác vụ thị giác máy tính, nó cạnh tranh trực tiếp với các mô hình CNN vốn đã thống trị rất lâu trong lĩnh vực thị giác máy tính. Vậy thì tại sao Vision Transformer lại có thể đánh bại được CNN, trong phần này sẽ trình bày về kiến trúc mô hình và cách hoạt động của mô hình ViT để hiểu rõ hơn tại sao.

Mô hình ViT có kiến trúc tổng quan như hình 2.18, về cơ bản thì mô hình ViT sử dụng lại khôi Encoder của mô hình Transformer. Các bước tính toán của mô hình ViT như sau:

1. Chia hình ảnh thành các mảng (patch) với kích thước từng mảng cố định
2. Làm phẳng các mảng hình ảnh
3. Tạo các feature embedding có chiều thấp hơn từ các mảng hình ảnh phẳng này
4. Bao gồm thứ tự các mảng
5. Chuỗi feature embedding được làm đầu vào cho transformer encoder
6. Thực hiện pre-train đối với mô hình ViT với các nhãn hình ảnh, sau đó được giám sát hoàn toàn trên một tập dữ liệu lớn
7. Tinh chỉnh model trên bộ dữ liệu riêng của từng bài toán

Ở bước biến hình ảnh đầu vào thành các vector embedding tương tự như token trong NLP, hình ảnh được chia thành các mảng có kích thước bằng nhau.

## 2. Cơ sở lý thuyết và các nghiên cứu liên quan



Hình 2.18: Kiến trúc mô hình ViT (Nguồn: Bài báo [5])

Ta có đầu vào là ảnh 2D  $x \in R^{H \times W \times C}$ , trong đó  $H, W, C$  tương ứng lần lượt là chiều cao, chiều rộng và số chiều của ảnh. Sau đó được chia thành các mảng bằng nhau  $x_p \in R^{N \times (P^2 \cdot C)}$ , trong đó  $(P, P)$  là kích thước của từng mảng ảnh,  $H = HW / P^2$ . Vì đầu vào của Transformer Encoder chỉ chấp nhận vector có kích thước cố định  $D$ , các mảng ảnh được duỗi thẳng và đi qua một lớp tuyến tính có thể học được để chuyển sang vector có  $D$  chiều.

Cũng tương tự như trong mô hình Transformer, thứ tự của các token rất quan trọng, thứ tự của các mảng ảnh trong ViT cũng tương tự. Do đó, một lớp Position Embedding được thêm vào để giữ được thông tin vị trí của các mảng ảnh để tạo thành các Embedded Patches. Theo như bài báo [5], họ chỉ sử dụng vị trí 1 chiều thay vì 2 chiều vì không có sự cải thiện hiệu năng khi họ thực nghiệm với thông tin vị trí 2 chiều. Khối Encoder vẫn được giữ nguyên như trong Transformer, các Embedded Patch sẽ đi qua các lớp chuẩn hóa, một lớp Multi-head attention và lớp MLP. Sau cùng, đầu ra từ khối Encoder sẽ đi qua một lớp MLP Head có số chiều là  $K$  bằng với số lớp trong tập dữ liệu.

Mô hình ViT có nhiều phiên bản, bằng cách thay đổi số lớp, số head trong Multi-head attention, số chiều ẩn  $D$ , số chiều của lớp MLP, ViT có 3 phiên bản

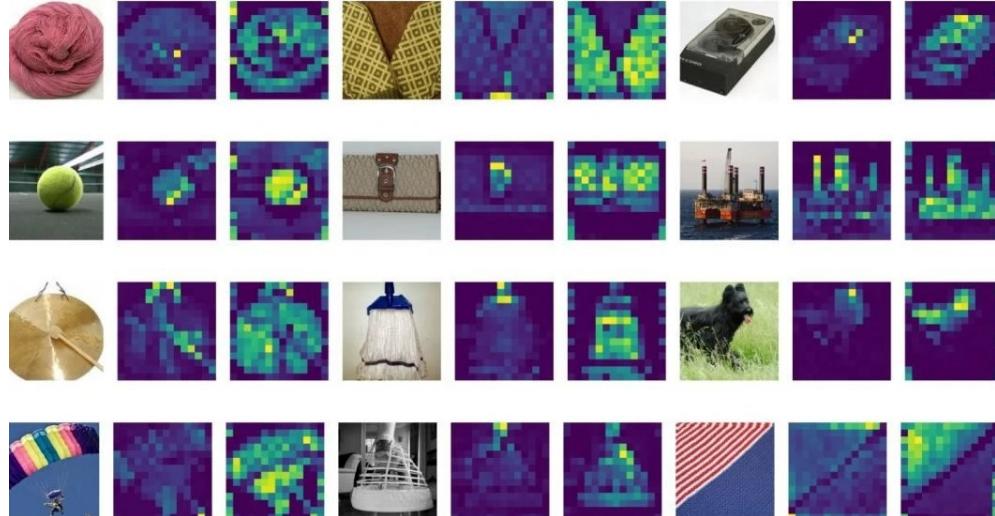
## 2. Cơ sở lý thuyết và các nghiên cứu liên quan

Mô hình	Số lớp	Số chiều ẩn D	Số chiều của MLP	Số Head	Số trọng số
ViT-Base	12	768	3072	12	86 Triệu
ViT-Large	24	1024	4096	16	307 Triệu
ViT-Huge	32	1280	5120	16	632 Triệu

Bảng 2.1: Các phiên bản của mô hình ViT

như trong bản sau 2.1

Để cho có tương quan với mô hình Transformer trong xử lý ngôn ngữ tự nhiên, mô hình ViT cũng sẽ dựa vào từng mảng ảnh và tập trung chú ý vào mảng ảnh có liên quan nhất đến nó. Hình 2.19 bên dưới là hình ảnh trực quan hóa các lớp của mô hình khi nó đã được huấn luyện. Có thể thấy rằng là sau khi được huấn luyện, bản đồ attention gần như đã chú ý được hoàn toàn vào đối tượng (các vùng sáng màu) trong ảnh, do đó có thể bỏ qua được các thành phần gây nhiễu như nền của bức ảnh, từ đó nâng cao hiệu suất cho mô hình.



Hình 2.19: Ảnh gốc (bên trái) và bản đồ Attention tương ứng với ảnh đầu vào của mô hình ViT đã được huấn luyện (bên phải) (Nguồn: Internet <sup>3</sup>)

Hơn nữa, mô hình chủ yếu dựa vào khối Transformer Encoder, do đó chi phí tính toán thấp hơn nhiều so với CNNs và cho độ chính xác cao hơn khi được

<sup>3</sup><https://vinbigdata.com/camera-ai/tong-quan-ve-vision-transformer-vit.html>

## **2. Cơ sở lý thuyết và các nghiên cứu liên quan**

huấn luyện trên tập dữ liệu lớn. Trong bài báo gốc, nhóm tác giả huấn luyện mô hình ViT trên các tập dữ liệu như ImageNet, CIFAR-10 và VTAB, sau đó mô hình sẽ được tinh chỉnh cho các tác vụ cụ thể hơn.

## **Chương 3**

# **MỘT SỐ PHƯƠNG PHÁP NHẬN DẠNG VĂN BẢN ÁP DỤNG CHO CHỮ NGHỆ THUẬT TIẾNG VIỆT**

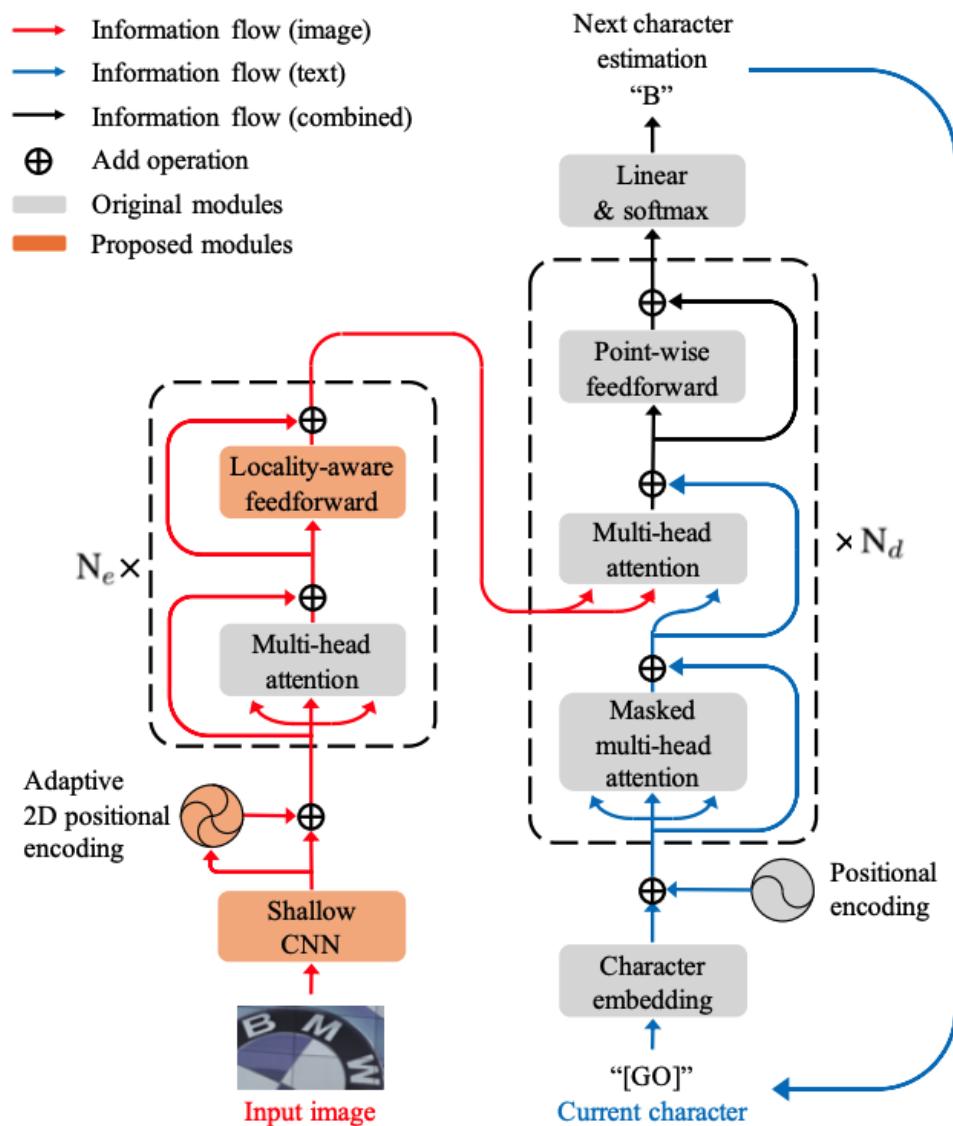
Trong chương này, sinh viên sẽ trình bày các phương pháp sẽ sử dụng để thực nghiệm trên tập dữ liệu đã xây dựng, mỗi mô hình sẽ được trình bày về kiến trúc mô hình và cách hoạt động của mô hình. Các mô hình thực nghiệm bao gồm ViTSTR, SATRN, Corner Transformer, ABINet và PARSeq.

### **3.1 Mô hình SATRN**

Mô hình SATRN (Self-Attention Text Recognition Network) được giới thiệu trong bài báo [6], mô hình lấy cảm hứng từ Transformer, SATRN muốn hướng đến việc có thể nhận dạng được văn bản ở hình dạng bất kỳ thay vì chỉ giả định là văn bản chỉ ở dạng thẳng như các nghiên cứu trước đó.

### 3.1.1 Kiến trúc mô hình

Để có thể nhận dạng được văn bản ở hình dạng bất kỳ mà không cần phải giả định là văn bản thẳng, SATRN tận dụng kiến trúc self-attention của mô hình Transformer, và để có thể biểu diễn thông tin 2 chiều, nhóm tác giả đã thực hiện một số thay đổi đối với khối Encoder.



Hình 3.1: Kiến trúc mô hình SATRN (Nguồn: Bài báo [6])

---

### 3.1.1.1 Khối Encoder

Khối encoder (khối bên phải hình 3.1) hoạt động tương tự như khối Encoder của Transformer. Tuy nhiên, có một lưu ý đó chính là phần đặc trưng được dùng làm **queries, keys, vectors** sẽ một số thay đổi. Hình ảnh trước khi đi vào khối encoder sẽ đi qua một mạng CNN nồng để trích xuất đặc trưng cục bộ và kết cấu của bức ảnh, việc này giúp bỏ qua được nhiều ở ảnh gốc, giảm chi phí tính toán cho khối Encoder. Trong bài báo, mạng CNN nồng này bao gồm 2 lớp tích chập với kernel 3x3, theo sau là một lớp pooling 2x2 với stride là 2. Với mạng CNN nồng này đầu ra sẽ có tỉ lệ 1/4 so với ảnh gốc, tỉ lệ này sẽ giúp cân bằng giữa chi phí tính toán và độ hiệu năng của mô hình.

Sau khi hình ảnh được trích xuất đặc trưng, vector đặc trưng sẽ được đi qua một lớp Adaptive 2D positional encoding để bổ sung thông tin về vị trí. Lớp này tương ứng với lớp positional encoding trong mô hình Transformer, tuy nhiên đã được tùy chỉnh để tương thích với sự đa dạng về kích thước của ảnh đầu vào, đặc biệt đối với nhận dạng văn bản với hình dạng bất kỳ, thông tin vị trí đóng một vai trò cực kỳ quan trọng. Lớp Adaptive 2D positional encoding (A2DPE) sẽ tự quyết định tỷ lệ giữa chiều cao và chiều rộng dựa trên kích thước có ảnh đầu vào. Ta có đầu ra của mạng CNN nồng là  $\mathbf{E}$  là đặc trưng 2 chiều,  $\mathbf{e}_{hw}$  là vùng ảnh ở vị trí  $(h, w) \in [1, \dots, H] \times [1, \dots, W]$ . Đối với cơ chế Self-attention áp dụng trên đặc trưng 2 chiều thông thường, ta sẽ có công thức sau:

$$\mathbf{att-out}_{hw} = \sum_{h'w'} \text{softmax}(rel_{(h'w') \rightarrow (hw)}) \mathbf{v}_{h'w'} \quad (3.1)$$

Trong đó  $v_{hw} = \mathbf{e}_{hw} \mathbf{W}^v$  là phép biến đổi tuyến tính đầu vào thông qua ma trận trọng số  $\mathbf{W}^v$ .  $rel_{(h'w') \rightarrow (hw)}$  được tính theo công thức sau:

$$rel_{(h'w') \rightarrow (hw)} \propto \mathbf{e}_{hw} \mathbf{W}^q \mathbf{W}^{k^T} \mathbf{e}_{h'w'}^T \quad (3.2)$$

---

trong đó  $\mathbf{W}^q$  và  $\mathbf{W}^k$  là ma trận trọng số biến đổi đầu vào tương ứng sang query  $\mathbf{q}_{hw} = \mathbf{e}_{hw}\mathbf{W}_q$  và key  $\mathbf{k}_{hw} = \mathbf{e}_{hw}\mathbf{W}_k$ . Có thể hiểu hàm  $\text{rel}_{(h'w') \rightarrow (hw)}$  thể hiện đặc trưng ở  $(h'w')$  chú ý bao nhiêu đến đặc trưng ở  $(hw)$ . Đối với A2DPE, công thức của hàm  $\text{rel}_{(h'w') \rightarrow (hw)}$  như sau:

$$\text{rel}_{(h'w') \rightarrow (hw)} \propto (\mathbf{e}_{hw} + \mathbf{p}_{hw})\mathbf{W}^q\mathbf{W}^{kT}(\mathbf{e}_{h'w'} + \mathbf{p}_{h'w'})^T \quad (3.3)$$

Có thể thấy từ công thức là A2DPE có thêm thành phần  $\mathbf{p}_{hw}$  và  $\mathbf{p}_{h'w'}$  lần lượt cộng vào đặc trưng từ mạng CNN nồng  $\mathbf{e}_{hw}$  và  $\mathbf{e}_{h'w'}$ . Trong đó thì  $\mathbf{p}_{hw}$  được tính theo công thức:

$$\mathbf{p}_{hw} = \alpha(\mathbf{E})\mathbf{p}_h^{sinu} + \beta(\mathbf{E})\mathbf{p}_w^{sinu} \quad (3.4)$$

