Enhanced Textual Feature Extraction for Visual Question Answering: A Simple Convolutional Approach

Zhilin Zhang
New York University
New York, USA
zz10068@nyu.edu

Fangyu Wu*
University of Illinois Urbana-Champaign
Urbana, USA
fangyuw2@illinois.edu

Abstract

Visual Question Answering (VQA) has emerged as a highly engaging field in recent years, with increasing research focused on enhancing VQA accuracy through advanced models such as Transformers. Despite this growing interest, limited work has examined the comparative effectiveness of textual encoders in VQA, particularly considering model complexity and computational efficiency. In this work, we conduct a comprehensive comparison between complex textual models that leverage long-range dependencies and simpler models focusing on local textual features within a well-established VQA framework. Our findings reveal that employing complex textual encoders is not always the optimal approach for the VQA-v2 dataset. Motivated by this insight, we propose ConvGRU, a model that incorporates convolutional layers to improve text feature representation without substantially increasing model complexity. Tested on the VQA-v2 dataset, ConvGRU demonstrates a modest yet consistent improvement over baselines for question types such as Number and Count, which highlights the potential of lightweight architectures for VQA tasks, especially when computational resources are limited.

CCS Concepts

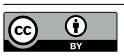
• **Computing methodologies** → Artificial intelligence; Natural language processing; Information extraction; Machine learning; Machine learning approaches; Neural networks; Artificial intelligence; Computer vision; Computer vision representations; Image representations.

Keywords

Visual Question Answering, Textual Feature Extraction, N-Grams, Convolution

ACM Reference Format:
Zhilin Zhang and Fangyu Wu. 2025. Enhanced Textual Feature Extraction for Visual Question Answering: A Simple Convolutional Approach. In *2025 6th International Conference on Computer Vision and Computational Intelligence (CVCI 2025), January 10–12, 2025, Hong Kong, Hong Kong*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3744725.3744731

*Corresponding author.



CVCI 2025, Hong Kong, Hong Kong

ACM ISBN 979-8-4007-1292-0/2025/01
https://doi.org/10.1145/3744725.3744731

1 Introduction

Visual Question Answering (VQA) [1] has emerged as an increasingly interesting area of research at the intersection of computer vision and natural language processing. The objective of VQA is to answer questions about a given image, requiring models capable of understanding both visual and textual modalities. One particular task in the VQA domain that attracts additional attention is the counting task. This specific task requires the model to quantitatively identify the number of objects in a given image. Despite its apparent simplicity, this task demands precise identification, localization, and counting of objects, and avoiding potential pitfalls like multiple counting or not acknowledging occluded items.

Advanced sequential models, such as Transformers, have seen great success in various tasks and have been widely adopted. However, these models are not always the best choice for every task, particularly in scenarios where balancing model performance with computational cost is essential. As highlighted in previous studies that evaluate models under specific conditions [2], carefully considering model selection and exploring simple yet effective improvements is also crucial for optimizing task performance.

In this paper, we pose the question: Are complex sequential models always the most suitable approach for handling textual modality in VQA tasks? To investigate this problem, we conduct comprehensive experiments applying both complex models like Transformer Encoder and attention-based models, as well as simpler structures such as RNNs and CNNs on a well-established VQA-v2 dataset. We focus our analysis on different text feature extraction methods and their impact on model accuracy. We discover that simpler models, specifically those good at capturing local interdependencies within the text, can provide improvements over some complex models. Furthermore, the incorporation of convolutional layers into simpler structures like Gated Recurrent Units (GRU), forming what we term as ConvGRU, proves quite feasible.

Our main contributions are threefold: (1) We present ConvGRU, a GRU-based text encoder enhanced by applying convolutional layers. Experimental results demonstrate that leveraging local textual features with ConvGRU can marginally improve VQA accuracy without increasing computational costs. (2) We conduct an extensive comparison of various VQA text encoders, and find that complex models such as Transformer Encoders may underperform compared to simpler architectures better at extracting local textual features.

2 RELATED WORK

Visual Question Answering. Visual Question Answering (VQA) has rapidly evolved since [1] first introduced the concept of free-form and open-ended VQA tasks, accompanied by a benchmark

dataset and evaluation metrics. [3] introduces complementary image-question pairs to mitigate the imbalance caused by language priors. Early VQA research primarily utilized Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) for image and question encoding, respectively [1] [4] [5] [6]. [4] exemplifies this approach through a two-branch neural network combining a CNN-based image encoder with an LSTM question encoder. The emergence of Transformer models has then catalyzed the development of cross-modal architectures in the VQA domain. Notably, models like ViLBERT [7], VisualBERT [8], and CLIP [9] have led the transition towards unified frameworks. These models have been instrumental in recent VQA research efforts [10] [11] [12] [13], showcasing remarkable results on VQA tasks.

Spatial Attention in VQA. Spatial attention mechanism is adopted in order to improve the performance of VQA and mimic human behavior, i.e., looking at certain parts of the regions [14] [15] [16]. Stacked attention networks (SAN) [14] and Spatial Memory Network (SMem) [15] similarly utilize two-stage attention framework to produce better glimpses through multiple reasoning. To generate a query-aware context representation without early summarization, [16] proposes a multi-stage hierarchical bi-directional attention network.

Text Representations in VQA. The task of Visual Question Answering (VQA) necessitates effective textual representation to comprehend questions that guide the visual understanding. Traditional approaches have predominantly employed Recurrent Neural Networks (RNNs), including Gated Recurrent Units (GRU) and Long Short-Term Memory (LSTM) networks [1] [4] [17]. Their ability to model the temporal dependencies inherent in natural language questions makes them robust baselines for text representation in VQA tasks. Beyond RNNs, the Text Convolutional Neural Network (TextCNN) [18] emerges as an alternative that leverages convolutions to extract local features, offering a different perspective on capturing text semantics [5]. The breakthrough of the Transformer [19] has introduced a shift towards attention-based methods for textual representation. The Transformer Encoder, with its self-attention mechanism, allows for the direct modeling of relationships between all words in a question, irrespective of their positional distances. Building on the success of Transformers, BERT (Bidirectional Encoder Representations from Transformers) [20] has emerged as another milestone in text representation adept at grasping the nuanced context of words in a sentence from both directions. This progression has culminated in Transformers and BERT becoming the dominant models in recent VQA works [10] [21].

