

DẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH  
TRƯỜNG ĐẠI HỌC BÁCH KHOA  
KHOA KHOA HỌC - KỸ THUẬT MÁY TÍNH



**LUẬN VĂN TỐT NGHIỆP ĐẠI HỌC**

---

**Nhận dạng biểu thức toán học**

Hội đồng : Khoa học máy tính

Giảng viên hướng dẫn : TS. Lê Thành Sách

Giảng viên phản biện : TS. Nguyễn Đức Dũng

Nhóm sinh viên thực hiện : Phan Tân Phúc - 51303058

Bùi Khánh Ngọc - 51302567

Tp. Hồ Chí Minh, Tháng 12/2017



## Lời cam đoan

Luận văn của nhóm có tham khảo các tài liệu từ nhiều nguồn khác nhau và các nguồn tham khảo này đều được trích dẫn rõ ràng trong phần tài liệu tham khảo. Ngoài những phần được trích dẫn, nhóm xin cam đoan toàn bộ nội dung báo cáo là do các thành viên nhóm tự soạn thảo dựa trên những tìm hiểu và kết quả thực tế do nhóm tạo ra.

Thành viên nhóm sẽ hoàn toàn chịu xử lý theo quy định nếu có bất kỳ sai phạm nào xảy ra liên quan đến những gì nhóm đã cam đoan.

Hồ Chí Minh, ngày 08 tháng 12 năm 2017

Nhóm sinh viên thực hiện

Phan Tân Phúc

Bùi Khánh Ngọc



## Lời nói đầu

Luận văn tốt nghiệp là thử thách cuối cùng của chặng đường 4.5 năm trên giảng đường đại học của những sinh viên Bách Khoa. Nó khép lại một giai đoạn mà ở đó chứa đựng đầy những nỗ lực, những phấn đấu, cả niềm vui và thỉnh thoảng là những nỗi buồn, sự luyến tiếc. Nhưng cũng từ đây một cánh cửa mới sẽ mở ra- cánh cửa của lao động, của công hiến.

Để đến thời điểm này, nhóm muốn gửi lời cảm ơn chân thành đến Ban giám hiệu và các thầy cô Trường Đại học Bách Khoa Thành phố Hồ Chí Minh, cách riêng cho các thầy cô Khoa Khoa học và Kỹ thuật Máy tính đã chỉ dẫn cho các thành viên nhóm trong suốt những năm học vừa qua.

Trong quá trình thực hiện luận văn, nhóm đã nhận được rất nhiều sự hỗ trợ về mặt chuyên môn cũng như những đóng góp trong vấn đề xây dựng tập dữ liệu cho đề tài từ các anh và các bạn trong và ngoài khoa Khoa học và Kỹ thuật Máy tính. Chính sự giúp sức này là động lực và cũng là sức mạnh giúp nhóm vượt qua những khó khăn trong quá trình thực hiện và đến thời điểm này có thể nói đã hoàn thành thành công. Nhóm xin bày tỏ sự cảm kích và biết ơn chân thành đến những con người này, cách riêng cho Thạc sĩ Huỳnh Chí Kiên- người anh đã luôn hỗ trợ và đưa ra những lời khuyên hữu ích bất kỳ khi nào nhóm gặp khó khăn.

Trên tất cả, lời cảm ơn chân thành nhất và sâu sắc nhất xin được gửi đến thầy hướng dẫn đề tài- Tiến sĩ Lê Thành Sách. Cảm ơn thầy đã luôn theo sát, hỗ trợ cũng như định hướng công việc cho nhóm. Cảm ơn thầy đã luôn tạo ra áp lực để thúc đẩy sinh viên của mình tiến về phía trước và cung cảm ơn thầy vì những khi áp lực nhất đều được thầy chia sẻ.

Nhóm cũng không quên gửi lời cảm ơn đến nhóm tác giả khoá 2011[1] và tác giả Kuangliu[2] vì trong quá trình thực hiện luận văn, nhóm có sử dụng mã nguồn của các tác giả này.

Sau cùng, vì những hạn chế về mặt thời gian cũng như khả năng trong cách trình bày và viết báo cáo nên không thể tránh khỏi những thiếu sót, rất mong nhận được sự thông cảm và những ý kiến đóng góp từ quý thầy cô và các bạn để giúp nhóm hoàn thiện hơn.

Chân thành cảm ơn.

Hồ Chí Minh, ngày 08 tháng 12 năm 2017  
Nhóm sinh viên thực hiện

**Phan Tấn Phúc**  
**Bùi Khánh Ngọc**



## Tóm tắt luận văn

Luận văn này chỉ ra quá trình thực hiện của nhóm từ tìm hiểu những kiến thức cần thiết đến hiện thực đề tài và đánh giá kết quả đạt được. Bố cục của luận văn gồm 6 chương, không kể các mục lục, phụ lục khác.

Chương 1 là chương giới thiệu tổng quan về đề tài. Ở đó sẽ trình bày nhu cầu của đề tài trong xã hội cũng như đưa ra lý do vì sao nhóm chọn đề tài này và sau cùng là quá trình thực hiện của nhóm từ sau giai đoạn thực tập tốt nghiệp.

Chương 2 sẽ đề cập những kiến thức liên quan trực tiếp đến đề tài mà nhóm đã tìm hiểu, cụ thể là kiến trúc mạng SSD.

Chương 3 tóm lược một số công trình liên quan trực tiếp đến đề tài. Trong mục này, nhóm sẽ giới thiệu 4 công trình là cơ sở tham khảo cho quá trình hoàn thiện luận văn.

Chương 4 trình bày về phương pháp đã được sử dụng để hiện thực đề tài luận văn. Phương pháp này được xây dựng dựa trên những kiến thức đã tìm hiểu được trình bày ở chương 2 và những công trình được giới thiệu ở chương 3.

Chương 5 chỉ ra quá trình hiện thực đề tài bao gồm chuẩn bị tập dữ liệu, xây dựng hệ thống và chương trình thử nghiệm. Cụ thể, nhóm sẽ diễn giải về cách cấu hình, các thông số trong từng giai đoạn của quá trình nhận dạng. Tiếp theo, nhóm sẽ giới thiệu cách thức thu thập dữ liệu cùng một số mô tả cho tập dữ liệu hiện tại mà nhóm đang sử dụng. Cuối cùng là kết quả từ chương trình thử nghiệm và những so sánh, đánh giá giữa các phiên bản được tạo ra từ việc thay đổi một số yếu tố liên quan đến quá trình nhận dạng.

Chương 6 đánh giá tổng kết luận văn. Nhóm sẽ nêu lên những điều còn tồn đọng và những điểm nổi bật, cũng như hướng phát triển tiếp theo của đề tài. Ngoài ra, sẽ là một vài dòng đúc kết lại chặng đường thực hiện luận văn tốt nghiệp của các thành viên nhóm: những điều đã học được, cảm xúc khi hoàn thành đề tài luận văn.



## Mục lục

|  |           |
|--|-----------|
| <b>Mục lục</b>   | <b>4</b>  |
| <b>Danh sách hình vẽ</b>   | <b>6</b>  |
| <b>Danh sách bảng</b>  | <b>7</b>  |
| <b>1 Giới thiệu</b>  | <b>9</b>  |
| 1.1 Giới thiệu đề tài .....  | 9         |
| 1.2 Lý do chọn đề tài .....  | 9         |
| 1.3 Phạm vi đề tài .....   | 9         |
| 1.4 Quá trình thực hiện .....  | 10        |
| <b>2 Kiến thức đã tìm hiểu</b>   | <b>12</b> |
| 2.1 Bộ mã hóa (Encoder) .....  | 12        |
| 2.2 Bộ phát hiện - phân loại .....   | 14        |
| 2.3 Tính giá trị lỗi .....   | 15        |
| 2.4 Bộ giải mã (Decoder) .....   | 16        |
| 2.5 Một số vấn đề khác trong mạng SSD[3] .....   | 16        |
| <b>3 Công trình liên quan</b>  | <b>18</b> |
| 3.1 Tổng quan .....  | 18        |
| 3.2 Công trình tham khảo .....   | 18        |
| 3.2.1 Watch, Attend and Parse: An End-to-end Neural Network Based Approach to Handwritten Mathematical Expression Recognition[4] ..... | 19        |
| 3.2.2 Context-aware Recognition[5] .....   | 19        |
| 3.2.3 QAK[1] .....   | 20        |
| <b>4 Mô hình đề xuất</b>   | <b>22</b> |
| 4.1 Tổng quan .....  | 22        |
| 4.2 Giai đoạn nhận diện ký tự .....  | 22        |
| 4.2.1 Mạng cơ sở .....   | 22        |
| 4.2.2 Thân mạng SSD[3] .....   | 23        |
| 4.3 Giai đoạn phân tích cú pháp .....  | 26        |
| 4.3.1 Sinh cây BST[6] .....  | 26        |
| 4.3.2 Sinh cây Lexed - BST[6] .....  | 29        |
| <b>5 Hiệu thực, đánh giá</b>   | <b>30</b> |
| 5.1 Chuẩn bị dữ liệu .....   | 30        |
| 5.1.1 Xây dựng tập ký tự .....   | 30        |
| 5.1.2 Xây dựng tập biểu thức .....   | 30        |
| 5.1.2.a Quá trình thu thập dữ liệu .....   | 30        |
| 5.1.2.b Thông tin mô tả tập dữ liệu biểu thức .....  | 34        |
| 5.2 Xây dựng mô hình .....   | 37        |
| 5.2.1 Giảm kích thước các default box - Phiên bản II .....   | 37        |
| 5.2.2 Giảm kích thước các default box và tăng kích thước ảnh đầu vào - Phiên bản III .....   | 38        |
| 5.2.3 Giảm kích thước các default box, tăng kích thước ảnh đầu vào và thêm lớp tích chập - Phiên bản IV .....                          | 38        |
| 5.3 Xây dựng bản thử nghiệm .....  | 39        |
| 5.4 Kết quả .....  | 40        |
| 5.4.1 So sánh định tính giữa bốn phiên bản .....   | 40        |
| 5.4.1.a Quá trình matching .....   | 40        |



---

|                 |   |           |
|-----------------|---|-----------|
| 5.4.1.b         | Quá trình nhận diện . . . . .                     | 40        |
| 5.4.2           | So sánh định lượng . . . . .                      | 43        |
| <b>6</b>        | <b>Tổng kết</b>                                   | <b>46</b> |
| 6.1             | Kết luận . . . . .                                | 46        |
| 6.2             | Dánh giá ưu, nhược điểm . . . . .                 | 46        |
| 6.2.1           | Ưu điểm . . . . .                                 | 46        |
| 6.2.2           | Nhược điểm . . . . .                              | 46        |
| 6.3             | Hướng phát triển trong tương lai . . . . .        | 46        |
| 6.3.1           | Mở rộng tập dữ liệu biểu thức . . . . .           | 47        |
| 6.3.2           | Khắc phục lỗi nhận dạng sai ở ký tự lớn . . . . . | 48        |
| 6.3.3           | Các hướng phát triển khác . . . . .               | 48        |
| <b>7</b>        | <b>Lời kết</b>                                    | <b>50</b> |
| <b>8</b>        | <b>Phụ lục</b>                                    | <b>51</b> |
| 8.1             | Những lỗi sai mà hệ thống gặp phải . . . . .      | 51        |
| <b>Tài liệu</b> |   | <b>52</b> |



## Danh sách hình vẽ

|    |  |    |
|----|--|----|
| 1  | Mô phỏng chỉ số Jaccard[7] . . . . .   | 14 |
| 2  | Cấu tạo của một mạng SSD[3] . . . . .  | 15 |
| 3  | Hình minh họa các bước thực hiện của phương pháp Watch, Attend and Parser. . . . .                               | 19 |
| 4  | Mô hình học được đề xuất [5]. . . . .  | 20 |
| 5  | Quy trình nhận dạng. . . . .   | 21 |
| 6  | Mô hình phương pháp. . . . .   | 22 |
| 7  | Cấu tạo mạng VGG16[8] . . . . .  | 23 |
| 8  | Sơ đồ các lớp trong mạng SSD[3] . . . . .  | 24 |
| 9  | Một cây BST[6] (Các đường màu đỏ biểu thị mối quan hệ nút cha con) . . . . .                                     | 26 |
| 10 | Một cây Lexed - BST[6] . . . . .   | 26 |
| 11 | Biểu mẫu để thu thập dữ liệu. . . . .  | 31 |
| 12 | Ví dụ mẫu biểu thức gợi ý cho người viết. . . . .  | 32 |
| 13 | Ví dụ một mẫu thu thập được từ người tham gia viết. . . . .  | 33 |
| 14 | Ví dụ một ảnh thu được sau khi xử lý biểu mẫu đã qua scan bằng đoạn mã Matlab. .                                 | 34 |
| 15 | Giao diện công cụ hỗ trợ quá trình gán nhãn và xác định bounding boxes. . . . .                                  | 34 |
| 16 | Sự phân phối ký tự theo từng nhãn đối với tập huấn luyện và tập kiểm tra. . . . .                                | 36 |
| 17 | Ảnh ví dụ chữ nhỏ (với kích thước thật) . . . . .  | 37 |
| 18 | Cấu trúc mạng của phiên bản III . . . . .  | 38 |
| 19 | Cấu trúc mạng của phiên bản III . . . . .  | 39 |
| 20 | Giao diện bản thử nghiệm . . . . .   | 40 |
| 21 | Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự nhỏ tăng dần qua 4 phiên bản. (1) . . . . .            | 41 |
| 22 | Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự nhỏ tăng dần qua 4 phiên bản. (2) . . . . .            | 41 |
| 23 | Kết quả dự đoán của 4 phiên bản. Phiên bản III và IV tỏ ra vượt trội hai phiên bản trước. . . . .                | 42 |
| 24 | Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự lớn giảm mạnh ở phiên bản II. . . . .                  | 42 |
| 25 | Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự lớn ở phiên bản IV không thua kém phiên bản I. . . . . | 43 |
| 26 | Ảnh kết quả của phiên bản IV(với kích thước thật) . . . . .  | 43 |
| 27 | Trực quan kết quả mAP cho 4 phiên bản thử nghiệm. . . . .  | 45 |
| 28 | Một số ảnh kết quả sau quá trình nhận dạng sử dụng phiên bản IV. . . . .   | 45 |
| 29 | Trường hợp không nhận được chỉ số dưới. . . . .  | 47 |
| 30 | Kết quả nhận dạng chỉ số dưới trên bộ ảnh test mới. . . . .  | 47 |
| 31 | Kết quả nhận dạng sau khi được huấn luyện với dữ liệu bổ sung. . . . .   | 47 |
| 32 | Ví dụ thu nhỏ ảnh. . . . .   | 48 |
| 33 | Kết quả thu được sau khi thu nhỏ nội dung ảnh. . . . .   | 48 |
| 34 | Trường hợp sinh chuỗi Latex sai khi biểu thức chứa các ký tự có chân. . . . .                                    | 51 |
| 35 | Trường hợp hệ thống gán sai nhãn cho những ký tự có nét tương tự nhau. . . . .                                   | 51 |



## Danh sách bảng

|    |  |    |
|----|--|----|
| 1  | Một số công trình tiêu biểu về nhận dạng biểu thức toán học trước năm 2000 và các phương pháp được sử dụng. . . . .  | 18 |
| 2  | Cấu hình các lớp tích chập trong mạng SSD300[3] . . . . .  | 25 |
| 3  | Thông tin của các feature map trích được từ mạng SSD300[3] . . . . .   | 25 |
| 4  | Bảng phân lớp các ký tự cần nhận diện và các ngữ cảnh xác định phân vùng con . . . . .   | 27 |
| 5  | Thông tin mô tả hai tập dữ liệu dùng cho huấn luyện và kiểm tra. . . . .   | 35 |
| 6  | Thông tin của các feature map trích được từ mạng SSD đã giảm kích thước default box và tăng kích thước ảnh đầu vào . . . . .                               | 38 |
| 7  | Cấu hình các lớp tích chập thêm vào mạng SSD300[3] để được mạng SSD chỉnh sửa . . . . .  | 39 |
| 8  | Thông tin của các feature map trích được từ mạng SSD[3] đã giảm kích thước default box, tăng kích thước ảnh đầu vào và tăng số lớp tích chập . . . . .     | 39 |
| 9  | Kích thước bounding box ground truth nhỏ nhất mà mạng có thể match với default box được của bốn mô hình đề xuất. (Đơn vị tính: Điểm ảnh - Pixel) . . . . . | 40 |
| 10 | Sự khác nhau giữa 4 phiên bản thử nghiệm. . . . .  | 44 |
| 11 | Kết quả mAP của 4 phiên bản thử nghiệm. . . . .  | 44 |



## Danh mục từ viết tắt

| Thuật ngữ | Giải thích  |
|-----------|---|
| CNN       | Mạng nơron tích chập (Convolutional Neural Network)                                       |
| FCN       | Mạng nơron tích chập đầy đủ (Fully convolutional network)                                 |
| SSD       | Single Shot Mutilbox Detector   |
| BST       | Cây ngữ pháp dựa trên đường cơ sở (Baseline Syntax Tree)                                  |
| DRACULAE  | Diagram Recognition Application for Computer Understanding of Large Algebraic Expressions |



## 1 Giới thiệu

### 1.1 Giới thiệu đề tài

Khoa học công nghệ phát triển đồng nghĩa với việc xã hội "trở nên tự động hoá" bởi lẽ con người ngày càng có nhu cầu tìm kiếm sự hỗ trợ từ các thiết bị công nghệ và hạn chế dùng sức lực của chính mình. Dựa trên nền tảng công nghệ và nhu cầu của xã hội, hàng loạt những thiết bị, ứng dụng công nghệ ra đời. Chúng phục vụ nhu cầu của con người trong đời sống hằng ngày cũng như trong hầu hết mọi lĩnh vực của xã hội từ giao thông, y tế, thương mại,... đến giáo dục. Trong lĩnh vực y tế, phải kể đến ứng dụng giám sát sức khoẻ người dùng trên những chiếc đồng hồ thông minh của Apple hay SamSung. Với giao thông, những ứng dụng chỉ đường, định vị và giám sát xe để phòng trộm ngày càng phổ biến và giúp ích thực sự cho con người. Riêng với giáo dục, vấn đề thường gặp đối với các bạn học sinh là giải, trực quan hoá các phương trình, hàm số toán học hay soạn các giáo án chứa nhiều công thức, ký hiệu phức tạp đối với thầy cô. Họ cần một giải pháp nào đó giúp giảm thiểu công sức trong những tình huống như vậy. Giải pháp này cần phải giải quyết những vấn đề cơ bản sau:

- Số hoá các công thức in trong sách hay được viết tay.
- Giải tham khảo một số dạng phương trình.
- Trực quan hoá các hàm số
- Biểu diễn công thức, ký hiệu viết tay dưới những lệnh mà các trình soạn thảo toán có thể hiểu được, ví dụ Latex.
- ...

