Visual Question Answering Using CLIP with Local Feature Enhancement

Sai Surya Vidul Chinthamaneni
SID - 862466578
schin055@ucr.edu

Tushar Baid
SID - 862469499
tbaid002@ucr.edu

Yash Sanjay Kathe
SID - 862464930
ykath001@ucr.edu

Abstract

Visual Question Answering (VQA) requires a model to understand the overall context as well as the fine-grained details of an image. The CLIP model by OpenAI is effective at extracting global features from an image but struggles with tasks that require a detailed understanding of local features. To address this, we propose a dual encoder architecture that combines CLIP's global features with local features extracted using a Vision Transformer (ViT). These features are fused through either concatenation or attention-based layers to enhance the model's ability to answer complex questions. Our approach tries to improve the model's output metrics and capability to recognize fine-grained details, making it more effective for VQA tasks.

1 Introduction

Visual Question Answering (VQA) [1] tasks include both natural language understanding and computer vision. The model should understand textual queries based on specific aspects of the image and accurately analyze the image to generate appropriate responses. One such model is CLIP [10] released by OpenAI which primarily focuses on extracting global features but often struggles with fine-grained details such as counting objects, identifying small elements, or understanding spatial relationships [12]. Such tasks require an architecture that understands and captures both global and local features. Therefore, in this project, we propose a dual encoder approach that takes advantage of CLIP's strength for global feature extraction and an additional Vision Transformer (ViT) [2] to extract the detailed local features. The fusion of features is done by either using concatenation or attention-based mechanisms. This approach enhances the model to focus on specific regions of the image while also understanding the broader context within the image. With this architecture, the model can achieve a more comprehensive understanding of both the image and text query, making it more adaptable to real-world VQA tasks.

2 Related Work

Radford et al. (2021) [11] introduced CLIP, demonstrating impressive zero-shot capabilities by learning from a large dataset of image-text pairs. This enables CLIP to recognize and classify images from textual descriptions without task-specific fine-tuning which makes the model highly flexible for various applications. However, it relies on global feature extraction which limits its performance in tasks that require attention to local details 1. For example, in (VQA) [1], where precise object identification and spatial reasoning are crucial, CLIP's approach of global extraction struggles to handle fine-grained queries that depend on local regions [12]. Nevertheless, CLIP's ability to learn combined embeddings for images and text has laid the foundation for advanced multimodal models 1.

Dosovitskiy et al. (2020) [2] showcased how Vision Transformers (ViTs) can achieve state of the art performance by processing images in patches and capturing both global and local features in an

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

image. ViTs break down images into smaller patches which help the model understand complex local structures along with global context. This makes ViTs well-suited for tasks requiring high-resolution details. VQA models often make use of attention mechanisms to focus on specific image regions which enhances their performance in complex question-answering tasks. Attention mechanisms in ViTs enable the model to identify objects or spatial relationships by assigning attention to important image regions, that are otherwise difficult for traditional CNN-based models.

