Towards Accurate Visual Question Answering with Improved Image-Text Alignment

Javahir Abbasova Hitvarth Diwanji Gurnoor Singh Khurana Somanshu Singla

Abstract

Recent advancements in multimodal AI have opened doors for Visual Language Models that can process the modalities of language and vision simultaneously to perform advanced vision-language tasks. Visual Question Answering (VQA) is a particularly challenging task that requires comprehending an image and answering a natural language question about its content. Achieving high performance in VQA relies heavily on effective image-text alignment. In this work, we propose novel approaches to enhance image-text alignment during the pre-training phase. While these approaches have the potential to improve performance across various downstream tasks, our primary focus is on enhancing VQA. Our code is publicly available at https://github.com/JavahirAbbasova7/VQA

1. Introduction

The task of Visual Question Answering (VQA) requires a model to answer a given textual question based on an input image. The idea of free-form question answering on an image was introduced in [2]. Here, the authors use a VGGNet [22] to get visual features and an LSTM for textual features. With the advent of Transformers [26], it has been possible to scale up architectures for better visual [9] and language understanding [4, 8]. However, striving for better performance in VQA, which deals with both image and text, requires dealing with several limitations in the current works.

First, current state-of-the-art models have been observed to lack spatial reasoning [6]. They find it challenging to answer questions like "Which can is the closest to the hand?". This stems from a lack of fine-grained association of image patches with the question to be answered. Furthermore, the global attention capabilities of vision encoders can miss out on local information that might be crucial in answering spatial reasoning questions.

Another limitation is that the current models are not very parameter-efficient. There have been attempts to reduce the number of parameters by leveraging existing pre-trained networks [24], [1], [15]. However, they are still huge models having billions of parameters, making them difficult to use.

1.1. ALBEF

We decided to base our work on ALBEF [14]. In this work the authors proposed a novel method for aligning vision and language encoder embeddings. They proposed a combination of Image-Text Contrastive, Image-Text Matching and Masked Language Modeling tasks to align the vision and text encoders.

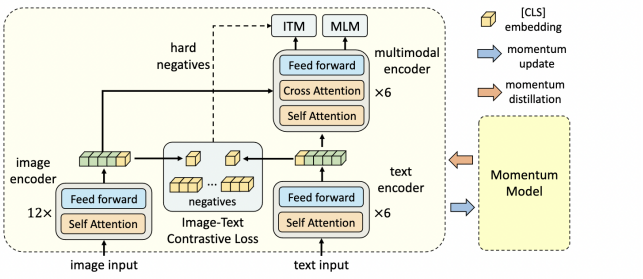

Figure 1. ALBEF

While this work achieved good performance there were several areas we could improve:

- **Image-Text Contrastive Loss**: In ALBEF the authors only use [CLS] token for the contrastive loss and there is a potential to try out other loss functions to improve the alignment.
- **Vision Encoder**: In ALBEF, the vision encoder is a fully transformer based architecture which have been shown in literature to display excellent global reasoning, but a task like VQA also requires local reasoning and is an area where we can improve upon ALBEF.
- **Multimodal Encoder**: The current multimodal encoder takes a lot of data to learn and is not explainable and there could be a potential to improve the data efficiency and explainability of this block.

- **Answer Decoder**: ALBEF treats VQA as an answer generation problem, initializing the answer decoder with pretrained weights from the multimodal encoder. This method achieves good results; however, the model's world knowledge and zero-shot capabilities remain limited.

2. Improvements

Building on our analysis of ALBEF, we will now propose several approaches to overcome all the limitations we listed in the section above.

- **Patch Aligned Contrastive Learning**: We propose Patch Aligned Contrastive Loss to preserve patch level granularity while aligning image-text pairs.
- **Non-Contrastive Loss**: We propose a non-contrastive loss to improve the alignment of vision and text encoders.
- **Local Visual Information**: We propose adding a shallow CNN in the visual encoder pipeline to include local information in the vision embeddings.
- **InFusion**: We propose an InFusion block to replace/support the multimodal encoder block building upon on our idea of explainable and data efficient multimodal encoders.
- **LLM as Decoder**: We propose leveraging Large Language Models (LLMs) as decoders.

We will now discuss all of the approaches and their motivations in detail.

