ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN
KHOA KHOA HỌC MÁY TÍNH

BÁO CÁO ĐỒ ÁN
SCENE TEXT DETECTION SỬ DỤNG TEXTFUSENET
CS114 – MÁY HỌC

Giảng viên hướng dẫn: PGS. TS. Lê Đình Duy
ThS. Phạm Nguyễn Trường An

Nhóm sinh viên thực hiện: Lớp CS114.N11.KHCL

STT	MSSV	Họ và tên
1	20520278	Phạm Hoàng Phúc
2	20521663	Nguyễn Đặng Bảo Ngọc
3	20521446	Huỳnh Nguyễn Vân Khánh

TP. HỒ CHÍ MINH, 02/2023

MỤC LỤC

LỜI MỞ ĐẦU

Hiện nay, việc ứng dụng Máy học và một số công nghệ nhận diện đang được sử dụng rộng rãi trong rất nhiều ngành, đem lại hiệu quả công việc cao, tiết kiệm thời gian làm việc và giảm bớt nhân sự. Trong đó, không thể không kể đến là ngành ngân hàng và giao thông là hai ngành nghề có ứng dụng phổ biến nhất về công nghệ nhận diện.

Các luồng công việc kinh doanh nói chung và ngân hàng nói riêng sẽ bao gồm việc nhận thông tin từ các phương tiện truyền thông dạng bản in. Các biểu mẫu, hóa đơn dạng giấy, bản quét tài liệu pháp lý và bản in hợp đồng đều là một phần trong quy trình kinh doanh. Khối lượng giấy tờ lớn như vậy làm mất rất nhiều thời gian và khó để lưu trữ, quản lý. Mặc dù quản lý tài liệu không cần giấy tờ là cách thức hiệu quả nhất, nhưng việc quét tài liệu thành hình ảnh sẽ tạo ra nhiều thách thức. Quá trình này đòi hỏi phải có thao tác can thiệp thủ công, hơn nữa việc số hóa nội dung tài liệu này sẽ tạo ra các tệp hình ảnh với văn bản ẩn bên trong. Vấn đề này được giải quyết bằng cách chuyển đổi hình ảnh văn bản thành dữ liệu văn bản. Sau đó, có thể sử dụng dữ liệu để tiến hành phân tích, hợp lý hóa hoạt động, tự động hóa các quy trình và cải thiện năng suất.

Đối với lĩnh vực giao thông, ở các thành phố lớn số lượng phương tiện giao thông ngày càng tăng dẫn đến việc khó kiểm soát các tình trạng như: tai nạn, ùn tắc, vi phạm luật lệ giao thông. Thị giác máy tính đã tạo ra các ứng dụng quản lý giao thông, giúp làm giảm thiểu các vấn đề trên. Thông qua hệ thống camera giám sát phần mềm nhận dạng phát hiện và xử lý các hành vi vi phạm an toàn giao thông. Trong giao thông, công nghệ nhận dạng có tác dụng nhận diện biển số xe có hành vi vi phạm giao thông như vượt đèn đỏ, vượt quá tốc độ… từ hình ảnh của hệ thống camera giám sát sau đó lấy thông tin của chủ phương tiện rồi tiến hành xử lý vi phạm. Ngoài ra, công nghệ này còn hỗ trợ lưu trữ các hình ảnh, tìm hiểu diễn biến, nguyên nhân các vụ tai nạn.

Có thể thấy công nghệ nhận diện ngày càng trở nên phổ biến bởi những tiện ích mà nó mang lại cho cả các cơ quan tổ chức cũng như người dùng trong nhiều lĩnh vực đời sống của con người. Trong tương lai không xa, các phần mềm nhận diện sẽ được tích hợp với rất nhiều công nghệ tiên tiến khác để tạo ra những sản phẩm đột phá phục vụ nhiều hoạt động của con người.

CHƯƠNG I. GIỚI THIỆU TỔNG QUAN

1. Đặt vấn đề

Hiện nay Máy học nói chung và nhận diện ký tự quang học (OCR) nói riêng đã và đang phát triển rất mạnh, được ứng dụng không chỉ trong lĩnh vực ngân hàng và giao thông mà còn được áp dụng rộng rãi trong các lĩnh vực chính trị, y tế, giáo dục và hỗ trợ các doanh nghiệp phát triển sản phẩm, dịch vụ trong quá trình sản xuất. Việc nhận dạng các ký tự trên nền trắng như quét mã vạch, scan bài báo đã đạt được độ chính xác cao do không có nhiễu của cảnh nền, các ký tự không bị méo và được xếp thẳng hàng.

Tuy nhiên, việc nhận dạng văn bản trên cảnh nền phức tạp, bố cục văn bản xô lệch, văn bản ở nhiều font chữ khác nhau, ánh sáng trong ảnh không đồng đều, độ phân giải ảnh thấp và nội dung trình bày trong nhiều ngôn ngữ là các thách thức lớn.

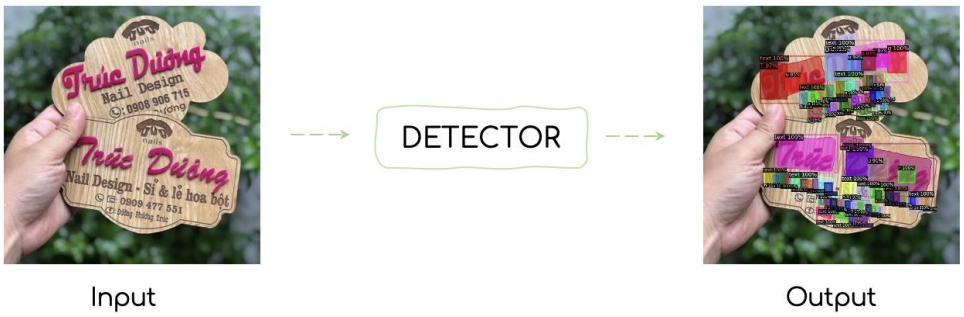
Để giải quyết các thách thức đó, nhóm chúng em sẽ sử dụng framework *TextFuseNet* với bộ dữ liệu *VinText* để nhận dạng và xác định vị trí của các văn bản, có thể thấy đây là một bước đệm trong việc nhận diện ký tự. *Text Dectection* tuy là một bài toán nhỏ trong việc nhận diện ký tự nhưng lại đóng vai trò cực kỳ quan trọng, giúp cho việc nhận diện các ký tự trở nên dễ dàng hơn.

2. Scene Text Detection

2.1. Khái niệm

Phát hiện văn bản (*Text Detection*) là kỹ thuật phát hiện văn bản trong **Scene Text** – ảnh mà có text xuất hiện ở dạng tự nhiên – text là thành phần có sẵn trong ảnh và sau đó bao bọc nó bằng một hộp giới hạn (*bounding box*) hình tứ giác.

2.2. Input và output



Hình 1. Input và output của bài toán

2.2.1. Input

Một bức ảnh với định dạng .jpg.

Số lượng và khoảng cách giữa các ký tự là không quan trọng.

2.2.2. Output

Toạ độ bốn góc của 1 boundingbox đánh dấu vị trí của văn bản trong ảnh.

Bốn điểm này nằm trên một mặt phẳng có gốc tọa độ là góc trái phía trên của ảnh, trục hoành là cạnh trên của ảnh hướng từ trái qua phải, trục tung là cạnh bên trái của ảnh hướng từ trên xuống dưới. Các giá trị trên trục hoành, trục tung đều là các giá trị không âm.

2.2.3. Mục đích và ứng dụng

Mục đích: xác định vùng chứa ký tự trong ảnh.

Ứng dụng: *Scene Text Detection* là một công nghệ nhận dạng văn bản được sử dụng để nhận diện và phân tích các ký tự trên ảnh chụp từ môi trường thực tế (có thể là từ các bức ảnh chụp trong đời sống hàng ngày, các bức ảnh trên mạng, hoặc các ảnh từ camera giám sát). Các ứng dụng của *Scene Text Detection* gồm:

- *Trích xuất thông tin từ ảnh*: *Scene Text Detection* giúp hỗ trợ phát hiện vùng có văn bản từ đó có thể giúp tự động trích xuất thông tin từ các tài liệu quan trọng

như chứng minh thư, hộ chiếu, giấy tờ tùy thân và các tài liệu y tế. Thay vì phải nhập liệu thủ công, kỹ thuật này giúp tiết kiệm thời gian và tăng tính chính xác.

