CSL 4020 - Course Project

Visual-Question-Answering

March 2025

Team Members

- Akshat Jain (B22CS007)
- Dev Jayesh Pandya (B22AI016)
- Ujjwal Jain (B22CS057)
- Bhavyadeep Singh Hada (B22BB013)

Abstract

Visual Question Answering (VQA) is a challenging multi-modal task where a system must reason over both visual content and natural language to generate accurate responses. In this work, we implement the VQA task using four distinct methodologies that integrate state-of-the-art techniques from computer vision and natural language processing. Our first approach leverages a ResNet50 convolutional network for image feature extraction combined with an LSTM for encoding the question. In the second method, we incorporate a multi-layer attention mechanism to better align visual features with language representations. The third model enhances the language component by integrating BERT into the attention framework, thereby providing contextualized embeddings for improved question understanding. Finally, the fourth approach employs a Faster R-CNN backbone for refined object-level image features and utilizes a multi-head attention module in combination with BERT to achieve superior fusion of the modalities. Experimental evaluations on benchmark datasets demonstrate the effectiveness of these techniques in capturing complex interactions between image and text modalities in VQA. We train and evaluate our models on the VQA v2 dataset, which contains approximately 440k question-answer pairs, and analyze how each enhancement contributes to performance improvements

1. Problem Statement

The Visual Question Answering (VQA) problem lies at the intersection of computer vision and natural language processing. It requires AI models to generate precise answers to free-form questions about images—a task that demands a nuanced understanding of specific objects, actions, and contextual details, far beyond the general descriptions offered by traditional image captioning. This multi-modal challenge involves object detection, scene understanding, activity recognition, commonsense reasoning, and language comprehension, making it inherently complex. Furthermore, evaluating VQA models is challenging due to the subjectivity of some responses and the intricacies of mapping visual content to textual answers.

Figure 1: Fig. 1. Examples of open-ended questions

VQA can be used in a variety of applications, which include aiding visually-impaired users in understanding their surroundings ("What temperature is this oven set to?"), analysts in making decisions based on large quantities of surveil-lance data ("What kind of car did the man in the red shirt drive away in?"),and interacting with a robot ("Is my laptop in my bedroom upstairs?"). This project has the potential to fundamentally improve the way visually-impaired users live their daily lives, and revolutionize how society at large interacts with visual data.

In our project, we implement the VQA task using a series of advanced deep learning techniques. We begin with a baseline model that employs a ResNet50 for image feature extraction paired with an LSTM for question encoding. Building on this, we incorporate multi-layer attention mechanisms to better align and fuse visual and textual features. We further enhance the system by integrating BERT for contextualized language embeddings, which improves question understanding. Finally, we adopt a Faster R-CNN backbone combined with a multi-head attention module and BERT, enabling refined object-level feature extraction and superior fusion of modalities. Through these approaches, our project aims to improve the model's ability to attend to the most relevant image regions for a given question, thereby enhancing overall accuracy and interpretability.

2. Related Work

The task of image captioning is highly related to Visual Question Answering (VQA). The authors extract high-level image feature vectors using GoogleNet, which are then fed into an LSTM to generate captions. Later, extended this approach by incorporating an attention mechanism into the caption generation process. While both image captioning and VQA require understanding of visual content and natural language, the key difference in VQA is that a specific question is provided, and the system must infer the answer based on a detailed understanding of the image.

Several works have explored the intersection of vision and language for VQA. Earlier approaches, such as, combine a CNN for image feature extraction with an LSTM for question encoding—conditioning the LSTM on the CNN features at each time step and using the final hidden state to decode the answer. In contrast, the method employs a "late fusion" strategy: image and question embeddings are computed independently and then fused via element-wise multiplication before being passed through a Multi-Layer Perceptron to generate a probability distribution over answer classes.

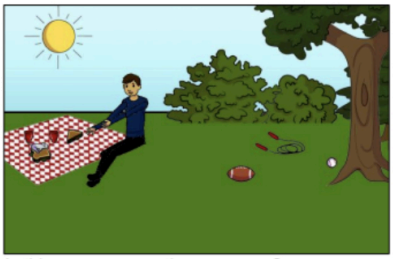
Other approaches include encoder-decoder architectures, where an LSTM encodes both the image and question and a separate LSTM decodes the answer, with image features fed into every LSTM cell. Additionally, techniques such as those in have utilized CNNs for question modeling and convolutional operators to merge question and image feature vectors.