2.1. Patch Aligned Contrastive Learning (PACL)

In this section, we introduce Patch Aligned Contrastive Learning (PACL), a modified similarity function designed to align the patch tokens of the vision encoder and the [CLS] token of the text encoder [18]. We demonstrate that this alignment can be utilized to identify image regions corresponding to the text input, providing a superior alternative to the image-text contrastive loss employed by ALBEF and most Vision-Language Models.

The contrastive training of ALBEF ensures that the [CLS] tokens obtained from vision and text encoder are aligned. The training objective aims to learn a similarity function $s = g_v(v_{\text{cls}})^\top g_w(w_{\text{cls}})$, where v_{cls} is the vision class token, w_{cls} is the text class token, g_v and g_w are linear transformations mapping the [CLS] embeddings to normalized lower-dimensional representations. For each image and text, the softmax-normalized image-to-text and text-to-image similarities are then calculated as follows:

$$p_m^{i2t}(I) = \frac{\exp(s(I, T_m)/\tau)}{\sum_{m=1}^{M} \exp(s(I, T_m)/\tau)}, \tag{1}$$

$$p_m^{t2i}(T) = \frac{\exp(s(T, I_m)/\tau)}{\sum_{m=1}^{M} \exp(s(T, I_m)/\tau)} \tag{2}$$

where M is the number of image-text pairs we store in memory, τ is a learnable temperature parameter. Finally, The image-text contrastive loss follows the InfoNCE [25] formulation and is defined as the cross-entropy H between p and y

$$\mathcal{L}_{\text{itc}} = \frac{1}{2}\mathbb{E}_{(I,T)\sim D}\left[H(y^{i2t}(I), p^{i2t}(I)) + H(y^{t2i}(T), p^{t2i}(T))\right] \tag{3}$$

where the boolean one-hot vectors $y^{i2t}(I)$ and $y^{t2i}(T)$ represent the ground-truth similarity, with the positive pair indicated by a 1 and a 0 for all negatives.

Note that the above loss function produces an alignment between the [CLS] image and text tokens. However, it doesn't achieve the desired granularity at patch level. To train a patch-level alignment, we made several modifications to the contrastive loss. First, we used the patch tokens instead of the [CLS] token from the vision encoder. We mapped each of these tokens to a corresponding lower-dimensional space and calculated the patch-level similarity of each token to the [CLS] token obtained from the text encoder. We normalized the patch-level similarities to the range [0, 1] by applying softmax across all tokens. Next, we took a weighted sum of all vision patch embeddings, where the weights are derived from the patch-level similarities. We normalized this aggregated sum and use it to calculate the image-to-text and text-to-image similarities, as described earlier. These similarities were then used to calculate the InfoNCE loss as previously outlined. We didn't change ALBEF's approach of softening hard targets with soft targets generated by the momentum model.

2.2. Non-contrastive loss

Our baseline, ALBEF, makes use of a contrastive loss to align image and text embeddings before fusion. This loss function is quite robust and effective. However, it requires use of large batch size [7] during training and relies on the presence of good negatives in data.

Prior work [29] proposes using a non-contrastive objective ($\mathcal{L}_{\text{nCLIP}}$) for image-language pretraining, thereby relieving the dependency on negative examples. The idea is to assign each image and text sample to a semantic cluster by projecting the image and text features into a probability space. The objective is to minimize the cross entropy between image and text distributions.

Mathematically, if the features are represented by $f_I = \text{Enc}_I(\text{image})$, $f_T = \text{Enc}_T(\text{text})$, we first pass these features through a projection head to get $g_I := \text{proj}_I(f_I), g_T := \text{proj}_T(f_T)$ and then finally take the softmax to get $p_I = \text{softmax}(g_I), p_T = \text{softmax}(g_T)$. The cross entropy loss is then calculated as

$$\mathcal{L}_{CE} = -p_I \log(p_T) - p_T \log(p_I) \tag{4}$$

Simply minimizing this loss function could lead to collapsing solutions, as the model could learn to always predict the same class. Therefore, entropy minimization and mean entropy maximization regularizers are incorporated [3, 27].

$$\mathcal{L}_{EH} = -p_I^T \log(p_I^T) - p_T^T \log(p_T^T) \tag{5}$$

$$\mathcal{L}_{HE} = \overline{p}_I^T \log(\overline{p}_I^T) + \overline{p}_T^T \log(\overline{p}_T^T) \tag{6}$$

where $\overline{p} = \frac{1}{B}\sum_{i=1}^{B} p_i$ is the average distribution over the batch.

