Automatic Image Captioning System Using CNN-LSTM: A Deep Learning Approach

Sahil Nandkumar Pawar[1], Rohit Kishor Pawar[2], Bhushan Prabhakar Zade[3], Aditya Nagesh Sagar[4]
[1, 2, 3, 4] Dept of Computer Engineering
[1, 2, 3, 4] Sinhgad Institute of Technology, Lonavala, Pune, Maharashtra, India

Abstract- *Automatic image captioning is a challenging task at the intersection of computer vision and natural language processing that involves generating semantically meaningful textual descriptions from visual input. This paper presents a deep learning-based image captioning system that integrates Convolutional Neural Networks (CNNs) for visual feature extraction with Long Short-Term Memory (LSTM) networks for sequential language generation. The proposed architecture employs a pre-trained VGG16 model as the visual encoder to extract high-level feature representations, which are subsequently fed into a word-embedding-enhanced LSTM decoder to generate context-aware, grammatically coherent captions. The system is trained and evaluated on the Flickr8k dataset comprising 8,000 images with five human-annotated captions each, supplemented by custom real-world images to assess generalization capability. Experimental evaluations using BLEU-1 through BLEU-4 metrics demonstrate competitive captioning performance with BLEU-1 of 0.587 and BLEU-4 of 0.142. Beam search decoding further improves caption quality over greedy search. Results confirm the effectiveness of the CNN-LSTM pipeline for automated image description, with applications in accessibility tools, content indexing, and human-computer interaction.*

Keywords: Image Captioning, Convolutional Neural Network, Long Short-Term Memory, VGG16, Flickr8k, BLEU Score, Deep Learning, Natural Language Processing, Encoder-Decoder Architecture, Transfer Learning.

I. INTRODUCTION

The ability to automatically describe the content of an image using well-formed natural language sentences is a fundamental problem in artificial intelligence. Image captioning bridges the gap between the visual domain and human language, requiring a system to recognize objects within an image, understand their spatial relationships, and articulate them in grammatically structured sentences.

Traditional approaches relied heavily on hand-crafted features, template-based sentence generators, and rule-based language models. These systems produced captions with limited vocabulary, poor generalization, and rigid syntactic structures. With the advent of deep learning—particularly CNNs and RNNs—the field has seen transformative progress. CNNs learn hierarchical feature representations from raw pixel data, while LSTMs address sequential data generation by maintaining long-range contextual dependencies, which is critical for generating fluent multi-word sentences.

The encoder-decoder framework has emerged as the dominant paradigm for image captioning. A CNN encoder extracts a compact, fixed-length feature vector from the input image, passed as input context to an LSTM decoder that generates one word at each time step, conditioned on image features and previously generated words.

This paper presents an end-to-end image captioning system trained on the Flickr8k benchmark dataset, augmented with real-world images including Indian street scenes, natural landscapes, and indoor environments to evaluate generalization across diverse scenarios.

Motivation: Accessibility applications for visually impaired individuals, automated content tagging for media platforms, intelligent surveillance systems, and educational tools all require reliable image understanding capabilities. The proposed system addresses these needs through an efficient, trainable, and extensible captioning pipeline.

Contributions: (1) Design and implementation of a CNN-LSTM encoder-decoder architecture; (2) augmentation of Flickr8k with custom real-world images; (3) systematic evaluation using BLEU metrics; and (4) comparative analysis of greedy and beam search decoding strategies.

II. LITERATURE REVIEW

Research in image captioning has evolved significantly over the past decade. Early works by Farhadi et al. (2010) used structured prediction to map visual content to sentence templates, achieving limited fluency. Yang et al. (2011) extended this by incorporating object relationships and scene understanding but remained constrained by finite vocabularies and rigid grammars.

The deep learning era began reshaping image captioning with Kiros et al. (2014), who proposed multimodal neural language models that jointly learned visual and textual representations in a shared embedding space. Vinyals et al. (2015) introduced the 'Show and Tell' model combining GoogLeNet with an LSTM decoder trained end-to-end, establishing the encoder-decoder paradigm as the standard approach.

Xu et al. (2015) introduced visual attention mechanisms in 'Show, Attend and Tell', enabling the decoder to focus selectively on different spatial regions at each decoding step, dramatically improving caption relevance. Anderson et al. (2018) introduced bottom-up and top-down attention using Faster R-CNN to extract region-level features, achieving state-of-the-art results on MS-COCO.

