Visual Question Answering

University: University of Verona (UNIVR)
Degree: Master's Degree in Artificial Intelligence
Courses: Computer Vision & Deep Learning - Deep Learning
Academic Year: 2024 – 2025
Authors: Dario Simone – VR512966
Gjeka Silver – VR527645

Date: July, 2025

Contents

1 Motivation **2**

2 State of the Art **2**

3 Objectives **3**

4 Methodology **3**

4.1 Dataset Overview and Analysis . . . 3
4.2 Vision Transformer . . . 4
4.2.1 Architecture . . . 4
4.2.2 Images transformations . . . 5
4.2.3 Loss function . . . 5
4.2.4 Results and plots . . . 5
4.3 ResNet . . . 7
4.3.1 Architecture . . . 7
4.3.2 Results and plots . . . 7
4.4 LSTM . . . 8
4.4.1 Architecture . . . 9
4.5 LSTM with Attention . . . 9
4.5.1 Attention module . . . 9
4.6 1D CNN . . . 10
4.6.1 Architecture . . . 10
4.7 RNN . . . 11
4.7.1 Architecture . . . 11
4.8 Loss function & initialization of model weights . . . 11

5 Experiments & Results **11**

5.1 ViT-LSTM Results . . . 12
5.2 ResNet-LSTM Results . . . 12
5.3 ViT-LSTM with Attention Results . . . 13
5.4 ResNet-LSTM with Attention Results . . . 13
5.5 ViT-1DCNN Results . . . 15
5.6 ResNet-1DCNN Results . . . 15
5.7 ViT-RNN Results . . . 16
5.8 ResNet-RNN Results . . . 17
5.9 Final comparison and considerations . . . 17
5.9.1 ResNet-based Encoder Performance . . . 17
5.9.2 ViT-based Encoder Performance . . . 18
5.10 Test results . . . 19

6 Conclusions **19**

Abstract

Visual Question Answering (VQA) is a task in artificial intelligence that combines image analysis and natural language understanding. It involves answering questions about images, requiring models to process both visual and textual information.

This project explores and compares different deep learning models for solving VQA problems. For image feature extraction, ResNet50 and Vision Transformer (ViT) are used, representing both traditional and transformer-based approaches. For understanding the questions, several models are tested, including Recurrent Neural Networks (RNN), one-dimensional Convolutional Neural Networks (1D-CNN), Long Short-Term Memory networks (LSTM), and LSTM with Attention.

The models are trained and evaluated on VQA v2 dataset. This study provides a structured way to compare different model combinations and helps in understanding how various architectures perform in visual question answering tasks.

1 Motivation

Visual Question Answering (VQA) is a growing area of research in artificial intelligence that combines computer vision and natural language processing. It focuses on building systems that can look at an image and answer a question about it using natural language. This task is complex because it requires understanding both the image and the question in a meaningful way.

The idea for this project came from reading different research papers that explored how to build VQA models using deep learning. These papers often used combinations of image models and question models, and we were interested in understanding how these components work together.

We focus on connecting image and question processing models to create a complete system that can answer questions based on visual input. This project is mainly about exploring and learning, not about beating the best models.

2 State of the Art

This section outlines key work that influenced our project.

Our approach is inspired by the model introduced in [1], one of the first to formally define the Visual Question Answering (VQA) task. This work also provided the first large-scale VQA dataset, which became the standard benchmark for training and evaluation.

We also refer to the "Show, Ask, Attend, and Answer" model [4], which improves on earlier methods by introducing attention mechanisms. These allow the model to focus on relevant image regions based on the question, leading to stronger performance across different VQA benchmarks.

3 Objectives

Building on these foundations, our work explores various combinations of encoders for both images and questions. We test CNNs, RNNs, LSTMs, attention-based LSTMs, ResNet-based model and transformer-based models like ViT. We evaluate models using metrics such as accuracy, loss curves, and confusion matrices to gain insight into their strengths and weaknesses.

Figure 1 shows the general model structure, where image and question features are extracted, combined, and passed to a classifier to predict the answer.

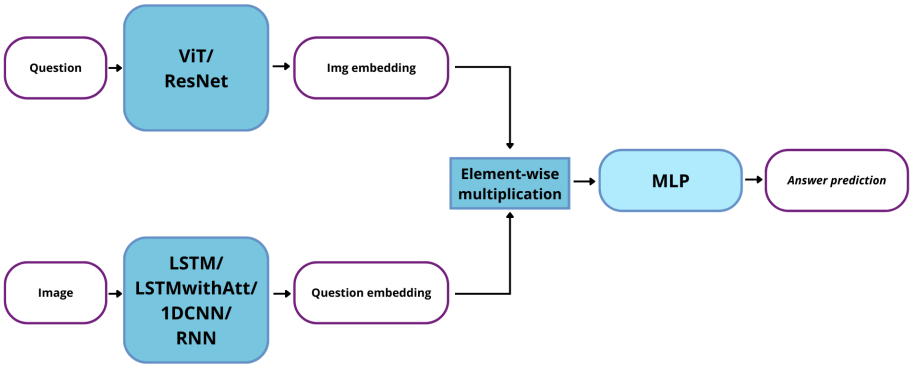


Figure 1: Model overview used in the project

4 Methodology

In this section, the dataset is analyzed and all the models employed to address the task are introduced and described.

4.1 Dataset Overview and Analysis

This project uses two widely adopted datasets in the Visual Question Answering (VQA) domain: **COCO** and **VQA v2**.

[5] **COCO** (Common Objects in Context) is a large-scale dataset for object detection, segmentation, and image captioning. It contains diverse and realistic images with detailed object annotations, making it a common benchmark in computer vision tasks.

[8] **VQA v2** builds upon COCO by adding over 1.1 million question-answer pairs grounded in those images. These questions cover a wide range of reasoning tasks, including

object identification, counting, and commonsense inference. VQA v2 improves over its predecessor by reducing language biases through balanced question-answer pairs, making it a more reliable benchmark for evaluating multimodal models.

Together, COCO and VQA v2 provide a rich framework for training and testing models that jointly process visual and textual information.

