Hình ảnh sang Latex

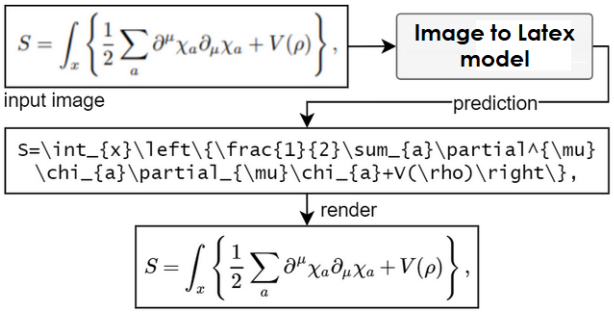
1st Nguyễn Văn Anh Tuấn
IUH - Trường Đại Học Công Nghiệp TP.HCM
Việt Nam
email: nvatuan3@gmail.com

***Tóm tắt nội dung*—Ta chứng kiến được càng ngày càng có nhiều người có nhu cầu trong việc viết và sử dụng LaTeX, nhưng việc đọc một tài liệu hay hình ảnh có công thức latex mà chưa biết ký hiệu LaTeX của nó khiến cho người đọc và người sử dụng LaTeX gặp khó khăn. Trong bài nghiên cứu này, nhóm kết hợp các kiến trúc Encoder khác nhau để cấu tạo lên mô hình Encoder-Decoder trong việc giải quyết bài toán Image to Latex. Trong số các kết quả, sự kết hợp giữa Convolution Feature Encoder và BiLSTM Row Encoder đạt được kết quả tốt nhất trên BLEU4 là 77% trong bộ dữ liệu IMG2LATEX-100k.**

I. Giới thiệu

Bài toán hình ảnh sang LaTeX (Image to Latex) nhằm mục đích giải quyết vấn đề chuyển đổi hình ảnh chứa các công thức toán học LaTeX trong sách, hình ảnh sang chuỗi LaTeX thực tế để tiện cho việc biên soạn, chỉnh sửa hay chỉ đơn giản là tò mò xem công thức ở trong ảnh kia nó có tên LaTeX là gì để có thể tìm kiếm các tài liệu nghiên cứu liên quan đến ký tự đó.

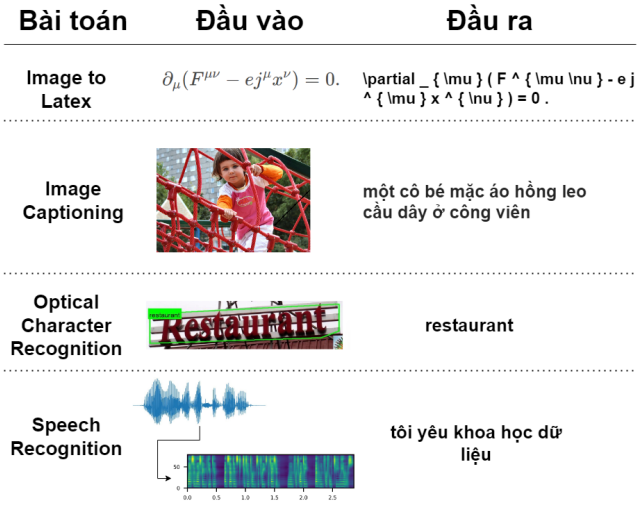
Đầu vào của bài toán này là một hình ảnh chứa chuỗi LaTeX, đầu ra sẽ là một chuỗi LaTeX, Hình 1 mô tả sơ đồ bài toán.



Hình 1: Mô hình của bài toán Image to LaTeX, đầu vào là ảnh và đầu ra là chuỗi các ký tự LaTeX.

Bài toán này thoạt nhìn thì khá tương tự với bài toán Chú thích hình ảnh (Image Captioning) [3], Nhận dạng giọng nói (Automatic Speech Recognition - ASR) [4] hay Nhận dạng chữ quang học (Optical Character Recognition - OCR) [5]. Tuy nhiên, bài toán Image to Latex có những sự khác biệt nhất định với ba bài toán trên. Hình ảnh đầu vào của bài toán chỉ đơn giản là ảnh trắng đen, không nhiều màu sắc và thông tin như bài toán Image Captioning, nên mô hình sẽ học khó hơn về độ phức tạp. Các công thức LaTeX có cách hiển thị và kết xuất khác nhau (ví dụ công thức $\sum_{i=1}^{N} i^2$ trong Latex sẽ được hiển thị là `\sum_{i=1}^N i^2`), nên sẽ dẫn đến chồng chéo các công thức, ký hiệu Latex lên nhau, trong khi đó bài toán OCR lại chỉ nhận dạng các ký tự theo nhau liên tục, đơn điệu, không bị chồng chéo lên nhau. Thêm nữa, mặc dù bài toán ASR có dữ liệu gốc là dạng sóng của âm thanh, nhưng khi chuyển về dạng Mel Spectrogram thì vẫn sẽ được hiển thị như một ảnh, nhưng đầu ra cũng sẽ là các ký tự đơn điệu với nhau, không chồng chéo lên nhau, vẫn sẽ khác so với bài toán Image to Latex. Vì thế, các kiến trúc mô hình, các phương pháp của Image Captioning, OCR hay ASR không thể sử dụng ngay cho bài toán Image to Latex được. Hình 2 mô tả các nội dung trên.