### 1.2 Lý do chọn đề tài

Bởi sự cần thiết về một ứng dụng nhận dạng biểu thức toán học đã được trình bày ở mục 1.1, trên thị trường cũng đã xuất hiện nhiều sản phẩm như vậy, đáng chú ý là PhotoMath<sup>1</sup>. Tuy nhiên, nhóm nhận thấy thách thức nếu phải hiện thực thành công đề tài này. Một số câu hỏi đã được đặt ra:

- Làm sao có thể nhận dạng được các ký hiệu?
- Làm cách nào để nhận dạng cả một biểu thức?
- Làm sao biết được đây là loại biểu thức gì?
- Liệu có chắc chắn bất kỳ những gì mình viết ra đều được hiểu đúng?
- Nhóm có thể tạo ra được một sản phẩm hoàn thiện như PhotoMath không?

Để tự mình trả lời những câu hỏi đó, nhóm quyết tâm thực hiện đề tài này. Ngoài ra, việc áp dụng kiến thức đã học để tạo ra một sản phẩm vừa cần thiết cho xã hội vừa tự bản thân mình có thể sử dụng được tạo cho nhóm một động lực để tiến hành.

### 1.3 Phạm vi đề tài

- Nhận dạng biểu thức toán học viết tay dạng offline<sup>2</sup>.
- Chuyển biểu thức từ dạng hình ảnh sang dạng máy có thể hiểu được (Latex).

<sup>1</sup>Một ứng dụng về nhận dạng và giải các biểu thức toán học nổi bật trên Google Play.

<sup>2</sup>Nhận dạng từ ảnh chưa biểu thức toán học.



## 1.4 Quá trình thực hiện

Bước 1: Tìm hiểu những công trình trong nước và nước ngoài liên quan đến vấn đề nhận dạng biểu thức toán học.

- Mục tiêu:
  - Biết được khả năng giải quyết vấn đề của các hệ thống nhận dạng biểu thức toán học đã được xây dựng.
  - Hiểu được phương pháp mà các nhóm tác giả sử dụng để thực hiện công trình của họ.
- Công việc:
  - Tìm kiếm và chọn lọc những bài báo với nội dung nhận dạng biểu thức toán học.
  - Đọc và nghiên cứu các thông tin được cung cấp bởi các bài báo.
  - Tải mã nguồn (nếu có), chạy thử nghiệm.

Bước 2: Chọn ra một phương pháp trong các phương pháp đã đọc để hiện thực.

- Mục tiêu:
  - Xây dựng từ đầu một hệ thống nhận dạng biểu thức toán học theo phương pháp đề xuất.
  - Kiểm chứng độ chính xác được nêu trong bài báo và thực tế nhóm làm được. Trên cơ sở đó, tiến tới việc cải tiến để thu được kết quả tốt hơn.
- Công việc:
  - Đọc kỹ lại bài báo đã chọn.
  - Tìm hiểu những kiến thức được giới thiệu trong bài báo ở mức độ có thể vận dụng được.
  - Triển khai và hiện thực phương pháp.

Bước 2+<sup>3</sup>: Lựa chọn một phương pháp mới để giải quyết đề tài trên cơ sở mã nguồn mở đã có.

- Mục tiêu:
  - Có được phương án thay thế phương án cũ không khả thi.
  - Có được chương trình thử nghiệm.
- Công việc:
  - Tìm và tải bộ mã nguồn liên quan đến vấn đề nhận dạng biểu thức toán học[2].
  - Thiết lập và chạy thử mã nguồn.

Bước 3: Đề xuất phương pháp.

- Mục tiêu:
  - Quyết định phương pháp giải quyết vấn đề của nhóm.
- Công việc:
  - Thay đổi, hoàn thiện mã nguồn trên cơ sở phương pháp nhóm đề xuất.

<sup>3</sup>2+ là phương án thay thế của bước 2. Vì trong quá trình thực hiện bước 2, nhóm không thu được kết quả như mong đợi nên dẫn tới việc chọn một cách tiếp cận khác để giải quyết vấn đề.



#### Bước 4: Xây dựng tập dữ liệu

- Mục tiêu:
  - Có được dữ liệu phục vụ quá trình huấn luyện và kiểm thử hệ thống của nhóm.
- Công việc:
  - Liên hệ nhóm sinh viên Khoa 2011<sup>4</sup> để nhận được toàn bộ tập dữ liệu của họ bao gồm ảnh của các ký tự toán học.
  - Chuẩn bị mẫu thu và bổ sung thêm ảnh cho một số loại ký tự mới.
  - Xây dựng bộ ảnh chứa đựng các biểu thức toán học từ đơn giản đến phức tạp.

#### Bước 5: Cải tiến phương pháp

- Mục tiêu:
  - Hoàn thiện hệ thống nhận dạng biểu thức toán học.
- Công việc:
  - Kết hợp kiến thức và kết quả thử nghiệm trên tập dữ liệu để tìm ra phương pháp cải tiến.
  - Xây dựng các phiên bản thử nghiệm khác nhau dựa trên các yếu tố có thể cải tiến.

#### Bước 6: Xây dựng chương trình demo và đánh giá kết quả

- Mục tiêu:
  - Có kết quả đánh giá và điều chỉnh phương pháp sao cho phù hợp.
  - Có chương trình demo cho luận văn.
- Công việc:
  - Xây dựng bản demo.
  - Xem xét ưu, nhược điểm của các cải tiến dựa trên đánh giá các độ đo.
  - Cấu hình và lựa chọn phương án cải tiến khả thi.

<sup>4</sup>Nhóm sinh viên này đã từng hiện thực đề tài nhận dạng biểu thức toán học



## 2 Kiến thức đã tìm hiểu

### Mạng Single Shot Multibox Detector[3]

SSD[3] như là một mạng cải tiến của phương pháp phát hiện bằng cửa sổ trượt<sup>5</sup>. Thay vì sử dụng một (một số) cửa sổ có kích thước cố định, thì SSD sinh ra một số lượng hữu hạn các **default box**<sup>6</sup> để rồi từ các default box đó để hệ thống có thể xác định vị trí các ký tự cần nhận diện cho quá trình huấn luyện, qua đó mạng cần phải học cách dự đoán cả kích thước của các **bounding box**<sup>7</sup> quanh ký tự thay vì chỉ chấp nhận kích thước cho trước. SSD[3] cũng sử dụng các lớp tích chập<sup>8</sup> (hay cụ thể hơn là các mạng nơ-ron tích chập<sup>9</sup>) để trích đặc trưng, tạo tiền đề cho việc phát hiện và phân loại ký tự.

Nhóm xin phép được tách quá trình huấn luyện thành ba giai đoạn: mã hóa, phát hiện - phân loại và tính giá trị lỗi. Và song song với huấn luyện, quá trình kiểm tra, kiểm định, chạy thực tiễn cũng được chia thành ba giai đoạn: trích đặc trưng - phân loại và giải mã.

#### 2.1 Bộ mã hóa (Encoder)

Bộ mã hóa chỉ được sử dụng trong quá trình huấn luyện nhằm mã hóa, chuyển đổi ground truth<sup>10</sup> thô (được lưu trong tệp tin định dạng txt hoặc xml) sang ground truth mà mạng SSD[3] có thể hiểu được.

#### Sinh default box

Để mã hóa ground truth, bộ mã hóa trước hết có nhiệm vụ sinh ra nhiều default box có nhiều kích thước khác nhau, để làm được điều này, mạng cần phải được cung cấp một số thông số:

- Danh sách kích thước của các feature map (Phần này sẽ được giải thích rõ hơn ở mục sau): Mỗi feature map trong danh sách tương ứng với một mức kích thước<sup>11</sup> trong việc phát hiện ảnh. Trong một mức kích thước, tất cả các default box đều có kích thước như nhau và cách đều nhau, mỗi pixel trong feature map thể hiện một (một số default box), vì vậy, ta có thể tính được danh sách khoảng cách giữa trọng tâm giữa các default box bằng cách lấy kích thước ảnh chia cho kích thước của feature map ở mức kích thước tương ứng. Tất cả các dữ liệu liên quan đến kích thước sau đó sẽ được chuyển về tập giá trị  $\{x \in R | x \in [0, 1]\}$
- Danh sách các aspect ratio<sup>12</sup> ứng với mỗi mức kích thước. Mạng SSD dựa vào các aspect ratio đó để tạo ra các default box có hình dạng khác nhau tại cùng một vị trí (trọng tâm của các default box có cùng một vị trí).
- Kích thước nhỏ nhất và lớn nhất của các default box.

Từ những thông số trên, các default box sẽ được sinh ra theo các bước sau:  
Trước hết, bộ phận này tiến hành sinh ra kích thước của ô chuẩn ứng với mỗi mức kích thước tương ứng bằng công thức:

$$s_k = s_{min} + \frac{s_{max} - s_{min}}{m - 1}(k - 1) \quad (1)$$

<sup>5</sup>Thuật ngữ tiếng Anh: Sliding Window

<sup>6</sup>Tiếng việt: Ô chuẩn

<sup>7</sup>Tiếng việt: Ô Bọc

<sup>8</sup>Thuật ngữ tiếng Anh: Convolution layer

<sup>9</sup>Convolutional neural network

<sup>10</sup>Thuật ngữ tiếng Anh: Ground truth

<sup>11</sup>Thuật ngữ tiếng Anh: Scale

<sup>12</sup>Tiếng việt: Tỉ lệ diện mạo



Trong đó  $s_{min}$  và  $s_{max}$  là kích thước nhỏ nhất và lớn nhất của default box,  $m$  là số lượng feature map. Việc chọn các thông số này khá quan trọng vì nó sẽ ảnh hưởng đến các ký tự có thể nhận diện được sau này, các kích thước cần được chọn sao cho không quá lớn cũng như không quá nhỏ đối với những đối tượng được kỳ vọng nhận diện được.

Ứng với mỗi mức kích thước:

- Mỗi default box đều được liên kết với một điểm ảnh trên feature map, ta cũng đã tính được danh sách khoảng cách trọng tâm theo đề cập ở trên. Từ đó ta dễ dàng tính được tọa độ của các trọng tâm của các default box trong ảnh theo công thức:

$$(x, y) = \left( \left( i + \frac{1}{2} \right) \times step, \left( j + \frac{1}{2} \right) \times step \right) \quad (2)$$

Trong đó:  $x, y$  là tọa độ của các trọng tâm;  $i, j$  là các số nguyên dương nằm trong khoảng từ 0 đến kích thước feature map đang xét và  $step$  là khoảng cách giữa hai trọng tâm liền kề trong mức kích thước đang xét.

- Sau khi có được tọa độ của các trọng tâm, bộ phận mã hóa tiến hành sinh các default box ứng với mỗi vị trí trọng tâm:

- Tạo một default box có kích thước  $s_k$  là kích thước ứng với mức kích thước  $k$  hiện tại.
- Tạo một default box có kích thước bằng  $\sqrt{s_k \times s_{k+1}}$ <sup>13</sup>.
- Với mỗi phần tử trong danh sách aspect ratio tương ứng, ta tạo một default box có kích thước chiều rộng:

$$w = s_k \times \sqrt{\text{aspect\_ratio}} \quad (3)$$

và chiều cao:

$$h = s_k \div \sqrt{\text{aspect\_ratio}} \quad (4)$$

với  $\text{aspect\_ratio}$  là aspect ratio đang xét.

Như vậy, với mỗi vị trí được chọn để đặt trọng tâm các default box, bộ phận mã hóa sẽ sinh ra  $2 + n$  default box với  $n$  là số aspect ratio ứng với mức kích thước hiện tại. Kết quả của quá trình sinh default box là một tập hợp nhiều vector. Mỗi vector ứng với một default box mang thông tin tọa độ trọng tâm, chiều dài, chiều rộng của default box.

## Matching<sup>14</sup>

Sau khi hoàn tất sinh default box, bộ phận encoder có nhiệm vụ matching dữ liệu thô là các bounding box<sup>15</sup> từ ground truth<sup>16</sup> qua các default box vừa được tạo. Kết quả là ta sẽ thu được một tập nhiều vector, mỗi vector đại diện cho mỗi default box chứa thông tin gồm tọa độ trọng tâm, kích thước của default box và nhãn của default box đó là gì (có thể là phần nền - không có gì cả, hoặc một ký tự nào đó cần được nhận diện).

Việc matching được thực hiện bằng cách so sánh từng default box với từng bounding box trong ground truth bằng cách tính chỉ số Jaccard[7], nếu chỉ số này thỏa điều kiện thì default box đó được gán nội dung là ký tự chứa trong bounding box phù hợp nhất, nếu không thì default

<sup>13</sup>Do dùng công thức này nên với  $n$  feature map thì cần sinh ra  $n + 1$  giá trị  $s_k$  trong khi công thức bên trên chỉ giải quyết được  $n$  giá trị  $s_k$  ứng với  $n$  mức kích thước. Để giải quyết vấn đề này, ta có thể thêm một giá trị  $s_0$  phù hợp

<sup>14</sup>Thuật ngữ tiếng Anh: Matching

<sup>15</sup>Thuật ngữ tiếng Anh: Bounding box

<sup>16</sup>Thuật ngữ tiếng Anh: Ground truth

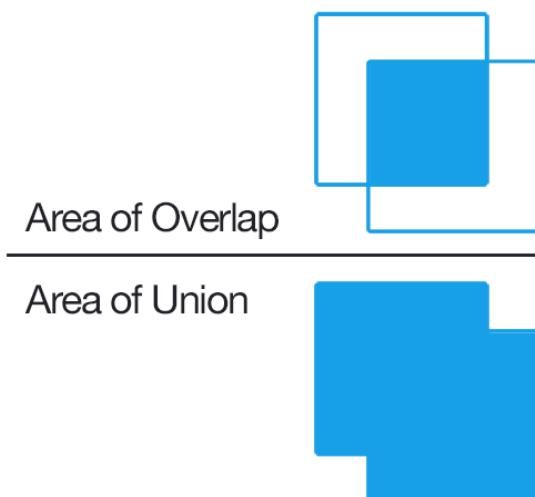


box được gán nội dung là "nền".

Chỉ số Jaccard[7] giữa hai bounding box được tính bằng công thức:

$$J(A, B) = \frac{A \cap B}{A \cup B} \quad (5)$$

Trong đó, A, B lần lượt là phần ảnh mà bounding box A và bounding box B chiếm. Công thức này mang ý nghĩa hai bounding box có kích thước càng tương đồng và phần trùng nhau càng nhiều thì có chỉ số Jaccard[7] càng gần giá trị 1.



**Hình 1:** Mô phỏng chỉ số Jaccard[7]

Các bounding box này có bản chất như một "hệ quy chiếu" cho các bounding box, sau quá trình matching, các bounding box ground truth được tính độ lệch so với các default box tương ứng và khi huấn luyện, mạng SSD[3] có nhiệm vụ dự đoán các bounding box theo các độ lệch đó. Giá trị hàm lỗi được tính dựa trên độ lệch của bounding box ground truth và độ lệch của bounding box dự đoán được. Cách tính độ lệch sẽ được đề cập rõ hơn ở phần tính giá trị lỗi.

## 2.2 Bộ phát hiện - phân loại

Đây là bộ phận có nhiệm vụ trích đặc trưng từ ảnh thô đầu vào, tạo ra các feature map để đưa vào quá trình phân loại ký tự.

### Bộ phận trích đặc trưng (Hay mạng cơ sở<sup>17</sup>)

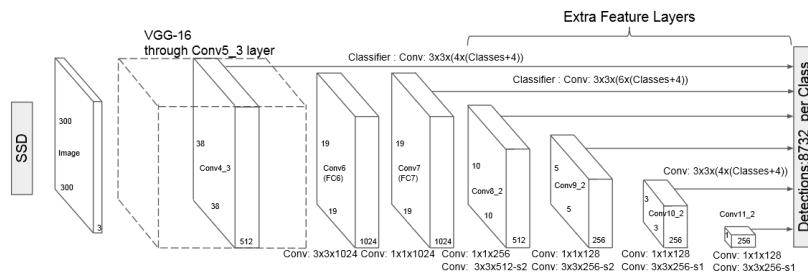
Mạng cơ sở của SSD[3] là một mạng nơron tích chập bất kỳ, có thể là VGG[9] hoặc lenet[10],... Hầu hết những mạng này có mục đích phân loại<sup>18</sup>, vì vậy ở phía cuối mạng thường có các lớp liên kết đầy đủ nhằm giảm kích thước feature map và tạo ra dữ liệu đầu ra có kích thước phù hợp (bằng với số nhãn cần dự đoán). Khi kết hợp với mạng SSD[3], các lớp liên kết đầy đủ này sẽ được thay thế bằng các lớp tích chập và song hành là các bước trích ra feature map để đưa vào lớp Multibox [11], chi tiết phần này sẽ được đề cập ở mục tiếp theo.

<sup>17</sup>Thuật ngữ tiếng Anh: Base Network

<sup>18</sup>Thuật ngữ tiếng Anh: Classification

## Phân loại

Bộ phận phân loại có nhiệm vụ đưa ra dự đoán về vị trí và nhãn của các ký tự có trong ảnh. Để làm được điều đó, bộ phân loại phải sinh ra các feature map ứng với từng mức kích thước khác nhau, các feature map đó được đưa vào lớp Multibox để sinh ra các vị trí dự đoán và nhãn tương ứng. Các feature map được lấy sau một (một số) lớp tích chập. Danh sách feature map có được ở giai đoạn sinh default box chính là được lấy từ đây.



**Hình 2:** Cấu tạo của một mạng SSD[3]

Sau khi có được các feature map, lớp Multibox sẽ tiến hành đưa ra dự đoán về bounding box và nhãn gắn với bounding box đó. Mỗi feature map sử dụng một tập hợp các lớp tích chập sẽ cho ra một số lượng dự đoán nhất định. Với mỗi vị trí trên feature map được kernel trượt qua, thì một dự đoán về ký tự được sinh ra. Kernel có kích thước:

$$3 \times 3 \times (n \times (\text{classes} + 4)) \quad (6)$$

Với  $n$  là số lượng aspect ratio với fearture map tương ứng và  $\text{classes}$  là số lượng nhãn cần nhận diện, số 4 trong công thức đại diện cho 4 thông số về vị trí của bounding box (tọa độ trọng tâm và kích thước). Do kích thước của các kernel không đồng nhất (vì số lượng aspect ratio ứng với từng feature map có thể khác nhau), nên để tạo ra dữ liệu đầu ra phù hợp với các default box đã tạo từ trước thì các vector dự đoán sẽ được sắp xếp lại sao cho dữ liệu đầu ra chỉ có  $\text{classes} + 4$  kênh và vị trí của các vector phải tương ứng với vị trí các default box đã sinh từ trước.

### 2.3 Tính giá trị lỗi

Giá trị lỗi được tính bằng tổng có trọng số giữa lỗi về vị trí<sup>19</sup> và lỗi về độ tin cậy<sup>20</sup>

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \quad (7)$$

Trong đó:

- $x$  biểu thị các phép matching, cụ thể  $x_{ij}^p$  biểu thị phép matching giữa default box thứ  $i$  với bounding box thứ  $j$  đối với nhãn  $p$ . Vì vậy  $x_{ij}^p$  có tập giá trị  $\{0, 1\}$ .
- $c$  biểu thị nhãn kỳ vọng cho phép matching đang xét.
- $l$  biểu thị bounding box dự đoán.
- $g$  biểu thị bounding box ground truth.

