Tạp chí Khoa học Trường Đại học Cần Thơ
Phần A: Khoa học Tự nhiên, Công nghệ và Môi trường
website: sj.ctu.edu.vn

DOI:10.22144/ctu.jsi.2017.017

TÓM TẮT VĂN BẢN TIẾNG VIỆT TỰ ĐỘNG VỚI MÔ HÌNH SEQUENCE-TO-SEQUENCE

Lâm Quang Tường[1,2] , Phạm Thế Phi[1] và Đỗ Đức Hào[2,3]

[1]*Khoa Công nghệ Thông tin và Truyền thông, Trường Đại học Cần Thơ*
[2]*Phòng R&D Công ty TNHH Công nghệ Olli*
[3]*Trường Đại học Khoa học Tự nhiên, Đại học Quốc gia Thành phố Hồ Chí Minh*

Thông tin chung:
Ngày nhận bài: 15/09/2017
Ngày nhận bài sửa: 10/10/2017
Ngày duyệt đăng: 20/10/2017

Title:
Vietnamese text summarization with Sequence-to-Sequence

Từ khóa:
Beam Search, học sâu, Sequence-to-sequence, tóm tắt văn bản

Keywords:
Beam Search, deep learning, Sequence-to-sequence, text summarization

ABSTRACT

Deep learning is a machine learning method that has been studied and used extensively in recent years, opening up new directions for problems such as image processing, speech processing, and natural language processing, etc. This article focuses on the use of deep learning for automatic text summarization for Vietnamese. Previous approaches such as statistics, machine learning, language analysis, etc. have been successful at different levels and purposes. In this paper, the Word2vec model was used to extract the specific characteristics of Vietnamese text for the Sequence to Sequence with Attention model to produce a sequence of words. Finally, the results were re-selected using the Beam Search algorithm, and a summary sentence was generated. The accuracy of the model was estimated using the ROUGE method on a dataset of over twenty-seven million words collected from newspapers in the country. The result was the summary statement reflecting the text content. Although the results were not high yet, the model has successfully solved the problem, and the dataset needs improving to enhance the efficiency of the model.

TÓM TẮT

Học sâu là phương pháp học máy được nghiên cứu và sử dụng rộng rãi trong những năm gần đây, mở ra hướng đi mới cho các bài toán như xử lý ảnh, xử lý tiếng nói và xử lý ngôn ngữ tự nhiên... Bài báo tập trung nghiên cứu sử dụng học sâu cho bài toán tóm tắt văn bản tự động đối với tiếng Việt. Các hướng tiếp cận trước đây như: thống kê, máy học, phân tích ngôn ngữ... đã thành công trên những cấp độ và mục đích tóm tắt khác nhau. Trong bài báo này, chúng tôi sử dụng mô hình Word2vec để rút trích những đặc trưng riêng của văn bản tiếng Việt, phục vụ cho mô hình Sequence to sequence with Attention nhằm tạo kết quả đầu ra là chuỗi các từ. Cuối cùng kết quả được chọn lọc lại bằng giải thuật Beam Search và sinh ra câu tóm tắt. Độ chính xác của mô hình được đánh giá bằng phương pháp ROUGE trên tập dữ liệu hơn hai mươi bảy triệu từ thu thập từ các trang báo trong nước. Kết quả thu được là các câu tóm tắt phản ánh đúng nội dung văn bản. Tuy kết quả còn chưa cao nhưng mô hình đã giải quyết thành công mục tiêu của bài toán, chúng tôi sẽ cố gắng cải thiện tập dữ liệu để nâng cao hiệu quả của mô hình.

Trích dẫn: Lâm Quang Tường, Phạm Thế Phi và Đỗ Đức Hào, 2017. Tóm tắt văn bản tiếng Việt tự động với mô hình Sequence-to-Sequence. Tạp chí Khoa học Trường Đại học Cần Thơ. Số chuyên đề: Công nghệ thông tin: 125-132.

1 GIỚI THIỆU

Tóm tắt văn bản là quá trình rút trích những thông tin quan trọng nhất từ một văn bản để tạo ra phiên bản ngắn gọn, xúc tích mang lại đầy đủ lượng thông tin của văn bản gốc kèm theo đó là tính đúng đắn về ngữ pháp và chính tả. Bản tóm tắt phải giữ được những thông tin quan trọng của toàn bộ văn bản chính. Bên cạnh đó, bản tóm tắt cần phải có bố cục chặt chẽ có tính đến các thông số như độ dài câu, phong cách viết và cú pháp của văn bản.

Nhìn chung, tóm tắt văn bản có hai hướng tiếp cận: tóm tắt kiểu trích chọn-"extraction" và tóm tắt kiểu tóm lược ý -"abstraction". Phương pháp "extraction" làm công việc chọn ra một tập con của những từ đã có, những lời nói hoặc những câu trong văn bản gốc để đưa vào khuôn mẫu tóm tắt. Ngược lại, phương pháp "abstraction" xây dựng một biểu diễn ngữ nghĩa bên trong và sau đó sử dụng kĩ thuật xử lý ngôn ngữ để tạo ra một bản tóm tắt gần gũi hơn so với những gì con người có thể tạo ra. Bản tóm tắt như vậy có thể chứa các từ không có trong bản gốc. Nghiên cứu về phương pháp "abstraction" là một bước tiến quan trọng và tạo sự chủ động, tuy nhiên do các ràng buộc phức tạp nên nghiên cứu cho đến nay đã tập trung chủ yếu vào phương pháp "extraction". Trong một vài lĩnh vực ứng dụng, phương pháp "extraction" mang lại nhiều tri thức hơn.