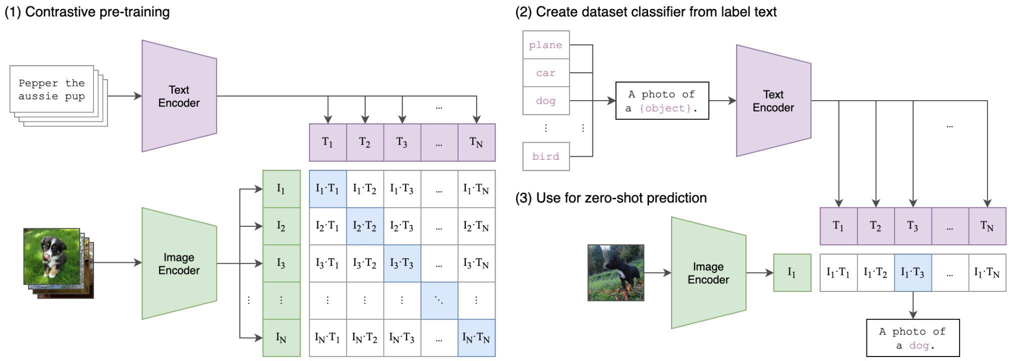
(1) Contrastive pre-training
Pepper the aussie pup
Text Encoder
Image Encoder
T_1 T_2 T_3 ... T_N
I_1 I_2 I_3 ⋮ I_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
$I_2 \cdot T_1$ $I_2 \cdot T_2$ $I_2 \cdot T_3$ ... $I_2 \cdot T_N$
$I_3 \cdot T_1$ $I_3 \cdot T_2$ $I_3 \cdot T_3$ ... $I_3 \cdot T_N$
$I_N \cdot T_1$ $I_N \cdot T_2$ $I_N \cdot T_3$ ... $I_N \cdot T_N$
(2) Create dataset classifier from label text
plane
car
dog
bird
A photo of a {object}.
Text Encoder
(3) Use for zero-shot prediction
Image Encoder
I_1
T_1 T_2 T_3 ... T_N
$I_1 \cdot T_1$ $I_1 \cdot T_2$ $I_1 \cdot T_3$ ... $I_1 \cdot T_N$
A photo of a dog.

Figure 1: Architecture of the CLIP Model

3 Methodology

3.1 Overview

This project aims to address the limitations of CLIP in Visual Question Answering (VQA) tasks by introducing a novel architecture that integrates global and local feature extraction for enhanced reasoning capabilities. To compare our results, we implement a baseline CLIP for benchmarking. Our methodology consists of three key components:

- **Baseline CLIP Implementation:** This implementation leverages CLIP's pre-trained capabilities as a starting point and serves as a foundation for comparison. It acts as a reference to evaluate the effectiveness of the proposed dual encoder architecture [11].
- **Dual Encoder Architecture:** This architecture builds on CLIP's strengths in extracting global features [11] by adding a local encoder specifically designed to capture finer details. To integrate the outputs of the global and local encoders with text embeddings, the architecture introduces two fusion mechanisms: concatenation-based and attention-based [2].
- **Training Framework:** This framework encapsulates the entire training process. It employs optimization and regularization techniques such as weight decay and mixed precision training [8]. It also includes all the required components to run the model, such as training, plotting, and evaluation.

3.2 Baseline CLIP Implementation

The baseline CLIP architecture is used as a foundation for comparison in this project. By building on CLIP's pre-trained capabilities, this baseline is specifically adapted for Visual Question Answering (VQA) tasks. CLIP uses a Vision Transformer (ViT-B/32) as its **Image Encoder** to process images at a resolution of 224x224 pixels [2, 11]. It extracts global image features and outputs a 768-dimensional embedding that represents the overall context of the scene. We utilize CLIP's **Text Encoder** (text transformer) to take input questions and process them. Questions are tokenized to a maximum of 77 tokens (based on the text of the question categories considered), ensuring they fit within the model's input constraints. The text encoder outputs embeddings in the shared 768-dimensional space as the image features, allowing effective alignment of textual and visual information. The features in this shared embedding space are then concatenated into a single vector and provided as input to the answer prediction head 2.

The **Answer Head** is a custom prediction layer designed for VQA. It includes several components to improve training stability and performance, such as batch normalization, dropout, and GELU activations for handling complex feature mappings [6]. The final output layer maps the unified features to a vocabulary of 3,129 possible answers. This layer is the only trainable component of the baseline model, making the approach efficient while still adaptable to the VQA task 2. We made a key design choice to freeze the weights of both the image and text encoders to retain their pre-trained knowledge while training only the answer head [11]. This approach ensures stability and computational efficiency during fine-tuning.

