Team 7 Silver Team IDS 705 Final Project Comparison of Pre-Inject Architecture and Merge Architecture for Image Captioning

Dean Huang
th284@duke.edu
Duke University
Durham, North Carolina, USA

Zihao Lin
zl293@duke.edu
Duke University
Durham, North Carolina, USA

Jeremy Zeng
xz301@duke.edu
Duke University
Durham, North Carolina, USA

Yuwei Zhang
yz667@duke.edu
Duke University
Durham, North Carolina, USA

Abstract

Blind and visually impaired people (BVIP) have consistently expressed their frustration on the lack of accessibility to visual content via social media. The goal of this project is to build high-performance encoder-decoder image captioning models using Pre-inject Architecture and Merge Architecture to assist BVIP in comprehending the message and social context of the images. The two image encoders we decided to implement were VGG and ResNet, and the language model we decided to implement was LSTM. Our experiment results showed that Pre-Inject Architecture outperformed the Merge Architecture by a large margin. In addition, the total training duration for Pre-inject Architecture was approximately 13 times higher than the Merge Architecture.

1 Introduction

According to the statistic from Our World In Data, the percentage of adults who use social media in the US increased from 5% in 2005 to 79% in 2019[13]. One of the byproducts of the increasing popularity of social media is the bountiful and rich information, news, stories, and experiences shared by social media users. Blind and visually impaired people (BVIP) have consistently expressed their frustration on the lack of accessibility to visual content via social media [8]. Understanding the content of images in social contexts enables BVIPs to acquire latest news coverage, participate in meaningful social conversations, and enjoy entertainment or humor for mental stress relief [7]. Automatic image captioning tool is one promising solution to assist BVIP in comprehending the message and social context of the images without relying on the assistance of sighted person or human-authored alt-text. In this project, we will build high-performance encoder-decoder image captioning models using Pre-inject Architecture and Merge Architecture, and conduct comprehensive comparison of these two methods based on the results and findings of our models.

2 Background

Image captioning is the task of generating a natural language description of an image. The most common neural network architecture for image captioning is encoder-decoder architecture. The encoder helps to extract information from embedded captions and features from images, while the decoder helps to generate sequential data (sentences) based on extracted information of images and captions. In the past six years, most of the existing encoder-decoder image captioning techniques can be categorized into one of these four architectures: Init-inject Architecture, Pre-inject Architecture, Par-inject Architecture, and Merge Architecture [17]. Since we only implemented Pre-Inject Architecture and Merge Architecture, we will focus on articulating high level past works done by researchers using these two architectures:

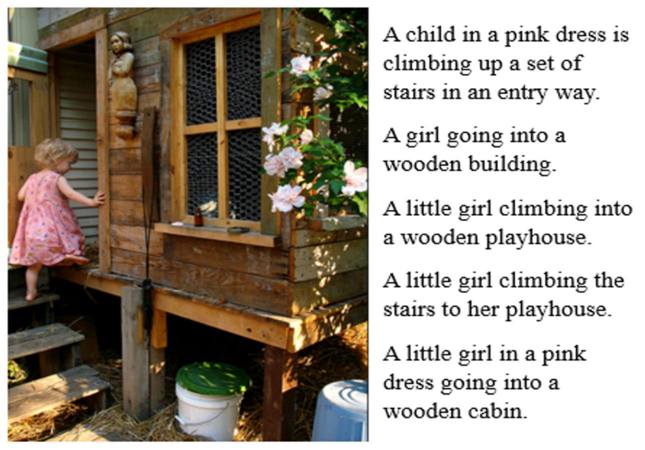
- `Pre-inject architecture`: Pre-inject architecture treats image vectors as the first input word of the RNN decoder. Vinyals et al. (2015) [19], Nina and Rodriguez, (2015) [12], and Rennie et al. (2016) [15] utilized this architecture for their image captioning models, and were able to achieve high accuracy in their models. Instead of injecting only the image vectors, Yao et al. (2016) passed the image vector as the first (or second) word and image attributes as the second (or first) word [22].
- `Merge architecture`: Merge architecture combines image features with linguistic features outside of the RNN decoder. Mao et al. (2014), (2015a,b) utilized this architecture for their image captioning models, and were able to produce fast and accurate results [10] [9] [11].

In addition, some researchers applied attention mechanisms in their architectures to optimize their image captioning models [17]. Attention mechanism allows modeling of dependencies without considering their distances to the input or output sequences. Xu et al. (2015) [21] and Lu et al.