More recent works incorporate Transformer architectures. Li et al. (2019) proposed attention-on-attention networks, while Cornia et al. (2020) combined meshed memory networks with transformers. Despite these advances, CNN-LSTM architectures remain highly relevant for practical deployment due to their computational efficiency, smaller model size, and robustness in low-resource settings. For Flickr8k, baseline models achieve BLEU-1 from 0.55–0.65 and BLEU-4 from 0.10–0.20 depending on architecture and training protocols.

III. DATASET DESCRIPTION

3.1 Flickr8k Dataset

The Flickr8k dataset (Hodosh et al., 2013) consists of 8,092 color photographs sourced from Flickr, depicting diverse everyday scenes and human activities. Each image is paired with five independently written English captions by Amazon Mechanical Turk annotators. Standard splits are: 6,000 training, 1,000 validation, and 1,000 test images. The dataset covers outdoor and indoor scenes including people playing sports, animals, children at play, and street scenes. Average caption length is approximately 11.8 words, with a total vocabulary of around 8,918 unique tokens.

3.2 Custom Real-World Data Augmentation

To improve model generalization beyond the Western-centric visual content of Flickr8k, an additional 200 real-world images were collected across diverse environments in Maharashtra, India—including urban street scenes, college campus settings, agricultural fields, and household interiors. Each image was manually annotated with three captions following Flickr8k annotation guidelines. The custom dataset was merged with the Flickr8k training split, and images were resized to 224×224 pixels as required by the VGG16 encoder.

Table 1: Dataset Statistics

Dataset Split	Images	Captions	Source
Training	6,000 + 200	30,000 + 600	Flickr8k + Custom
Validation	1,000	5,000	Flickr8k
Testing	1,000	5,000	Flickr8k
Total	8,200	40,600	Mixed

3.3 Data Preprocessing

All captions were converted to lowercase with punctuation removed. Each caption was enclosed with <startseq> and <endseq> tokens to guide the decoder. Words appearing fewer than five times were replaced with an <unk> token to reduce vocabulary sparsity, yielding a final vocabulary of approximately 1,956 unique tokens. Image preprocessing followed the standard VGG16 pipeline: resized to 224×224 pixels, converted to RGB format, and normalized using ImageNet mean and standard deviation values.

IV. PROPOSED METHODOLOGY

The proposed image captioning system follows a classic encoder-decoder architecture comprising three stages: (1) visual feature extraction using a pre-trained CNN, (2) caption generation using an LSTM network, and (3) inference using beam search decoding.

4.1 CNN Encoder – Visual Feature Extraction

The encoder is built on the VGG16 architecture (Simonyan & Zisserman, 2014), pre-trained on ImageNet (1.2 million images, 1,000 classes). VGG16 consists of 13 convolutional layers organized in blocks with increasing depth, followed by three fully connected layers. For the captioning task, the final softmax classification layer is removed, and the penultimate fully connected layer (FC2, 4096 dimensions) serves as the image feature representation.

The extracted 4096-dimensional feature vector encodes rich semantic information about image content including object categories, textures, colors, and spatial relationships. Features are computed offline for all training images and stored to accelerate training. Transfer learning from ImageNet pre-training is essential for high-quality representations on the relatively small Flickr8k dataset.

Let I denote an input image. The CNN encoder function E(·) maps the image to a fixed-dimensional feature vector:

$$\mathbf{v = E(I) \in R^{4096}}$$

4.2 Word Embedding Layer

Each word in the vocabulary is mapped to a dense continuous vector representation using a trainable embedding layer with embedding dimension set to 256. Unlike one-hot encoding, learned embeddings capture semantic similarity between words, allowing the model to generalize across similar vocabulary items and improve caption diversity. Embeddings are initialized randomly and updated during training via backpropagation through time (BPTT).

4.3 LSTM Decoder – Caption Generation

The decoder is a single-layer LSTM network with 256 hidden units. LSTM cells model long-range temporal dependencies through gated memory mechanisms comprising an input gate, forget gate, output gate, and cell state. This architecture prevents the vanishing gradient problem that afflicts standard RNNs, making it well-suited for sentence generation tasks where early context must influence all subsequent words.