Trong đó  $\mathbf{p}_h^{sinu}$  và  $\mathbf{p}_w^{sinu}$  là positional encoding theo tọa độ hình sin trên chiều cao và chiều rộng, theo như trong bài báo gốc của Transformer, công thức để tính như sau:

$$\mathbf{p}_{p,2i}^{sinu} = \sin(p/10000^{2i/D}) \quad (3.5)$$

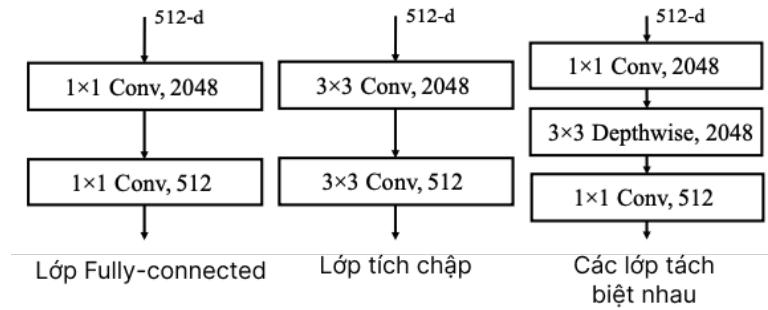
$$\mathbf{p}_{p,2i+1}^{sinu} = \cos(p/10000^{2i/D}) \quad (3.6)$$

Hệ số scaling  $\alpha(\mathbf{E})$  và  $\beta(\mathbf{E})$  được tính từ đặc trưng  $\mathbf{E}$  với 2 lớp mạng học sâu, sau đó đi qua một lớp global average pooling. Công thức tính như sau:

$$\alpha(\mathbf{E}) = \text{sigmoid}(\max(0, \mathbf{g}(\mathbf{E})\mathbf{W}_1^h)\mathbf{W}_2^h) \quad (3.7)$$

$$\beta(\mathbf{E}) = \text{sigmoid}(\max(0, \mathbf{g}(\mathbf{E})\mathbf{W}_1^w)\mathbf{W}_2^w) \quad (3.8)$$

trong đó  $\mathbf{W}_1^h, \mathbf{W}_2^h, \mathbf{W}_1^w, \mathbf{W}_2^w$  lần lượt là trọng số tuyến tính có thể học được.  $\mathbf{g}(\mathbf{E})$  là average pooling trên tất cả các đặc trưng trong  $\mathbf{E}$ . Hệ số  $\alpha$  và  $\beta$  sẽ ảnh



Hình 3.2: Sự thay đổi so với các lớp feedforward trong Encoder của Transformer  
(Nguồn: Bài báo [6])

hướng đến chiều cao và chiều rộng trong positional encoding. Bằng việc cho mô hình tự học các tham số  $\alpha$  và  $\beta$ , nó có thể tự tương thích với các kích cỡ ảnh khác nhau.

Sau cùng, thay đổi so với khôi Encoder của Transformer là lớp feedforward. Khi nhận dạng văn bản thì ngoài vấn đề phụ thuộc dài, các thành phần trong ảnh cũng nên chú ý đến những đặc trưng cục bộ xung quanh nó, do đó nhóm tác giả đề xuất sử dụng "Locality-aware feedforward layer". Thay vì sử dụng lớp feedforward point-wise, tác giả sử dụng các lớp tích chập 3x3 và lớp deep-wise 3.2.

### 3.1.1.2 Khối Decoder

Khối decoder được dùng để tạo ra chuỗi dự đoán. Từ đặc trưng 2 chiều rút trích được từ khôi encoder, khôi decoder tạo ra từ dự đoán bằng cách sử dụng từ hiện tại để làm query trên đặc trưng 2 chiều. Sự liên quan giữa ảnh và text xảy ra ở module multihead-attention. Hầu hết các module trong khôi decoder đều được giữ nguyên như trong mô hình Transformer.

---

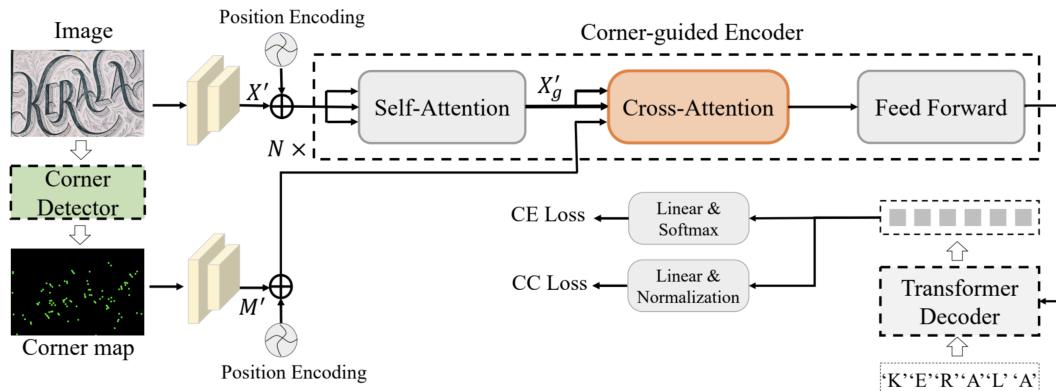
## 3.2 Mô hình Corner Transformer

Corner Transformer là phương pháp được giới thiệu trong bài báo [1], là một trong những bài báo đầu tiên đề cập đến vấn đề chữ nghệ thuật trong nhận dạng văn bản, đồng thời, bài báo cũng đề xuất mô hình Corner Transformer để nhận dạng chữ nghệ thuật được tốt hơn. Mô hình Corner Transformer được xây dựng để giải quyết 3 vấn đề trong nhận dạng văn bản bao gồm:

- Các đặc trưng cục bộ trong từng kí tự: các kí tự luôn xuất hiện dưới nhiều hình dạng khác nhau và vô cùng đa dạng, và có thể khác nhau giữa các kí tự trong cùng một từ, hơn nữa các kí tự đó còn có các nét kéo dài sang cả các kí tự khác. Do đó, Corner Transformer đề xuất sử dụng bản đồ corner point để xóa bỏ sự liên kết mạnh giữa các kí tự, sự chồng chéo lên nhau cũng như nhiều do nền của hình ảnh gây ra.
- Các đặc trưng ở mức kí tự: nhận dạng đúng từng kí tự là cực kỳ quan trọng trong nhận dạng văn bản. Với cùng một kí tự, tuy nhiên chúng có thể xuất hiện dưới muôn hình vạn trạng các hình dạng khác nhau, do đó đặc trưng của chúng trong không gian vector cũng phân tán rất nhiều. Do đó, Corner Transformer đề xuất một hàm mất mát dựa trên Contrastive learning nhằm kéo các đặc trưng của kí tự thuộc cùng một lớp lại gần nhau và các kí tự khác lớp ra xa hơn.
- Các đặc trưng ở mức hình ảnh: mục tiêu sau cùng của nhận dạng văn bản vẫn là nhận dạng được toàn bộ các kí tự trong ảnh, do đó, việc có thể tận dụng được thông tin ngữ nghĩa trong ảnh để hỗ trợ việc nhận dạng là rất cần thiết. Vì vậy, Corner Transform sử dụng backbone là mô hình Transformer để có thể tận dụng được tốt thông tin ngữ nghĩa trong ảnh.

### 3.2.1 Kiến trúc mô hình

Về tổng quan, kiến trúc mô hình CornerTransformer được biểu diễn như hình 3.3. Với một ảnh đầu vào  $X \in R^{H \times W \times 3}$ , Corner Transformer sử dụng thuật toán Shi-Tomasi để tạo ra bản đồ góc (corner map)  $M \in R^{H \times W \times 1}$ .  $X$  và  $M$  sau đó sẽ đi qua hai lớp tích chập để tạo thành đặc trưng  $X' \in R^{\frac{H}{4} \times \frac{W}{4} \times C}$  và  $X' \in R^{\frac{H}{4} \times \frac{W}{4} \times C}$  tương ứng, trong đó  $C$  là số chiều của đặc trưng từ lớp tích chập, các đặc trưng này cũng sẽ được tích hợp thông tin vị trí thông qua Position Encoding trước khi đi vào khối Encoder tương tự như trong ViT. Mặt khác, đặc trưng toàn cục  $X'_g$  sẽ được học từ  $X'$  thông qua cơ chế Multi-head attention của Transformer, sau đó,  $X'_g$  sẽ được kết hợp với  $M'$  thông qua cơ chế Multi-head cross-attention. Sau cùng, đầu ra của khối Encoder và embedded vector của chuỗi kí tự sẽ được đi qua khối Transformer Decoder để dự đoán chuỗi đầu ra.



Hình 3.3: Kiến trúc mô hình Corner Transformer (Nguồn: Bài báo [1])

### 3.2.2 Khối Corner-Guided Encoder

Về việc sử dụng bản đồ góc 3.4 từ thuật toán phát hiện các góc (corner detection) Shi-tomashi [26], đặc trưng này được xem là thông tin bổ sung cho

---

mô hình, nhằm tạo ra các đặc trưng bất biến đối với ảnh đầu vào. Với bản đồ góc, các liên kết phức tạp giữa các kí tự, các hiệu chồng chéo, hiệu ứng viền. Bản đồ góc chỉ chứa lại các điểm quan trọng (key point), do đó nó có thể khử các hiệu ứng gây nhiễu ở nền ảnh 3.5. Về lý do lựa chọn thuật toán Shi-tomashi thay vì các thuật toán khác dựa trên học sâu như SuperPoint [27], thuật toán Shi-tomashi là phiên bản cải tiến của thuật toán Harris corner detection [28], nó đem lại sự ổn định cao hơn thuật toán Harris.



Hình 3.4: Hình ảnh trực quan hóa kết quả của bản đồ key point (Nguồn: Bài báo [1])

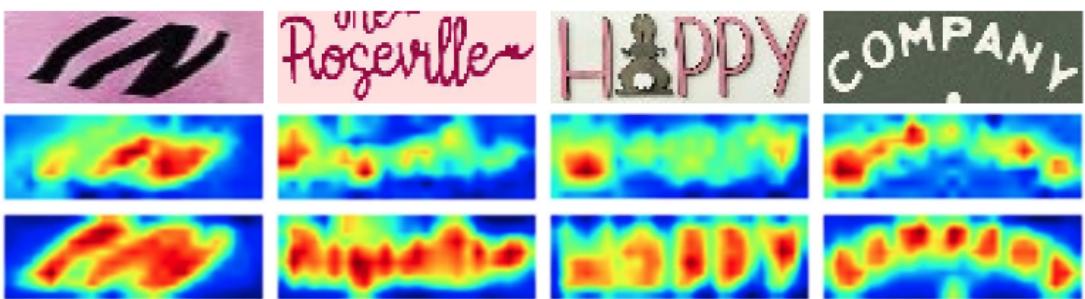
Có thể thấy trong hình 3.4, các đặc trưng vẫn còn rời rạc nhau, do đó, khôi Corner-Guided Encoder được thiết kế để kết hợp thông tin giữa đặc trưng ảnh và bản đồ góc với nhau ở mỗi khôi, điều này được thực hiện thông qua cơ chế Multi-head cross-attention. Tương tự như SATRN[6], đặc trưng (CNN)  $X'_g$  vẫn được sử dụng làm **keys - K** và **values - V**; tuy nhiên, **queries - Q** sử dụng trong cơ chế attention sẽ sử dụng đặc trưng bản đồ góc  $M'$  - xem cách thay thế biến trong công thức 3.9. Cũng vì việc sử dụng bản đồ góc làm **Q**, cho nên tác giả đặt tên cho cơ chế attention trong đây là corner-query cross-attention. Công thức của corner-query attention có dạng sau:

$$\text{CA}(Q, K, V) = \text{CA}(M', X'_g, X'_g) = \text{softmax}\left(\frac{M'X'^T_g}{\sigma}\right)X'_g \quad (3.9)$$

Trong đó, cơ chế attention được đặt tên là **CA** - viết tắt của Cross-Attention

---

với  $\sigma$  là hệ số scaling (giống với  $\sqrt{d_k}$  trong công thức gốc của [4] - 2.8). Nhờ vào việc sử dụng corner-query, mô hình có thể tập trung chú ý tốt hơn vào kí tự chính xác hơn. Trong mô hình Corner Transformer sẽ có  $N$  khối Corner-Guided Encoder được xếp chồng lên nhau tương tự như trong mô hình Transformer.



Hình 3.5: Hình ảnh bản đồ nhiệt của attention không có và có sử dụng corner-query (Nguồn: Bài báo [1])

### 3.2.3 Hàm mất mát Character contrastive loss

Các đặc trưng dựa trên góc (corner-based) chỉ tập trung vào các đặc trưng cục bộ của kí tự, trong khi Transformer sẽ học các đặc trưng toàn cục của bức ảnh, nhóm nghiên cứu còn đề xuất thêm một phương pháp học nhằm giúp mô hình có hiệu năng tốt hơn ở mức kí tự. Ý tưởng là trong quá trình học, mô hình sẽ cố gắng đẩy các vector khác lớp ra xa nhau và gom cụm các vector có cùng lớp lại trong không gian vector 3.6. Để làm được điều này thì nhóm tác giả đã đề xuất một hàm mất mát là Character contrastive loss (viết tắt là CC loss). Trong cùng một minibatch với số ảnh là  $N$ , giới hạn số lượng kí tự dài nhất là  $m = 25$ , tổng số kí tự sẽ là  $Nxm$  trong một minibatch. Với kí tự thứ  $i$ , ta sẽ có vector đặc trưng  $x_i$  và nhãn tương ứng  $y_i$ , với  $i \in I = \{1, 2, \dots, N\}$ . Vậy, khi kí tự thứ  $i$  là điểm neo - anchor, ta sẽ có tập các điểm tích cực (positive set) (cùng lớp với lớp của kí tự  $i$ )  $P(i) \equiv \{p \in I : y_p \equiv y_i; p \neq i\}$ ; và ngược lại, ta có tập tiêu cực

---

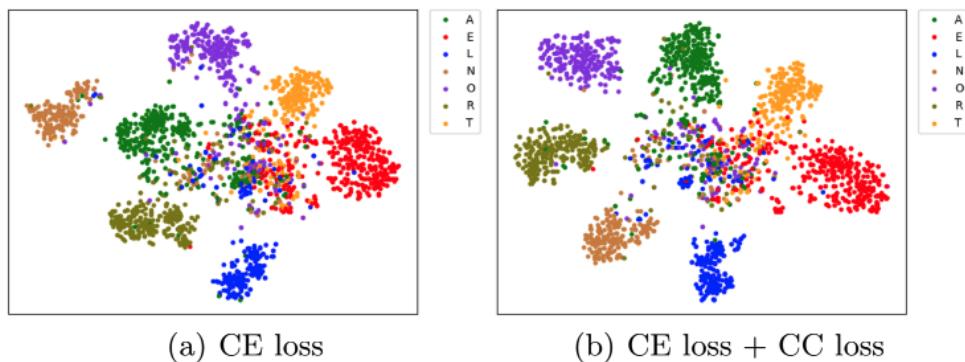
(negative set)  $N(i) \equiv \{p \in I : y_n \neq y_i; n \neq i\}$ . Như vậy ta sẽ được công thức tính độ lỗi *CC loss* như sau:

(3.10)

Trong đó  $N_p$  là số lượng kí tự tiêu cực (tức là khác lớp với kí tự đang là điểm neo) và  $\tau$  là hệ số scaling. Hàm mất mát sau cùng được sử dụng là tổng có hệ số giữa hàm Cross-Entropy loss và CC loss:

$$L = L_{CE} + \lambda L_{CC} \quad (3.11)$$

Trong đó  $\lambda$  được cài đặt mặc định bằng 0.1.