Hàm lỗi về vị trí có thể được biểu diễn bằng công thức:

$$L_{loc}(x, l, g) = \sum_{i \in Pos}^N \sum_{m \in \{cx, cy, w, h\}} x_{ij}^k \text{smooth}_{L1}(l_i^m - \hat{g}_j^m) \quad (8)$$

Trong đó:

<sup>19</sup>Thuật ngữ tiếng Anh: Localization Loss

<sup>20</sup>Thuật ngữ tiếng Anh: Confidence Loss



- $N$  là số lượng phép matching giữa default box và bounding box có ý nghĩa (nhận của phép matching không phải là "nền". Nếu như  $N = 0$  thì ta đặt giá trị lỗi bằng 0.
- Pos là tập hợp các phép matching có ý nghĩa.
- $smooth_{L1}$  (theo [12]) là một hàm tính lỗi được định nghĩa là:

$$L(x) = \begin{cases} 0.5x^2 & \text{if } |x| < 1 \\ |x| - 0.5 & \text{otherwise} \end{cases} \quad (9)$$

•

$$\hat{g}_j^{cx} = (g_j^{cx} - d_i^{cx})/d_i^w \quad (10)$$

Với  $d$  biểu thị cho default box.

•

$$\hat{g}_j^{cy} = (g_j^{cy} - d_i^{cy})/d_i^h \quad (11)$$

•

$$\hat{g}_j^w = \log\left(\frac{g_j^w}{d_i^w}\right) \quad (12)$$

•

$$\hat{g}_j^h = \log\left(\frac{g_j^h}{d_i^h}\right) \quad (13)$$

Bốn công thức  $\hat{g}$  trên chính là độ lệch giữa ô đang xét và default box, mà nhóm đã đề cập ở mục trước.

Vậy ta có thể thấy khi tính giá trị lỗi về vị trí, ta tính dựa trên sai lệch so với default box mà bounding box ground truth ban đầu match được thay vì tính lỗi trực tiếp.

Hàm lỗi về độ tin cậy có thể được tính theo công thức:

$$L_{conf}(x, c) = - \sum_{i \in Pos}^N x_{ij}^p \log(\hat{c}_i^p) - \sum_{i \in Neg} \log(\hat{c}_i^0) N_u \quad (14)$$

Với

$$\hat{c}_i^p = \frac{\exp c_i^p}{\sum_p \exp c_i^p} \quad (15)$$

## 2.4 Bộ giải mã (Decoder)

Bản chất của bộ giải mã khá đơn giản, nó có chức năng chuyển dữ liệu các bounding box mà mạng dự đoán (tức là những bounding box được tính kích thước và tọa độ theo default box mà bounding box đó được match vào) sang dữ liệu tọa độ của ảnh (có gốc tọa độ nằm ở góc trên bên trái và tọa độ nằm trong miền  $[0, 1]$ ). Dữ liệu này cần thiết để trực quan hóa các ký tự mà hệ thống dự đoán và để đưa vào bộ phân tích cú pháp để sinh mã Latex.

## 2.5 Một số vấn đề khác trong mạng SSD[3]

### Tăng cường dữ liệu<sup>21</sup>

Ngay trước quá trình matching, ảnh và các bounding box được đi qua một bước gọi là cắt ngẫu nhiên<sup>22</sup>. Ở bước này, ảnh được cắt đi một số phần ngẫu nhiên (có tác dụng giống như phỏng to

<sup>21</sup>Thuật ngữ tiếng Anh: Data Augmentation

<sup>22</sup>Thuật ngữ tiếng Anh: Random Crop



(zoom) ảnh. Điều này giúp làm đa dạng dữ liệu huấn luyện và làm giảm bớt hiện tượng overfit.



### 3 Công trình liên quan

Là một phần quan trọng của hệ thống **Nhận dạng ký tự thuộc thị giác**<sup>23</sup>, nhận dạng **biểu thức toán học**<sup>24</sup> đã được nghiên cứu trong hơn nửa thế kỷ qua. Đã có rất nhiều công trình tiêu biểu giải quyết những vấn đề xung quanh đề tài này. Trong chương này, nhóm sẽ trình bày tóm lược về 4 công trình- là những điểm tham khảo cho hệ thống nhận dạng biểu thức toán học mà nhóm sẽ hiện thực sau này.

#### 3.1 Tổng quan

Nhận dạng biểu thức toán học bao gồm 2 vấn đề chính: **nhận dạng ký tự**<sup>25</sup> và **phân tích cấu trúc**<sup>26</sup>. Hai vấn đề này có thể được xử lý tuần tự hoặc kết hợp. Đối với hướng tiếp cận tuần tự, đầu tiên phân tách<sup>27</sup> ảnh chứa biểu thức toán học thành những mảnh ảnh chỉ chứa các ký tự và nhận dạng chúng. Sau đó phân tích cấu trúc 2 chiều của biểu thức toán học dựa trên kết quả phân tách và nhận dạng ký tự ở bước trước đó. Khác với phương pháp xử lý tuần tự, xử lý kết hợp mong muốn thực hiện 2 quá trình nhận dạng ký tự và phân tích cấu trúc một cách đồng thời. Quan tâm đến cách giải quyết vấn đề nhận dạng biểu thức toán học theo hướng tuần tự, nhóm có liệt kê một số công trình của các tác giả và chỉ ra phương pháp họ sử dụng cho từng giai đoạn cụ thể. Thống kê này được rút trích từ [13].

| Tác giả               | Nhận dạng ký tự   | Phân tích cấu trúc              |
|-----------------------|---|---------------------------------|
| P.A.Chou              | Template matching   | Stochastic context-free grammar |
| M.Okamoto             | Recursive projection và Template matching                       | Không parsing                   |
| J.Ha                  | X-Y cut và mạng nơ-ron  | Expression tree                 |
| R.J.Fateman           | Template matching dựa trên khoảng cách Hausdorff                | Recursive descent parser        |
| H.-J.Lee và J.-S.Wang | Kỹ thuật rút trích đặc trưng và giải thuật nearest-neighborhood | Expression Tree                 |

Bảng 1: Một số công trình tiêu biểu về nhận dạng biểu thức toán học trước năm 2000 và các phương pháp được sử dụng.

Dựa trên quan điểm cá nhân, nhóm có một số đánh giá:

- Với các phương pháp phân tách sử dụng phép chiếu như X-Y cut sẽ trở nên khó khăn đối với các ký tự dính liền nhau hoặc những ký tự đặc biệt như dấu cản.
- Với phương pháp phân tích cấu trúc sử dụng cấu trúc cây đòi hỏi phải có kiến thức để xây dựng các luật sinh và có giải thuật parse cây phù hợp để có thể phù hợp với cách đọc biểu thức từ trái sang phải cũng như khắc phục được một số lỗi sai liên quan đến ngữ pháp.

#### 3.2 Công trình tham khảo

Dưới đây là những công trình mà nhóm tham khảo trực tiếp để hoàn thành luận văn của mình.

<sup>23</sup>Thuật ngữ tiếng Anh: Optical Character Recognition, viết tắt OCR.

<sup>24</sup>Thuật ngữ tiếng Anh: Mathematical Expression, viết tắt ME.

<sup>25</sup>Thuật ngữ tiếng Anh: Symbol recognition.

<sup>26</sup>Thuật ngữ tiếng Anh: Structural analysis.

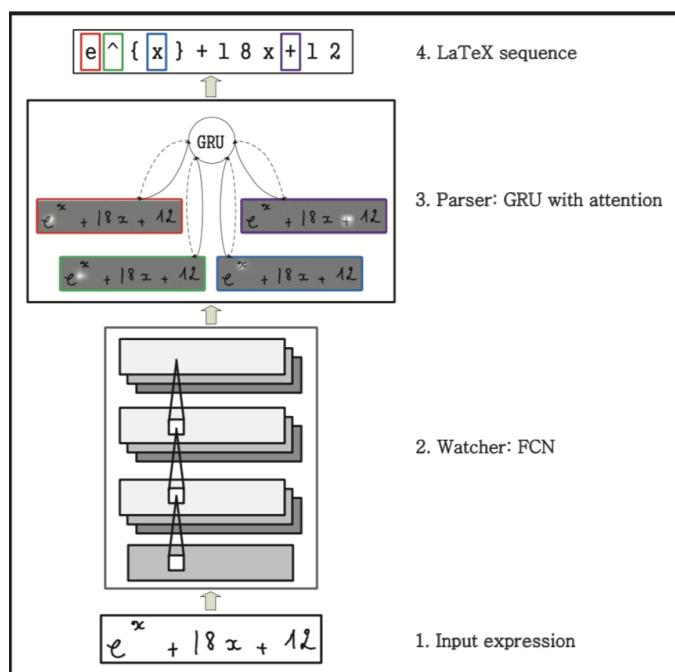
<sup>27</sup>Thuật ngữ tiếng Anh: Segmentation

### 3.2.1 Watch, Attend and Parse: An End-to-end Neural Network Based Approach to Handwritten Mathematical Expression Recognition[4]

Để tránh những vấn đề nảy sinh liên quan đến quá trình phân tách và nhận dạng ký tự và việc phải định nghĩa trước những luật ngữ pháp trong quá trình phân tích cấu trúc theo hướng tiếp cận tuần tự, nhóm tác giả của công trình này đã đề xuất một phương pháp theo hướng kết hợp mà ở đó họ đồng thời xử lý cả hai quá trình cơ bản của vấn đề nhận dạng biểu thức toán học. Với phương pháp này, họ xây dựng một mạng tích chập dùng để mã hoá ảnh đầu vào tạo ra các đặc trưng- quá trình này được gọi là encode, tiếp theo đó là quá trình decode- những đặc trưng này qua mạng RNN, kết hợp với cơ chế attention[4] mà nhóm tác giả giới thiệu sẽ tạo ra được các chuỗi Latex của ảnh biểu thức đầu vào. Cụ thể, kiến trúc mạng của họ gồm 2 phần được đặt tên là Watcher- ứng với quá trình encode và Parser- ứng với quá trình decode.

- **Watcher** thực chất là một mạng nơ-ron tích chập đầy đủ-FCN- với chỉ gồm các lớp tích chập và pooling. Watcher nhận input là một bộ gồm 9 ảnh bao gồm 1 ảnh nhị phân của biểu thức và ảnh 8 hướng của ảnh này. Đầu ra của Watcher là những vector đặc trưng tương ứng với từng pixel trong ảnh.
- **Parser** là kiến trúc mạng GRU[14], nhận kết quả trả ra của Watcher như input, kết hợp với cơ chế attention để sinh ra chuỗi Latex. Cụ thể, cơ chế attention giúp Parser xác định đúng vùng ảnh để tính toán cho ra ký tự Latex qua từng vòng lặp.

Dưới đây là mô hình trực quan của phương pháp này.



**Hình 3:** Hình minh họa các bước thực hiện của phương pháp Watch, Attend and Parser.

### 3.2.2 Context-aware Recognition[5]

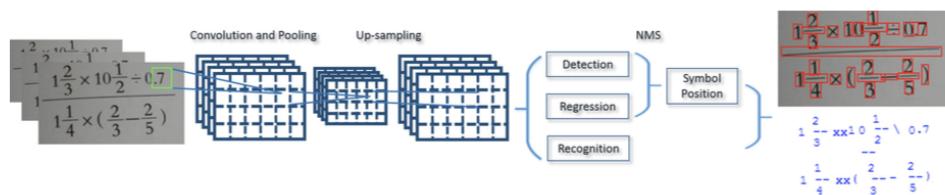
Nhận thấy nhược điểm của phương pháp nhận dạng theo hướng tuần tự là thực hiện hai quá trình nhận dạng ký tự và phân tích cấu trúc một cách độc lập, do đó thông tin cấu trúc của biểu thức không được đưa vào quá trình nhận dạng. Như vậy sẽ dẫn đến việc lỗi tích luỹ qua từng giai đoạn[5]. WenHao He cùng các cộng sự của ông đã đề xuất một phương pháp dựa trên CNN, kết hợp với cách học đa nhiệm vụ<sup>28</sup>, cố gắng kết hợp hai giai đoạn chuẩn của quá trình nhận dạng biểu thức toán lại với nhau.

<sup>28</sup>Thuật ngữ tiếng Anh: Multi-task learning

Phương pháp mà các ông đưa ra đảm bảo thông tin cấu trúc của biểu thức được đưa vào quá trình nhận dạng và thể hiện trong các ma trận đặc trưng<sup>29</sup>[10].

Dối với các phương pháp trước, biểu thức phải được phân tách thành những ký hiệu rồi từng ký hiệu này mới được đưa qua bộ nhận dạng. Một số phương pháp phân tách ký hiệu thường được sử dụng như: phân tích thành phần liên thông<sup>30</sup>, cắt dựa trên các phép chiếu<sup>31</sup>[15]. Tuy nhiên với phương pháp mới này, cả ảnh của biểu thức toán học được đưa qua bộ nhận dạng, chính vì vậy mà thông tin cấu trúc của biểu thức được bảo toàn.

Dưới đây là mô hình học của phương pháp này:



Hình 4: Mô hình học được đề xuất [5].

Ảnh đầu vào sẽ qua một số lớp tích chập, down-sampling và up-sampling để tạo ra một feature map. Feature map này sẽ được gửi đến ba nhiệm vụ, mỗi nhiệm vụ sẽ tạo ra 1 feature map có cùng kích thước với feature map đầu vào. Kiến trúc mạng của 3 nhiệm vụ này là sự kết hợp của các module inception[5].

Giả sử một điểm  $i$  được cho đặt tại toạ độ  $(w_i, h_i)$  của feature map được tạo ra bởi các nhiệm vụ.

- **Nhiệm vụ phát hiện** (Detection task) sẽ cho ra một con số  $s$  thể hiện độ tin cậy rằng một ký hiệu được đặt tại  $i$ .
- **Nhiệm vụ hồi quy** (Regression task) cho ra một vector 4 chiều  $x_1, y_1, x_2, y_2$  thể hiện thông tin về bounding box của ký hiệu được đặt tại  $i$ .
- **Nhiệm vụ nhận dạng** (Recognition task) gán nhãn cho ký hiệu đặt tại  $i$  cùng với xác suất của nhãn đó.

Như vậy nhiệm vụ phát hiện và hồi quy được thiết kế để định vị trí của ký hiệu trong biểu thức toán học, nhiệm vụ nhận dạng quyết định xem đó là ký hiệu gì.

Phương pháp nhận dạng như trên có thể giải quyết cả những trường hợp là thách thức đối với các phương pháp phân tách và nhận dạng ký hiệu trước đây, cụ thể đó là vấn đề gom nhóm ký hiệu đối với các ký hiệu nhiều phần nhỏ, tách những ký hiệu bị dính nhau.

### 3.2.3 QAK[1]

QAK là tên dự án được thực hiện bởi nhóm sinh viên khoa 2011 cũng về đề tài nhận dạng biểu thức toán học.

Phương pháp nhóm sinh viên này xây dựng dựa trên nghiên cứu chính từ hai công trình của Aderson[16] và Zanibbi[6]. Do đó, qua trình nhận dạng biểu thức toán học vẫn đi theo 2 bước chuẩn đó là phân tách ký hiệu và phân tích cấu trúc như đã trình bày ở mục 3.2.2.

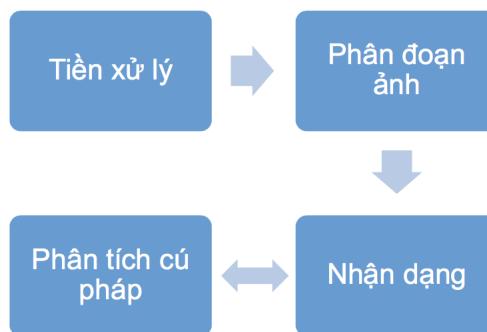
Dưới đây là mô hình phương pháp mà nhóm đã đề xuất:

- Trong bước tiền xử lý, nhóm áp dụng một số kỹ thuật trong xử lý ảnh như: loại nhiễu, ảnh nhị phân,... để tăng cường chất lượng ảnh, hỗ trợ cho bước phân đoạn.

<sup>29</sup>Thuật ngữ tiếng Anh: Feature map

<sup>30</sup>Thuật ngữ tiếng Anh: Connected components

<sup>31</sup>Thuật ngữ tiếng Anh: Projection cutting



**Hình 5:** Quy trình nhận dạng.

- Trong bước phân đoạn ảnh, nhóm dùng kỹ thuật chính là cắt theo hình chiếu[15] và phân tích thành phần liên thông[15] cho toán tử lấy cẩn. Mục tiêu của giai đoạn này là phân tách ảnh chứa biểu thức toán học ban đầu ra thành các mảnh ảnh, mỗi mảnh chỉ chứa 1 ký hiệu toán học. Ngoài ra, ở bước này một cấu trúc cây được xây dựng lưu thông tin của các mảnh ảnh, hỗ trợ cho quá trình phân tích ngữ pháp sau này.
- Ở bước nhận dạng, nhóm sinh viên khoa 2011 đã sử dụng một kiến trúc mạng tương tự Lenet-5[10].
- Với phân tích cú pháp, nhóm sử dụng tập luật **văn phạm phi ngữ cảnh**<sup>32</sup> do chính nhóm đề xuất để giới hạn kết quả đầu ra của quá trình nhận dạng, từ đó tăng khả năng nhận dạng đúng biểu thức.

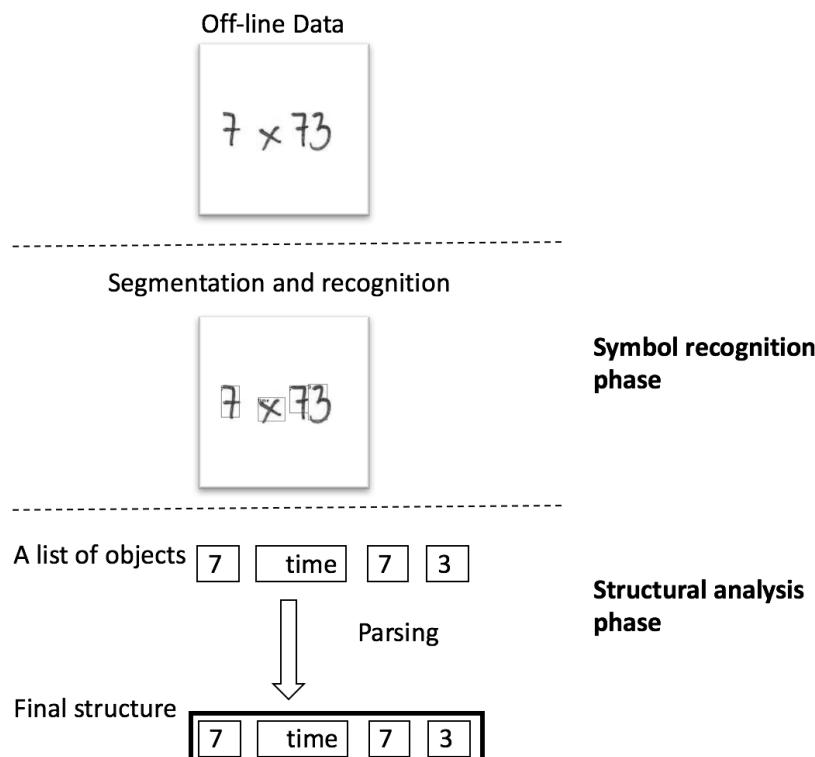
<sup>32</sup>Thuật ngữ tiếng Anh: Context-free grammar



## 4 Mô hình đề xuất

### 4.1 Tổng quan

Nhóm chọn hướng tiếp cận là thực hiện tuần tự hai quá trình nhận dạng ký tự và phân tích cấu trúc để giải quyết đề tài. Quy trình thực hiện như hình bên dưới.



