applied sciences

MDPI

Article

An Effective Med-VQA Method Using a Transformer with Weights Fusion of Multiple Fine-Tuned Models

Suheer Al-Hadhrami [1,2], Mohamed El Bachir Menai [1], Saad Al-Ahmadi [1] and Ahmad Alnafessah [3,*,†]

[1] College of Computer and Information Sciences, King Saud University, P.O. Box 2614, Riyadh 13312, Saudi Arabia; 437204437@student.ksu.edu.sa or s.alhadhrami@hu.edu.ye (S.A.-H.); menai@ksu.edu.sa (M.E.B.M.); salahmadi@ksu.edu.sa (S.A.-A.)
[2] Computer Engineering Department, Hadhramout University, Al Mukalla 10587, Yemen
[3] King Abdulaziz City for Science and Technology, Riyadh 11442, Saudi Arabia
* Correspondence: aalnafessah@kacst.edu.sa or nafessah@mit.edu
† Current address: Sociotechnical Systems Research Center, Massachusetts Institute of Technology, Massachusetts Ave, Cambridge, MA 02139, USA.

Abstract: Visual question answering (VQA) is a task that generates or predicts an answer to a question in human language about visual images. VQA is an active field combining two AI branches: NLP and computer vision. VQA in the medical field is still at an early stage, and it needs vast efforts and exploration to reach practical usage. This paper proposes two models that are utilized in the latest vision and NLP transformers that outperform the SOTA and have not yet been utilized in medical VQA. The ELECTRA-base transformer is used for textual feature extraction, whereas SWIN is used for visual feature extraction. In the SOTA medical VQA, selecting the model is based on the model that achieves the highest validation accuracy or the last model in training. The first proposed model, the best-value-based model, is selected based on the highest validation accuracy. The second model, the greedy-soup-based model, uses a greedy soup technique based on the fusion of multiple fine-tuned models to set the model parameters. The greedy soup selects the model parameters by fusing the model parameters that have significant performance on the validation accuracy in training. The greedy-soup-based model outperforms the best-value-based model, and both proposed models outperform the SOTA, which has an accuracy of 83.49%. The greedy-soup-based model is optimized with batch size and learning rate. During the optimization, seven extra models exceed the SOTA accuracy. The best model trained with a learning rate of 1.0×10^{-4} and batch size 16 achieves an accuracy of 87.41%.

Keywords: VQA; medical; NLP; vision; transformer; greedy soup; visual question answering; SWIN; ELECTRA; classification

Citation: Al-Hadhrami, S.; Menai, M.E.B.; Al-Ahmadi, S.; Alnafessah, A. An Effective Med-VQA Method Using a Transformer with Weights Fusion of Multiple Fine-Tuned Models. *Appl. Sci.* **2023**, *13*, 9735. https://doi.org/10.3390/app13179735

Academic Editor: Byung-Gyu Kim

Received: 23 June 2023
Revised: 27 July 2023
Accepted: 9 August 2023
Published: 28 August 2023

1. Introduction

Visual question answering (VQA) is a process that provides meaningful information from images to a user based on a given question. With the rapid advancements in computer vision (CV) and naturalanguage processing (NLP), medical visual question answering (Med-VQA) has attracted much attention. The Med-VQA model seeks to retrieve accurate answers by fusing clinical visuals and inquiries. The Med-VQA model can aid in medical diagnosis, automatically extract data from medical images, andower medical professionals' training costs. Additionally, the Med-VQA system has many benefits for the medical industry. Here are a few illustrations:

- **Diagnosis and treatment:** By offering a quick and precise method for analyzing medical images, Med-VQA can help medical practitioners diagnose and treat medical disorders. Healthcare experts canearn more about the patient's condition by inquiring about medical imaging (such as X-rays, CT scans, and MRI scans), which can aid in making a diagnosis and selecting the best course of therapy. In addition, it would

reduce the doctors' efforts byetting the patients obtain answers to the most frequent questions about their images.

- **Medical education:** By giving users a way toearn from medical images, Med-VQA can be used to instruct medical students and healthcare workers. Students canearn how to assess and understand medical images—a crucial ability in the area of medicine—by posing questions about them.
- **Patient education:** By allowing patients to ask questions about their medical photos, Med-VQA can help them better comprehend their medical issues. Healthcare practitioners can improve patient outcomes by assisting patients in understanding their problems and available treatments by responding to their inquiries regarding their medical photos.
- **Research:** arge collections of medical photos can be analyzed using Med-VQA to glean insights that can be applied to medical research. Researchers can better comprehend medical issues and create new remedies by posing questions regarding medical imaging and examining the results.