3. Dataset & Features

The project utilizes the latest version of a standard Visual Question Answering dataset - Visual VQA dataset v2.0. This release consists of 82,783 MS COCO training images, 40,504 MS COCO validation images and 81,434 MS COCO testing images (images are obtained from the MS COCO website). Along with the images, this release also has 443,757 questions for training, 214,354 questions for validation and 447,793 questions for testing. This dataset comprises of 4,437,570 answers for training and 2,143,540 answers for validation (approximately 10 responses per question).

What color are her eyes?
What is the mustache made of?

How many slices of pizza are there?
Is this a vegetarian pizza?



Is this person expecting company?
What is just under the tree?

The questions and answers within the dataset are collected by the authors of the VQA paper which we use as a baseline. The authors chose MS COCO images to elicit a more interesting set of questions and answers since the MS COCO images have a lot of diversity and richness. The authors used a user interface called Amazon Mechanical Turk to collect the questions and the answers. To make sure that the users who frame the questions create a set of interesting and complex questions, they told the users that the questions have to fool a smart robot. The authors did this to make sure there weren't any simple yes/no questions or questions which could be answered easily and weren't dependent on the image such as "What is the color of the banana?". In all, they collect 3 unique questions per image from each user.

For the answers, they use the same user interface. They collect 10 answers from each user and ask for short phrases or single words as answers. The collected answers for a particular question might have multiple possible correct answers such as "red", "maroon", "dark orange". Even for yes/no questions, both the 'yes' or 'no' might be correct. The authors choose the most common answer as the correct answer.

From our analysis of the dataset, most questions comprise of around 4 to 10 words. Most of the answers (89.32%) are one word answers and almost all answers are at most 3 word answers.

4. Methodology

Our project implements the Visual Question Answering (VQA) task using four distinct deep learning architectures. Each model progressively incorporates more sophisticated components for visual feature extraction, textual encoding, and cross-modal attention. The following subsections detail the methodology of each architecture.

4.1 Baseline: ResNet50 + LSTM

Image Processing:

- **Feature Extraction:** A pretrained ResNet50 is used as the backbone. The final two layers (classification head) are removed to retain only the feature extraction layers.
- **Spatial Encoding:** An adaptive average pooling layer resizes the extracted feature maps to a fixed size. For example, after the CNN, the features with shape *(B, 2048, H, W)* are pooled to produce a spatial size of (2×2), resulting in a flattened feature vector of size 8192 per image.
- **Dimensionality Reduction:** A sequence of fully connected layers gradually reduces the dimensionality:
    - 8192 → 4096 (with BatchNorm, ReLU, and Dropout 0.3)
    - 4096 → 2048
    - 2048 → 1024

Question Processing:

- **Embedding:** Questions are tokenized and fed into an embedding layer (e.g., with an embedding dimension of 512).
- **Sequential Modeling:** A three-layer bidirectional LSTM with hidden dimension 512 (per direction) processes the embedded sequence. Dropout of 0.3 is applied between layers.
- **Feature Fusion:** The last hidden states from the forward and backward passes are concatenated (resulting in a 1024-dimensional vector) and further processed through a fully connected layer (with BatchNorm, ReLU, and Dropout 0.5) to produce a final question representation of 1024 dimensions.

Fusion and Prediction:

- **Concatenation:** The 1024-dimensional image feature and the 1024-dimensional question feature are concatenated to form a 2048-dimensional vector.
- **Joint Processing:** This combined feature is processed by an additional fully connected layer (reducing it to 1024 dimensions) with BatchNorm, ReLU, and Dropout.
- **Output:** A final linear layer maps the 1024-dimensional joint feature to the answer vocabulary size, producing the answer prediction.