Hình 6: Mô hình phương pháp.

- Ở bước nhận dạng ký tự, ảnh đầu vào là ảnh xám sẽ qua mạng SSD[3] để tạo ra một danh sách các nhãn và bounding box được kỳ vọng là tương ứng với các ký tự có trong ảnh.
- Với giai đoạn phân tích cấu trúc, từ danh sách nhãn và bounding box trả ra từ giai đoạn trước, sử dụng bộ phân tích cấu trúc DRACULAE [6] để sinh ra chuỗi Latex.

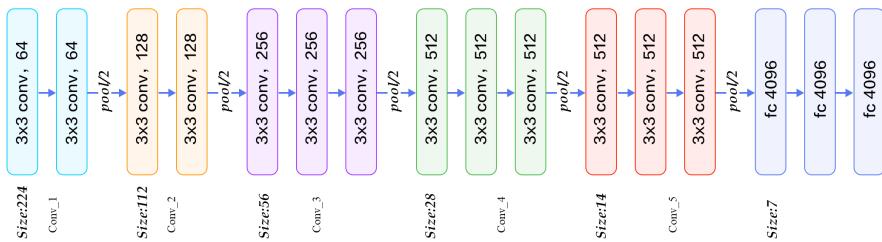
Việc lựa chọn SSD cho giai đoạn nhận dạng ký tự là để khắc phục hạn chế của cách tiếp cận tuần tự khi thông tin cấu trúc ảnh vẫn được lưu giữ qua thông tin các bounding box của ký tự.

### 4.2 Giai đoạn nhận diện ký tự

#### 4.2.1 Mạng cơ sở

Về cấu trúc của SSD[3], nhóm đã sử dụng mạng VGG16[8] cho phần mạng cơ sở. Cấu trúc của mạng VGG16[8] gồm tổ hợp các lớp tích chập và lớp pooling xếp chồng lên nhau, ở cuối mạng có các lớp liên kết đầy đủ <sup>33</sup> để giảm kích thước tensor và cuối cùng là xuất ra vector có số chiều phù hợp (có số chiều bằng với số lớp đối tượng cần dự đoán).

<sup>33</sup>Thuật ngữ tiếng Anh: Fully connected layer

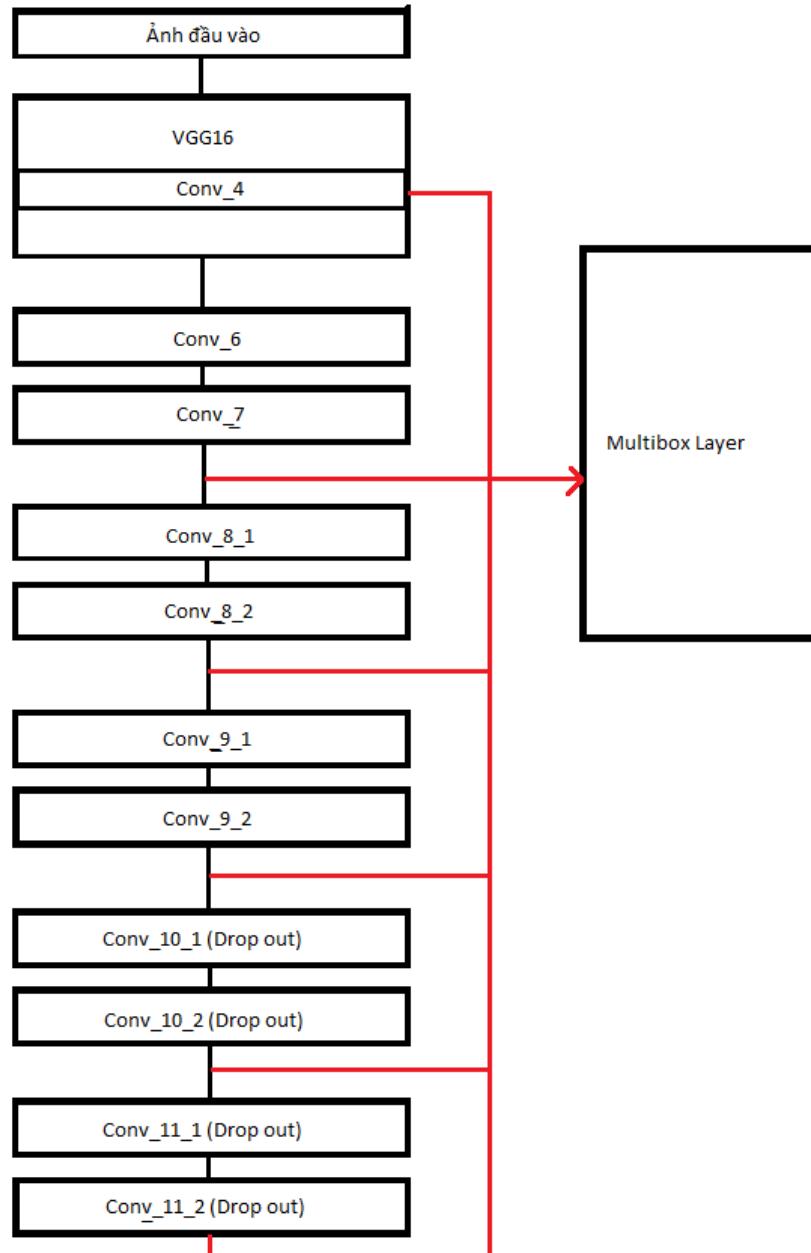


**Hình 7:** Cấu tạo mạng VGG16/8

Như hình vẽ bên trên, các lớp tích chập và lớp pooling được gom lại thành các cụm, sau hai đến ba lớp tích chập là một lớp pooling. Số kênh của tensor chạy trong mạng được nâng dần từ 64 lên 512 về phía cuối mạng. Khi đưa mạng cơ sở vào mạng SSD[3], các lớp liên kết đầy đủ sẽ được bỏ đi và bộ phận phân loại sẽ được chèn vào vị trí tương ứng. Khi đó, ta cần phải chuyển model của mạng VGG[8] sang mạng SSD[3].

#### 4.2.2 Thân mạng SSD[3]

Sau khi ảnh đi qua mạng cơ sở và sinh ra một feature map, dữ liệu này sẽ tiếp tục đi qua các lớp tích chập và pooling kết hợp với việc trích ra feature map ở một số vị trí nhất định để đưa vào lớp multibox. Mô hình cụ thể của phần thân mạng SSD[3] nhóm để xuất được thể hiện ở ảnh dưới.



Hình 8: Sơ đồ các lớp trong mạng SSD[3]

Trong đó Khối VGG166 là phần mạng VGG16[8] đã được lược bỏ các lớp liên kết đầy đủ, khối Conv\_4 nằm trong khối VGG16 này do Conv\_4 trong ảnh là lớp tích chập thứ 10 trong mạng VGG[8]. Các khối còn lại được thể hiện trong bảng sau:



| Khối      | Số kênh đầu vào | Số kênh đầu ra | Kích thước nhân | Chèn thêm (Padding) | Bước (Stride) |
|-----------|-----------------|----------------|-----------------|---------------------|---------------|
| Conv_6    | 512             | 1024           | 3               | 1                   | 1             |
| Conv_7    | 1024            | 1024           | 3               | 6                   | 1             |
| Conv_8_1  | 1024            | 256            | 1               | 0                   | 1             |
| Conv_8_2  | 256             | 512            | 3               | 1                   | 2             |
| Conv_9_1  | 512             | 128            | 1               | 0                   | 1             |
| Conv_9_2  | 128             | 256            | 3               | 1                   | 2             |
| Conv_10_1 | 256             | 128            | 1               | 0                   | 1             |
| Conv_10_2 | 128             | 256            | 3               | 1                   | 2             |
| Conv_11_1 | 256             | 128            | 1               | 0                   | 1             |
| Conv_11_2 | 128             | 256            | 3               | 0                   | 1             |

Bảng 2: Cấu hình các lớp tích chập trong mạng SSD300[3]

Các lớp tích chập đều sử dụng hàm truyền là hàm ReLu[17]. Ở sau các lớp Conv\_10\_1, Conv\_10\_2, Conv\_11\_1, Conv\_11\_2 thì có thêm một lớp dropout[18] với hệ số dropout là 0.2. Giữa một số lớp tích chập và pooling là nơi các feature map được trích ra để thực hiện công đoạn phát hiện, phân loại ký tự (được thể hiện bởi các mũi tên màu đỏ trên hình), cụ thể:

- Ngay sau khối tích chập - pooling 512 thứ nhất trong mạng VGG[8] (hay sau lớp pooling của lớp tích chập thứ 10 của mạng VGG[8] - hay Conv\_4).
- Sau các khối tích chập Conv\_7, Conv\_8, Conv\_9, Conv\_10, Conv\_11.

Đối với mạng SSD300[3], kích thước của các feature map ứng với số dự đoán được thể hiện ở bảng sau:

| Feature map<br>Sinh bởi | Kích thước                 | Aspect Ratio                            | Số dự đoán |
|-------------------------|----------------------------|---|------------|
| Conv_4                  | $512 \times 38 \times 38$  | $\{1, \frac{1}{2}, 2\}$                 | 5776       |
| Conv_7                  | $1024 \times 19 \times 19$ | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 2166       |
| Conv_8                  | $512 \times 10 \times 10$  | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 600        |
| Conv_9                  | $256 \times 5 \times 5$    | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 150        |
| Conv_10                 | $256 \times 3 \times 3$    | $\{1, \frac{1}{2}, 2\}$                 | 36         |
| Conv_11                 | $256 \times 1 \times 1$    | $\{1, \frac{1}{2}, 2\}$                 | 4          |

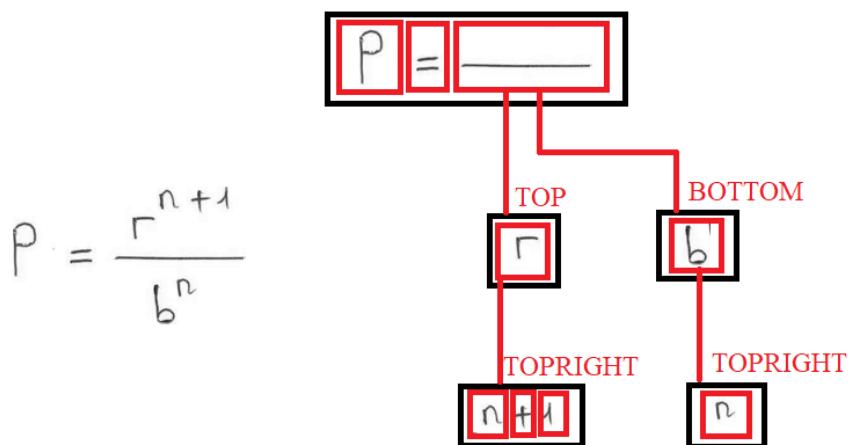
Bảng 3: Thông tin của các feature map trích được từ mạng SSD300[3]

Như vậy, ứng với mỗi ảnh, hệ thống sinh ra 8732 dự đoán. Tương ứng, bộ phận mã hóa của mạng SSD300[3] cũng có những thông số:

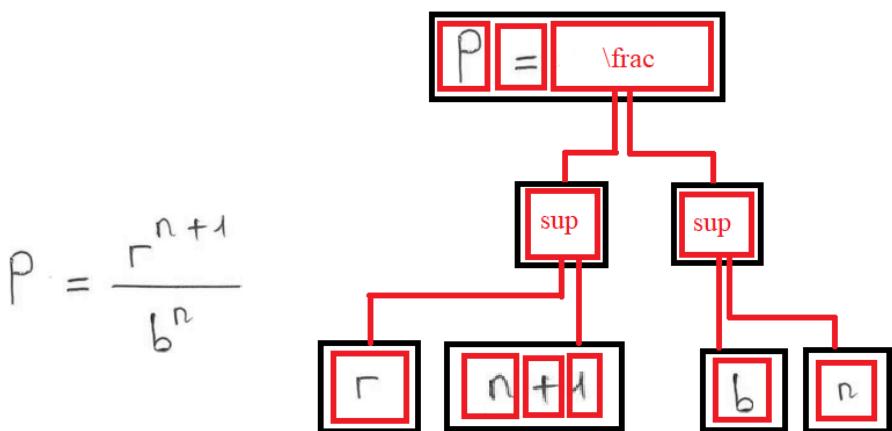
- Danh sách kích thước feature map: (38, 19, 10, 5, 3, 1)
- Danh sách kích thước các ô chuẩn (với  $s_0 = 0.1$ ,  $s_{min} = 0.2$  và  $s_{max} = 0.9$ ): (30, 60, 111, 162, 213, 264, 315)
- Ngưỡng Jaccard[7]: 0.5.

### 4.3 Giai đoạn phân tích cú pháp

Nhóm sử dụng phương pháp phân tích cú pháp DRACULAE được giới thiệu trong [6]. Bộ phân tích cú pháp DRACULAE[6] này có nhiệm vụ chuyển dữ liệu dự đoán là các bounding box và nhãn của ký tự sang dữ liệu biểu thức toán học dạng Latex. Trong quá trình phân tích cú pháp, biểu thức được lưu trữ dưới dạng cây BST[6] là một cấu trúc cây dựa trên các đường cơ sở, các nhánh của cây thể hiện một phân vùng bên trên, bên dưới, ... của nút và là một phân vùng ảnh có đường cơ sở riêng và Lexed - BST[6] là một cấu trúc cây tương tự như BST nhưng thay vì chỉ biểu thị vị trí bên trên, bên dưới thì cây biểu thị quan hệ phân số, chỉ số trên, chỉ số dưới, ...



**Hình 9:** Một cây BST[6] (Các đường màu đỏ biểu thị mối quan hệ nút cha con)



**Hình 10:** Một cây Lexed - BST[6]

#### 4.3.1 Sinh cây BST[6]

Đây là giai đoạn phức tạp nhất và cũng là quan trọng nhất, ở giai đoạn này, dữ liệu đầu ra từ mạng SSD[3] là danh sách các bounding box sẽ được phân tích và xây dựng một cây BST[6]. Công việc này có thể được phân ra thành bốn công đoạn: Tìm ký tự chủ đạo<sup>[6]</sup><sup>34</sup>, xác định những ký tự trên đường cơ sở, tái phân vùng và xử lý nút con.

<sup>34</sup>Thuật ngữ tiếng Anh: Dominate symbol.



Trước khi tiến hành sinh cây, ta cần phải phân loại các ký tự vào những thể loại khác nhau. Theo Zanibbi đã đề xuất cho DRACUALE[6], các ký tự được phân vào các lớp như bảng dưới:

| Lớp ký tự   | Tung độ trọng tâm | Bên dưới                       | Bên trên            | Chỉ số dưới                    | Chỉ số trên         |
|---|-------------------|--------------------------------|---------------------|--------------------------------|---------------------|
| Non-Script<br>Các ký tự phép tính<br>$(+, -, \rightarrow, \dots)$                   | $\frac{1}{2}H$    | $\frac{1}{2}H$                 | $\frac{1}{2}H$      | -                              | -                   |
| Open Bracket<br>Ký tự mở ngoặc<br>$($   | $cH$              | $\min(Y)$                      | $\max(Y)$           | -                              | -                   |
| Root<br>Ký tự căn<br>$\sqrt{\dots}$   | $cH$              | $\min(Y)$                      | $\max(Y)$           | $tH$                           | $H - tH$            |
| Variable Range<br>Ký tự có thể viết<br>ở trên hoặc dưới<br>$\sum, \int, \lim \dots$ | $\frac{1}{2}H$    | $tH$                           | $H - tH$            | $tH$                           | $H - tH$            |
| Ascender<br>Ký tự nổi lên<br>$A..Z, 0..9, b, d, f \dots$                            | $cH$              | $tH$                           | $H - tH$            | $tH$                           | $H - tH$            |
| Descender<br>Ký tự chìm xuống<br>$g, p, y, j, \dots, b, d, f \dots$                 | $H - cH$          | $\frac{1}{2}H + t\frac{1}{2}H$ | $H - t\frac{1}{2}H$ | $\frac{1}{2}H + t\frac{1}{2}H$ | $H - t\frac{1}{2}H$ |
| Centered<br>Ký tự trung tâm<br>$o, n, u, \{, \} \dots$                              | $\frac{1}{2}H$    | $tH$                           | $H - tH$            | $tH$                           | $H - tH$            |

**Bảng 4:** Bảng phân lớp các ký tự cần nhận diện và các ngưỡng xác định phân vùng con

Trong đó:

- Các cột thứ hai trở đi là tọa độ của một số điểm, ngưỡng đặc biệt tính từ góc trái bên dưới của ký tự. Cột "Tung độ trọng tâm" thể hiện tung độ y của trọng tâm của ký tự, cột "bên dưới" và "bên trên" thể hiện ngưỡng trên và ngưỡng dưới của ký tự, các ký tự vượt ngoài ngưỡng đó sẽ được xem là nằm bên trên hoặc nằm bên dưới ký tự đang xét. Cột chỉ số trên, chỉ số dưới thể hiện ngưỡng để được phân vào vùng chỉ số trên, chỉ số dưới. Một ký tự nằm trong khoảng chỉ số dưới đến chỉ số trên của một ký tự khác thì hai ký tự xem là liền kề nhau. Riêng hai lớp Non-script và Open Bracket có giá trị hai cột này không xác định là do những ký tự của lớp này không có chỉ số trên, dưới.
- $H$  là chiều cao của ký tự.
- $c$  (Viết tắt của centroid ratio): là tham số để xác định trọng tâm cho một số ký tự đặc biệt (ví dụ như các ký tự ngoặc hoặc chìm xuống dưới), tùy người viết chữ có thể có những giá trị phù hợp khác nhau.
- $t$  Là tham số ảnh hưởng đến ngưỡng cho chỉ số trên, dưới và chỉ phần bên trên, bên dưới. Tham số này biểu thị sự nhạy cảm đối với những ký tự trên dưới.

Trước khi tiến hành sinh cây BST, danh sách bounding box cần phải được sắp xếp theo chiều tăng dần hoành độ của mép phải bounding box  $x$  (hay cụ thể hơn là  $\min(x)$ ).