Although much Med-VQA research has been accomplished, it requires much enhancement due to public data and data sizeimitations to achieve practical usage. There are a few public data available: the VQA-RAD [1], VQA-Med 2019 [2], VQA-MED 2020 [3], SLAKE dataset [4], and Diabetic Macular Edema (DME) dataset [5]. Only the VQA-RAD and SLAKE datasets are manually generated and validated by clinicians. In addition, they have more question diversity among all medical VQA datasets. The DME dataset is manually generated but not validated by specialists.

Several models have been developed to solve this problem. These models rely on four types of methods: joint embedding approaches [6–8], attention mechanisms [9–13], composition models [12,14–18], and knowledge base-enhanced approaches [1,19–21]. In Med-VQA, VGGNet [22], ResNet [23], and the ensemble of vision pre-trained models [24,25] are the vision features extraction methods widely used, whileSTM [26], Bi-LSTM [27], and BERT [28] are the text features extraction mainly utilized.ately, most models have aimed to use attention mechanisms to align between the text and image features [10,29–31]. In addition, vision andanguage (V + L) pre-trained models, such as visualBERT [32], VilBERT [33], UNITER [34], and CLIP [35]. Researchers claimed that Med-VQA requires more text information about images to facilitate the classification task for the model. Therefore, they utilize image captioning generation to give the model extra information about the image [36].

Regarding the state-of-the-art (SOTA), the multi-model structure's complex has been increased to obtain better performance than previous models. This is because deepearning is a black-box method and the VQA multi-model has three main parts: text phase, vision, phase, and fusion, in which no researchers have proven that a method used in each part is the most significant in VQA behavior. For example, utilizing VGGNet to obtain vision features in a model achieved high performance, but it does not guarantee that another model with the same structure will achieve as high performance as the first model's due to the difference in the hyper-parameters configuration. Medical VQA is still at an early stage and needs more exploration to develop efficient models for practical application, which requires a profound understanding of theimitations in this field. A big issue in Med-VQAies in dataimitation. Although most medical VQA researchers employed a CNN pre-trained vision model to solve the dataimitation, this does not reduce the variation between the feature scales of visual entities with a high pixel resolution and the text words. Using transformers for vision manipulates the images as text and reduces this variation [37]. Therefore, this paper uses the transformer for the text and vision feature extraction process. In addition, text transformers, such as BERT and BioBERT, are widely used for textual feature extraction, but the ELECTRA transformer (Efficientlyearning an Encoder that Classifies Token Replacements Accurately), which outperforms the SOTA text transformer, is not utilized. Various researchers utilize ensembleearning to obtain the efficiency of different models. Wortsman et al. [38] designed another method called model soup, which has three types: uniform, greedy, andearnable. The greedy soup

achieved the highest performance among the three types. The authors discovered that the SOUPS architecture offers several advantages over traditional ensemble models, including improved accuracy, generalization, interpretability, and more efficient computation. The computation cost for ensemble k models is $O(k)$, whereas the computation cost of greedy soup is $O(1)$. Therefore, in this paper, the proposed model uses the greedy soup method to obtain those advantages. Overall, the contributions of this paper can be summarized as follows:

- Since the ELECTRA-base transformer that is pre-trained on aarge corpus of text data using contrastive estimation and outperforms other text transformers, such as BERT and BioBERT, is not utilized yet in medical VQA for the textual feature extraction, the proposed model exploits the ELECTRA-base transformer to extract text features from the question.
- The proposed model aims to solve the issue of thearge feature variation between the question and the image. Therefore, it exploits theast vision transformer, swin-base-patch4-window7-224. The SWIN transformer utilized in this model to extract visual features from the image is used for the first time in medical VQA.
- The extracted visual and textual features are combined and fed into an MLP for classification.
- The proposed model is fine-tuned based on the model parameters that achieved the highest validation accuracy and the greedy soup approach to show the significant impact of the greedy soup method.
- The performance of theast two models are compared with the model by Tascon et al. [5], which we denoted as the SOTA because it is the unique research on this dataset.
- The model based on the greedy soup fine-tuning technique is optimized to select the bestearning rate and batch size.

The remainder of this paper is organized as follows. The related work is summarized in Section 2. Section 3 presents the proposed methods, while Section 4 discusses and analyzes the model performance, utilized dataset, and environment setup. Section 5 provides the conclusions and future research direction.

2. Related Work

VQA usually has four components: vision featurization, text featurization, fusion model, and classifier. Vision featurization is a part of the multi-model responsible for extracting the vision features. Text featurization is another part of the VQA multi-model responsible for extracting text features. The combination of both features and their processes is the fusion component. Theast component is the classifier that classifies the queries about the images and generates the answer.