4.2 Multi-Layer Attention with LSTM

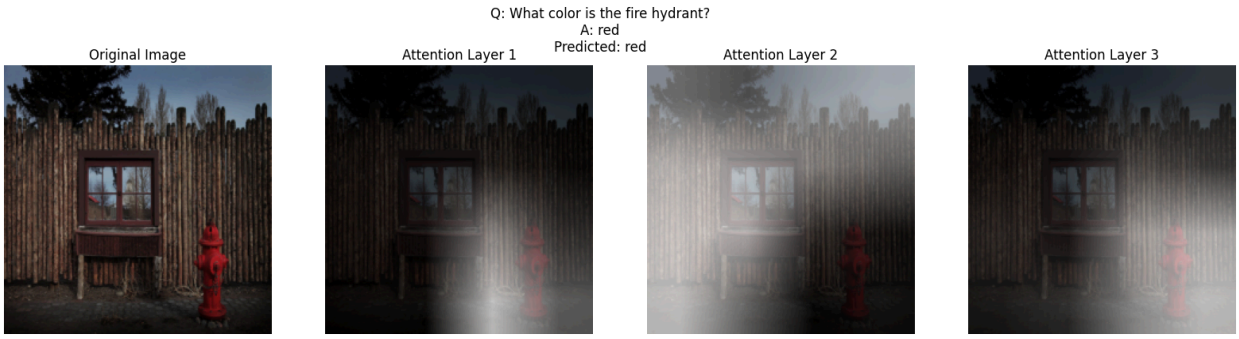
Image Processing:

- The image encoder is similar to the baseline but uses an adaptive average pooling layer that outputs a (4×4) spatial grid. This produces feature maps of shape (B, 2048, 4, 4).

Question Processing:

- A slightly modified question encoder is employed: it uses an embedding dimension (e.g., 768) and a three-layer bidirectional LSTM with hidden dimension 512 per direction, yielding a final representation of 2048 dimensions after concatenation and processing through a fully connected layer.

Multi-Layer Attention Module:



Projection:

- The 2048-channel image feature map is projected using a 1×1 convolution to a hidden dimension of 1024. After pooling, the feature map is reshaped to (B, 16, 1024) (since 4×4 = 16).
- The question feature (2048 dimensions) is projected via a linear layer to match the 1024-dimensional hidden space and then unsqueezed to shape (B, 1, 1024).

Attention Mechanism:

- A series of 3 attention layers are applied. In each layer:
  - The question representation is broadcast to all 16 spatial locations.
  - The image features and the broadcast question features are summed and normalized using LayerNorm.
  - A linear layer with output size 1 is applied on a tanh activation to generate attention scores for each spatial location.
  - Softmax is used to compute attention weights, which are then used to perform a weighted sum of the image features.
  - The resulting summary vector (1024 dimensions) is broadcast back to 16 locations for the next attention layer.
- The final output is a refined attended image feature vector of size 1024.

Fusion and Prediction:

- Concatenation: The attended image feature (1024) is concatenated with the 2048-dimensional question feature, forming a 3072-dimensional vector.

- Fusion Layer: A fully connected layer reduces this to 1024 dimensions (with BatchNorm, ReLU, and Dropout 0.4) before the final classification layer maps it to the answer vocabulary.

4.3 Multi-Layer Attention with BERT

Image Processing:

- The image encoder remains similar to the multi-layer attention approach, using ResNet50 with adaptive pooling to obtain a (4×4) grid and projecting the features to a 1024-dimensional hidden space.

Question Processing with BERT:

- BERT Encoder: Instead of an LSTM, a pretrained BERT model (e.g., bert-base-uncased) is used. The input question tokens (with attention masks) are processed by BERT, and the CLS token's 768-dimensional embedding is extracted.
- Refinement: This embedding is passed through a fully connected network:
- Linear layer: 768 → 1024 (with BatchNorm, ReLU, and Dropout 0.4)
- Linear layer: 1024 → 768 (with BatchNorm, ReLU, and Dropout 0.4)
- The final question representation is 768-dimensional, tailored to the attention module.

Attention Module:

- The multi-layer attention module is adjusted to accept the 768-dimensional question feature (projected to the 1024-dimensional hidden space via a linear layer) while keeping the image features processing similar. The attended image feature remains 1024-dimensional.

Fusion and Prediction:

- **Concatenation:** The attended image feature (1024) and the BERT-based question feature (768) are concatenated to form a 1792-dimensional vector.
- **Fusion Layer:** This vector is processed by a fully connected layer (with BatchNorm, ReLU, and Dropout 0.4) to reduce it to 1024 dimensions before the final classification layer produces the answer prediction.