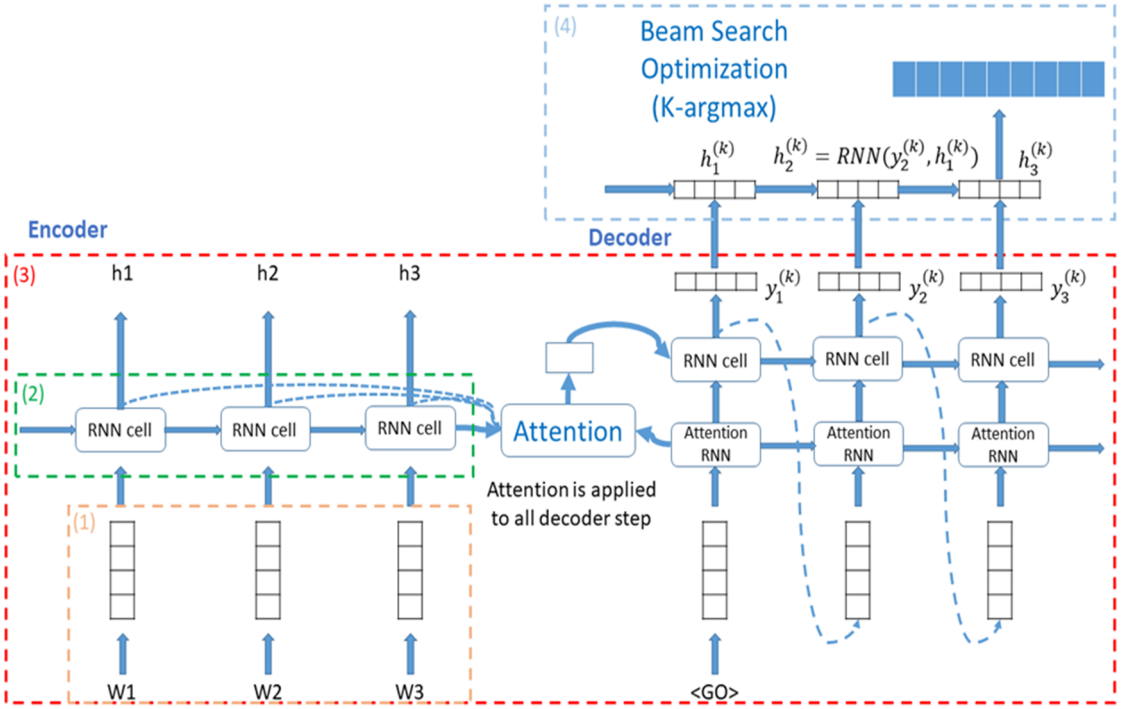
Kỹ thuật tóm tắt phổ biến và gần đây nhất sử dụng phương pháp thống kê hoặc các kĩ thuật xử lý ngôn ngữ. Các từ có tần số cao, từ khóa chuẩn, phương pháp tiêu đề, phương pháp vị trí được sử dụng làm trọng số của câu. Đối với hướng tiếp cận cho tóm tắt đơn văn bản ta có các phương pháp như: thống kê, TF.IDF, máy học (Naïve-Bayes, Decision Tree, Hidden Makov Model, Log-Linear, Neural Network, SVM), phân tích ngôn ngữ tự nhiên (Ono *et al.,* 1994, Barzilay and Elhadad, 1997), (Marcu, D. 1998). Đối với hướng tiếp cận cho tóm tắt đa văn bản: Phương pháp dùng template (McKeown and Radev, 1995-1998), gom cụm chủ đề và hợp nhất thông tin (McKeown, 1999, Barzilay *et al.,* 1999), gom cụm (cluster-based) với MMR, gom cụm với lý thuyết đồ thị, kích hoạt lan truyền trên đồ thị, phương pháp dựa trên trọng tâm.

Mỗi ngôn ngữ đều có những đặc trưng riêng ở các mức độ hình thái, mức độ cú pháp và mức độ ngữ nghĩa. Đối với tiếng Việt có những nét đặc trưng về từ loại, cấu trúc câu, dấu câu… rất cần phải xử lý và khai thác. Các nghiên cứu trước đây cho tóm tắt văn bản tiếng Việt như: "Một giải pháp tóm tắt văn bản tiếng Việt tự động" của nhóm tác giả Trương Quốc Định và Nguyễn Quang Dũng (2012) nghiên cứu về các giải thuật thống kê dựa trên từ vựng để xác định độ tương tự giữa các câu, "Ứng dụng mô hình đồ thị trong tóm tắt đa văn bản tiếng việt" (Nguyễn Thị Ngọc Tú *và ctv*., 2015); sử dụng mô hình đồ thị có trọng số và trọng số của câu ở mỗi nút…

Trong những năm gần đây, cùng với sự phát triển mạnh mẽ của các mô hình huấn luyện end-to-end đã tạo ra hướng đi mới để giải quyết bài toán tóm tắt văn bản tự động. Cụ thể bài báo này đề cặp đến tóm tắt văn bản tự động với mô hình Sequence-to-Sequence (Sutskever *et al.,* 2014). Dựa vào ý tưởng sinh ra câu mới từ câu đã có của mô hình Sequence-to-Sequence, chúng tôi xây dựng mô hình rút trích những thông tin quan trọng của cả đoạn văn, thu gọn lại thành một câu tóm tắt mang đầy đủ thông tin của đoạn văn; bên cạnh đó, phối hợp với các phương pháp xử lý ngôn ngữ tự nhiên như tách từ, loại bỏ dấu câu… để tinh chỉnh và xử lý tiếng Việt.