2.1. Vision Featurization

Applying mathematical operations to an image requires representing it as a numerical vector, called image featurization. There are several techniques to extract the features of the image, such as scale-invariant feature transform (SIFT) [39], simple RGB vector, a histogram of oriented gradients (HOG) [40], Haar transform [41], and deepearning. In deepearning, such as CNNs, visual feature extractionearns using a neural network. Using deepearning can be accomplished by training the model from scratch, which requires aarge data size, or using transferearning, which behaves significantly with aimited data size. Since medical VQA datasets areimited, most researchers aim to use per-trained models, such as AlexNet [42], VGGNet [22,43–46], GoogLeNet [47], ResNet [5,23,48–52], and DenseNet-121 [53]. Ensemble models can be stronger than the single model, so there is a direction to use it as vision feature extraction [25,54–57].

2.2. Text Featurization

As a vision featurization, a question has to be converted into a numeric vector using word-embedding methods for mathematical computations. A suitable text embedding

method is based on trial and error [58]. Various text embedding methods are used in the SOTA to impact the multi-model significantly. The most common methods used in question models areSTM [5,53,56,57,59], GRU [59], RNNs [44,60–62], Faster-RNN [59], and the encoder–decoder method [45,48–50,63,64]. In addition to the previous methods, pre-trained models have been used, such as Generalized Auto-regressive Pre-training foranguage Understanding (XLNet) [65] and the BERT model [28,45,51,52]. Some models have ignored text featurization and converted the problem into an image classification problem [55,66,67].

2.3. Fusion

Extracting the features of text and images is processed independently. Therefore, those features are fused using the fusion method. Manmadhan et al. [58] classified the fusion into three types: baseline fusion models, end-to-end neural network models, and joint attention models. In baseline fusions, various methods are used, such as element-wise addition [7], element-wise multiplication, concatenation [68], all of them combined [69], or a hybrid of these methods with a polynomial function [70]. End-to-end neural network models can be used to fuse image and text featurization. Various methods are currently used, including neural module networks (NMNs) [12], multimodal, MCB [48], dynamic parameter prediction networks (DPPNs) [71], multimodal residual network (MRNs) [72], cross-modal multistep fusion (CMF) networks [73], basic MCB model with a deep attention neural tensor network (DA-NTN) module [74], multi-layer perceptron (MLP) [75], and the encoder–decoder method [32,76]. The main reason for using the joint attention model is to address the semantic relationship between text attention and question attention [58]. There are various joint attention models, such as the word-to-region attention network (WRAN) [29], co-attention [30], the question-guided attention map (QAM) [10], and question type-guided attention (QTA) [31].

Neural network methods such asSTM and encoder–decoder are also used in the fusion phase. Verma and Ramachandran [45] designed a multi-model that used encoder–decoder,STM, and GloVe. Furthermore, vision + language pre-trained models are also utilized, such as in [52].

In the VQA system, the question and image are embedded separately using one or a hybrid of text and vision featurization techniques mentioned above. Then the textual and visual feature victors are combined with a fusion technique, such as concatenation, element-wise multiplication, or attention. The obtained victor from the fusion phase is classified using a classification technique, or it can be used to generate an answer as a VQA generation problem. Figure 1 shows the overall VQA system.

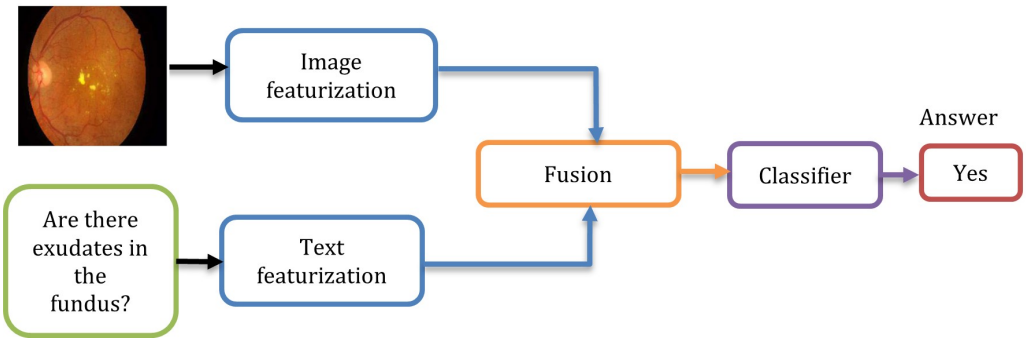


Figure 1. The Overall VQA Structure.