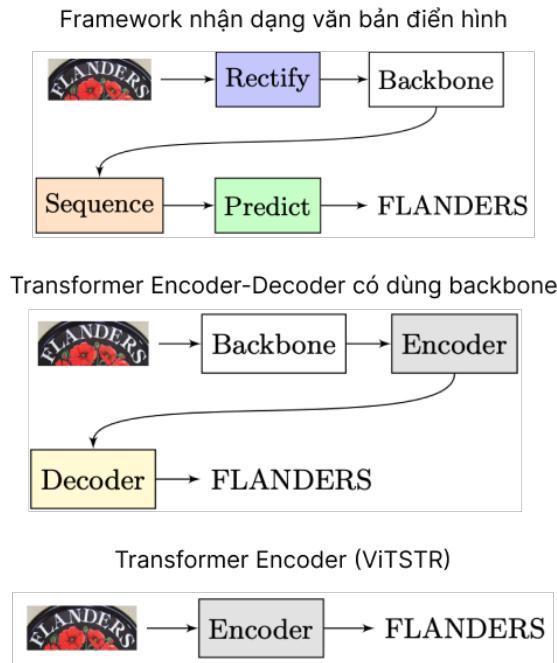


Hình 3.6: Trực quan hóa các điểm dữ liệu trong không gian khi huấn luyện mô hình chỉ dùng CE loss và dùng CE loss kết hợp với CC loss (Nguồn: Bài báo [1])

### 3.3 Mô hình ViTSTR

Mô hình ViTSTR được giới thiệu trong bài báo [7] là mô hình nhận dạng văn bản cân bằng về hiệu năng và chi phí tính toán, mô hình cho độ chính xác cao trong khi chi phí tính toán thấp hơn so với các phương pháp trước đó như TRBA (TPS-ResNet-BiLSTM-Attention) [16]. Mô hình ViTSTR tận dụng ưu điểm của

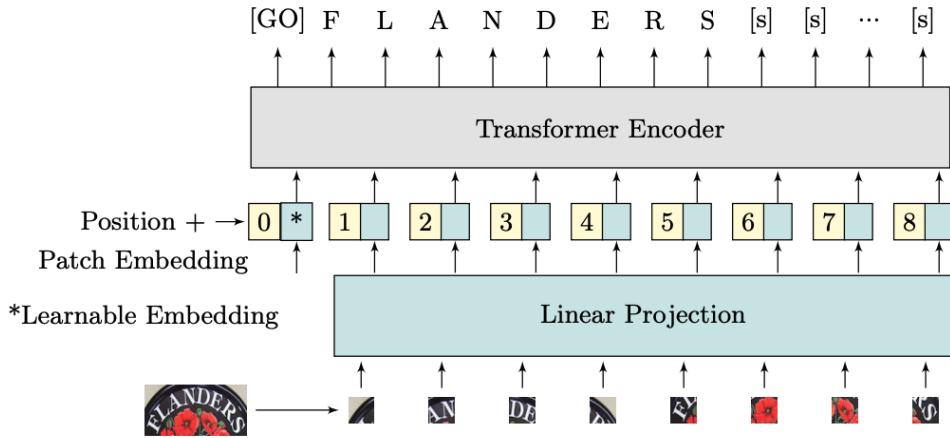
mô hình Vision Transformer, đơn giản nhưng hiệu quả khi nó chỉ sử dụng mỗi khôi Encoder của Transformer [3.7](#).



Hình 3.7: Kiến trúc của ViTSTR so với các kiến trúc STR khác (Nguồn: Bài báo [\[7\]](#))

Do tận dụng tính đơn giản và hiệu quả của mô hình ViT, sự khác duy nhất giữa ViT và ViTSTR là ở đầu dự đoán (prediction head). ViT được huấn luyện để dự đoán nhãn của một đối tượng, trong khi đó, ViTSTR phải dự đoán nhiều kí tự cùng lúc và phải giữ đúng thứ tự trong khi dự đoán. Do mô hình ViTSTR giống với ViT, khôi Encoder cũng được giữ nguyên như trong mô hình Transformer, khôi Encoder bao gồm: lớp chuẩn hóa, lớp Multi-head Self-attention, các residual connection và lớp MLP, và sẽ xếp chồng  $L$  khôi lên nhau (bên phải [2.18](#)).

Trong mô hình ViT, đầu ra dự đoán sẽ được dùng để phân loại đối tượng trong ảnh, trong ViTSTR, đầu ra này sẽ tương ứng với token đặc biệt [GO], dùng để thể hiện sự bắt đầu của câu dự đoán. Để dự đoán được nhiều kí tự song song với nhau, thay vì chỉ trích xuất một vector đặc trưng từ khôi Encoder, ViTSTR



Hình 3.8: Kiến trúc của mô hình ViTSTR (Nguồn: Bài báo [7])

Phiên bản ViTSTR	Kích cỡ mảng ảnh P	Chiều sâu L	Số chiều vector ẩn D	Số Head	Chiều dài câu
Tiny	16	12	192	3	27
Small	16	12	384	6	27
Base	16	12	768	12	27

Bảng 3.1: Các phiên bản của mô hình ViTSTR

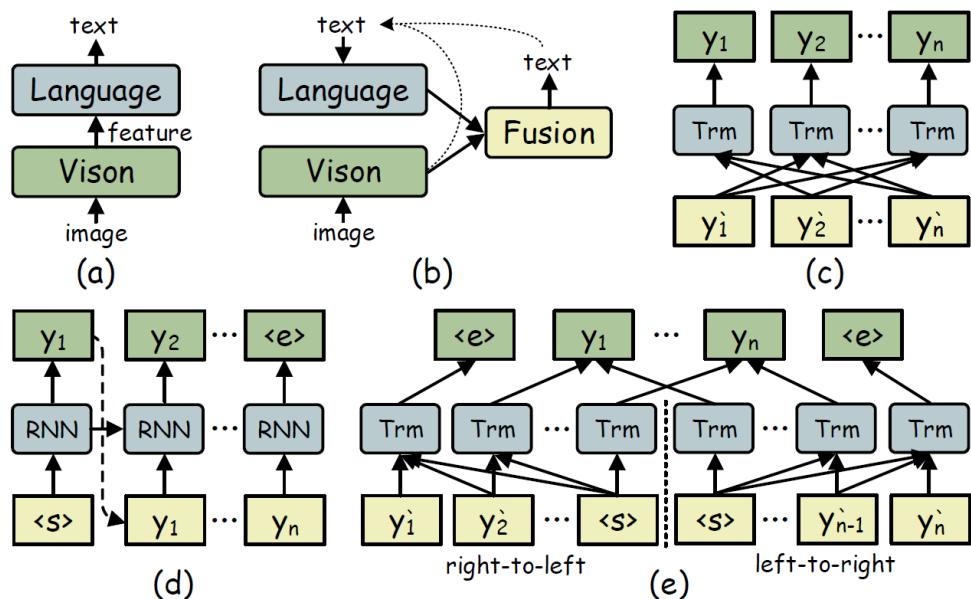
trích xuất nhiều vector đặc trưng cùng lúc, số lượng vector đặc trưng sẽ bằng với chiều dài tối đa của câu sinh ra được quy định từ trước. Do chiều dài được quy định từ trước, nếu nội dung thực tế ít hơn chiều dài đó, token đặc biệt [s] sẽ được sử dụng để biểu thị kết thúc câu hoặc khoảng trắng.

Cũng tương tự như ViTSTR, bằng việc thay đổi số khối encoder  $L$ , số chiều của vector ẩn  $D$  và số head  $H$  ở lớp Multi-head self-attention, ViTSTR tạo ra nhiều phiên bản khác nhau 3.1.

### 3.4 Mô hình ABINet

Trong những năm gần đây, lĩnh vực xử lý ngôn ngữ tự nhiên đã đạt được nhiều kết quả đáng chú ý nhờ vào sự phát triển của các mô hình như Transformer,

BERT [29], các mô hình đó ngày càng "hiểu" rõ được thông tin kiến thức về ngôn ngữ, nếu chúng ta có thể tận dụng được kiến thức này để dự đoán văn bản trong ảnh được tốt hơn thì sao? Đó là lý do mô hình ABINet (tên đầy đủ là Read Like Humans: Autonomous, Bidirectional and Iterative Language Modeling for Scene Text Recognition)[8] được đề xuất, mô hình này sử dụng thông tin thị giác và cả thông tin ngôn ngữ học được để có thể đưa ra dự đoán chính xác hơn thay vì chỉ sử dụng mỗi thông tin thị giác như các phương pháp nhận dạng văn bản khác.



Hình 3.9: Hình cắt ra từ bài báo ABINet. a) Mô tả mô hình ngôn ngữ phụ thuộc vào kết quả đầu ra mô hình thị giác. b) Cải thiện của tác giả, không cho mô hình ngôn ngữ lệ thuộc vào mô hình thị giác nữa. Về ngoài ra, quá trình dự đoán có sự kết hợp của cả hai. c) Cải thiện tiếp theo của tác giả: Áp dụng đặc trưng hai chiều *bidirectional* - đặt tên là **BCN** - trong mô hình ngôn ngữ - dựa trên ý tưởng của BERT. d) Mô hình *tự hồi qui* (autoregressive) - dự đoán tuần tự (nghĩa là kết quả đầu ra  $y_t$  có phụ thuộc vào  $y_{t-1}$ ). e) Áp dụng **transformer** theo cả 2 hướng *unidirectional* - để dự đoán cùng một lúc tất cả kí tự. (Nguồn: Bài báo [8])

---

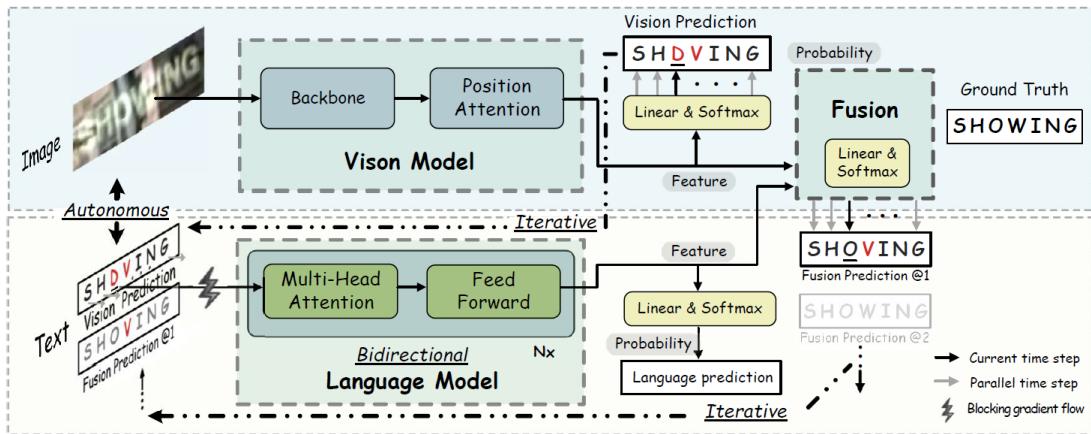
### 3.4.1 Cách nhìn tổng quan về mô hình ABINet

Như đã được mô tả ở phần phía trên, ABINet đặc biệt quan tâm đến đặc trưng ngôn ngữ để cải thiện kết quả cuối cùng của thuật toán. Tác giả của ABINet đã chỉ ra các hạn chế của các mô hình đi trước như sau:

1. Các mô hình đi trước gộp chung mô hình thị giác với mô hình ngôn ngữ, điều này làm ảnh hưởng đáng kể đến kết quả cuối cùng. Theo như tác giả, hai mô hình, thị giác và ngôn ngữ cần được hoạt động độc lập với nhau (với từng mô hình được huấn luyện rồi sau đó sẽ dự đoán mà không lệ thuộc vào nhau). Nhân thể, đây cũng là chữ đầu của mô hình **Autonomous** - độc lập.
2. Tiếp theo, một điều phải công nhận là thông tin theo hai hướng **Bidirectional** sẽ mang nhiều ngữ nghĩa hơn thông tin đến chỉ từ một hướng (Unidirectional). Đã có rất nhiều công trình chứng minh điều này, nổi tiếng nhất đó chính là BERT[29].
3. Đa số các thuật toán tính tới thời điểm ra đời của ABINet thường cho kết quả dự đoán chỉ một lần duy nhất. Tuy nhiên, việc làm này không tận dụng hết được đặc trưng thị giác. Và ngoài ra hạn chế của mô hình autoregressive d) hình [24] (ví dụ như thuật toán 3.9) là lỗi (nếu có) sẽ tích tụ, làm ảnh hưởng đến bước dự đoán tiếp theo. Việc tinh chỉnh liên tục **Iterative** sẽ giúp tận dụng và chắc lọc lại thông tin ngữ nghĩa (Giai đoạn Fusion và dự đoán của hình 3.10).

Nhằm khắc phục những hạn chế nêu trên, tác giả của ABINet đã đề xuất việc như sau:

1. **Tách** mô hình ngôn ngữ khỏi mô hình thị giác - điều này là đương nhiên là để quá trình tinh chỉnh chính tả không ảnh hưởng đến mô hình VM.



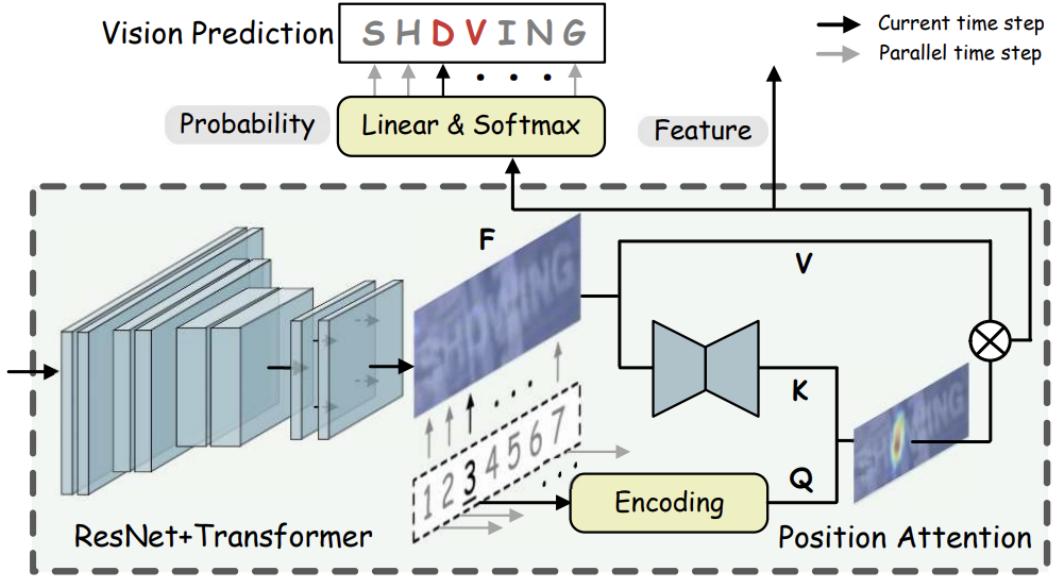
Hình 3.10: Hình tổng quan thuật toán ABINet (Nguồn: Bài báo [8])

2. Thiết kế lại mô hình ngôn ngữ - đặt tên là BCN - bidirectional cloze network (gần giống các hoạt động của BERT (dự đoán từ) - nhưng giờ áp dụng cho việc dự đoán kí tự).
3. Nhóm tác giả còn đề xuất sử dụng cơ chế **Iterative Correction** để có thể sửa lỗi dự đoán một cách chính xác nhất, tận dụng được tối đa thông tin từ mô đun VM và LM. Ngoài ra, để chứng minh tính hiệu quả của cơ chế sửa lỗi này, nhóm tác giả cũng đề xuất phương thức học bán giám sát (self-supervised learning) cho VM trên một khối lượng lớn dữ liệu text.

### 3.4.1.1 Vision Model

Trong phần mô hình thị giác này (Vision Model viết tắt là VM), backbone và một mô đun Positional Attention. Backbone để rút trích đặc trưng bao gồm một mạng ResNet [18] và các đơn vị Transformer, chúng đóng vai trò như là mô hình trích xuất đặc trưng (feature extraction) và mô hình hóa chuỗi (sequence modeling).

