A LSTM based Neural Network Literature Generation Approach: Replicating Kazi Nazrul Islam

No Author Given

No Institute Given

Abstract. The rich history of Bengali literature and its impact on every aspect of Bengali culture, history, and motivation towards a just and fair society as well as in defining the collective identity is immeasurable. Even with such inseparable bonding between the literature and Bengali identity, it is unfortunate how few digitized versions of works exist from the great literary figures. While the advancement in the NLP in any given language is driven by the availability of data in easily usable formats, the lack of such works in the Bengali language is striking and the main motivating factor behind this work. We set out to create a comprehensive collection of works by Kazi Nazrul Islam, one of the most prominent figures in Bengali literature, and usher in a movement of applying the state-of-the-art NLP tools and techniques for various analytical and generative tasks to gain a better insight of such revolutionary works by Nazrul. To the best of our knowledge, this is the first of its kind dataset comprising the works of Nazrul. We applied the poem and song datasets from the collection to a character-level RNN model consisting of stacked-LSTM layers to generate texts for respective genres. We hope a plethora of generative works will take place using this dataset and the current model as the baseline to improve the quality of the generated text.

Keywords: RNN, LSTM, NLP, text generation, literature, Kazi Nazrul Islam

1 Introduction

The new era of Natural Language Processing (NLP) has emerged in many fields like computer vision, pattern recognition, network traffic analysis, text generation, text summarization, and so on. Since NLP research is advancing at a breakneck pace. Because NLP tasks are difficult to solve, deep learning is essential to bring us up to speed on many difficult challenges in this field. However, there is a substantial gap in applying Artificial Intelligence (AI) advances to better understand human literature and conduct literary analysis in terms of themes, concepts, societal values, and norms at the time of writing. [1] As a result, modern machine learning systems are unable to use our priceless literary works, especially literature from other languages, like Bangla literature. Due to a lack of research, and language boundaries, modern machine learning systems are unable to use our priceless literary works.

The rest of the paper is organized as follows: Section 2 describes the literature review of the topic, and Section 3 explains the proposed model with dataset compilation, analysis process, and the methodology. Section 4 analyzes the result obtained from the proposed model and the paper concludes with a conclusion in Section 5.

2 Literature Review

In [2] a three-stage multi-modal Chinese poetry generation approach has been proposed that generated the first line, the title, and the other lines of the poem. They have used the hierarchy-attention seq2seq model that collects characters, phrases, and sentences to generate quatrains. From a dataset built on image to the theme, a phrase feature was added in the HieAS2S model which was shown to outperform strong baseline and variants. In [3], they ensure coherence and semantic consistency relative to human intent. In this poetry generation method in two steps, subtopics of the poem and sentence generation are executed by the RNN encoder-decoder model. They [4] conducted a poetry generation system using CNN in classifying image objects, a module in finding related words or rhyme pairs, and LSTM trained on song lyrics data. But this work could not portray tasteful outcomes according to everyone's perseverance and created a weak correlation between the evaluation and the stanzas containing non-zero predictions. In [5], innovatively poems have been generated using the simple memory-augmented neural model. In another approach [6], the researchers have divided the poetry generation into sub-tasks by multi adversarial training. The model provided cross-model relevance and language standards of poetry. RNN has been applied with policy gradient optimization. They have implemented an image-poem dataset (MultiM-Poem) and a huge poem corpus (UniM-Poem). On a scale of 10 average scores of relevance to images for three types of human-written poems have occurred at 7.22. [7] Adapted sequence-to-sequence learning technique and built word-to-line, line-to-line, and context-to line blocks based on RNN Encoder-Decoder in generating quatrain. In this method [8] [9] a caption is searched from the given image as a dataset and these procedures lack accuracy in describing the outcome. Works like [10] explain solutions to image captioning procedural lacking through template filling and CNN, RNN integration [11] [12] [13] [14] generating human-readable sentences. [15] and [16] generate Chinese poems and couplets using RNN models attaining attention mechanism and polishing scheme. [17] Implemented Conditional Variational Auto Encoder (CVAE) [18] [19] [20] for Chinese poetry generation.