2 TÓM TẮT VĂN BẢN TỰ ĐỘNG VỚI MÔ HÌNH SEQUENCE TO SEQUENCE

Quá trình tóm tắt văn bản tự động được huấn luyện end-to-end với mô hình Sequence-to-sequence with Attention (Hình 1) (Nallapati *et al.,* 2016). Tại bước encoder, đầu vào của Recurrent Neural Networks (RNN) là các vector được tạo ra bằng cách mã hoá chuỗi từ với mô hình word embedding. Khi decoder, sử dụng một RNN để sinh ra chuỗi từ mới dựa vào chuỗi đầu vào và các từ được sinh ra phía trước. Trong mô hình tóm tắt văn bản tự động, thay vì tìm ra xác suất lớn nhất của mỗi từ sinh ra tại bước decoder, chúng ta tạo ra danh sách các từ ứng viên tại mỗi bước giải mã. Sau đó, sử dụng giải thuật Beam Search để lựa chọn các từ ứng viên và kết nối danh sách các từ ứng viên đó lại tạo thành một câu có điểm số cao nhất tạo ra chuỗi tóm tắt.



Hình 1: Tóm tắt văn bản tự động sử dụng mô hình Sequence-to-Sequence with Attention

(1) Word Embedding

(2) Recurrent Neural Network

(3) Sequence-to-Sequence with Attention

(4) Beam Search with Optimization

(1) Mô hình Word Embedding

Mục tiêu của word embedding là mã hoá từ ngữ tiếng Việt thành những vector đặc trưng mang lại một ý nghĩa ở mức độ nào đó cho từ và nó là phương pháp biểu diễn dữ liệu cho mô hình tóm tắt văn bản tự động. Ngoài cách biển diễn theo mô hình túi từ (Bag of word), xây dựng dựa trên số lượng từ được sử dụng trong bài toán Text Classification, chúng ta có hai phương pháp mới là Word2vec (Rong, 2016) và Glove. Ở đây, chúng tôi tập trung đề cập đến mô hình Word2vec. Word2vec là một phương thức biểu diễn mỗi từ thành một vector có các phần tử mang giá trị diễn tả mối quan hệ giữa từ này với từ khác bằng cách sử dụng một mạng nơ-ron với duy nhất một trạng thái ẩn. Đối với tiếng Việt, trước tiên, các từ được xử lý chuyển về từ đơn, từ ghép, và in thường rồi loại bỏ các dấu câu ngoại trừ dấu chấm để phân biệt các câu với nhau. Sau đó, mỗi từ trong câu sẽ được sắp xếp theo thứ tự từ điển và chuyển về dạng one-hot vector. One-hot vector là một vector có độ dài tối đa bằng với kích thước tập từ vựng, chỉ có duy nhất một vị trí có giá trị "1" tương ứng với vị trí của từ cần biểu diễn trong tập từ vựng, các vị trí còn lại đều bằng "0". Những vector này là đầu vào của mạng nơ-ron. Trong nghiên cứu của Mikolov *et al*. (2013), tác giả đề xuất hai kiến trúc mạng nơ-ron để xây dựng Word2vec đó là Continuous Bag-of-Words model (CBOW) và Continuous Skipgram model. Hai mô hình trên giống nhau đều sử dụng mạng nơ-ron nhưng chỉ khác nhau ở đầu vào và đầu ra. Đối với Continuous Skip-gram model đầu vào chỉ là một one-hot vector của một từ và đầu ra là các vector biểu diễn quan hệ với các từ còn lại. Còn đối với CBOW thì ngược lại, mục đích để tìm mối quan hệ xác suất của nhiều từ đầu vào đối với một từ đầu ra. Cụ thể như sau:

Cho C word vectors đầu vào, hidden layer $\mathbf{h}$ được tính bằng cách lấy trung bình cộng của các dòng trong ma trận W. Từ hidden layer ra được output layer, ma trận trọng số W' được sử dụng để bỏ phiếu (scoring) cho các từ trong từ điển. Đặt $u_i = W'^T_i \times h$, $W'_i = v_w$, ta được: $u_i = v_w^T \times h = v_w^T \times v_c$. Softmax function được sử dụng để thu về phân phối hậu nghiệm (posterior distribution) cho các từ này.

$$y_i = \frac{e^{v_w^T v_c}}{\sum_{w' \in T} e^{v_{w'}^T v_c}} = P(w|c)$$

Sử dụng maximine likelihood để tối thiểu hóa hàm lỗi:

$$\theta = \{v_w, v_c\}$$

$$likelihood\ (\theta) = \prod_{w \in T} P(w|c;\theta)$$

$$l(\theta) = \sum_{w \in T} \log P\ (w|c;\theta)$$

$$= \sum_{w \in T} log \frac{e^{v_w^T v_c}}{\sum_{w' \in T} e^{v_{w'}^T v_c}}$$

$$= \sum_{w \in T} v_w^T v_c - log \sum_{w' \in T} v_{w'}^T v_c$$

$$\frac{\partial l}{\partial v_w} = v_c - \frac{1}{\sum_{w' \in T} e^{v_{w'}^T v_c}} \times e^{v_w^T v_c} \times v_c$$

$$= v_c - P(w|c) \times v_c$$

$$= v_c \times [1 - P(w|c)]$$

Từ đó ta có công thức truy hồi cho v_w và v_c bằng phương pháp trượt dốc gradient như sau:

$$v_w = v_w - \eta \times v_c \times [1 - P(w|c)]$$

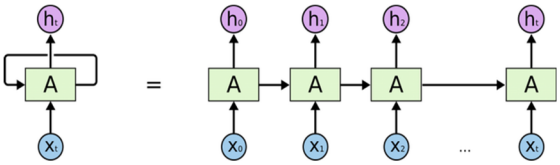
$$v_c = v_c - \eta \times [1 - P(w|c)]$$