3 Methodology

3.1 Problem Formulation

The primary objective of Visual Question Answering (VQA) is to accurately predict an answer given an image and a corresponding natural-language question. Therefore, the VQA task can be regarded as a classification problem, and its goal is to identify the most probable answer $\hat{a}$ from a predefined set of possible answers, based on the provided image i and a question q. Formally, this is expressed as:

$$\hat{a} = \underset{a}{arg\max}\, P(a|i,q) \tag{1}$$

where a denotes a potential answer within the set $A = \{a_1, a_2, a_3, ..., a_n\}$, with n indicating the total number of possible answers. In scenarios involving counting tasks within VQA, $\hat{a}$ signifies the number that most accurately responds to the posed question concerning the image.

3.2 Method Overview

Our methodology seeks to improve VQA accuracies by focusing on enhanced text processing techniques, leveraging the foundational work of [22]. We introduce a model that primarily innovates in textual feature extraction through a novel Convolutional GRU framework, and also integrate existing methods for image feature extraction and a counting component to address both visual data and numerical queries effectively.

At the heart of our proposed model, as depicted in Figure 1, is the integration of a Convolutional GRU framework for improved extraction of textual features from questions. For image modality, we employ a pre-trained R-CNN model for extracting features, accompanied by a stacked attention mechanism and a counting module from the base model.

3.3 Standard Models

We directly adopt several established methods to highlight our contributions in the textual modality and give a fair comparison of different textual encoders under the same setting.

3.3.1 Image Encoder. We employ a pre-trained R-CNN model with a ResNet-101 backbone [23] for image feature extraction to leverage its strengths of depth and computational efficiency through bottleneck blocks. Table 1 provides an overview of ResNet-101 structure and the arrangement of its bottleneck blocks.

3.3.2 Stacked Attention Mechanism. We utilize the stacked attention network proposed by [14], which employs multiple layers of attention to pinpoint relevant areas within an image iteratively.

At each step t, the mechanism calculates a glimpse x_t, a weighted mix of image features v and question features q, using:

$$x_t = \sigma\left(Conv\left(\left[dropout\left(v\right);\, Tile\left(dropout\left(q\right)\right)\right]\right)\right) \tag{2}$$

Here, σ represents the softmax function, applying the convolutional operation $Conv$ on the combined and processed features, where $dropout$ enhances model generalization, and $Tile$ replicates q across all spatial dimensions of v.

The attention map x_t is then employed to calculate a focused weighted sum of image features:

$$a_t = \sum_i x_t^i \cdot v^i \tag{3}$$

where x_t^i and v^i correspond to the i-th elements of x_t and v, respectively.

By applying this attention sequence iteratively, the network refines its focus, progressively integrating pertinent image information to inform the answer generation process.

3.3.3 Counting Module. Adopted from [22], the counting module integrates attention weights a and bounding boxes b as input to enable precise object counting. The process unfolds through several key operations, streamlined for clarity:

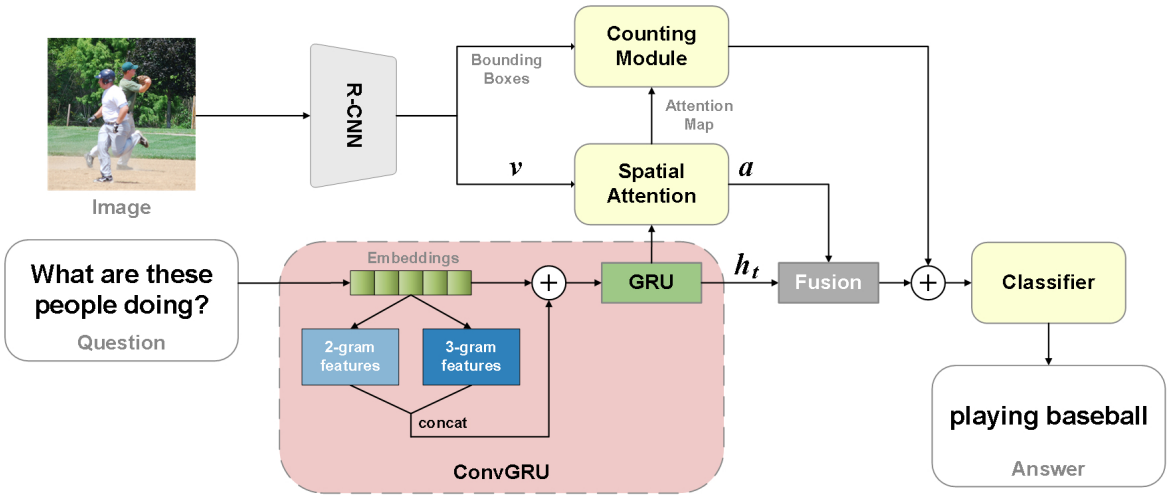


Figure 1: Overview of the VQA architecture highlighting the ConvGRU module. Questions are embedded and processed through convolutional layers to capture local semantics, then integrated via a GRU unit. Image features are extracted using ResNet-101 with spatial attention focusing on key areas. The fusion module combines image and text information, and the classifier outputs the answer. Counting, attention and classifier modules follow the configurations of [22].

Table 1: Detailed Structure of ResNet-101 Used for Image Feature Extraction. The table outlines the configuration of layers in ResNet-101, including convolutional layers, max pooling, and the bottleneck blocks.