Phần còn lại của bài nghiên cứu sẽ theo thứ tự: Phần II sẽ trình bày về các nghiên cứu nhóm đã tổng hợp được trong quá trình tìm hiểu về bài toán. Phần III trình bày về các kiến trúc Encoder, Decoder, Attention nhóm sử dụng trong bài nghiên cứu. Phần IV trình bày thực nghiệm và kết quả thực nghiệm còn phần V trình bày hướng phát triển tương lai của nhóm.



Hình 2: So sánh giữa bài toán Image to Latex với các bài toán: Image Captioning, Optical Character Recognition và Speech Recognition về đầu vào và đầu ra. Ta thấy điểm chung có được là mô hình đều nhận dạng hình ảnh và trả về văn bản.

II. Nghiên cứu liên quan

Bài toán này đã được nghiên cứu một thời gian. Nhóm tác giả [1] sử dụng kiến trúc Encoder-Decoder (Encoder là các lớp CNN kết hợp với Max Pooling và Decoder là LSTM 1 lớp) với Luong Attention đạt kết quả 75% trên BLEU và 35% trên Exact Match (EM).

Nhóm tác giả [2] trình bày mô hình *What You Get Is What You See* (WYGIWYS), là một trình dịch ngược từ hình ảnh, cũng sử dụng Encoder-Decoder, nhưng Encoder hiện tại là sự kết hợp của CNN, Max Pooling, Batch Normalization [6] và thêm một Bi-LSTM 1 lớp (gọi là Row Encoder) chạy trên từng hàng của dữ liệu ảnh được trích xuất từ Convolution Encoder, và Decoder là 1 lớp LSTM. Kết quả đạt được của kiến trúc này là 87.73% trên BLEU và 77.46% trên EM.

Cả hai bài nghiên cứu trên đều được thực nghiệm trên bộ dữ liệu IM2LATEX-100K [7], nhóm cũng tham khảo hai bài nghiên cứu trên cho việc thực nghiệm.

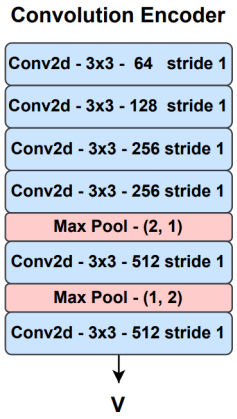
III. Phương pháp

Trong bài nghiên cứu này, nhóm chúng tôi sử dụng kiến trúc mô hình Encoder-Decoder với Attention, bao gồm Encoder là các kiến trúc được cấu thành từ CNN làm trung tâm, Decoder là LSTM 1 lớp và Attention sử dụng cơ chế chú ý để kết nối giữa Encoder và Decoder nhằm gia tăng khả năng học ngữ cảnh của mô hình.

A. Encoder

1) Convolution Encoder

Trong bài nghiên cứu [1], nhóm tác giả đã đề xuất một cấu trúc kết hợp các lớp CNN lại với nhau để tạo thành Encoder cho mô hình nhằm giải quyết bài toán Image to Latex. Hình dưới đây mô tả kiến trúc của nhóm tác giả. Tổng quan, kiến trúc này bao gồm 6 lớp CNN đều có kernel size là 3x3, stride 1 và 2 lớp Max Pooling. Hình 3 mô tả kiến trúc này.



Hình 3: Kiến trúc của Convolution Encoder

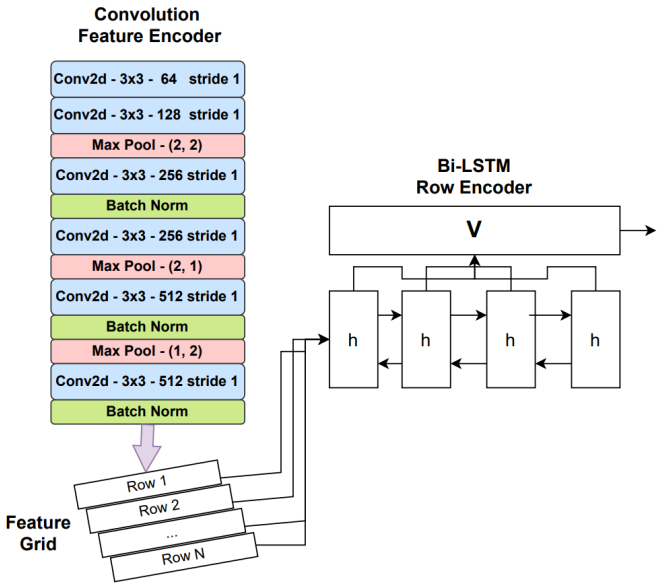
Kích cỡ đầu vào là $C \times W \times H$, sau khi đưa qua mô hình sẽ trở thành ma trận V có kích thước $512 \times W' \times H'$ (gọi là feature grid).