The final loss function is given by

$$\mathcal{L}_{\text{nCLIP}} = \mathcal{L}_{CE} + \lambda_1 \mathcal{L}_{EH} + \lambda_2 \mathcal{L}_{HE} \tag{7}$$

Although the non-contrastive loss mitigates the need for negative samples during training, it negatively impacts the discriminative ability of the model. Our results in section 4.2 demonstrates this. Therefore, we also perform experiments by using a linear combination of the contrastive objective present in ALBEF and the nCLIP objective defined above. Using the terminology presented in [29], we define this loss as $\mathcal{L}_{\text{xCLIP}}$

$$\mathcal{L}_{\text{xCLIP}} = \lambda_{\text{itc}} \mathcal{L}_{\text{itc}} + \lambda_{\text{nCLIP}} \mathcal{L}_{\text{nCLIP}} \tag{8}$$

In our experiments, we observed that the choice of projection layer (proj_I and proj_T) and the choice of hyperparameters (λ_1, λ_2, λ_{itc} and λ_{nCLIP}) has a considerable impact on downstream VQA performance.

2.3. Local visual Information

We also explored the idea of improving the local visual information in image embeddings using a *shallow CNN*.

Formally, in ALBEF [14] we have:

$$\mathcal{E}_I = Enc_I(image) \tag{9}$$

, here $\mathcal{E}_I \in \mathcal{R}^{T \times d}$ are the image token embeddings, T is the number of tokens, d is the internal dimension of the model, Enc_I is the transformer based image encoder.

We update the image embeddings to:

$$\mathcal{E}_I = w_{Enc_I} * Enc_I(image) + w_{cnn} * Enc_{cnn}(image) \tag{10}$$

, here $w_{Enc_I} + w_{cnn} = 1$, are hyperparameters that need tuning and Enc_{cnn} is the shallow CNN model we introduce.

Motivation: The main motivation stems from the fact that CNNs (especially shallow ones) are very good at capturing local information and transformers are known for capturing global information.

We introduced a Local CNN in the visual encoding part of the ALBEF pipeline to achieve embeddings which are locally more aware. We believe that for a task such as VQA [2], local awareness and reasoning are essential for better performance.

Finally, introducing a shallow CNN doesn't change the memory and training cost a lot, so it seemed an idea worth exploring.

2.4. InFusion

Most modern architectures for visual question answering use different forms of multimodal transformers for fusion [12, 14, 28]. Here, we instead explore the idea of leveraging trivial fusion methods like addition, concatenation and element-wise dot product of image and text embeddings. Furthermore, we also inquire if it is possible to make the architectures lighter by introducing such operations.

Next, we encounter the question of which fusion would prove to be most useful. To this end, we take inspiration from the Inception network [23] and design the InFusion block.

The InFusion block, as shown in Fig. 2, takes in a d-dimensional image embedding and a d-dimensional text embedding and performs three operations: addition, concatenation, and element-wise product. The output of concatenation is resized so that all three outputs are of the same size. Then, these three outputs are concatenated and passed through a linear layer to get a fused embedding of dimension d.

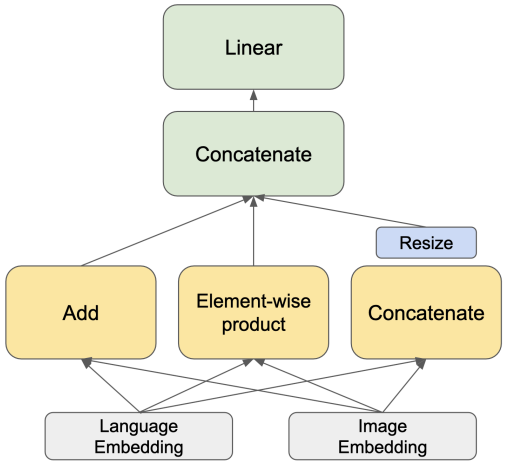

Figure 2. InFusion block

2.4.1 InFusion as an attachment

Here, we think about how we can incorporate the InFusion block along with other architectures like ALBEF [14]. We need to ensure that while doing so, we do not change the

existing architecture so that we can still initialize with the pre-trained weights of the architecture.

