University Abdelmalek Essaâdi
National School of Applied Sciences of Tétouan

Second Year of the Engineering Cycle – BDIA
Module: Deep Learning

Visual Question Answering

Authors:
BOUHAFA TAHA
LOUBABA MALKI L'HLAIBI

Supervisor:
Prof. BELCAID ANASS

Academic Year: 2024/2025

Contents

I. Acknowledgments

We take this opportunity to express our heartfelt gratitude to all those who contributed to the successful completion of this project. Your guidance, support and encouragement were invaluable throughout this journey.

First, we extend our deepest thanks to Prof.BELCAID ANASS, our project guide, for their constant support, insightful feedback, and expert guidance. Their deep knowledge in the field of deep learning helped us navigate the complexities of this project and refine our approach. Their patience and encouragement kept us motivated, even during the challenging phases of the project.

We would like to acknowledge the creators of the COCO VQA V2.0 dataset for making their data publicly available. This dataset was instrumental in training and evaluating our model, and we appreciate the effort that went into its creation.

Our sincere thanks go to the open-source community for providing access to powerful tools and libraries such as PyTorch, Hugging Face Transformers, and Matplotlib, which were essential for the implementation and visualization of our project. Without these resources, this project would not have been possible.

We also thank our colleagues and peers for their constructive feedback, discussions, and moral support during the project. Their input helped us refine our ideas and overcome challenges.

Lastly, we express our gratitude to our family and friends for their unwavering support, encouragement, and patience throughout this journey. Their belief in us kept us going, even during the most demanding times.

This project has been a rewarding learning experience, and we are grateful to everyone who contributed to its success.

II. Introduction

Visual Question Answering (VQA) is a challenging task in the field of artificial intelligence that combines computer vision and natural language processing. The goal of VQA is to enable machines to answer questions about images in a way that is both accurate and contextually relevant. This project focuses on building a VQA system that leverages state-of-the-art deep learning models to achieve this goal.

The system integrates two key components: a Convolutional Neural Network (CNN) for image feature extraction and a Bidirectional Encoder Representations from Transformers (BERT) model for processing textual questions. By combining these features, the model is trained to predict the most appropriate answer from a predefined vocabulary. The project involves several stages, including data preprocessing, model training, validation, and testing, with the ultimate aim of achieving high accuracy in answering questions based on visual content.

This report documents the methodology, implementation, and evaluation of the VQA system, highlighting the challenges faced and the solutions adopted. The results demonstrate the effectiveness of combining visual and textual features for the VQA task, providing insights into the potential of multimodal learning in AI applications.

III. Literature

III.1 Definition

Visual Question Answering (VQA) is an interdisciplinary field that combines computer vision and natural language processing to enable machines to answer natural language questions about visual content. This capability demands a deep integration of image understanding, language comprehension, and reasoning processes. A VQA system typically involves three key stages:

- **Image Processing:** Utilizing computer vision techniques such as convolutional neural networks (CNNs) to extract meaningful visual features from images.
- **Question Understanding:** Parsing and embedding natural language questions using methods ranging from traditional natural language processing (NLP) techniques to advanced transformer-based models.
- **Multimodal Fusion and Reasoning:** Combining image and question representations through mechanisms like attention networks or multimodal transformers to generate relevant answers.

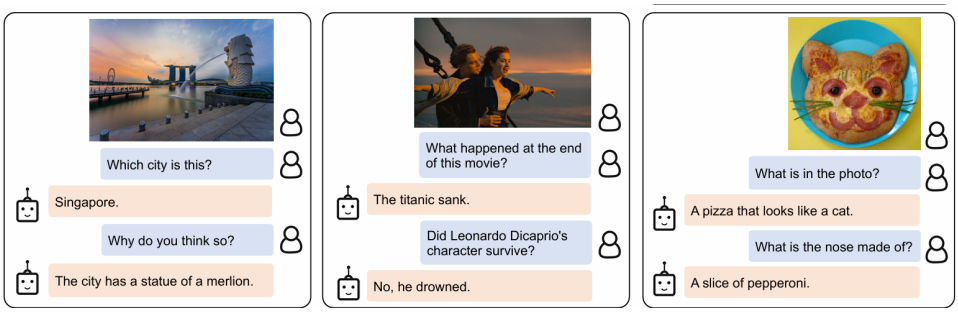

Figure III.1: **Figure 1:** Examples of questions and answers in VQA tasks, showcasing different question types.

To illustrate how VQA aligns with common computer vision tasks, Table III.1 provides examples of representative VQA questions associated with specific CV tasks.

CV Task	**Representative VQA Question**
Object recognition	What is in the image?
Object detection	Are there any dogs in the picture?
Attribute classification	What color is the umbrella?
Scene classification	Is it raining?
Counting	How many people are there in the image?
Activity recognition	Is the child crying?
Spatial relationships among objects	What is between the cat and the sofa?
Commonsense reasoning	Does this person have 20/20 vision?
Knowledge-base reasoning	Is this a vegetarian pizza?