(2016) [6] utilized this mechanism to generate word probability based on different parts of image.

3 Data

The Flickr8k Dataset comes from a data repository at Kaggle entitled "Flickr8k Dataset for image captioning" [kaggle.com]. The dataset constitutes 8,000 images with each image associated with five different captions that describe the entities and events depicted by the picture. As illustrated by the example from Figure 1, the five captions associated with the image focus on different aspects of the scene with different linguistic constructions. To perform model training and evaluation, the dataset was split into a training set, a validation set, and a testing set with proportions equal to 75%, 12.5%, and 12.5% respectively.



A child in a pink dress is climbing up a set of stairs in an entry way.

A girl going into a wooden building.

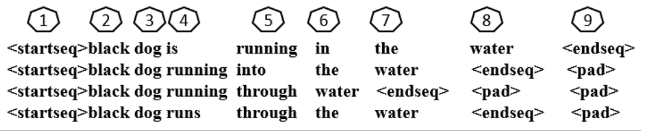
A little girl climbing into a wooden playhouse.

A little girl climbing the stairs to her playhouse.

A little girl in a pink dress going into a wooden cabin.

Figure 1. Example of an image from the Flickr8K dataset.

Since we utilized different pretrained encoders like VGG, we had to reshape the images into the preferred size before passing the images to the encoders. To reduce the size of the vocabulary and elevate the efficiency of our models, we performed basic text pre-processing for our captions like removing punctuations, removing numbers, and converting to lower cases before feeding to the language model. We utilized <startseq> and <endseq> to indicate the starting point and ending point of the sentence. In addition, we added <pad>tokens to ensure the passing of consistent fixed size captions to the decoder (Figure 2).



1	2	3	4	5	6	7	8	9
<startseq>	black	dog	is	running	in	the	water	<endseq>
<startseq>	black	dog	running	into	the	water	<endseq>	<pad>
<startseq>	black	dog	running	through	water	<endseq>	<pad>	<pad>
<startseq>	black	dog	runs	through	the	water	<endseq>	<pad>

Figure 2. Example of padded captions.

One major drawback of our dataset was its small dataset size, which might result in overfitting of models. In addition, our dataset might not capture the general trends and features of the overall Flickr dataset well due to its small data size. Another major drawback of the dataset was the fact that the images and captions are not sampled from posts published by users in social media; therefore, the results and findings drawn from our model might not be applicable to social media posts due to the difference of contexts and applications.

4 Methods

4.1 CNN Image Encoder

The goal of CNN Image Encoder is to read the photograph input and encode the content into a fixed-length vector using an internal representation. The two image encoders we decided to implement were VGG and ResNet. VGG is known for its simplicity, using only 3*3 convolutional layers stacked on top of each other to increase depth. In addition, the architecture uses max pooling to reduce the volume size [16]. ResNet is a residual learning framework that utilizes shortcut connections to perform identity mapping [3]. ResNet is able to overcome the previous shortcomings of training deeper neural networks like optimization difficulty and high training/testing errors. Demonstrated from past studies, these two computer vision models were shown effective in extracting features from images [4].

4.2 RNN Language Model

The goal of RNN Language Model is to extract the word features and learn a dense feature embedding for each word, and forward the semantic temporal context to the recurrent layers. Language decoder then generates captions by decoding the multimodal space part maps generated by recurrent layers. We will be using LSTM for our RNN Language Model. LSTM has special units in addition to standard units to retain information in memory for long periods of time. The major advantage of utilizing LSTM is its ability to retain important information through internal gates that can regulate the flow of information. Almost all existing models utilized LSTM for the language model due to its consistent high performance compared to other language models [4].

4.3 Architectures

The two architectures we implemented for our projects were Merge Architecture (Figure 3) and Pre-inject Architecture (Figure 4). For Merge Architecture, the RNN is not exposed to the image vector at any point, and the image is only introduced to the language model after the prefix has been encoded by the RNN in its entirety. This type of architecture allows the separate training of images and captions. For Pre-inject Architecture, the first input to the RNN is the image vector encoded by CNN with the word vectors of the caption prefix coming later. According to the experiments done by Tanti, Pre-inject Architecture is a much more memory-hungry architecture that requires large RNN hidden state vectors in order to function well [17]. The number of parameters for Merge Architecture is between three and four times smaller than the number of parameters for Pre-Inject Architecture.