Layer Name	Conv1	Pooling	Conv2_x	Conv3_x	Conv4_x	Conv5_x
Structures	7×7, 64	3×3	$1\times1,\ 64$ $(\ 3\times3,\ 64\ )\times3$ $1\times1,\ 256$	$1\times1,\ 128$ $(3\times3,\ 128)\times4$ $1\times1,\ 512$	$1\times1,\ 256$ $(\ 3\times3,\ 256\ )\times23$ $1\times1,\ 1024$	$1\times1,\ 512$ $(\ 3\times3,\ 512\ )\times3$ $1\times1,\ 2048$

Attention Matrix Formation: Attention weights are transformed into an attention matrix $A = aa^T$, mapping the relationship between different object proposals.

Duplicate Removal: Intra-object duplicates are eliminated by applying a mask created from the inverse of the Intersection over Union (IoU) scores between bounding boxes:

$$D_{ij} = 1 - IoU\left(b_i,\ b_j\right) \tag{4}$$

This produces a refined attention matrix $\hat{A} = A \odot D$, where $\odot$ denotes element-wise multiplication.

Inter-Object Differentiation: Inter-object duplicates are addressed by calculating a uniqueness score for each proposal. This involves assessing the similarity between proposals based on their attention weights and adjusted attention matrix, $\hat{A}$. Each proposal's score is inversely related to the number of similar proposals.

Count Matrix and Output: The final counting matrix C is computed by considering the scaled attention matrix and adding self-loops based on the uniqueness scores. The count output vector O is derived from C, and adjusted for the degree of overlap and distance from expected counts.

This module's design enables the VQA model to differentiate between and accurately count overlapping objects, thereby enhancing the model's overall counting accuracy.

3.3.4 Fusion and Classifier Module. Following the methodology described in [22], the fusion of features is executed as follows:

$$Fuse_{out} = ReLU\left(w_1x + w_2y\right) - \left(w_1x - w_2y\right)^2 \tag{5}$$

In this equation, w_1 and w_2 are weights assigned to features x (visual) and y (textual), respectively. This method combines linear and quadratic interactions, activated by $ReLU$, to merge these feature sets effectively.

For classification, a multi-modal approach is utilized, concatenating visual and question representations, along with the counting module's output. This concatenated result undergoes $ReLU$ activation and is further refined through batch normalization before proceeding to a fully-connected layer designed for a 3000-category classification task.

The model's accuracy is assessed using the Negative Log-Likelihood Loss Function, defined as:

$$Loss = -\frac{1}{N}\sum_{n=1}^{N}\log P(a_n|i,q) \tag{6}$$

Here, the loss is averaged across all correct answers to compute the final value, ensuring a comprehensive evaluation of the model's performance.

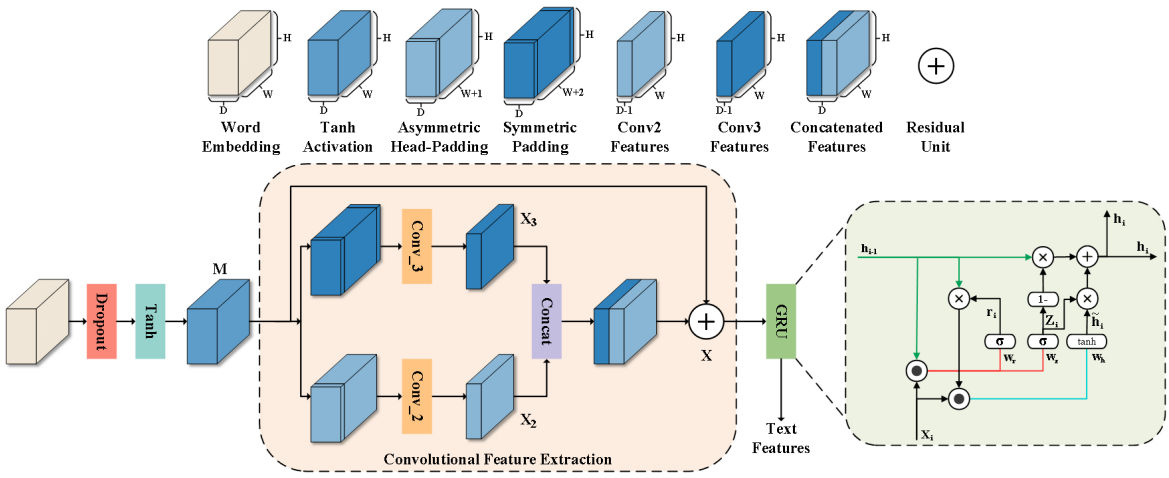


Figure 2: Schematic representation of the textual modality feature extraction model. Word embeddings are processed through dropout, Tanh activation, padding, and convolutional layers to capture multi-scale features. A residual unit is applied to ensure even questions with subtle n-gram features retain robust initial representations. Then, features are concatenated and fed into a GRU to extract sequential semantics. The inset details the GRU's internal gating mechanisms.

3.4 Convolutional GRU for Text Encoding

In Visual Question Answering (VQA), textual feature extraction traditionally utilizes Recurrent Neural Networks (RNNs) like Long Short-Term Memory (LSTM) and Gated Recurrent Units (GRU) due to their proficiency in modeling sequential data. GRUs are often favored for their computational efficiency, especially given the concise and straightforward nature of questions in VQA tasks. Our main contribution lies in enhancing the GRU architecture by incorporating convolutionally extracted n-gram features as inputs instead of direct word embeddings. This approach is theoretically grounded in the understanding that local n-gram patterns are crucial for capturing semantic meaning in text [24].