We attach the InFusion block to the ALBEF architecture before its multimodal encoder. As suggested in CLIP [20], we use the [CLS] embeddings as the text/image representation. Hence, we pass the [CLS] text embedding and the [CLS] image embedding from the image and text encoders, respectively, to the InFusion block and get the fused embedding f.

$$f = \text{InFusion}([CLS]_{img}, [CLS]_{text}) \tag{11}$$

Now, this fused embedding is added to the rest of the image and text embeddings in a weighted manner. That is, $\forall k \in \{1, ..., seq\_len\}$ we have,

$$\hat{I}_k = \sigma(\theta_{img})f + (1 - \sigma(\theta_{img}))I_k \tag{12}$$

$$\hat{T}_k = \sigma(\theta_{text})f + (1 - \sigma(\theta_{text}))T_k \tag{13}$$

Note that $\sigma(x) = 1/(1 + e^{-x})$ is the sigmoid function and the parameters θ_{img} and θ_{text} are learned. The modified embeddings $\hat{I}_k$ and $\hat{T}_k$ are then passed to the multimodal encoder.

2.4.2 InFusion as an architecture

In this part, we think about the architecture from the ground up, focusing on the InFusion block. Our motivation is to keep the architecture small and have less trainable parameters by leveraging existing pre-trained models.

InFusion requires an image embedding and a text embedding that represents the entire image and text, respectively. Keeping that in mind, we use the pre-trained CLIP encoders [20]. The [CLS] text embedding from CLIP's text encoder and the [CLS] image embedding from CLIP's image encoder become the inputs to the InFusion block.

$$img\_seq = \text{CLIP}_{img}(img) \tag{14}$$

$$text\_seq = \text{CLIP}_{text}(text) \tag{15}$$

$$f = \text{InFusion}([CLS]_{img}, [CLS]_{text}) \tag{16}$$

We then use VisualBERT [16], which serves the purpose of a decoder as well as sharing information between the image tokens and the text tokens. Different from InFusion as an attachment, here, the fused embedding is passed as a distinct embedding to VisualBERT.

$$out\_seq = \text{VisualBERT}(f, img\_seq, text\_seq) \tag{17}$$

We keep the CLIP encoders frozen and train VisualBERT from scratch.

2.5. Large Language Model (LLM) as decoder

We also attempted to enhance performance by employing a Large Language Model (LLM) as the decoder. LLMs are renowned for their strong text generation, zero-shot transfer, and instruction-following capabilities. Our idea was to replace ALBEF's answer decoder with a decoder-based LLM, such as LLama3, inspired by recent research [1, 15, 24]. However, due to limited computational resources, even for smaller LLMs, we were unable to complete this integration. While we could get the pipeline work with LLama2-7B model, due to significant quantization, the model failed to follow instructions properly.

Future work could implement this approach by utilizing an appropriate prompt and enriching it with additional information, such as generated captions. Img2LLM [11] employs a similar approach and leverages GradCam to identify the text-relevant parts of an image. We suggest that the combination of this idea with the new Patch Aligned Contrastive Loss (PACL) could potentially yield better results.

3. Experiments

3.1. Dataset

Our experiments involved a pretraining phase and a finetuning phase. Therefore, we created separate datasets for both.

ALBEF4M model has been pretrained on two web datasets: Conceptual captions [21] and SBU captions [19] and two in-domain datasets: COCO [17] and Visual Genome [13]. Since we used it for initialising our model, we wanted to pretrain on unseen data. Therefore we used a subset of the Conceptual 12M dataset [5] consisting of 33,177 image-caption pairs.

For finetuning on the VQA downstream task, we chose a subset consisting of randomly sampled 20k image-text pairs from the train split of VQAv2 dataset [10].

For evaluation, we used first 5k image-text pairs chosen from the validation split of VQAv2 dataset [10].

3.2. Evaluation

We began by acquiring the ALBEF4M model from checkpoints and further trained it for an additional 15 epochs on the aforementioned pre-training dataset. Subsequently, we fine-tuned the model on the VQA dataset, also mentioned previously, for 8 epochs. The resulting model served as our baseline ALBEF model for comparative analysis. To assess the effectiveness of our proposed ideas against this baseline, we followed identical procedures for each idea. Specifically, we conducted pre-training using our modified code for each idea while maintaining consistent hyperparameters throughout. We evaluated each idea in isolation, without the influence of other ideas, to facilitate clearer conclusions.