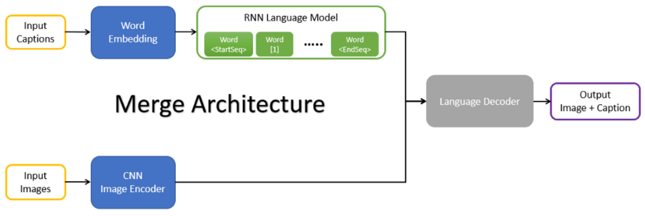


Figure 3. Merge Architecture

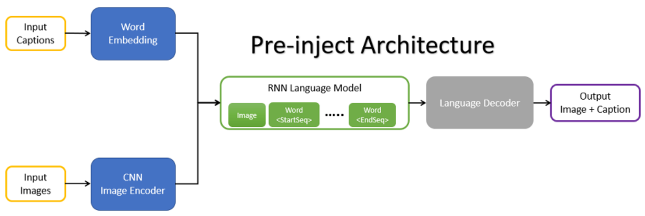


Figure 4. Pre-inject Architecture

4.4 Attention-based Image Captioning

For the language decoder of our Pre-inject Architecture, we added an attention-based mechanism. Attention-based method concentrates on the salient parts of the image to generate the corresponding words at the same time by using stochastic hard attention and deterministic soft attention to generate attention [4]. Traditional neural encoder-decoder based approach does not consider the spatial aspects of the image that is relevant to the parts of the image captions. Instead, it generates captions by looking at the image scene as a whole. In addition, we used beam search to transform the decoder's output into a score for each word in the vocabulary. Beam search expands all the possible sequences of sentences, and only keeps the k most likely phrases/sentences in each decode step, where k controls the number of beams or parallel searches [1].

4.5 Evaluation Metrics

The three performance metrics we used were BLEU, CIDEr, and ROUGE, which compare the closeness of the candidate text with the human-generated text known as the reference text. BLEU calculates the score by counting the number of same n-grams in the candidate text and the reference text then divides the counts by the number of n-grams in the candidate text [14]. CIDEr calculated the score based on the average cosine similarity between the candidate text and the reference text [18]. In addition, it takes both precision and recall into consideration. Thus, it will assign less weight to high frequency words in the captions since these words are less informative. ROUGE-L is a Longest Common Subsequence (LCS) based method that takes sentence level structure similarity and longest co-occurring in sequence n-grams into consideration when calculating the number of overlapping units between candidate texts and reference texts [5].

5 Results

5.1 Merge Architecture

We selected the final CNN Image Encoder for our architecture based on the evaluation metrics scores. Among the three pre-trained models we picked (ResNet50, ResNet152 and VGG16), ResNet50 had the highest score (Figure 5). The first hyperparameter we tuned was the activate function, which dictates the information that will be retained for each layer. We compared the performance of tanh, relu, and selu activate functions using 0.5 as the dropout rate, and found tanh generated the highest score (Table 1). The second hyperparameter we tuned was the dropout rate with the goal of preventing overfitting. We tuned the dropout rate in the range of 0.1 to 0.6. Since our dataset is very small, a high dropout rate will result in the loss of a large amount of information leading to less accurate caption generation. Thus, our experiment showed that dropout rate 0.2 had the best performance (Table 2 & Figure 6). Based on the results of the experiment, the optimal set of hyperparameters was tanh as our activation function and 0.2 as our dropout rate, with the BLEU-4, ROUGE, and CIDEr scores equal to 0.103, 0.462, and 0.294 respectively.

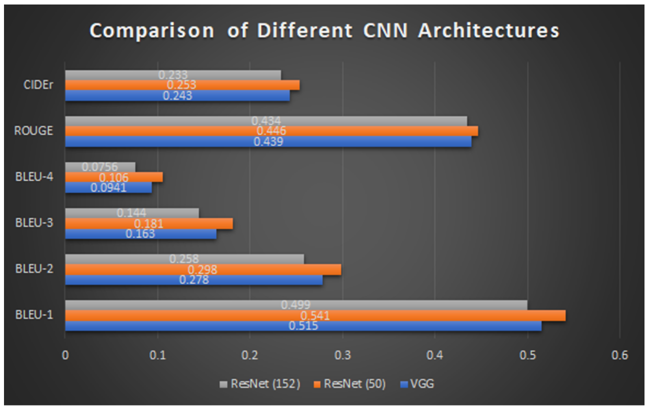


Figure 5. Comparison of Dropout Rates (with ResNet50 & Activation Function= tanh)