- *Tăng trải nghiệm người dùng*: *Scene Text Detection* có thể được sử dụng để giúp người dùng có trải nghiệm tốt hơn trên các ứng dụng di động và trang web. Ví dụ, khi người dùng chụp một tấm hình, kỹ thuật này có thể giúp tự động phát hiện vùng có văn bản và trích xuất nó thành nội dung có thể tìm kiếm.
- *Nhận dạng biển số xe*: *Scene Text Detection* có thể được sử dụng để phát hiện và nhận dạng biển số xe từ camera giám sát. Điều này có thể giúp cho việc giám sát giao thông và giảm thiểu vi phạm luật giao thông.
- *Giúp cho người khiếm thị*: *Scene Text Detection* có thể được sử dụng để đọc văn bản cho những người khiếm thị. Khi văn bản được phát hiện, nó có thể được đọc to lên hoặc được chuyển thành văn bản Braille.
- *Tìm kiếm hình ảnh*: *Scene Text Detection* có thể được sử dụng để tìm kiếm hình ảnh trên mạng với từ khóa được tìm thấy trong văn bản trên hình ảnh. Điều này có thể giúp cải thiện trải nghiệm tìm kiếm hình ảnh và tăng tính khả dụng của các hình ảnh trên mạng.

2.3. Một số phương pháp cơ bản

Scene Text Detection là một lĩnh vực nghiên cứu khá phức tạp trong Máy học nói chung và Thị giác máy tính nói riêng. Dưới đây là một số phương pháp cơ bản được sử dụng trong *Scene Text Detection*:

- *Phân vùng vật thể*: phương pháp này chia ảnh thành các vùng khác nhau để phân tích, từ đó phát hiện văn bản. Nó có thể được thực hiện bằng cách sử dụng các thuật toán như GrabCut hoặc Mask R-CNN.
- *Kết hợp các phương pháp trích xuất đặc trưng*: Phương pháp này kết hợp các phương pháp trích xuất đặc trưng như HOG (*Histogram of Oriented Gradients*) và LBP (*Local Binary Patterns*) để xác định vị trí của văn bản trên ảnh.
- *Sử dụng mô hình Deep Learning*: Sử dụng các mô hình Deep Learning như Convolutional Neural Networks (*CNNs*) đã cho thấy hiệu quả cao trong việc phát hiện văn bản trong cảnh. Ví dụ như Faster R-CNN, YOLO (*You Only Look Once*) và SSD (*Single Shot Detector*).

- *Sử dụng phương pháp tiền xử lý*: Phương pháp này sử dụng các kỹ thuật tiền xử lý, chẳng hạn như cân bằng sáng, làm sạch nhiễu và làm mịn ảnh, để cải thiện độ chính xác của quá trình phát hiện văn bản.
- *Sử dụng kết hợp nhiều phương pháp*: Nhiều phương pháp khác nhau có thể được kết hợp để tăng độ chính xác và hiệu suất của quá trình phát hiện văn bản, chẳng hạn như kết hợp giữa phân vùng vật thể và mô hình Deep Learning.

3. Mục tiêu đề tài

Mục tiêu chính của *Scene Text Detection* là tìm kiếm, phát hiện và xác định vị trí của văn bản trong ảnh chứa các cảnh phức tạp và đa dạng. Điều này có thể giúp cho máy tính có thể xử lý và hiểu được thông tin văn bản trong các cảnh phức tạp và đa dạng của thế giới thực, có thể đọc và hiểu được thông tin trong các tài liệu văn bản, báo cáo, biển báo, nhãn hiệu và các thông tin khác, từ đó giúp cải thiện các ứng dụng xử lý ảnh và trí tuệ nhân tạo trong nhiều lĩnh vực khác nhau. Một số mục tiêu cụ thể của *Scene Text Detection* là:

- Phát hiện văn bản trong các bức ảnh chụp từ smartphone, máy ảnh hoặc camera an ninh.
- Xác định vị trí của văn bản trên các bảng hiệu, nhãn hàng, bảng thông báo, bảng đen và các bộ phận khác của hệ thống giao thông.
- Nhận dạng các ký tự và chữ viết tay trên các vật phẩm và bề mặt khác nhau, chẳng hạn như thư tay, giấy tờ tùy thân, băng chuyền sản xuất, bánh xe lăn, và nhiều hơn nữa.
- Xử lý và nhận dạng dữ liệu văn bản trong các tài liệu văn bản kỹ thuật, báo cáo và chứng chỉ.
- Phân tích và đánh giá các văn bản từ các camera giám sát để giám sát hoạt động, tìm kiếm thông tin, và truy xuất dữ liệu cho các mục đích pháp lý và an ninh.

CHƯƠNG II. TEXTFUSENET

1. Giới thiệu

TextFuseNet là một framework được sử dụng cho bài toán *Scene Text Detection* được giới thiệu vào năm 2020 tại hội nghị IJCAI-20 (*International Joint Conference on Artificial Intelligence*), bởi nhóm tác giả đến từ Đại học Vũ Hán (*Wuhan University*), Trung Quốc và Đại học Sydney (*The University of Sydney),* Úc.

TextFuseNet giải quyết các vấn đề khó trong bài toán *Scene Text Detection* bằng cách sử dụng các phương pháp kết hợp đặc trưng (*feature fusion*) để cải thiện độ chính xác. Không giống với các phương pháp trước đó, *TextFuseNet* khai thác các đặc trưng ở nhiều mức độ khác nhau (mức toàn cục, mức từ và mức chữ cái) để bổ sung thêm thông tin cho quá trình dự đoán. *TextFuseNet* thu thập các đặc trưng ở nhiều mức và sử dụng multi-path fusion architecture (tạm dịch: khối kết hợp đa luồng) để kết hợp các đặc trưng này với nhau.

Phương pháp này gồm có 5 thành phần chính: (1) một *feature pyramid network* (FPN) để trích xuất đặc trưng ở các mức độ khác nhau, (2) *region proposal network* (RPN) để dự đoán những vùng ảnh có thể chứa văn bản, (3) một nhánh *sematic segmentation* để khai thác thông tin trên toàn cục của ảnh, (4) một nhánh *detection* để phát hiện từ và chữ cái trong ảnh, (5) một nhánh *mask segmentation* để thực hiện *instance segmentation* cho các từ và chữ cái trong ảnh.

2. Phương pháp

2.1. Một số khái niệm được đề cập đến

2.1.1. Feature map

Feature map là một ma trận 2 chiều (hoặc 3 chiều trong trường hợp ảnh màu) được tạo ra bởi các bộ lọc (*filters*) qua các lớp tích chập của convolutional *neural* network (*CNN*). Các feature map này giúp phân tích hình ảnh bằng cách tìm kiếm các đặc trưng khác nhau của ảnh, chẳng hạn như các cạnh, góc, texture, hoặc các đối tượng. Feature map thường có kích thước nhỏ hơn ảnh gốc, giúp giảm số lượng thông tin và làm cho

việc xử lý ảnh trở nên nhanh hơn. *Feature map* được đưa vào các lớp tiếp theo của mạng nơ-ron để giúp nhận diện đối tượng hoặc phân loại ảnh.

2.1.2. Anchor box

Anchor box là một phần quan trọng của các thuật toán *Object Detection* trong *Computer Vision*. Nó được sử dụng để dự đoán vị trí và kích thước của đối tượng trong một hình ảnh. *Anchor box* là một hình chữ nhật (hoặc hình vuông) được xác định trước đó với các kích thước và tỷ lệ khác nhau. Mỗi *anchor box* đại diện cho một loại đối tượng khác nhau, ví dụ như một *anchor box* có kích thước lớn hơn thường sẽ được sử dụng để phát hiện các đối tượng lớn hơn. Trong quá trình huấn luyện, thuật toán *Object Detection* sẽ tìm cách phân loại các *anchor box* và dự đoán vị trí và kích thước của đối tượng trong hình ảnh. Thuật toán sẽ sử dụng các giá trị phân loại và dự đoán này để xác định đối tượng nào xuất hiện trong hình ảnh và đưa ra kết quả phát hiện đối tượng. Việc sử dụng *anchor box* giúp tăng độ chính xác của các thuật toán *Object Detection* và giúp chúng có thể phát hiện được các đối tượng có kích thước và tỷ lệ khác nhau trong hình ảnh.

Anchor box được sử dụng trong các thuật toán *Scene Text Detection* để phát hiện và nhận diện văn bản trong các hình ảnh. Trong bài toán này, *anchor box* được sử dụng để dự đoán vị trí và kích thước của các ký tự, từ và dòng văn bản trong hình ảnh. Việc sử dụng *anchor box* giúp tăng độ chính xác của các thuật toán *Scene Text Detection* và giúp chúng có thể phát hiện được các ký tự và từ có kích thước và tỷ lệ khác nhau trong hình ảnh.