3 Proposed Model

3.1 Dataset Compilation

Even though Bengali literature has a rich history of thousands of years filled with great figures in every branch of literature, the lack of digitized versions of these works creates an obstacle in conducting research in the field of AI. Without the datasets produced from the literary works of great poets, novelists, songwriters of Bengali culture, we cannot dive deep into the gold mines of Bengali literature and conduct literary analysis or take advantage of the recent progress in the field of AI, especially in Natural Language Processing to extract insights from these literary works. In order to reduce this gap, we set out to create a comprehensive dataset from the works of Kazi Nazrul Islam. To the best of our knowledge, it is the first of a kind dataset that presents the works of Nazrul in a format that is ready for various analysis and NLP tasks. To create an authentic dataset we relied upon the excellent digitized works of Nazrul from the Department of Information Technology and Electronics, Government of West Bengal, India [21]. The publisher of the entire Nazrul collection divided his works into 8 following genres,

1. Novel, 2. Drama, 3. Poem, 4. Letters, 5. Story, 6. Essay, 7. Song, and 8. Translation

Each of these genres contained collections of Nazrul's works. In order to retrieve content from the website, we created a crawler that crawled the starting URLs of each genre and then extracted individual starting links for each collection in that genre. After that, an extractor was used to extract content from each page of a given collection. At the first stage of the crawler, raw contents were stored in txt format in a hierarchical way where all collections of any genre were stored inside that genre. Later during the data preparation stage, each of these text files from all genres was loaded onto memory and various CSV and txt files were created to facilitate different analytical and NLP tasks. Figure 1 shows the crawling process of the dataset,

Before we could use these raw text files, we needed to run some formatting and preprocessing steps to transform the raw text from files into more useful and easy to use formats. The steps are described below:

1. For each content text file,
   (a) Create an entry with name, collection, genre, and preprocessed content in csv format
   (b) Add each entry into genre collection csv, e.g. "poem.csv" and also into a global all csv file called "all.csv"
   (c) Append preprocessed content into genre aggregated txt file, e.g "poem.txt"

Two different versions of the dataset were created because csv formats are useful for exploratory analysis, whereas txt formats are more useful for deep learning tasks. To publish the dataset for use in various NLP tasks, we chose kaggle as the platform [22].

3.2 Dataset Analysis:

We have analyzed our dataset before testing in order to check the anomalies and correctness of the dataset. Initially, we used the Exploratory Data Analysis

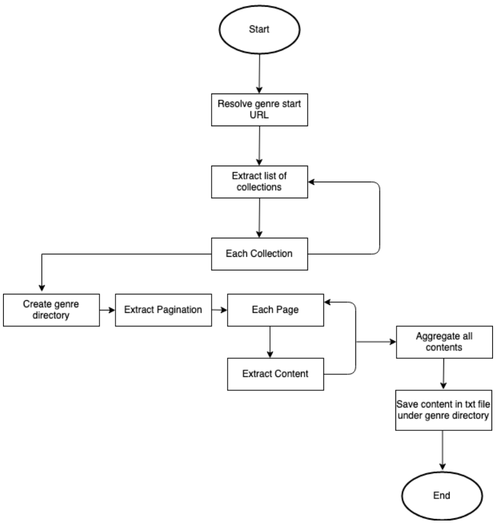


Fig. 1: Crawling Process Diagram

(EDA) techniques and made the dataset usable for our model. After that, we analyzed the dataset in order to gain knowledge about how Kazi Nazrul Islam wrote his literature. To find out this, we have used a dataset combining his poem, novel, drama, and all the other types of literature. After that, we ran an algorithm to find out the types of words he used in his literature and found out a group of word frequencies that has been mostly used. Figure 2 shows the word frequecy histogram.