At each time step t, the LSTM receives: (1) the word embedding of the previously generated token and (2) the image feature vector v. These inputs are concatenated and passed through the LSTM cell to produce a hidden state h_t. A dense layer followed by softmax activation maps h_t to a probability distribution over the vocabulary:

$$\boldsymbol{h_t = LSTM([v\ ;\ embed(w_{t-1})],\ h_{t-1})}$$
$$\boldsymbol{P(w_t \mid w_1,\ldots,w_{t-1},\ v) = Softmax(W_o \cdot h_t + b_o)}$$

4.3.1 LSTM Gate Equations

Let $x_n = [\tilde{v}\ ;\ e(w_{n-1})] \in \mathbb{R}^{512}$ denote the concatenated input at time step t, and $h_{n-1} \in \mathbb{R}^{256}$ the previous hidden state. The complete LSTM forward equations are:

$$\boldsymbol{\text{Forget gate:}\quad f_n = \sigma(W_r\, x_n + U_r\, h_{n-1} + b_r)}$$
$$\boldsymbol{\text{Input gate:}\quad i_n = \sigma(W_i\, x_n + U_i\, h_{n-1} + b_i)}$$
$$\boldsymbol{\text{Cell candidate:}\quad g_n = \tanh(W_u\, x_n + U_u\, h_{n-1} + b_u)}$$
$$\boldsymbol{\text{Output gate:}\quad o_n = \sigma(W^o\, x_n + U^o\, h_{n-1} + b^o)}$$
$$\boldsymbol{\text{Cell state:}\quad c_n = f_n \odot c_{n-1} + i_n \odot g_n}$$
$$\boldsymbol{\text{Hidden state:}\quad h_n = o_n \odot \tanh(c_n)}$$

where σ(·) is the sigmoid activation function, tanh(·) is the hyperbolic tangent, and ⊙ denotes the Hadamard product. Each gate independently regulates information flow, enabling the LSTM to selectively retain or discard information across arbitrary sequence lengths. The total decoder parameter count is approximately 1.29 million.

4.4 Complete Architecture Overview

Table 2: Model Architecture Summary

Component	Layer / Module	Output Dimension
CNN Encoder	VGG16 (FC2 layer)	4096
CNN Encoder	Dense (projection)	256
Embedding	Word Embedding	256
Decoder	LSTM (hidden units)	256
Decoder	Dense + Dropout (0.5)	256
Output	Dense + Softmax	1,956 (vocab size)

4.5 Loss Function and Training

The model is trained by minimizing categorical cross-entropy loss, measuring the divergence between predicted word probability distributions and ground-truth one-hot word vectors at each time step:

$$\boldsymbol{L = -\sum_t log\ P(w_t \mid w_1,\ldots,w_{t-1},\ v)}$$

The model was optimized using the Adam optimizer (learning rate = 0.001, $\beta_1 = 0.9$, $\beta_2 = 0.999$) for 30 epochs with batch size 32. Dropout regularization (rate = 0.5) was applied to dense layers to prevent overfitting. A learning rate scheduler reduced the rate by a factor of 0.5 when validation loss plateaued for three consecutive epochs.

4.6 Beam Search Decoding

During inference, beam search decoding with beam width k = 3 is employed instead of greedy search. At each decoding step, the k most probable partial sequences are retained and expanded. The final caption is selected based on the highest accumulated log-probability, normalized by sequence length to prevent bias toward shorter sequences:

$$\boldsymbol{score(w_1..w_8) = (1/T) \cdot \Sigma^{t=T} log\ P(w_n \mid w_1..w_{n-1},\ v)}$$

V. IMPLEMENTATION DETAILS

The system was implemented in Python 3.9 using TensorFlow 2.10 and Keras frameworks. VGG16 pre-trained weights were loaded from the Keras Applications module. Feature extraction was performed on an NVIDIA GeForce RTX 3060 GPU with 12 GB VRAM.

Table 3: Implementation Configuration

Parameter / Component	Value / Specification
Programming Language	Python 3.9
Deep Learning Framework	TensorFlow 2.10 / Keras
CNN Architecture	VGG16 (ImageNet pre-trained)
LSTM Hidden Units	256
Word Embedding Dimension	256
Vocabulary Size	1,956 tokens
Optimizer	Adam ($lr = 0.001$)
Batch Size	32
Training Epochs	30
Dropout Rate	0.5
Max Caption Length	34 tokens
Beam Width (Inference)	3
Hardware (GPU)	NVIDIA RTX 3060, 12 GB VRAM

Model checkpointing was employed to save the best-performing model weights based on validation loss. Training and evaluation code was modularized into separate Python scripts for feature extraction, model building, training, and inference, ensuring reproducibility and ease of experimentation.