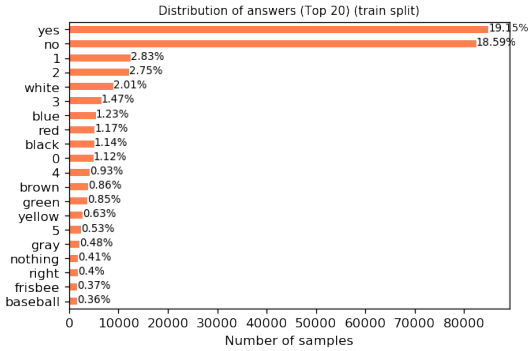


Figure 2: Top 20 answer distribution

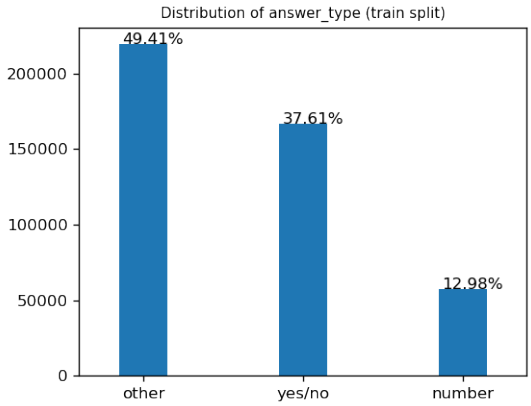


Figure 3: Answer distribution by type

As shown in Figures 2 and 3, the most frequent answers in the dataset are simple binary responses like "yes" and "no", which together account for approximately 37.6% of all answers. Other common categories include numeric answers and short descriptive words, indicating a strong class imbalance.

4.2 Vision Transformer

The backbone is a Vision Transformer (ViT). Starting from a CLIP ViT-L/14 vision transformer [7], a SimCLR-style architecture is developed for fine-tuning the ViT.

4.2.1 Architecture

Images embedding via ViT: the encoder was frozen by default to retain pretrained representations, with the exception of the final transformer blocks (layers 21, 22, and the post-layernorm), which can be unfrozen to enable partial fine-tuning. During contrastive training, the model processes two augmented views of the same image, while in linear evaluation mode, a single view is used.

The encoder outputs a 1024-dimensional CLS token, which is passed through a projection head composed of three linear layers. The projection head first maps the input to 512 dimensions, followed by batch normalization, a ReLU activation function, and dropout with a rate of 0.1. This is followed by a linear transformation to 256 dimensions, again with batch normalization and ReLU activation. The final layer reduces the representation to 128 dimensions. The resulting embeddings are used in contrastive learning objectives such as

the normalized temperature-scaled cross-entropy loss (NT-Xent).
Since in our task, Visual Question Answering, semantic understanding of the scene is essential to answer the question, using a CLIP ViT as backbone is an appropriate and efficient way to let the model learns this semantic features.

Moreover, Contrastive Language-Image pre-training is trained over 400 milion text-image pairs, so it is already suitable for natural language concepts. with the respect to models involving convolutional neural network, for example ResNet, that we are going to explore in the following paragraph.
By introducing a projection head, the model gains additional capacity to reshape and refine the pretrained features, allowing the downstream layers to learn representations that are more discriminative and better suited to the new task.

The inclusion of dropout serves as a regularization mechanism, mitigating overfitting during fine-tuning. Meanwhile, batch normalization stabilizes training and helps maintain healthy gradient flow across layers. The use of intermediate non-linearities such as ReLU further increases the expressiveness of the projection head, enabling it to capture more complex transformations of the input space.

4.2.2 Images transformations

To obtain robust image embeddings, we apply standard SimCLR augmentations, that are *RandomResizedCrop*, *RandomHorizontalFlip*, *ColorJitter* (brightness/contrast/saturation/hue), *RandomGrayscale*, *GaussianBlur*[2]. These transformations encourage the model to learn representations that are stable under varying image appearances, thereby improving generalization and contrastive learning performance.

4.2.3 Loss function

The projection head is trained with the NT-Xent (Normalized Temperature-Scaled Cross-Entropy)[2] loss:

$$\mathcal{L}_i = -\log \frac{\exp(\mathrm{sim}(z_i, z_{i^+})/\tau)}{\sum_{j \neq i} \exp(\mathrm{sim}(z_i, z_j)/\tau)},$$

where each batch contains two augmentations per image. This loss is typical of contrastive learning training because contrastive learning aims to bring positive pairs (similar samples) closer together in the embedding space while pushing negative pairs (dissimilar samples) apart. This is precisely what the NT-Xent loss is designed to do.

4.2.4 Results and plots

Model performances are shown in the following figures. The training plots indicate that the model learns more slowly than initially expected. This behavior is likely influenced by the relatively small number of training images available. The training was conducted using Google Colaboratory, which imposes computational and time constraints. Consequently, the

number of training epochs was limited to 15. A batch size of 64 was used, and the training was carried out on approximately 1,000 images, with 500 images used for validation.

Despite later increasing the training set to around 2,000 images and the validation set to approximately 1,000 images, no significant improvements in performance were observed. This suggests that either the model requires substantially more data to generalize effectively, or that further tuning of the learning process and architectural components is necessary.

After observing that increasing the size of the image dataset did not lead to significant improvements in performance, attention was shifted toward modifying the data augmentation strategy used to generate the two distinct views required for SimCLR training. These augmentations are crucial for the model to learn invariant and robust representations through contrastive learning. For this reason, an additional transformation step was introduced, which normalizes the image values after the application of the previously defined augmentations. This normalization ensures that the pixel value distribution remains consistent stable.

In addition to refining the transformation pipeline, the batch size had to be increased. This may be the best choice since a larger batch size increases the number of negative samples, which strengthens the contrastive signal. Moreover, using a larger batch reduces the risk of accidentally comparing two views of different images that share the same class label, thereby improving the quality of negative sampling and reducing false negatives. However, due to the limited GPU memory available on Google Colaboratory, training with a batch size of 256 was not feasible in our case.