3.3 Methodology:

In order to generate text in the style of Nazrul, we trained a character-level RNN model containing stack-LSTM layers for 250 epochs. We also trained a single LSTM layer model with the same configurations to compare the result with the stack-LSTM model. Details of this comparison is explained in the Results section of this paper. For training, we used both the aggregated poem and song content from the prepared dataset separately. Various components of the algorithm are described below:

1. Analysis and Preprocessing : Before training the model with the selected RNN model, we ran a preliminary analysis step on the dataset. First, we calculated the total number of characters in the dataset, e.g the poem dataset contained 777492 characters in total and 116 unique characters.
   Before feeding the text into the model for training, we converted the dataset from sequence of characters to sequence of numbers. At this step, we also created two mappings - a character-to-index mapping where each unique character was mapped to its index and an index-to-character mapping where

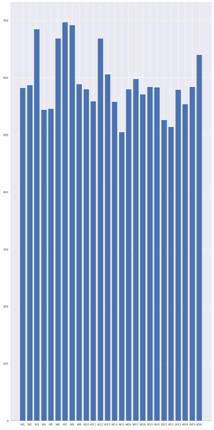
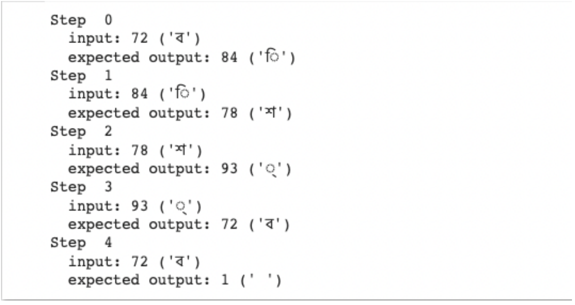


Fig. 2: Words frequency of Nazrul's literature

the predicted index during the generation step was mapped to an actual character.

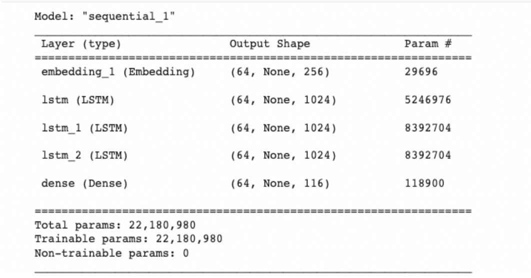
2. Creating Training Batch : In order to create the training batch with length of 100 characters for our model, we first extracted sequence of 101 indexes each from the dataset. This allowed us to generate an input-output pair from each sequence of indexes by removing the trailing index for input sequence and first index for output sequence.
   At each step, the model tries to predict the index of the next character given the index of the previous character. Figure 3 shows this process,



```
Step  0
  input: 72 ('র')
  expected output: 84 ('ি')
Step  1
  input: 84 ('ি')
  expected output: 78 ('শ')
Step  2
  input: 78 ('শ')
  expected output: 93 ('্')
Step  3
  input: 93 ('্')
  expected output: 72 ('র')
Step  4
  input: 72 ('র')
  expected output: 1 (' ')
```

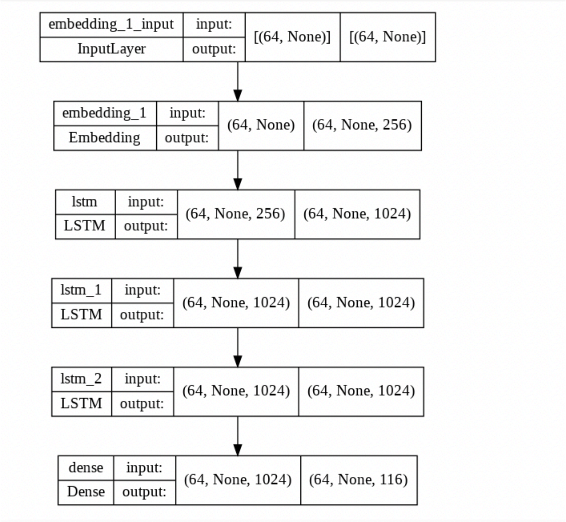
Fig. 3: Prediction of output character given input

3. Building and training model : We structured our learning model by adding an embedding layer first as the input layer. Then 3 LSTM layers were stacked and finally, a dense layer was added as the output layer. Figure 4 shows the model summary detailing various parameter values and Figure 5 shows the model structure.