3.3. Overall results

Results achieved by each of our ideas can be found in Table 1.

Approach	Accuracy (%)	Trainable params
ALBEF baseline	60.42	210M
PACL	**60.85**	210M
PACL with Nonlinearity	**60.54**	210M
Non-contrastive (nCLIP)	60.33	484M
Non-contrastive (xCLIP)	60.18	484M
Local Visual Information	59.32	372M
InFusion (as attachment)	55.32	293M
InFusion (as architecture)	15.16	27M

Table 1. Comparison of accuracies achieved by different ideas

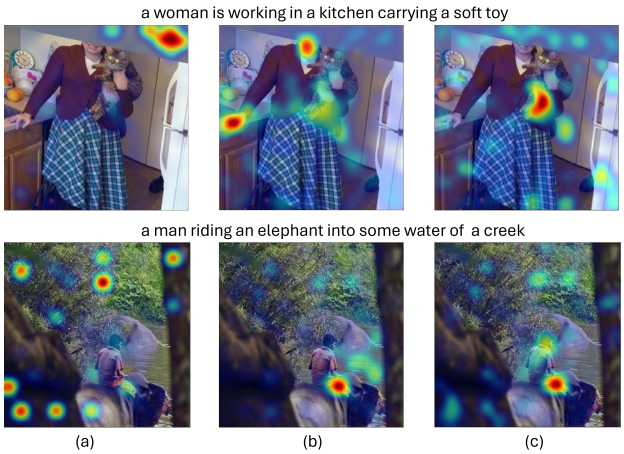

Figure 3. Grad-CAM visualizations on the self-attention maps of the visual encoder: (a) the [CLS] token of the original ALBEF implementation, (b) the weighted average of patch tokens with PACL, (c) the [CLS] token with PACL

4. Insights and Analysis

4.1. PACL

As shown in Table 1, the model utilizing the new patch-aligned loss outperformed the baseline ALBEF model, which employed image-text contrastive loss. While the improvement may seem marginal, it's noteworthy to consider that the difference between the original ALBEF4M and ALBEF14M models is also only 1.3%. This suggests that achieving substantial improvements is generally challenging.

To demonstrate the effectiveness of our new loss function, we computed Grad-CAM visualizations on the self-attention maps in the last layer of the visual encoder, averaging the heatmaps across all attention heads. For the original ALBEF implementation, we computed the gradients with respect to the [CLS] token. In contrast, for the new implementation with PACL, we calculated the gradients of the weighted average of the patch tokens.

As shown in the Figure 3 (b), the weighted average of the patch tokens, where the patches most similar to the given caption have the highest influence, effectively attends to relevant parts of the image. In comparison, the [CLS] tokens of the original loss function fail to identify these relevant parts (Figure 3 (a)). Additionally, we visualize the [CLS] token of the new implementation to demonstrate that the new loss function also improves the learning of the [CLS] tokens (Figure 3 (c)).

4.1.1 Ablation study

To analyze whether a non-linear projection layer improves overall performance, we made a slight modification to our original PACL implementation. Besides the linear transformation to map the patch tokens to a lower dimension, we introduced a smaller neural network comprising two layers separated by a ReLU nonlinearity. The weights of the linear layer were initialized using the pretrained ALBEF model, while the weights of the additional neural network were learned from scratch.

The accuracy on VQA can be found in the Table 1. As shown, the performance with the non-linear projection layer still outperformed the original ALBEF implementation, although it was slightly less than the vanilla PACL. One possible explanation is that the additional neural network was trained for fewer epochs since no pre-trained weights were available for initialization.

Therefore, our conclusions are based on the vanilla PACL, and the impact of non-linearity requires further analysis.

4.2. Non-contrastive loss

As we see in Table 1, the aggregate performance marginally decreases on using a non-contrastive loss. However, we observed that for a particular category of questions, which require a nuanced understanding of specific elements within a scene, the performance improved. Whereas for other questions, which require a discriminative ability (such as categorization questions), the performance deteriorated. Figure 4 provides some examples for the same.