Ngoài ra, bên cạnh Convolution Encoder như trên, nhóm phát triển thêm một phiên bản kết hợp với Batch Normalization [6] vào sau các lớp CNN thứ 2, 3, 5 và 6. Batch Normalization sẽ giúp mô hình được huấn luyện nhanh và ổn định hơn bằng cách chuẩn hóa theo từng mini-batch dữ liệu đầu vào của mỗi lớp, sẽ giúp giảm được Covariance Shift, khiến phân phối dữ liệu đều nhau và thêm một ít chỉnh hóa cho mô hình tránh bị overfit.

2) Convolution Row Encoder

Theo tự nhiên, ta có thể hiểu rằng các ký hiệu LaTeX sẽ được viết theo thứ tự từ trái sang phải. Vì thế để có thể học được ngữ cảnh "*trái sang phải*" này, nhóm tác giả [2] kết hợp thêm kiến trúc của mạng neuron hồi tiếp (RNN) vào sau feature grid được trích xuất từ các lớp CNN (Convolution Encoder).

Cụ thể, kiến trúc Convolution Row Encoder được kết hợp từ Convolution Feature Encoder và Bi-LSTM Row Encoder. Mà Convolution Feature Encoder sẽ làm việc trên ảnh đầu vào, còn Row Encoder (LSTM 1 lớp 2 chiều có kích thước lớp ẩn là 256) sẽ nhận từng hàng của đầu ra của Convolution Feature Encoder để học và đưa ra một ma trận V có kích cỡ $512 \times W' \times H'$. Hình 4 mô tả kiến trúc của Convolution Row Encoder.



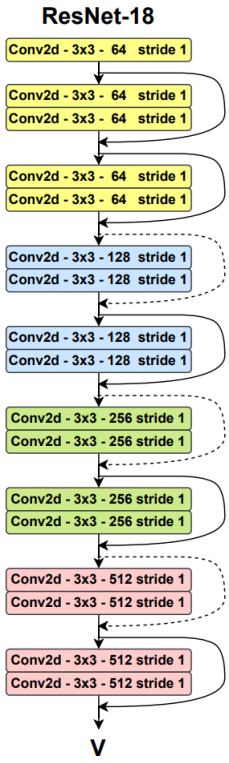
Hình 4: Kiến trúc của Convolution Row Encoder

3) ResNet-18

Sự phát triển của mô hình ResNet [8] giúp làm kiến trúc nền đằng sau các bài toán Computer Vision hiện nay như Image Recognition, Object Detection, GAN, ... [8]–[10]. Kiến trúc ResNet chủ yếu là sự kết hợp của các lớp CNN, Max Pooling, Batch Normalization và activation ReLU. Hình 5 mô tả kiến trúc tổng quát của ResNet-18, là kiến trúc nhóm sử dụng trong bài này.

Batch Normalization sẽ được gắn theo sau mỗi lớp Conv2d, bên cạnh đó, kẹp giữa các lớp Conv2d đôi sẽ là activation ReLU. Các skip connection dạng nối liên tục sẽ cộng trực tiếp giá trị gốc vào lớp đó, nghĩa là $\mathrm{g(x)} = \mathrm{x} + \mathrm{f(x)}$, còn dạng nét đứt sẽ cộng giá trị từ một lớp Conv2d thứ 3 (nhằm downsample [11]), nghĩa là $\mathrm{g(x)} = \mathrm{Conv2d(x)} + \mathrm{f(x)}$. Trong đó $\mathrm{f(x)}$ là giá trị của khối hiện tại sau khi đưa qua đầu vào, x là giá trị đầu vào, còn $\mathrm{g(x)}$ là kết quả đầu ra của lớp đó. Đầu ra của ResNet-18 sẽ là ma trận V có kích thước $512 \times W' \times H'$.

Ngoài ra, nhóm kết hợp thêm một phiên bản mô hình BiLSTM Row Encoder vào Feature Grid của ResNet-18.



Hình 5: Kiến trúc của ResNet-18

B. Decoder & Attention

1) Decoder

Trong bài này, nhóm chỉ sử dụng một kiểu decoder duy nhất là LSTM 1 lớp với kích thước lớp ẩn là 512. Decoder này sẽ nhận giá trị đầu vào là feature grid V có kích thước $C' \times W' \times H'$, kết hợp với Luong-Attention để lần lượt đưa ra các ký hiệu LaTeX.