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 embedding_1 (Embedding)     (64, None, 256)           29696

 lstm (LSTM)                 (64, None, 1024)          5246976

 lstm_1 (LSTM)               (64, None, 1024)          8392704

 lstm_2 (LSTM)               (64, None, 1024)          8392704

 dense (Dense)               (64, None, 116)           118900

=================================================================
Total params: 22,180,980
Trainable params: 22,180,980
Non-trainable params: 0
_________________________________________________________________
```

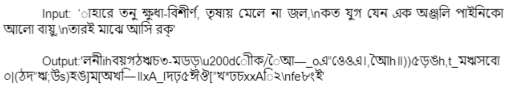
Fig. 4: Model parameter summary



embedding_1_input InputLayer	input: output:	[(64, None)]	[(64, None)]
embedding_1 Embedding	input: output:	(64, None)	(64, None, 256)
lstm LSTM	input: output:	(64, None, 256)	(64, None, 1024)
lstm_1 LSTM	input: output:	(64, None, 1024)	(64, None, 1024)
lstm_2 LSTM	input: output:	(64, None, 1024)	(64, None, 1024)
dense Dense	input: output:	(64, None, 1024)	(64, None, 116)

Fig. 5: Model structure

Following is an excerpt of the generated text using the pre-trained model parameters which shows that before training the model is simply predicting the index of the next character at random since it does not know anything about the dataset yet.



Input: 'াহারে তনু ক্ষুধা-বিশীর্ণ, তৃষায় মেলে না জল,\nকত যুগ যেন এক অঞ্জলি পাইনিকো আলো বায়ু,\nতারই মাঝে আসি রক্'

Output:'লনীlhবয়গঠঋচ৩-মডড়\u200dৗীক/ৈআ—_oএ"ওেওএ।,ৌআh॥))৫ড়ঙh,t_মঋসবো ০|(ঠদ"ঋ;উs)হঙ]ম[অখ—॥xA_।দঢ়৫ঈঔ["খ*ঢচxxAি২\nfe৮ংই'

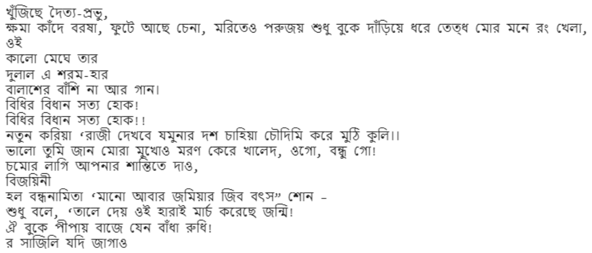
Fig. 6: Input Output

In Figure 6 During training the problem of learning became similar to a classification problem where given the input character, the model tries to predict the class of the next character. It produces a log-likelihood for all classes of vocabulary size which is the output dimension. For the loss function, we used “Categorical Cross Entropy” and “Adam Optimizer” to update the weights during back-propagation.

4. Generating content : After training the model for 250 epochs, we sampled a sequence of characters using a starting text and temperature value which controlled how predictable or surprising the generated next character would be.

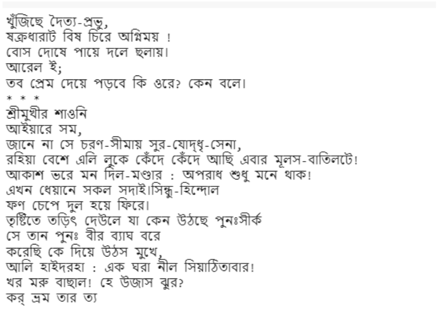
4 Result Analysis

As previously mentioned, we executed the same training procedure with 2 different data-sets from different genres, namely song, and poem.