Nevertheless, significant improvements in performance were observed after applying CLIP-specific normalisation to the input images prior to feeding them into the ViT model. This normalisation uses values corresponding to the preprocessing expected by the ViT-L/14 model that was pretrained on the CLIP dataset. Using these values ensures that the pixel distribution of the input images matches that seen during pretraining, enabling the model to generate more meaningful and consistent embeddings.

This change led to a performance boost in contrastive learning with SimCLR, shown in the next figures.

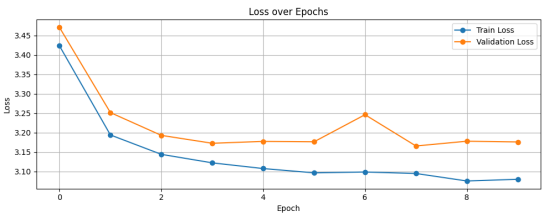


Figure 4: Loss - ViT

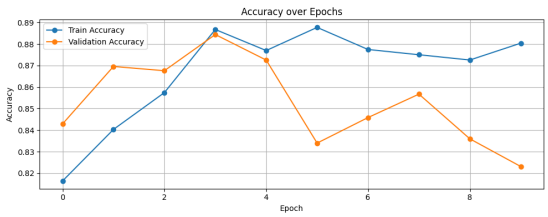


Figure 5: Accuracy - ViT

We also plot the cosine similarity between positive pairs 6. As can be seen from the figure, the model has learned to align the representations of positive pairs strongly (see the peak at 1), providing very consistent embeddings.

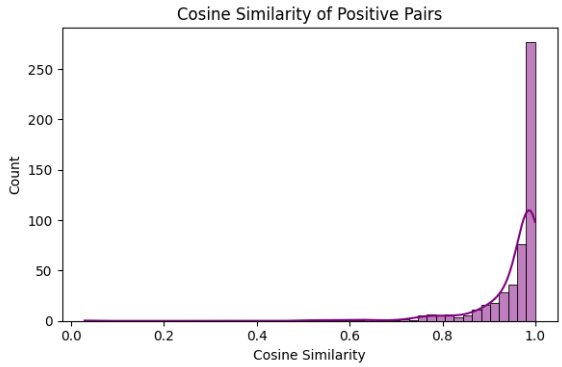


Figure 6: Cosine similarity - ViT

4.3 ResNet

After developing a novel ViT-based model to address the task objectives, an additional model was introduced for comparative evaluation.

4.3.1 Architecture

Images embeddings via ResNet: this second model is based on a pretrained ResNet50 backbone and follows the SimCLR contrastive learning framework, as previously described. The classification head of ResNet50 is removed and replaced with a custom 3-layer projection head, identical to the one used in the ViT-based architecture.

The ResNet-based architecture was trained for 10 epochs using a batch size of 256. Since this model is not computationally intensive, Colaboratory allowed training with this configuration without exceeding resource limits. To mitigate the risk of overfitting, the best-performing model across epochs was saved based on validation metrics, following an early stopping strategy.

4.3.2 Results and plots

These are the results we obtained at the end of the training:

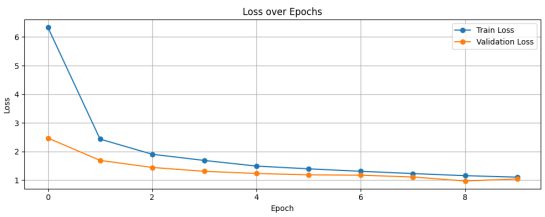


Figure 7: Loss - ResNet

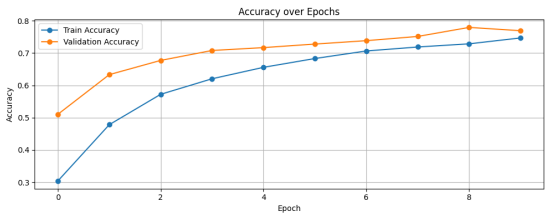


Figure 8: Accuracy - ResNet

In addition, we also plot the cosine similarity between positive pairs (contrastive learning)9. The distribution shape is similar to a Gaussian distribution, with a peak around 0.75–0.85. The variance is greater than that of the ViT model, indicating moderate to high similarity between positive pairs.

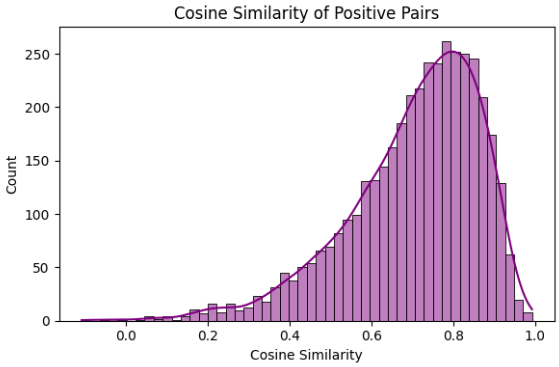


Figure 9: Cosine similarity - ResNet

Comparison of ViT and ResNet

From the previous results, it can be seen that the ViT architecture performs better than the ResNet model. The ViT model shows superior alignment between positive pairs in the cosine similarity plot, suggesting that it has learned a more robust and invariant representation space than the ResNet50 backbone. This is to be expected, given that ViTs are known to handle global context better, which is crucial in contrastive setups. Nevertheless, the ResNet results are reasonable, and the training dynamics suggest that additional epochs could lead to further improvements. In any case, we extract the image embeddings from both models and then compare them for the VQA task.

4.4 LSTM

Long Short-Term Memory (LSTM) networks are a class of recurrent neural networks (RNNs) capable of modeling sequential data while capturing both short and long range dependencies. In Visual Question Answering (VQA), where understanding the syntactic and semantic structure of natural language questions is essential, LSTMs serve as a powerful component for encoding text.

In this work, we adopt an LSTM-based approach to process textual questions, align them with visual features extracted from images, and predict the most likely answer.

The model consists of three main components: an LSTM-based question encoder, a multimodal fusion mechanism, and a multi-layer perceptron (MLP) for classification.