4.4 CNN with Multi-Head Attention and BERT

Image Processing with RCNN:

- **Feature Extraction:** A pretrained Faster R-CNN (based on ResNet50 with a Feature Pyramid Network) is used to extract object-level features. The backbone of Faster R-CNN (i.e., the ResNet50 FPN) produces a feature map with 256 channels.
- **Pooling and Reshaping:**
    - An adaptive average pooling layer resizes the feature map to a (4×4) grid.
    - The pooled feature map is flattened and permuted to yield a tensor of shape *(B, 16, 256)*.
- **Question Processing with BERT:**
    - Similar to the previous architecture, the question is encoded using a pretrained BERT model. The CLS token is extracted and then passed through a fully connected network to obtain a 768-dimensional representation.

Multi-Head Attention Module:

- **Projection:**
  - The 256-dimensional image features are projected to a hidden space of 1024 dimensions using a linear layer, converting the shape from *(B, 16, 256)* to *(B, 16, 1024)*.
  - The 768-dimensional question feature is projected via a linear layer to 1024 dimensions and unsqueezed to form a query tensor of shape *(B, 1, 1024)*.
- **Attention Operation:**
  - A multi-head attention mechanism (with 8 heads) is applied where the question query attends over the projected image features. The PyTorch nn.MultiheadAttention layer is used with an embedding dimension of 1024.
  - The attention outputs are combined via a residual connection and layer normalization, yielding an attended image vector of 1024 dimensions.

Fusion and Prediction:

- **Fusion:**
  - Both the attended image vector and the BERT question vector are projected into a common 512-dimensional space using separate linear layers.
  - An element-wise multiplication is performed on these 512-dimensional vectors to fuse the modalities.
- **Final Processing:**
  - The fused representation is further processed by a fully connected fusion layer (which increases the dimension to 1024, with BatchNorm, ReLU, and Dropout 0.4).
  - Finally, a linear layer maps the 1024-dimensional vector to the answer vocabulary size to produce the final prediction.

5. Results

Our experiments evaluated the four VQA architectures across several question categories. We report accuracy for “Yes/No,” “Number,” and “Other” question types, as well as overall accuracy. The results indicate that as we incorporate more advanced techniques, such as attention mechanisms, contextual language embeddings, and object-level feature extraction, the model performance improves across most categories.

Quantitative Results

The table below summarizes the category-wise accuracy for each of our models:

Model Architecture	Yes/No (%)	Number (%)	Other (%)	Overall Accuracy (%)
Baseline: ResNet50 + LSTM	55.64	24.74	31.74	39.75
Multi-Layer Attention (with LSTM)	58.72	26.44	35.56	43.01
Multi-Layer Attention + BERT	60.72	27.40	37.49	44.48
RCNN + Multi-Head Attention + BERT	62.91	29.76	36.86	45.66

Analysis of Results

Baseline Model (ResNet50 + LSTM):

- **Performance**: The baseline model achieves moderate accuracy, with the highest performance on “Yes/No” questions.
- **Observations**:
    - Simple concatenation of image and question features may limit the model’s ability to focus on the relevant parts of the image.
    - The LSTM-based question encoder, while effective for sequential data, may not fully capture the nuances in language required for more complex “Other” type questions.

Multi-Layer Attention with LSTM:

- **Performance**: Adding a multi-layer attention mechanism improves the overall accuracy by approximately 2–3%.
- **Observations**:
    - Attention helps the model better align visual features with the question context, improving responses in all categories.
    - Improvement in “Other” questions suggests that spatially attending to different image regions helps in understanding complex scene details.

Multi-Layer Attention + BERT:

- **Performance**: Replacing the LSTM with a BERT-based question encoder further boosts accuracy by leveraging contextual embeddings.
- **Observations**:
    - BERT provides richer semantic representations that enhance the model’s understanding of the question.
    - The improvements are especially notable in “Number” and “Other” questions, where context is critical.

RCNN + Multi-Head Attention + BERT:

- **Performance**: The most advanced model achieves the highest overall accuracy, with notable gains in all question categories.
- **Observations**:
    - Utilizing a Faster R-CNN backbone enables the model to extract fine-grained, object-level features, which are crucial for questions requiring detailed visual reasoning.
    - The multi-head attention mechanism allows the model to focus on different regions simultaneously, capturing a broader set of visual cues.
    - Combining these with BERT’s powerful language representation leads to a significant performance boost.