Table III.1: Representative VQA Questions for Common CV Tasks

III.2 Datasets

Datasets are fundamental to the development and evaluation of Visual Question Answering (VQA) systems. Below are some of the most commonly used datasets in the field:

- **COCO VQA V2.0 2017 Dataset:** Visual Question Answering (VQA) v2.0 is a dataset containing open-ended questions about images. These questions require an understanding of vision, language, and commonsense knowledge to answer. It is the second version of the VQA dataset.
- **CLEVR:**A synthetic dataset aimed at testing a model's compositional and logical reasoning skills.Questions involve reasoning about shapes, colors,sizes,and spatial

Images	Questions	Annotations	Questions per Image	Split (Training/Testing)
250,000+	1M+	7M+	5.4 questions average	82,000+ / 80,000+

Table III.2: **S**tatistics of the COCO VQA Dataset

relationships in 3D-rendered scenes.

Category	Images	QA pairs	Split(Training/testing)
CLEVR	70.000+	700.000+	60.000+/10.000+

Table III.3: **S**tatistics of CLEVR Dataset

- **Visual7W:** Visual7W is a large-scale visual question answering (VQA) dataset that includes object-level groundings and multimodal answers. Each question in the dataset begins with one of the seven Ws (what, where, when, who, why, how, and which), designed to test a model's understanding of various types of inquiries.

Images	QA pairs	Object Groundings	Objects	Split (Training/Testing)
47.000+	327.000+	561.000+	36.000+	42.000+ / 5.000+

Table III.4: **S**tatistics of the Visual7W Dataset

III.3 State-of-the-Art Algorithms

Recent advances in deep learning have driven the development of state-of-the-art algorithms for VQA. These algorithms integrate visual and textual modalities using sophisticated techniques:

- **Transformer Models:** Advanced multimodal transformers such as BeiT-3 and VLMO are capable of jointly learning visual and textual representations. These models, like ViLBERT and LXMERT, allow the VQA system to effectively understand the interplay between images and questions, improving accuracy.

- **Neuro-Symbolic Methods:** Models such as One-Peace combine neural networks with symbolic reasoning. This hybrid approach strengthens the model's ability to perform compositional and logical reasoning, making it particularly effective for tasks that involve complex question structures.

- **Large Language Models:** Models like BLIP-2, Flamingo, and OFA, adapted from large language models like CLIP and GPT, leverage extensive pretraining on vast multimodal datasets. This enables them to achieve enhanced reasoning capabilities and handle more diverse VQA tasks.

Model Name	**Test-Dev**	**Test-std**
PaLI	84.30	84.30
BeiT-3	84.20	84.00
VLMO	82.88	82.78
One-Peace	82.60	82.50
BLIP-2	82.30	82.30
Flamingo	82.00	82.10
OFA	82.00	82.00

Table III.5: Performance Analysis of VLP Architectures in VQA. The models are evaluated on the test-dev and test-std splits of the VQAv2 dataset.

Source: *From Image to Language: A Critical Analysis of Visual Question Answering (VQA) Approaches, Challenges, and Opportunities (2024)*

IV. Datasets

IV.1 Dataset Overview

The COCO-VQA dataset is one of the most widely used datasets in Visual Question Answering (VQA) research. It contains open-ended questions about images, requiring an understanding of vision, language, and commonsense knowledge to answer. The dataset is based on the MS COCO image dataset and includes two JSON files: one for annotations and one for questions, along with the corresponding images.

Dataset: https://visualqa.org/download.html

IV.1.1 Key Statistics

- **Total Images**: 250,000+
- **Total Questions**: 1 million+
- **Total Answers**: 7 million+
- **Questions per Image**: 5.4 (average)
- **Split**: 82,000+ training images, 80,000+ testing images

IV.1.2 Data Structure:

The dataset consists of:

- **Annotations File**: Contains the answers for each question, with multiple answers provided by different annotators.
- **Questions File**: Contains the questions and their corresponding image IDs.
- **Images**: Each image is associated with multiple questions and answers.

IV.1.3 Examples:

Below are examples of the data structure in the JSON files and an example image from the dataset.

Annotations File Example:

The annotations file contains detailed information about each question, including the multiple-choice answer and a list of answers provided by different annotators. Here is an example:


```
{
    "question_type": "what is",
    "multiple_choice_answer": "sharpening knife",
    "answers": [
        {"answer": "sewing", "answer_confidence": "no", "answer_id": 1},
        {"answer": "sharpening knife", "answer_confidence": "yes", "answer_id": 2},
        {"answer": "grinding knife", "answer_confidence": "yes", "answer_id": 3},
        {"answer": "sharpening knife", "answer_confidence": "yes", "answer_id": 4},
        {"answer": "riding", "answer_confidence": "maybe", "answer_id": 5},
        {"answer": "knife sharpening", "answer_confidence": "yes", "answer_id": 6},
        {"answer": "sharpening knife", "answer_confidence": "maybe", "answer_id": 7},
        {"answer": "sharpening knives", "answer_confidence": "maybe", "answer_id":8},
        {"answer": "sharpening knife", "answer_confidence": "yes", "answer_id": 9},
        {"answer": "knife sharpening", "answer_confidence": "yes", "answer_id": 10}
    ],
    "image_id": 262136,
    "answer_type": "other",
    "question_id": 262136003
}
```