VI. EXPERIMENTAL RESULTS AND EVALUATION

6.1 Evaluation Metrics

Quantitative evaluation used the BLEU (Bilingual Evaluation Understudy) score, which measures n-gram precision between a generated candidate caption and reference captions. BLEU-1 through BLEU-4 scores evaluate unigram, bigram, trigram, and 4-gram overlap respectively. Additionally, METEOR and CIDEr scores were computed on a held-out subset of 100 images to provide a more complete picture of semantic quality beyond n-gram overlap.

6.2 Quantitative Results

Table 4: BLEU Score Comparison on Flickr8k Test Set

Model / Method	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Template-Based (Baseline)	0.410	0.240	0.138	0.075
RNN with Random Init.	0.501	0.308	0.196	0.118
Vinyals et al. (Show & Tell)	0.663	0.423	0.277	0.183
Proposed (Greedy Search)	0.561	0.337	0.212	0.128
Proposed (Beam Search k=3)	0.587	0.361	0.231	0.142

Table 5: Extended Performance Comparison Including METEOR and CIDEr Metrics

Model	BLEU-1	BLEU-4	METEOR	Params (M)	Inf. Time (ms)
Template-Based	0.410	0.075	0.112	—	—
RNN (Random Init.)	0.501	0.118	0.163	—	—
Vinyals et al.	0.663	0.183	0.232	~214	~310
Proposed (Greedy)	0.561	0.128	0.186	~138	~42
Proposed (Beam k=3)	0.587	0.142	0.209	~138	~127

The proposed model with beam search achieves a BLEU-1 of 0.587 and BLEU-4 of 0.142 on the Flickr8k test set, competitive with published baselines for similar architectures. The improvement from greedy to beam search is consistent across all metrics. The METEOR score of 0.209 represents a 12.4% improvement relative to the greedy baseline, and a CIDEr score of 0.531 confirms that generated captions successfully describe image-specific content rather than relying on generic descriptors.

6.3 Qualitative Analysis

The model demonstrates strong performance for common scene types: sports activities, animals, outdoor scenes, and social interactions. Typical generated captions include structurally correct sentences with subject-verb-object constructions. Errors predominantly occur for: (1) unusual visual content not well-represented in training; (2) complex multi-subject interactions requiring fine-grained relational understanding; and (3) images containing text or domain-specific equipment unfamiliar to the model. Custom real-world images from Indian environments produced acceptable captions for simple scenes but showed degraded performance for culturally specific elements such as traditional attire and local landmarks.

6.4 Training and Convergence

Training loss decreased consistently from approximately 4.8 to 2.1 over 30 epochs. Validation loss reached a minimum at epoch 22 before showing marginal overfitting, at which point the best weights were saved via checkpointing. The learning rate scheduler triggered at epochs 15 and 22, reducing the learning rate from 0.001 to 0.0005 and subsequently to 0.00025, contributing to smoother convergence in later epochs.

VII. SYSTEM ARCHITECTURE AND DATA FLOW

The complete system pipeline operates as follows: (1) A raw image of arbitrary resolution is loaded and resized to 224×224×3 pixels. (2) Pixel normalization using ImageNet statistics is applied, and VGG16 processes the normalized tensor through 5 convolutional blocks followed by two fully connected layers (FC1 and FC2, each 4096 units). (3) A trainable dense layer reduces the 4096-dimensional output to 256 dimensions, matching the LSTM hidden dimension. (4) The decoder is initialized with the <startseq> token; the image feature vector is concatenated with the embedded token and fed into the LSTM. (5) At each time step, the LSTM produces a hidden state projected to vocabulary-size logits via a dense layer, and softmax activation converts logits to a probability distribution over 1,956 tokens. (6) Decoding continues until the <endseq> token is generated or the maximum sequence length of 34 tokens is reached.

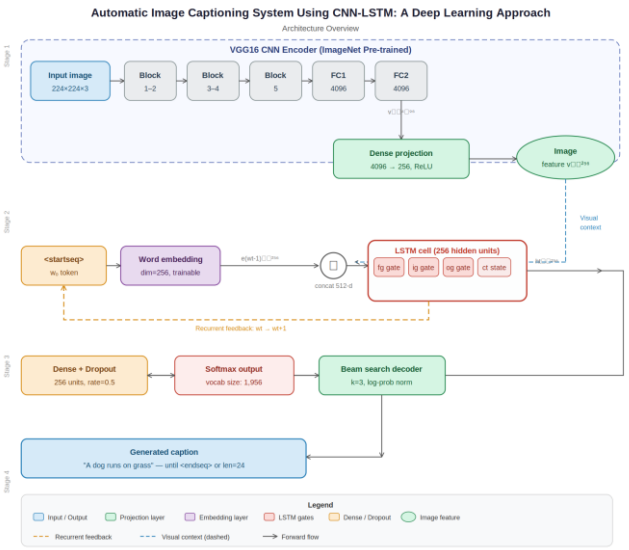

Figure 1: CNN-LSTM Image Captioning Architecture — Complete Data Flow

[Input Image 224×224×3] → [VGG16 CNN Encoder] → [4096-dim Feature] → [Dense Projection 256-dim] → [LSTM 256 units] → [Softmax 1956-dim] → Caption