(2) Recurrent Neural Networks

Trong một mô hình ngôn ngữ việc tính xác suất $p(w_t|w_{t-1}, \dots, w_1)$ được xấp xỉ bằng cách sử dụng mô hình Makov bậc N:

$$p(w_t|w_{t-1}, \dots, w_1) \approx p(w_t|w_{t-1}, .., w_{t-N})$$

nó chưa biểu diễn một cách chính xác xác suất của một từ khi biết từ phía trước nếu chúng nằm cách xa nhau. Chính vì thế, chúng ta cần có một mô hình có thể tính được $p(w_t|w_{t-1}, \dots, w_1)$, từ đó ý tưởng mô hình Recurrent Neural Network (RNN) (Cho *et al.,* 2015) ra đời. RNN được hiểu cơ bản là một mạng nơ-ron có vòng lặp với trạng thái ẩn đầu ra của một nơ-ron lại trở thành đầu vào của nơ-ron kế tiếp từ đó có thể lưu giữ được thông tin của chuỗi đầu vào, được mô tả ở Hình 2.



Hình 2: Recurrent Neural Networks (Colah, 2015)

RNN nhận đầu vào là một chuỗi $(x_1, \dots, x_T)$ là kết quả của mô hình word embedding để xây dựng một chuỗi đầu ra $(y_1, \dots, y_T)$ bằng cách lặp lại các phương trình:

$$h_t = sigm(W^{hx} x_t + W^{hh} h_{t-1})$$

$$y_t = W^{yh} h_t$$

Trong đó:

x_t là input tại thời điểm thứ t,

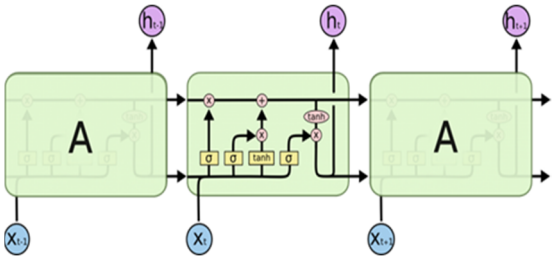
h_t là trạng thái ẩn tại thời điểm thứ t,

y_t là output tại thời điểm t, chứa xác suất của các từ trong bộ từ vựng.

Khi huấn luyện RNN chúng ta sử dụng kĩ thuật Back-Propagation Through Time (BPPTT) để cộng dồn gradient của các bước trước lại với nhau. Đây là một biện pháp để giải quyết vấn đề gradient hội tụ về 0 qua các bước lặp (vanishing problem) nhưng cũng cần điều chỉnh phù hợp để gradient không phân kì (exploding problem). Đó cũng là vấn đề khiến RNN gặp khó khăn trong nhiều năm và Long-Short Term Memory được sinh ra để giải quyết vấn đề này.

Long Short Term Memory

Long Short Term Memory networks – thường được gọi là "LSTM", là trường hợp đặc biệt của RNN, có khả năng học với sự phụ thuộc lâu dài của các nơ-ron. Mô hình này được giới thiệu bởi Hochreiter & Schmidhuber (1997), và được cải tiến lại bởi Ayako Mikami (2016). Mục tiêu chính của LSTM là quyết định thông tin nào được lưu lại và loại bỏ tại mỗi nơ-ron của RNN, khắc phục hiệu ứng vanishing/exploding gradient. LSTM sử dụng ba cổng sigmoid và một cổng tanh thay cho một nơ-ron của RNN được mô tả ở Hình 3. Các cổng làm nhiệm vụ xác định thông tin nào được lưu lại, loại bỏ, sinh ra ứng viên mới và quyết định thông tin đầu ra là gì.



Hình 3: LSTM chứa bốn lớp tương tác (Colah, 2015)

(3) Sequence-to-sequence with Attention

Mục đích của việc tóm tắt văn bản là chuyển từ một văn bản gốc thành một văn bản có độ dài ngắn hơn và mang đầy đủ ý nghĩa. Theo mục đích đó, chúng ta xây dựng một RNN để rút trích đặc trưng từ chuỗi đầu vào và sinh ra chuỗi từ mới phụ thuộc vào chuỗi từ trước đó bằng một RNN thứ hai. Hai RNN này được điều chỉnh trọng số phù hợp để chuỗi đầu ra ngắn hơn và phụ thuộc xác suất chuỗi đầu vào. Đó chính là ý tưởng cơ bản của mô hình Sequence-to-Sequence (seq2seq). Cụ thể mô hình seq2seq là mở rộng của mô hình "encoder-decoder" (Cho *et al*., 2014), đầu tiên sử dụng một mô hình mã

hóa (encoder) để chuyển đổi một đối tượng nguồn thành một biểu diễn mã hóa x. Mỗi lần các chuỗi đầu vào được mã hoá, mô hình seq2seq tạo ra một chuỗi mục tiêu sử dụng một bộ giải mã (decoder). Bộ giải mã được giao nhiệm vụ tạo ra một chuỗi từ mục tiêu từ bộ từ vựng mục tiêu. Các từ được tạo ra tuần tự bằng cách điều chỉnh trên biểu diễn đầu vào và trên các từ được tạo ra trước đó. Sau đây là cụ thể về mô hình Sequence-to-Sequence:

Đặt $w_{1:T}$ là chuỗi từ bất kì có độ dài T, $y_{1:T}$ là chuỗi từ mục tiêu cho mỗi input x. Đặt $m_1, \dots, m_T$ là dãy gồm T vector và h_0 là trạng thái khởi tạo của vector. Áp dụng một RNN encoder cho chuỗi như vậy mang lại các trạng thái ẩn h_t tại mỗi bước thời gian t, như sau:

$$h_t \leftarrow RNN(m_t, h_t, \theta)$$

θ là danh sách các tham số của mô hình, được chia theo thời gian. Vector m_t tương ứng với sự gắn kết của một chuỗi từ mục tiêu $w_{1:T}$. Vì vậy, ta có thể viết:

$$h_t \leftarrow RNN(w_t, h_t, \theta)$$

RNN decoder thường được huấn luyện để hoạt động như các mô hình ngôn ngữ có điều kiện. Đó là cố gắng tính toán xác suất của từ mục tiêu thứ *t* đảm bảo điều kiện *x* khi biết *t-1* từ trước đó:

$$p(w_t|w_{1:t-1}, x) = g(w_t, h_{t-1}, x)$$

với một vài tham số của hàm g được tính toán với một lớp affine theo một hàm softmax. Trong việc tính toán xác suất này, trạng thái h_{t-1} đại diện cho các từ đã được huấn luyện trước từ mục tiêu và h_0 được thiết lập thành một số hàm của biến x. Mô hình hoàn chỉnh (bao gồm encoder) được huấn luyện, tương tự như mô hình ngôn ngữ nơ ron, cần phải tối thiểu hóa hàm lỗi (cross-entropy loss) tại mỗi lần lặp trong khi việc điều chỉnh những từ mục tiêu đã được tạo ra trước đó trong tập dữ liệu huấn luyện. Đó là, mô hình được huấn luyện để tối thiểu hóa:

$$-ln \prod_{t=1}^{T} p(y_t|y_{1:t-1}, x)$$

Encoder và decoder khác trọng số với nhau. Trong mô hình seq2seq cơ bản, mỗi đầu vào phải được mã hóa thành một vector trạng thái có chiều dài cố định, bởi vì đó là thứ duy nhất được truyền đến bộ giải mã. Cơ chế Attention cho phép bộ giải mã truy cập trực tiếp vào đầu vào.

***Attention**

Mô hình Sequence-to-Sequence có thể bị phá vỡ khi chuỗi đầu vào quá dài. Nguyên nhân là ở mỗi bước nếu chỉ có một vector ngữ cảnh *c* giao tiếp giữa encoder và decoder, vector đó sẽ phải mã hoá cho toàn bộ chuỗi đầu vào, dẫn đến nó có thể bị tan biến khi nó xử lý chuỗi từ dài. Kĩ thuật Attention (Bahdanau *et al.,* 2016) cho phép bộ giải mã tập trung vào một phần khác nhau từ đầu ra của encoder bằng tìm ra chuỗi vector ngữ cảnh của mỗi bước encoder $c_{1:n}$ với

$$c_i = \sum_{j=1}^{n} \alpha_{ij} h_j$$

Trong đó, tập trọng số α_{ij} của mỗi trạng thái ẩn h_j là đầu ra của một hàm softmax:

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=0}^{n} \exp(e_{ik})}$$

$$e_{ij} = q(s_{i-1}, h_j)$$

với s_{i-1} là trạng thái ẩn tại bước *i-1* của decoder.

(4) Thuật toán Beam Search

Trong mô hình Sequence-to-Sequence, bộ giải mã được điều khiển bởi một câu đã được mã hoá để tạo ra một câu mới. Tại mỗi bước lặp t, bộ giải mã cần đưa ra quyết định từ nào để sinh ra như từ thứ t trong câu. Vấn đề là chúng ta không biết chính xác chuỗi từ cần phải sinh ra để cực đại hoá xác suất có điều kiện tổng thể. Để giải quyết vấn đề này, thuật toán Beam Search Optimization đã được Sam Wiseman và Alexander M. Rush đề xuất (2016). Beam Search với độ rộng K sao cho tại mỗi bước đưa ra K đề xuất và tiếp tục giải mã với một trong số chúng.

Thay vì tính xác suất của từ kế tiếp, chúng ta sử dụng cách tạo ra điểm số (không phải xác suất) cho các chuỗi câu. Mặc định điểm số của một chuỗi bao gồm các từ đứng trước $w_{1:t-1}$ theo sau bởi một từ w_t là $f(w_t, h_{t-1}, x)$, với f là một hàm kiểm tra trạng thái ẩn tại thời điểm $t-1$ của một RNN. Ở đây, hàm f không phải là một hàm softmax để tránh các vấn đề sai lệch nhãn. Thuật toán Beam Search sử dụng trong mô hình Sequence-to-Sequence cụ thể gồm 3 bước chính:

a. Search-Based Loss: sử dụng một hàm tính điểm để tính tổng điểm của mỗi chuỗi.