Questions File Example:

The questions file contains the questions and their corresponding image IDs. Here is an example:

```
{
    "image_id": 524286,
    "question": "Is there a computer mouse on the desk?",
    "question_id": 524286002
}
```

Image Example:

An example image from the dataset is shown below:

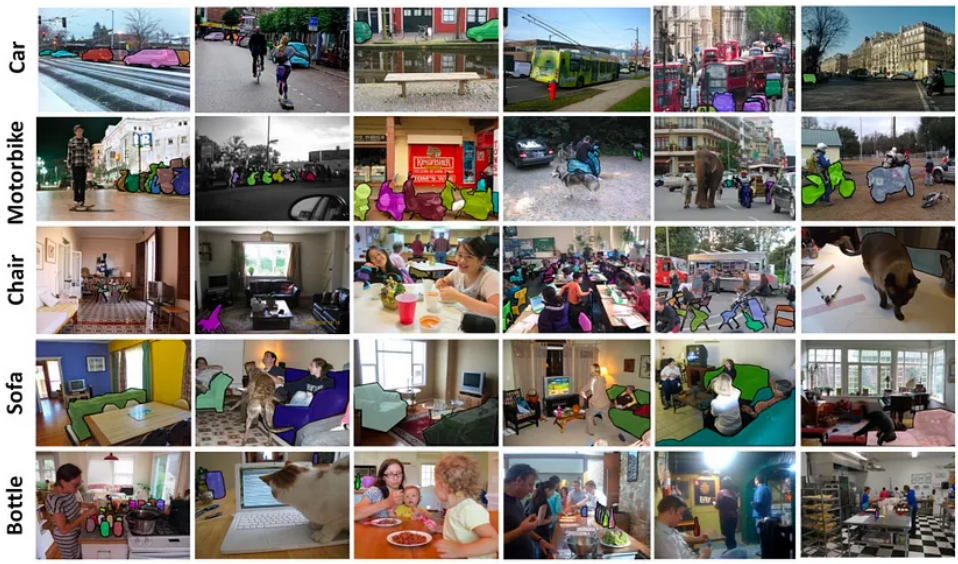

Figure IV.1: Example image from the COCO-VQA dataset.

IV.1.4 Why COCO-VQA?

COCO-VQA was chosen for our project due to its large size, diversity of questions, and real-world applicability. It provides a comprehensive benchmark for evaluating VQA models and is widely used in the research community.

IV.2 Preprocessing:

To make the dataset manageable for our project, we preprocessed the data and split it into smaller subsets.

IV.2.1 Data Preprocessing:

The preprocessing steps included:

Building the Vocabulary:

We created a vocabulary of the **top 1000 most frequent answers** from the training set. An additional `<unk>` token was added for unseen answers. This vocabulary allowed us to map answers to numerical indices for training.

Mapping Answers to Indices:

Each answer was mapped to its corresponding index in the vocabulary. If an answer wasn't in the top 1000, it was assigned the `<unk>` index. This step converted the textual answers into numerical form for the model.

Tokenizing Questions:

We used the BERT tokenizer (`bert-base-uncased`) to tokenize the questions. Each question was converted into:

- **Input IDs**: Token IDs representing the question.
- **Attention Mask**: A mask to handle padding tokens.

Handling Multiple Answers:

For questions with multiple answers, we selected the **majority answer** as the target label. If there was no majority, the most common answer was chosen. This ensured consistency in the training data.

Image Preprocessing:

The images were preprocessed as follows:

- **Resizing**: All images were resized to 224x224 pixels to ensure uniformity.
- **Normalization**: The images were normalized using the mean and standard deviation of the ImageNet dataset. This step improved the model's convergence during training.

IV.2.2 Dataset Splitting:

Since the COCO-VQA dataset is very large, we split the training set into smaller subsets for our project:

- **Training Set**: 20% of the original training set.
- **Validation Set**: 5% of the original training set.
- **Test Set**: 5% of the original training set.

This splitting strategy allowed us to train and evaluate our model efficiently while ensuring that we had enough data for each phase of the project.

V. Architecture

V.1 Vision Model (Resnet Model)

V.1.1 ResNet34

ResNet34 is a convolutional neural network architecture that belongs to the ResNet family of models, introduced by He et al. in 2015. It was specifically designed to address the problem of vanishing gradients in deep networks by utilizing residual connections, which allow gradients to flow more easily through deeper layers.

The ResNet34 model consists of 34 layers, including convolutional layers, batch normalization, and activation functions. The key feature of ResNet34 is the use of *residual blocks*, which consist of skip connections that bypass one or more layers. These skip connections allow the model to learn residual mappings rather than trying to directly learn the underlying mapping, helping the network to train deeper architectures effectively.