4.4.1 Architecture

Question Encoding with LSTM: The input question is first tokenized and converted into a sequence of word embeddings, each of size 300. These embeddings are fed into a two-layer LSTM with a hidden size of 512. To form a comprehensive representation of the question, we concatenate the final hidden and cell states from both layers. This combined representation captures both the local and global structure of the question, enabling better semantic understanding.

Multimodal Fusion: To represent the visual input, we extract image features using the two different pretrained models introduced before, ResNet and Vision Transformer (ViT), both fine-tuned with SimCLR. The image feature vector is passed through a fully connected (FC) layer that projects it into a 1024-dimensional space, followed by L2 normalization. The encoded question vector is similarly projected into a 1024-dimensional space.

The two modalities are then fused using element-wise multiplication. This simple yet effective fusion technique allows the model to learn joint representations that capture meaningful interactions between the image content and the linguistic structure of the question.

MLP Classifier: The resulting fused vector is passed through a two-layer MLP for answer classification. The first layer maps the 1024-dimensional input to 1000 dimensions, followed by a GELU activation function and dropout for regularization. The second layer projects the vector to the size of the answer vocabulary, producing a logit distribution over possible answers.

This modular architecture comprising sequential question encoding, feature fusion, and a feedforward classifier offers an interpretable and efficient pipeline for deal with the VQA task.

4.5 LSTM with Attention

After testing LSTM, we add a module call Attention in order to helps the model focus on relevant parts of an image, depending on the semantics of the question. It produces a weighted image feature vector that highlights important regions conditioned on the question embedding.

4.5.1 Attention module

We define an *Attention* class that takes image features and question features and outputs a weighted image representation. The image features are passed through a 1×1 convolutional layer, which reduces or adjusts the channel dimensions. This transformation keeps spatial structure intact while bringing all image features into a common representation space. The question feature vector (a single vector per question) is expanded to match the spatial size of the image features. The transformed image features and the expanded question features are added together element-wise. The combined features are passed through another

convolutional layer that produces multiple attention maps (called "glimpses"). Each map highlights different spatial regions of the image that might be relevant to answering the question. Using these attention maps, the image features are reweighted and then summed and flattened into a single vector, which becomes the final attended image representation. This vector is then combined with the question features and passed through fully connected layers for answer prediction.

This attention mechanism allows the model to be more selective in how it interprets the image. For example, when asked "What is the man holding?", the model can focus specifically on the region of the image containing the man's hands. Without attention, the model might try to interpret all image features equally, leading to less accurate results.

4.6 1D CNN

The 1D CNN is based on the model explained in the cited paper [6]. This paper proposes a model where separate CNNs are used for encoding the image and the question, followed by a fusion module to combine visual and textual representations for classification. We implement the idea of using 1D convolutions over text for question encoding.

4.6.1 Architecture

Embeddings and convolutions: each word in the input question is converted into a dense vector using an embedding layer. This embedding vector is reshaped to fit the input format expected by 1D convolutions. Multiple convolutional filters with different kernel sizes (3, 4, 5) are applied in parallel. Each filter captures local features in the question text (like short phrases or word patterns).

Max-Pooling : each convolutional output is passed through a 1D max-pooling operation, extracting the most important feature from each feature map, than all resulting vectors are concatenated to form a compact representation of the question.

Question Projection : The concatenated feature vector is passed through a fully connected (linear) layer that projects it to a 1024-dimensional space. We add a GELU activation and dropout layer for regularization and non-linearity.

Image Feature Projection : the input image feature extracted previously is L2-normalized and projected to the same 1024-dimensional space via another linear layer with GELU activation.

Finally, image and question features are fused through element-wise multiplication. and feed into two fully connected layers, with GELU activation and dropout in between. A final softmax layer produces a probability distribution over the set of possible answers.

4.7 RNN

As last model, we implement a RNN-based architecture.

4.7.1 Architecture

As the previous model, the question input is converted into an embedding vector. A fully connected layer further transforms the embeddings before feeding them into the RNN. The activation function used is GELU. Unlike LSTM, the RNN keeps a simpler hidden state update mechanism. It outputs the hidden states for each time step and the final hidden state from each layer. These hidden states collectively represent the semantic structure and context of the question. The final hidden states from all RNN layers are concatenated to form a question representation. This method captures hierarchical information from different layers of the RNN, enabling richer encoding of the question semantics.

As we previously did in 1D CNN, we fused through element-wise multiplication the question and image projections and than pass the representation through two fully connected layers, with dropout and GELU activation in between. The final layer uses a softmax activation to produce a probability distribution over the answer vocabulary.

4.8 Loss function & initialization of model weights

Since the VQA task is formulated as a multi-class classification problem over a predefined set of possible answers, NLLLoss (Negative Log-Likelihood) is an appropriate loss choice. The model predicts a log-probability distribution over the answer vocabulary, and the NLLLoss penalizes incorrect predictions based on the ground-truth index.

[3]Kaiming Initialization is a weight initialization technique in deep learning that adjusts the initial weights of neural network layers to facilitate efficient training by addressing the vanishing or exploding gradient problem. After trying to train our models with randomly initialised weights, we decided to use this method of initialisation to achieve better performance.

5 Experiments & Results

Given the model overview shown in the previous figure 1, the effectiveness of the proposed architectures, both for text and image processing, is evaluated using a batch size of 512 and training for 15 epochs. To optimize the VQA model, the Adamax optimizer is employed with a learning rate of 0.001. Adamax is a variant of the Adam optimizer that uses the infinity norm (i.e., the maximum of the absolute gradient values) instead of the L2 norm for scaling the updates. This variant is particularly suitable when dealing with high-dimensional parameter spaces or sparse gradients, as it can lead to more stable updates and often better generalization. In the context of the VQA task, where multimodal inputs and discrete output spaces can produce varying gradient dynamics, Adamax offers a robust choice.

For the loss function, as previously said, we used Negative Log-Likelihood Loss (NLL-Loss).