One possible reason for this behavior is that a non-contrastive loss improves the representation capacity of the model, thereby leading to better performance in tasks requiring fine-grained understanding of the scene. This is in agreement with the analysis presented in [29]. However, at the same time, due to lack of a contrastive objective, the model loses its discriminative ability and exhibits poor performance on questions requiring some form of categorization.

In our experiments, we observed that the results are sensitive to the projection layers used. We experimented with

Figure 4. (top) Examples where non-contrastive loss improves performance and (bottom) examples where it deteriorates

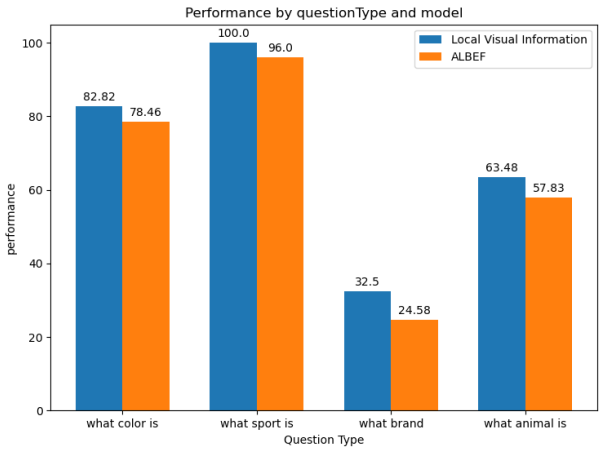

Figure 5. Performance Improvement in 'Vision Intensive' Questions

two variations for proj_I and proj_T. Using a simple linear layer projecting the embeddings to a 512 dimensional space led to a downstream accuracy of 59.23%. As the dimension of output space is increased, the accuracy also increases. As described in [29], we tried using a 2 layer MLP with 4096 hidden layers, GELU activation and 32,768 sized output and that led to a downstream accuracy of 60.33%.

We also observed that the model is highly sensitive to the choice of hyperparameters λ_{EH}, λ_{HE}, λ_{itc}, and λ_{nCLIP}. In our experiments for a purely non-contrastive setting (i.e. $\lambda_{\text{itc}} = 0$), we found that the following values worked best: $\lambda_{\text{EH}} = 0.5, \lambda_{\text{HE}} = -1.5$ and $\lambda_{\text{nCLIP}} = 1$. Moreover, we see that equation 6 requires batch average to calculate the mean entropy. In our experiments, we used a batch size of 16, which may not be sufficient to *approximate* the *mean* of the distribution. A study of variation of accuracy with batch size would be ideal to verify this.

Therefore, even though the performance of our model stays roughly the same as the ALBEF baseline in our experiments, we did observe that both ALBEF and our model work well on a different subset of data. Therefore, with appropriate experimentation with their combination, we might be able to get considerable improvements.

4.3. Local Visual Information

Overall results as seen in Table 1 depict that performance degraded with this approach. Even though the overall performance is not great, we observed some interesting results in certain subclasses of questions and in alignment of vision encoders.

As we can observe in Figure 5, using the Local Visual Information approach we witnessed an improvement in question types such as 'what color is', 'what sport is', 'what brand', 'what animal is'. All these question types are vision intensive and require visual identification and local reasoning about objects in the image. This clearly shows superiority of our method in such tasks.

4.3.1 Lack of Alignment

While the improvement in performance over 'vision intensive' tasks is important, the overall degradation in performance is still concerning and we tried to dig deeper into the probable cause. We tried to look at the 'Alignment Score' of both the vision encoder (ALBEF's transformer based and our shallow CNN). The 'Alignment score' was calculated as follows:

$$\textit{Alignment Score} = \sum_{i=1}^{N} E_{v_i} \cdot E_{t_i} \tag{18}$$

,here we took N image-text pairs, and computed vision embeddings E_{v_i} and text embeddings E_{t_i}, since the image-text pairs are aligned, an aligned visual encoder should give us a high score.

Approach	Alignment Score
ALBEF Vision Encoder	2.36
shallow CNN	1.10

Table 2. Comparison of 'Alignment scores'

The Table 2 shows that the shallow CNN introduced by our approach has not been able to align well when compared

with the ALBEF Vision Encoder and is the probable cause of degradation of performance.