### Tìm ký tự chủ đạo[6]

Trong hầu hết trường hợp, ký tự chủ đạo[6] là ký tự nằm bên trái nhất, tuy vậy, ta vẫn cần phải kháng được những trường hợp người dùng viết chữ không chuẩn gây thusat ra ngoài, hoặc những ký tự có dạng như:

$$\sum_{n=100000} a$$

Thì ký tự chủ đạo[6] là  $\Sigma$  nhưng ký tự bên trái nhất lại là  $n$ , DRACULAE[6] được thiết kế để có thể chống lại những trường hợp như vậy.

Để tìm ký tự chủ đạo[6], ta tìm ký tự bên trái nhất dựa trên 3 điều kiện:

- Không bị thống trị bởi ký tự phân số. Kiểm tra bằng cách tìm kiếm một ký tự dấu gạch ngang trong vùng ảnh phía trên, phía dưới.
- Không bị bọc bởi ký tự căn ( $\sqrt{}$ ).
- Nếu ký tự đang xét thuộc lớp Variable Range thì không được phép có ký tự liền kề bên trái.

### Xác định ký tự trên đường cơ sở

Sau khi đã có được ký tự chủ đạo[6], ta cần phải tìm những ký tự liền kề theo chiều ngang với ký tự vừa tìm được, những ký tự này chính là ký tự trên đường cơ sở. Để xác định các ký tự trên đường cơ sở, ta lấy ký tự chủ đạo[6] vừa xác định được để xét:

- Ta phần vùng các ký tự nằm ở vùng bên trên, bên dưới, góc trên bên trái và góc dưới bên trái (không phân vùng các ký tự nằm bên phải) vào nút con của nút đang xét.
- Nếu như ký tự đang xét thuộc lớp Non-script, ký tự đó được đánh dấu nằm trên đường chủ đạo[6], ta quay lại tìm ký tự chủ đạo[6] trong các ký tự còn lại và xác định các ký tự trên đường cơ sở với ký tự chủ đạo[6] đó.
- Ta kiểm tra danh sách các ký tự còn lại xem có ký tự nào liền kề bên phải với ký tự đang xét hay không, nếu có, ta kiểm tra ký tự đó có bị thống trị bởi ký tự khác hay không và quay lại bước 1 với ký tự được xét là ký tự thống trị nhất (hoặc chính ký tự liền kề nhất vừa tìm được đối với trường hợp nó không bị thống trị bởi ký tự nào).
- Nếu hệ thống chạy đến bước này thì có nghĩa là trong các ký tự còn lại, không còn ký tự nào liền kề với ký tự đang xét, vì vậy ta tiến hành phân vùng dựa các ký tự còn lại vào nút con góc trên bên phải (chỉ số trên) và góc dưới bên phải (chỉ số dưới) của nút đang xét.

Sau khi thực hiện bước này lần đầu tiên, ta thu được một cây có ba tầng, tầng đầu chứa nút gốc, tầng thứ hai chứa các ký tự nằm trên đường cơ sở, tầng thứ ba chứa các nút con của các nút chứa ký tự nằm trên đường cơ sở, các nút con này có thể chứa một hoặc nhiều ký tự, chưa là một cây.

### Tái phân vùng

Ở bước trước, nút con được phân vào vùng góc trên bên trái và góc dưới bên trái, điều này không có ý nghĩa đối với ngữ nghĩa của biểu thức toán học và khó khăn trong việc sinh cây Lexed BST[6] sau này. Vì vậy ở bước này, các nút con sẽ được thay đổi vị trí cho phù hợp với ngữ nghĩa của biểu thức. Cụ thể:

- Các nút con thuộc vùng góc trên bên trái và góc dưới bên trái sẽ được phân vùng vào vùng góc trên bên phải và góc dưới bên phải của ký tự trước đó.
- Riêng đối với các ký tự Variable Range, hệ thống thực hiện thêm bước kiểm tra sự liền kề đối với những ký tự trong vùng con để phân vùng phù hợp.



### Xử lý nút con

Tới bước này, ta đã có một cây có ba tầng với tầng thứ hai có phân vùng nút con khá chuẩn xác. Nhiệm vụ bây giờ là phải tiếp tục phân vùng cho các nút con của các ký tự trên đường cơ sở. Việc này được thực hiện bằng cách xem từng nút con là từng vùng ảnh mới riêng biệt và thực hiện lại ba bước trên để thu được các cây con.

Sau quá trình này, ta thu được một cây BST biểu thị vị trí tương đối giữa các nút. Cây này tiếp tục được đi qua [Lexical Pass] để thu được Lexed - BST[6].

#### 4.3.2 Sinh cây Lexed - BST[6]

Sau khi đã thu được cây BST từ danh sách các bounding box, nhiệm vụ còn lại là không quá phức tạp. Trước khi tạo được chuỗi Latex, ta cần phải sinh ra được một cây Lexed - BST[6], cây này chứa các thông tin như "nhóm ký tự" và cấu trúc biểu thức:

- Nhóm ký tự: Ban đầu, các ký tự được gom thành từng nhóm có nghĩa (Ví dụ các ký tự "s", "i", "n" thì sẽ được gộp lại thành một ký tự "sin" duy nhất).
- Ta sử dụng các luật khác nhau để sinh ra cây Lexed - BST[6].

#### Luật sinh

Để chuyển từ cây BST[6] sang cây lexed - BST[6], nhóm có đề xuất một số luật sinh:  
Gọi ký tự đang xét là  $A$

- Nếu cây BST  $B$  nằm trong nút con góc phải bên dưới của  $A$  thì sinh ra một nút sub( $A, B$ )
- Nếu cây BST  $B$  nằm trong nút con góc phải bên trên của  $A$  thì sinh ra một nút sup( $A, B$ )
- Nếu cây BST  $B$  nằm trong nút con bên trong của  $A$  và  $A$  là ký tự căn ( $\sqrt{ }$ ) thì sinh ra một nút sqrt( $B$ )
- Nếu  $A$  có nút con bên trên hoặc bên dưới (hoặc cả hai), thì tùy thuộc vào ký tự  $A$  mà ta sinh ra các nút khác nhau (Ví dụ như các ký tự  $\sum, \lim, \prod, \int, \rightarrow$  và quan trọng nhất là dấu gạch ngang trong phân số)

Sau khi sinh được cây Lexed - BST[6], hệ thống sẽ duyệt qua cây theo chiều tiền thư tự để sinh mã Latex.



## 5 Hiện thực, đánh giá

Để xây dựng hệ thống nhận diện biểu thức toán học viết tay, nhóm cần hoàn thành ba công việc chính bao gồm thu thập dữ liệu, huấn luyện mạng SSD và xây dựng chương trình sinh mã latex.

### 5.1 Chuẩn bị dữ liệu

Việc chuẩn bị dữ liệu bao gồm xây dựng tập ký tự viết tay thường dùng trong Toán học và xây dựng bộ những ảnh biểu thức Toán học viết tay sử dụng những ký hiệu đó.

#### 5.1.1 Xây dựng tập ký tự

Như đã đề cập trong chương 1, nhóm kế thừa tập ký hiệu Toán viết tay bao gồm chữ cái, chữ số, các ký hiệu Hy Lạp của nhóm sinh viên khoá 2011[1].

Một số thông tin về tập dữ liệu<sup>35</sup>:

- Số người tham gia: khoảng 20
- Số lượng ký hiệu: 46197
- Số lượng nhãn: 88
- Các nhãn ký hiệu trong tập dữ liệu: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, (, ), +, -, \*, /, =,  $\int$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$ ,  $\epsilon$ ,  $\theta$ ,  $\lambda$ ,  $\mu$ ,  $\pi$ ,  $\rho$ ,  $\sigma$ ,  $\phi$ ,  $\omega$ ,  $\Delta$ ,  $\Pi$ ,  $\Sigma$ ,  $\Phi$ ,  $\Omega$ .

Tuy nhiên, nhóm đã bổ sung thêm 18 nhãn ký hiệu vào tập dữ liệu này dựa trên cách thu thập được giới thiệu trong luận văn của nhóm sinh viên khoá 2011[1].

Thông tin mô tả tập ký hiệu được sử dụng cho đề tài:

- Số người tham gia: 36
- Số lượng ký hiệu: 52353
- Số lượng nhãn: 106
- Các nhãn ký hiệu trong tập dữ liệu: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, a, b, c, d, e, f, g, h, i, j, k, l, m, n, o, p, q, r, s, t, u, v, w, x, y, z, A, B, C, D, E, F, G, H, I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W, X, Y, Z, (, ), +, -, \*, /, =,  $\int$ ,  $\alpha$ ,  $\beta$ ,  $\gamma$ ,  $\delta$ ,  $\epsilon$ ,  $\theta$ ,  $\lambda$ ,  $\mu$ ,  $\pi$ ,  $\rho$ ,  $\sigma$ ,  $\phi$ ,  $\omega$ ,  $\Delta$ ,  $\Pi$ ,  $\Sigma$ ,  $\Phi$ ,  $\Omega$ ,  $\sin$ ,  $\cos$ ,  $\tan$ ,  $\log$ ,  $\lim$ ,  $\sqrt$ ,  $\rightarrow$ ,  $\geq$ ,  $\leq$ ,  $\forall$ ,  $\exists$ ,  $\in$ , !, ..,  $\div$ ,  $\neq$ ,  $\infty$ .

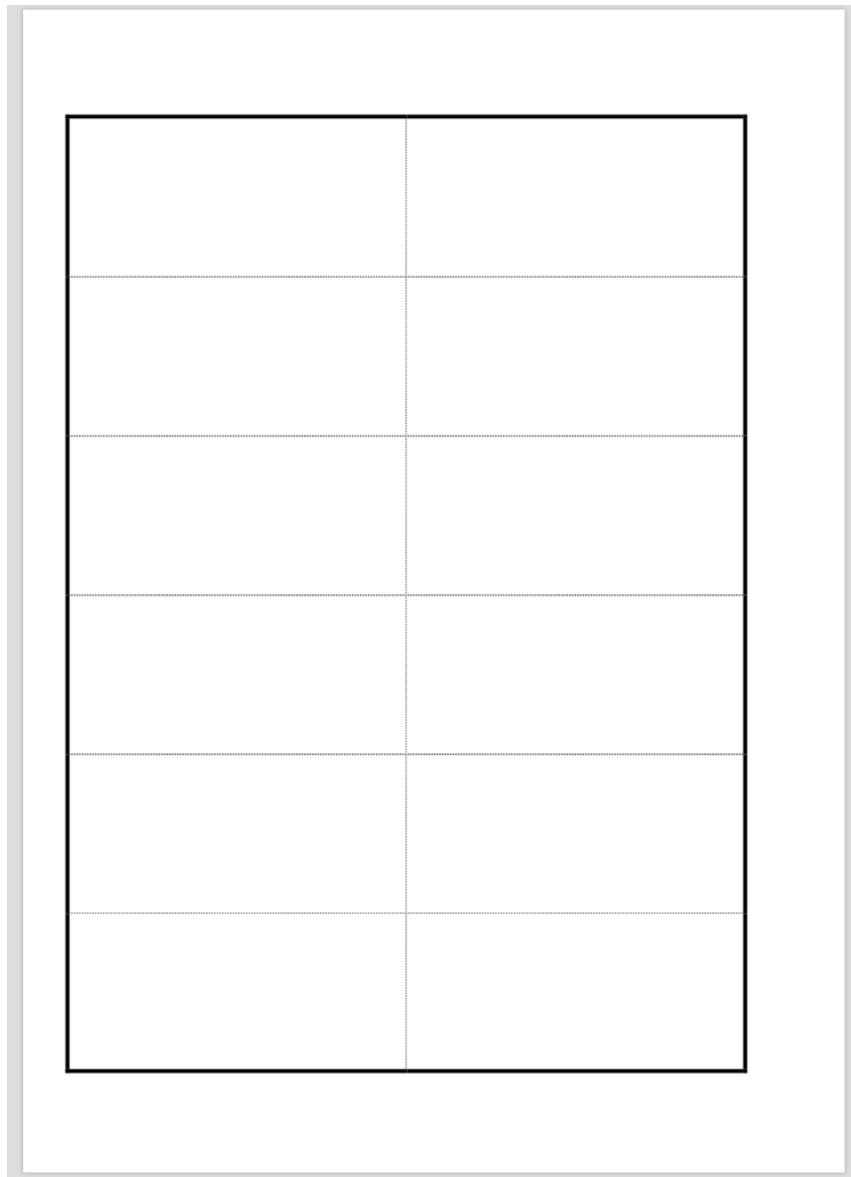
#### 5.1.2 Xây dựng tập biểu thức

Để phục vụ quá trình huấn luyện của mạng SSD được nêu trong Chương 4, ngoài tập ký tự nhóm cần phải xây dựng thêm tập ảnh các biểu thức Toán học viết tay.

##### 5.1.2.a Quá trình thu thập dữ liệu

- Tạo biểu mẫu để thu dữ liệu. Biểu mẫu này có 6 hàng và 2 cột như hình bên dưới.

<sup>35</sup>Sử dụng thông tin được nêu trong luận văn [1].



**Hình 11:** Biểu mẫu để thu thập dữ liệu.

- Chuẩn bị bộ biểu thức làm gợi ý cho người viết. Các biểu thức này được sao chép từ tập dữ liệu CROHME 2014[19].



|  |  |
|--|--|
| 1. $x = r \cos \theta$   | 1. $1 + 2 + \dots + n = \frac{n(n+1)}{2}$  |
| 2. $\cos \theta = \frac{x}{\sqrt{x^2 + y^2}}$                    | 2. $\frac{1}{4} + \left(\frac{1}{4}\right)^2 + \left(\frac{1}{4}\right)^3 + \dots = \frac{1}{3}$ |
| 3. $c^2 = a^2 + b^2 - 2ab \cos C$                                | 3. $H \in P$   |
| 4. $a = b \cos C + c \cos B$                                     | 4. $P^n \neq P_{\bar{n}}$  |
| 5. $P = \sqrt{a^2 + b^2 - 2ab \cos A}$                           | 5. $G \in X$   |
| 6. $\frac{1}{r^2} = \frac{1}{(R-m)^2} + \frac{1}{(R+m)^2}$       | 6. $M < N$   |
| 7. $\mathbf{x} = x \cos \theta + y \sin \theta$                  | 7. $\forall x \in E \exists y \in E \alpha R y$  |
| 8. $b = c \sin B = c \cos A$                                     | 8. $\exists M, R > 0$  |
| 9. $S = \left( \sum_{i=1}^n \theta_i - (n-2)\pi \right) r^2$     | 9. $\psi(x)$   |
| 10. $y^4 + y + 1 = 0$  | 10. $x = \log_a b$   |
| 11. $a^x q^y = a^{x+y}$<br>$\oplus$ $a^x q^y = a^{x+y}$<br>Nhưng | 11. $w_n = aq^{n-n_0}$   |
|  | 12. $151 + 143 : 97$   |

Hình 12: Ví dụ mẫu biểu thức gợi ý cho người viết.

- Người tham gia viết sẽ nhận được biểu mẫu và công thức gợi ý. Sau đó họ sẽ viết lại những công thức này lên tờ biểu mẫu bằng chính nét chữ thường ngày của họ.

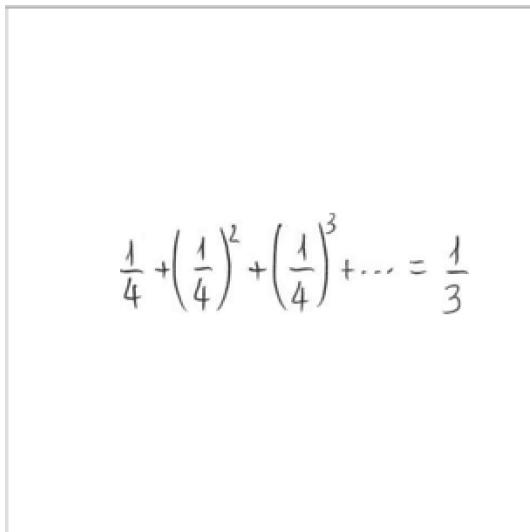


|   |   |
|---|---|
| Đoàn Thị Ngọc Nhung   |   |
| $1+2+\dots+n = \frac{n(n+1)}{2}$  | $\forall x \in E \exists y \in E x R y$ |
| $\frac{1}{4} + \left(\frac{1}{4}\right)^2 + \left(\frac{1}{4}\right)^3 + \dots = \frac{1}{3}$ | $R > 0$                                 |
| $H \in P$   | $y(x)$                                  |
| $P^\mu P_\mu$   | $x = \log_a b$                          |
| $\sigma \in X$  | $\mu_n = a q^{n-n_0}$                   |
| $M < N$   | $151 + 143 : 97$                        |

**Hình 13:** Ví dụ một mẫu thu thập được từ người tham gia viết.

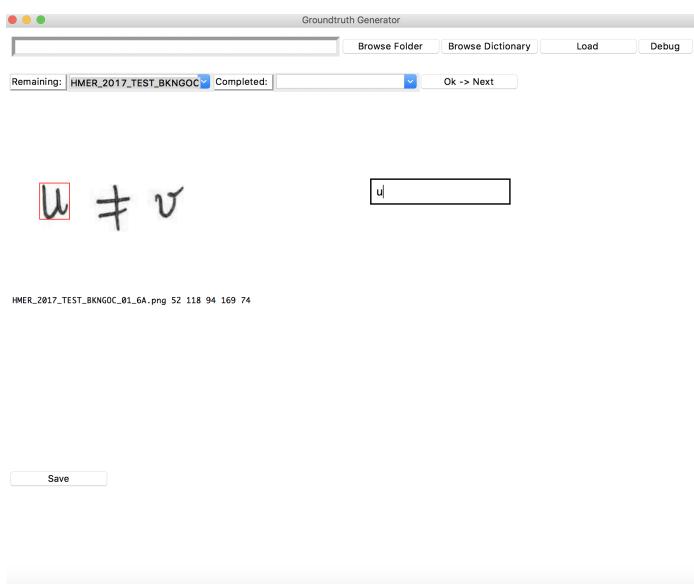
- Tập hợp lại các biểu mẫu sau khi cho người tham gia viết rồi mang đi scan. Sau đó xử lý qua một đoạn mã Matlab[1]<sup>36</sup> để từ một ảnh gồm nhiều biểu thức thu được ảnh chỉ chứa đúng một biểu thức như hình bên dưới. Ảnh này là ảnh xám, với kích thước  $300 \times 300$ .

<sup>36</sup>Đoạn mã này đã được chỉnh sửa để phù hợp với hình dạng mẫu lấy biểu thức của nhóm.



Hình 14: Ví dụ một ảnh thu được sau khi xử lý biểu mẫu đã qua scan bằng đoạn mã Matlab.

- Thực hiện gán nhãn và xác định bounding box cho từng ký hiệu trong ảnh thu được từ bước trên bằng công cụ do nhóm tự thực hiện với giao diện như hình bên dưới. Công cụ này cho phép tải ảnh, xác định bounding boxes cho các đối tượng trong ảnh bằng cách kéo thả chuột từ góc trên bên trái đến góc dưới bên phải của từng đối tượng và nhập nhãn cho mỗi đối tượng đó. Quá trình này kết thúc sẽ thu được file với định dạng .txt lưu thông tin cần thiết về dữ liệu cho quá trình huấn luyện và kiểm tra.