Considering the concise nature of questions, our analysis reveals that self-attention mechanisms might lead to overfitting, which is observed in our comparative analysis with RCNN+SA-GRU models (Section 4.3). We propose a one-dimensional convolutional approach as a more effective alternative for feature extraction. By employing kernels of sizes 2 and 3, we capture bi-gram and tri-gram features, respectively, which are then processed through distinct padding strategies-asymmetrical for bi-grams to prioritize the first word of the sequence, and symmetrical for tri-grams to preserve context evenly. Similar to the observations in [25] regarding the shift problem encountered with even-numbered convolutions in image processing, we identify a parallel issue in the realm of one-dimensional text convolutions. To avoid the potential complexity and overfitting that might arise from directly addressing this shift problem, we choose to simply add a unit of padding at the beginning of sequences to accurately identify the initial words of a question, which are crucial for defining its category. For instance, "what" typically signals a query about an object, while "what color" suggests a question about color. The utility of this asymmetric head padding technique is also validated by the results in Section 4.3.

Figure 2 outlines our Convolutional GRU (ConvGRU) model's architecture, highlighting the integration of convolutional n-gram feature extraction with GRU processing. Given a sequence of question words q, each word is first embedded into a high-dimensional space, yielding the matrix $E \in \mathbb{R}^{B\times L\times d}$, and followed by a Dropout and Tanh activation, the matrix is calculated as:
where B is the batch size, L is the sequence length, and d is the embedding size.

$$M = Tanh\left(Dropout\left(E\right)\right) \tag{7}$$

Convolution operations with kernels of sizes 3 and 2 are then applied to M, and their outputs are concatenated along with a residual connection to M:

$$X_3 = Conv3\left(SymPad\left(M\right)\right) \tag{8}$$

$$X_2 = Conv2\left(AsyHeadPad\left(M\right)\right) \tag{9}$$

$$X = [X_3; X_2] + M \tag{10}$$

where $SymPad$ represents symmetric padding, and $AsyHeadPad$ represents asymmetric head-padding.

The GRU then processes the enriched feature set X, calculating the update gate z_i, reset gate r_i, and the new memory cell $\widetilde{h_i}$ at each time step i, leading to the updated hidden state h_i:

$$r_i = \sigma\left(W_r \cdot x_i + U_r \cdot h_{i-1} + b_r\right) \tag{11}$$

$$z_i = \sigma\left(W_z \cdot x_i + U_z \cdot h_{i-1} + b_z\right) \tag{12}$$

$$\widetilde{h_i} = Tanh\left(W_h \cdot x_i + U_h \cdot \left(r_i \odot h_{i-1}\right) + b_h\right) \tag{13}$$

$$h_i = \left(1 - z_i\right) \odot h_{i-1} + z_i \odot \widetilde{h_i} \tag{14}$$

where σ is the sigmoid function, $\odot$ is the element-wise multiplication, W_r, W_z, W_h, U_r, U_z, U_h are weight matrices, and b_r, b_z, b_h are bias vectors.

With this schema, our Convolutional GRU improves upon traditional methods by adeptly capturing and emphasizing pivotal textual features, thus providing a robust foundation for more accurate responses.

4 Experiments

4.1 Dataset and Evaluation Metric

VQA-v2 Dataset. A notable change of Visual Question Answering was the transition from the VQA dataset to VQA-v2 dataset in order to address a number of biases and language priors found in the original dataset [3]. For instance, "blue" is frequently selected as the answer to questions beginning with "What color is ..." regardless of the image content, which may lead to improper correlations between the question text and the answer. To cope with these constraints, VQA-v2 introduces complementary pairs, wherein there is a "complementary" image that yields a different answer to the same question for every question linked to an image. In our experiments, VQA-v2 dataset is chosen as our primary focused task, and we assess our model on validation set after training on the training set.

Evaluation metric. In the realm of VQA, accuracy is calculated using a unique method. For each question, the model's answer is compared with human responses. The accuracy for a particular answer is calculated by:

$$Acc = \frac{1}{|A|} \sum_{a \in A, |a|=|A|-1} \min\left(\frac{1}{3} Agree\,(a), 1\right) \tag{15}$$

where A is the human answer set including 10 answers, and $Agree(\cdot)$ is the number of human answers same with the predicted one. An answer is considered correct if at least 3 humans give the same answer to ensure accuracy while accounting for the possibility of multiple correct answers to the same question.

To address language priors and answer imbalance in the dataset, we also employ the **Balanced Pairs Accuracy** [26] metric. It forms complementary pairs by asking the same question about two different images requiring different answers. The model receives credit only if it answers both correctly. Mathematically, Balanced Pairs Accuracy is calculated as:

$$\overline{Acc} = \frac{1}{B} \sum_{k=1}^{B} \delta\left(Acc_k^{(1)} == 1 \; and \; Acc_k^{(2)} == 1\right) \tag{16}$$

where $\overline{Acc}$ is the Balanced Pairs Accuracy, B denotes the number of complementary pairs, $Acc_k^{(1)}$ and $Acc_k^{(2)}$ are the accuracies for two images in the k-th pair, computed using Equation (15). $\delta(\cdot)$ returns 1 if both accuracies are 1, and 0 otherwise.

4.2 Experimental Settings

4.2.1 Training Details. All experiments described in this study were conducted on a single RTX 3090 GPU with 24GB of memory. For the purpose of maintaining consistency and fairness across all tests, we utilized uniform parameters when evaluating a variety of Text Feature Extraction Models (**under the same random seed**). Specifically, the following settings were applied to each experiment:

Training Duration: Each model was trained for a total of 100 epochs, incorporating an early stopping mechanism to prevent overfitting. Specifically, training would be terminated if there was no improvement in performance observed over a span of 20 consecutive epochs.