2.1.3. Sliding window

Sliding window là một kỹ thuật được sử dụng trong *Computer Vision* để phân tích các đặc trưng trong một hình ảnh. Kỹ thuật này hoạt động bằng cách di chuyển một cửa sổ (window) qua từng vùng của hình ảnh và tính toán các đặc trưng trong cửa sổ đó. Cửa sổ có thể có kích thước và hình dạng khác nhau, tùy thuộc vào nhiệm vụ mà chúng ta muốn thực hiện.

2.1.4. Backbone network

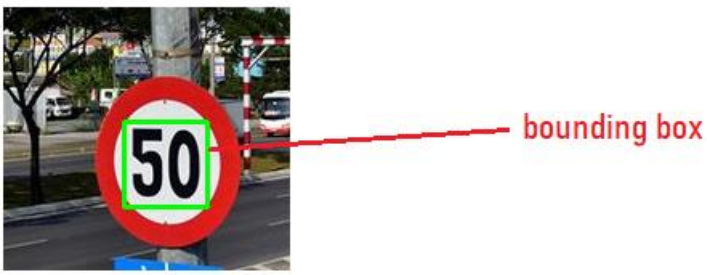
Backbone network là một thành phần quan trọng trong các mô hình *Deep Learning*, đặc biệt là các mô hình được sử dụng trong lĩnh vực *Computer Vision*. *Backbone network* thường được sử dụng để trích xuất các đặc trưng từ dữ liệu đầu vào để tạo ra một biểu diễn ẩn cho hình ảnh, video hoặc âm thanh.

Backbone network thường được xây dựng dựa trên kiến trúc mạng *neural network* sâu (*Deep Neural network*), bao gồm nhiều lớp tích chập (*convolutional layers*), các lớp kích hoạt phi tuyến (*non-linear activation layers*) và các lớp *pooling* để giảm kích thước của đầu ra. Các lớp này sẽ giúp trích xuất các đặc trưng cấp cao và cấp thấp từ dữ liệu đầu vào và tạo ra một biểu diễn dày đặc của hình ảnh hoặc video.

Các *backbone network* phổ biến được sử dụng trong các mô hình *Deep Learning* bao gồm *VGG*, *ResNet*, *Inception*, và *MobileNet*. Các mô hình đó thường sử dụng các *pre-trained backbone network* đã được huấn luyện trên các tập dữ liệu lớn, giúp cải thiện độ chính xác và tốc độ huấn luyện của các mô hình mới.

2.1.5. Bounding box

Bounding box là một khái niệm trong *Computer Vision* để xác định vị trí và kích thước của một đối tượng trong hình ảnh. Nó là một hình chữ nhật được xác định bởi tọa độ của hai điểm trên và dưới (hoặc trái và phải) của hình ảnh, thường là các tọa độ góc trên bên trái và góc dưới bên phải.



Hình 2. Ví dụ về một bounding box trong ảnh

Bounding box được định nghĩa bằng cách sử dụng các thông số như tọa độ x, y của điểm góc trên bên trái và chiều rộng, chiều cao của hình chữ nhật. Tùy thuộc vào nhiệm vụ cụ thể, *bounding box* có thể được tạo ra bằng cách sử dụng các kỹ thuật như *sliding window* hoặc *anchor box* để phát hiện các đối tượng trong hình ảnh.

Bounding box thường được sử dụng trong các nhiệm vụ như *Object Detection, Instance Segmentation,* và *Face Recognition.*

2.1.6. RoIAlign

RoIAlign là một phương pháp được sử dụng trong các tác vụ *Object Detection* và *Instance segmentation* để trích xuất đặc trưng từ khu vực quan tâm (RoIs) trên một hình ảnh. *RoIAlign* giải quyết vấn đề lỗi định lượng được trong phương pháp *RoI pooling*, mà trước đây được sử dụng để trích xuất đặc trưng trong RoIs.

Trong *RoI pooling*, một RoI được chia thành một số ô (bin) cố định và giá trị lớn nhất trong mỗi ô được trích xuất. Phương pháp này dẫn đến mất thông tin về không gian do lỗi định lượng, điều này có thể dẫn đến sự không khớp không gian giữa RoI và *feature map*.

RoIAlign sử dụng phép nội suy song tuyến tính để tính toán các giá trị chính xác của các đặc trưng đầu vào tại bốn vị trí được lấy mẫu thường xuyên trong mỗi ô RoI và kết quả sau đó được tổng hợp (sử dụng giá trị lớn nhất hoặc trung bình). Điều này đảm bảo rằng các đặc trưng RoI được căn chỉnh chính xác với *feature map* bản và giảm thiểu vấn đề về lệch không gian.

Cách thức hoạt động *RoIAlign* có thể được tóm tắt như sau:

- Chia RoI thành một số ô cố định.
- Đối với mỗi ô, tính vị trí chính xác của bốn vị trí được lấy mẫu thường xuyên trong ô đó.
- Sử dụng nội suy song tuyến tính để thu được giá trị đặc trưng tại bốn vị trí trong mỗi ô.
- Tổng hợp bốn giá trị đặc trưng để thu được một giá trị đặc trưng duy nhất cho mỗi ô.

RoIAalign chủ yếu được sử dụng để trích xuất các đặc trưng từ RoI mà không làm mất thông tin không gian. Tuy nhiên, nó cũng có thể được sử dụng để thay đổi kích thước của *feature map,* trong khi vẫn duy trì việc trích xuất đặc trưng một cách chính xác. Để thay đổi kích thước của *feature map* bằng cách sử dụng *RoIAalign*, người ta có

thể sửa đổi kích thước của các RoI được chuyển qua *RoIAalign*. Cụ thể, việc giảm kích thước của RoI sẽ dẫn đến *feature map* đầu ra nhỏ hơn, trong khi việc tăng kích thước của RoI sẽ dẫn đến *feature map* đầu ra lớn hơn.

2.2. FPN (Feature Pyramid Network)

Feature Pyramid Network (FPN) là một mạng *neural* được thiết kế để trích xuất thông tin từ các ảnh với nhiều kích thước. Mô hình này được giới thiệu trong bài báo "*Feature Pyramid Networks for Object Detection*" của Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, và Serge Belongie trong năm 2017. *Feature Pyramid Network* (FPN) đã có một ảnh hưởng lớn đến lĩnh vực thị giác máy tính (Computer Vision) bởi vì nó giải quyết được vấn đề quan trọng của việc phát hiện đối tượng trong ảnh. Một trong những thách thức lớn trong quá trình phát hiện đối tượng là làm thế nào để phát hiện đối tượng có độ chính xác cao và đồng thời xử lý ảnh với nhiều kích thước khác nhau. Khi ảnh có độ phân giải cao, thông tin chi tiết về đối tượng có thể bị mất đi trong quá trình tính toán, trong khi khi ảnh có độ phân giải thấp, thông tin về đối tượng có thể không đủ để phát hiện đối tượng một cách chính xác.

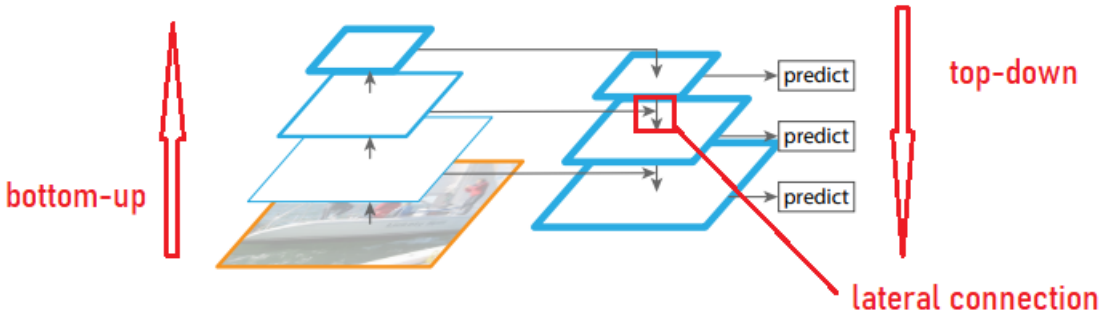
Kiến trúc FPN dựa trên một tháp gồm các *feature map*, trong đó mỗi cấp độ của tháp tương ứng với một tỷ lệ khác nhau của hình ảnh đầu vào. FPN được thiết kế để trích xuất các đặc trưng phong phú và có ý nghĩa từ các hình ảnh với độ phân giải và tỷ lệ khác nhau, đồng thời kết hợp các đặc trưng này thành một *feature map* thống nhất để sử dụng cho các tác vụ thị giác máy tính khác nhau, chẳng hạn như phát hiện đối tượng (*Object Detection*), phân đoạn cá thể (*instance segmentation*) và phân đoạn ngữ nghĩa (*sematic segmentation*).

