Image Captioning

Paper

Laiba Qureshi

January 24, 2025

Abstract

Image captioning is the art of automatically generating informative textual captions for images. The process combines visual feature extraction and language modelling to generate coherent and contextually relevant captions. This paper investigates a Convolutional Recurrent Neural Network (CRNN) architecture, where a convolutional neural network (CNN) encoder extracts visual features, and a recurrent neural network (RNN) decoder—specifically a Long Short-Term Memory (LSTM) network—generates descriptive captions. The proposed model's performance is evaluated through extensive testing and compared against existing implementations using the BLEU (Bilingual Evaluation Understudy) metric. The results demonstrate that the proposed model achieves an average BLEU-4 score of 0.198, which could be further improved in the future with additional training epochs. These findings advance vision-to-language systems for generating natural language descriptions from visual inputs.

1 Introduction

Image captioning is a challenging, yet significant task in the field of deep learning. Given an image, the purpose is to obtain a sentence that describes what the image consists of. Using a computer to automatically generate this natural language description for an image, which is defined as Image captioning, connects the fields of computer vision and natural language processing which demands not only a high level knowledge of an image's semantic contents, but also requires the ability to articulate the information in a human-like sentence [3].

Early image captioning methods include retrieval-based approaches, which match query images to similar ones in a database to retrieve captions [6, 11], and template-based systems, which generate captions using predefined visual elements and structured templates, such as a Nouns-Verbs-Scenes-Prepositions framework [15]. On other hand, modern deep learning methods, like the multi-modal neural language model proposed by Kiros et al., learn directly from data, enabling more expressive

& flexible captions without relying on templates or structured models [9].

Inspired by the encoder-decoder framework of RNNs [5], Vinyals et al. introduced the Neural Image Captioning (NIC) model, combining a CNN encoder and an RNN decoder within an encoder-decoder framework influenced by a similar setup in machine translation, to generate descriptive image captions [13]. Leading on to this framework, Attention mechanisms further improved such models by focusing on significant image regions during caption generation, as demonstrated by Xu et al.'s attention-based encoder-decoder network [4, 14].

This paper also focuses on the encoder-decoder framework, particularly employs a pre-trained ResNet-50 model as the encoder to extract high-dimensional feature vectors from the input images, and a Long Short-Term Memory (LSTM) network, a specialized type of Recurrent Neural Network (RNN) equipped with memory cells as a decoder, to process these feature embeddings and sequentially generate coherent and semantically descriptive captions for the given images.

2 Methodology

This section provides a detailed explanation of the architecture of the encoder and decoder, emphasizing their respective roles in addressing the image captioning task.

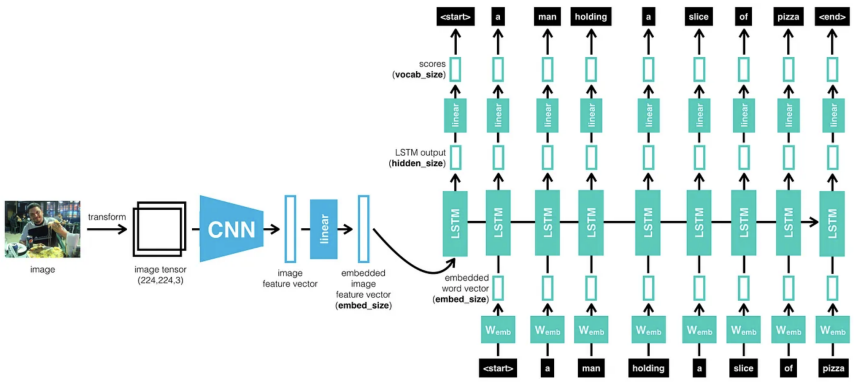

Figure 1: Proposed CNN-RNN Architecture [7].