Table 1. Comparison of Activate Functions (with ResNet50 & dropout rate=0.5)

Activation Function	Dropout Rate	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	CIDEr
relu	0.5	0.492	0.264	0.183	0.085	0.442	0.243
selu	0.5	0.495	0.269	0.191	0.090	0.442	0.238
tanh	0.5	**0.535**	**0.292**	**0.199**	**0.091**	**0.454**	**0.258**

Table 2. Comparison of Dropout Rates (with ResNet50 & Activation Function= tanh)

Activation Function	Dropout Rate	BLEU-1	BLEU-2	BLEU-3	BLEU-4	ROUGE	CIDEr
tanh	0.6	0.526	0.278	0.188	0.087	0.445	0.249
tanh	0.5	0.535	0.292	0.199	0.091	0.454	0.258
tanh	0.4	0.516	0.278	0.194	0.091	0.449	0.279
tanh	0.3	**0.544**	0.302	0.212	0.103	0.461	0.275
tanh	0.2	0.540	**0.310**	**0.218**	**0.104**	**0.463**	**0.294**
tanh	0.1	0.528	0.290	0.201	0.096	0.456	0.267

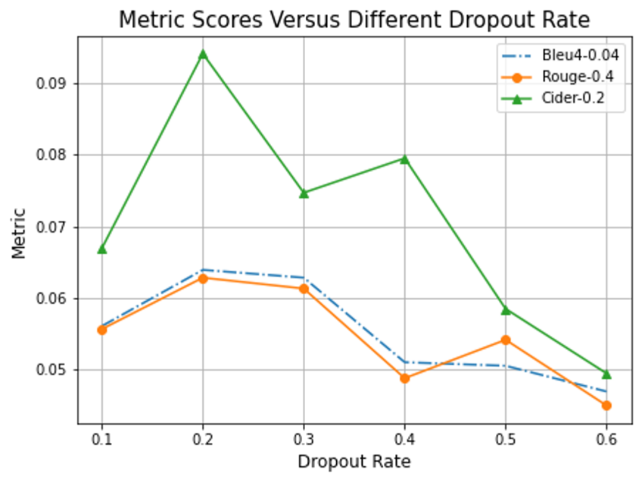


Figure 6. Linear Graph- Comparison of Dropout Rate (with ResNet50 & Activation Function= tanh)

5.2 Pre-inject Architecture

Three independent trials were performed to identify the best set of hyperparameters. For the first trial, we kept the weights of the Image Encoder the same as the pretrained ResNet101 since pretrained ResNet101 exhibited good test performance on ImageNet-5k and COCO datasets [20]. The entire training process was relatively fast because only the RNN Language Model weights were updated. However, the training result was not satisfactory with the best set of BLEU-4, CIDEr, and ROUGE scores equal to 0.228, 0.622 and 0.492 respectively (Table 3).

To improve the performance, we decided to train the weights of the image encoder for the second trial. For the image encoder, we used a smaller learning rate of 1e-4. We set the learning rate of the RNN Language Model relatively large as 1e-3 (the default learning rate of PyTorch Adam optimizer) to elevate the training speed. The second trial outperformed the first trial with BLEU-4, CIDEr, and ROUGE scores equal to 0.330, 0.622, and 0.492 respectively. Quote from Goodfellow: "If you have time to tune only one hyperparameter, tune the learning rate." (Ian Goodfellow, Yoshua Bengio, Aaron Courville. 2016) [2]. To further optimize our model, we reduced the decoder learning rate from 1e-3 to 1e-4 for the third trial in order to obtain better performance at the expense of computational efficiency. The third trial outperformed the second trial, achieving a satisfying BLEU-4, CIDEr, and ROUGE scores of 0.316, 0.892 and 0.550 respectively. To evaluate the training progress at different epochs, we generated captions on randomly selected images at different epochs, and compared the quality of captions by examining the captions ourselves. As the number of epochs increased, the generated captions grew from chaotic rough features to more organized and detailed descriptions that closely resembled human descriptions (Figure 7).