FPN lấy một hình ảnh có kích thước tùy ý làm đầu vào và xuất ra các *feature map* có kích thước tương ứng ở nhiều mức độ. Quá trình trích xuất các *feature maps* được thực hiện dựa trên kiến trúc của backbone network. Tuy nhiên, việc trích xuất các *feature maps* trong FPN là một quá trình độc lập với kiến trúc của *backbone*, có nghĩa là việc sử dụng một *backbone network* cụ thể không ảnh hưởng đến quá trình trích xuất các *feature maps* trong FPN.

Cấu trúc của FPN bao gồm một luồng *bottom-up*, một luồng *top-down* và một kết nối bên (*lateral connection*).

Luồng Bottom-up là một mạng CNN để thực hiện trích xuất đặc trưng. Càng lên cao, độ phân giải (resolution) càng giảm (kích thước của feature maps giảm), và giá trị thông tin về ngữ cảnh càng cao (*semantic value*). Các *feature map* sẽ được xuất ra tại một số lớp trong mạng *backbone*, những lớp như vậy được gọi là *stage*.

Luồng Top-down và lateral connection: FPN xây dựng thêm luồng *top-down*, nhằm mục đích xây dựng các *feature map* có độ phân giải cao từ các *feature map* có ngữ nghĩa cao bằng cách lấy mẫu (upsampling). Các feature map này sau đó sẽ được tăng cường qua các *lateral connection*. Mỗi *lateral connection* sẽ gộp các *feature map* từ luồng *bottom-up* và luồng *top-down* (các feature map này có cùng kích thước) bằng cách sử dụng phép cộng *element-wise* (tức là cộng từng phần tử tương ứng của hai feature map). Việc sử dụng *lateral connection* là nhằm để cải thiện độ phân giải của các *feature map* và đồng thời giúp các *feature map* có độ bao quát rộng hơn và bao gồm các vật thể có kích thước khác nhau. Điều này giúp cải thiện độ chính xác của việc phát hiện vật thể trong ảnh.



Hình 3. Kiến trúc của Feature Pyramid Network

Tóm lại, cấu trúc của FPN bao gồm một luồng *bottom-up* để trích xuất các *feature maps*, một luồng *top-down* để tạo ra các *feature maps* ở các tỉ lệ khác nhau và kết nối bên để kết hợp các *feature maps* ở các tỉ lệ khác nhau để tạo ra một tháp đặc trưng có độ phân giải cao và thông tin đa tầng.

Output tại mỗi stage trong FPN có thể được sử dụng làm input trong cho các mô hình phân loại, nhận dạng khác. Đối với *TextFuseNet* các output của FPN được sử dụng như input cho nhánh *sematic segmentation* và RPN.

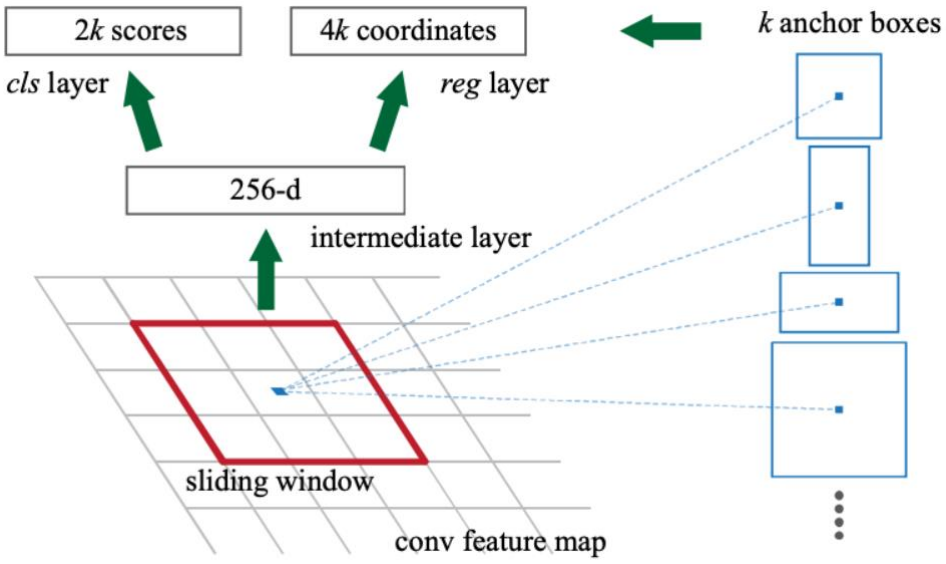
2.3. RPN (Region Proposal Network)

Region Proposal Network (RPN) là một phần của mô hình *Faster R-CNN* trong lĩnh vực *Computer Vision*, được sử dụng để tìm kiếm các khu vực tiềm năng chứa đối tượng được quan tâm trong ảnh. Mô hình này được giới thiệu lần đầu tiên trong bài báo *"Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks"* của Shaoqing Ren, Kaiming He, Ross Girshick và Jian Sun, được công bố tại Hội nghị *IEEE International Conference on Computer Vision (ICCV)* năm 2015. Sự ra đời của mô hình đã gây được sự chú ý lớn trong cộng đồng *Computer Vision*. Trước khi RPN được giới thiệu, các phương pháp tiền đề khác thường sử dụng các giải thuật như *Selective Search* hoặc *Edge Boxes* để tạo ra các đề xuất vùng quan tâm. Tuy nhiên, các phương pháp đó thường rất chậm và tốn nhiều tài nguyên tính toán. RPN đã giúp giải quyết vấn đề này bằng cách tích hợp quá trình tạo ra đề xuất vùng quan tâm trực tiếp vào mạng *neural*, tạo ra một mô hình *end-to-end* cho bài toán phát hiện đối tượng. Kết quả đạt được bởi RPN là nhanh hơn và chính xác hơn so với các phương pháp tiền đề trước đó. Do đó, RPN đã trở thành một phần quan trọng của nhiều mô hình phát hiện đối tượng hiện đại và được sử dụng rộng rãi trong cộng đồng *Computer Vision.*

RPN sử dụng một mạng *neural* để đưa ra các đề xuất vùng quan tâm (Region of Interest - RoI) trong một hình ảnh.

Đầu tiên *feature map* được trích xuất từ ảnh đầu vào bằng cách sử dụng một mạng *neural* tiền xử lý, thường là mạng *convolutional neural network* (CNN), để áp dụng phép tích chập (convolution) lên ảnh đầu vào và trích xuất các đặc trưng (features) của ảnh. Mạng CNN này thường được huấn luyện trước trên tập dữ liệu lớn, chẳng hạn như tập dữ liệu *ImageNet*, để học cách trích xuất các đặc trưng một cách hiệu quả từ ảnh. Ảnh đầu vào được truyền qua mạng CNN để tạo ra feature map. *Feature map* được tạo ra từ mạng CNN là một tensor 4 chiều với kích thước *(W, H, C, D)*, trong đó W và H là chiều rộng và chiều cao của feature map, C là số kênh và D là độ sâu (depth) của *feature*

map. *Feature map* này sẽ được sử dụng bởi RPN để tạo ra các đề xuất vùng quan tâm (RoI). RPN thực hiện điều này bằng cách sử dụng các *sliding window* với nhiều kích thước và tỷ lệ khác nhau trên một *feature map* đã được trích xuất từ ảnh đầu vào.