Cụ thể hơn, tại mỗi bước thời gian t, đầu vào của LSTM decoder sẽ là sự kết hợp giữa vector word embedding của ký hiệu Latex y_t và vector ngữ cảnh attention c_t, và vector trạng thái ẩn h_{t-1}. Mà c_t sẽ được tính từ vector trạng thái ẩn trước đó của decoder h_{t-1} và feature grid của encoder V. Sau cùng LSTM sẽ xuất ra một phân phối xác suất của các ký hiệu Latex là đầu ra của bước thời gian t của decoder.

$$c_t = \text{Attention}(h_{t-1}, V) \tag{1}$$
$$e_t = \text{Embedding}(y_t) \tag{2}$$
$$o_t, h_t = \text{LSTM}(h_{t-1}, [c_t, e_t]) \tag{3}$$
$$p(y_{t+1}|y_1, \cdots, y_t) = \text{Softmax}(W_o \cdot o_t + b_o) \tag{4}$$

Khi bắt đầu, h_0 sẽ được khởi tạo bằng cách tổng theo H' và W' trên feature grid V của encoder.

$$h_0 = \tanh(W_h \cdot (\frac{1}{W' \times H'} \sum_{i=1}^{W' \times H'} v_i) + b_h), \quad v_i \in V \tag{5}$$

Trọng số ban đầu của lớp Embedding và LSTM được khởi tạo theo Orthogonal [13].

2) Attention

Cơ chế Attention nhóm sử dụng trong bài là Luong Attention [12].

$$\text{attn} = \tanh((W_h h_{t-1} + b_h) + (W_V V + b_V)) \tag{6}$$
$$\alpha^t = \text{Softmax}(W_a \text{attn} + b_a) \tag{7}$$
$$c_t = \sum_{i=1}^{W' \times H'} \alpha_i^t v_i, \quad v_i \in V \tag{8}$$

IV. Thực nghiệm & Kết quả

A. Thiết lập thực nghiệm

1) Dữ liệu

Nhóm sử dụng tập dữ liệu giống với hai tác giả trong phần II cho mục đích so sánh. Bộ dữ liệu tên là `im2latex-100k` [14] có tổng cộng 100000 cặp dữ liệu (ảnh, latex) được chia thành 3 tập: train (dùng cho việc huấn luyện mô hình), test (dùng cho việc đánh giá trong phần IV-B) và valid (dùng như tập test nhưng đánh giá trong quá trình huấn luyện) với tỉ lệ lần lượt là (80%, 9% và 11%).

2) Tiền xử lý dữ liệu

Đối với ảnh, nhóm chuyển dữ liệu của các điểm ảnh về khoảng số $[-1, 1]$ để dữ liệu có một khoảng giá trị đồng nhất, giúp mô hình có thể học tốt hơn.

Đối với Latex, nhóm thêm ký tự `<s>` và `<e>` vào đầu và cuối mỗi câu latex đã được tokenize để biểu thị cho mô hình có thể biết được vị trí đầu và cuối mỗi câu Latex. Khi đưa câu này vào mô hình, sẽ tạm thời bỏ ký hiệu `<e>` ra, và khi đưa giá trị đầu ra của mô hình cho việc tính giá trị mất mát, sẽ để lại `<e>` nhưng bỏ ký tự `<s>`.

Sau khi tokenize trên toàn bộ các câu Latex của tập dữ liệu huấn luyện, nhóm tổng hợp được 512 ký hiệu Latex khác nhau, đây cũng là số lớp đầu ra của mô hình.

3) Phương pháp đánh giá

Nhóm sử dụng 3 phương pháp đánh giá: BLEU, Exact Match (EM) và Edit distance.

- BLEU là một phương pháp thường được sử dụng trong các bài toán dịch máy, tóm tắt văn bản, nhận dạng giọng nói hay sinh nhãn ảnh (image captioning). Cách tính của BLEU là đếm số n-gram khớp nhau giữa câu được dự đoán và câu thực tế, sau đó chia số lượng ký tự của câu dự đoán. BLEU có khoảng giá trị $[0, 1]$, càng cao càng tốt.
- EM được tính bằng cách lấy số lượng ký hiệu trong câu dự đoán giống với trong câu thực tế chia cho số lượng ký hiệu latex có trong câu, khoảng giá trị $[0, 1]$, càng cao càng tốt.
- Edit distance là phép đo số lượng thay đổi chúng ta phải thực hiện đối với một chuỗi để biến đổi nó thành chuỗi mà chúng ta đang so sánh với, các phép đo bao gồm: thêm, xóa, sửa, khoảng giá trị $[0, 1]$, càng thấp càng tốt.