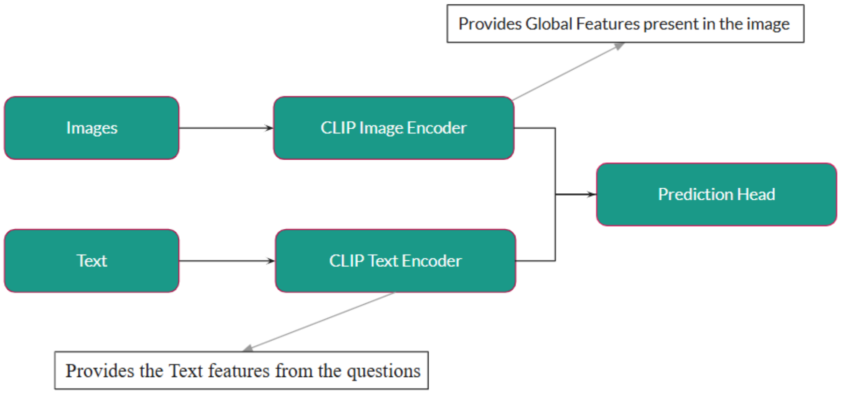
Provides Global Features present in the image
Images
CLIP Image Encoder
Prediction Head
Text
CLIP Text Encoder
Provides the Text features from the questions

Figure 2: Baseline CLIP Architecture: Overview showing the integration of global image features, text embeddings, and prediction head.

3.3 Dual Encoder Implementation

The Dual Encoder Implementation builds on CLIP's strengths by introducing a second Vision Transformer (ViT) designed to capture fine-grained local features. While CLIP's encoder focuses on the overall context of the scene (global features), the local encoder detects intricate patterns and spatial relationships that might otherwise be missed. This combination allows the model to better handle complex VQA tasks. To bring everything together, the architecture uses two fusion methods—concatenation-based and attention-based, to integrate global, local, and text features effectively.

CLIP's **Image** and **Text Encoder** as explained earlier are used to capture and obtain global-image and text features respectively. To complement the global encoder, a second Vision Transformer (ViT-B/16) is added to focus on fine-grained details. With a smaller patch size of 16x16 pixels, this local encoder captures detailed patterns and spatial relationships [2] 3. Most layers in the local encoder remain frozen to preserve pre-trained knowledge, but the last two layers (Layers 11 and 12) are unfrozen, allowing for task-specific fine-tuning. Like the global encoder, it produces 768-dimensional embeddings that align with the global and text features.

The architecture integrates global, local, and textual features using two different **Fusion Mechanisms**, each designed to make the most of the unique strengths of these feature streams.

3.4 Concatenation-Based Fusion

This approach is simple yet effective, aligning all features—global, local, and text—to the same 768-dimensional space and combining them into a single vector. By preserving the individual characteristics of each feature, this method ensures that no stream dominates or overshadows the others. The combined vector is then passed through a series of normalization layers, non-linear transformations (like GELU), and residual connections, which enhance stability and support smooth gradient flow during training [6] 3. Finally, the processed features are fed into a multi-layer network that maps the unified representation to the answer space. This method's straightforward design balances computational efficiency with robust feature integration 3.

3.5 Attention-Based Fusion

For a more dynamic integration, this method uses an attention mechanism to assign varying importance to the global, local, and textual feature streams. Parallel attention heads capture different aspects of the data, helping the model focus on the most relevant details while still considering the broader context [3] 3. The process is hierarchical, refining the fused features step by step to create a richer and more accurate representation. This mechanism combines the strengths of weighted pooling for local features and cross-attention for global and textual integration, allowing the model to reason more effectively across modalities 3.