5.1 ViT-LSTM Results

When using ViT image features fine-tuned with SimCLR, the LSTM-based model achieves strong performance on the VQA dataset. The transformer-based visual features contribute to improved spatial reasoning and object-level understanding, enhancing alignment between image and question. The model performs particularly well on complex and relational questions that require global visual context.

Training Dynamics and Overfitting: As shown in figures 10 and 11, the training loss continues to decrease while validation loss increases after epoch 11, indicating overfitting. This occurs because the combination of ViT (a high-capacity visual encoder) and LSTM (a sequential language model) can easily memorize training data, especially when the dataset is limited. ViT's rich image features can lead the model to overfit to specific question-image pairs rather than learning generalizable reasoning patterns. To avoid preserving overfitted weights, we saved the model checkpoint with the lowest validation loss.

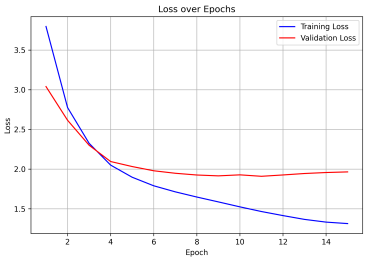


Figure 10: Training and validation loss over epochs for the ViT + LSTM model.

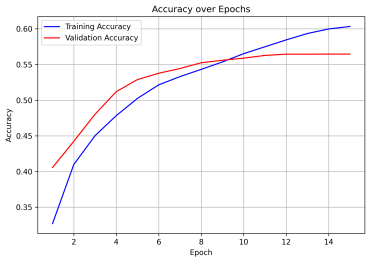


Figure 11: Training and validation accuracy over epochs for the ViT + LSTM model.

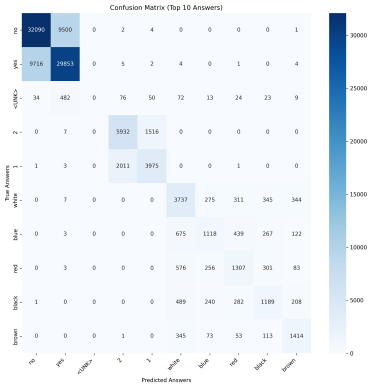


Figure 12: Confusion matrix for the top 10 answers of the ViT + LSTM model.

Based on Figure 12, the most frequent top-10 answers predicted by the LSTM model are "yes" or "no" responses, which means the model tends to work better with yes/no questions. This likely reflects both the answer distribution in the validation set and the model's ability to handle simpler binary classifications more effectively.

5.2 ResNet-LSTM Results

Although its overall accuracy is slightly lower than the ViT-based model, the ResNet-LSTM setup is still effective for simpler visual reasoning tasks.

Training Dynamics and Overfitting: Figure 13 shows training loss decreasing while validation loss plateaus and slightly rises after epoch 11, signaling overfitting. Training accuracy improves beyond 55%, but validation accuracy stays near 50.2%, indicating the model memorizes training data instead of generalizing. This likely results from the LSTM's limited complexity and ResNet's less rich features compared to ViT. We saved the checkpoint with the lowest validation loss to avoid overfitted weights.

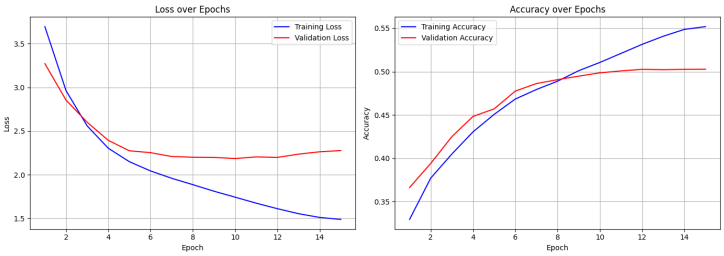


Figure 13: Training and validation loss (left) and accuracy (right) over epochs for the ResNet + LSTM model.

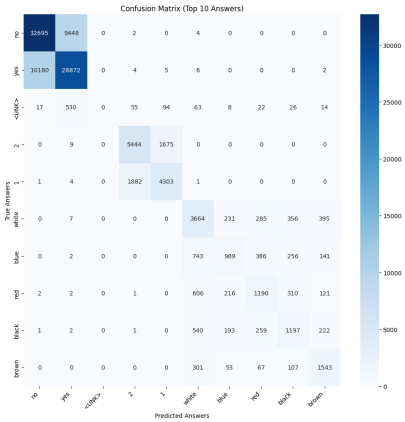


Figure 14: Confusion matrix for the top 10 answers of the ResNet + LSTM model.

Based on the confusion matrices in Figures 12 and 14, we can see that the LSTM-ViT model better captures diverse question types beyond yes/no, whereas the ResNet-LSTM performs better mainly on yes/no questions.

5.3 ViT-LSTM with Attention Results

Using ViT image features fine-tuned with SimCLR, the LSTM model enhanced with an attention mechanism shows improved performance on the VQA dataset. Attention helps the model focus on relevant parts of the question and image features, boosting its ability to handle complex reasoning and relational questions requiring global context.

Training Dynamics and Overfitting: As seen in figures 15 and 16, training loss steadily decreases while validation loss starts increasing after epoch 11, indicating overfitting. Also in this case, the best checkpoint was saved at the lowest validation loss to avoid overfitting.

5.4 ResNet-LSTM with Attention Results

With ResNet features, the LSTM + attention model performs well on object-centric and detail-focused questions. Though slightly behind the ViT-based variant, it remains strong for simpler reasoning tasks.

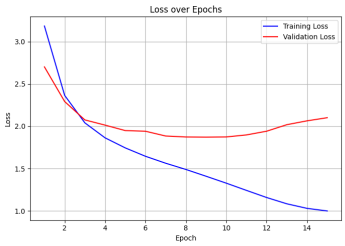


Figure 15: Training and validation loss over epochs for the ViT + LSTM + Attention model.

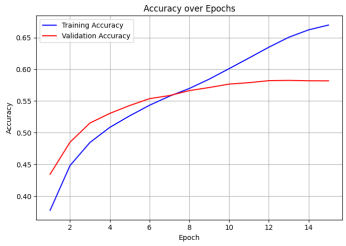


Figure 16: Training and validation accuracy over epochs for the ViT + LSTM + Attention model.