```
খুঁজিছে দৈত্য-প্রভু,
ক্ষমা কাঁদে বরষা, ফুটে আছে চেনা, মরিতেও পরুজয় শুধু বুকে দাঁড়িয়ে ধরে তেত্ধ মোর মনে রং খেলা,
ওই
কালো মেঘে তার
দুলাল এ শরম-হার
বালাশের বাঁশি না আর গান।
বিধির বিধান সত্য হোক!
বিধির বিধান সত্য হোক!!
নতুন করিয়া ‘রাজী দেখবে যমুনার দশ চাহিয়া চৌদিমি করে মুঠি কুলি।।
ভালো তুমি জান মোরা মুখোও মরণ কেরে খালেদ, ওগো, বন্ধু গো!
চমোর লাগি আপনার শান্তিতে দাও,
বিজয়িনী
হল বন্ধনামিতা ‘মানো আবার জমিয়ার জিব বৎস” শোন -
শুধু বলে, ‘তালে দেয় ওই হারাই মার্চ করেছে জন্মি!
ঐ বুকে পীপায় বাজে যেন বাঁধা রুধি!
র সাজিলি যদি জাগাও
```

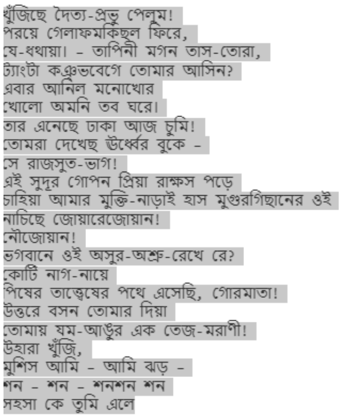
Fig. 7: Generated poem from stacked-LSTM with starting text as ”খুঁজিছে দৈত্য-প্রভু” and temperature=1.0