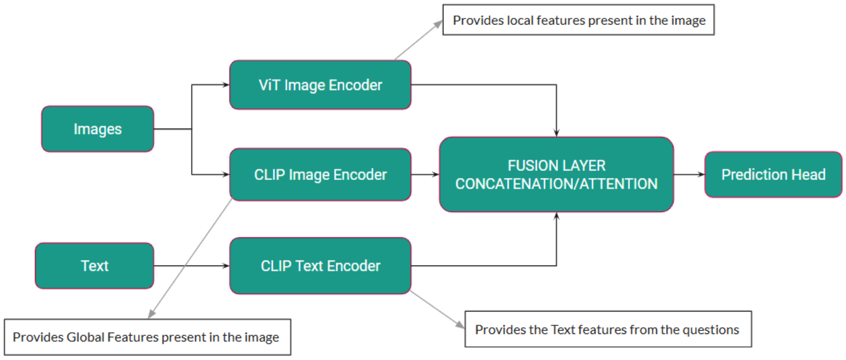
Provides local features present in the image
ViT Image Encoder
Images
CLIP Image Encoder
FUSION LAYER
CONCATENATION/ATTENTION
Prediction Head
Text
CLIP Text Encoder
Provides Global Features present in the image
Provides the Text features from the questions

Figure 3: Dual Encoder Architecture: Overview showcasing the integration of global, local and textual features using fusion mechanisms.

3.6 Training Framework

The training framework is designed to ensure efficient, stable, and effective fine-tuning for both implementations. It includes all essential components such as training, evaluation on validation and test sets, checkpointing, and performance plotting. By leveraging advanced optimization techniques, regularization methods, and computational efficiency strategies, the framework aims to maximize model performance while minimizing computational overhead. To optimize the model, the framework uses the **AdamW** optimizer, which combines the strengths of Adam with weight decay to enhance generalization [9]. The objective function is **cross-entropy loss**, which ensures accurate predictions for a wide variety of question types. Additionally, a fixed **learning rate** is implemented during training, as the model is trained for fewer epochs 4.

Dropout is applied in the answer head and fusion layers to combat overfitting by randomly dropping connections during training [4]. **Weight decay** encourages simpler models by penalizing large weights, thereby promoting better generalization [9]. Additionally, **batch normalization** is introduced in intermediate layers to stabilize training and mitigate the effects of internal covariate shifts. The figure 4 highlights the extensive hyper-parameter tuning done during the course of this project. This depicts why our project experimentation was so time intensive. [7].

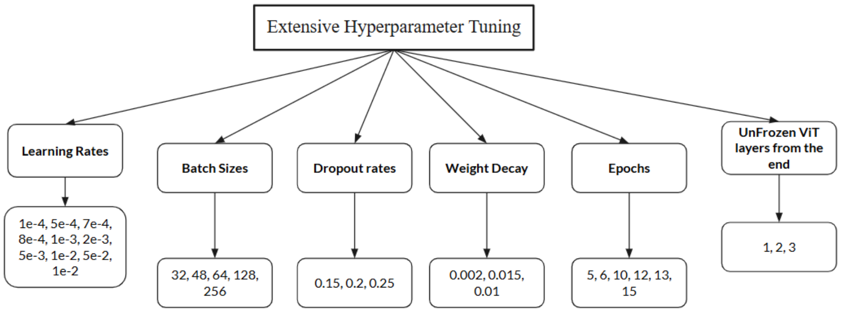

Figure 4: Hyperparameter Tuning: The ranges for learning rates, batch sizes, dropout rates, weight decay, epochs, and unfrozen ViT layers explored during training.

To make training more efficient, the framework incorporates **mixed precision training**. This approach allows computations to be performed in both 16-bit and 32-bit precision, significantly reducing memory usage and speeding up GPU training [5]. **Gradient scaling** is applied to maintain numerical stability during backpropagation, ensuring that the model performs reliably even under lower precision.

4 Dataset

The VQA v2 dataset is a widely-used, large-scale dataset designed to evaluate models on their ability to answer natural language questions about images [1]. It includes more than 1.1 million questions paired with 204,721 images, covering a diverse range of real-world scenarios. Each question is annotated with answers from 10 human annotators, providing robust ground truth for model evaluation. The dataset contains various question types, including Yes/No, Number-based, and descriptive questions, making it a comprehensive benchmark for VQA tasks. For this project, we work with a carefully selected subset of the VQA v2 dataset that focuses on three specific question types: Yes/No questions (40%), Color questions (30%), and Count questions (30%) - in the total number of samples. This subset consists of 50,000 image-text pairs, split into 35,000 samples for training (70%), 7,500 for validation (15%), and 7,500 for testing (15%). By narrowing the focus to these question types, the subset allows us to evaluate the model's ability to understand both high and low level contexts. This makes our model training simpler, thus helping us provide a detailed study for this project with limited time and computational resources.