The architecture of ResNet34 can be broken down as follows:

- **Input:** The input is an image with a size of 224×224 pixels, typically with 3 color channels (RGB).
- **Initial Convolution:** The first layer of ResNet34 is a convolutional layer with a kernel size of 7×7 and a stride of 2, followed by batch normalization and a ReLU activation function.
- **Residual Blocks:** ResNet34 is built with 34 layers, where the majority of layers are grouped into four residual blocks. Each block consists of two or more convolutional layers, followed by batch normalization and ReLU activation. The output of each block is added to the input of the block via a skip connection.

- **Downsampling:** To reduce the spatial dimensions of the feature maps, ResNet34 uses downsampling in the first block and at each transition between blocks. This is typically achieved with a 2×2 average pooling layer or a stride of 2 in convolution layers.

- **Fully Connected Layer:** After the residual blocks, the feature maps are passed through a global average pooling layer, which reduces the spatial dimensions to a single value per channel. This output is then flattened and passed through a fully connected (dense) layer, which outputs the final predictions.

- **Output:** The final layer outputs a vector of size 1001, corresponding to the 1000 class labels in the ImageNet dataset, plus an additional class for background or unused categories.

The key innovation of ResNet34 lies in the residual connections, which allow the model to be much deeper (34 layers in this case) without suffering from the vanishing gradient problem. This makes ResNet34 capable of achieving high performance on image classification tasks, such as the ImageNet challenge.

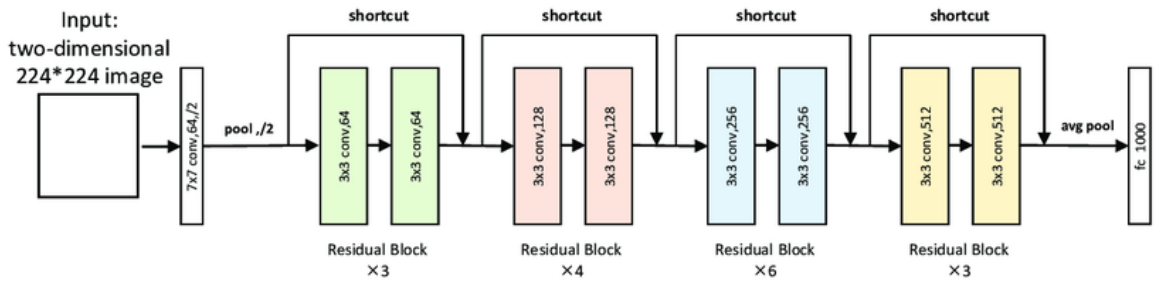

Figure V.1: ResNet34 Architecture (1000 classes)
Source: ResearchGate

ResNet34 was chosen for our Visual Question Answering (VQA) model due to its balance of depth, computational efficiency, and strong feature extraction capabilities. This architecture, with its residual learning framework, is particularly well-suited for tasks that require robust visual feature representation.

The decision to use 1001 classes in our model corresponds to the number of answer in our answers vocabulary.

- The ResNet-34 model was trained on our dataset with the following configurations:

Hyperparameters	**Values**
Input Image Size	224x224
Labels (Number of Vocabulary)	1001
Number of Epochs	8
Learning Rate	1e-3
Batch Size	16
Optimizer	AdamW

Table V.1: Training Configurations for ResNet-34

- **Result:** The model achieved a validation accuracy of **20.41%** after training.

V.1.2 ResNet50

ResNet50 is an advanced variant of the ResNet architecture, consisting of 50 layers. It builds upon the foundational principles of ResNet34 but incorporates greater depth and additional optimizations to enhance feature extraction and model accuracy. This deeper architecture is particularly advantageous for complex tasks like Visual Question Answering (VQA), where detailed image understanding is critical.

The architecture of ResNet50 can be broken down as follows:

- **Input Layer:** Similar to ResNet34, the input layer processes images of size $224 \times 224 \times 3$ (height, width, channels).
- **Initial Convolution:** The architecture starts with a 7×7 convolutional layer with a stride of 2, followed by batch normalization and ReLU activation. This is coupled with a 3×3 max-pooling layer to downsample the input.
- **Residual Blocks:** ResNet50 employs *bottleneck blocks*, which differ from the basic blocks used in ResNet34. Each bottleneck block consists of three convolutional layers:
  - A 1×1 convolution reduces the dimensionality.
  - A 3×3 convolution processes the reduced-dimensionality features.
  - Another 1×1 convolution restores the original dimensions.

These bottleneck blocks allow for a deeper network while maintaining computational efficiency.

- **Depth:** ResNet50 consists of 16 residual blocks, each containing 3 convolutional layers, resulting in 48 convolutional layers and 2 additional layers (1 initial convolution and 1 fully connected layer), making a total of 50 layers.

- **Global Average Pooling and Fully Connected Layer:** The feature maps from the last residual block are passed through a global average pooling layer. The resulting feature vector is fed into a fully connected layer that outputs 1001 classes, corresponding to the answer space of the VQA model.