Discussion and Future Improvements

- **Differences in Accuracy:**
    - The observed improvements can be attributed to more sophisticated feature extraction and fusion strategies. For instance, the shift from a simple LSTM to BERT for question encoding enhances language understanding, while the progression from simple concatenation to advanced attention mechanisms helps in better aligning visual and textual modalities.
    - The RCNN-based approach's ability to isolate object-level features and apply multi-head attention further improves performance, particularly for complex questions where detailed image understanding is essential.
- **Potential Improvements:**
    - **Enhanced Vision Models:** Experimenting with transformer-based vision models (e.g., Vision Transformers) could further improve object detection and feature representation.
    - **Fine-Tuning Strategies:** More extensive fine-tuning of both the image and language encoders, including layer-wise learning rate adjustments, may yield better results.
    - **Multi-Modal Fusion Techniques:** Exploring more sophisticated fusion techniques (e.g., bilinear pooling or cross-modal transformers) could further enhance the integration of visual and textual features.
    - **Data Augmentation:** Augmenting the dataset with more diverse examples or using synthetic data might help the model generalize better across question types.
    - **Ensemble Methods:** Combining predictions from multiple models might improve robustness and overall accuracy.

Overall, the results demonstrate that integrating attention mechanisms and leveraging state-of-the-art encoders significantly enhance the model's capability in Visual Question Answering, paving the way for further research and refinement in multi-modal deep learning approaches

6. Conclusion

In this project, we explored the challenging task of Visual Question Answering by implementing four distinct architectures that progressively integrate state-of-the-art techniques from both computer vision and natural language processing. Our journey began with a baseline model that combined ResNet50 for image feature extraction with an LSTM for question encoding. This initial approach provided a solid foundation, establishing key performance metrics across various question categories.

Building on this, we introduced a multi-layer attention mechanism, which allowed the model to better align visual features with the nuances of the question. This adjustment not only improved the overall accuracy but also enhanced the model's ability to handle complex queries by focusing on the most relevant image regions.

The subsequent integration of BERT into the attention framework further enriched the question representations. By leveraging BERT’s contextualized embeddings, the model achieved noticeable gains in understanding and reasoning over the questions, particularly in categories that demand a deeper semantic comprehension. This enhancement demonstrated the critical role of advanced language models in multi-modal tasks.

Finally, our most advanced architecture, which combined a Faster R-CNN backbone with multi-head attention and BERT, showcased the benefits of extracting fine-grained, object-level features. The multi-head attention mechanism allowed the model to simultaneously focus on different parts of the image, leading to significant improvements in answer prediction. The success of this model underscores the importance of a well-integrated fusion strategy for visual and textual modalities.

Overall, our experiments highlight that as the complexity of the architecture increases—from a simple concatenation model to advanced multi-modal fusion techniques—so does the model’s performance on the VQA task. Future work could explore transformer-based vision models, more sophisticated fusion techniques, and enhanced fine-tuning strategies to further bridge the gap between visual perception and language understanding, ultimately pushing the boundaries of what can be achieved in Visual Question Answering.

Team Contributions

Bhavyadeep Singh Hada – Baseline Model (ResNet50 + LSTM)

- **Responsibilities:**
  - Developed the initial VQA model using ResNet50 for image feature extraction and an LSTM for question encoding.
  - Implemented pre-processing, feature extraction, and dimensionality reduction pipelines.
  - Conducted baseline experiments across different question categories.
- **Highlights:**
  - Established a robust baseline for subsequent improvements.

Ujjwal Jain – Multi-Layer Attention with LSTM

- **Responsibilities:**
  - Designed and implemented a multi-layer attention mechanism to better align visual and textual features.
  - Extended the baseline by integrating attention layers that focus on key image regions.
  - Evaluated category-wise performance improvements.
- **Highlights:**
  - Demonstrated the benefits of spatial attention in bridging visual and text modalities.

Dev Pandya – Multi-Layer Attention + BERT

- **Responsibilities:**
  - Replaced the LSTM question encoder with a BERT-based encoder for richer language embeddings.
  - Integrated BERT into the attention framework, adjusting for different embedding dimensions.
  - Fine-tuned hyperparameters and evaluated performance across question types.
- **Highlights:**
  - Enhanced language understanding, leading to improved overall model performance.

Akshat Jain – RCNN + Multi-Head Attention + BERT

- **Responsibilities:**
  - Developed the advanced architecture using a Faster R-CNN backbone for object-level feature extraction.
  - Implemented a multi-head attention mechanism to focus on multiple image regions concurrently.
  - Integrated BERT for refined question processing and conducted fusion experiments.
- **Highlights:**
  - Achieved the highest accuracy gains by leveraging fine-grained visual features and multi-modal attention.