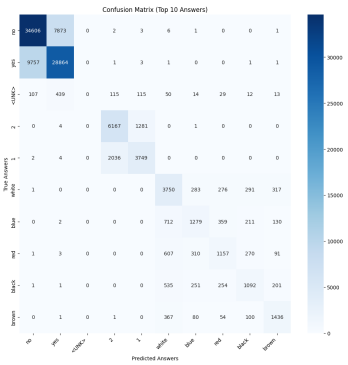


Figure 17: Confusion matrix for the top 10 answers of the ViT + LSTM + Attention model.

Training Dynamics and Overfitting: Figure 18 shows the training loss decreasing steadily, while validation loss plateaus and rises after epoch 11, signaling overfitting. Training accuracy surpasses 55%, but validation accuracy remains around 50.2%. This gap suggests memorization of training samples instead of generalization. The LSTM's limited capacity combined with less rich ResNet features contributes to this behavior. The checkpoint with lowest validation loss was saved to avoid overfitting.

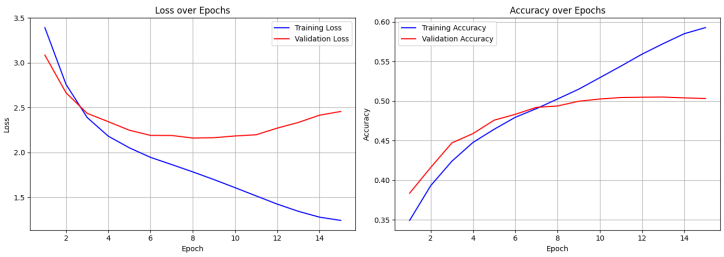


Figure 18: Training and validation loss (left) and accuracy (right) over epochs for the ResNet + LSTM + Attention model.

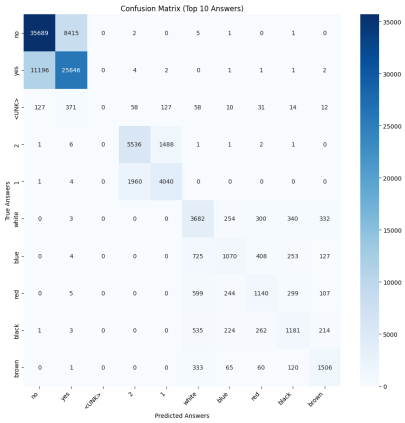


Figure 19: Confusion matrix of the top 10 answers for the ResNet + LSTM + Attention model.

Comparing Figures 17 and 19, the ViT-based LSTM + Attention model better captures varied question types, while the ResNet version remains more focused on yes/no answers.

5.5 ViT-1DCNN Results

The introduction of 1D CNN to handle text input and extract embeddings, has a slightly improvement in term of validation accuracy with the respect to the LSTM with attention model.

Training Dynamics and Overfitting: As shown in figure 20 and 21, the most prominent issue observed is overfitting. The model is learning the training data too well, to the detriment of its generalization ability. In order to mitigate overfitting we save the model with the lowest validation loss.

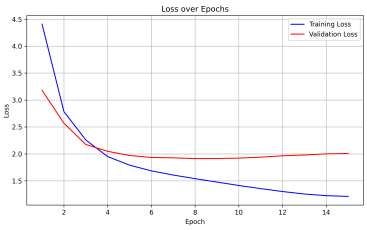


Figure 20: Training and validation loss over epochs for ViT + 1D CNN model.

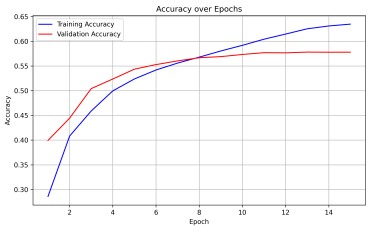


Figure 21: Training and validation accuracy over epochs for ViT + 1D CNN model.

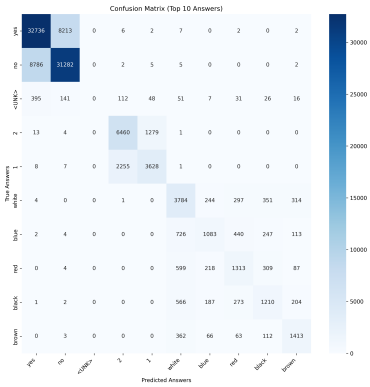


Figure 22: Confusion matrix for the top 10 answers of the ViT + 1D CNN model.

Based on Figure 22, the 1D CNN model performs well in both "yes" and "no" predictions and in other top answers, until now it seems to be the best model to capture true positives.

5.6 ResNet-1DCNN Results

As we expected, using ResNet instead of ViT to handle images input has worse performance in term of validation accuracy and loss.

Training Dynamics and Overfitting: Figure 23 shows the training loss decreasing steadily, while validation loss plateaus and rises after epoch 8, signaling overfitting. The checkpoint with lowest validation loss was saved to avoid overfitting.

A comparison of Figures 25 and 22 shows that the ViT-based 1D CNN model captures yes/no answers more effectively than the ResNet model. Regarding varied question types, the models are similar in terms of confusion matrix results.

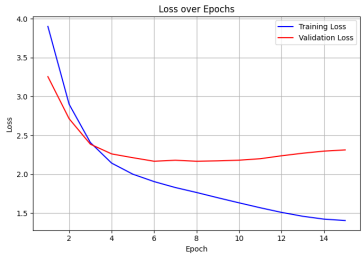


Figure 23: Training and validation loss for the ResNet + CNN model over epochs.

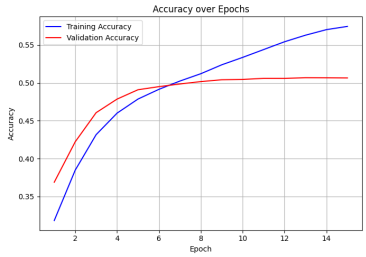


Figure 24: Training and validation accuracy for the ResNet + CNN model over epochs.