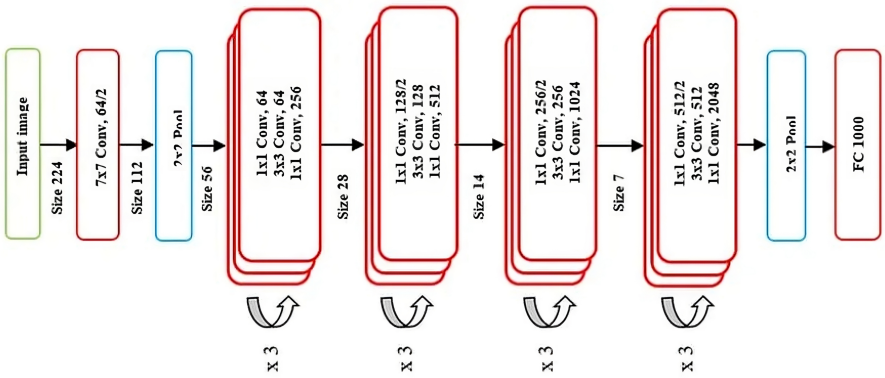

Figure V.2: ResNet50 Architecture
Source: ResearchGate

The additional depth and improved feature extraction capabilities of ResNet50 enhance the ability of the VQA model to handle more complex and nuanced questions. By integrating the image features extracted by ResNet50 with the textual features from the question, the model is likely to achieve a more comprehensive understanding of the multimodal input, resulting in better performance across a variety of VQA tasks.

Similarly to ResNet34, the decision to use 1001 classes in our model corresponds to the number of answer in our answers vocabulary.

Hyperparameters	Values
Input Image Size	224x224
Labels (Number of Vocabulary)	1001
Number of Epochs	5
Learning Rate	1e-3
Batch Size	16
Optimizer	AdamW

Table V.2: Training Configurations for ResNet-50

- **Result:** The best model achieved had a validation accuracy of **19.34%** after training.

V.2 Text Model (BERT)

The text component of our VQA model is powered by BERT, which processes natural language questions and converts them into embeddings for answer prediction. BERT's role is crucial for understanding the questions and generating features that can be fused with visual features from the vision model.

Role of BERT in the VQA Pipeline

BERT is responsible for interpreting the textual questions in the VQA pipeline. By leveraging its pretraining on large language corpora, BERT provides robust representations of the questions, which are integrated with image embeddings using a late fusion mechanism.

Using `BertForSequenceClassification`

For this task, we used `BertForSequenceClassification`, a variant of BERT with an added classification layer to predict answers. This architecture, comprising 12 transformer blocks, allows the model to understand contextual relationships in the input question.

Components of the BERT Model

The BERT model consists of the following key components:

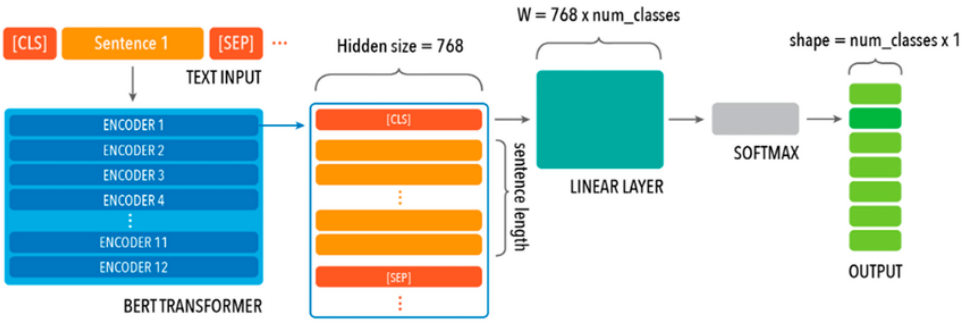

Figure V.3: BERT Architecture: 12 transformer blocks and a classification layer.
Source: ResearchGate

1. Input Tokenization

- The input text (e.g., a question) is tokenized into subwords or words using the BERT tokenizer.
- Special tokens like `[CLS]` (start of sequence) and `[SEP]` (separator for multiple sentences) are added.
- Padding tokens (`[PAD]`) are used to ensure all inputs have the same length.

2. Transformer Encoder Blocks BERT's core consists of 12 transformer encoder blocks (in the base model). Each block has the following components:

- **Multi-Head Self-Attention**: Captures relationships between words in the input sequence by computing attention scores. This allows BERT to understand the context of each word bidirectionally.
- **Feed-Forward Neural Network**: A two-layer fully connected network processes the output of the attention mechanism.
- **Layer Normalization and Residual Connections**: These stabilize training and improve gradient flow.

3. Output Representation

- The output of BERT includes embeddings for each token in the input sequence.
- The `[CLS]` token output is used for classification tasks (e.g., predicting the answer in VQA).

4. Classification Layer In `BertForSequenceClassification`, a linear layer is added on top of the `[CLS]` token output to predict the final class (e.g., the answer in VQA).

Input and Output

The input to BERT consists of tokenized questions, while the output is the predicted answer class.

Input:

- Tokenized questions generated by the BERT tokenizer, which provides:
  - **Input IDs**: Numerical representation of tokens.
  - **Attention Mask**: Identifies padding tokens.