Learning Rate: We initiated the training with a learning rate of 0.001. To manage the learning rate dynamically, an exponential decay strategy was used, formulated as:

$$lr_{i+1} = lr_i \times 0.5^{(1/lr\_half)} \tag{17}$$

Here, lr_i represents the learning rate at the current epoch, and lr_{i+1} denotes the learning rate for the next epoch. The term lr_half is set to 50,000, indicating that the learning rate is halved every 50,000 epochs.

Batch Size: The batch size was consistently set to 256 for all models.

Object Proposals: In our experiments, the maximum number of object proposals per image was set to 100.

4.2.2 Compared Methods. Image features are computed from R-CNN model with ResNet-101 as backbone mentioned in Section 3.3.1. Different text models are adopted, including basic one-layer Gated Recurrent Unit (GRU), Text Convolutional Neural Network (TextCNN), two-layer Long Short-Term Memory Network(2-Layer-LSTM), Bidirectional GRU (BiGRU), Bidirectional LSTM (BiLSTM), Self-Attention GRU (SA-GRU), Multi-Head Self-Attention GRU (MHA-GRU), Transformer Encoder (TE) and Transformer Encoder with GRU (TE-GRU). Specific settings are as follows:

One-Layer Gated Recurrent Unit (GRU): The baseline of our experiments, which is directly based on the work of [22].

TextCNN: Similar to the work of [18] except that incorporates a single convolutional layer with an assortment of kernel sizes, specifically 3, 4, and 5 (referring [5]).

2-Layer-LSTM, BiGRU, BiLSTM: Extended from the One-Layer GRU and One-layer LSTM by simply adding bidirectional processing or an extra layer.

Self-Attention GRU (SA-GRU) and Multi-head Self-Attention GRU (MHA-GRU): The SA-GRU incorporates a self-attention mechanism [19] into the standard GRU architecture, while the MHA-GRU, an extension of SA-GRU, employs a multi-head self-attention mechanism that divides the input sequence into multiple segments for simultaneous processing. In our experiments, the configuration with 5 attention heads was utilized.

Transformer Encoder (TE) and Transformer Encoder GRU (TE-GRU): Transformer Encoder (TE) directly utilizes the encoder component of the original Transformer model proposed by [19]. In the Transformer Encoder GRU (TE-GRU), input embeddings are first processed through Transformer Encoder, and then fed into the GRU layer. Similar to MHA-GRU, different heads were tested, and the best performance was achieved when set to 4 or 5.

4.2.3 ConvGRU Settings. For our ConvGRU model (Section 3.4), we experimented with different kernel sizes (1, 2, 3, and 4) for convolutional layers to see how they affect text feature extraction. Following the idea from [27], we used multiple kernel sizes to capture text features at various scales. Instead of pooling, we chose to concatenate features from different kernels to avoid losing too much textual information since most input questions are quite short. Our main focus was on multi-scale kernels, specifically combinations of $k = 2 + 3$ and $k = 1 + 2 + 3$, to leverage multiple text feature representations.

Table 2: Comparison of different textual models. "SA" and "MHA" refer to Self-Attention and Multi-Head Self-Attention respectively, and all configurations are the same except for textual encoders.

Method	VQA Accuracy (%)			Balanced Pair Accuracy (%)		
	Number	Count	All	Number	Count	All
RCNN+GRU [22]	49.40	57.13	65.42	22.98	26.47	37.26
RCNN+TextCNN	47.68	55.23	64.00	21.42	24.71	35.12
RCNN+2-Layer-LSTM	48.23	55.74	64.98	22.34	25.81	36.76
RCNN+BiGRU	48.84	56.57	65.44	22.34	25.81	37.38
RCNN+BiLSTM	48.11	55.68	65.15	21.70	25.09	36.97
RCNN+SA-GRU	46.04	53.11	64.47	18.96	21.85	35.96
RCNN+MHA-GRU	48.71	56.25	64.92	22.65	26.11	36.94
RCNN+TE	46.70	53.98	63.53	20.47	23.72	35.41
RCNN+TE-GRU	48.20	55.77	64.66	21.77	25.14	36.31

Table 3: Performance comparison of ConvGRU configurations with varying kernel sizes (denoted by numbers in parentheses) and the residual connections (indicated by 'Res'). "Head" and "Tail" refer to asymmetric head padding and asymmetric tail padding discussed in Section 3.4, respectively.

GRU Setting	VQA Accuracy (%)					Balanced Pair Accuracy (%)				
	Yes/No	Number	Count	Other	All	Yes/No	Number	Count	Other	All
GRU [22]	81.81	49.40	57.13	57.20	65.42	55.01	22.98	26.47	28.29	37.26
+ Conv(1)	81.76	49.07	56.84	57.14	65.33	54.71	22.64	26.20	28.25	37.09
+ Conv(2)	81.70	49.46	57.18	57.03	65.31	54.60	23.02	26.60	28.20	37.08
+ Conv(3)	81.80	49.22	56.86	57.21	65.40	54.93	23.00	26.54	28.40	37.29
+ Conv(4)	81.53	46.40	53.61	57.11	64.88	54.64	19.45	22.44	28.17	36.59
+ Conv(1+2+3)	81.85	49.11	56.84	57.27	65.43	54.78	22.95	26.47	28.22	37.15
+ Conv(2+3+Res,Tail)	81.70	49.26	56.96	57.20	65.36	54.68	23.11	26.69	28.26	37.15
+ Conv(3+Res)	81.88	49.36	57.12	57.19	65.44	54.93	**23.43**	**27.03**	28.10	37.21
+ Conv(2+3+Res,Head)	81.96	49.53	57.26	57.33	**65.56**	55.09	**23.36**	**26.94**	28.41	**37.41**

4.3 Results

Table 2 presents the performance of different Text Feature Extraction Models in terms of VQA Accuracy (%) and Balanced Pair Accuracy (%) on VQA-v2 validation set. Our experiments show that using complex models on the text modality doesn't always help with VQA tasks. In fact, models with complex features like Transformer Encoders and attention mechanisms (Self-Attention and Multi-Head Self-Attention) perform worse than simpler ones. This suggests that for VQA tasks, where questions are short and have similar meanings, these complex models good at capturing long-distance dependencies or global features might not always be necessary.