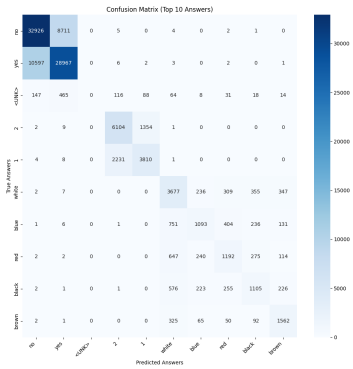


Figure 25: Confusion matrix of the top 10 answers for the ResNet + CNN model.

5.7 ViT-RNN Results

The models that uses RNN for image processing is the only one that does not exhibit severe overfitting. This is probably because RNN models are simpler and have fewer parameters than LSTM or CNN models.

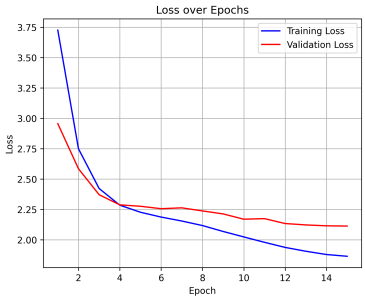


Figure 26: Training and validation loss for the ViT + RNN model.

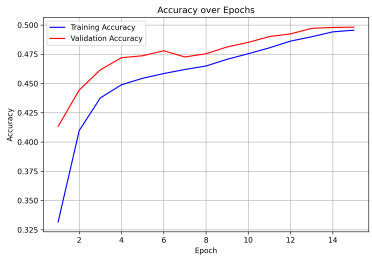


Figure 27: Training and validation accuracy for the ViT + RNN model.

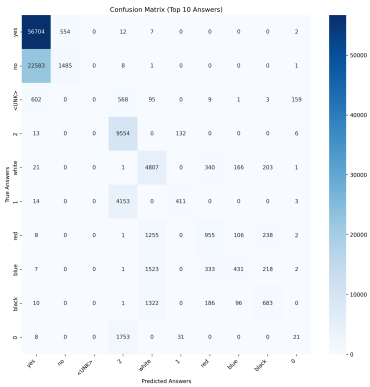


Figure 28: Confusion matrix of the top 10 answers for the ViT + RNN model.

Training Dynamics and Overfitting: As seen in figures 26 and 27 the model is clearly learning, as both accuracy increases and loss decreases on the training set. However, both the accuracy and loss curves indicate that the model's performance on the validation set begins to plateau significantly after around 8–10 epochs. There is a hint of overfitting in the later epochs, where the training loss continues to drop while the validation loss remains

largely unchanged. Validation accuracy also plateaus, indicating limited improvement in generalisation in the latter half of training.

Based on Figure 28, The RNN model with ViT, compared to the previous models, presents worse general performance, primarily in the yes/no predictions.

5.8 ResNet-RNN Results

As previously mentioned, this model involving RNN and utilising ResNet instead of ViT does not suffer from severe overfitting.

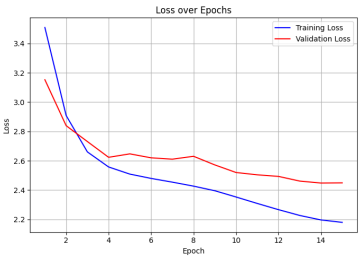


Figure 29: Training and validation loss for the ResNet + RNN model.

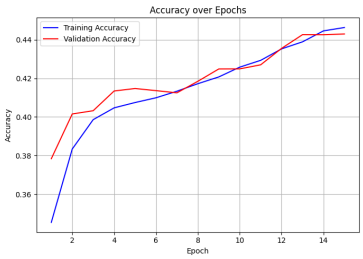


Figure 30: Training and validation accuracy for the ResNet + RNN model.

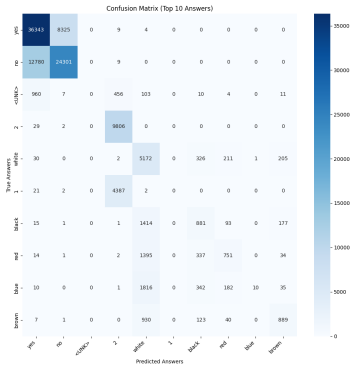


Figure 31: Confusion matrix of the top 10 answers for the ResNet + RNN model.

Training Dynamics and Overfitting: Figures 29 and 30 confirm what we concluded from the previous model. The only difference between the last two models is shown in the confusion matrix.

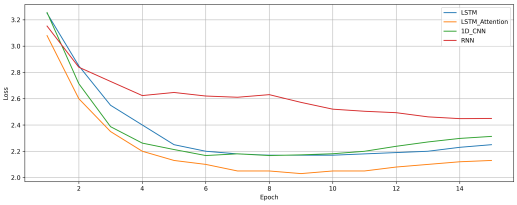
Comparing Figures 31 and 28, the confusion matrix for the final model shows that it performs better in terms of yes/no answers, correctly predicting more true positives and true negatives with respect to the model of ViT and RNN.

5.9 Final comparison and considerations

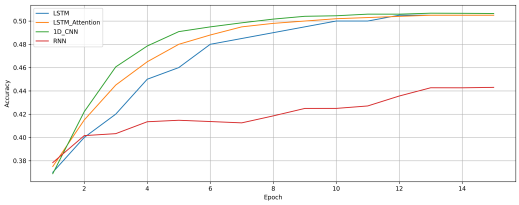
Finally, we present the overall performance of the validation metrics for the ResNet and ViT based image encoders, demonstrating how they perform over 15 epochs.

5.9.1 ResNet-based Encoder Performance

Figure 32 shows the ResNet encoder results with different sequence heads. Validation loss decreases for all heads, with `LSTM_Attention` lowest, followed by `LSTM`, then `1D_CNN`, and `RNN` highest. Accuracy peaks around 0.50–0.51 for `LSTM_Attention` and `1D_CNN`, 0.50 for `LSTM`, and 0.44 for `RNN`.