5 Results

5.1 Performance Metrics and Per-Model Analysis

The performance of the three model variants—Baseline CLIP, Dual Encoder with Concatenation, and Dual Encoder with Attention—was evaluated using two widely-used metrics: BLEU-1 and METEOR, as well as accuracies across different question types. These metrics measure the relevance and accuracy of the models' predictions compared to the ground truth. The Baseline CLIP model achieved a BLEU-1 score of 0.4672 and a METEOR score of 0.2382, demonstrating its ability to leverage global feature understanding effectively. The Concatenation-based Dual Encoder performed similarly, with a BLEU-1 score of 0.4634 and a METEOR score of 0.2372, showing its capability to balance global and local feature integration. The Attention-based Dual Encoder scored slightly lower, with a BLEU-1 of 0.4326 and METEOR of 0.2214, reflecting the trade-off between architectural complexity and the dataset's limitations. Table 1 summarizes these results.

Table 1: Performance Metrics for BLEU-1 and METEOR

Model	**BLEU-1**	**METEOR**
Baseline CLIP	0.4672	0.2382
Dual Encoder + Concatenation	0.4634	0.2372
Dual Encoder + Attention	0.4326	0.2214

5.1.1 Baseline CLIP

The Baseline CLIP model demonstrated stable and efficient training, achieving a training accuracy of 50.59%, validation accuracy of 47.03%, and test accuracy of 46.45% (Table 2). Among all the question categories, Yes/No questions exhibited the highest performance, with a validation accuracy of 53.83%. This result highlights the model's strength in global feature processing, making it particularly effective for binary decision tasks. However, the Baseline CLIP struggled with tasks requiring local feature analysis, achieving a validation accuracy of 47.11% for color identification and 37.87% for counting tasks. Despite these challenges, the model's computational efficiency, with a training time of approximately 3 hours per run, established a reliable benchmark for comparison with the dual encoder models. As shown in Figure 5, the Baseline CLIP model's attention heatmaps highlight its ability to focus on the primary subject in an image, effectively capturing global context but lacking detailed attention to finer features.

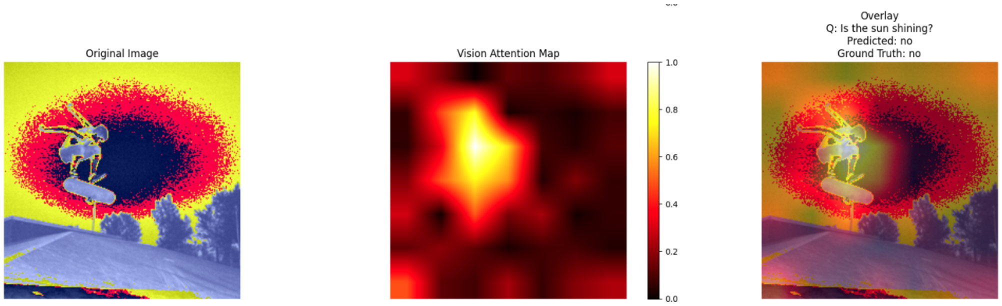

Figure 5: CLIP Model Heatmaps

5.1.2 Dual Encoder with Concatenation

The Concatenation-based Dual Encoder achieved improved performance compared to the Baseline CLIP in terms of training accuracy (55.79%), though its validation accuracy (45.25%) and test accuracy (46.07%) were comparable (Table 2). This difference can be attributed to the increased epochs and different training parameters. It performed well in Yes/No questions, achieving a validation accuracy of 51.60%, and showed improvements in color identification and counting tasks, with validation accuracies of 44.89% and 37.16%, respectively. These improvements indicate the model's ability to integrate global and local features effectively. However, the additional computational demands resulted in a longer training time of approximately 5 hours per run, making it resource-intensive. As shown in Figure 6, the heatmaps for the Concatenation-based Dual Encoder reveal its ability to focus on both global and local features. The model highlights multiple areas of interest within the image, enabling better feature integration for complex reasoning.