4) Mô hình & tham số

Để kiểm tra kết quả và khả năng của các mô hình, nhóm có tổng hợp các phương pháp trên để đem đi so sánh. Cụ thể các mô hình sẽ có Encoder của một trong số Convolution Encoder, Convolution Encoder + Batch Normalization, Convolution Row Encoder, ResNet-18, ResNet-18 Row Encoder (ResNet-18 + BiLSTM Row Encoder). Đối với V, số channel đầu ra luôn luôn là 512 (BiLSTM Row Encoder có kích thước lớp ẩn là 256, sau khi đi qua BiLSTM sẽ trở thành 512).

Decoder luôn luôn là LSTM 1 lớp có 512 node của lớp ẩn.

5) Thông số huấn luyện

Nhóm sử dụng optimizer AdamW [15] với learning rate $\eta = 0.002 * \sqrt{\text{batch size}}$, kết hợp thêm OneCycleLR [16] để lập lịch cho learning rate với tỉ lệ bước để learning rate lên vị trí cao nhất là 30% tổng số bước huấn luyện.

Do sự khác nhau giữa kích thước mô hình và phần cứng hạn chế (Kaggle 1 GPU P100), nhóm sử dụng các kích thước batch size khác nhau cho các mô hình (1, 2, 8, ... sao cho vừa GPU), nhưng nhóm kết hợp thêm kỹ thuật Accumulate Gradient [17] để tổng hợp các batch size nhỏ nhiều lần thành batch size 64, nên có thể nói các mô hình được huấn luyện trên batch size là 64.

Số epoch tối đa là 10 cho các mô hình, tuy nhiên nhóm huấn luyện ResNet-18 và ResNet-18 Row Encoder lâu hơn, có epoch tối đa là 20.

6) Thông tin khác

Để cải thiện kết quả dự đoán, nhóm sử dụng Beam Search [18] trên mô hình với Beam Width là 5. Source code của nhóm có thể lấy ở https://github.com/tuanio/image2latex.

B. Kết quả

Train loss (hình 6) cho thấy Convolution Encoder và Convolution Encoder với Batch Norm có mức độ dao động cao nhất, 3 mô hình còn lại có giá trị loss theo thời gian nhìn ổn định hơn (ResNet-18 Row Encoder đôi khi vẫn có dao động). Ở validation edit distance (hình 9) hay validation exact match (hình 10) chỉ có duy nhất Convolution Row Encoder có sự thay đổi trong giá trị, trong khi các mô hình khác không có sự thay đổi theo thời gian, còn ở validation bleu (hình 8) thì tất cả mô hình đều thay đổi, chỉ có Convolution Row Encoder có sự thay đổi rõ ràng nhất, kết hợp thêm kết quả từ bảng I chứng tỏ Convolution Row Encoder học ổn nhất trong các mô hình. Bảng II trình bày một số mẫu câu latex được dự đoán ra.

Bảng I: Bảng so sánh giá trị BLEU4, EM và Edit distance của nhóm với hai bài nghiên cứu trong phần II. Kết quả tốt nhất của nhóm và kết quả tốt nhất của các nhóm khác được in đậm. Các giá trị trong bảng đã được $\times 100$.

Mô hình	**BLEU4** (%)	**EM** (%)	**Edit distance** (%)
Convolution Encoder	11.18	0	100
+ Batch Normalization	14.3	0	100
Convolution Row Encoder	**77**	**16.52**	**31.23**
ResNet-18	25.54	0.02	100
ResNet-18 Row Encoder	18.5	0	100
G. Genthial và cộng sự [1]	78	35	24
Y. Deng và cộng sự [2]	87.73	77.46	-

Ngoài ra, kết quả của bảng I cho ta thấy được:

- Việc dùng một BiLSTM cho Encoder đem lại kết quả tốt khi kết hợp với Convolution Feature Encoder (Kết quả có sự thay đổi rõ rệt từ Convolution Encoder, Convolution Encoder + Batch Normalization sang Convolution Row Encoder).
- Batch Normalization cải thiện kết quả đánh giá và giúp quá trình huấn luyện ổn định hơn (dù chỉ một ít) (hình 6).
- Đối với ResNet-18, mô hình này có sự kết hợp các lớp CNN sâu (18 lớp), sẽ khiến Feature Grid V đầu ra có kích thước của $W' \times H'$ nhỏ, vì thế số hàng có thể học sẽ ít, khiến BiLSTM không thực sự phát huy tác dụng.
- Tuy nhiên mô hình Convolution Encoder hay Convolution Encoder + Batch Normalization có số lớp CNN nhỏ (không sâu) như ResNet-18, nhưng cũng không có kết quả tốt, chứng tỏ cần phải cân bằng giữa mức độ sâu của các lớp CNN và Batch Normalization.

Bảng II: Bảng trình bày một số câu dự đoán từ mô hình Convolution Row Encoder.