2.1 Encoder CNN:

The encoder is constructed using a Convolutional Neural Network (CNN), which is highly effective in extracting rich feature representations from images through multiple layers of convolution and pooling. This proposed model employs a pre-trained ResNet-50 model [8], an architecture comprising 50 layers, including 16

residual blocks. The ResNet-50 framework utilizes residual connections, which facilitate the training of deeper networks by mitigating vanishing gradient issues, thereby making it an optimal choice for feature extraction in image captioning tasks.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 64 \\ 3\times3, 64 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 64 \\ 3\times3, 64 \\ 1\times1, 256 \end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 128 \\ 3\times3, 128 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times4$	$\begin{bmatrix} 1\times1, 128 \\ 3\times3, 128 \\ 1\times1, 512 \end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 256 \\ 3\times3, 256 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times6$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times23$	$\begin{bmatrix} 1\times1, 256 \\ 3\times3, 256 \\ 1\times1, 1024 \end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times2$	$\begin{bmatrix} 3\times3, 512 \\ 3\times3, 512 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$	$\begin{bmatrix} 1\times1, 512 \\ 3\times3, 512 \\ 1\times1, 2048 \end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure 2: ResNet Architectures with ResNet-50 architecture particularly highlighted [8].

As the objective of this implementation is not image classification, the final fully connected (FC) layer of the ResNet-50 model is removed, allowing for the extraction of feature vectors rather than class probabilities. These feature vectors are subsequently flattened and passed through an embedding layer to be transformed into a dense, lower-dimensional space. The resulting embedded feature vectors are then provided as input to the decoder, which generates descriptive captions for the images. To maintain the integrity of the pre-trained weights, the parameters of the ResNet-50 model are frozen during training, ensuring that the feature extraction process remains efficient and accurate without modifying the model's learned representations.

2.2 Decoder RNN:

The decoder in this image captioning model utilizes a Recurrent Neural Network (RNN), specifically a Long Short-Term Memory (LSTM) network, which is a special kind of RNN that incorporates a memory cell to retain information over extended periods and deals with the vanishing gradients problem.

The RNN decoder receives the embedded feature vector from the CNN encoder, which is provided as an input alongside a *'START'* token. At each time step, the LSTM takes the current word and the previous hidden state to update its memory cell and predict the next word in the caption, generating a probability distribution over the vocabulary. The word with the highest probability is selected and fed back as input for the subsequent time step. This iterative process enables the decoder to predict each successive word in the sequence, utilizing both the image content and the context established by the previously generated words. The process

continues until the decoder generates a *'STOP'* token or reaches the maximum caption length. By leveraging the sequential modeling capabilities of LSTMs and incorporating context from the encoder, the decoder ensures that the generated captions are both grammatically correct and contextually relevant.

3 Implementation Details

The implementation of this model involves four key stages: loading the dataset, image preprocessing, caption preprocessing, and encoder-decoder pipeline training.

3.1 Data set

The network uses the Microsoft Common Objects in Context (MS COCO) dataset, a comprehensive dataset designed for various tasks, including object detection, segmentation, keypoint detection, and caption generation [1]. The training set consists of 118,288 images, and the validation is performed on 5,000 image captions.

3.2 Image Preprocessing

Since the CNN encoder is based on the ResNet architecture, which is pre-trained on the ImageNet dataset, the image preprocessing steps are aligned with those outlined in the original ResNet paper [8] to preserve the model's optimal performance. This process includes:

- Resizing the images so that the shorter edge is 256 pixels.
- Random crop to a 224x224 pixel resolution.
- Augmentations such as random horizontal flipping and normalization

3.3 Caption Preprocessing

The image captions undergo preprocessing to facilitate their use in training. Each caption is tokenized into a sequence of words, which are then mapped to unique numerical indices using a vocabulary dictionary. This dictionary is constructed by iterating through all training captions and assigning an integer value to each distinct word. The tokenized captions are then converted into PyTorch tensors, making them suitable for model training. By predicting each subsequent token based on the preceding tokens, the model learns to generate coherent captions.