Phần VM được lấy cảm hứng từ các phương pháp đi trước [30], đặc trưng thị giác được rút trích thông qua mạng học sâu ResNet [18] và có sử dụng FPN



Hình 3.11: Tổng quan phần Vision Model (VM) của ABINet (Nguồn: Bài báo [8])

[31] để cộng hợp các đặc trưng ở các tầng khác nhau, mà cụ thể là cộng hợp đặc trưng tại mỗi lớp cuối của tầng 3, 4 và 5; phần transformer bao gồm positional encoding, và khối encoder (với từng layer trong mỗi khối gồm đầy đủ các module multi-head attention (8 head) và mạng feed-forward) - tham khảo công thức 3.5, 2.8 có nhiệm vụ . Kết quả là ta thu được đặc trưng 2 chiều  $F$  với chiều cao  $H$  và rộng chỉ  $W$  bằng  $1/4$  kích thước ảnh ban đầu - xem công thức 3.12. Trong đó  $T$  biểu diễn cho phần Transformer,  $R$  cho phần backbone Resnet,  $x$  là ảnh đầu vào:

$$F_b = T(R(x)) \in \mathbb{R}^{\frac{H}{4} \times \frac{W}{4} \times C} \quad (3.12)$$

Tiếp theo, đặc trưng  $F_b$  được biến đổi thành **keys** -  $\mathbf{K}$  và **values** -  $\mathbf{V}$  cho cơ chế attention theo công thức dưới đây, với  $T$  lần lượt là độ dài của transcript,  $\mathcal{H}$  là ma trận identity và  $\mathcal{G}$  phiên bản mini của mạng U-Net [32]:

---

1.

$$\mathbf{V} = \mathcal{H}(F_b) \in \mathbb{R}^{\frac{HW}{16} \times C} \quad (3.13)$$

2.

$$\mathbf{K} = \mathcal{G}(F_b) \in \mathbb{R}^{\frac{HW}{16} \times C} \quad (3.14)$$

Còn đối với *queries* -  $\mathbf{Q}$ , ta sử dụng lại từ 3.5.

Sau bước này ta được:

$$F_v = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{C}}\right)\mathbf{V} \in \mathbb{R}^{T \times C} \quad (3.15)$$

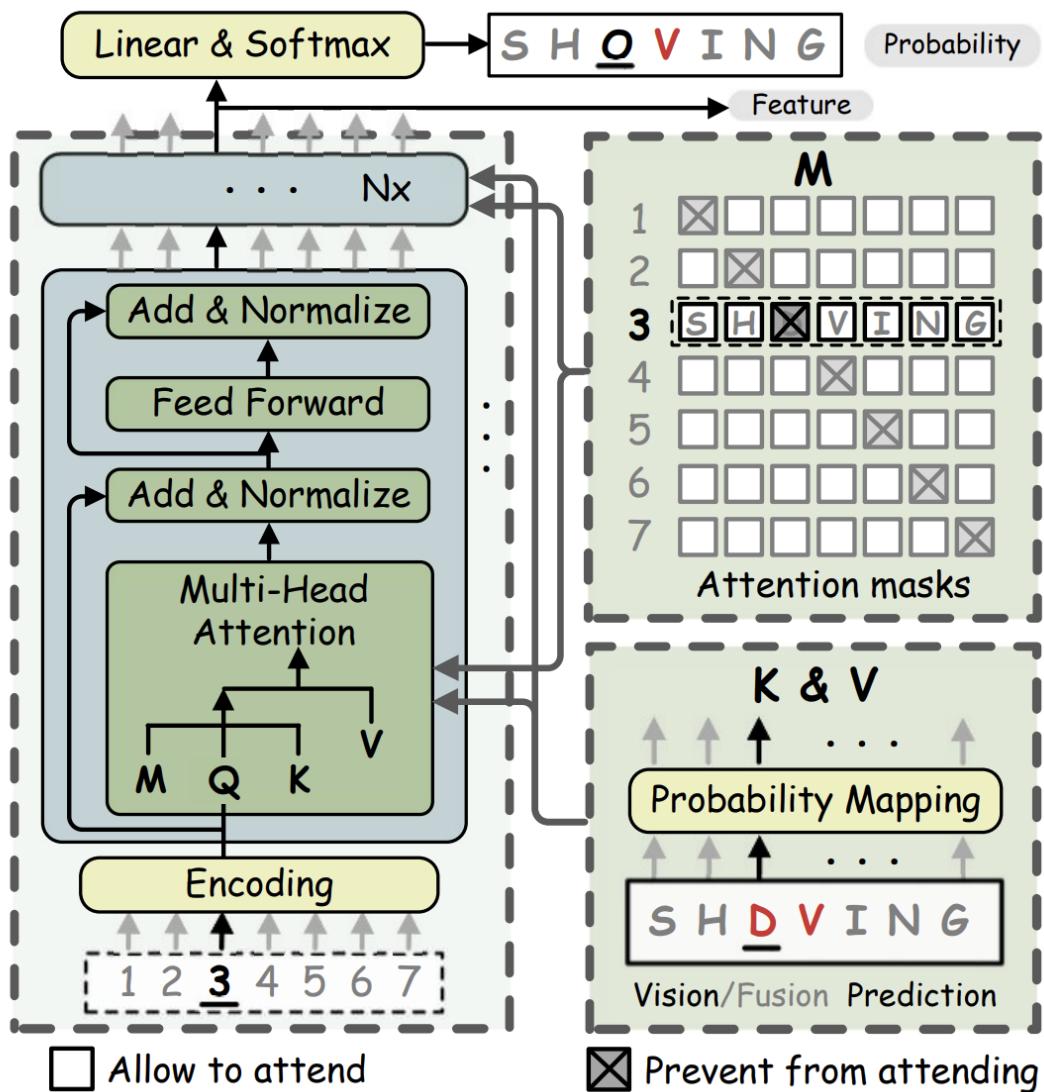
**Lưu ý:** 2.8 là self-attention (vì  $\mathbf{Q}$  bằng  $\mathbf{K}$ ) còn 3.15 thì không phải ( $\mathbf{Q}$  khác  $\mathbf{K}$ ).

#### 3.4.1.2 Language Model

Như đã giới thiệu ở phần mở đầu, cơ chế thực hiện của phần LM dựa trên 3 đặc điểm sau:

1. Độc lập với Vision model (VM) - **Autonomous**;
2. Tận dụng đặc trưng từ cả 2 phía - **Bidirectional**;
3. Tinh chỉnh nhiều lần kết quả dự đoán - **Iterative**.

**Autonomous:** (1) Do kết quả dự đoán (*Fusion Prediction*) là sự kết hợp giữa VM và LM, và điều ta mong muốn là tránh sự ảnh hưởng của gradient VM đến LM, và ngược lại. Chính vì nguyên nhân này mà ta chặn không cho gradient của *Fusion Prediction* back-prop - xem hình 3.10. (2) Không giống như BERT[29] - dự đoán **từ** tại vị trí được che, VM dự đoán **kí tự** tại vị trí được che *mask* - xem khôi Attention masks của hình 3.12 - cho ra vector phân phối xác suất của kí tự dự đoán. (3) Chính vì nguyên nhân (1), ta có thể huấn luyện phần VM một cách



Hình 3.12: Tổng quan phần Language Model (LM) của ABINet - Kiến trúc BCN (Nguồn: Bài báo [8])

độc lập, tác giả của ABINet có huấn luyện self-supervised VM trên dữ liệu text từ tập MJ[33], ST[34] và WiKiText-103[35].

**Bidirectional:** Nhắc lại về công thức biểu diễn đầu ra của từng vị trí sẽ là 1

---

vector xác suất, với

$$P(y_i | y_n, \dots, y_{\mathbf{i}+1}, y_{\mathbf{i}-1}, \dots, y_2, y_1) \quad (3.16)$$

và

$$P(y_t | y_{t-1}, y_{t-2}, \dots, y_1) \quad (3.17)$$

lần lượt biểu diễn cho kết quả dự đoán khi sử dụng đặc trưng hai chiều - *bidirectional* và một chiều - *unidirectional*, với **c** là số "lớp", **i** là vị trí và **n** là độ dài chuỗi  $\mathbf{y} = (y_1, y_2, \dots, y_n)$  cho trước. Tuy nhiên, cả hai cách này đều bị hạn chế, *unidirectional* - đương nhiên là do hạn chế do chỉ sử dụng thông tin từ một hướng; và đối với các kỹ thuật đi trước, *bidirectional* - cũng chỉ là kết hợp giữa 2 mô hình *unidirectional* (xem hình 3.9 - e).

Như vậy, để khắc phục, tác giả lấy cảm hứng từ Mask Language Model của BERT [29] để tạo ra **bidirectional cloze network - BCN** - hình 3.12. BCN là một khối encoder bao gồm N lớp transformer, không những vậy mà BCN còn sử dụng attention-mask  $\mathbf{M} \in \mathbb{R}^{T \times C}$  để ngăn kí tự thứ  $i$  không tự thấy chính nó - xem công thức 3.18 (che theo đường chéo chính) và hình 3.12. Ngoài ra, điểm khác biệt cơ chế attention dùng trong các khối transformer này **không** phải là self-attention - nghĩa là **queries Q** sẽ không bằng **keys - K** trong công thức 3.20.

$$\mathbf{M}_{i,j} = \begin{cases} 0, & i \neq j, \\ -\infty, & i = j \end{cases} \quad (3.18)$$

**keys - K**  $\in \mathbb{R}^{T \times C}$  và **values - V**  $\in \mathbb{R}^{T \times C}$  cho cơ chế attention sẽ đạt được từ đầu ra của vector dự đoán từ lần lặp trước theo công thức:

$$\mathbf{V}_i = \mathbf{K}_i = P(y_i) \times \mathbf{W}_l \in \mathbb{R}^{1 \times C} \quad (3.19)$$

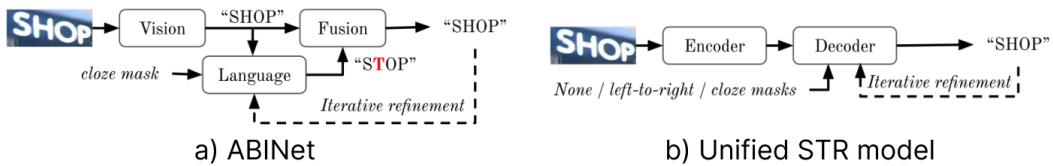
và cuối cùng là cơ chế attention với mask được sử dụng trong đó là:

$$F_{mha} = \text{softmax}\left(\frac{\mathbf{QK}^T}{\sqrt{C}} + \mathbf{M}\right)\mathbf{V} \in \mathbb{R}^{T \times C} \quad (3.20)$$

**Iterative:** Đây là quá trình tinh chỉnh kết quả, bao gồm bước dự đoán đầu tiên - (Vision Prediction) và n lần tinh chỉnh tiếp theo (Fusion Prediction) và theo kết quả thực nghiệm của tác giả thì con số tối ưu cho bước này là **3** lần tinh chỉnh - cho quá trình huấn luyện; và số lần tinh chỉnh khi chạy test là **3** vì sau đó độ chính xác đã được bão hòa (từ kết quả hình thực nghiệm Fig. 6 của [8]).

### 3.5 Mô hình PARSeq

Cũng như ABINet[1], PARSeq[9] sử dụng hướng tiếp cận tích hợp mô hình ngôn ngữ vào thuật toán của họ. Tuy nhiên, tác giả của PARSeq đã chỉ ra những hạn chế của mô hình ABINet[1]. Cụ thể, tác giả chỉ ra rằng chính vì cách hoạt động độc lập của mô hình LM (so với VM/đặc trưng thị giác) không những khiến LM có xu hướng dự đoán sai mà còn lãng phí chi phí tính toán.



Hình 3.13: Hình so sánh mô hình a) ABINet với b)PARSeq. Với cách hoạt động của ABINet, ta tách rời VM (không ngữ cảnh - **context-free**) với LM (tinh chỉnh - **iterative-refinement**), hay nói cách khác, ta sử dụng LM như một bộ từ điển - để chỉnh chính tả. Điều này đôi lúc dẫn tới LM sẽ dự đoán sai nếu như từ sai đó lại có xác suất cao hơn (do quá trình huấn luyện pretrainng LM). Ngược lại, mô hình LM mà nhóm của PARSeq đề xuất kết hợp **context-free** và **iterative-refinement**, một cách linh hoạt thông qua mô hình **Permutation Language Modeling** - gọi tắt là (PLM). (Nguồn: Bài báo [9])

---

### 3.5.1 Permutation Language Modeling (PLM) - thiết lập tổ hợp

Như vậy, PLM là gì, PLM là một cơ chế kết hợp ưu điểm của mô hình **autoregressive** và **non-autoregressive** (ví dụ như BERT[29]) đồng thời tránh được nhược điểm của cả hai.

Cụ thể hơn, PLM sẽ huấn luyện trên "mọi" tổ hợp của chuỗi  $\mathbf{y} = [y_1, y_2, \dots, y_T]$ . Như công thức 3.16, ta cần tối ưu mọi mô hình sau (với  $X$  là ảnh và  $Y$  là chuỗi cần dự đoán), như vậy log của xác suất đầu ra của  $Y$  sẽ xác suất kết hợp - *joint probability*:

$$\log p(Y|X) = \sum_{t=1}^T \log(p_\theta)(y_t|Y_{<t}, X) \quad (3.21)$$

hay nói cách khác, phuong trình trên khiến ta chỉ có thể dự đoán kí tự  $y$  tại thời điểm  $t$  i.e  $y_t$  dựa trên chiều của "quá khứ"  $y_{t-1}, \dots, y_2, y_1$  và không thể theo chiều ngược lại!. Như vậy PLM sẽ tối ưu hàm sau:

$$\log p(Y|X) = \mathbb{E}_{z \sim \mathbb{Z}_T} [\sum_{t=1}^T \log(p_\theta)(y_{z_t}|Z_{<t}, X)] \quad (3.22)$$

với  $\mathbb{Z}_T$  là mọi hoán vị ( $T!$ ) có thể xảy ra cho thứ tự - index của chuỗi  $Y$ , ví dụ  $[1,2,3]$  sẽ có hoán vị  $Z = [1,2,3]$  hoặc  $[1,3,2]$  hoặc  $[2,1,3]$  hoặc  $[2,3,1]$  hoặc  $[3,1,2]$  hoặc  $[3,2,1]$ ; và  $z_t$  với  $Z_{<t}$  kí tự thứ  $t$  trong chuỗi  $Z$  và  $t$  kí tự đầu tiên trong chuỗi  $Z$ . Biểu diễn một cách "python" ta sẽ có  $\mathbf{Y}[Z[t]]$  và  $\mathbf{Y}[Z[:t]]$ .

Lợi thế của hướng tiếp cận này sẽ giúp một kí tự có thể nhìn thấy tất cả kí tự còn lại (non-autoregressive). *Tuy nhiên*, ta dễ dàng nhận ra rằng việc sử dụng mọi tổ hợp sẽ sử dụng rất nhiều tài nguyên. Để khắc phục được điều này, thực tế thì tác giả của PARSeq không sử dụng tất cả mọi tổ hợp mà ra  $K$  tổ hợp từ  $T!$  theo tiêu chuẩn sau. Cụ thể, tác giả *luôn* sử dụng tổ hợp gốc - từ *trái sang phải*  $[1,2,3]$  với phiên bản đối lập của nó là *phải sang trái*  $[3,2,1]$  làm một cặp; và còn lại  $K - 2$  hoán vị sẽ được ta làm như sau: ta sẽ chọn ngẫu nhiên  $(K-2)/2 =$

---

K/2 -1 tổ hợp, và nửa còn lại K/2 -1 là các tổ hợp đối lập với K/2 -1 tổ hợp đó, ví dụ [1,3,2] vs [2,3,1], ... - xem [3.23](#):

$$\begin{aligned} [1,2,3] &\text{ vs } [3,2,1] \\ [1,3,2] &\text{ vs } [2,3,1] \\ [3,1,2] &\text{ vs } [2,1,3] \end{aligned} \tag{3.23}$$

Như vậy, tương tự như công thức [3.16](#), ta sẽ triển khai được các xác suất đồng thời sau:

$$\begin{aligned} p(\mathbf{y})_{[1,2,3]} &= p(y_1)p(y_2|y_1)p(y_3|y_1,y_2) \quad \text{vs} \quad p(\mathbf{y})_{[3,2,1]} = p(y_3)p(y_2|y_3)p(y_1|y_2,y_3) \\ p(\mathbf{y})_{[1,3,2]} &= p(y_1)p(y_3|y_1)p(y_2|y_1,y_3) \quad \text{vs} \quad p(\mathbf{y})_{[2,3,1]} = p(y_2)p(y_3|y_2)p(y_1|y_2,y_3) \\ p(\mathbf{y})_{[3,1,2]} &= p(y_3)p(y_1|y_3)p(y_2|y_1,y_3) \quad \text{vs} \quad p(\mathbf{y})_{[2,1,3]} = p(y_2)p(y_1|y_2)p(y_3|y_1,y_2) \end{aligned} \tag{3.24}$$

Trong phần thực nghiệm của nhóm tác giả, **K = 6** là con số K cân bằng cho độ hiệu quả và chi phí tính toán/thời gian.