On the contrary, local text features seem to play a more pivotal role in determining model accuracy considering the nature of questions in VQA-v2 dataset. This can be seen from the results shown in Table 3. The convolutional layer and residual unit in ConvGRU tend to capture extra local dependencies within the text data effectively without losing information from initial text representations. When comparing the ConvGRU variants with kernel sizes ranging from 1 to 4 (Conv(1) to Conv(4)), a clear pattern emerges: The Conv(1) model, equally performing a linear projection, offers limited benefits in terms of capturing the nuanced relationships between words in a question. On the other hand, models with larger kernel sizes, like Conv(4), demonstrate a decline in performance, suggesting that excessively large kernels may overlook crucial local textual features. This observation implies that kernels larger than 4 are too broad, potentially diluting the model's focus on the immediate contextual relevance between adjacent words. Conversely, kernel sizes of 2 and 3 strike a balance.

This insight led us to explore multi-scale convolution strategies, combining the strengths of kernel sizes 2 and 3. Our experiments confirm that this multi-scale approach, particularly with asymmetric head-padding (Conv(2+3+Res,Head)), marginally outperforms the equivalent configuration with tail-padding. The slight preference for head-padding may be attributed to its emphasis on the initial segments of questions, which often direct the types of questions.

Although the Conv(3+Res) shows competitive, and in some aspects, superior performance to Conv(2+3+Res,Head), the latter's broader applicability across different text scenarios, as further discussed in Section 4.6, positions it as a more versatile model. The Conv(2+3+Res,Head) model not only capitalizes on multi-scale convolution benefits but also enhances the interpretability and adaptability of the model to various question types within the VQA

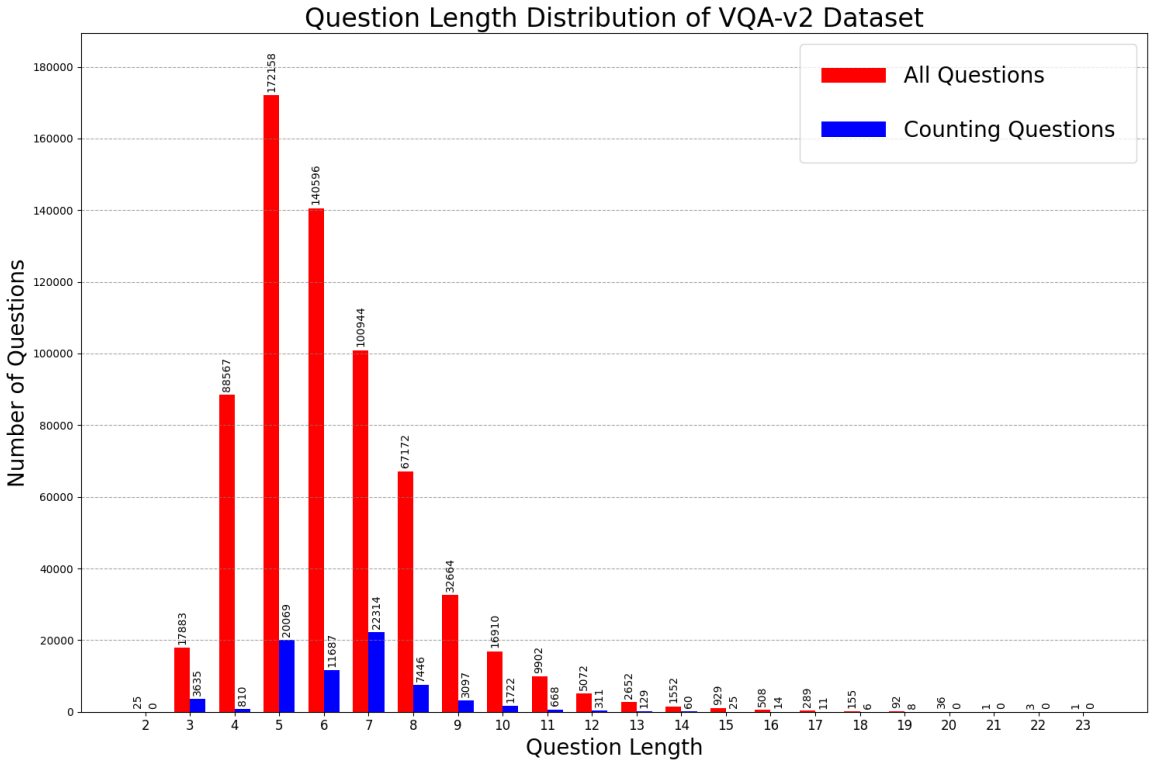


Figure 3: Distribution of question lengths within the VQA-v2 dataset. Red bars represent the total number of questions at each length, while blue bars show the number of counting questions that start with 'How many'.

framework. This makes it an optimal choice for addressing the diverse challenges presented by the VQA tasks.

Also, it needs to point out that relatively incremental improvement by ConvGRU over the baseline GRU model may stem from limitations inherent in the Image Encoder, attention mechanisms, and the visual-textual fusion methods employed. Therefore, advancements in these areas might further unlock the potential of ConvGRU in VQA tasks.

4.4 Unexpected Shortfalls: Why Transformer Encoder Struggles with VQA Text

The underperformance of Transformer Encoder in Visual Question Answering (VQA) tasks may initially appear surprising, given their success in broader NLP applications. This discrepancy arises from the unique textual characteristics within VQA, where questions are typically short and semantically similar. Unlike many NLP tasks where global contextual insights are crucial, VQA demands precise extraction of information from concise queries.