3.4 Training Setup

The training process involves defining hyperparameters and utilizing optimization techniques to ensure efficient learning. The batch size is set to 128, with the embedding size defined as 256, representing the dimensionality of both image and

word embeddings. The hidden size is set to 512, corresponding to the number of features in the hidden state of the RNN decoder. Cross-Entropy Loss is employed as the loss function, Adam optimizer is utilized to facilitate efficient gradient descent during training and the model's performance is subsequently validated using the BLEU score.

$x_{-1} = CNN(Img)$	Extracts image feature vector x_{-1} using a CNN to represent image content. x_{-1} is fed only once at $t = -1$ to the LSTM as feeding the image at every time step can lead to overfitting and inferior results.
$x_t = W_e S_t$, $t \in \{0, \ldots, N-1\}$	Each one-hot encoded word in S_t is mapped to vector space via embedding matrix W_e to generate x_t, which is the input to the LSTM at time t.
$p_{t+1} = LSTM(x_t)$, $t \in \{0, \ldots, N-1\}$	LSTM generates hidden state m_t and outputs word probability p_{t+1}, a probability distribution over all possible words in the vocabulary for the next word.

Table 1: Equation-Based Model Implementation Details [13]

4 Experimental Results

This proposed model has been trained over **four epochs**. To evaluate the quality of the generated captions, the BLEU (Bilingual Evaluation Understudy) score has been used which is a widely recognized metric for assessing machine-generated text by comparing it against human-generated reference captions. The BLEU score quantifies n-gram precision to assess content and structural alignment. The table below presents the BLEU scores for each epoch, highlighting the model's progressive improvement over time.

Epoch	**BLEU Score**
1	0.174
2	0.180
3	0.191
4	0.198

Figure 3: Evolved BLEU scores across training epochs.

Figure 4: Visualization of model outputs: 1st epoch (left) vs. 4th epoch (right) on same images.

The performance achieved in the final 4th epoch of this CRNN model has also been compared with the evaluation scores from existing implementations, as shown in the table below. This comparison helps to contextualize the model's performance and highlight its strengths and areas for improvement in caption generation relative to other state-of-the-art models.

Title	**Approach**	**Results**	**Year**
Show and Tell: A Neural Image Caption Generator [13]	NIC; vision CNN followed by a language generating RNN.	Achieved BLEU score of 0.277.	2015
Image Captioning with Semantic Attention [16]	CNN, LSTM, and Semantic attention.	Achieved BLEU score of 0.304.	2016
Self-critical Sequence Training for Image Captioning [12]	CNN, LSTM, and Reinforcement learning.	Achieved BLEU-4 score of 0.38.	2017
Camera2Caption: A real-time image caption generator [10]	Encoder and Decoder based implementation with significant modifications & optimizations.	Achieved BLEU-4 score of 0.22.	2017
Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering [2]	CNN, LSTM, and Attention mechanism.	Achieved BLEU-4 score of 0.36.	2018
This paper's proposed Implementation	ResNet-50 CNN with Seq2Seq LSTM	Achieved BLEU-4 score of 0.198	-

Figure 5: Comparison of various Image captioning approaches with their respective BLEU scores.

Furthermore, the model's performance, as reflected by the progressive increase in BLEU scores from epoch 1 to epoch 4, underscores its capacity for improvement with continued training. Figure 6 below provides a comparison between the model's strengths and limitations, illustrating examples of both accurate and inaccurate predictions observed after training for four epochs.

Figure 6: Top row: Examples of good results produced by the model. Bottom row: Instances of inaccurate predictions illustrating the model's limitations.

The value for minimum loss and perplexity evolved over each epoch during training is shown in Figure 7 below.