Cho tập S_t gồm K chuỗi ứng viên có độ dài t. Chúng ta có thể tính được thứ hạng cho mỗi chuỗi trong S_t sử dụng một hàm tuyến tính f với một RNN. Chúng ta khai báo một chuỗi $\hat{y}_{1:t}^{(K)}$ xếp hạng thứ K trong tập S_t dựa vào hàm f. Nghĩa là, ta có khoảng cách tối đa giữa các thứ hạng:

$$\left|\{\hat{y}_{1:t}^{(k)} \in S_t \,|f(\hat{y}_{1:t}^{(k)}, \hat{h}_{t-1}^{(k)}) > f(\hat{y}_t^{(K)}, \hat{h}_{t-1}^{(K)})\}\right| = K - 1$$

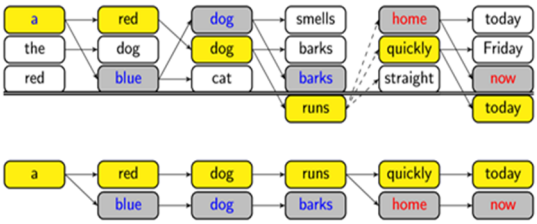
với $\hat{y}_t^{(k)}$ là token thứ t trong $\hat{y}_{1:t}^{(k)}$, $\hat{h}_{t-1}^{(k)}$ là trạng thái ẩn của RNN tương ứng ở bước thứ *t - 1*.

Hàm $\mathcal{L}(f)$ thể hiện lỗi tại mỗi lần điểm số của chuỗi $y_{1:t}$ không vượt quá điểm số của chuỗi $\hat{y}_{1:t}^{(k)}$ tính bằng công thức:

$$\mathcal{L}(f) = \sum_{t=1}^{T} \Delta(\hat{y}_{1:t}^{(K)})[1 - f(y_t, h_{t-1}) + f(\hat{y}_t^{(K)}, \hat{h}_{t-1}^{(K)})]$$

với $\Delta(\hat{y}_{1:t}^{(K)})$ biểu thị độ lỗi cụ thể của hàm lỗi; nó trả về giá trị 0 khi các yêu cầu lề thỏa mãn và ngược lại trả về một số dương. Hàm lỗi được tối ưu ở hai bước tiếp theo.

b. Forward: (Find Violations) xây dựng tập chuỗi S_t và tìm kiếm vi phạm lề mô tả ở Hình 4.



Hình 4: Beam search với vi phạm lề [7]

Để tối hiểu hóa hàm lỗi, ta xét các trường hợp như sau:

Nếu không có vi phạm lề ở bước $t-1$ thì S_t được xây dựng từ giải thuật Beam Search tiêu chuẩn.

Nếu có vi phạm lề, S_t được xây dựng từ K chuỗi, bắt đầu bằng chuỗi mục tiêu $y_{1:t-1}$.

Giả sử hàm *succ* ánh xạ một chuỗi $w_{1:t-1} \in \mathcal{V}^{t-1}$ vào một danh sách những chuỗi có độ dài t, điều này có thể làm bằng cách bổ sung cho nó một từ hợp lệ $w \in \mathcal{V}$. Trong trường hợp đơn giản, ta có:

$$succ(w_{1:t-1}) = \{w_{1:t-1}, w | w \in \mathcal{V}\}$$

Sau khi xác định một hàm *succ* thích hợp, chúng ta khai báo một tập hợp:

$$S_t = topK \begin{cases} succ(y_{1:t-1}) & vi\ phạm\ tại\ t-1 \\ \bigcup_{k=1}^{K} succ(\hat{y}_{1:t}^{(k)}) & ngược\ lại \end{cases}$$

ta có một vi phạm lề tại $t-1$ nếu $f(y_{t-1}, h_{t-2}) < f(\hat{y}_{t-1}^{(K)}, \hat{h}_{t-2}^{(K)}) + 1$ và $topK$ để xem xét điểm số cho bởi hàm f.

c. Backward: (Merge Sequences) sử dụng lan truyền ngược để tối ưu hàm lỗi của mô hình RNN.

Giả sử một vi phạm lề tại bước thứ t giữa chuỗi dự đoán $\hat{y}_{1:t}^{(K)}$ và chuỗi nhãn $y_{1:t}$. Mô hình Sequence-to-Sequence chuẩn thực hiện lan truyền ngược để tối ưu hàm lỗi thông qua các từ mục tiêu; tuy nhiên ở đây chúng ta có thêm một gradient cho các từ dự đoán sai trước đó. Để lan truyền ngược hàm lỗi thông qua một RNN, chúng ta sử dụng một thủ tục đệ quy *BRNN* – tại mỗi bước thứ t, nó chứa gradients của bước kế tiếp và lỗi trong tương lai đối với h_t. Ta có:

$$\nabla_{h_t}\mathcal{L} \leftarrow BRNN(\nabla_{h_t}\mathcal{L}_{t+1}, \nabla_{h_{t+1}}\mathcal{L})$$

với $\mathcal{L}_{t+1}$ là lỗi tại bước $t+1$. Chạy hàm BRNN từ $t = T-1$ đến $t = 0$ được xem như lan truyền ngược theo thời gian (back-propagation through time).