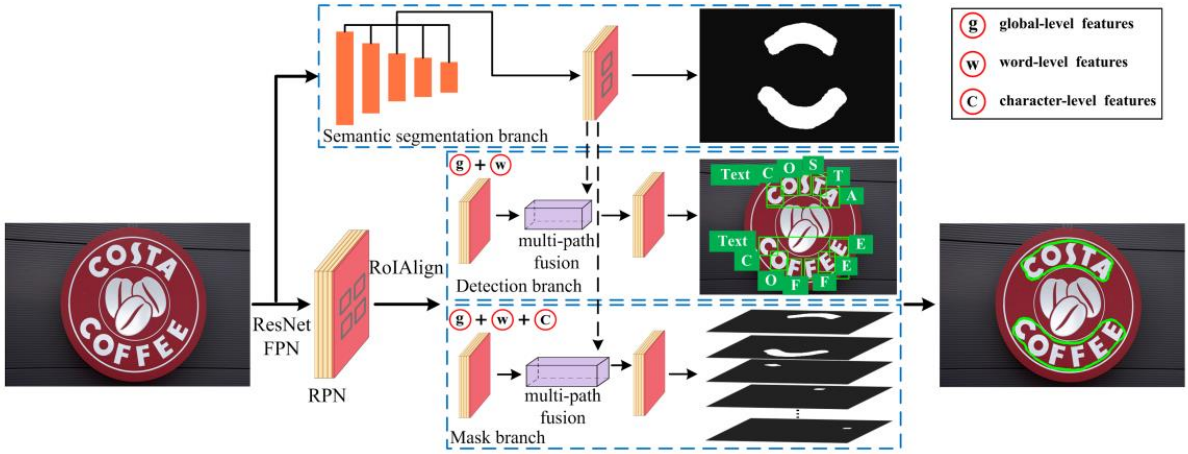
Hình 4. Region Proposal Network

Tiếp theo, RPN sử dụng một mạng *neural* để ước tính xác suất của mỗi *sliding window* có chứa đối tượng hay không, cũng như ước tính hộp giới hạn (bounding box) của mỗi đối tượng. Cụ thể, RPN ước tính xác suất của mỗi *sliding window* có chứa đối tượng hay không bằng cách sử dụng một mạng *neural* nhỏ được gọi là *"Objectness Classifier"*. Đầu vào của *objectness classifier* là *feature map* được trích xuất từ ảnh đầu vào và đầu ra là một tensor 3 chiều có kích thước (W, H, 2B), trong đó B là số *anchor boxes* được sử dụng. Mỗi phần tử của *tensor* này đại diện cho xác suất của *anchor box* tương ứng có chứa đối tượng hay không. Ngoài ra, RPN cũng sử dụng một mạng *neural* khác để ước tính hộp giới hạn (bounding box) của mỗi đối tượng trong anchor box được chọn. Mạng này được gọi là *"bounding box regressor"* và được huấn luyện để ước tính vị trí và kích thước của hộp giới hạn dựa trên thông tin về *anchor box* và đặc trưng của vùng đó trên *feature map*. Đầu vào của *bounding box regressor* là *feature map* và *vector* biểu diễn của *anchor box*, đầu ra là một *tensor* 3 chiều có kích thước (W, H, 4B), trong đó B là số *anchor box* được sử dụng. Mỗi phần tử của *tensor* này đại diện cho giá trị của một thuộc tính của hộp giới hạn (ví dụ như tọa độ x, y, chiều rộng, chiều cao). Các

giá trị này sẽ được sử dụng để cập nhật vị trí và kích thước của *anchor box* để tạo ra các đề xuất vùng quan tâm chính xác hơn cho việc phát hiện đối tượng.

Các vùng quan tâm được xếp hạng dựa trên xác suất và các hộp giới hạn được ước tính. Cuối cùng, các vùng quan tâm tốt nhất được chọn làm đầu vào cho một mạng *neural* khác để phân loại đối tượng và để dự đoán hộp giới hạn chính xác hơn.

2.4. TextFuseNet pipeline



Hình 5. TextFuseNet pipeline

Pipeline của *TextFuseNet* có thể được mô tả như sau:

- ***Bước 1:*** Sử dụng **FPN** để trích xuất các feature map với các mức độ thông tin khác nhau
- ***Bước 2:*** Các *feature map* sẽ được *sao chép* thành hai bản và đi theo hai nhánh riêng biệt, một bản đi theo ***nhánh sematic segmentation***, bản còn lại được truyền vào mô hình ***RPN***
- ***Bước 3:*** Các *feature map* thu được được truyền qua ***nhánh sematic segmentation*** để dự đoán mask cho các văn bản xuất hiện trong ảnh, tại đây chúng ta ***thu được các feature map ở mức toàn cục***, các *feature map* này sẽ được gộp với các *feature map* được trích xuất từ *nhánh detection* và *nhánh mask* bằng cách sử dụng khối *multi-path fusion*
- ***Bước 4:*** Các *feature map (output của FPN)* được ***truyền qua một mô hình RPN để dự đoán những vùng có thể chứa văn bản*** (*text proposal*), sau đó thực hiện

trích xuất đặc trưng sử dụng RoIAlign trên các vùng này và đưa chúng về cùng kích thước. ***Feature map thu được lại tiếp tục được truyền vào hai nhánh riêng biệt*** *(nhánh detection và nhánh mask)**.***

- ***Bước 5**: **Nhánh detection nhận các feature map là output của RPN*** sau khi thực hiện RoIAlign
- ***Bước 6**:* *Feature maps* được đưa qua các lớp tích chập để trích xuất đặc trưng
- ***Bước 7**:* *Feature maps* ở các mức độ ngữ nghĩa khác nhau được gộp lại *bằng* một khối ***multi-path fusion***
- ***Bước 8**:* Các *feature maps* sau khi đã được gộp lại thành một feature map thống nhất sẽ được tiếp tục truyền vào một lớp *fully connected* để phân loại và dự đoán ***bounding box***
- ***Bước 9**:* Tại ***nhánh mask***, ***output của RPN*** cũng được truyền qua các lớp tích chập để trích xuất đặc trưng
- ***Bước 10**:* Các *feature maps* thu được lại được tiếp tục truyền qua khối ***multi-path fusion*** để gộp các đặc trưng ở các mức độ khác nhau thành một feature map duy nhất.
- ***Bước 11**:* Tương tự như ở nhánh detection *feature map* này lại được truyền qua các lớp *fully connected* để phân loại và ***dự đoán các đường contour.***

Tại các khối *multi-path fusion* không chỉ có thông tin được trích xuất trên cùng một nhánh (nhánh detection hoặc nhánh mask) được gộp lại, mà *feature map* thu được từ nhánh *sematic segmention* cũng được truyền đến để gộp chung với các *feature map* khác. Kết quả thu được một *feature map* cuối cũng với các đặc trưng ở mức độ chữ cái (đối với nhánh mask), mức độ từ và mức độ toàn cục.

3. Đặc trưng đa mức (multi-level feature)

Nhìn chung, các đặc trưng ở mức ký tự và mức từ có thể được trích xuất dễ dàng thông qua *nhánh detection* và *nhánh mask* của mô hình. Chúng ta có thể thực hiện điều này bằng cách dò tìm cả từ lẫn chữ cái xuất hiện trong các đề xuất (text proposals). Trong trường hợp này *RoIAlign* được sử dụng để trích xuất các đặc trưng khác nhau và thực hiện dò tìm các từ và ký tự trong ảnh. *TextFuseNet* còn sử dụng thêm *nhánh*

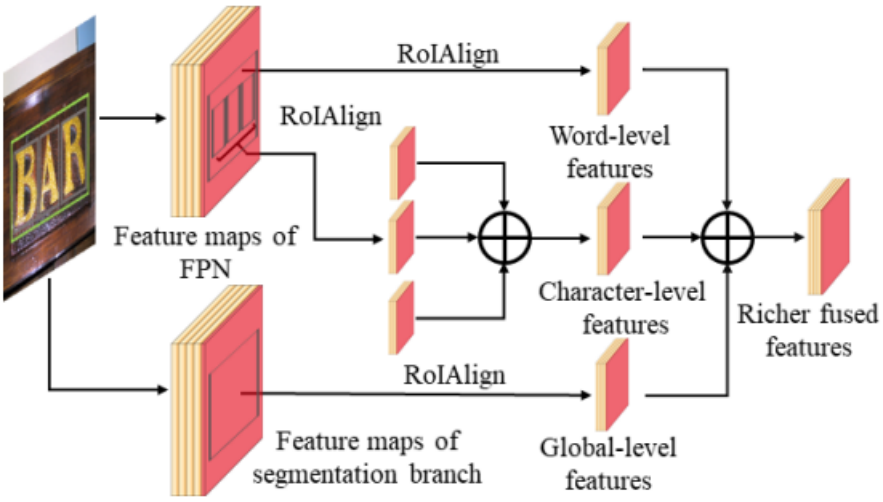
semantic segmentation để trích xuất các đặc trưng ở mức toàn cục (global-level features).

Như được mô tả trong hình 2, *nhánh semantic segmentation* được xây dựng dựa trên output của FPN. Chúng ta gộp các đặc trưng ở tất cả mức độ (toàn cục, từ và chữ cái) thành một thể hiện (representation) thống nhất và thực hiện các thao tác dự đoán trên thể hiện này. Trên thực tế, một lớp tích chập 1x1 được sử dụng để làm cho các feature map có cùng một kích thước, nhằm phục vụ cho quá trình gộp đặc trưng.