Thực tế	**Dự đoán**
$\mathcal{L}(J) = \frac{1}{2}\partial_\mu\phi\partial^\mu\phi + \frac{J}{2}\phi^2 + \frac{\lambda\mu^{2\varepsilon}}{4!}\phi^4 + \mathcal{L}_{\mathrm{CT}}(J) - \mu^{-2\varepsilon}\frac{\zeta}{2}J^2.$	$\mathcal{L}(J) = \frac{1}{2}\partial_\mu\phi\partial^\mu\phi + \frac{J}{2}\phi^2 + \frac{\lambda\mu^{2s}}{4!}\phi^4 + \mathcal{L}_{\mathrm{CT}}(J) - \mu^{-2\varepsilon}\frac{\zeta}{2}J^2.$
$\frac{d^2G(\mu)}{d\mu^2} + (d/l)\coth(\mu/l)\frac{dG(\mu)}{d\mu} - m^2G(\mu) = 0,$	$\frac{d^2G(\mu)}{d\mu^2} + (d/l)\coth(\mu/l)\frac{dG(\mu)}{d\mu} - m^2G(\mu) = 0,$
$T_{\mu\nu} = (\partial_\mu\phi)(\partial_\nu\phi) - g_{\mu\nu}\frac{1}{2}[g^{\alpha\beta}(\partial_\alpha\phi)(\partial_\beta\phi) - m^2\phi^2].$	$T_{\mu\nu} = (\partial_\mu\phi)(\partial_\nu\phi) - g_{\mu\nu}\frac{1}{2}[g^{\alpha\beta}(\partial_\alpha\phi)(\partial_\beta\phi) - m^2\phi^2].$
$d^k_{m,1} + d^k_{1,n} - d^k_{m,n} = \frac{(m-1)(n-1)}{4} \quad \forall k.$	$d^k_{m,1} + d^k_{1,n} - d^k_{m,n} = \frac{(m-1)(n-1)}{4} \quad \forall j,$
$\frac{\mathrm{d}}{\mathrm{d}z}p + \frac{\mathrm{d}}{\mathrm{d}\bar{z}}\bar{p} = \frac{\mathrm{d}}{\mathrm{d}z}\frac{\partial\mathcal{L}}{\partial v} + \frac{\mathrm{d}}{\mathrm{d}\bar{z}}\frac{\partial\mathcal{L}}{\partial\bar{v}} = \partial_\varphi\mathcal{L}.$	$\frac{\mathrm{d}}{dz}p + \frac{\mathrm{d}}{\mathrm{d}\bar{z}^\varphi}\bar{p} = \frac{\mathrm{d}}{\mathrm{d}z}\frac{\partial\mathcal{L}}{\partial v} + \frac{\mathrm{d}}{4\bar{z}}\frac{\partial\mathcal{L}}{\partial\bar{v}} = \partial_\varphi\mathcal{L}.$

V. Hướng nghiên cứu trong tương lai

Trong bài nghiên cứu tiếp theo, nhóm sẽ sử dụng mô hình Vision Transformer [19] để thực nghiệm cho bài toán này. Kết hợp thêm tập dữ liệu IM2LATEX-170K [20] để mô hình có thể học được nhiều dữ liệu hơn.

VI. Kết luận

Trong bài nghiên cứu này, nhóm đã trình bày về các cách kết hợp với nhau để tạo lên mô hình Encoder-Decoder dành cho bài toán Image to Latex. Kết quả tốt nhất đạt được từ Convolution Row Encoder có BLEU là 77%, EM 16.52% và Edit distance 31.23% trên tập dữ liệu IM2LATEX-100K, chứng tỏ được rằng thêm một BiLSTM vào encoder giúp học ngữ cảnh tốt hơn. Bên cạnh đó, thực nghiệm cũng cho thấy mô hình ResNet-18 có số lượng lớp quá sâu có thể không thích hợp với bài toán Image to Latex, khi mà thông tin từ ảnh chỉ là trắng đen và cần phải cân bằng giữa số lượng các lớp CNN và Batch Normalization để mô hình có khả năng tổng quát hóa dữ liệu hơn.

TÀI LIỆU

[1] G. Genthial và R. Sauvestre, "Image to Latex," Truy cập ở http://cs231n.stanford.edu/reports/2017/pdfs/815.pdf (2017).

[2] Y. Deng, A. Kanervisto và A. M. Rush, "What You Get Is What You See: A Visual Markup Decompiler," Truy cập ở http://arxiv.org/abs/1609.04938 (2016).

[3] Paper With Code, "Image Captioning," Truy cập ở https://paperswithcode.com/task/image-captioning (Truy cập lần cuối vào 2022).

[4] Paper With Code, "Speech Recognition," Truy cập ở https://paperswithcode.com/task/speech-recognition (Truy cập lần cuối vào 10/2022).