We believe that increasing the training data scale can potentially help us overcome this challenge, but more analysis needs to be done to confirm our hypothesis.

4.4. InFusion

From Table 1, we find that Infusion as an attachment to ALBEF could not outperform ALBEF. However, it might be too early to conclude that the fused embedding would not be able to enhance performance.

First of all, we find that the training process is heavily dependent on the initialization of the parameters θ_{img} and θ_{text}. When initialized randomly, the accuracy drops to just 36%. This is because if the fused embedding is more dominant than the embeddings I_k and T_k, then the multimodal encoder will receive random signals initially, resulting in the optimization process going in unwanted directions. Hence, we initialize the weights θ_{img} and θ_{text} such that $\sigma(\theta_{img}) = \sigma(\theta_{text}) = 0.01$. The evolution of the weight of the fused embedding during pretraining and fine-tuning can be seen in Table 3. We observe that even when the weight of the fused embedding is initialized to a low value of 0.01, it increases up to 0.12, indicating that the fused embedding might be important.

Second, it might not be the best idea to add the fused embedding to every other image and text token. Passing the same signal through all the embeddings might be redundant, especially when the multimodal encoder can reason based on all the input embeddings.

	Initial	After Pretraining	After Fine-tuning
$\sigma(\theta_{img})$	0.010	0.111	0.119
$\sigma(\theta_{text})$	0.010	0.112	0.120

Table 3. Evolution of weight of the fused embedding f when using InFusion as an attachment

For InFusion as an architecture, we see in Table 1 that its performance is not up to the mark. There can be several reasons for this. First, as shown in Table 1, we can see that it has a very small number of trainable parameters compared to other methods. This is because we only train the VisualBERT, keeping the CLIP encoders frozen. Furthermore, even the chosen VisualBERT model is not very large. It has 6 transformer layers with 8 attention heads. We should also note that VisualBERT is trained entirely from scratch, whereas other methods use pre-trained weights. Hence, it is not easy for a small model to capture enough information when trained on a subset of the dataset for just 15 epochs. Another possible reason might be the usage of CLIP encoders. CLIP encoders are not specialized for the task of visual question answering. On top of that, it is kept frozen. This might result in inferior quality of the extracted text and image features.

For InFusion as an attachment, we also analyze the weight of the linear layer of the InFusion block. We find that the importance of the Add, Concatenate, and Element-wise product block is roughly the same. However, if we just look at the top 10 important neurons, then 70% of those neurons come from the Add block.

4.4.1 Next steps

Considering the arguments presented above, here are the next steps we intend to take to improve this idea.

When using InFusion as an attachment, we observed that the training is heavily dependent on the initialization of θ_{img} and θ_{text}. We would try freezing these parameters for alternate epochs so that the rest of the model can get used to the updated weight of the fused embedding. This should result in more stable training.

One other idea is to have different weight parameters θ^k_{img} and θ^k_{text} for kth image and text embedding, respectively.

We can also try adding the fused embedding to just the [CLS] tokens of image and text instead of all the tokens.

For InFusion as an architecture, there are a lot of directions to explore. Apart from increasing the size of the decoder or trying different decoders, we can try training the model while keeping parts of the CLIP encoders trainable. Or we can replace the CLIP encoder altogether with encoders from ALBEF.

We can take inspiration from BLIP2 [15] and have the Q-former bridge the gap between a pre-trained image encoder and an LLM. The InFusion block will be placed between the Q-former and the fully connected layer.

Finally, we can also design or incorporate new fusion techniques other than the currently used "Add", "Concatenate", and "Product" blocks. It would be interesting to try out some lightweight parametric non-linear functions.

5. Conclusion

In this work, we proposed several approaches to address the limitations of current Vision Language Models for VQA, focusing specifically on the ALBEF model. Despite limited resources and without increasing the model size, we successfully outperformed the baseline model with one of our approaches. We conducted a detailed analysis for all our approaches and outlined potential future directions. Future research could involve training our proposed methods to convergence using more computational resources to validate our findings.

5.1. Future Works

- Combining the proposed approaches to jointly train a model presents a promising opportunity for further improvement.
- Ensembling the models learned through current approaches can potentially improve performance.
- Leveraging LLMs to improve the grounding, logical reasoning ability of the decoder.