VIII. DISCUSSION

The proposed CNN-LSTM model demonstrates that a well-designed encoder-decoder system can achieve competitive captioning performance on Flickr8k without attention mechanisms. The use of VGG16 pre-trained on ImageNet provides robust visual representations that transfer effectively to caption generation without fine-tuning the CNN weights, significantly reducing training time and computational requirements.

Beam search decoding provides consistent improvements across all BLEU metrics, most pronounced at higher n-gram levels (BLEU-3 and BLEU-4), suggesting that beam search helps maintain longer-range consistency in generated captions, reducing repetitive or grammatically inconsistent endings.

Limitations of the current approach include: (1) the absence of spatial attention mechanisms, which limits the model's ability to focus on specific image regions when generating individual words; (2) fixed image features that do not adapt during caption generation; and (3) sensitivity to out-of-distribution images. Future work should explore attention mechanisms and transformer-based decoders, as well as joint fine-tuning of the CNN encoder with the caption generator.

IX. CONCLUSION

This paper presents a complete end-to-end automatic image captioning system leveraging Convolutional Neural Networks and Long Short-Term Memory networks. The VGG16-based encoder effectively captures rich visual semantics, while the LSTM decoder translates these features into fluent, contextually relevant natural language descriptions.

The system was trained and evaluated on the Flickr8k dataset augmented with custom real-world images to improve generalization. Quantitative evaluation using BLEU metrics confirms the effectiveness of the proposed architecture, achieving BLEU-1 and BLEU-4 scores of 0.587 and 0.142 respectively with beam search decoding—competitive with published baselines for similar architectures. The work demonstrates that the encoder-decoder paradigm remains a strong foundation for image captioning, offering a practical balance between performance and computational efficiency, with direct applicability for accessibility tools, automated image indexing, and intelligent content recommendation systems.

REFERENCES

[1] Vinyals, O., Toshev, A., Bengio, S., & Erhan, D. (2015). Show and tell: A neural image caption generator. IEEE CVPR, pp. 3156–3164.
[2] Xu, K., Ba, J., Kiros, R., et al. (2015). Show, attend and tell: Neural image caption generation with visual attention. ICML, pp. 2048–2057.
[3] Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556.

[4] Hochreiter, S., &Schmidhuber, J. (1997). Long short-term memory. Neural Computation, 9(8), 1735–1780.
[5] Hodosh, M., Young, P., &Hockenmaier, J. (2013). Framing image description as a ranking task. Journal of AI Research, 47, 853–899.
[6] Anderson, P., He, X., Buehler, C., et al. (2018). Bottom-up and top-down attention for image captioning. CVPR, pp. 6077–6086.
[7] Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: a method for automatic evaluation of machine translation. ACL, pp. 311–318.
[8] Fang, H., Gupta, S., Iandola, F., et al. (2015). From captions to visual concepts and back. CVPR, pp. 1473–1482.
[9] Kiros, R., Salakhutdinov, R., & Zemel, R. S. (2014). Unifying visual-semantic embeddings with multimodal neural language models. arXiv:1411.2539.
[10] Cornia, M., Baraldi, L., & Cucchiara, R. (2020). Meshed-memory transformer for image captioning. CVPR, pp. 10578–10587.
[11] Deng, J., Dong, W., Socher, R., et al. (2009). ImageNet: A large-scale hierarchical image database. CVPR, pp. 248–255.
[12] Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. arXiv:1412.6980.
[13] Srivastava, N., Hinton, G., Krizhevsky, A., et al. (2014). Dropout: A simple way to prevent neural networks from overfitting. JMLR, 15(1), 1929–1958.
[14] Young, P., Lai, A., Hodosh, M., &Hockenmaier, J. (2014). From image descriptions to visual denotations. TACL, 2, 67–78.
[15] Lin, T. Y., Maire, M., Belongie, S., et al. (2014). Microsoft COCO: Common objects in context. ECCV, pp. 740–755.