Hình 15: Giao diện công cụ hỗ trợ quá trình gán nhãn và xác định bounding boxes.

### 5.1.2.b Thông tin mô tả tập dữ liệu biểu thức

- Số người tham gia: khoảng 88
- Số lượng ảnh<sup>37</sup>: 2256 với 1746 ảnh dùng cho training, 138 ảnh dùng cho quá trình validation và 372 ảnh để test. Trong đó, 372 ảnh test được thu thập từ 2 nguồn:

<sup>37</sup>Mỗi ảnh chỉ chứa một biểu thức.

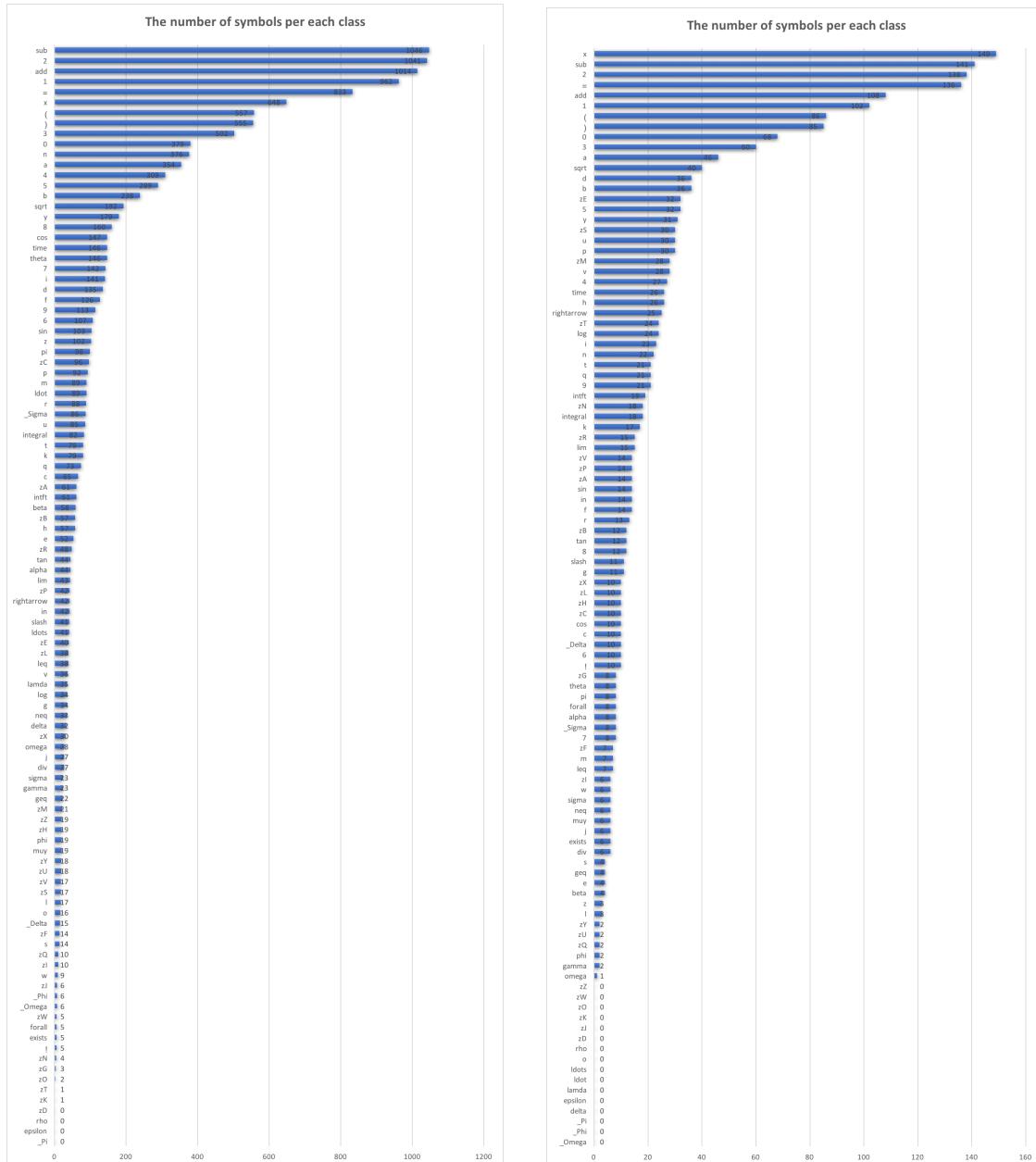


- 192 ảnh được viết bởi người chưa từng tham gia viết cho quá trình thu thập ảnh huấn luyện.
- 102 ảnh còn lại được viết bởi 15 người đã tham gia quá trình thu thập ảnh huấn luyện.
- Số loại biểu thức: khoảng 552
- Số người gán nhãn: 3

| Thông tin thêm                        | Tập huấn luyện | Tập kiểm tra   |
|---------------------------------------|----------------|----------------|
| Tổng số ký tự                         | 13697          | 2237           |
| Chiều dài trung bình<br>của biểu thức | $\approx 8$    | $\approx 6$    |
| Kích thước trung bình<br>của ký tự    | $24 \times 29$ | $31 \times 41$ |

**Bảng 5:** Thông tin mô tả hai tập dữ liệu dùng cho huấn luyện và kiểm tra.

Hình minh họa cho sự phân phối số lượng ký tự trong từng nhãn đối với tập huấn luyện và tập kiểm tra.



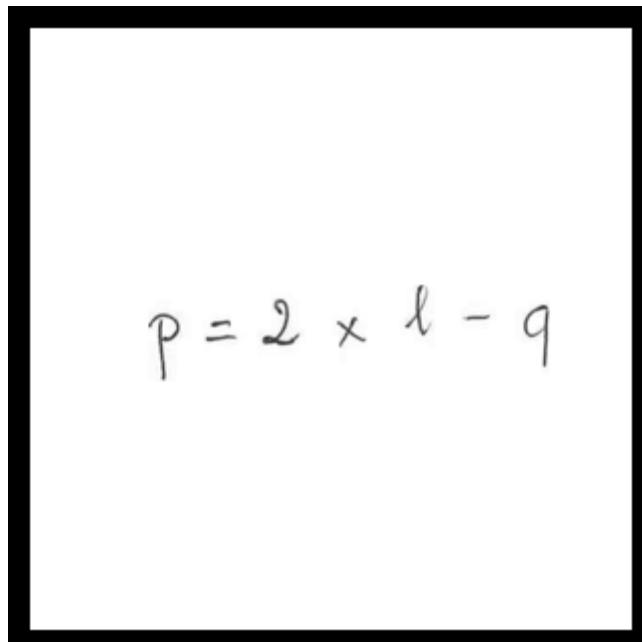
Hình 16: Sự phân phối ký tự theo từng nhãn đối với tập huấn luyện và tập kiểm tra.



## 5.2 Xây dựng mô hình

Do bản chất mạng SSD300[3] mô tả trong phần mô hình đề xuất gặp rất nhiều khó khăn trong việc nhận dạng ký tự nhỏ với một số lý do:

- Trong quá trình matching, các bounding box ground truth khi match với các default box phải có chỉ số Jaccard[7] lớn hơn 0.5, nhưng kích thước tối thiểu của default box lại là  $30 \times 30$  pixel, vì vậy nhiều ký tự nhỏ không thể match được (Trường hợp này xảy ra với hầu hết những ảnh có chữ nhỏ, ví dụ như ảnh dưới)



Hình 17: Ảnh ví dụ chữ nhỏ (với kích thước thật)

Khi không thể match được thì vùng ký tự đó được hiểu là phần nền, điều này gây bất lợi lớn cho quá trình huấn luyện do vùng ảnh không trống nhưng lúc là nền lúc lại được gán nhãn.

- Nhiều ký tự có aspect ratio đặc biệt (Ví dụ như ký tự dấu — trong phân số hoặc  $\frac{1}{2}$ ) cũng gặp nhiều khó khăn trong việc matching do không đạt được ngưỡng Jaccard[7].

Vì vậy, nhóm đã sửa một số thông số để tìm cách giải quyết vấn đề trên, trong nhiều bộ thông số thì có 3 phiên bản nổi bật sẽ được đặc tả bên dưới. Để thuận tiện, nhóm xin phép được đặt tên phiên bản SSD300 nguyên thủy[3] với sự thiết lập đã được giới thiệu trong 4 là phiên bản I. Phiên bản này được xây dựng dựa trên mã nguồn của tác giả tại [2]. Ba phiên bản được đề xuất sau được đặt tên lần lượt là phiên bản II, III, IV.

### 5.2.1 Giảm kích thước các default box - Phiên bản II

Phiên bản này dựa trên phiên bản SSD300 nguyên thủy[3], chỉ chỉnh sửa kích thước các bounding box thông qua chỉnh hai thông số  $s_{min} = 0.08$  và  $s_{max} = 0.5$ , từ đó danh sách kích thước các default box được thay đổi thành (9, 24, 54, 84, 114, 144, 174). So với phiên bản nguyên thủy, phiên bản này không thể match tốt các ký tự lớn (lớn hơn khoảng  $200 \times 200$  pixel). Tuy vậy, đối với bài toán về phát hiện các ký tự trong một ảnh, thì kích thước các ký tự không thể quá lớn, nên những ký tự đặc biệt lớn như vậy có thể được xem xét ở độ ưu tiên thấp hơn. Việc thay đổi tham số ở phiên bản này không làm ảnh hưởng lớn đến cấu trúc mạng cũng như quá trình mã hóa, giải mã. Số lượng dự đoán không thay đổi.

Mục đích chính của thay đổi này là để quá trình matching diễn ra tốt hơn khi kích thước của default box nhỏ nhất là  $9 \times 9$ , điều này giúp cho những ký tự rất nhỏ vẫn có thể được match, tránh hiện tượng bỏ sót ký tự như mạng SSD300 nguyên thủy[3].

### 5.2.2 Giảm kích thước các default box và tăng kích thước ảnh đầu vào - Phiên bản III

Để có được phiên bản này, nhóm cần phải có hai thay đổi so với mạng SSD300 nguyên thủy[3]:

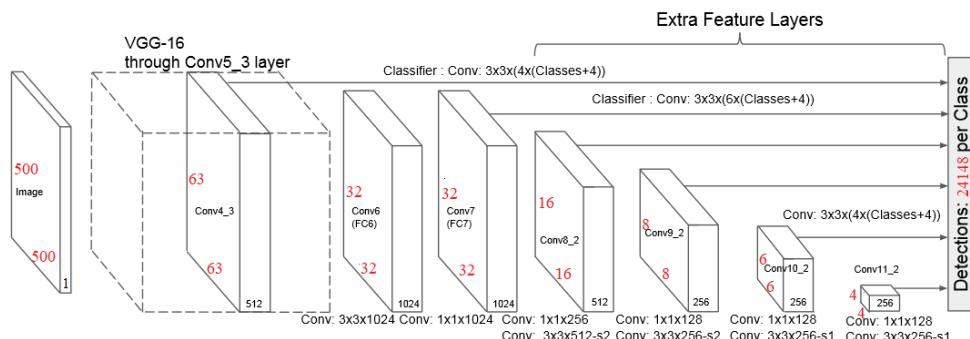
- Thay vì sử dụng ảnh  $300 \times 300$  như nguyên thủy, ảnh sẽ được phóng to lên kích thước  $500 \times 500$ .
- Như ở phiên bản đề xuất trước, nhóm thay đổi  $s_0 = 0.03$ ,  $s_{min} = 0.08$  và  $s_{max} = 0.5$  từ đó thu được danh sách kích thước  $(15.0, 40, 80, 120, 160., 200, 240, 280)$ .

Những thay đổi trên có làm thay đổi đến cấu trúc SSD[3], khi tăng kích thước ảnh đầu vào nghĩa là kích thước của các feature map thu thập được cũng tăng lên, từ đó kéo số lượng dự đoán trong một ảnh tăng lên như bảng dưới.

| Feature map<br>Sinh bởi | Kích thước                 | Aspect Ratio                            | Số dự đoán |
|-------------------------|----------------------------|---|------------|
| Conv_4                  | $512 \times 63 \times 63$  | $\{1, \frac{1}{2}, 2\}$                 | 15876      |
| Conv_7                  | $1024 \times 32 \times 32$ | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 6144       |
| Conv_8                  | $512 \times 16 \times 16$  | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 1536       |
| Conv_9                  | $256 \times 8 \times 8$    | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 384        |
| Conv_10                 | $256 \times 4 \times 4$    | $\{1, \frac{1}{2}, 2\}$                 | 144        |
| Conv_11                 | $256 \times 2 \times 2$    | $\{1, \frac{1}{2}, 2\}$                 | 64         |

Bảng 6: Thông tin của các feature map trích được từ mạng SSD đã giảm kích thước default box và tăng kích thước ảnh đầu vào

Như vậy với mỗi ảnh, hệ thống sinh ra 24148 dự đoán, gấp gần 3 lần so với ảnh SSD300 nguyên thủy[3]. Với những thay đổi này bounding box thậm chí còn có thể match với những default box nhỏ hơn, bên cạnh đó, mật độ các default box cũng dày đặc hơn nhiều so với mạng SSD nguyên thủy[3].



Hình 18: Cấu trúc mạng của phiên bản III

### 5.2.3 Giảm kích thước các default box, tăng kích thước ảnh đầu vào và thêm lớp tích chập - Phiên bản IV

Phiên bản này thừa kế các thay đổi ở hai phiên bản trước, tuy nhiên ở phần thân mạng SSD[3], nhóm đã gắn thêm 2 lớp tích chập ở cuối với các tham số:

| Khối      | Số kênh đầu vào | Số kênh đầu ra | Kích thước nhân | Chèn thêm (Padding) | Bước (Stride) |
|-----------|-----------------|----------------|-----------------|---------------------|---------------|
| Conv_12_1 | 256             | 128            | 1               | 0                   | 1             |
| Conv_12_2 | 128             | 256            | 3               | 0                   | 1             |

**Bảng 7:** Cấu hình các lớp tích chập thêm vào mạng SSD300[3] để được mạng SSD chỉnh sửa

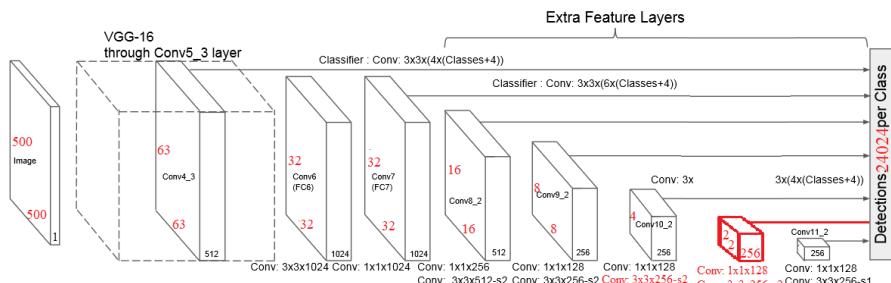
Hai lớp tích chập này cũng có hàm truyền là hàm relu và đi chung với lớp dropout[18] với hệ số dropout là 0.2.

Thay đổi này cũng làm thay đổi cấu trúc mạng:

| Feature map Sinh bởi | Kích thước                 | Aspect Ratio                            | Số dự đoán |
|----------------------|----------------------------|---|------------|
| Conv_4               | $512 \times 63 \times 63$  | $\{1, \frac{1}{2}, 2\}$                 | 15876      |
| Conv_7               | $1024 \times 32 \times 32$ | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 6144       |
| Conv_8               | $512 \times 16 \times 16$  | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 1536       |
| Conv_9               | $256 \times 8 \times 8$    | $\{1, \frac{1}{2}, 2, \frac{1}{3}, 3\}$ | 384        |
| Conv_10              | $256 \times 4 \times 4$    | $\{1, \frac{1}{2}, 2\}$                 | 64         |
| Conv_11              | $256 \times 2 \times 2$    | $\{1, \frac{1}{2}, 2\}$                 | 16         |
| Conv_12              | $256 \times 2 \times 1$    | $\{1, \frac{1}{2}, 2\}$                 | 4          |

**Bảng 8:** Thông tin của các feature map trích được từ mạng SSD[3] đã giảm kích thước default box, tăng kích thước ảnh đầu vào và tăng số lớp tích chập

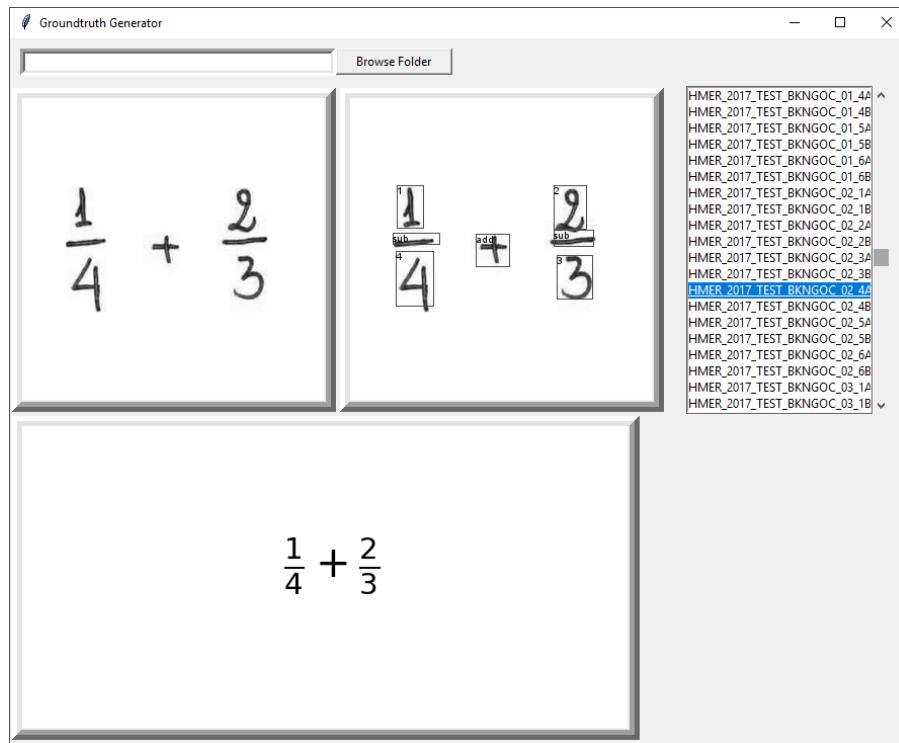
Lớp Conv\_12 được thêm vào cuối mạng với mục đích tăng độ dài danh sách feature map đưa vào lớp Multibox, điều này giúp trong quá trình matching, các ký tự lớn hơn có thể được match.



**Hình 19:** Cấu trúc mạng của phiên bản III

### 5.3 Xây dựng bản thử nghiệm

Khi đã thu được mô hình hoàn thiện của SSD, nhóm xây dựng một bản thử nghiệm bằng python. Giao diện khá đơn giản, người dùng chỉ việc chọn thư mục chứa ảnh, sau đó chọn ảnh cần nhận diện. Chương trình sẽ xử lý và đưa ra 3 ảnh gồm ảnh gốc ban đầu, ảnh thể hiện các bounding box mà mạng dự đoán và ảnh biểu thức toán học được tạo ra từ đoạn mã Latex.