4. Multi-path Fusion Architect

Sau khi có được các đặc trưng đa mức, chúng ta sử dụng một khối *multi-path fusion* ở cả *nhánh detection* và *nhánh mask*. Ở *nhánh detection*, dựa trên những vùng văn bản được gợi ý (text proposal) bởi RPN, chúng ta trích xuất đặc trưng ở mức toàn cục (global-level features) và mức từ (word-level features) để thực hiện dò tìm văn bản theo nhiều luồng khác nhau. Chúng ta gộp hai loại đặc trưng với nhau trước khi thực hiện dò tìm văn bản (dò tìm cả từ và chữ cái). Trên thực tế, với một vùng văn bản được gợi ý, *RoIAlign* được sử dụng trên các output của FPN để trích xuất đặc trưng ở mức độ từ và mức độ toàn cục với kích thước 7x7. Chúng ta gộp các đặc trưng này với nhau thông qua một phép cộng *element-wise*, kết quả thu được sẽ tiếp tục được đưa qua một lớp tích chập 3x3 và một lớp tích chập 1x1. Đặc trưng sau khi gộp sẽ được sử dụng cho quá trình phân loại và xác định hộp giới hạn (*bounding box*).

Ở *nhánh mask*, với mỗi trường hợp ở mức từ chúng ta có gộp các đặc trưng ở mức chữ cái, mức từ và mức toàn cục với nhau sử dụng một kiến trúc *multi-path fusion* đặc biệt dành riêng cho tác vụ *instance segmentation*. Ở *nhánh mask*, với mỗi trường hợp ở mức từ, chúng ta gộp các đặc trưng ở mức chữ cái, mức từ và mức toàn cục với nhau sử dụng một kiến trúc *multi-path fusion* đặc biệt dành riêng cho tác vụ *instance segmentation*. Hình 3 là minh họa cho kiến trúc *multi-path fusion* được sử dụng. Trong đó, chúng ta trích xuất các đặc trưng đa mức từ nhiều luồng khác nhau và gộp chúng lại để có được đặc trưng mới, việc này giúp các đặc trưng được thể hiện tốt hơn, với nhiều thông tin hơn.



Hình 6. Kiến trúc multi-path fusion được sử dụng ở nhánh mask

Với một từ đầu vào ký hiệu là r_i, đầu tiên chúng ta xác định tập các ký tự C_i thuộc về vùng quan tâm đó bằng cách dựa trên tỉ lệ giao giữa ký tự đó với vùng chứa ký tự, nghĩa là tỉ lệ này sẽ bằng 1 nếu khung chứa từ cũng chứa toàn bộ khung chứa ký tự, và bằng 0 nếu ngược lại. Chúng ta sử dụng ký hiệu c_j để chỉ từng ký tự trong tập ký tự C_i. Các ký tự trong tập C_i có thể được xác định theo công thức sau:

$$C_i = \{c_i | \frac{b_i \cap b_j}{b_j} > T\}$$

Trong đó, b_i và b_j lần lượt là *bounding box* của từ r_i và chữ cái c_j, T là một giá trị ngưỡng có giá trị nằm trong khoảng $0 < T \leq 1$.

Bởi vì số lượng ký tự của mỗi từ là không cố định, một từ r_i có thể chứa từ không cho đến hàng trăm ký tự, chúng ta sẽ gộp các đặc trưng của các từ trong tập các chữ cái C_i thành một *feature map* thống nhất. Cụ thể, đầu tiên *RoIAlign* được sử dụng để trích xuất các đặc trưng với kích thước 14x14 tương ứng với mỗi ký tự trong tập C_i, sau đó các *feature maps* này sẽ được gộp chung sử dụng phép cộng *element-wise*. Kết quả của phép cộng *element-wise* sau đó lại được đưa qua một lớp tích chập 3x3 và một lớp tích chập 1x1, cuối cùng chúng ta sẽ thu được các đặc trưng ở mức chữ cái.

Bằng cách tiếp tục sử dụng *RoIAlign* để trích xuất các đặc trưng về từ và các đặc trưng toàn cục tương ứng, chúng ta gộp tất cả các *features* ở ba mức độ (toàn cục, từ và

chữ cái) với nhau bằng phép cộng *element-wise*, truyền kết quả thu được qua một lớp tích chập 3x3 và một lớp tích chập 1x1. Các đặc trưng cộng gộp thu được sau khi thực hiện các thao tác trên sẽ được sử dụng để thực hiện *instance segmentation*. Lưu ý rằng lợp tích chập 3x3 và lớp tích chập 1x1 được sử dụng nhằm mục đích kết nối các khác biệt về ngữ nghĩa (semantic gap) giữ các đặc trưng khác nhau.

Công thức đánh giá tổng quan (Overall objective):

$$L = L_{rpn} + L_{mask} + L_{seg} + L_{det}$$

Trong đó, $L_{rpn}, L_{mask}, L_{seg}$ và L_{det} lần lượt là hàm los của RPN, *nhánh mask*, *nhánh sematic segmetation* và *nhánh detection*.

CHƯƠNG III. DATASET

1. Bộ dữ liệu VinText

1.1. Giới thiệu

Trong phần này, nhóm sẽ giới thiệu về bộ dữ liệu VinText – bộ dữ liệu cho ***Vietnamese scene text***. Đây là bộ dữ liệu được công bố vào năm 2021 bởi nhóm tác giả đến từ VinAI Research, Trường Đại học Công nghệ Thông tin (UIT), Trường Đại học Khoa học Tự nhiên (US) và các trường Đại học khác (VNU, Oregon và Stony Brook).

Bộ dữ liệu này bao gồm 2000 ảnh được gán nhãn đầy đủ *(fully annotated)* với 56,084 trường hợp văn bản *(text instances)* khác nhau. Mỗi trường hợp được khoanh vùng bởi một *bounding box* hình tứ giác *(quadrilateral)* và được gán với *ground truth* là dãy ký tự tương ứng. Bộ dữ liệu VinText được nhóm tác giả phân chia ngẫu nhiên thành ba tập riêng biệt là: tập *training* (1200 ảnh), tập *validation* (300 ảnh) và tập *testing* (500 ảnh).

Tập dữ liệu VinText bao gồm những cảnh có chứa dòng người bận rộn và đông đúc cùng với nhiều biển hiệu cửa hàng, biển quảng cáo và biển tuyên truyền khác nhau. Chi tiết về các hình ảnh và các nhãn tương ứng sẽ được mô tả ở phần dưới đây.

1.2. Mô tả chi tiết

Các hình ảnh trong bộ dữ liệu được tải xuống từ *Internet* hoặc *được chụp* bởi chính nhóm tác giả. Mục đích của việc thu nhập bộ dữ liệu này là có được một tập hợp đa dạng về các *scene text* thường gặp trong cuộc sống thường ngày ở Việt Nam.

Để đảm bảo tính đa dạng của tập dữ liệu, nó được chia ra thành các nhóm chính *(categories)* và các nhóm nhỏ hơn bên trong đó *(subcategories)*. Danh sách các nhóm chính *(at the first level)* là: *shop signs* (biển hiệu cửa hàng), *notice boards* (bảng thông báo), *bulletins* (bản tin), *banners* (biểu ngữ), *flyers* (tờ rơi), *streetwalls* (tường), *vehicles* (xe cộ) và *miscellaneous items* (các vật dụng linh tinh).

Các nhóm chính này lại được chia thành các nhóm con *(subcategories)* và nhiều nhóm con lại được chia thành các nhóm nhỏ hơn nữa *(sub-subcategories)*. ***Ví dụ:*** nhóm

chính *"miscellaneous items"* chứa các nhóm con như *book covers* (bìa sách), *product labels* (nhãn sản phẩm), *clothes* (quần áo).

Hình 7. Một bức ảnh trong bộ dữ liệu huấn luyện với hộp giới hạn (bounding box) tương ứng

Hình 8. Vị trí của văn bản trong ảnh dựa trên tọa độ của các điểm Bezier

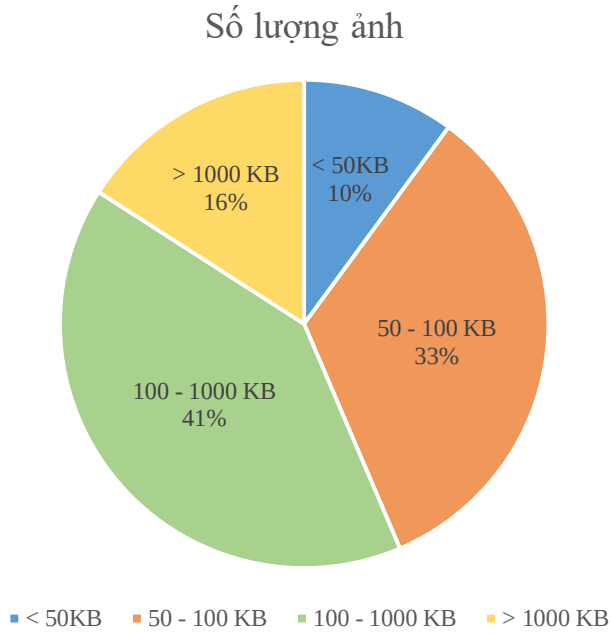
Hình ảnh cho các nhóm này có rất nhiều trên Internet nhưng sẽ có nhiều ảnh không liên quan và không phù hợp. Do đó, nhóm tác giả đã tự chụp ảnh cho một số nhóm vì việc này sẽ dễ dàng hơn là lọc ra các kết quả không phù hợp.