Transformer Encoder is good at capturing relationships in extensive text sequences through self-attention mechanisms. However, VQA tasks often involve brief textual prompts seeking specific visual details. This context significantly limits the utility of global contextual analysis, as the essence of VQA questions lies in localized cues rather than extensive narrative contexts. Furthermore, the semantic similarity among VQA questions adds another layer of complexity. Transformer Encoder may not effectively prioritize the subtle distinctions critical in VQA. Thus, their architectural advantages, while transformative for general NLP, may inadvertently obscure the localized, specific cues vital for accurate Visual Question Answering.

In essence, the architectural design of Transformer Encoder, though revolutionary for capturing global textual relationships, aligns less effectively with the requirements of VQA tasks. This misalignment underscores the necessity for approaches that emphasize local feature extraction, suggesting a tailored adaptation of text models to better accommodate the concise and semantically concentrated nature of VQA queries.

4.5 How Short Can VQA Questions Get?

We conduct an analysis of the length distribution of questions in the VQA-v2 dataset to demonstrate how the convolutional extraction of local features can lead to performance improvements.

As shown in Figure 3, we use bar chart to represent the distribution, with the X-axis indicating the length of questions and the Y-axis indicating the frequency of questions at each length. In our comprehensive analysis of 658,111 questions from VQA-v2 dataset, we found that a significant majority, 636,894 questions (96.78%), have lengths ranging from 3 to 10 words. Notably, 569,437 questions (86.53%) are within the 4 to 8-word range. A steep decrease in question frequency is observed with increasing length. Focusing on

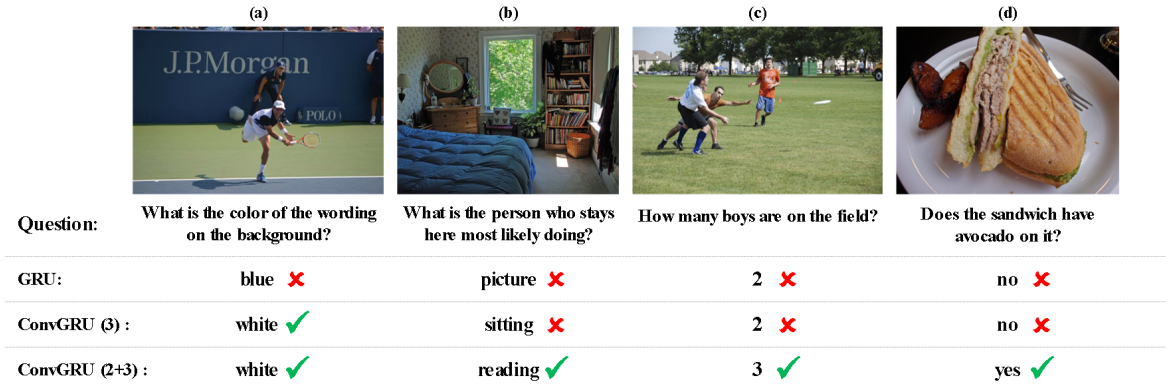


Figure 4: Qualitative results on validation samples indicating how GRU models enhanced with convolutional layer can predict more accurate answers.

the "How many" counting questions (72,012 in total), a predominant 61,516 questions (85.42%) fall within the 5 to 8-word range, with questions exceeding 10 words represent a mere 4.1% of counting queries.

These insights highlight the predominance of short questions in VQA-v2 dataset, demonstrating the effectiveness of smaller convolution kernels for local feature extraction. The prevalence of question-type determinants at the start of queries ("How many", "Is there", "Do", etc.) further advocates for the superiority of compact kernels. This observation also aligns with the performance dip for k=4 scenarios depicted in Table 3, reinforcing smaller kernels' utility in enhancing textual modality within VQA frameworks.

4.6 Qualitative Results

In this section, we present four distinct cases from validation set to demonstrate the impact of different GRU configurations on prediction accuracy (Figure 4). We compare the **GRU**, **ConvGRU(3)**, and **ConvGRU(2+3)** models across cases (a) to (d).

As shown in Case (a), the original GRU incorrectly predicts the answer to be "blue", likely confusing the color of the background with that of the text. Conversely, both the ConvGRU(3) and the ConvGRU(2+3) models accurately identify the correct answer as "white". This illustrates the capability of these enhanced models to capture key phrase pairs such as "color of" and "wording on" through 2-grams, and richer context through 3-grams like "the color of" and "of the wording," which effectively utilize the convolutional layers to discern the finer details of the question.

In Case (b), the GRU model gives a completely wrong prediction of "picture" as it relies solely on simplistic word-level features. The ConvGRU(3) model improves by predicting "sitting"—a rationally deduced yet contextually incorrect response, revealing that 3-gram features effectively capture a person's state or action in a specific location. The ConvGRU(2+3), by incorporating both 2-gram and 3-gram features, correctly comprehends the query's intent and the subject's context, leading to an accurate "reading" prediction. This case exemplifies how combining different n-grams can enrich linguistic feature understanding and enhance alignment with image features for more accurate predictions.

In both the counting task and the yes/no task, as shown in Case (c) and Case (d), ConvGRU(2+3) showcases best performance. The ability to understand relational context and specific details allows it to correctly count "3" boys on the field and accurately confirm the presence of avocado on the sandwich with a "yes". These outcomes illustrate the effectiveness of integrating multi-gram features into GRUs.

Across all cases, the results consistently demonstrate that ConvGRU(2+3) achieves higher accuracy compared to ConvGRU(3) and the original GRU. This highlights the effectiveness of integrating multi-gram features in enhancing the GRU's comprehension of textual information, thereby improving VQA accuracy without significantly increasing computational costs.