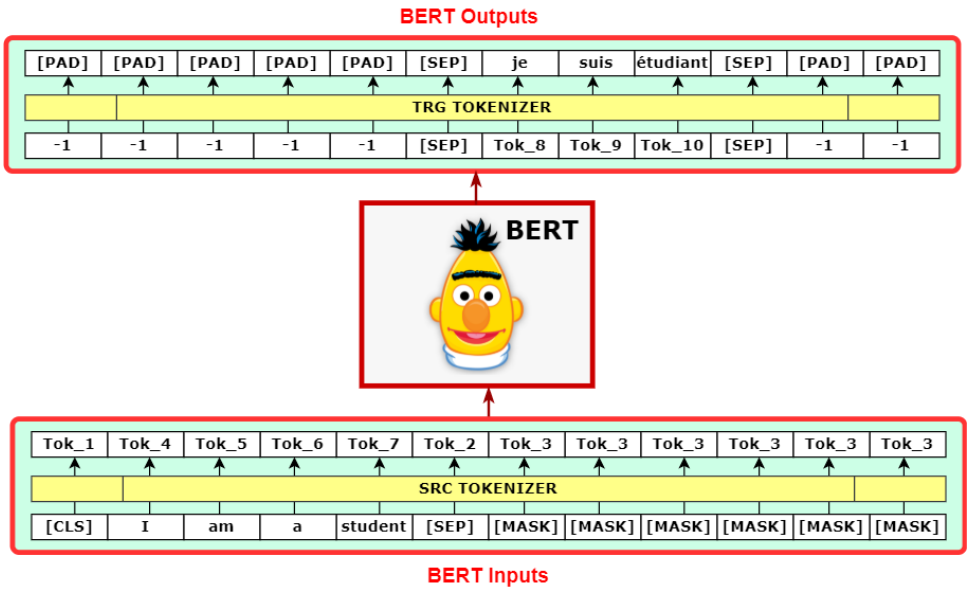

Figure V.4: BERT Tokenizer: Input and output structure.
Source: Yoon's Challenge

Output:

- Predicted class corresponding to one of 1001 possible answers in the vocabulary.

Training Configuration

The BERT model was trained with the following hyperparameters:

Hyperparameter	Value
Input Type	Tokenized Questions
Labels	1001
Number of Epochs	10
Learning Rate	5×10^{-5}
Batch Size	16
Optimizer	AdamW

Table V.3: Training Configuration for BERT Model

Training Results

The model achieved a validation accuracy of **40.48%** after training, demonstrating its effectiveness in processing textual questions for the VQA task.

V.3 Visual Question Answering (VQA) Model

The Visual Question Answering (VQA) model integrates both textual and visual modalities using a multi-modal late fusion mechanism to predict answers based on input images and text. The architecture of our own VQA model further elaborates on this fusion process and introduces specific components for effective feature integration and classification.

V.3.1 Multi-Modal Late Fusion Mechanism

The VQA multi-modal late fusion mechanism, illustrated in Figure V.5, combines features extracted from text and images:

- **Textual Features:** The text input is processed using the BERT model to generate contextual embeddings, referred to as *textual features*.
- **Image Features:** The image input is processed using a convolutional neural network, such as ResNet, to extract visual embeddings, referred to as *image features*.
- **Late Fusion:** The textual and image features are processed independently and fused at a later stage before being passed to the classifier. This approach ensures that each modality contributes complementary information to the prediction task.

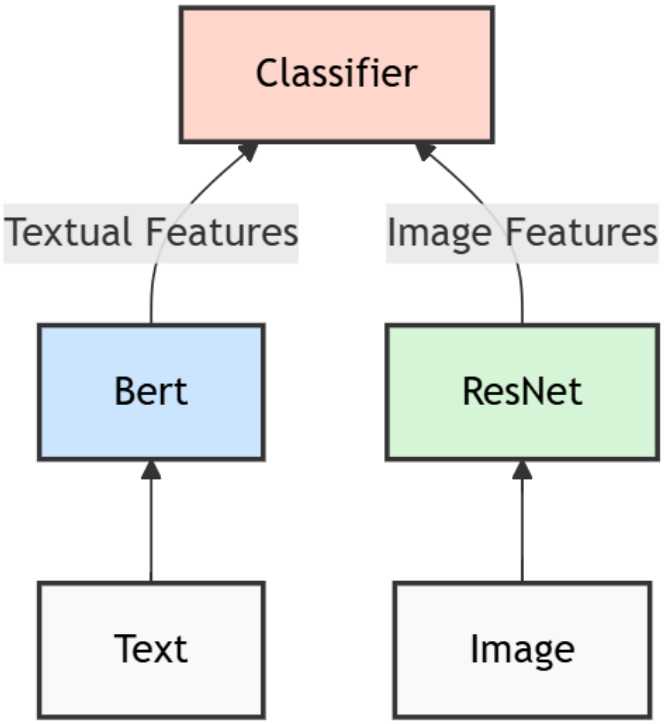

Figure V.5: VQA Multi-Modal Late Fusion Mechanism

V.3.2 Architecture of Our VQA Model

Our VQA model builds upon the multi-modal fusion approach and introduces a specific pipeline for feature integration and classification, as shown in Figure V.6:

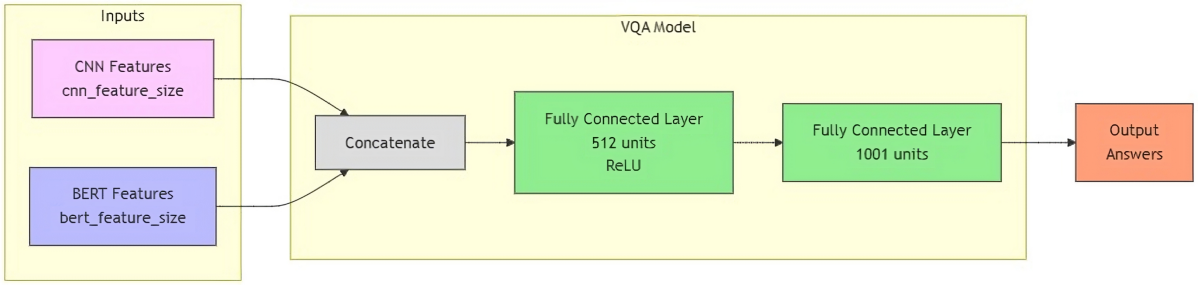

Figure V.6: Architecture of Our VQA Model

- **Input Features:**
  - **BERT Features:** Contextual embeddings extracted from BERT (`bert_feature_size`) after removing its FC layer.

- **ResNet50 Features:** Visual embeddings extracted from ResNet50 (`cnn_feature_size`) after removing its FC layer.

- **Concatenation:** The extracted textual and image features are concatenated to form a unified representation of the inputs.

- **Fully Connected Layers:**
  - The concatenated features are passed through a fully connected layer with 512 units and ReLU activation to transform and combine the features.
  - The output is then passed through another fully connected layer with 1001 units, where each unit corresponds to a potential answer in the dataset.

- **Output:** The model predicts the final answer based on the transformed features.

We trained our VQA Model on our dataset was with the following configurations:

Hyperparameters	**Values**
Input Image Size	224x224 image + question
Output Vocabulary Size	1001
Number of Epochs	10
Learning Rate	5e-5
Batch Size	16
Optimizer	AdamW

Table V.4: Training Configurations for VQA Model

- **Result:** The model achieved a validation accuracy of **43.73%** and a **43.46%** test accuracy.

VI. Evaluation of VQA Model

VI.1 Accuracy and Metrics

This section evaluates the performance of the VQA model using accuracy and other relevant metrics.

VI.1.1 Metrics Used

The following metrics were used to assess the model's performance:

- **Validation Accuracy**: The accuracy achieved on the validation set during training.
- **Test Accuracy**: The final accuracy on the test set, reflecting the model's generalization capability.
- **Training and Validation Loss**: The loss values during training and validation, which indicate how well the model is optimizing.

VI.1.2 Training Configuration

The VQA model was trained with the following hyperparameters:

Hyperparameter	Value
Input Image Size	224x224 image + Question
Output Vocabulary Size	1001
Number of Epochs	10
Learning Rate	5×10^{-5}
Batch Size	16
Optimizer	AdamW

Table VI.1: Training Configuration for VQA Model

VI.1.3 Results

The VQA model achieved the following results:

- **Validation Accuracy**: **43.73%**
- **Test Accuracy**: **43.46%**

VI.1.4 Loss Curves

The training and validation loss curves provide insights into the model's optimization process. The loss values decrease over epochs, indicating that the model is learning effectively.

VI.2 Feature Activation Analysis

This section analyzes the feature activations of the model to understand how well it is capturing and representing the input data.

VI.2.1 Feature Activation Distributions

The feature activation distributions show how the model's internal representations (features) are distributed across different dimensions. This helps in understanding the model's ability to capture meaningful patterns in the data.