```
খুঁজিছে দৈত্য-প্রভু,
ষক্রধারাট বিষ চিরে অগ্নিময় !
বোস দোষে পায়ে দলে ছলায়।
আরেল ই;
তব প্রেম দেয়ে পড়বে কি ওরে? কেন বলে।
* * *
শ্রীমুখীর শাওনি
আইয়ারে সম,
জানে না সে চরণ-সীমায় সুর-যোদ্ধৃ-সেনা,
রহিয়া বেশে এলি লুকে কেঁদে কেঁদে আছি এবার মূলস-বাতিলটে!
আকাশ ভরে মন দিল-মণ্ডার : অপরাধ শুধু মনে থাক!
এখন ধেয়ানে সকল সদাই।সিন্ধু-হিন্দোল
ফণ চেপে দুল হয়ে ফিরে।
তৃষ্টিতে তড়িৎ দেউলে যা কেন উঠছে পুনঃসীর্ক
সে তান পুনঃ বীর ব্যাঘ বরে
করেছি কে দিয়ে উঠস মুখে,
আলি হাইদরহা : এক ঘরা নীল সিয়াঠিতাবার!
খর মরু বাছাল! হে উজাস ঝুর?
কর্ ভ্রম তার ত্য
```

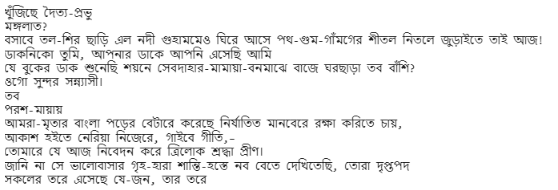
Fig. 8: Generated poem from stacked-LSTM with starting text as ”খুঁজিছে দৈত্য-প্রভু” and temperature=1.0

First, we present the output of the model after it was trained on the poem dataset for single-layer LSTM (see Figure 7) and 3-layer stacked-LSTM (see



খুঁজিছে দৈত্য-প্রভু পেলুম!
পরয়ে গেলাফমকিছুল ফিরে,
যে-ধথায়া। - তাপিনী মগন তাস-তোরা,
ট্যাংটা কঞ্জভবেগে তোমার আসিন?
এবার আনিল মনোখোর
খোলো অমনি তব ঘরে।
তার এনেছে ঢাকা আজ চুমি!
তোমরা দেখেছ ঊর্ধ্বের বুকে -
সে রাজসুত-ভাগ!
এই সুদূর গোপন প্রিয়া রাক্ষস পড়ে
চাহিয়া আমার মুক্তি-নাড়াই হাস মুগুরগিছানের ওই
নাচিছে জোয়ারেজোয়ান!
নৌজোয়ান!
ভগবানে ওই অসুর-অশ্রু-রেখে রে?
কোটি নাগ-নায়ে
পিষের তাণ্ডেষের পথে এসেছি, গোরমাতা!
উত্তরে বসন তোমার দিয়া
তোমায় যম-আঙুর এক তেজ-মরাণী!
উহারা খুঁজি,
মুশিস আমি - আমি ঝড় -
শন - শন - শনশন শন
সহসা কে তুমি এলে

Fig. 9: Generated poem from 3 layer stacked-LSTM with starting text as ”খুঁজিছে দৈত্য-প্রভু” and temperature=1.5



খুঁজিছে দৈত্য-প্রভু
মঙ্গলাত?
বসাবে তল-শির ছাড়ি এল নদী গুহামমেও ঘিরে আসে পথ-গুম-গাঁমগের শীতল নিতলে জুড়াইতে তাই আজ!
ডাকনিকো তুমি, আপনার ডাকে আপনি এসেছি আমি
যে বুকের ডাক শুনেছি শয়নে সেবদাহার-মামায়া-বনমাঝে বাজে ঘরছাড়া তব বাঁশি?
ওগো সুন্দর সন্ন্যাসী।
তব
পরশ-মায়ায়
আমরা-মৃতার বাংলা পড়ের বেটারে করেছে নির্যাতিত মানবেরে রক্ষা করিতে চায়,
আকাশ হইতে নেরিয়া নিজেরে, গাইবে গীতি,-
তোমারে যে আজ নিবেদন করে ত্রিলোক শ্রদ্ধা প্রীণ।
জানি না সে ভালোবাসার গৃহ-হারা শান্তি-হস্তে নব বেতে দেখিতেছি, তোরা দৃপ্তপদ
সকলের তরে এসেছে যে-জন, তার তরে

Fig. 10: Generated poem from 3 layer stacked-LSTM starting text as ”খুঁজিছে দৈত্য-প্রভু” temperature=1.0

Figure 8) with temperature value set to 1 and starting text as ”খুঁজিছে দৈত্য-প্রভু” We can observe that the stacked LSTM model not only learned how to form correct words but the generated text seems more coherent than the single-layer LSTM model. However, when we provided the stacked-LSTM model with the same starting sequence of characters but a temperature value of 1.5, it produced more surprising text (see Figure 8). Next, we present the output of the stacked-LSTM model and the output of the 3 layer stacked-LSTM model after it was trained on a poem dataset and song dataset with the same starting sequence but different temperature values see Figure 9 and 10. We can notice the 3-fold increase in elapsed time during training from single-layer LSTM to stacked-LSTM with epoch=250 using trained models for different genres. Currently, we are producing texts of 500 to 1000 characters of two genres on which the model was trained upon. We plan to experiment with much longer texts as found in stories and essays.

5 Conclusion

The paper mainly focuses on developing an advanced NLP system that will imitate the literature pattern of the national poet of Bangladesh Kazi Nazrul Islam. The paper's main focus is on creating a model which is applicable to Bangla literature. In order to achieve this, we initially created novel a dataset on Kazi Nazrul Islam. The model used an LSTM-based character level RNN system which will generate literature work of Kazi Nazrul Islam. The main challenge of this paper was to create the database and develop the NLP system which will work with the Bangla language and literature. Our initial work has produced the expected results considering we have worked on word level. To conclude, the proposed model will focus on implementing sentence levels in the future in order to gain more accuracy.