Giải thuật Beam Search sử dụng cho mô hình Sequence-to-Sequence được tóm tắt ở Hình 5:

Algorithm 1 Seq2seq Beam-Search Optimization

procedure BSO($\boldsymbol{x}, K_{tr}$, succ)
  /*FORWARD*/
  Init empty storage $\hat{y}_{1:T}$ and $\hat{\boldsymbol{h}}_{1:T}$; init S_1
  $r \leftarrow 0$; $violations \leftarrow \{0\}$
  for $t = 1, \ldots, T$ **do**
    $K = K_{tr}$ if $t \neq T$ else $\underset{k:\hat{y}_{1:t}^{(k)} \neq y_{1:t}}{\arg\max} f(\hat{y}_t^{(k)}, \hat{\boldsymbol{h}}_{t-1}^{(k)})$
    if $f(y_t, \boldsymbol{h}_{t-1}) < f(\hat{y}_t^{(K)}, \hat{\boldsymbol{h}}_{t-1}^{(K)}) + 1$ **then**
      $\hat{\boldsymbol{h}}_{r:t-1} \leftarrow \hat{\boldsymbol{h}}_{r:t-1}^{(K)}$
      $\hat{y}_{r+1:t} \leftarrow \hat{y}_{r+1:t}^{(K)}$
      Add t to $violations$
      $r \leftarrow t$
      $S_{t+1} \leftarrow \text{topK}(\text{succ}(y_{1:t}))$
    else
      $S_{t+1} \leftarrow \text{topK}(\bigcup_{k=1}^{K} \text{succ}(\hat{y}_{1:t}^{(k)}))$
  /*BACKWARD*/
  $grad\_\boldsymbol{h}_T \leftarrow \mathbf{0}$; $grad\_\widehat{\boldsymbol{h}}_T \leftarrow \mathbf{0}$
  for $t = T-1, \ldots, 1$ **do**
    $grad\_\boldsymbol{h}_t \leftarrow \textbf{BRNN}(\nabla_{\boldsymbol{h}_t}\mathcal{L}_{t+1}, grad\_\boldsymbol{h}_{t+1})$
    $grad\_\widehat{\boldsymbol{h}}_t \leftarrow \textbf{BRNN}(\nabla_{\widehat{\boldsymbol{h}}_t}\mathcal{L}_{t+1}, grad\_\widehat{\boldsymbol{h}}_{t+1})$
    if $t-1 \in violations$ **then**
      $grad\_\boldsymbol{h}_t \leftarrow grad\_\boldsymbol{h}_t + grad\_\widehat{\boldsymbol{h}}_t$
      $grad\_\widehat{\boldsymbol{h}}_t \leftarrow \mathbf{0}$

Hình 5: Sequence-to-Sequence as

Beam Search Optimization [8]

3 KẾT QUẢ THỰC NGHIỆM

3.1 Xây dựng tập dữ liệu

Để xây dựng thành công mô hình trên tập dữ liệu tiếng Việt, trong bài báo này, chúng tôi thu thập các đoạn văn bản lấy từ các trang báo trực tuyến ở Việt Nam (Hình 6). Với đoạn văn bản cần tóm tắt là cả

bài báo và phần tóm tắt là đoạn mô tả phía dưới tiêu đề.

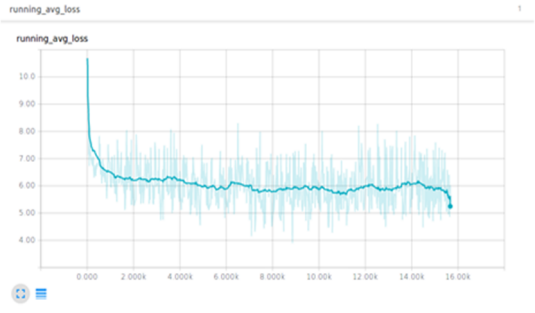
Hình 6: Thu thập dữ liệu từ các trang báo online

Kết quả thu được 39.287 bài báo online Việt Nam. Tập dữ liệu mẫu được định dạng chia thành 3 phần là article, abstract và publisher. Để sử dụng mô hình ta cần chuyển tập dữ liệu tiếng Việt về dạng từ đơn, từ ghép bằng gói công cụ vnTokenizer của Hong Phuong L. *et al*. (2008), loại bỏ các dấu câu, giữ lại dấu chấm để phân biệt các câu, chuyển số về chữ và sau đó chuyển văn bản về dạng nhị phân. Trong đó, tập dữ liệu huấn luyện gồm 31.429 article, tập đánh giá gồm 5893 article, tập test gồm 1964 article. Tập từ vựng gồm 159.772 từ vựng trong toàn bộ tập dữ liệu 28.193.515 từ tiếng Việt.

3.2 Huấn luyện mô hình

Chúng tôi sử dụng thư viện TensorFlow để xây dựng tập dữ liệu Word2vec và huấn luyện mô hình Sequence-to-Sequence with Attention cho bài toán tóm tắt văn bản tiếng Việt tự động. Các thông số cần cân chỉnh là số lượng nơ-ron của RNN encoder và decoder phù hợp với độ dài của dữ liệu đầu vào. Số lượng câu lấy từ arcticle là max_arcticle_sentences=5, độ dài tối đa của đoạn abstract là max_abstract_sentences=100. Chúng tôi sử dụng Gradient Descent Optimizer với learning rate giảm từ 0,15 đến 0,000015 để tối thiểu hàm lỗi và Beam Search với độ rộng K=5 để sinh ra câu.