### 3.5.2 Tổng quát mô hình

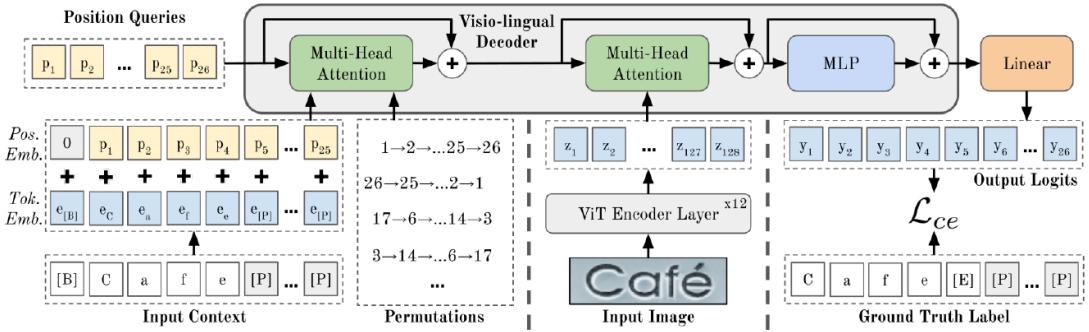
Sau khi đã thiết lập được cơ chế hoán vị, ta cần nói về cấu trúc tổng quan của PARSeq trước khi đi vào cách hoạt động encoding và decoding của PARSeq. Kiến trúc tổng quan của mô hình PARSeq như hình bên dưới [3.14](#).

#### 3.5.2.1 Encoder

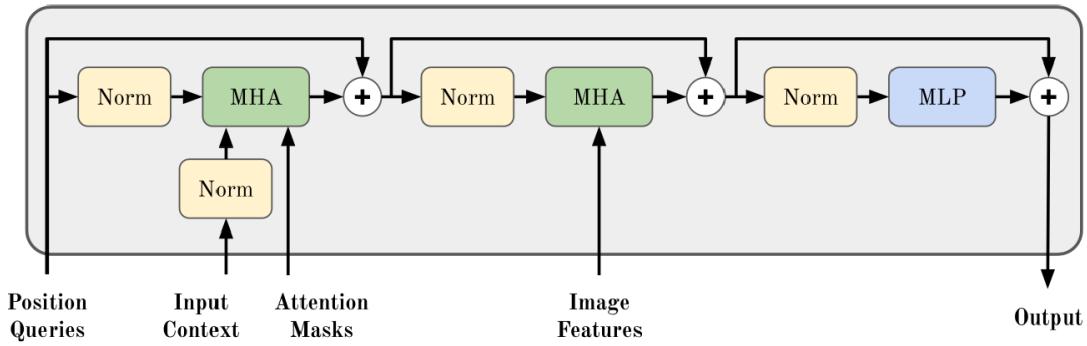
Khối encoder của PARSeq không khác gì với encoder của ViT [[5](#)] (vẫn 12 lớp - mỗi lớp là **Norm - MHA - [Residual] - Norm - MLP - [Residual]**).

#### 3.5.2.2 Decoder - Visio-lingual decoder

Khối decoder của thuật toán sẽ có vài điểm khác biệt so với các khối decoder mà ta thường gặp:



Hình 3.14: Tổng quan mô hình và cách huấn luyện của PARSeq (Nguồn: Bài báo [9])



Hình 3.15: Khối **decoder** của PARSeq - được đặt tên là Visio-lingual decoder. Mỗi lớp của khối sẽ có điểm khác biệt là có đến 2 lớp attention (Nguồn: Bài báo [9])

1. Chỉ sử dụng một lớp và lớp đó có đến 2 lớp MHA. Và đương nhiên là chức năng cũng như queries, keys, vectors của từng lớp cũng sẽ khác nhau.
2. Tiếp theo, số heads trong mỗi lớp MHA gấp đôi (thực tế là do việc ta chia đều đặc trưng mỗi lớp  $d_{model}$  cho 32 thay vì cho 64 như trong [5].

Cụ thể ý (1) phía trên sẽ được diễn tả như sau:

1. MHA *context-position attention* đầu tiên (khối MHA bên trái hình 3.15) có chức năng attend:

$$a_{\mathbf{p}-\mathbf{c}} = MHA(\mathbf{p}, \mathbf{c}, \mathbf{c}, \mathbf{m}) \quad (3.25)$$

---


$$\mathbf{h}_c = \mathbf{p} + a_{\mathbf{p}-\mathbf{c}} \in \mathbb{R}^{(T+1) \times d_{model}} \quad (3.26)$$

với vector ngữ cảnh - context  $\mathbf{c}$  sử dụng làm **keys - K** và **values - V**; còn vector positional embedding làm **p queries - Q**. Vì vậy đây được đặt tên là *context-position attention*.

2. MHA thứ hai *image-position attention* (khối MHA ở giữa của hình 3.15) sẽ có chức năng attend giữa đặc trưng ảnh  $\mathbf{z}$  với vị trí  $\mathbf{h}_c$ :

$$a_{\mathbf{h}-\mathbf{z}} = \mathbf{h}_c + MHA(\mathbf{h}_c, \mathbf{z}, \mathbf{z}) \quad (3.27)$$

$$\mathbf{h}_i = \mathbf{p} + a_{\mathbf{h}-\mathbf{z}} \in \mathbb{R}^{(T+1) \times d_{model}} \quad (3.28)$$

tương tự,  $\mathbf{h}_c$  sử dụng làm **keys - K**, **values - V**;  $\mathbf{z}$  làm **queries - Q**.

Cuối cùng, lớp MLP (khối phải hình 3.15) cho kết quả theo phương trình sau:

$$\mathbf{y} = Dec(\mathbf{z}, \mathbf{p}, \mathbf{c}, \mathbf{m}) \in \mathbb{R}^{(T+1) \times (S+1)} \quad (3.29)$$

### 3.5.2.3 Cách thức decoding

Như vậy một điểm quan trọng nhất mà nhóm chưa trình bày đó là cách decoding và attention mask -  $\mathbf{m}$ .

Hoán vị	[1,2,3]	[3,2,1]	[1,3,2]	[2,3,1]
$y_1$	[B] 1 0 0 0	$y_1$ 1 0 1 1	$y_1$ 1 0 0 0	$y_1$ 1 0 1 1
$y_2$	1 1 0 0	$y_2$ 1 0 0 1	$y_2$ 1 1 0 1	$y_2$ 1 0 0 0
$y_3$	1 1 1 0	$y_3$ 1 0 0 0	$y_3$ 1 1 0 0	$y_3$ 1 0 1 0
[E]	1 1 1 1	[E] 1 1 1 1	[E] 1 1 1 1	[E] 1 1 1 1

Bảng 3.2: Bảng biểu diễn cách để hoạt động của attention mask -  $\mathbf{m}$  với các hoán vị đã như nêu ra ở 3.23. Với [B] và [E] lần lượt là token biểu diễn khởi đầu và kết thúc từng chuỗi. Và cũng như đã nêu ở phương trình 3.24 - dễ dàng thấy, ngoài việc một xác suất có điều kiện sẽ của vị trí theo cột (có [E]) phụ thuộc vào vị trí theo hàng (có [B]); thì mọi vị trí đương nhiên đều phụ thuộc vào token khởi đầu [B]; và token [E] sẽ phụ thuộc vào mọi vị trí.

---

Như vậy cách masking hoàn toàn phù hợp với cả 3 cách masking được đưa ra từ 3 mô hình trước:

1. Unidirectional (trái sang phải/phải sang trái) - attention mask của AR;
2. Attention mask cho NAR (non-autoregressive) không cần masking bất kì chỗ nào vì mô hình NAR dự đoán cùng lúc/song song mọi kí tự;
3. Cloze mask (xem lại ABINet [8]).

	AR mask						NAR mask			cloze mask				
	[B]	y <sub>1</sub>	y <sub>2</sub>	...	y <sub>T</sub>		[B]			[B]	y <sub>1</sub>	y <sub>2</sub>	...	y <sub>T</sub>
y <sub>1</sub>	1	0	0	0	0		y <sub>1</sub>	1		y <sub>1</sub>	1	0	1	1
y <sub>2</sub>	1	1	0	0	0		y <sub>2</sub>	1		y <sub>2</sub>	1	1	0	1
:	1	1	1	...	0		:	1		:	1	1	1	...
y <sub>T</sub>	1	1	1	1	0		y <sub>T</sub>	1		y <sub>T</sub>	1	1	1	1
[E]	1	1	1	1	1		[E]	1		[E]	1	1	1	1

Bảng 3.3: Bảng minh họa cho các biểu diễn ba chiến lược masking của mô hình AR, NAR và cloze (mà ABINet sử dụng). Cả ba chiến lược này đều có thể được học thông qua một trong các tổ hợp liệt kê từ 3.23

## Chương 4

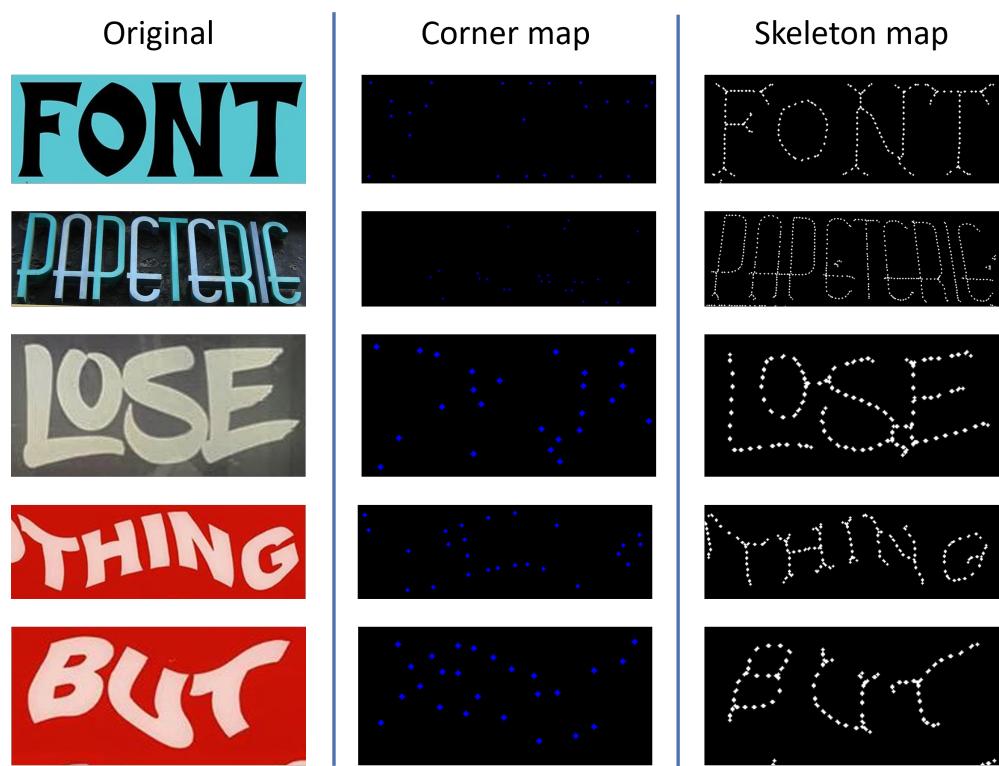
# ÁP DỤNG CHỈ DẪN VỀ NÉT CHO BÀI TOÁN NHẬN DẠNG VĂN BẢN NGHỆ THUẬT

### 4.1 Ý tưởng

Theo tác giả [1], hình dáng và hình dạng của các ký tự trong văn bản nghệ thuật rất đa dạng. Vì thế cần xây dựng những đặc trưng bất biến trong các ký tự để giữ lại các điểm quan trọng (key point) của nét ký tự, nhằm giảm sự ảnh hưởng của hiệu ứng gây nhiễu ở phông nền ảnh. Hơn nữa, sự rời rạc của bản đồ góc sẽ cắt đứt sự kết nối và sự chồng chéo của các ký tự. Đồng thời sự thưa thớt sẽ triệt tiêu hầu hết nhiễu ở nền. Dựa trên ý tưởng đó, tôi đưa ra hướng tiếp cận sử dụng nét chính của ký tự (Skeleton-Guided). Để trích xuất được bản đồ nét chính của văn bản trong ảnh, tôi sử dụng kết hợp hai thuật toán phân cụm có điều chỉnh và dò tìm khung xương (skeleton tracing). Bản đồ nét chính là ảnh nhị phân biểu diễn cấu trúc khung xương với chiều rộng là một điểm ảnh (pixel), bản đồ khung xương so với bản đồ góc 4.1. Tương tự với cách áp dụng của nhóm nghiên cứu [1]. Tôi sử dụng cơ chế Cross-Attention, coi các điểm khung xương là truy vấn và hình ảnh là giá trị (query) để các điểm khung xương tìm kiếm các

#### 4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật

đặc trưng hình ảnh quan tâm. Bằng cách này, bản đồ truy vấn khung xương sẽ hướng dẫn mô hình chú ý chính xác hơn đến nét chính của ký tự trong văn bản nghệ thuật.



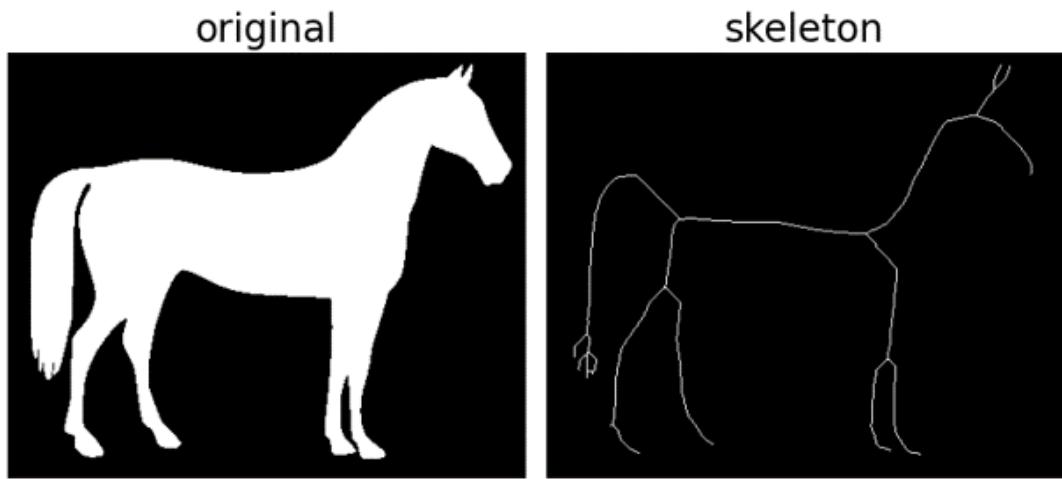
Hình 4.1: Hình minh họa bản đồ góc (corner map) và bản đồ khung xương (skeleton-map)

## 4.2 Thuật toán dò tìm khung xương

Dựa trên ý tưởng về trích xuất đặc trưng về nét chính đã đề ra, nhóm tiến hành tìm hiểu và nghiên cứu để đưa ra cách áp dụng thuật toán skeletonize[36] (tạm dịch: khung xương hóa) để tìm kiếm nét chính. Ý tưởng chính của thuật toán là làm giảm các đối tượng nhị phân thành các biểu diễn rộng 1 pixel (ảnh minh họa 4.2). Điều này có thể hữu ích cho việc trích xuất đặc trưng và/hoặc

#### 4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật

biểu diễn cấu trúc liên kết của đối tượng.