Kết quả là sau khi chụp ảnh, kết hợp với các ảnh tìm được từ Internet và thực hiện tìm, xóa các ảnh trùng lặp *(nếu có)* thì bộ dữ liệu cuối cùng chứa *764 hình ảnh từ Internet* và *1236 hình ảnh* là *thu nhập được từ thực tế*.

Hình 9. Một số ví dụ khác trong bộ dữ liệu huấn luyện

2. Bộ dữ liệu huấn luyện

Bộ dữ liệu huấn luyện của bộ dữ liệu VinText gồm có 1500 ảnh, tuy nhiên trong đó bao gồm nhiều ảnh có kích thước lớn dẫn đến việc tràn bộ nhớ trong quá trình huấn luyện mô hình. Cụ thể, có 486 ảnh có kích thước từ 100 đến 1000 KB, 191 ảnh có kích trên 1000 KB.



Hình 10. Biểu đồ số lượng ảnh theo kích thước ảnh

Để giải quyết vấn đề này nhóm đã loại bỏ các ảnh không thỏa điều kiện sau:

$$W \times H < 300000$$

Trong đó:

- *W là chiều dài của ảnh* (đơn vị: pixel)
- *H là chiều rộng của ảnh* (đơn vị: pixel)

Sau khi tiến hành loại bỏ các ảnh không thỏa điều kiện bộ dữ liệu huấn luyện còn lại 384 ảnh, với 6003 trường hợp văn bản (*text instances*).

Ngoài ra, để huấn luyện mô hình TextFuseNet bộ dữ liệu cần có thông tin về vị trí của văn bản trong ảnh ở dạng phân đoạn (segmentation). Tuy nhiên bộ dữ liệu VinText chỉ sử dụng bounding box để đánh dấu vị trí của văn bản. Nhóm đã sử dụng thông tin về các điểm Bezier làm phân đoạn văn bản (text segmentation) cho các văn bản xuất hiện trọng ảnh.

Hình 11. Phân đoạn văn bản sử dụng các điểm Bezier

3. Pseudo Label

Bộ dữ liệu VinText chỉ gán nhãn dữ liệu ở mức độ từ (sử dụng bounding box), do đó bộ dữ liệu này còn thiếu thông tin về vị trí của từng chữ cái trong ảnh.

Hình 12. Vị trí của các ký tự được dự đoán bởi pretrained

Hình 13. Vị trí được dự đoán của các ký tự trong ảnh (thuộc bộ dữ liệu huấn luyện)

Để giải quyết vấn đề này, nhóm đã sử dụng một mô hình pretrained của nhóm tác giả TextFuseNet để dự đoán vị trí của các chữ cái trong ảnh, kết quả gán nhãn thu được là các pseudo-label, đây cũng là cách làm được tác giả của TextFuseNet gợi ý [link].

Mô hình pretrained được sử dụng là mô hình TextFuseNet được huấn luyện trên bộ dữ liệu TotalText. Do TotalText là một bộ dữ liệu chứa nhiều trường hợp khó như chữ

cong (curved), chữ có nhiều phương (perspective), chữ có hình dạng tùy ý (arbitrary shaped),... nên việc pretrained trên bộ dữ liệu này đã được lựa chọn để dự đoán các pseudo-label cho các chữ cái (vì pretrained có khả năng dự đoán chữ cái ở nhiều hình dạng khác nhau).

CHƯƠNG IV. CÀI ĐẶT MÔ HÌNH

1. Kiến trúc của các mạng nơ-ron được sử dụng

1.1. FPN

Sử dụng *backbone* là *ResNet-101*, với 4 *output* tại *4 residual blocks* $\{C_2, C_3, C_4, C_5\}$ được xem như *output* của luồng *Bottom-up*

1.2. RPN

Sử dụng 3 lớp tích chập:

- Lớp đầu tiên dùng để trích xuất đặc trưng với *input* là một *feature map* 256x256 và *output* là một *feature map* có cũng kích thước
- Lớp thứ hai dùng để phân loại với *input* là một *feature map* kích thước 256x256 *output* là một *feature map* kích thước 3x3
- Lớp thứ ba dùng để xác định *bounding box* với *input* cũng là một *feature map* kích thước 256x256 và *output* là một *feature map* kích thước 12x12.

1.3. Nhánh semantic segmentation

Theo thứ tự các *feature maps* sẽ được truyền qua các lớp:

- Một lớp tích chập 1x1
- Một lớp tích chập 3x3
- Một lớp RoIAlign với kích thước 14x14
- Một lớp tích chập 3x3
- Một lớp tích chập 1x1
- Một lớp tích chập 1x1 để dự đoán mask cho input
- Hàm ReLU

1.4. Nhánh detection

Các *feature maps* sẽ được truyền qua các lớp theo thứ tự sau:

- Một lớp *RoIAlign* với kích thước 7x7
- Hai lớp *fully connected* với kích thước lần lượt là: (12544, 1024), (1024, 1024)

- Hai lớp fully connected để tính toán cls_score (xác suất để một vật thể thuộc một lớp nào đó) và xác định *bounding box*.

1.5. Nhánh mask

Các *feature map* sẽ được truyền qua các lớp theo thứ tự sau:

- Một lớp RoIAlign với kích thước 7x7
- Một lớp tích chập với kích thước 3x3
- Một lớp tích chập chuyển vị với kích thước 2x2
- Một lớp tích chập 1x1 để dự đoán mask (dự đoán các đường contour)

2. Giá trị của một số tham số khác

Trong quá trình huấn luyện mô hình, chúng tôi có sử dụng pretrain trên bộ dữ liệu *Syntext* được cung cấp tại (link). Mô hình được huấn luyện với số iter là 20000 (chúng tôi có thực hiện huấn luyện vưới số iter lên đến 75000, tuy nhiên nhận thấy mô hình được huấn luyện với 20000 iter cho kết quả tốt nhất), *learning rate* là 0.001, mô hình ResNet-101 (backbone của FPN) sử dụng *pretrain* trên tập dữ liệu *ImageNet*.

Chi tiết về các tham số và kiến trúc của các mạng nơ-ron có thể được tham khảo tại file log.txt.

3. Về output của mô hình

Trên thực tế mô hình có thể xuất ra được cả bounding box và đường contour bao quanh văn bản (ở cả mức từ và mức chữ cái), tuy nhiên do bộ dataset được nhóm sử dụng (VinText) chỉ sử dụng bounding box để đánh dấu vị trí của văn bản trong hình, cho nên nhóm *chỉ thực hiện đánh giá mô hình dựa trên output là bounding box* (do thiếu groundtruth để có thể đánh giá dựa trên các đường contour).

Tương tự, đối với trường hợp output là vị trí của các ký tự (thay vì từ) cũng sẽ không được đánh giá (các chữ cái trong hình ảnh được đánh dấu sử dụng pseudo-label, do đó về cơ bản chúng ta không có groundtruth thực sự để đánh giá kết quả trả về).

Mặc dù các output đánh dấu vị trí của các ký tự và các đường contour không được sử dụng để đánh giá mô hình, nhưng việc huấn luyện mô hình lại cần đến những thông

tin này. Cụ thể, nhánh sematic segmentation cần thông tin về các đường contour bao bọc văn bản trong ảnh để huấn luyện mạng nơ-ron nhằm trích xuất những đặc trưng ở mức toàn cục. Ở nhánh mask và nhánh detection cũng cần thông tin về vị trí của từng ký tự cụ thể, những thông tin này sẽ được sử dụng để huấn luyện mạng nơ-ron ở hai nhánh này nhằm trích xuất các đặc trưng ở mức chữ cái (nhánh mask), mức từ và mức toàn cục. Các đặc trưng ở mức toàn cục và mức chữ cái sẽ được gộp với đặc trưng ở mức từ tại các khối multi-path fusion. Nhờ đó mô hình có thêm thông tin để có thể đưa ra các dự đoán chính xác hơn.