Hình 20: Giao diện bản thử nghiệm

## 5.4 Kết quả

### 5.4.1 So sánh định tính giữa bốn phiên bản

#### 5.4.1.a Quá trình matching

Nhóm đã giảm kích thước các default box, đồng thời phóng to ảnh đầu vào, một phần mục đích là để quá trình matching có thể thực hiện tốt hơn đối với các ký tự nhỏ. Công việc này thật sự đã cải thiện được quá trình matching khi kích thước tối thiểu của bounding box ground truth được cải thiện một cách đáng kể:

| Mô hình                     | I              | II             | III           | IV            |
|-----------------------------|----------------|----------------|---------------|---------------|
| bounding box vuông          | $22 \times 22$ | $12 \times 12$ | $7 \times 7$  | $7 \times 7$  |
| bounding box có tỉ lệ 1 : 2 | $16 \times 32$ | $16 \times 32$ | $5 \times 10$ | $5 \times 10$ |

Bảng 9: Kích thước bounding box ground truth nhỏ nhất mà mạng có thể match với default box được của bốn mô hình đề xuất. (Đơn vị tính: Điểm ảnh - Pixel)

#### 5.4.1.b Quá trình nhận diện

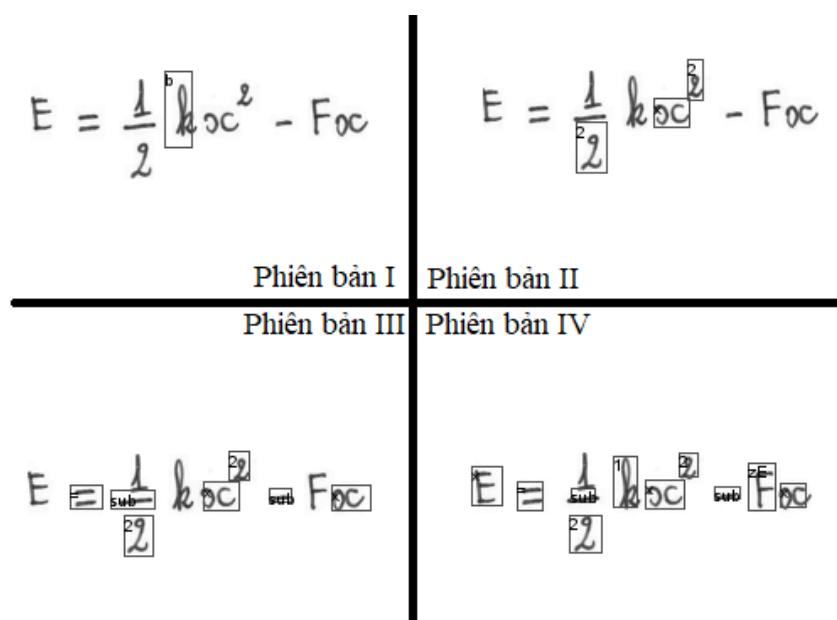
Kết quả của phiên bản II không tốt, về mặt lý thuyết thì khi giảm kích thước các bounding box, khả năng nhận diện các ký tự nhỏ sẽ tốt hơn, tuy nhiên khi áp dụng cho phiên bản II, mạng đã gặp một số vấn đề:

- Giảm kích thước bounding box sẽ khiến cho các ký tự lớn "bị hỏng", do các bounding box quá lớn khi tính chỉ số Jaccard[7] với các default box quá nhỏ sẽ không đủ ngưỡng, vì vậy mạng không thể match được các ký tự lớn, từ đó mạng cũng không thể nhận diện các ký tự lớn.
- Mạng có thể nhận diện được các ký tự nhỏ hơn, tuy nhiên các ký tự này cũng không đủ nhỏ để bao phủ phần lớn các ký tự nhỏ có trong tập dữ liệu.

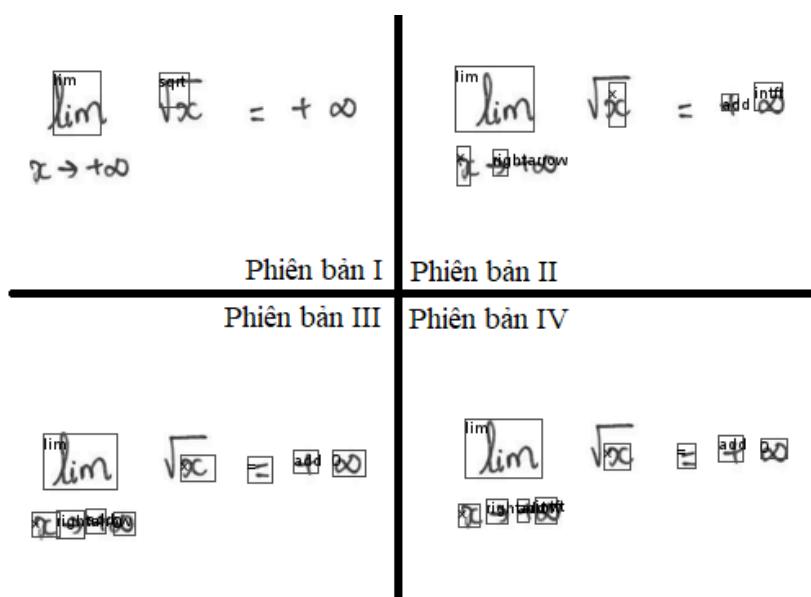


Vì vậy, qua thực nghiệm, phiên bản II đưa ra dự đoán về ký tự nhỏ tốt hơn phiên bản I một ít, nhưng kết quả vẫn còn rất tệ. Tuy vậy đối với những ký tự lớn, phiên bản II tỏ ra yếu hơn so với phiên bản I.

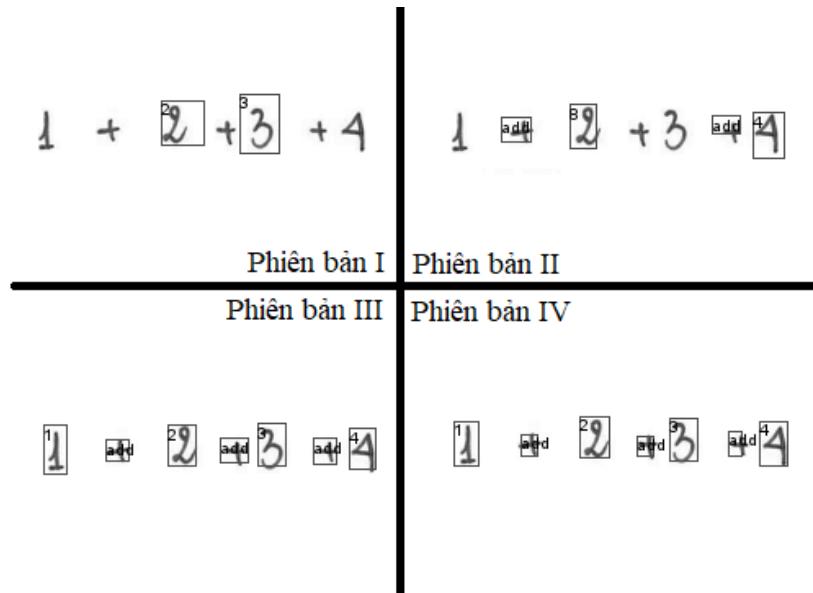
Ngược lại với phiên bản II, kết quả của phiên bản III và IV khá tốt, hệ thống đã có thể nhận diện nhiều ký tự rất nhỏ và tương đối chuẩn xác, phiên bản IV hầu như không thua kém phiên bản I trong việc nhận diện ký tự lớn, một số ký tự phiên bản I nhận diện được nhưng phiên bản IV không thể và ngược lại. Để hoàn thiện phiên bản III, IV trong việc nhận diện ký tự lớn, nhóm có đề xuất một hướng đi trong tương lai để cải thiện khả năng nhận diện ký tự lớn này.



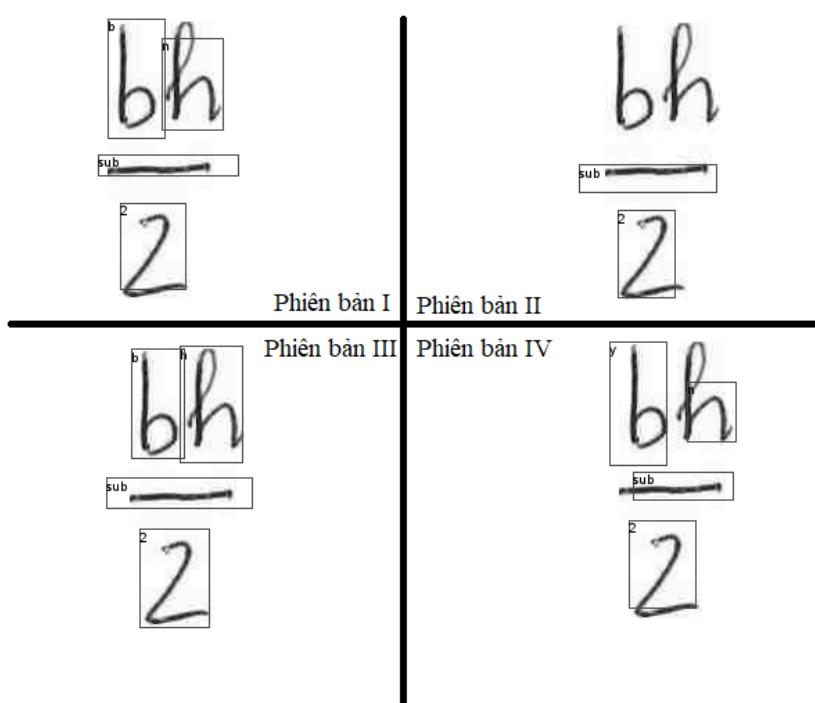
Hình 21: Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự nhỏ tăng dần qua 4 phiên bản. (1)



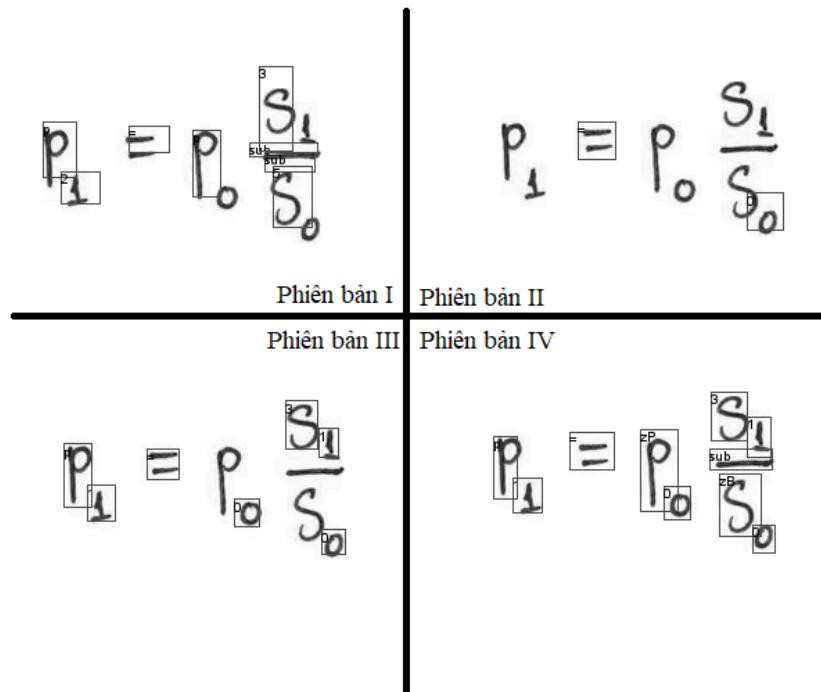
Hình 22: Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự nhỏ tăng dần qua 4 phiên bản. (2)



Hình 23: Kết quả dự đoán của 4 phiên bản. Phiên bản III và IV tỏ ra vượt trội hai phiên bản trước.

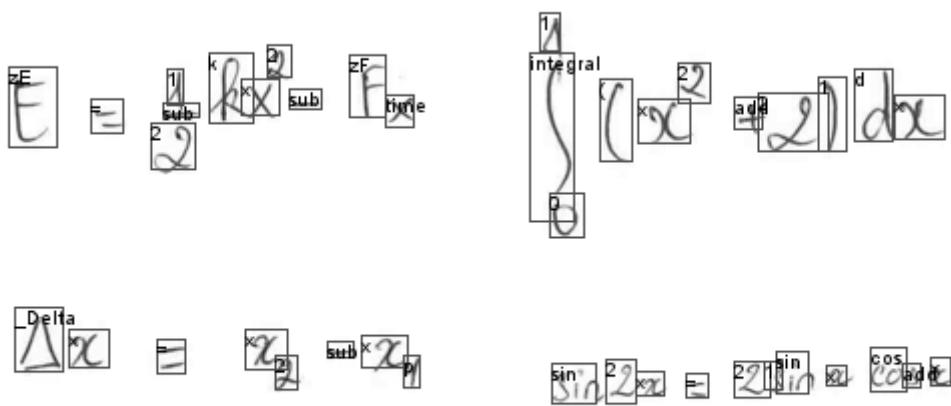


Hình 24: Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự lớn giảm mạnh ở phiên bản II.



**Hình 25:** Kết quả dự đoán của 4 phiên bản. Khả năng nhận diện ký tự lớn ở phiên bản IV không thua kém phiên bản I.

Đặc biệt, có nhiều biểu thức có ký tự rất nhỏ nhưng phiên bản IV đã làm việc rất tốt:



**Hình 26:** Ảnh kết quả của phiên bản IV(với kích thước thật)

#### 5.4.2 So sánh định lượng

Trong quá trình thực hiện, nhóm đã xây dựng 4 phiên bản huấn luyện dựa trên những thay đổi của 3 yếu tố sau:

- Kích thước ảnh đầu vào của mạng SSD.



- Kích thước default boxes.
- Số lượng default boxes.

Thông tin cụ thể đã được trình bày ở phần 5.2 của chương này, bên dưới là bảng tóm tắt.

| Phiên bản                | I  | II   | III  | IV   |
|--------------------------|--|--|--|--|
| Kích thước ảnh đầu vào   | 300x300  | 300x300  | 500x500  | 500x500  |
| Kích thước default boxes | min_ratio: 20<br>max_ratio: 90<br>min_scale: 0.1 | min_ratio: 8<br>max_ratio: 50<br>min_scale: 0.03 | min_ratio: 8<br>max_ratio: 50<br>min_scale: 0.03 | min_ratio: 8<br>max_ratio: 50<br>min_scale: 0.03 |
| Số lượng default boxes   | 8732   | 8732   | 24148  | 24024  |

Bảng 10: Sự khác nhau giữa 4 phiên bản thử nghiệm.

Điều kiện đánh giá:

- Cùng một tập ảnh kiểm tra với 372 ảnh. Phân phối số lượng ký tự trên từng nhãn của tập ảnh kiểm tra được thể hiện trong Hình 16.
- Độ đo sử dụng: mAP

Cách tính:

Tính precision cho từng nhãn theo công thức[20]:

$$precision = \frac{TP}{TP + FP}, \quad (16)$$

với  $TP$  được viết tắt từ *TruePositive*[20] và  $FP$  từ *FalsePositive*[20].

Tính trọng số của từng nhãn trong tập dữ liệu kiểm tra theo công thức:

$$class\_weight_i = \frac{N_i}{\sum_{u=0}^{106} N_u}, \quad (17)$$

với  $N_i$  là số lượng ký tự thuộc nhãn  $i$ .

mAP là trung bình có trọng số giữa *precision* và *class\_weight*.

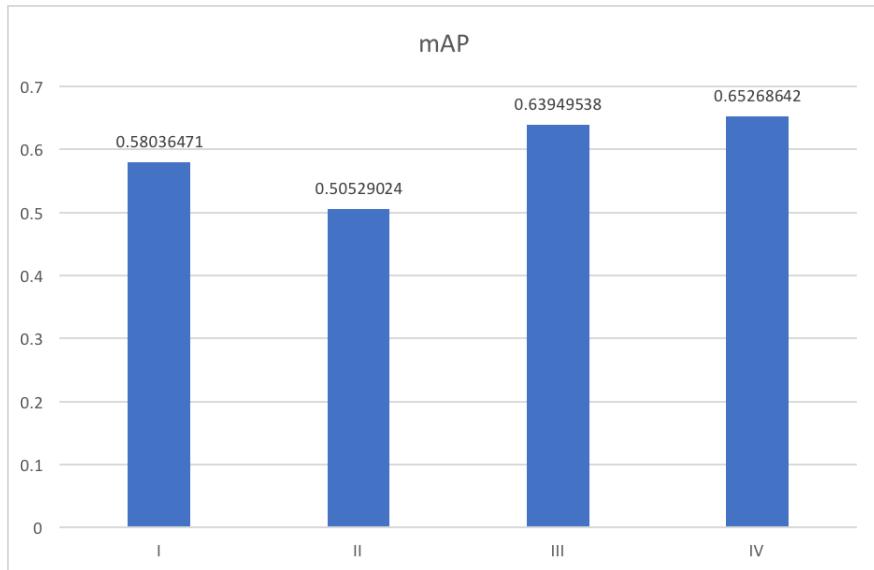
- Điều kiện máy chạy đánh giá:

- Hệ điều hành: Ubuntu 14.04 LTS
- CPU: Intel Core i5- 2500M 3.30GHz
- RAM: 8GB

Kết quả đánh giá của 4 phiên bản:

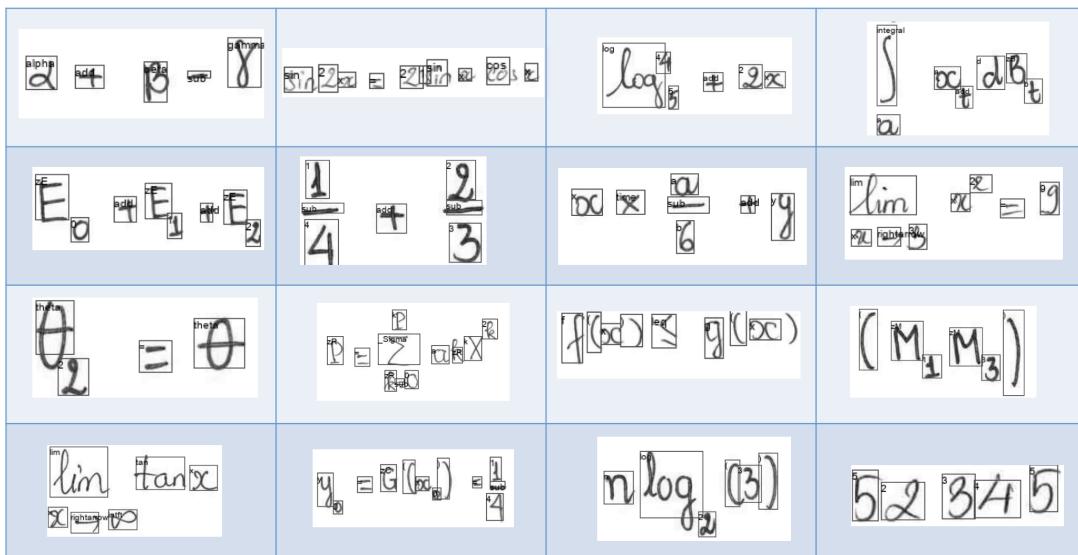
| Phiên bản                     | I          | II          | III        | IV         |
|-------------------------------|------------|-------------|------------|------------|
| mAP                           | 0.58036471 | 0.505290245 | 0.63949538 | 0.65268642 |
| Số lượng ký tự nhận dạng được | 1581       | 1054        | 1837       | 1920       |
| Số file ảnh không xử lý được  | 4/372      | 8/372       | 0/372      | 0/372      |

Bảng 11: Kết quả mAP của 4 phiên bản thử nghiệm.