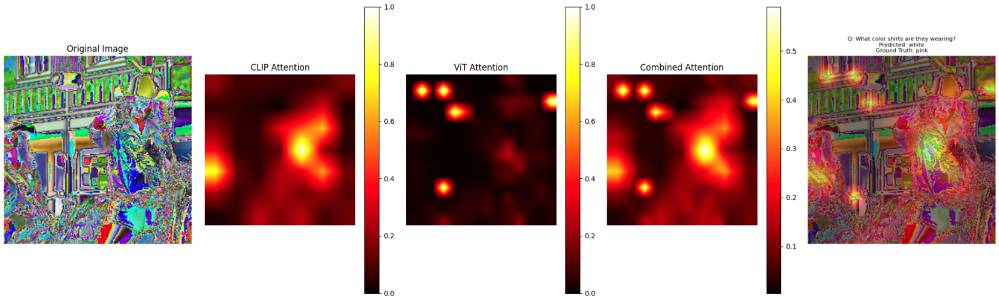

Figure 6: Dual Encoder with Concatenation Heatmaps - CLIP, ViT, combined and output

5.1.3 Dual Encoder with Attention

The Attention-based Dual Encoder employs complex feature integration mechanism. It achieved a training accuracy of 49.71%, validation accuracy of 42.52%, and test accuracy of 42.96% (Table 2). Among the question categories, its performance was moderate, with validation accuracies of 50.63% for Yes/No questions, 40.71% for color identification, and 33.51% for counting tasks. These results highlight the potential of the attention mechanism for nuanced feature integration while emphasizing the need for further optimization. Training this model required approximately 5 hours per run, similar to the concatenation-based variant. As shown in Figure 7, the attention heatmaps for this model demonstrate its ability of how the different features are combined. The Attention-based Dual Encoder highlights specific regions of interest with greater precision, supporting hierarchical reasoning and spatial understanding.

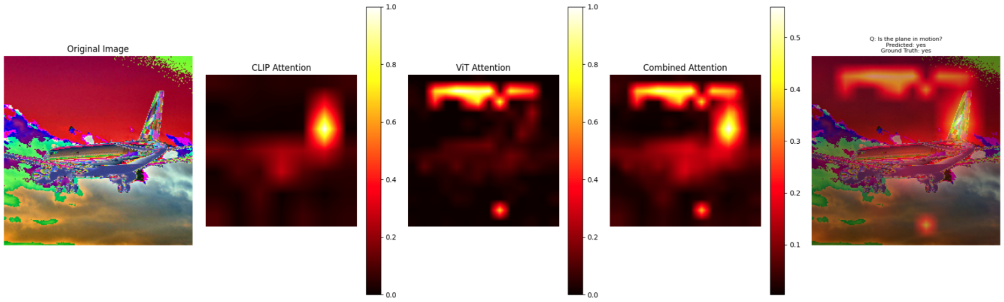

Figure 7: Dual Encoder with Attention Heatmaps - CLIP, ViT, combined and output

Table 2: Training, Validation, and Test Accuracies for Each Model

Model	**Training Accuracy**	**Validation Accuracy**	**Test Accuracy**
Baseline CLIP	50.59%	47.03%	46.45%
Dual Encoder + Concatenation	55.79%	45.25%	46.07%
Dual Encoder + Attention	49.71%	42.52%	42.96%

5.2 Performance Analysis by Question Type

Yes/No questions were the easiest for all models, showcasing their strength in binary decision-making. The Baseline CLIP performed well by effectively leveraging global context, while the Dual Encoder models added value through refined local feature integration (Table 3). Color questions posed moderate challenges, requiring a deeper focus on fine-grained details. The Dual Encoder models demonstrated comparable performance to the Baseline, emphasizing the importance of using a much bigger dataset for local feature detection (Table 3). Counting tasks proved to be the most challenging, as all models struggled to accurately process spatial relationships. Thus, highlight the need for more advanced mechanisms to handle complex spatial reasoning (Table 3).