Bibliography

[1] Asadullah Al Galib. Rabindranet, creating literary works in the style of rabindranath tagore. arXiv preprint arXiv:2202.00481, 2022.

[2] Dayiheng Liu, Quan Guo, Wubo Li, and Jiancheng Lv. A multi-modal chinese poetry generation model. In 2018 International Joint Conference on Neural Networks (IJCNN), pages 1--8. IEEE, 2018.

[3] Zhe Wang, Wei He, Hua Wu, Haiyang Wu, Wei Li, Haifeng Wang, and Enhong Chen. Chinese poetry generation with planning based neural network. arXiv preprint arXiv:1610.09889, 2016.

[4] Malte Loller-Andersen and Bjo□rn Gamba□ck. Deep learning-based poetry generation given visual input. In ICCC, pages 240--247, 2018.

[5] Jiyuan Zhang, Yang Feng, Dong Wang, Yang Wang, Andrew Abel, Shiyue Zhang, and Andi Zhang. Flexible and creative chinese poetry generation using neural memory. arXiv preprint arXiv:1705.03773, 2017.

[6] Bei Liu, Jianlong Fu, Makoto P Kato, and Masatoshi Yoshikawa. Beyond narrative description: Generating poetry from images by multi-adversarial training. In Proceedings of the 26th ACM international conference on Multimedia, pages 783--791, 2018.

[7] Xiaoyuan Yi, Ruoyu Li, and Maosong Sun. Generating chinese classical poems with rnn encoder-decoder. In Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data, pages 211--223. Springer, 2017.

[8] Ali Farhadi, Mohsen Hejrati, Mohammad Amin Sadeghi, Peter Young, Cyrus Rashtchian, Julia Hockenmaier, and David Forsyth. Every picture tells a story: Generating sentences from images. In European conference on computer vision, pages 15--29. Springer, 2010.

[9] Andrej Karpathy, Armand Joulin, and Li F Fei-Fei. Deep fragment embeddings for bidirectional image sentence mapping. Advances in neural information processing systems, 27, 2014.

[10] Girish Kulkarni, Visruth Premraj, Vicente Ordonez, Sagnik Dhar, Siming Li, Yejin Choi, Alexander C Berg, and Tamara L Berg. Babytalk: Understanding and generating simple image descriptions. IEEE transactions on pattern analysis and machine intelligence, 35(12):2891--2903, 2013.

[11] Xinlei Chen and C Lawrence Zitnick. Mind's eye: A recurrent visual representation for image caption generation. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2422--2431, 2015.

[12] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 3156--3164, 2015.

[13] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. Boosting image captioning with attributes. In Proceedings of the IEEE international conference on computer vision, pages 4894--4902, 2017.

[14] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4651--4659, 2016.

[15] Rui Yan. i, poet: Automatic poetry composition through recurrent neural networks with iterative polishing schema. In IJCAI, volume 2238, page 2244, 2016.

[16] Rui Yan, Cheng-Te Li, Xiaohua Hu, and Ming Zhang. Chinese couplet generation with neural network structures. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 2347--2357, 2016.

[17] Xiaopeng Yang, Xiaowen Lin, Shunda Suo, and Ming Li. Generating thematic chinese poetry using conditional variational autoencoders with hybrid decoders. arXiv preprint arXiv:1711.07632, 2017.

[18] Otto Fabius and Joost R Van Amersfoort. Variational recurrent auto-encoders. arXiv preprint arXiv:1412.6581, 2014.

[19] Xinchen Yan, Jimei Yang, Kihyuk Sohn, and Honglak Lee. Attribute2image: Conditional image generation from visual attributes. In European conference on computer vision, pages 776--791. Springer, 2016.

[20] Samuel R Bowman, Luke Vilnis, Oriol Vinyals, Andrew M Dai, Rafal Jozefowicz, and Samy Bengio. Generating sentences from a continuous space. arXiv preprint arXiv:1511.06349, 2015.

[21] Govt of West Bengal Department of Information Technology Electronics. Nazrul rachanabali. https://nazrul-rachanabali.nltr.org/index.php, 2022.

[22] A. A. Galib. Complete works of kazi nazrul islam | kaggle. https://www.kaggle.com/datasets/aagalib/complete-works-of-kazi-nazrul-islam, 2022.