Ngoài ra, với các dự đoán về vị trí của văn bản ở mức độ chữ cái, mô hình cũng có thể dự đoán được chữ cái xuất hiện trong vùng ảnh đó là chữ cái nào. Tuy nhiên, các dự đoán này chỉ có thể sử dụng có các chữ cái trong bảng chữ cái tiếng Anh (không hoạt động cho các ngôn ngữ khác chẳng hạn như tiếng Trung), đây là kết quả đồng thời có được trong quá trình detect, không phải là kết quả mong muốn cuối cùng của mô hình.

CHƯƠNG V. ĐÁNH GIÁ MÔ HÌNH

1. Phương pháp

1.1. Intersection over Union (IoU)

Với bài toán Scene Text Detection, để đánh giá độ chính xác của các bouding box được dự đoán từ mô hình, chúng tôi sẽ sử dụng một độ đo đặc biệt để đánh giá chúng *(bouding box evaluation)*.

Các bouding box sau khi được phát hiện sẽ được so sánh với ground truth tương ứng để đánh giá là đúng hay sai. Việc đánh giá này được thực hiện bằng cách giao các bouding box dự đoán và ground truth lại với nhau.

Để một dự đoán được xem là chính xác, diện tích của phần giao nhau a_0 giữa bouding box dự đoán B_p và bouding box ground truth B_{gt} phải vượt qua 0.5 (50%). Diện tích này được tính toán theo công thức:

$$a_0 = \frac{area(B_p \cap B_{gt})}{area(B_p \cup B_{gt})}$$

Trong đó:

- $B_p \cap B_{gt}$: phần giao của bounding box dự đoán và ground truth, tức là phần mà chúng chồng lấp lên nhau.
- $B_p \cup B_{gt}$: phần hợp của bounding box dự đoán và ground truth, tức là tất cả các phần bao gồm cả phần chồng lấp.

Do đó, công thức trên được gọi là ***Intersection over Union (IoU)***, nghĩa là lấy phần giao chia cho phần hợp của các *bounding box*.

Ngưỡng *(threshold)* 50% được đặt như vậy là tương đối thấp nhưng điều này để giải thích cho sự không chính xác trong các bounding box trong các ground truth. Nếu có nhiều lần detect cho cùng một đối tượng thì sẽ được xem là detect sai. ***Ví dụ:*** 5 lần detect cho một đối tượng được tính là 1 lần detect đúng và 4 lần detect sai. Do đó, hệ thống của chúng ta phải lọc ra các lần phát hiện trùng lặp.

1.2. Precision và Recall

Độ đo *IoU* chỉ dùng để xem xét một bouding box được dự đoán là đúng hay sai. Để có thể đánh giá trên tất các các bounding box được dự đoán từ mô hình, ta phải dùng *IoU* để xem xét các bouding box đó và tính *Precision* cũng như *Recall* cho chúng.

Trước khi đi vào công thức tính toán các độ đo này, ta phải tính được các giá trị ***True Positive*** *(TP)*, ***False Positive*** *(FP)* và ***False Negative*** *(FN)*. Theo đó, những vị trí được dự đoán có chứa văn bản là *positive,* ngược lại những vị trí không chứa văn bản là *negative*.

True Positive chính là các bounding box được dự đoán được xem là chính xác so với groundtruth (những vị trí được dự đoán có chứa văn bản và thực sự có chứa văn bản).

False Positive là các bounding box được dự đoán tại các vị trí không có trong ground truth (những vị trí được dự đoán có chứa văn bản nhưng thực tế không chứa văn bản).

False Negative là trường hợp có bounding box ở một vị trí có văn bản trong ground truth nhưng mô hình không detect ra được (những vị trí được dự đoán không chứa văn bản nhưng thực tế lại có chứa văn bản).

Từ các giá trị đã tính toán được, ta có công thức của *Precision* và *Recall* như sau:

$$Precision = \frac{TP}{TP + FP}; Recall = \frac{TP}{TP + FN}$$

Giá trị *Precision* sẽ cho ta biết được mô hình detect được đúng bao nhiêu phần trăm, tức độ chính xác của mô hình đó. Còn giá trị của *Recall* cho ta biết được mô hình detect được bao nhiêu kết quả có trong ground truth, tức độ phủ của mô hình đó.

Bên cạnh hai độ đo này, nhóm còn sử dụng một độ đo khác để đánh giá, đó là *F1*. Độ đo này sẽ phản ánh sự tương quan giữa *Precision* và *Recall.* Nếu *Precision* và *Recall* cùng lớn thì *F1* sẽ lớn và ngược lại, nếu *Precision* và *Recall* cùng nhỏ thì *F1* sẽ nhỏ. Độ đo *F1* được tính theo công thức sau:

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$$

2. Kết quả

Kết quả đánh giá trên 500 ảnh trên bộ dữ liệu *VinText* cho thấy, mô hình đạt được *Recall* ở mức 0.743, *Precision* đạt mức 0.892 và *F1* đạt mức 0.811.

Hình 14. Kết quả được dự đoán bởi mô hình

Mô hình có thể phát hiện được văn bản trong một số trường hợp như chữ cong (*curved text*), chữ méo (*distorded text*), chữ có hình dạng tùy biến (*arbitrary shaped text*), …

Tuy nhiên mô hình còn một số hạng chế như: không phát hiện được hết văn bản trong trường hợp văn bản có chiều đứng, trường hợp văn bản xuất hiện với mật độ dày đặc, có nhiều trường hợp *false positive*, bỏ qua những văn bản có *texture* phức tạp… Ngoài ra mô hình cũng bị ảnh hưởng nếu ảnh đầu vào có kích thước quá lớn.

Hình 15. Các trường hợp dự đoán sai

KẾT LUẬN

Mô hình TextFuseNet không chỉ sử dụng đặc trưng ở mức từ mà còn tổng hợp đặc trưng ở mức chữ cái và mức toàn cục, nhờ đó mô hình có thể phát hiện được văn bản trong các trường hợp phức tạp (như chữ cong, chữ có hình dạng đặc biệt, …). Tuy nhiên TextFuseNet cần gán nhãn ở mức chữ cái (hầu hết các bộ dữ liệu chỉ gán nhãn ở mức độ từ), nhưng vấn đề này có thể được giải quyết bằng cách sử dụng pseudo-label (hoặc sử dụng weakly supervised learning như nhóm tác giả TextFuseNet đã thực hiện). Mô hình vẫn có một số hạn chế như không thể phát hiện được những văn bản có bề mặt (texture) phức tạp. Thêm vào đó, do không có groundtruth ở mức độ chữ cái nên chúng ta khó có thể đánh giá mô hình ở mức độ chữ cái.

TÀI LIỆU THAM KHẢO

[1] Jian Ye, Z. C. (2020). TextFuseNet: Scene Text Detection with Richer Fused Features. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence* (pp. 516--522). International Joint Conferences on Artificial Intelligence Organization.

[2] Mark Everingham, L. V. (2009). The PASCAL Visual Object Classes (VOC) Challenge. International Journal of Computer Vision.

[3] Lee, D.-H. (2013). Pseudo-Label : The Simple and Efficient Semi-Supervised Learning Method for Deep Neural Networks.

[4] Nguyen Nguyen, T. N.-T. (2021). Dictionary-guided Scene Text Recognition. *Proceedings of the {IEEE} Conference on Computer Vision and Pattern Recognition (CVPR)*.

[5] Shaoqing Ren, K. H. (2016). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks.

[6] Tsung-Yi Lin, P. D. (2017). Feature Pyramid Networks for Object Detection.

[7] *A PyTorch implementation of "TextFuseNet: Scene Text Detection with Richer Fused Features"*. (n.d.). Retrieved from https://github.com/ying09/TextFuseNet

[8] *FPN — Feature Pyramid Network (Object Detection)*. (n.d.). Retrieved from Hamhochoi: https://dothanhblog.wordpress.com/2019/12/29/fpn-feature-pyramid-network-object-detection/

[9] K, S. (n.d.). *Region Proposal Network — A detailed view*. Retrieved from Towards Data Science: https://towardsdatascience.com/region-proposal-network-a-detailed-view-1305c7875853

[10] *Region Proposal Network (RPN) architecture explained*. (n.d.). Retrieved from towardsmachinelearning.org: https://towardsmachinelearning.org/region-proposal-network/

[11] *Tasks - Incidental Scene Text*. (n.d.). Retrieved from Robust Reading Competition: https://rrc.cvc.uab.es/?ch=4&com=tasks

[12] Tsang, S.-H. (n.d.). *Review: FPN — Feature Pyramid Network (Object Detection)*. Retrieved from Towards Datascience: https://towardsdatascience.com/review-fpn-feature-pyramid-network-object-detection-262fc7482610