Hình 4.2: Hình ảnh minh họa thuật toán dò tìm khung xương (Nguồn: Thư viện Scikit-image)

Tuy nhiên để đảm bảo bản đồ khung xương chỉ lấy những thông tin về nét chính, đồng thời cắt đứt sự kết nối và sự chồng chéo của các nét phụ trong các ký tự cũng như việc giảm thiểu yếu tố nhiễu ở nền, chúng tôi quyết định biểu diễn ảnh khung xương thành các biểu diễn điểm nằm trên khung xương.

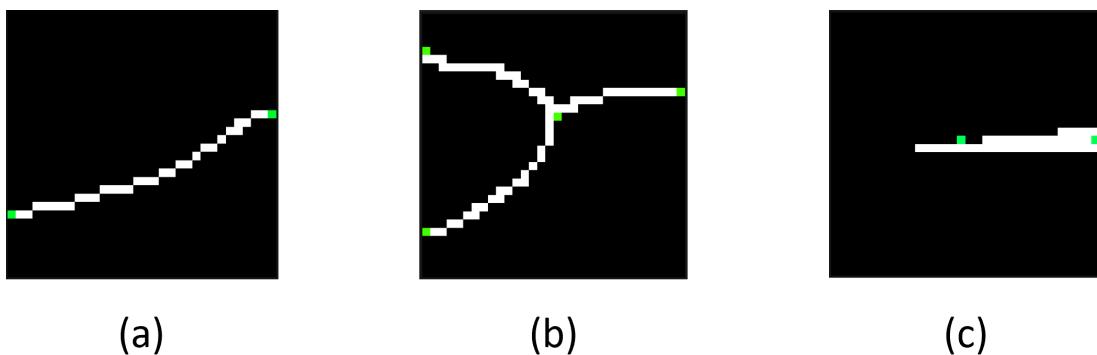
Để trích xuất các điểm đặc trưng của khung xương, thuật toán skeleton-tracing[37] đã được sử dụng và được tóm tắt như sau:

Xác định một hình ảnh nhị phân đã được khung xương hóa (skeletonize[36]) là một ma trận 2D bao gồm các pixel 0 (nền) và pixel 1 (đối tượng). Chia ảnh thành nhiều ảnh nhỏ có kích thước đã được xác định trước. Với mỗi ảnh đã chia, thực hiện tìm các đường (vector) theo các tiêu chí sau:

1. Ban đầu, đặt một cờ (flag) với giá trị là sai (false). Khi gặp một pixel có giá trị 1 trong khi cờ là sai, đặt cờ thành đúng (true) và đẩy tọa độ của pixel 1 vào một ngăn xếp (stack).

#### **4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật**

2. Khi gặp một pixel có giá trị 0 trong khi cờ là đúng, lấy tọa độ cuối cùng từ ngăn xếp và đẩy điểm trung điểm giữa điểm đó và tọa độ hiện tại vào ngăn xếp. Sau đó đặt cờ thành sai (false).
3. Sau khi đã duyệt qua tất cả các pixel biên, ngăn xếp hiện tại giờ chứa tọa độ cho tất cả các pixel "ra" (hoặc "vào") từ phần hình ảnh nhỏ này. Bằng cách kết nối các tọa độ này với tọa độ trung tâm của phần hình ảnh, ta có một ước lượng biểu diễn vector hóa của khung xương trong khu vực này bằng các đoạn thẳng này. Chúng ta cải thiện ước lượng bằng cách sử dụng các giả thiết (heuristics) sau đây (ảnh 4.3):
  - (a) Nếu có chính xác 2 pixel "ra". Có khả năng khu vực chứa một đường thẳng. Chúng ta trả về một đoạn thẳng kết nối hai pixel này.
  - (b) Nếu có 3 hoặc nhiều hơn 3 pixel "ra", có khả năng khu vực chứa một giao điểm hoặc "ngã tư". Chúng ta thực hiện tích chập trên ma trận để tìm ma trận con  $3 \times 3$  chứa nhiều pixel 1 nhất. Đặt tâm của tất cả các đoạn thẳng là tâm của ma trận con  $3 \times 3$  và trả về.
  - (c) Nếu chỉ có một pixel "ra", trả về đoạn thẳng kết nối nó với tâm của phần hình ảnh.



Hình 4.3: Hình minh họa cho ba giả thuyết (a), (b), (c)

#### **4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật**

4. Gộp kết quả từ hai mảng con lại và trả về tập hợp các đường (kết quả của thuật toán: ảnh 4.4).



Hình 4.4: Hình ảnh minh họa kết quả của thuật toán dò tìm khung xương

### **4.3 Thuật toán phân cụm**

Vì đầu vào của thuật toán dò tìm khung xương là ảnh nhị phân nên ta phải chuyển từ ảnh gốc với ba kênh màu sang ảnh nhị phân. Việc chuyển từ ảnh gốc

#### **4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật**

---

sang ảnh nhị phân tôi sử dụng thuật toán K-means để phân cụm được vùng văn bản trong ảnh. Tuy nhiên kết quả phân cụm bởi thuật toán K-means có thể tương ứng với vùng văn bản hoặc vùng nền. Vì vậy, dựa trên ý tưởng của tác giả bài báo [38], tác giả quan sát rằng các vùng của văn bản thường ở trung tâm của hầu hết dữ liệu văn bản trong ảnh, với bốn cạnh của hình ảnh chủ yếu là vùng nền. Tác giả thực hiện điều chỉnh thuật toán phân cụm để chọn các vùng văn bản thích hợp (Mã giả thuật toán: 1).

---

##### **Algorithm 1** Thuật toán phân cụm

---

Xem kết quả phân cụm là  $\Theta \in \{0, 1\}^{H \times W}$ .  
Định nghĩa  $\Theta_{i,j}$  là giá trị điểm ảnh ở hàng  $i$ , cột  $j$ .  
Tính tổng giá trị điểm ảnh cho mỗi cạnh biên:  
 $L = \sum_{i=1}^H \Theta_{i,1}; R = \sum_{i=1}^H \Theta_{i,W};$   
 $T = \sum_{j=1}^W \Theta_{1,j}; B = \sum_{j=1}^W \Theta_{H,j};$   
Tính điều kiện  $\Gamma$ :  
 $\Gamma = \mathbb{1}_{[T \geq \frac{W}{2}]} + \mathbb{1}_{[B \geq \frac{W}{2}]} + \mathbb{1}_{[L \geq \frac{H}{2}]} + \mathbb{1}_{[R \geq \frac{H}{2}]}$   
**if**  $\Gamma >= 3$  **then**  
     $result = 1 - \Theta$   
**else**  
     $result = \Theta$   
**end if**

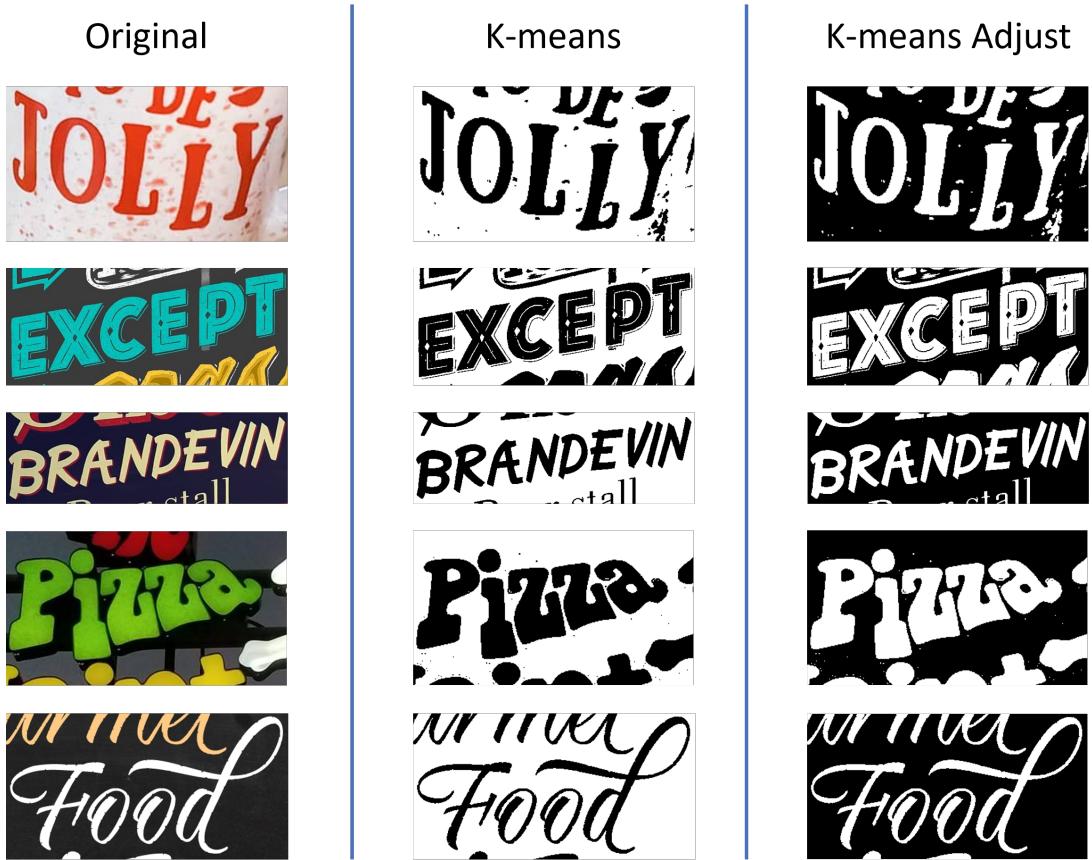
---

Nhìn vào một số ví dụ trong ảnh 4.5, ta có thể thấy được với các ảnh có nhiều ở bốn cạnh thì thuật toán K-means có điều chỉnh sẽ phân cụm tốt hơn so với thuật toán K-means gốc. Thuật toán K-means có điều chỉnh sẽ trả về kết quả phân cụm với giá trị của các pixel thuộc văn bản sẽ mang giá trị 1 và các pixel thuộc nền mang giá trị 0.

## **4.4 Áp dụng vào mô hình Transformer**

Như đã nói ở phần ý tưởng, tôi sử dụng đặc trưng về nét để hướng dẫn mô hình chú ý chính xác hơn đến nét chính của ký tự trong văn bản nghệ thuật. Vì thế, tôi thực hiện điều chỉnh dựa trên kiến trúc của mô hình CornerTransformer[1],

#### 4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật



Hình 4.5: Ảnh so sánh kết quả phân cụm của thuật toán K-means gốc và thuật toán K-means sau điều chỉnh

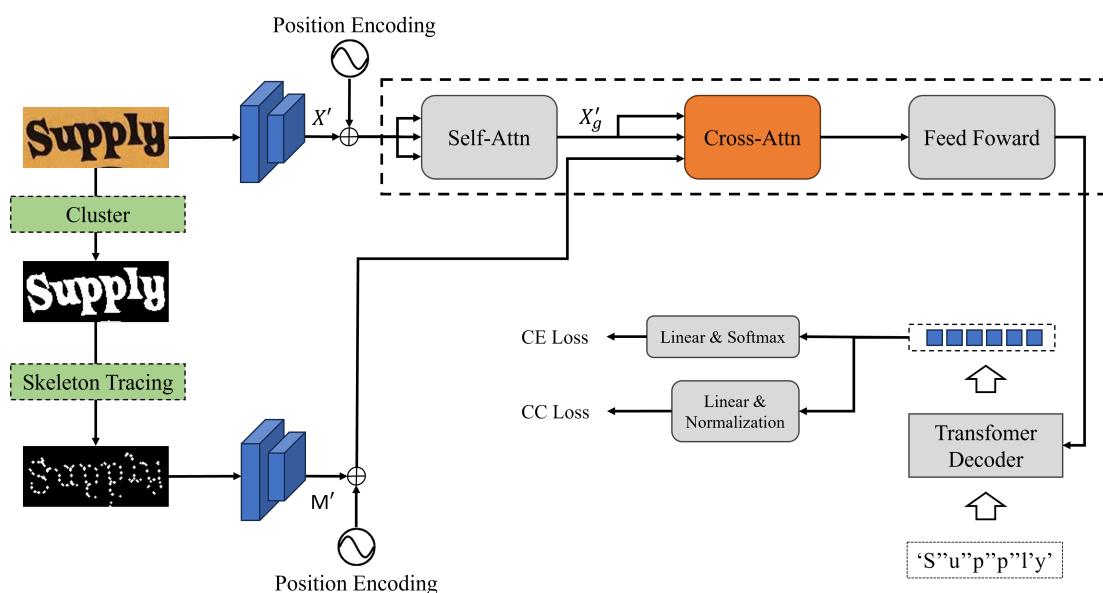
theo đó chúng tôi sử dụng các điểm trên khung xương là truy vấn (**queries - Q**) và hình ảnh sẽ là chìa khóa và giá trị (**keys - K; values - V**) để truy xuất, làm nổi bật đặc trưng tại các vị trí khung xương cần được chú ý trong hình ảnh (công thức Attention 4.1).

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (4.1)$$

Với ảnh đầu vào  $X \in R^{H \times W \times 3}$ , tôi sử dụng lần lượt sử dụng thuật toán phân cụm và thuật toán dò tìm khung xương để tạo ra bản đồ khung xương (skeleton map)  $M \in R^{H \times W \times 1}$ . X và M sau đó sẽ đi qua hai lớp tích chập để tạo thành đặc

#### 4. Áp dụng chỉ dẫn về nét cho bài toán nhận dạng văn bản nghệ thuật

trưng  $X' \in R^{\frac{H}{4} \times \frac{W}{4} \times C}$  và  $M' \in R^{\frac{H}{4} \times \frac{W}{4} \times C}$  tương ứng, trong đó  $C$  là số chiều của đặc trưng từ lớp tích chập, các đặc trưng này cũng sẽ được tích hợp thông tin vị trí thông qua Position Encoding trước khi đi vào khối Encoder tương tự như trong ViT. Mặt khác, đặc trưng toàn cục  $X'_g$  sẽ được học từ  $X'$  thông qua cơ chế Multi-head attention của Transformer, sau đó  $X'_g$  sẽ được kết hợp với  $M'$  thông qua cơ chế Multi-head cross-attention. Sau cùng, đầu ra của khối Encoder và embedded vector của chuỗi kí tự sẽ được đi qua khối Transformer Decoder để dự đoán chuỗi đầu ra. Hình minh họa mô hình Transformer sử dụng hướng tiếp cận chỉ dẫn về nét 4.6.



Hình 4.6: Mô hình Transformer sử dụng hướng tiếp cận chỉ dẫn về nét (Skeleton-Guided)

## **Chương 5**

# **THỰC NGHIỆM VÀ ĐÁNH GIÁ**

Trong chương này, tôi sẽ giới thiệu về bộ dữ liệu chữ nghệ thuật cho tiếng Anh và tiếng Việt cũng như trình bày về cách xây dựng tập dữ liệu chữ thư pháp tiếng Việt. Bên cạnh đó, sinh viên sẽ huấn luyện và đánh giá hướng tiếp cận đã nêu ở chương 4 cùng với các phương pháp đã trình bày trong chương 3 trên các tập dữ liệu chữ nghệ thuật.

### **5.1 Các bộ dữ liệu cho bài toán nhận dạng văn bản nghệ thuật**

#### **5.1.1 Bộ dữ liệu chữ nghệ thuật cho tiếng Anh và tiếng Việt**

Tôi tận dụng lại kết quả từ khóa luận tốt nghiệp của khóa trên về bài toán nhận dạng chữ nghệ thuật trong ảnh. Cụ thể tôi sử dụng lại bộ dữ liệu chữ nghệ thuật cho tiếng Anh và tiếng Việt, được gọi là bộ dữ liệu VNArtText.