[5] Paper With Code, "Optical Character Recognition", Truy cập ở https://paperswithcode.com/task/optical-character-recognition (Truy cập lần cuối vào 10/2022).

[6] S. Ioffe và C. Szegedy, "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift," Truy cập ở http://arxiv.org/abs/1502.03167 (2015).

[7] K. Anssi, "im2latex-100k," arXiv:1609.04938 [Data set]. Zenodo. doi: 10.5281/zenodo.56198 (2016)

[8] K. He, X. Zhang, S. Ren, J. Sun, "Deep Residual Learning for Image Recognition,", arXiv: 1512.03385, doi: 10.48550/ARXIV.1512.03385 (2015)

[9] W. Meng, L. Huafeng, L. Fang, "Generative Adversarial Network based on Resnet for Conditional Image Restoration," arXiv:1707.04881, doi: 10.48550/ARXIV.1707.04881 (2017)

[10] H. Foysal, L. H. Youn, K. D. Seong, "Object Detection Based on VGG with ResNet Network,". pp. 1-3. doi: 10.23919/ELINFOCOM.2019.8706476 (2019).

[11] A. Youssef, "Image Downsampling and Upsampling Methods," Truy cập tại: https://www2.seas.gwu.edu/ ayoussef/papers/ImageDownUpSampling-CISST99.pdf

[12] M. T. Luong, H. Pham, C. D. Manning, "Effective Approaches to Attention-based Neural Machine Translation," arXiv:1508.04025, doi: 10.48550/ARXIV.1508.04025 (2015).

[13] W. Hu, L. Xiao, J. Pennington, "Provable Benefit of Orthogonal Initialization in Optimizing Deep Linear Networks," arXiv:2001.05992, doi: 10.48550/ARXIV.2001.05992 (2020).

[14] Kanervisto, Anssi. (2016). im2latex-100k , arXiv:1609.04938 [Data set]. Zenodo. doi: 10.5281/zenodo.56198.

[15] I. Loshchilov, F. Hutter, "Decoupled Weight Decay Regularization," arXiv:1711.05101, doi: 10.48550/ARXIV.1711.05101 (2017).

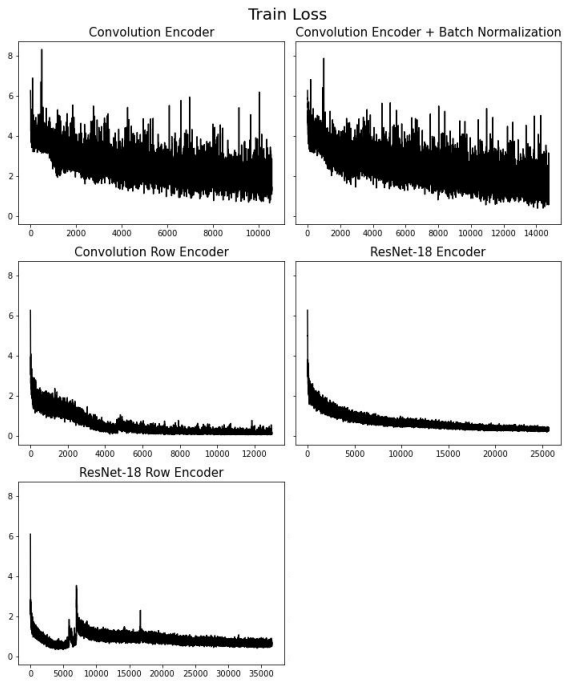
[16] L. N. Smith, N. Topin, "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates," arXiv:1708.07120, doi: 10.48550/ARXIV.1708.07120 (2017).

[17] J. Hermans, G. Spanakis, R. Möckel, "Accumulated Gradient Normalization," arXiv:1710.02368, doi: 10.48550/ARXIV.1710.02368 (2017).

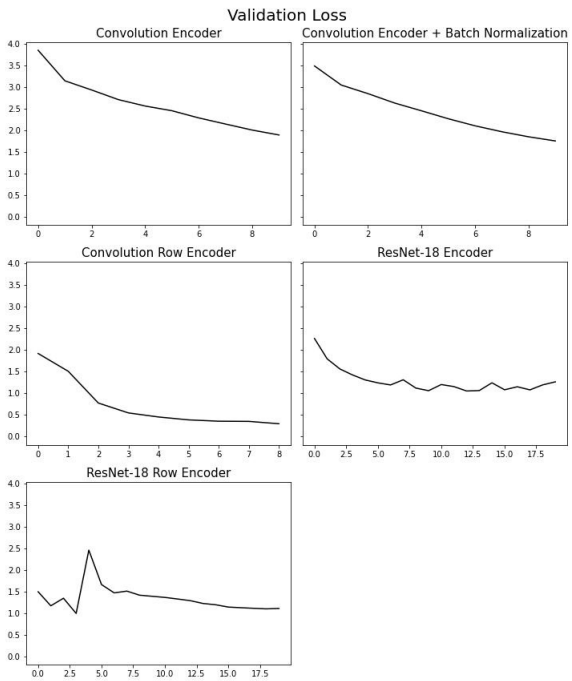
[18] Wikipedia, "Beam Search,". Truy cập tại:

[19] A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, "An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale," arXiv:2010.11929, doi: 10.48550/ARXIV.2010.11929 (2020).