Dưới đây là biểu diễn của hàm lỗi khi huấn luyện mô hình Sequence-to-Sequence (Hình 7).



Hình 7: Running average loss

3.3 Đánh giá độ chính xác

Phương pháp chính để xác định độ chính xác của mô hình tóm tắt văn bản là dựa vào ý nghĩa bên trong của đoạn văn. Ở đây, chúng tôi sử dụng phương pháp ROUGE (Svore *et al.,* 2007) để đánh giá độ chính xác của mô hình. Số điểm ROUGE-n của một bản tóm tắt được xác định như sau:

$$ROUGE - n = \frac{\sum_{C \in RSS} \sum_{gram_n \in C} Count_{match}(gram_n)}{\sum_{C \in RSS} \sum_{gram_n \in C} Count(gram_n)}$$

với $Count_{match}(gram_n)$ là số lượng n-grams lớn nhất có trong kết quả tóm tắt và bản tóm tắt tham khảo, $Count(gram_n)$ là số lượng n-grams có trong bản tóm tắt tham khảo. Có nhiều ROUGE score khác nhau như ROUGE-L là một Longest Common Subsequece Measure.

Bảng 1: Đánh giá độ chính xác trên tập gồm 1964 article tiếng Việt

	ROUGE-1	ROUGE-2	ROUGE-L
Precision	0,372	0,313	0,367
Recall	0,446	0,317	0,417
F-score	0,39	0,314	0,367

4 KẾT LUẬN VÀ HƯỚNG PHÁT TRIỂN

Sử dụng mô hình Sequence-to-Sequence with Attention mở ra một hướng đi mới cho bài toán tóm tắt văn bản tự động. Hơn thế nữa, học sâu kết hợp với xử lý ngôn ngữ tự nhiên đã áp dụng thành công và góp phần cải thiện độ chính xác trên ngôn ngữ tiếng Việt.

Bên cạnh đó, để cải thiện mô hình chúng ta cần xây dựng một tập đầu vào Word2vec có độ chính xác cao hơn nữa, thể hiện rõ mối liên hệ giữa các từ hơn nữa. Chính vì thế, việc chuẩn bị tập dữ liệu lớn và phong phú về mặt từ vựng là vô cùng cần thiết cho một mô hình tóm tắt văn bản tự động tiếng Việt.

TÀI LIỆU THAM KHẢO

Ayako Mikami, 2016. Long Short-Term Memory - Recurrent Neural Network Architectures for Generating Music and Japanese Lyrics. Honors Thesis Advised by Professor Sergio Alvarez, Computer Science Department, Boston College.

Colah, 2015. Understanding LSTM Networks, accessed on 10 February, 2017. Available from http://colah.github.io/posts/2015-08-Understanding-LSTMs/.

Dzmitry Bahdanau, Kyunghyun Cho, Yoshua Bengio, 2016. Neural Machine Translation by Jointly Learning to Align and Translate - In arXiv:1409.0473v7 [cs.CL].

Hong Phuong L., Thi Minh Huyen N., Roussanaly A., Vinh H.T. (2008) A Hybrid Approach to

Word Segmentation of Vietnamese Texts. In: Martín-Vide C., Otto F., Fernau H. (eds) Language and Automata Theory and Applications. LATA 2008. Lecture Notes in Computer Science, vol 5196. Springer, Berlin, Heidelberg.

Ilya Sutskever, Oriol Vinyals, Quoc V. Le, 2014. Sequence to Sequence Learning with Neural Networks – In arXiv:1409.3215v3 [cs.CL].

Josef Steinberger, Karel Je˘zek. Evaluation measures for text summarization. Computing and Informatics, Vol. 28, 2009, 1001–1026, V 2009-Mar-2.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, Yoshua Bengio, 2015. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation – In arXiv:1406.1078 [cs.CL].

Nguyễn Thị Ngọc Tú, Nguyễn Thị Thu Hà, Lê Thanh Hương, Hồ Ngọc Vinh, Đào Thanh Tĩnh, Nguyễn Ngọc Cương. "Ứng dụng mô hình đồ thị trong tóm tắt đa văn bản tiếng Việt". Kỷ yếu Hội nghị Quốc gia lần thứ VIII về Nghiên cứu cơ bản và ứng dụng Công nghệ thông tin (FAIR); Hà Nội, ngày 9-10/7/2015.

Ramesh Nallapati, Bowen Zhou, Cicero Nogueira dos santos, Caglar Gulcehre, Bing Xiang, 2016. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond – In arXiv:1602.06023 [cs.CL].

Sam Wiseman and Alexander M. Rush, 2016. Sequence-to-Sequence Learning as Beam-Search Optimization. School of Engineering and Applied Sciences Harvard University Cambridge, MA, USA-In arXiv:1606.02960v2 [cs.CL].

Tomas Mikolov, Kai Chen, Greg Corrado, Jeffrey Dean, 2013. Efficient Estimation of Word Representations in Vector Space – In arXiv:1301.3781v3 [cs.CL].

Trương Quốc Định, Nguyễn Quang Dũng. "Một giải pháp tóm tắt văn bản tiếng Việt tự động". Hội thảo quốc gia lần thứ XV: một số vấn đề chọn lọc của Công nghệ thông tin và Truyền thông Hà Nội 03-04/12/2012.

Xin Rong, 2016. word2vec Parameter Learning Explained - In arXiv:1411.2738v4 [cs.CL].