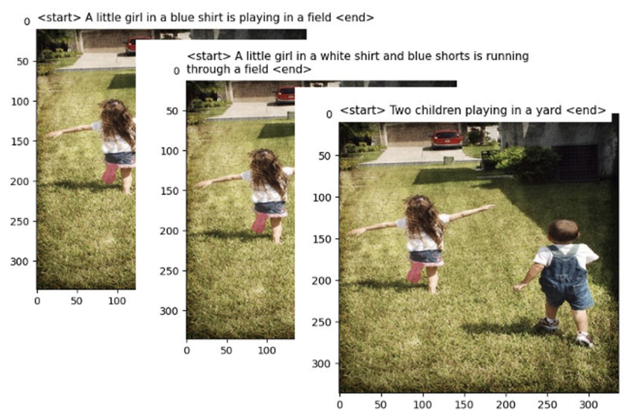

<start> A little girl in a blue shirt is playing in a field <end>
<start> A little girl in a white shirt and blue shorts is running through a field <end>
<start> Two children playing in a yard <end>

Figure 7. Captions Generated by Models of Different Epochs

Table 3. Model Performance of Different Trials

Trial	Encoder Learning Rate	Decoder Learning Rate	BLUE					CIDEr	ROUGE
			Epoch	BLUE-1	BLUE-2	BLUE-3	BLUE-4		
1	NA	4E-4	10	0.519	0.321	0.179	0.102	0.248	0.399
			15	0.650	0.470	0.330	0.228	0.622	0.492
2	1E-4	1E-3	10	0.666	0.492	0.352	0.250	0.660	0.500
			15	0.650	0.470	0.330	0.330	0.622	0.492
3	1E-4	4E-4	10	**0.721**	**0.560**	**0.422**	**0.316**	**0.892**	**0.550**
			11	0.700	0.536	0.399	0.294	0.848	0.543
			12	0.695	0.530	0.391	0.285	0.795	0.533

Table 4. Comparison of Two Architectures

Architecture	BLEU-4	CIDEr	ROUGE	Time
Merge	0.104	0.463	0.294	~ **12** hours
Pre-inject	**0.316**	**0.892**	**0.550**	~ 160 hours

6 Conclusion

According to our experiment results, the Pre-Inject Architecture outperformed the Merge Architecture by a large margin (Table 4). However, we did not have sufficient evidence to conclude Pre-Inject Architecture outperforms Merge Architecture as a whole due to other confounding factors like the usage of attention based mechanism in Pre-inject Architecture and the difference in CNN image encoders. The total training duration for Pre-inject Architecture was approximately 13 times higher than the Merge Architecture. One epoch training of Pre-inject models on a 2.3 GHz 8-Core Intel Core i9 CPU took about 8 hours, indicating 160h for only 20 epochs. The high-performance parallel computation of GPU was tried to solve the problem but failed due to Pytorch's lack of embedding support for the AMD graphics card in Big Sur.

For the application of models in social media for BVIP, we will recommend using Pre-Inject Architecture if the goal is to produce accurate and high quality close to human descriptions captions for BVIP like captions that accurately and concisely portray the different scenes in the same picture. However, if the goal is to only generate high level key words, hashtags or topics of the image, we will recommend using Merge Architecture, which is able to produce an accurate generalized description of the scene with a much lower computational cost.

7 Roles

Dean Huang:

- Individual Report Contribution:
  - Introduction
  - Data: Preprocessing and Exploration
  - Methods: CNN Image Encoder, RNN Language Model, Architectures, Attention-based Mechanism
  - Conclusion: Application of Models in Social Media
- Project Management Role:
  - Project Coordinator
  - Report Integrator Lead
- Executing a Machine Learning Experimental Condition:
  - Constructed Merge Architecture, and compared the performance of various image encoders (VGG16, ResNet50, ResNet152).
- Supporting your teammates on achieving project success:
  - Assist in laying out the weekly short term goals and tasks

Zihao Lin:

- Individual Report Contribution:
  - Background
  - Results: Merged Architecture
- Project Management Role:
  - Proposal and progress report lead
  - Report integrator lead
- Executing a Machine Learning Experimental Condition:
  - Constructed Merge Architecture (VGG16 & ResNet50), and finetunned the activate function and dropout rate of ResNet50.
- Supporting your teammates on achieving project success:
  - Provide our team with modeling thoughts, metrics and public reference resources.

Jeremy Zeng:

- Individual Report Contribution:
  - Abstract
  - Method: Evaluation metrics
- Project Management Role:
  - Video Lead
- Executing a Machine Learning Experimental Condition:
  - Hyperparameter tuning in merge architecture of ResNet50.

- Supporting your teammates on achieving project success:
  - Slides and video recording.