References

[1] Jean-Baptiste Alayrac, Jeff Donahue, Pauline Luc, Antoine Miech, Iain Barr, Yana Hasson, Karel Lenc, Arthur Mensch, Katie Millican, Malcolm Reynolds, Roman Ring, Eliza Rutherford, Serkan Cabi, Tengda Han, Zhitao Gong, Sina Samangooei, Marianne Monteiro, Jacob Menick, Sebastian Borgeaud, Andrew Brock, Aida Nematzadeh, Sahand Sharifzadeh, Mikolaj Binkowski, Ricardo Barreira, Oriol Vinyals, Andrew Zisserman, and Karen Simonyan. Flamingo: a visual language model for few-shot learning, 2022. 1, 4

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 2425–2433, 2015. 1, 3

[3] Mahmoud Assran, Mathilde Caron, Ishan Misra, Piotr Bojanowski, Florian Bordes, Pascal Vincent, Armand Joulin, Michael Rabbat, and Nicolas Ballas. Masked siamese networks for label-efficient learning, 2022. 3

[4] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. 1

[5] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts, 2021. 4

[6] Boyuan Chen, Zhuo Xu, Sean Kirmani, Brian Ichter, Danny Driess, Pete Florence, Dorsa Sadigh, Leonidas Guibas, and Fei Xia. Spatialvlm: Endowing vision-language models with spatial reasoning capabilities, 2024. 1

[7] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey E. Hinton. A simple framework for contrastive learning of visual representations. *CoRR*, abs/2002.05709, 2020. 2

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding, 2019. 1

[9] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. 1

[10] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[11] Jiaxian Guo, Junnan Li, Dongxu Li, Anthony Meng Huat Tiong, Boyang Li, Dacheng Tao, and Steven Hoi. From images to textual prompts: Zero-shot visual question answering with frozen large language models. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10867–10877, 2023. 4

[12] Zhicheng Huang, Zhaoyang Zeng, Yupan Huang, Bei Liu, Dongmei Fu, and Jianlong Fu. Seeing out of the box: End-to-end pre-training for vision-language representation learning, 2021. 3

[13] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A. Shamma, Michael S. Bernstein, and Li Fei-Fei. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *CoRR*, abs/1602.07332, 2016. 4

[14] Junnan Li, Ramprasaath R. Selvaraju, Akhilesh Deepak Gotmare, Shafiq Joty, Caiming Xiong, and Steven Hoi. Align before fuse: Vision and language representation learning with momentum distillation. In *NeurIPS*, 2021. 1, 3

[15] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models, 2023. 1, 4, 7

[16] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language, 2019. 4

[17] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, Lubomir D. Bourdev, Ross B. Girshick, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. *CoRR*, abs/1405.0312, 2014. 4

[18] Jishnu Mukhoti, Tsung-Yu Lin, Omid Poursaeed, Rui Wang, Ashish Shah, Philip H.S. Torr, and Ser-Nam Lim. Open vocabulary semantic segmentation with patch aligned contrastive learning. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 19413–19423, 2023. 2

[19] Vicente Ordonez, Girish Kulkarni, and Tamara Berg. Im2text: Describing images using 1 million captioned photographs. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2011. 4

[20] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision, 2021. 4

[21] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, im-

age alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, Melbourne, Australia, 2018. Association for Computational Linguistics. 4

[22] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition, 2015. 1

[23] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions, 2014. 3

[24] Maria Tsimpoukelli, Jacob Menick, Serkan Cabi, S. M. Ali Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models, 2021. 1, 4

[25] Aäron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748, 2018. 2

[26] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. 1

[27] Feng Wang, Tao Kong, Rufeng Zhang, Huaping Liu, and Hang Li. Self-supervised learning by estimating twin class distributions. *CoRR*, abs/2110.07402, 2021. 3

[28] Zirui Wang, Jiahui Yu, Adams Wei Yu, Zihang Dai, Yulia Tsvetkov, and Yuan Cao. Simvlm: Simple visual language model pretraining with weak supervision, 2022. 3

[29] Jinghao Zhou, Li Dong, Zhe Gan, Lijuan Wang, and Furu Wei. Non-contrastive learning meets language-image pre-training, 2022. 2, 3, 5, 6