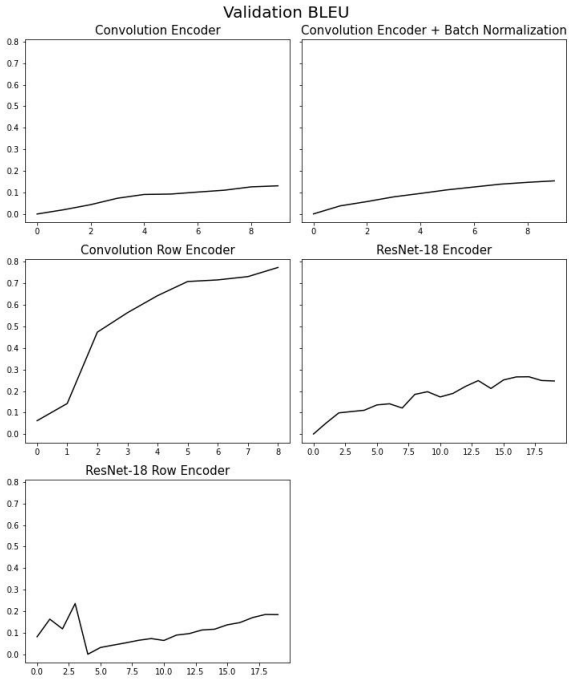
[20] Kaggle, "Image2Latex170k,". Truy cập ở https://www.kaggle.com/datasets/rvente/im2latex170k.



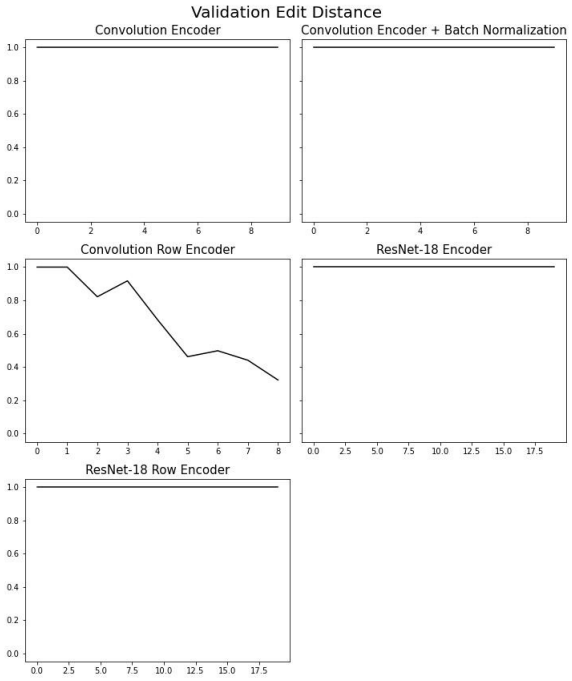
Hình 6: Train loss theo thời gian của các mô hình



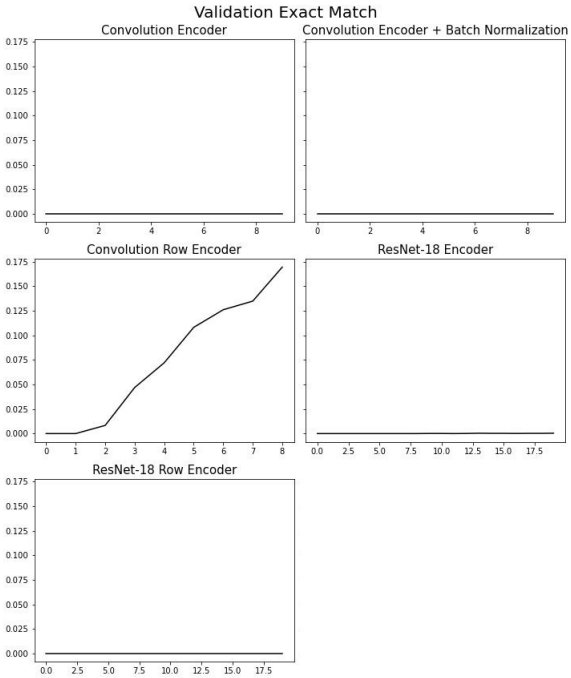
Hình 7: Validation loss theo thời gian của các mô hình



Hình 8: Validation BLEU4 theo thời gian của các mô hình



Hình 9: Validation Edit Distance theo thời gian của các mô hình



Hình 10: Validation EM theo thời gian của các mô hình