5 Conclusion

This paper explored the effectiveness of complex versus simpler models for capturing textual features in Visual Question Answering (VQA) tasks, specifically addressing whether complex sequential models are optimal for this purpose. Our experiments demonstrated that the ConvGRU model, designed to capture local text features, marginally outperformed the baseline GRU model, while advanced models like Self-Attention and Transformer Encoders showed reduced effectiveness. The primary advantage of ConvGRU lies in its ability to balance model performance with efficiency.

Nonetheless, our model is limited in scope, as it does not address other components of the VQA framework, such as the image encoder, where more advanced models beyond the Region-based Convolutional Neural Networks (R-CNN) could potentially improve results. Future research should focus on targeted improvements within both individual components and fusion methods to further optimize multi-modal understanding. Potential directions include exploring advanced neural architectures tailored for visual data and developing fusion techniques that can more effectively integrate visual and textual modalities to foster a deeper and more contextually relevant understanding in VQA tasks.

Acknowledgments

The authors explicitly declare that the writing and revision of this paper were conducted by the authors themselves without the use of AI-generated text.

References

[1] Antol, S., Agrawal, A., Lu, J., Mitchell, M., Batra, D., Zitnick, C. L., & Parikh, D. (2015). Vqa: Visual question answering. In Proceedings of the IEEE international conference on computer vision (pp. 2425-2433).

[2] Du, S., Li, J., & Noto, M. (2024). Comparison and Analysis of Three MobileNet-Based Models for Wildfire Detection. Journal of Advances in Information Technology, 15(4).

[3] Goyal, Y., Khot, T., Summers-Stay, D., Batra, D., & Parikh, D. (2017). Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 6904-6913).

[4] Malinowski, M., Rohrbach, M., & Fritz, M. (2015). Ask your neurons: A neural-based approach to answering questions about images. In Proceedings of the IEEE international conference on computer vision (pp. 1-9).

[5] Wang, Z., & Ji, S. (2018, May). Learning convolutional text representations for visual question answering. In Proceedings of the 2018 SIAM international conference on data mining (pp. 594-602). Society for Industrial and Applied Mathematics.

[6] Liu, F., Xiang, T., Hospedales, T. M., Yang, W., & Sun, C. (2018). ivqa: Inverse visual question answering. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 8611-8619).

[7] Lu, J., Batra, D., Parikh, D., & Lee, S. (2019). Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. Advances in neural information processing systems, 32.

[8] Li, L. H., Yatskar, M., Yin, D., Hsieh, C. J., & Chang, K. W. (2019). Visualbert: A simple and performant baseline for vision and language. arXiv preprint arXiv: 1908.03557.

[9] Radford, A., Kim, J. W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., ... & Sutskever, I. (2021, July). Learning transferable visual models from natural language supervision. In International conference on machine learning (pp. 8748-8763). PMLR.

[10] Gardères, F., Ziaeefard, M., Abeloos, B., & Lecue, F. (2020, November). Conceptbert: Concept-aware representation for visual question answering. In Findings of the Association for Computational Linguistics: EMNLP 2020 (pp. 489-498).

[11] Ravi, S., Chinchure, A., Sigal, L., Liao, R., & Shwartz, V. (2023). Vlc-bert: Visual question answering with contextualized commonsense knowledge. In Proceedings of the IEEE/CVF winter conference on applications of computer vision (pp. 1155-1165).

[12] Song, H., Dong, L., Zhang, W. N., Liu, T., & Wei, F. (2022). Clip models are few-shot learners: Empirical studies on vqa and visual entailment. arXiv preprint arXiv:2203.07190.

[13] Zhou, L., Palangi, H., Zhang, L., Hu, H., Corso, J., & Gao, J. (2020, April). Unified vision-language pre-training for image captioning and vqa. In Proceedings of the AAAI conference on artificial intelligence (Vol. 34, No. 07, pp. 13041-13049).

[14] Yang, Z., He, X., Gao, J., Deng, L., & Smola, A. (2016). Stacked attention networks for image question answering. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 21-29).

[15] Xu, H., & Saenko, K. (2016). Ask, attend and answer: Exploring question-guided spatial attention for visual question answering. In Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11–14, 2016, Proceedings, Part VII 14 (pp. 451-466). Springer International Publishing.

[16] Seo, M., Kembhavi, A., Farhadi, A., & Hajishirzi, H. (2016). Bidirectional Attention Flow for Machine Comprehension. ArXiv, abs/1611.01603.

[17] Kazemi, V., & Elqursh, A. (2017). Show, ask, attend, and answer: A strong baseline for visual question answering. arXiv preprint arXiv:1704.03162.

[18] Kim, Y. (2014). Convolutional Neural Networks for Sentence Classification. Conference on Empirical Methods in Natural Language Processing.

[19] Vaswani, A. (2017). Attention is all you need. Advances in Neural Information Processing Systems.

[20] Devlin, J. (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

[21] Yang, Z., Lu, Y., Wang, J., Yin, X., Florencio, D., Wang, L., ... & Luo, J. (2021). Tap: Text-aware pre-training for text-vqa and text-caption. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 8751-8761).

[22] Zhang, Y., Hare, J., & Prügel-Bennett, A. (2018). Learning to count objects in natural images for visual question answering. arXiv preprint arXiv:1802.05766.

[23] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

[24] Wang, P., Xu, B., Xu, J., Tian, G., Liu, C. L., & Hao, H. (2016). Semantic expansion using word embedding clustering and convolutional neural network for improving short text classification. *Neurocomputing*, *174*, 806-814.

[25] Wu, S., Wang, G., Tang, P., Chen, F., & Shi, L. (2019). Convolution with even-sized kernels and symmetric padding. Advances in Neural Information Processing Systems, 32.

[26] Teney, D., Anderson, P., He, X., & Van Den Hengel, A. (2018). Tips and tricks for visual question answering: Learnings from the 2017 challenge. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 4223-4232).

[27] Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., ... & Rabinovich, A. (2015). Going deeper with convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 1-9).