Yuwei Zhang:

- Individual Report Contribution:
  - Results: Pre-inject Architecture
  - Results: Conclusion: Model Performance analysis and limitation
- Project Management Role:
  - Github Lead: Model Implementation & Improvements
- Executing a Machine Learning Experimental Condition:
  - Implemented the Pre-inject model with attention and beam search optimization.
  - Conducted model trials and comparison
- Supporting your teammates on achieving project success:
  - Provided constructive advice to teammates regarding model implementation and Github organization.

8 Timeline of Activity

Date	Time	Content
2/28/2021	9 PM	Materials Learning & Reflections
3/01/2021	9 PM	Materials Learning & Reflections
3/14/2021	9 PM	Materials Learning & Reflections
3/21/2021	9 PM	Materials Learning & Reflections
4/28/2021	9 PM	Project Kickoff Meeting
4/04/2021	9 PM	Modeling & Computation
4/11/2021	9 PM	Modeling & Computation
4/18/2021	9 PM	Modeling & Computation
4/21/2021	9 PM	Project Presentation Discussion
4/23/2021	9 PM	Report Draft Review
4/24/2021	9 PM	Report Draft Review
4/25/2021	9 PM	Report Draft Review
4/27/2021	7 PM	Final Draft Review

References

[1] Jason Brownlee. [n.d.]. How to Develop a Deep Learning Photo Caption Generator from Scratch. machinelearningmastery.com/develop-a-deep-learning-caption-generation-model-in-python/ Accessed 22 Dec. 2020.

[2] Ian Goodfellow, Yoshua Bengio, Aaron Courville, and Yoshua Bengio. 2016. *Deep learning*. Vol. 1. MIT press Cambridge.

[3] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 770–778.

[4] MD Zakir Hossain, Ferdous Sohel, Mohd Fairuz Shiratuddin, and Hamid Laga. 2019. A comprehensive survey of deep learning for image captioning. *ACM Computing Surveys (CsUR)* 51, 6 (2019), 1–36.

[5] Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*. 74–81.

[6] Jiasen Lu, Caiming Xiong, Devi Parikh, and Richard Socher. 2017. Knowing when to look: Adaptive attention via a visual sentinel for image captioning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 375–383.

[7] Haley MacLeod, Cynthia L Bennett, Meredith Ringel Morris, and Edward Cutrell. 2017. Understanding blind people's experiences with computer-generated captions of social media images. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 5988–5999.

[8] Burak Makav and Volkan Kılıç. 2019. A new image captioning approach for visually impaired people. In *2019 11th International Conference on Electrical and Electronics Engineering (ELECO)*. IEEE, 945–949.

[9] Junhua Mao, Xu Wei, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan L Yuille. 2015. Learning like a child: Fast novel visual concept learning from sentence descriptions of images. In *Proceedings of the IEEE international conference on computer vision*. 2533–2541.

[10] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, Zhiheng Huang, and Alan Yuille. 2014. Deep captioning with multimodal recurrent neural networks (m-rnn). *arXiv preprint arXiv:1412.6632* (2014).

[11] Junhua Mao, Wei Xu, Yi Yang, Jiang Wang, and Alan L Yuille. 2014. Explain images with multimodal recurrent neural networks. *arXiv preprint arXiv:1410.1090* (2014).

[12] Oliver Nina and Andres Rodriguez. 2015. Simplified LSTM unit and search space probability exploration for image description. In *2015 10th International Conference on Information, Communications and Signal Processing (ICICS)*. IEEE, 1–5.

[13] Esteban Ortiz-Ospina. [n.d.]. Flickr8k Dataset for image captioning. https://ourworldindata.org/rise-of-social-media Accessed Sept. 18 2019.

[14] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*. 311–318.

[15] Steven J Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2017. Self-critical sequence training for image captioning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 7008–7024.

[16] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[17] Marc Tanti, Albert Gatt, and Kenneth P Camilleri. 2017. Where to put the image in an image caption generator. *arXiv preprint arXiv:1703.09137* (2017).

[18] Ramakrishna Vedantam, C Lawrence Zitnick, and Devi Parikh. 2015. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 4566–4575.

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

[20] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. 2017. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 1492–1500.

[21] Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*. PMLR, 2048–2057.

[22] Ting Yao, Yingwei Pan, Yehao Li, Zhaofan Qiu, and Tao Mei. 2017. Boosting image captioning with attributes. In *Proceedings of the IEEE International Conference on Computer Vision*. 4894–4902.