Table 3: Per-Question Type Accuracies for Each Model

Model	**Yes/No Accuracy**	**Color Accuracy**	**Count Accuracy**
Baseline CLIP	53.83%	47.11%	37.87%
Dual Encoder + Concatenation	51.60%	44.89%	37.16%
Dual Encoder + Attention	50.63%	40.71%	33.51%

6 Future Scope

The current prediction mechanism can be replaced with a cosine similarity-based head to compute the similarity between the text and image feature embeddings. This approach might improve performance for tasks that require high alignment between multi-modal features. It could also enhance the model's generalization across unseen data and complex queries. Implementing learning rate and weight decay scheduling during training to dynamically optimize hyper parameters may lead to improvement in performance. LoRa can be used to efficiently fine-tune the model by optimizing only a small number of low-rank matrices within pre-trained encoders. This will reduce the computational overhead significantly, enabling the exploration of larger models and datasets. The implementation of LoRa can help make the training process more scalable without sacrificing accuracy. Using a larger dataset with diverse and highly contextual image-text pairs can maximize the potential of the dual-encoder approach. A larger dataset will test the model's ability to handle more complex and intricate questions which can improve its ability to recognize local features and fine-grained details.

7 Conclusion

In this project, we successfully improved the Visual Question Answering process by combining CLIP's capabilities of global feature extraction with the fine-grained local features captured by

Vision Transformers (ViT). The dual encoder architecture with concatenation and attention-based fusion layer increased the model's capacity to perform complicated VQA tasks that require accurate contextual understanding as well as detailed visual reasoning. Although the results showed promising improvements in performance, there were problems such as computational intensity and overfitting, particularly with large models. Future work can build on this foundation by experimenting with different fusion strategies using larger datasets, and exploring advanced techniques like LoRa for model optimization.

8 Use of GPT

We used GPT to create boilerplate code and understand the Encoder architectures, which helped us move forward with our work. We also used it for 'Creation/Caching' section, and it was also used in fixing bugs and errors.

9 Contributions

Team Member	**Data Preprocessing**	**Baseline CLIP**	**Dual Encoder with Concatenation Fusion**	**Dual Encoder with Attention Fusion**	**Visualization and Testing**	**Presentation and Report**
Sai	15%	15%	70%	70%	15%	15%
Tushar	70%	15%	15%	15%	15%	70%
Yash	15%	70%	15%	15%	70%	15%

Table 4: Team Member Contributions

References

[1] S. Antol, A. Agrawal, and J. et al. Lu. Vqa: Visual question answering. In *ICCV Proceedings*, 2015.

[2] A. et al. Dosovitskiy. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint*, 2020.

[3] Ashish Vaswani et al. Attention is all you need. *arXiv preprint*, 2017.

[4] Nitish Srivastava et al. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 2014.

[5] Paulius Micikevicius et al. Mixed precision training. *arXiv preprint*, 2018.

[6] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint*, 2016.

[7] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint*, 2015.

[8] Z. et al. Lai. Veclip: Improving clip training via visual-enriched captions. *arXiv preprint*, 2023.

[9] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint*, 2017.

[10] OpenAI. Clip (contrastive language-image pretraining), 2021. GitHub repository.

[11] A. Radford, J. W. Kim, and C. et al. Hallacy. Learning transferable visual models from natural language supervision. *arXiv preprint*, 2021.

[12] J.-J. Shao, J.-X. Shi, X.-W. Yang, L.-Z. Guo, and Y.-F. Li. Investigating the limitation of clip models: The worst-performing categories. *arXiv preprint*, 2023.

[1]

[1] `https://github.com/Mathio11/CS_228_CLIP_LOCAL_FEATURE_ENHANCEMENT`