VNArtText là bộ dữ liệu tập trung vào vấn đề nhận dạng văn bản nghệ thuật. Bộ dữ liệu này được thu thập từ nhiều bộ dữ liệu bao gồm BKAI, ICDAR2013[13], WordArt[1], Vintext[10], TotalText[12], CUTE80[11], và VietSignBoard. Bộ dữ liệu có nguồn gốc rất đa dạng, từ các biển quảng cáo, biển hiệu của các cửa hàng

## **5. Thực nghiệm và đánh giá**

cho đến ảnh chụp áp phích, thiệp mời, logo, .... Bộ dữ liệu VNArtText bao gồm 11822 ảnh chữ nghệ thuật và 73873 ảnh không phải chữ nghệ thuật. Thông tin thống kê chi tiết về bộ dữ liệu VNArtText được thể hiện ở bảng [5.1](#).

<b>Dataset</b>	<b>Số ảnh không phải chữ nghệ thuật</b>	<b>Số ảnh chữ nghệ thuật</b>	<b>Ngôn ngữ</b>
VinText	39483	2947	VN
BKAI-TEXT	6105	583	VN
VietSignboard	24355	1387	VN
CUTE80	19	116	EN
TOTAL-TEXT	6966	4810	EN
ICDAR13	1445	468	EN
WordArt	-	6316	EN
<b>Tổng</b>	<b>78373</b>	<b>16627</b>	-

Bảng 5.1: Bảng thống kê về số lượng ảnh chữ nghệ thuật, ảnh không phải chữ nghệ thuật và số lượng đóng góp của các bộ dữ liệu

### **5.1.2 Bộ dữ liệu thư pháp tiếng Việt**

#### **5.1.2.1 Thu thập và gán nhãn dữ liệu**

Với mục đích mở rộng dữ liệu cho bài toán và xây dựng một bộ dữ liệu cho bài toán nhận dạng thư pháp tiếng Việt, tôi đã thực hiện việc thu thập dữ liệu từ nhiều nguồn, bao gồm cả chụp trực tiếp và thu thập từ các diễn đàn, mạng xã hội, cũng như công cụ tìm kiếm (xem hình [5.1](#)). Sau khi thu thập dữ liệu, tôi sử dụng công cụ LabelMe để gán nhãn cho dữ liệu (ảnh minh họa quá trình gán nhãn dữ liệu [5.2](#)). Cụ thể hình ảnh sẽ được cắt thành hình chữ nhật để chuẩn bị cho quá trình xử lý. Tuy nhiên, với đặc điểm thư pháp thường có từ viết không thẳng hàng, kích thước giống nhau, và các nét chữ đan xen, việc cắt hình chữ nhật có thể gây nhiễu và mất nét.

Để giải quyết vấn đề này, chúng tôi đã áp dụng phương pháp cắt ảnh theo

## 5. Thực nghiệm và đánh giá



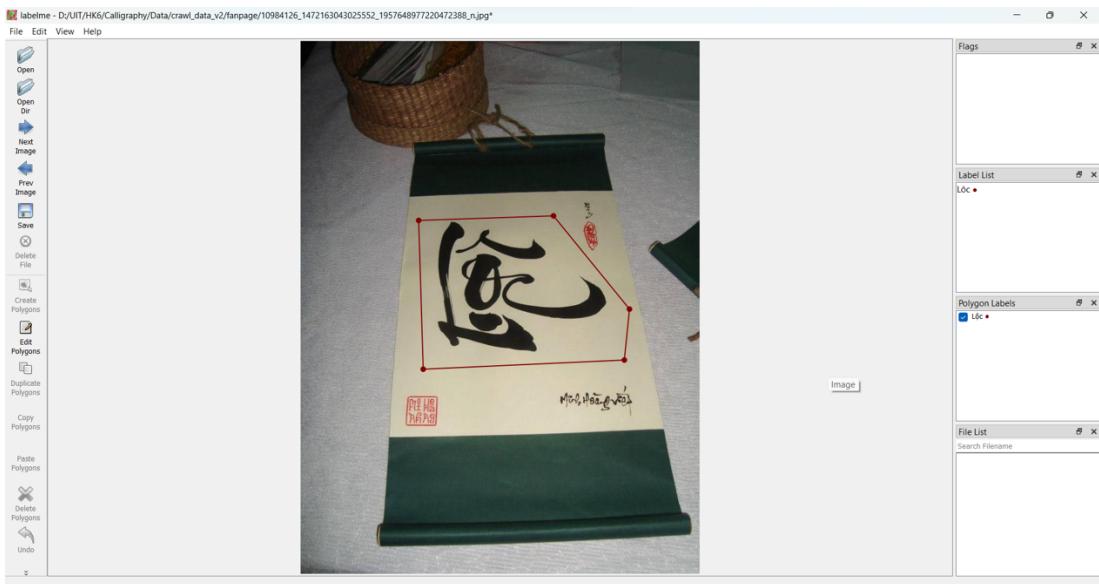
Hình 5.1: Ảnh chứa chữ thư pháp trên các phương tiện truyền thông (a) và ảnh được cắt bởi công cụ gán nhãn (b)

hình đa giác đối với những trường hợp cụ thể này, nhằm giảm thiểu nhiễu và bảo toàn nét chữ tối ưu (xem hình 5.1). Sau quá trình cắt, ảnh gốc được chia thành nhiều ảnh con, mỗi ảnh chứa một từ thư pháp tiếng Việt và được gán nhãn cho dữ liệu.

### 5.1.2.2 Một số đánh giá trên tập dữ liệu

Bộ dữ liệu chữ thư pháp tiếng Việt, mà chúng tôi đã xây dựng, có tổng cộng 15441 ảnh, được phân chia thành 2980 từ (nhóm). Trong đó, các từ có số lượng ảnh từ 6 đến 20 chiếm tỷ lệ 37.82% của tổng số ảnh. Con số thống kê này chỉ ra rằng tập dữ liệu mang đến một độ phong phú lớn, biểu diễn các từ trong nhiều ngữ cảnh khác nhau. Trong quá trình thu thập và gán nhãn, chúng tôi đã được

## 5. Thực nghiệm và đánh giá



Hình 5.2: Giao diện của công cụ gán nhãn dữ liệu LabelMe

hỗ trợ tư vấn của các chuyên gia thư pháp, đảm bảo tính chính xác của dữ liệu. Điều này bao gồm việc xác nhận các loại chữ và phương pháp thu thập, đó là một phần quan trọng trong việc đảm bảo chất lượng của tập dữ liệu.

Bên cạnh tính đa dạng về hình dáng và kích thước, tập dữ liệu cũng đưa ra những thách thức khác như ảnh được chụp từ nhiều góc độ, phông nền phức tạp, hình ảnh có hình dạng cắt bất thường như hình chữ nhật và đa giác, cùng với đặc điểm phức tạp của chữ viết (ảnh minh họa 5.3). Những khía cạnh này đặt ra một thách thức thực tế và hấp dẫn trong bài toán nhận dạng, đòi hỏi mô hình phải có khả năng hiểu và xử lý các biến thể đa dạng của chữ thư pháp tiếng Việt.



Hình 5.3: Một số hình ảnh trong bộ dữ liệu chữ thư pháp tiếng Việt

## 5.2 Các độ đo để thực hiện đánh giá

### 5.2.1 Độ chính xác (Accuracy)

Độ chính xác (Accuracy) là thước đo đánh giá hiệu suất của mô hình, tính theo tỷ lệ giữa số dự đoán đúng và tổng số dự đoán. Công thức tính độ chính xác (accuracy):

$$\text{Độ chính xác} = \frac{\text{Tổng số dự đoán đúng}}{\text{Tổng số dự đoán}} \quad (5.1)$$

Một dự đoán được xem là đúng khi và chỉ khi toàn bộ ký tự dự đoán trùng khớp với toàn bộ ký tự của nhãn tương ứng. Ví dụ, dự đoán “Chúc” và nhãn “Chúc” sẽ được xem là một dự đoán đúng. Khi so sánh kết quả dự đoán và nhãn ta có thể xem xét đến hai trường hợp có hoặc không có kí tự hoa, thường (case-insensitive).

## **5. Thực nghiệm và đánh giá**

---

sensitive). Trong khuôn khổ của khóa luận, sinh viên sẽ so sánh phân biệt hoa thường để có được kết quả khách quan nhất.

### **5.2.2 Tỉ lệ lỗi (CER)**

Ngoài độ chính xác (accuracy), một độ đo khác cũng được dùng rất phổ biến cho bài toán nhận dạng văn bản là tỉ lệ lỗi (character error rate - CER). Tùy thuộc trường hợp sử dụng và điều kiện liên quan, tỉ lệ lỗi ký tự CER được sử dụng để đánh giá mức độ hiệu quả của các mô hình nhận dạng văn bản tương ứng. Các tình huống và độ phức tạp khác nhau (ví dụ: văn bản in so với văn bản viết tay, văn bản chữ nghệ thuật) có thể dẫn đến các hiệu suất của các mô hình khác nhau. Công thức tính tỉ lệ lỗi (CER):

$$CER = \frac{S + D + I}{N} \quad (5.2)$$

Trong đó **S** là số lần thay thế, **D** là số lần xóa, **I** là số lần chèn và **N** là tổng số ký tự trong văn bản tham chiếu. Cũng như với độ chính xác, tôi sẽ so sánh phân biệt hoa thường để có được kết quả khách quan nhất.

### **5.2.3 Số khung hình trên giây (FPS)**

Để áp dụng được vào thực tế, mô hình không chỉ phải đạt độ chính xác cao mà tốc độ dự đoán còn phải nhanh. Do đó, nhóm sẽ đo số khung hình trên giây khi mô hình dự đoán để đánh giá tốc độ xử lý. Tốc độ nhanh và vẫn giữ được độ chính xác cao đồng nghĩa với việc mô hình có khả năng cao có thể ứng dụng trong thực tế. Số khung hình trên giây được tính theo công thức:

$$FPS = \frac{\text{Tổng số khung hình}}{\text{Tổng thời gian xử lý}} \quad (5.3)$$

### 5.3 Kết quả thực nghiệm và đánh giá

#### 5.3.1 Đánh giá hướng tiếp cận mới so với phương pháp nền tảng

Để kiểm tra xem hướng tiếp cận sử dụng chỉ dẫn về nét có ảnh hưởng khác với phương pháp nền tảng, ở đây là Corner Transformer. Tôi tiến hành huấn luyện và đánh giá hướng tiếp cận mới cùng với mô hình Corner Transformer. Cụ thể, tôi chia các ảnh nghệ thuật trong bộ dữ liệu VNArtText thành hai tập, tập huấn luyện gồm 12475 ảnh và tập kiểm thử gồm 4152 ảnh. Tiếp đến tôi huấn luyện hai mô hình với 20 epochs và có được kết quả thể hiện trong bảng 5.2.

Phương pháp	Accuracy (%)	CER
Corner Transformer	0.67	1.0003
Skeleton-Guided	0.75	0.9859

Bảng 5.2: Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của hai mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText

Nhìn vào bảng trên có thể thấy rằng hướng tiếp cận mà nhóm đề xuất có sự khác biệt với phương pháp Corner Transformer. Với thực nghiệm này, ở các bước học đầu tiên thì hướng tiếp cận Skeleton-Guided học tốt hơn khi có độ chính xác cao hơn cũng như độ lỗi ít hơn so với Corner Transformer.

#### 5.3.2 Đánh giá dựa trên việc điều chỉnh tham số có sẵn

Để so sánh với các phương pháp tiên tiến hiện nay như đã nêu ở phần 3. Tôi tiến hành điều chỉnh bộ tham số có sẵn trên bộ dữ liệu chữ nghệ thuật. Cụ thể nhóm sinh viên sử dụng lại tham số được huấn luyện trên dữ liệu tổng hợp MJSynth [33] và SynthText [34] của bài toán nhận dạng văn bản và huấn luyện thêm 20 epoch trên các ảnh nghệ thuật trong bộ dữ liệu VNArtText (tương tự với thực nghiệm trước). Do chưa có bộ tham số huấn luyện trên tập dữ liệu tổng hợp nên tôi sử dụng lại bộ tham số của mô hình Corner Transformer cho mô hình

## 5. Thực nghiệm và đánh giá

---

Skeleton-Guided. Kết quả thực nghiệm được thể hiện trong bảng 5.3 và 5.4.

Phương pháp	Accuracy (%)							
	BKAI	CUTE	IC13	Vin	Total	VietSB	WordArt	All
ABINet	55.06	75.00	73.31	44.34	57.04	50.72	65.85	58.09
Corner Transformer	<b>79.17</b>	79.31	82.87	<b>71.15</b>	<b>73.85</b>	73.68	<b>71.14</b>	<b>73.31</b>
SATRN	74.40	80.00	81.67	67.57	61.06	72.25	67.77	68.55
PARSeq	75.30	78.45	81.67	67.38	66.24	<b>75.12</b>	69.69	70.23
ViTSTR	59.52	58.62	63.75	46.27	43.82	48.80	53.28	51.01
Skeleton-Guided	70.54	<b>80.17</b>	<b>84.46</b>	59.83	72.84	62.68	70.95	69.12

Bảng 5.3: Bảng kết quả độ chính xác của các mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText

Phương pháp	Character Error Rate (CER)							
	BKAI	CUTE	IC13	Vin	Total	VietSB	WordArt	All
ABINet	0.138	0.169	0.101	0.200	0.189	0.170	0.144	0.163
Corner Transformer	<b>0.077</b>	0.147	0.070	0.118	0.109	0.129	0.130	<b>0.116</b>
SATRN	0.085	0.146	0.065	<b>0.117</b>	0.174	0.108	0.134	0.130
PARSeq	0.096	0.147	0.080	0.131	0.164	<b>0.096</b>	0.126	0.129
ViTSTR	0.149	0.245	0.148	0.220	0.320	0.217	0.220	0.232
Skeleton-Guided	0.089	<b>0.145</b>	<b>0.062</b>	0.148	<b>0.101</b>	0.158	<b>0.125</b>	0.120

Bảng 5.4: Bảng kết quả tỉ lệ lỗi của các mô hình trên dữ liệu kiểm thử là các ảnh nghệ thuật trong bộ dữ liệu VNArtText

Quan sát kết quả thực nghiệm điều chỉnh tham số, mô hình Corner Transformer cho thấy sự vượt trội ở cả hai độ đo là độ chính xác và tỉ lệ lỗi. Khi xét riêng độ chính xác của từng tập dữ liệu thì hướng tiếp cận mới (Skeleton-Guided) đã đạt được độ chính xác cao nhất trên tập dữ liệu CUTE80 và ICDAR13 và tỉ lệ lỗi thấp nhất trên các tập dữ liệu CUTE80, ICDAR13, TotalText và WordArt. Khi xét trên toàn bộ dữ liệu đánh giá thì Skeleton-Guided là đạt độ chính xác cao thứ ba sau mô hình Corner Transformer và PARSeq, đối với độ đo tỉ lệ lỗi thì hướng tiếp cận này có tỉ lệ lỗi thấp thứ hai chỉ sau mô hình Corner Transformer.

### **5.3.3 Đánh giá hướng tiếp cận mới trên toàn bộ dữ liệu chữ nghệ thuật**

Trong phần này, nhóm sẽ trình bày về việc cài đặt các mô trường thực nghiệm và đánh giá các phương pháp trên bộ dữ liệu chữ nghệ thuật dựa theo bài báo [1]. Đầu tiên là về phần dữ liệu, để đánh giá công bằng và có thể so sánh các phương pháp được với nhau, nhóm nghiên cứu bài báo [1] đã huấn luyện trên 2 tập dữ liệu tập hợp là MJSynth [33] và SynthText [34], sau đó đánh giá trên tập dữ liệu WordArt. Dựa trên ý tưởng đó, sinh viên sẽ chia dữ liệu huấn luyện và đánh giá như sau:

- Dữ liệu huấn luyện: Tất cả các ảnh không phải chữ nghệ thuật trong bộ dữ liệu VNArtText, gồm có 78373 ảnh.
- Dữ liệu đánh giá 1: Tất cả các ảnh là chữ nghệ thuật trong bộ dữ liệu VNArtText, gồm có 11822 ảnh (tập dữ liệu WordArt chỉ lấy tập test - 1511 ảnh).
- Dữ liệu đánh giá 2: Dữ liệu đánh giá của bộ dữ liệu thư pháp tiếng Việt gồm 3109 ảnh.