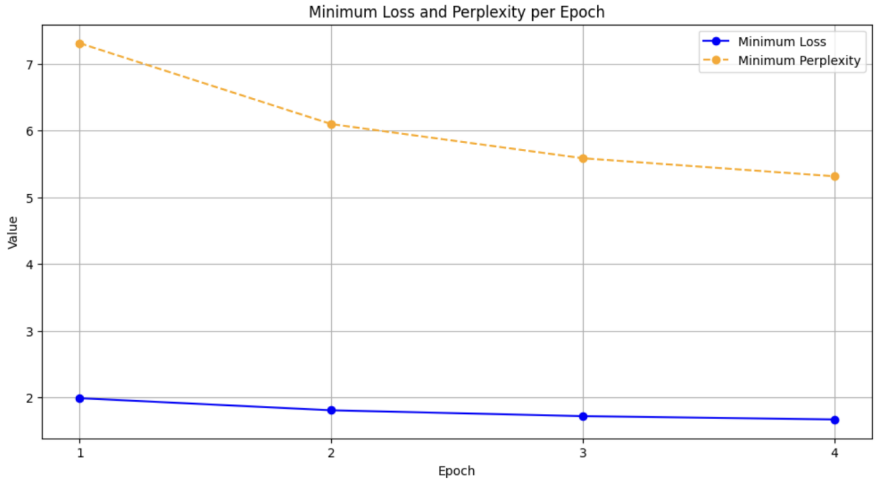

Figure 7: Minimum Perplexity and Loss Across Epochs.

5 Conclusion & Future Work

Due to limited computational resources, the model was trained for only 4 epochs, which itself took around 10 hours on the T4 GPU. However, as observed in Figure 3, the BLEU score shows a consistent improvement with each epoch, indicating a positive trend in the model's performance as training progresses. This suggests that further training over additional epochs could yield even better results by allowing the model to learn more complex patterns and improve the quality of

generated captions.

Further experimentation with hyperparameter tuning, optimization strategies, and data augmentation may also contribute to improved results. Incorporating advanced architectures or attention mechanisms, such as transformer-based models, could complement the CRNN framework to generate more accurate and contextually rich captions.

References

[1] COCO - Common Objects in Context — cocodataset.org. `https://cocodataset.org/#home`. [Accessed 10-12-2024].

[2] Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6077–6086, 2017.

[3] Shuang Bai and Shan An. A survey on automatic image caption generation. *Neurocomputing*, 311:291–304, 2018.

[4] Kyunghyun Cho, Aaron Courville, and Yoshua Bengio. Describing multimedia content using attention-based encoder-decoder networks. *IEEE Transactions on Multimedia*, 17(11):1875–1886, 2015.

[5] Kyunghyun Cho, Bart Merrienboer, Caglar Gulcehre, Fethi Bougares, Holger Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder-decoder for statistical machine translation. 06 2014.

[6] 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 Kostas Daniilidis, Petros Maragos, and Nikos Paragios, editors, *Computer Vision – ECCV 2010*, pages 15–29, Berlin, Heidelberg, 2010. Springer Berlin Heidelberg.

[7] Deepesh Garg. Automatic image captioning with pytorch. August 2020.

[8] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[9] Ryan Kiros, Ruslan Salakhutdinov, and Rich Zemel. Multimodal neural language models. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 595–603, Bejing, China, 22–24 Jun 2014. PMLR.

[10] Pranay Mathur, Aman Gill, Aayush Yadav, Anurag Mishra, and Nand Kumar Bansode. Camera2caption: A real-time image caption generator. In *2017 International Conference on Computational Intelligence in Data Science(ICCIDS)*, pages 1–6, 2017.

[11] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. In J. Shawe-Taylor, R. Zemel, P. Bartlett, F. Pereira, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 24. Curran Associates, Inc., 2011.

[12] Steven Rennie, E. Marcheret, Youssef Mroueh, Jarret Ross, and Vaibhava Goel. Self-critical sequence training for image captioning. 07 2017.

[13] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3156–3164, 2015.

[14] Kelvin Xu, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio. Show, attend and tell: neural image caption generation with visual attention. ICML'15, page 2048–2057. JMLR.org, 2015.

[15] Yezhou Yang, Ching Teo, Hal Daumé III, and Yiannis Aloimonos. Corpus-guided sentence generation of natural images. In Regina Barzilay and Mark Johnson, editors, *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 444–454, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

[16] Quanzeng You, Hailin Jin, Zhaowen Wang, Chen Fang, and Jiebo Luo. Image captioning with semantic attention. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4651–4659, 2016.