Hình 27: Trực quan kết quả mAP cho 4 phiên bản thử nghiệm.

Như vậy, từ mô hình ban đầu- mô hình I- với những thiết lập được giữ y nguyên như [3], nhóm đã thay đổi để tạo ra được mô hình IV- mô hình tốt nhất của nhóm. Từ đó nâng chỉ số mAP từ khoảng 58% lên khoảng 65%. Nhìn vào đánh giá giữa 4 mô hình, rõ ràng thấy được việc thay đổi kích thước ảnh đầu vào và kích thước default box có tác động mạnh đến việc cải thiện độ chính xác của phương pháp.



Hình 28: Một số ảnh kết quả sau quá trình nhận dạng sử dụng phiên bản IV.



## 6 Tổng kết

### 6.1 Kết luận

Hệ thống nhận dạng biểu thức toán học viết tay mà nhóm đang xây dựng là bước đầu cho một ứng dụng tiện ích sau này đối với các bạn học sinh, sinh viên cũng như với những giáo viên- những người thường xuyên tiếp xúc với các con số, công thức và mong muốn rằng có một sản phẩm nào đó hỗ trợ giải các bài toán không chỉ bằng cách đưa ra đáp số mà còn là hướng dẫn giải từng bước, vẽ đồ thị hàm số hay thậm chí cho phép tương tác lên hình ảnh đồ thị đó, gợi ý họ những đoạn lệnh Latex hay có thể là một ngôn ngữ tương tự từ ảnh của những biểu thức để tiết kiệm thời gian soạn thảo những công thức dài và phức tạp. Ngoài ra, với việc có thể trực quan những đồ thị hàm số hay hướng dẫn giải một số loại bài toán sẽ giúp cho việc dạy và học trở nên sinh động và thuận lợi hơn. Như vậy, rất nhiều những ứng dụng có thể phát triển từ hệ thống nhận dạng mà nhóm đang thực hiện.

Với hệ thống này, nhóm đã sử dụng phương pháp phát hiện và nhận dạng của mạng SSD thay vì những phương pháp phân tách phổ biến như phân tích hình chiếu, phân tích thành phần liên thông để khắc phục những lỗi sai có thể xảy ra khi gặp những ký tự dính nhau, bọc lấy nhau kết hợp với bộ phân tích cấu trúc DRACULAE, hệ thống sẽ giải quyết được nhiều loại công thức phức tạp hơn như những công thức có chứa chỉ số trên, chỉ số dưới,... Ngoài ra, trong quá trình thực hiện đề tài, nhóm đã phát triển 2 tập dữ liệu viết tay bao gồm 1 tập các ký tự và 1 tập những biểu thức từ đơn giản đến phức tạp. Với tập ký tự, dữ liệu đã lên tới con số 52353 file ảnh còn đối với tập biểu thức, hiện tại nhóm đã có khoảng 2400 ảnh. Đây là điều kiện quan trọng để nhóm có thể thực hiện những bước huấn luyện, đánh giá mô hình và cũng là cơ sở cho những đề tài liên quan đến chữ viết tay.

### 6.2 Đánh giá ưu, nhược điểm

#### 6.2.1 Ưu điểm

- Hệ thống có thể nhận diện được ký tự ở nhiều kích thước khác nhau.
- Nhận diện được nhiều biểu thức mà nếu chỉ dùng kỹ thuật phân vùng thì khó có thể làm được (ví dụ như biểu thức có ký tự dính nhau).
- Nhận diện được nhiều loại ký tự.

#### 6.2.2 Nhược điểm

- Khi có nhiều ký tự trong ảnh thì hệ thống nhận diện gặp nhiều khó khăn.
- Tốc độ xử lý một ảnh còn chậm.
- Bộ phân tích cú pháp còn đơn giản.
- Các ký tự hoàn toàn trùng lặp nhau dễ bị bỏ sót (dấu cản).

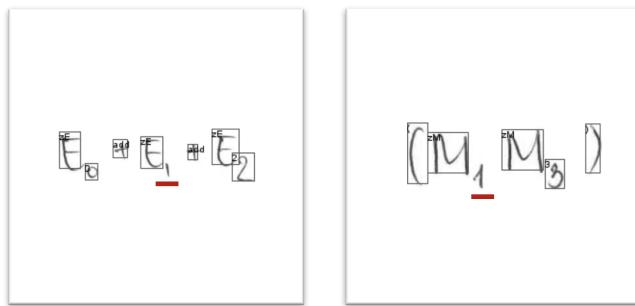
### 6.3 Hướng phát triển trong tương lai

Đến thời điểm kết thúc luận văn, hệ thống nhận diện vẫn còn tồn tại những khuyết điểm và cả những trường hợp mà nhóm chưa có thời gian để kiểm tra. Trong đó nổi lên 2 vấn đề mà nhóm đã tiến hành giải quyết thử và thu được những kết quả khả quan ở bước đầu là việc thu nhỏ nội dung ảnh để có thể khắc phục lỗi sai mà hệ thống mắc phải khi gặp những ký tự quá lớn và tăng cường thêm dữ liệu để giải quyết trực tiếp vô trường hợp chỉ số dưới của 2 ảnh cụ thể trong tập test không nhận diện được. Ngoài ra vẫn còn nhiều trường hợp khác nữa mà nhóm đó là hướng phát triển trong tương lai của đề tài.



### 6.3.1 Mở rộng tập dữ liệu biểu thức

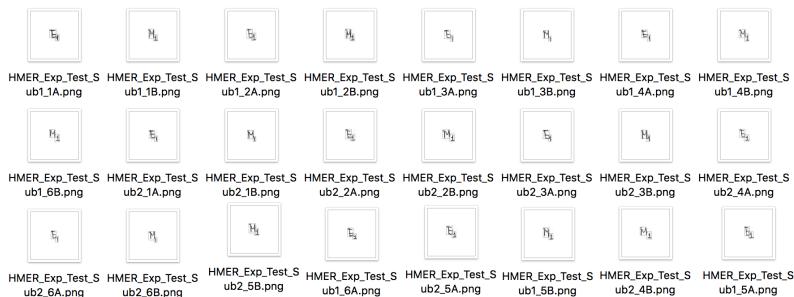
Trong quá trình đánh giá hệ thống trên tập kiểm tra với mô hình IV, nhóm nhận thấy đối với 2 ảnh bên dưới hệ thống không nhận dạng được chỉ số dưới- cụ thể là số 1 ở vị trí dấu gạch đỏ.



Hình 29: Trường hợp không nhận được chỉ số dưới.

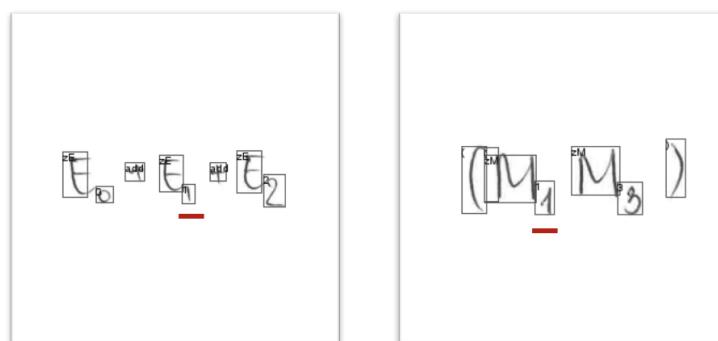
Trên nhận định lý do của vấn đề này là do trong tập huấn luyện số lượng mẫu với chỉ số dưới còn ít và thiếu những mẫu mà chỉ số dưới dính vào hệ số của nó, nhóm đã bổ sung thêm 96 ảnh với nội dung cố định là  $E_1$  và  $M_1$  vào tập huấn luyện và 24 ảnh để kiểm tra với hy vọng sau khi huấn luyện lại với lượng ảnh bổ sung thì tình trạng trong Hình 29 sẽ được giải quyết.

Sử dụng kiến trúc mạng theo mô hình IV, sau quá trình huấn luyện với tập dữ liệu đã được thêm mới, nhóm thu được kết quả là cả 24 ảnh kiểm tra đều được nhận dạng đúng.



Hình 30: Kết quả nhận dạng chỉ số dưới trên bộ ảnh test mới.

Và kết quả trên 2 ảnh gấp lõi ở Hình 29:



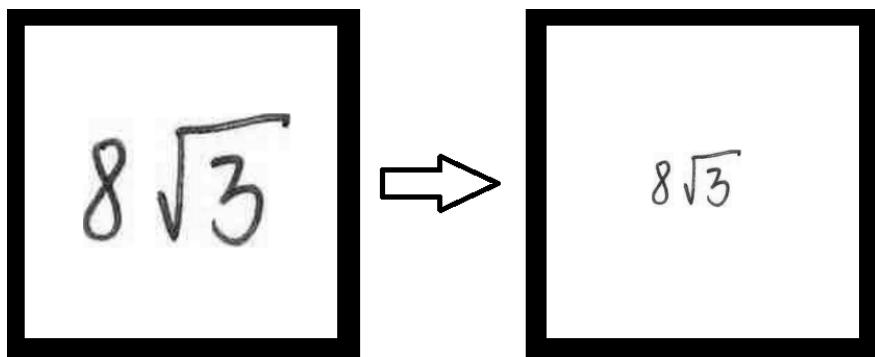
Hình 31: Kết quả nhận dạng sau khi được huấn luyện với dữ liệu bổ sung.



Có thể nhận thấy được rằng, ký tự tại vị trí dấu gạch đỏ ở cả 2 hình đều được phát hiện và nhận đúng. Do đó hướng giải quyết phát triển tập dữ liệu là điều cần thiết và khả năng sẽ khắc phục những một số lỗi sai khác.

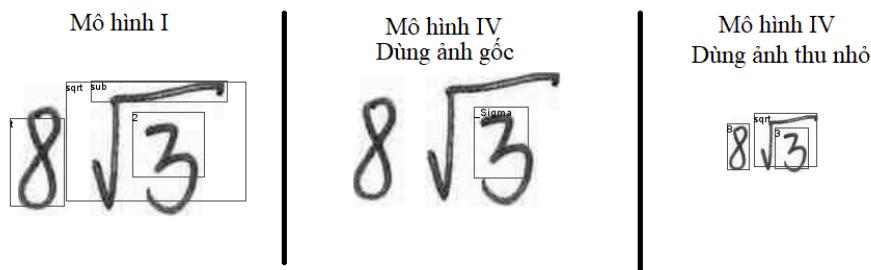
### 6.3.2 Khắc phục lỗi nhận dạng sai ở ký tự lớn

Như nhóm đã đề cập ở phần trước, phiên bản IV hoạt động chưa tốt với những ký tự quá lớn, nắm được điều này, nhóm đã thử thu nhỏ ảnh và tiến hành chèn thêm (padding) vào ảnh đã thu nhỏ đó để nội dung ảnh thì nhỏ lại nhưng kích thước ảnh không đổi, vẫn  $500 \times 500$ .



Hình 32: Ví dụ thu nhỏ ảnh.

Sau khi thí nghiệm với một số ảnh, nhóm đã nhận thấy kết quả nhận được là khá tốt.



Hình 33: Kết quả thu được sau khi thu nhỏ nội dung ảnh.

Vì vậy, một trong những điều cần được thực hiện ngay trong tương lai là tối ưu hoá bộ tiền xử lý để xử lý linh hoạt cho những ảnh có nội dung lớn và nhỏ khác nhau. Từ đó tạo ra bộ ảnh đầu vào phù hợp cho hệ thống.

### 6.3.3 Các hướng phát triển khác

Như đã đề cập ở đầu mục này, vẫn còn nhiều trường hợp mà nhóm không có đủ thời gian để kiểm tra trong giai đoạn luận văn. Đây cũng là những hướng đi mà trong tương lai nhóm cần phải thực hiện. Cụ thể, nhóm chưa kiểm tra được liệu hệ thống của mình có thể hoạt động tốt trong các tình huống sau:

- Ảnh chứa đựng các ký tự nghiêng.
- Ảnh có nhiễu.
- Ảnh với nền khác màu trắng.
- Ảnh chụp thay vì scan.



Ngoài ra, cũng còn những điều cần hoàn thiện:

- Thêm phần hậu xử lý ở các giai đoạn nhận diện ký tự và sinh cây Lexed - BST.
- Cải thiện khả năng nhận diện các ký tự có aspect ratio đặc biệt.
- Tối ưu hoá bộ phân tích cú pháp.



## Lời kết

Trải qua giai đoạn luận văn và cả thực tập tốt nghiệp, những thành viên nhóm đã học được nhiều điều. Về kiến thức, luận văn này là cơ hội cho các thành viên nhóm tìm hiểu và hiện thực những mô hình mạng Deep learning. Bước đầu biết được những vấn đề khó khăn khi huấn luyện một mạng và cách khắc phục chúng. Hiểu và xây dựng được một ứng dụng trên lĩnh vực nhận dạng đó chính là đề tài của nhóm, mặc dù về mặt ứng dụng nó còn ở mức đơn sơ. Về cách làm việc nhóm, trên thực tế các thành viên nhóm liên tục gặp phải những bất đồng, tranh chấp trên quan điểm làm việc cũng như xử lý vấn đề. Có những lúc gay gắt mà các thành viên cảm thấy không thể tiếp tục được với nhau. Tuy nhiên tất cả những điều khó khăn này xảy ra đều dựa trên nền tảng vì một luận văn thành công, vì một đề tài mà cả hai thành viên nhóm đều tự hào nên đã vượt qua được. Tất cả những điều đó là trải nghiệm, là kinh nghiệm để các thành viên nhóm hoàn thiện mình hơn cho giai đoạn công việc sắp tới. Cho đến thời điểm này, mặc dù vẫn còn đó những thiếu sót nhưng cả 2 thành viên nhóm tự hào vì những điều mình đã làm được.

Chân thành cảm ơn.



## Phụ lục

### Những lỗi sai mà hệ thống gấp phải

| Ảnh đầu vào          | Ảnh sau bước nhận dạng    | Ảnh sau bước phân tích cấu trúc |
|----------------------|---------------------------|---------------------------------|
| $p = 2 \times l - q$ | $p \equiv 2 \times l - q$ | $p = 2 \times k - q$            |
| $p = 2 \times l - q$ | $p \equiv 2 \times l - q$ | $p = 2 \times 1 - q$            |
| $y = 2x$             | $y \equiv 2x$             | $y = 2x$                        |

Hình 34: Trường hợp sinh chuỗi Latex sai khi biểu thức chứa các ký tự có chân.

| Ảnh đầu vào            | Ảnh sau bước nhận dạng | Ảnh sau bước phân tích cấu trúc |
|------------------------|------------------------|---------------------------------|
| $4\pi^2 a R$           | $4\pi^2 a R$           | $4\pi^2 a k$                    |
| $S^{4m+1}$             | $S^{4m+1}$             | $3^{4m+1}$                      |
| $h \rightarrow \infty$ | $h \Rightarrow \infty$ | $h - \infty$                    |

Hình 35: Trường hợp hệ thống gán nhãn cho những ký tự có nét tương tự nhau.



## Tài liệu

- [1] A. N. Quoc and K. N. Anh, *Nhận dạng biểu thức toán học*. 2016.
- [2] Kuangliu, “pytorch-ssd.” <https://github.com/kuangliu/pytorch-ssd/>, May. 2017.
- [3] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, “Ssd: Single shot multibox detector,” in *European conference on computer vision*, pp. 21–37, Springer, 2016.
- [4] J. Zhang, J. Du, S. Zhang, D. Liu, Y. Hu, J. Hu, S. Wei, and L. Dai, “Watch, attend and parse: An end-to-end neural network based approach to handwritten mathematical expression recognition,” *Pattern Recognition*, 2017.
- [5] W. He, Y. Luo, F. Yin, H. Hu, J. Han, E. Ding, and C.-L. Liu, “Context-aware mathematical expression recognition: An end-to-end framework and a benchmark,” in *Pattern Recognition (ICPR), 2016 23rd International Conference on*, pp. 3246–3251, IEEE, 2016.
- [6] R. Zanibbi, D. Blostein, and J. Cordy, “Recognizing mathematical expressions using tree transformation,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 11, pp. 1455 – 1467, Nov. 2002.
- [7] P. Jaccard, “Distribution de la flore alpine dans le bassin des dranses et dans quelques régions voisines.,” vol. 37, pp. 241–72, 01 1901.
- [8] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *arXiv preprint arXiv:1409.1556*, 2014.
- [9] K. Simonyan and A. Zisserman, “Very deep convolutional networks for large-scale image recognition,” *CoRR*, vol. abs/1409.1556, 2014.
- [10] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [11] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 2147–2154, 2014.
- [12] R. Girshick, “Fast r-cnn,” in *Proceedings of the IEEE international conference on computer vision*, pp. 1440–1448, 2015.
- [13] K.-F. Chan and D.-Y. Yeung, “Mathematical expression recognition: a survey,” *International Journal on Document Analysis and Recognition*, vol. 3, no. 1, pp. 3–15, 2000.
- [14] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, “On the properties of neural machine translation: Encoder-decoder approaches,” *arXiv preprint arXiv:1409.1259*, 2014.
- [15] K.-F. Chan and D.-Y. Yeung, “Mathematical expression recognition: a survey,” *International Journal on Document Analysis and Recognition*, vol. 3, no. 1, pp. 3–15, Aug. 2000.
- [16] R. H. Anderson, “Syntax-directed recognition of hand-printed two-dimensional mathematics,” *Symposium on Interactive Systems for Experimental Applied Mathematics: Proceedings of the Association for Computing Machinery Inc. Symposium*, pp. 436–459, Aug. 1967.
- [17] A. Karpathy, “Cs231n convolutional neural networks for visual recognition.” <http://cs231n.github.io/neural-networks-1/>.
- [18] N. Srivastava, G. E. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, “Dropout: a simple way to prevent neural networks from overfitting.,” *Journal of machine learning research*, vol. 15, no. 1, pp. 1929–1958, 2014.



- [19] C. C. on Recognition of Online Handwritten Mathematical Expressions. [http://www.isical.ac.in/~scrohme/CROHME\\_data.html](http://www.isical.ac.in/~scrohme/CROHME_data.html).
- [20] WIKIPEDIA. [https://en.wikipedia.org/wiki/Precision\\_and\\_recall](https://en.wikipedia.org/wiki/Precision_and_recall).