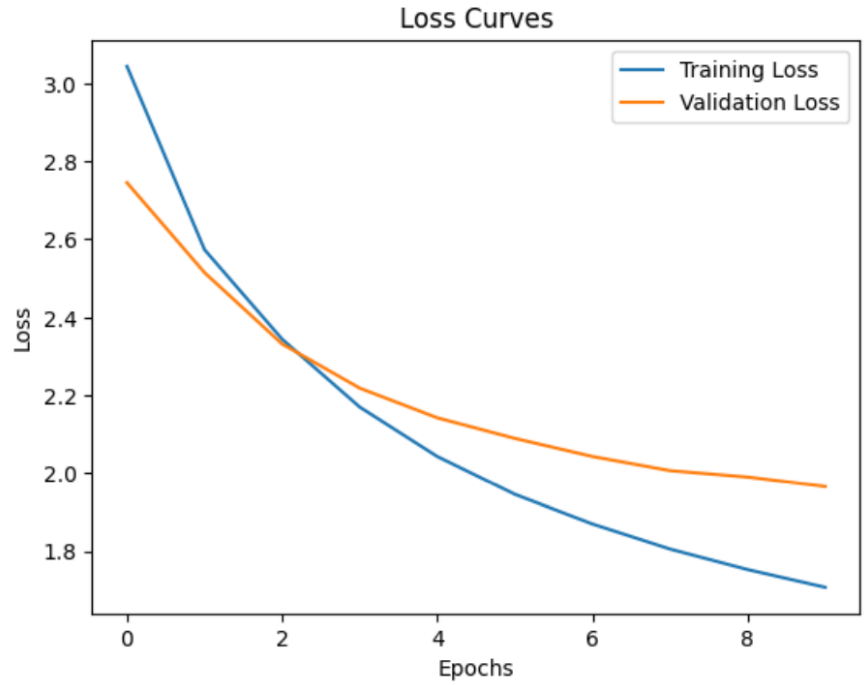

Figure VI.1: Training and Validation Loss Curves

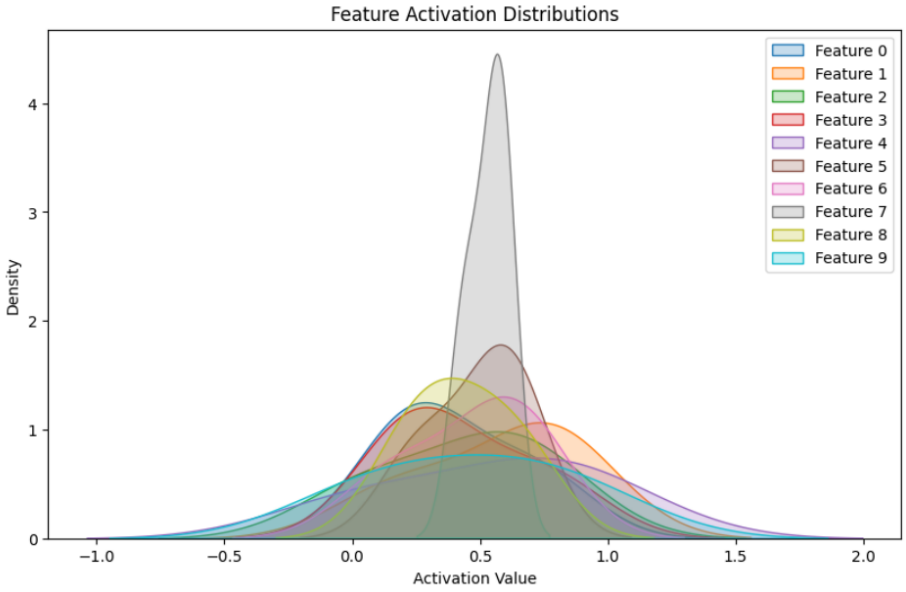

Figure VI.2: Feature Activation Distributions

VI.2.2 Insights

- The feature activation distributions indicate that the model is learning diverse representations, as seen by the varying shapes of the distributions for different feature dimensions.
- Features with broader distributions suggest that the model is capturing a wide range of patterns, while narrower distributions may indicate more specific patterns.

VI.3 Comparison with State-of-the-Art Models

This section compares the performance of our VQA model with state-of-the-art models.

VI.3.1 Comparison Table

The table below compares our model's test accuracy with other state-of-the-art models:

Model Name	**Test Accuracy**
ViLBERT	71.79%
VisualBERT	70.80%
Our Model	**43.46%**

Table VI.2: Comparison of Test Accuracy with State-of-the-Art Models

VI.3.2 Analysis

- **Performance Gap**: Our model achieves a test accuracy of 43.46%, which is significantly lower than ViLBERT (71.79%) and VisualBERT (70.80%).
- **Reasons for Lower Accuracy**:
  - Hardware limitations (e.g., insufficient GPU memory, frequent crashes).
  - Smaller training dataset (only 20% of the full VQA v2.0 dataset was used).
  - Simpler fusion mechanism (late fusion) compared to more advanced techniques used in ViLBERT and VisualBERT.

VI.3.3 Future Improvements

To bridge the performance gap, the following improvements can be considered:

- **Larger Datasets**: Using the full VQA v2.0 dataset or even larger datasets.
- **Advanced Fusion Mechanisms**: Incorporating more sophisticated fusion techniques (e.g., co-attention).
- **Hardware Upgrades**: Training on more powerful GPUs or using distributed training.

VII. Conclusion

In this project, we developed a Visual Question Answering (VQA) model designed to answer natural language questions about images. By combining a vision component (ResNet) and a text component (BERT) using a late fusion mechanism, the model achieved a validation accuracy of **43.73%** and a test accuracy of **43.46%**. While these results demonstrate the model's ability to handle the VQA task, there is still room for improvement compared to state-of-the-art models.

This project highlights the challenges and opportunities in multimodal learning, where vision and language understanding are integrated to solve complex AI tasks. Despite hardware limitations and computational constraints, the work provides a solid foundation for future research in visual question answering and contributes to the broader field of artificial intelligence.