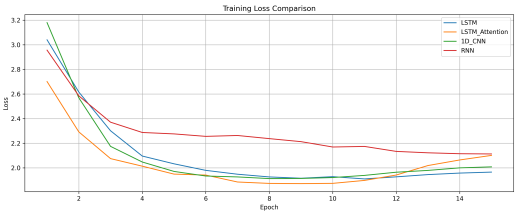


(a) Validation loss curve

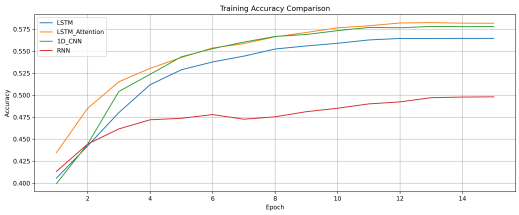


(b) Validation accuracy curve

Figure 32: ResNet-based image encoder validation metrics for different sequence modeling heads.



(a) Validation loss curve



(b) Validation accuracy curve

Figure 33: ViT-based image encoder validation metrics for different sequence modeling heads.

5.9.2 ViT-based Encoder Performance

Figure 33 presents ViT encoder results. All heads show loss reduction; `LSTM_Attention` performs best, followed by `LSTM` and `1D_CNN`, with `RNN` worst. Accuracy reaches 0.58 for `LSTM_Attention`, 0.57 for `LSTM` and `1D_CNN`, and 0.50 for `RNN`.

Comparative Summary

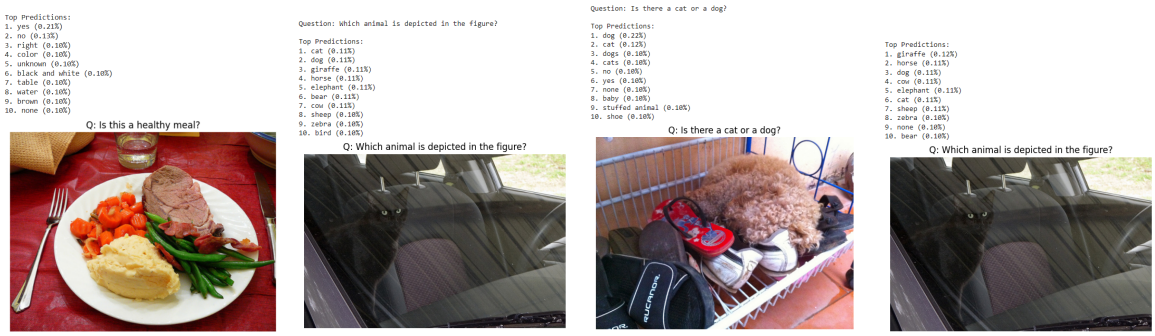
1. The ViT-based encoders consistently outperform the ResNet-based encoders, achieving higher validation accuracy in the range of approximately 0.57 to 0.58, compared to 0.50 to 0.51 for ResNet. This suggests that ViT's ability to model global context provides a stronger foundation for downstream tasks requiring complex visual reasoning.

2. Across both encoder types, the `LSTM_Attention` sequence head delivers the best performance, indicating that incorporating attention mechanisms effectively enhances the integration of visual features and sequential information, leading to improved accuracy and lower loss.

3. The `RNN` head consistently performs the worst for both encoder architectures, likely due to its limited capacity to capture long-range dependencies and vulnerability to optimization issues such as vanishing gradients.

In summary, the combination of ViT encoders with LSTM attention-based provides the most effective approach, offering superior performance and more robust validation results compared to ResNet encoders paired with simpler sequence models.

5.10 Test results



(a) ViT + RNN (b) ResNet + RNN (c) ViT + CNN (d) ResNet + CNN

(e) ViT + LSTMatt. (f) ResNet + LSTMatt. (g) ViT + LSTM (h) ResNet + LSTM

Figure 34: Qualitative test results for different encoder-sequence head combinations.

From the test images, it's clear that most models are good at predicting binary "yes" or "no" answers. However, when a question requires a more complex or specific response, not all models are able to predict the answer correctly.

6 Conclusions

The experiments demonstrate that ViT-based encoders consistently outperform ResNet-based encoders, achieving higher accuracy and more stable convergence. Among sequence heads, LSTM with attention delivers the best overall performance, indicating that incorporating attention improves temporal modeling. In contrast, simple RNNs underperform significantly, suggesting they lack the capacity to capture complex dependencies. All models

converge within 15 epochs, but ViT shows more consistent learning dynamics, confirming the advantage of transformer-based image representations in this multimodal setup.

References

[1] Stanislaw Antol et al. "VQA: Visual Question Answering". In: *CoRR* abs/1505.00468 (2015). arXiv: 1505.00468. URL: http://arxiv.org/abs/1505.00468.

[2] Ting Chen et al. "A Simple Framework for Contrastive Learning of Visual Representations". In: *CoRR* abs/2002.05709 (2020). arXiv: 2002.05709. URL: https://arxiv.org/abs/2002.05709.

[3] GeeksforGeeks. *Kaiming Initialization in Deep Learning*. https://www.geeksforgeeks.org/deep-learning/kaiming-initialization-in-deep-learning/. Accessed: 2025-07-14. 2025.

[4] Vahid Kazemi and Ali Elqursh. "Show, Ask, Attend, and Answer: A Strong Baseline For Visual Question Answering". In: *CoRR* abs/1704.03162 (2017). arXiv: 1704.03162. URL: http://arxiv.org/abs/1704.03162.

[5] Tsung-Yi Lin et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV].

[6] Lin Ma, Zhengdong Lu, and Hang Li. "Learning to Answer Questions From Image using Convolutional Neural Network". In: *CoRR* abs/1506.00333 (2015). arXiv: 1506.00333. URL: http://arxiv.org/abs/1506.00333.

[7] Alec Radford et al. "Learning Transferable Visual Models From Natural Language Supervision". In: *CoRR* abs/2103.00020 (2021). arXiv: 2103.00020. URL: https://arxiv.org/abs/2103.00020.

[8] *Visual Question Answering (VQA)*. https://visualqa.org/. Accessed: 2025-07-13.