Việc huấn luyện trên dữ liệu không có chứa chữ nghệ thuật và đánh giá trên tập dữ liệu chỉ bao gồm chữ nghệ thuật sẽ cho ta đánh giá chính xác được sự bất biến của mô hình đối với sự thay đổi của phân phối dữ liệu, khi các chữ nghệ thuật thường rất khác so với chữ bình thường. Các phương pháp được giữ cài đặt mặc định như trong bài báo gốc, và được huấn luyện trên tập dữ liệu huấn luyện trong 100 epoch với batch size là 42. Đối với phần đánh giá tốc độ, các mô hình sẽ dự đoán với batch size là 1 và tối ưu hóa gộp hai dữ liệu đánh giá lại để có bảng kết quả chung 5.7. Kết quả dự đoán theo độ chính xác và tỉ lệ lỗi được tổng hợp thành 5.5 bảng 5.6.

## 5. Thực nghiệm và đánh giá

---

Phương pháp	Accuracy (%)	CER
ABINet	52.01	0.2962
Corner Transformer	48.33	0.3598
SATRN	44.69	0.3939
PARSEq	<b>58.43</b>	<b>0.2558</b>
ViTSTR	31.94	0.5189
Skeleton-Guided	43.60	0.494

Bảng 5.5: Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của các mô hình trên dữ liệu đánh giá 1

Phương pháp	Accuracy (%)	CER
ABINet	<b>3.09</b>	<b>0.9083</b>
Corner Transformer	1.16	1.0311
SATRN	1.06	1.0782
PARSEq	2.12	1.0129
ViTSTR	0.10	0.9795
Skeleton-Guided	1.16	1.3787

Bảng 5.6: Bảng kết quả đánh giá độ chính xác và tỉ lệ lỗi của các mô hình trên dữ liệu đánh giá 2

Phương pháp	FPS
ABINet	15.19
Corner Transformer	1.82
SATRN	1.92
PARSEq	24.53
ViTSTR	<b>69.69</b>
Skeleton-Guided	1.67

Bảng 5.7: Bảng kết quả tốc độ dự đoán của các mô hình trên cả hai dữ liệu đánh giá

Nhìn vào bảng kết quả đánh giá độ chính xác (5.5) và tỉ lệ lỗi (5.6) trên hai bộ dữ liệu đánh giá, ta có thể thấy sự vượt trội của các phương pháp sử dụng mô hình ngôn ngữ. Cụ thể mô hình PARSeq đạt kết quả tốt nhất trên dữ liệu đánh giá 1 (gồm các ảnh chữ nghệ thuật VNArtText). và mô hình ABINet đạt kết quả tốt nhất trên dữ liệu đánh giá 2 (gồm các ảnh kiểm thử trong tập dữ liệu thư pháp tiếng Việt).

## 5. Thực nghiệm và đánh giá

---

Bảng đánh giá tốc độ dự đoán của các mô hình (5.7) cho thấy sự áp đảo của các mô hình sử dụng kiến trúc ViT như ViTSTR, PARSeq, ABINet. Với việc sử dụng hầu như hoàn toàn kiến trúc Transformer, các mô hình Corner Transformer và Skeleton-Guided cần rất nhiều thời gian để dự đoán.

Ảnh	Nhãn	Skeleton-Guided	ABINet	Corner-Guided	SATRN	PARSeq	ViSTR
	Joni	Joni	Tőn	Tői	Toiá	Joniv	JÖN
	FRANCE	FRANCE	PRANCE	PFRANCE	FRANGE	TRANCY	FRANCH
	CLUB	CLUB	BLLEE	ELUB	ELLUB	ELUS	FLLE
	THE	THE	TE	IFFE	INHE	THẾ	HH
	MS	MS	AUS	ANG	COLF	200Ly	HUOS
	Đăng	Đăng	Đặng	PINS	PHÚ	BẦN	PRRG

Hình 5.4: Một số kết quả dự đoán mà hướng tiếp cận Skeleton-Guided đúng và các phương pháp khác sai

Để phân tích kĩ hơn về hướng tiếp cận sử dụng chỉ dẫn về nét (Skeleton-Guided). Tôi tiến hành lọc ra những trường hợp mà Skeleton-Guided dự đoán đúng nhưng các mô hình khác dự đoán sai (hình 5.4) và ngược lại (hình 5.5). Hướng tiếp cận Skeleton-Guided sẽ gặp khó khăn khi dự đoán các trường hợp ảnh có màu nền phức tạp hay màu nền trùng với màu chữ, ảnh có dính nét của các văn bản nằm gần với văn bản chính. Các ảnh có hiệu ứng 3D cũng gây ra khó khăn cho mô hình vì thuật toán dò tìm khung xương sẽ cho ra nhiều đường nét dù chỉ có 1 ký tự. Tuy nhiên hướng tiếp cận Skeleton-Guided lại thể hiện tốt trên các ảnh chữ cong, các ảnh có nét chữ cách điệu, kéo dài.

## 5. Thực nghiệm và đánh giá

---

Ảnh	Nhãn	Skeleton-Guided	ABINet	Corner-Guided	SATRN	PARSeq	ViSTR
	ORGANS	DRGATS	ORGANS	ORGANS	ORGANS	ORGANS	ORGANS
	TELMO	TELMOR	TELMO	TELMO	TELMO	TELMO	TELMO
	HG	4G	HG	HG	HG	HG	HG
	TP	F	TP	TP	TP	TP	TP
	MAHI	MÀU	MAHI	MAHI	MAHI	MAHI	MAHI
	THE	FITE	THE	THE	THE	THE	THE

Hình 5.5: Một số kết quả dự đoán mà hướng tiếp cận Skeleton-Guided sai và các phương pháp khác đúng

# **Chương 6**

## **KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN**

### **6.1 Kết luận**

Trong khóa luận này, sinh viên đã tìm hiểu về bài toán nhận dạng văn bản, đặc biệt là tính thách thức của văn bản nghệ thuật. Xây dựng và đóng góp tập dữ liệu chữ thư pháp tiếng Việt với 15441 ảnh, được phân vào 2980 từ (nhóm). Ngoài ra, tôi còn tổng kết kết quả thu thập và đánh giá tập dữ liệu thư pháp tiếng Việt thành một bài báo khoa học (được đính kèm sau cuốn khóa luận). Tìm hiểu, nghiên cứu và cài đặt hướng tiếp cận "Áp dụng chỉ dẫn về nét" (Skeleton-Guided) cho bài toán nhận dạng văn bản nghệ thuật. Sau cùng, sinh viên thực hiện các thực nghiệm và đánh giá hướng tiếp cận mới cùng với các phương pháp tiên tiến hiện nay trên bộ dữ liệu chữ nghệ thuật (VNArtText) cũng như trên bộ dữ liệu thư pháp tiếng Việt.

### **6.2 Hướng phát triển**

Trong tương lai, sinh viên mong muốn hoàn thiện các mục tiêu sau:

## **6. Kết luận và hướng phát triển**

---

- Huấn luyện và đánh giá mô hình Skeleton-Guided trên hai tập dữ liệu tổng hợp là MJSynth[33] và SynthText[34]
- Tìm hướng cải thiện bước dò tìm khung xương (skeleton-tracing) nhằm cho ra bản đồ khung xương chính xác với nét của kí tự hơn
- Tìm hiểu các phương pháp và thực hiện sinh dữ liệu chữ nghệ thuật
- Áp dụng hướng tiếp cận dò tìm khung xương, dò tìm góc vào các phương pháp sử dụng mô hình ngôn ngữ
- Xây dựng ứng dụng minh họa cho bài toán

# Tài liệu tham khảo

- [1] X. Xie, L. Fu, Z. Zhang, Z. Wang, and X. Bai, “Toward understanding wordart: Corner-guided transformer for scene text recognition,” 2022. [iii](#), [ix](#), [xi](#), [2](#), [4](#), [9](#), [10](#), [16](#), [42](#), [43](#), [44](#), [45](#), [46](#), [56](#), [62](#), [67](#), [70](#), [78](#)
- [2] M. Jaderberg, K. Simonyan, A. Zisserman, *et al.*, “Spatial transformer networks,” *Advances in neural information processing systems*, vol. 28, 2015. [x](#), [14](#), [15](#)
- [3] M.-T. Luong, H. Pham, and C. D. Manning, “Effective approaches to attention-based neural machine translation,” *arXiv preprint arXiv:1508.04025*, 2015. [x](#), [19](#), [22](#)
- [4] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, “Attention is all you need,” *Advances in neural information processing systems*, vol. 30, 2017. [x](#), [xi](#), [24](#), [26](#), [30](#), [33](#), [45](#)
- [5] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, *et al.*, “An image is worth 16x16 words: Transformers for image recognition at scale,” *arXiv preprint arXiv:2010.11929*, 2020. [xi](#), [13](#), [33](#), [34](#), [58](#), [59](#)

## TÀI LIỆU THAM KHẢO

---

- [6] J. Lee, S. Park, J. Baek, S. J. Oh, S. Kim, and H. Lee, “On recognizing texts of arbitrary shapes with 2d self-attention,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pp. 546–547, 2020. [xi](#), [4](#), [16](#), [37](#), [38](#), [41](#), [44](#)
- [7] R. Atienza, “Vision transformer for fast and efficient scene text recognition,” in *International Conference on Document Analysis and Recognition*, pp. 319–334, Springer, 2021. [xi](#), [4](#), [16](#), [46](#), [47](#), [48](#)
- [8] S. Fang, H. Xie, Y. Wang, Z. Mao, and Y. Zhang, “Read like humans: Autonomous, bidirectional and iterative language modeling for scene text recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7098–7107, 2021. [xii](#), [4](#), [49](#), [51](#), [52](#), [54](#), [56](#), [61](#)
- [9] D. Bautista and R. Atienza, “Scene text recognition with permuted autoregressive sequence models,” in *European Conference on Computer Vision*, (Cham), pp. 178–196, Springer Nature Switzerland, 10 2022. [xii](#), [xiii](#), [4](#), [56](#), [59](#)
- [10] N. Nguyen, T. Nguyen, V. Tran, M.-T. Tran, T. D. Ngo, T. H. Nguyen, and M. Hoai, “Dictionary-guided scene text recognition,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7383–7392, 2021. [4](#), [11](#), [70](#)
- [11] A. Risnumawan, P. Shivakumara, C. S. Chan, and C. L. Tan, “A robust arbitrary text detection system for natural scene images,” *Expert Systems with Applications*, vol. 41, no. 18, pp. 8027–8048, 2014. [4](#), [70](#)
- [12] C. K. Ch’ng and C. S. Chan, “Total-text: A comprehensive dataset for scene text detection and recognition,” in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, pp. 103–107, 2017. [4](#), [16](#), [37](#), [38](#), [41](#), [44](#)

## TÀI LIỆU THAM KHẢO

---

- ence on document analysis and recognition (ICDAR)*, vol. 1, pp. 935–942, IEEE, 2017. [4](#), [70](#)
- [13] D. Karatzas, F. Shafait, S. Uchida, M. Iwamura, L. G. i Bigorda, S. R. Mestre, J. Mas, D. F. Mota, J. A. Almazan, and L. P. De Las Heras, “Icdar 2013 robust reading competition,” in *2013 12th international conference on document analysis and recognition*, pp. 1484–1493, IEEE, 2013. [4](#), [70](#)
- [14] X. Chen, L. Jin, Y. Zhu, C. Luo, and T. Wang, “Text recognition in the wild: A survey,” *ACM Computing Surveys (CSUR)*, vol. 54, no. 2, pp. 1–35, 2021. [7](#), [13](#)
- [15] Q. Ye and D. Doermann, “Text detection and recognition in imagery: A survey,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 37, no. 7, pp. 1480–1500, 2014. [7](#)
- [16] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, “What is wrong with scene text recognition model comparisons? dataset and model analysis,” in *International Conference on Computer Vision (ICCV)*, 2019. [13](#), [46](#)
- [17] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014. [13](#), [16](#)
- [18] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778, 2016. [13](#), [16](#), [51](#)
- [19] H. Touvron, M. Cord, M. Douze, F. Massa, A. Sablayrolles, and H. Jégou, “Training data-efficient image transformers & distillation through at-

## TÀI LIỆU THAM KHẢO

---

- tention,” in *International Conference on Machine Learning*, pp. 10347–10357, PMLR, 2021. [13](#)
- [20] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, “Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks,” in *Proceedings of the 23rd international conference on Machine learning*, pp. 369–376, 2006. [14](#)
- [21] D. Bahdanau, K. Cho, and Y. Bengio, “Neural machine translation by jointly learning to align and translate,” *arXiv preprint arXiv:1409.0473*, 2014. [14, 22](#)
- [22] W. Liu, C. Chen, K.-Y. K. Wong, Z. Su, and J. Han, “Star-net: a spatial attention residue network for scene text recognition.,” in *BMVC*, vol. 2, p. 7, 2016. [14, 16](#)
- [23] B. Shi, X. Wang, P. Lyu, C. Yao, and X. Bai, “Robust scene text recognition with automatic rectification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4168–4176, 2016. [14, 16](#)
- [24] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, “Aster: An attentional scene text recognizer with flexible rectification,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 41, no. 9, pp. 2035–2048, 2018. [14, 16, 50](#)
- [25] I. Sutskever, O. Vinyals, and Q. V. Le, “Sequence to sequence learning with neural networks,” *Advances in neural information processing systems*, vol. 27, 2014. [18, 20](#)
- [26] J. Shi *et al.*, “Good features to track,” in *1994 Proceedings of IEEE conference on computer vision and pattern recognition*, pp. 593–600, IEEE, 1994. [43](#)

## TÀI LIỆU THAM KHẢO

---

- [27] D. DeTone, T. Malisiewicz, and A. Rabinovich, “Superpoint: Self-supervised interest point detection and description,” in *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 224–236, 2018. [44](#)
- [28] C. Harris, M. Stephens, *et al.*, “A combined corner and edge detector,” in *Alvey vision conference*, vol. 15, pp. 10–5244, Manchester, UK, 1988. [44](#)
- [29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, “Bert: Pre-training of deep bidirectional transformers for language understanding,” *arXiv preprint arXiv:1810.04805*, 2018. [49](#), [50](#), [53](#), [55](#), [57](#)
- [30] D. Yu, X. Li, C. Zhang, T. Liu, J. Han, J. Liu, and E. Ding, “Towards accurate scene text recognition with semantic reasoning networks,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 12113–12122, 2020. [51](#)
- [31] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2117–2125, 2017. [52](#)
- [32] O. Ronneberger, P. Fischer, and T. Brox, “U-net: Convolutional networks for biomedical image segmentation,” in *International Conference on Medical image computing and computer-assisted intervention*, pp. 234–241, Springer, 2015. [52](#)
- [33] M. Jaderberg, K. Simonyan, A. Vedaldi, and A. Zisserman, “Synthetic data and artificial neural networks for natural scene text recognition,” *arXiv preprint arXiv:1406.2227*, 2014. [54](#), [76](#), [78](#), [83](#)

## TÀI LIỆU THAM KHẢO

---

- [34] A. Gupta, A. Vedaldi, and A. Zisserman, “Synthetic data for text localisation in natural images,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 2315–2324, 2016. [54](#), [76](#), [78](#), [83](#)
- [35] S. Merity, C. Xiong, J. Bradbury, and R. Socher, “Pointer sentinel mixture models. corr abs/1609.07843 (2016),” *arXiv preprint arXiv:1609.07843*, 2016. [54](#)
- [36] T. Y. Zhang and C. Y. Suen, “A fast parallel algorithm for thinning digital patterns,” *Communications of the ACM*, vol. 27, no. 3, pp. 236–239, 1984. [63](#), [64](#)
- [37] L. Huang, “Skeleton Tracing.” <https://github.com/LingDong-/skeleton-tracing>, 2020. [64](#)
- [38] T. Guan, W. Shen, X. Yang, Q. Feng, Z. Jiang, and X. Yang, “Self-supervised character-to-character distillation for text recognition,” pp. 19473–19484, October 2023. [67](#)