3. Methodology

This paper proposes two VQA models that aim to improve the SOTA performance by fusing fine-tuned models and utilizing the most recent techniques for vision and textual feature extraction and reducing the feature variance between the textual and visual using the same techniques for both, namely, the use of transformers. The SWIN-base transformer,

ELECTRA-base transformer, and MLP are used for vision featurization, text featurization, and classification for the base model. The first model, the best-value-model, uses the data to fine-tune the base model, whereas the second model, the greedy-soup-model, fuses several fine-tuned based models based on the greedy soup technique, which creates the final model weights based on the fusion of the weights of the best three fine-tuned models, which significantly impacts the validation accuracy. Figure 2 shows the architecture of the base model with greedy soups technique fusion. More details about each phase are discussed below.

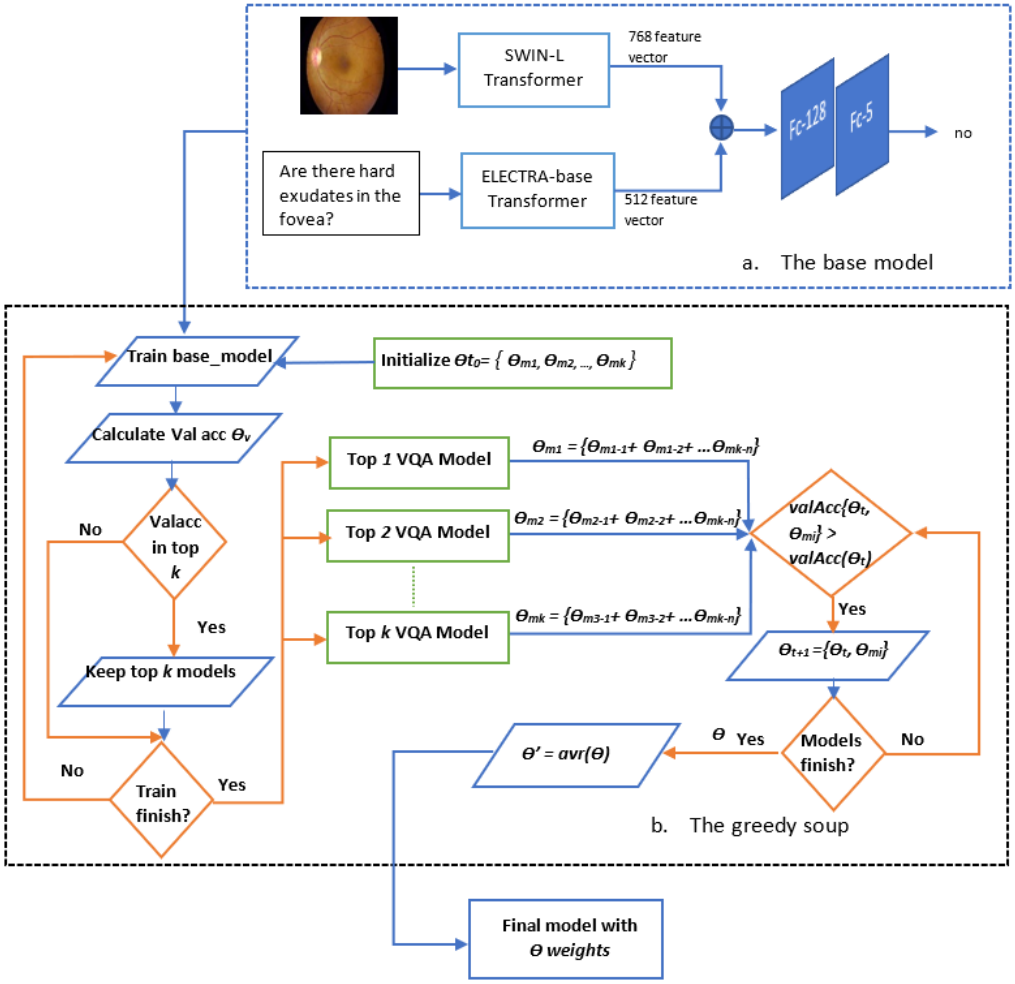


Figure 2. The Overall model Structure.

3.1. SWIN Transformer

A transformer is utilized for vision featurization to reduce the variation between the text and vision features by manipulating the image bitches as tokens. The model uses the SWIN transformer to reduce the memory and computation resources required to process high-resolution images by dividing the image into non-overlapping patches and processing each patch using a set of shifted windows that capture different spatial relationships between the patches, as shown in Figures 3 and 4. The shifted windows allow the model to capture bothocal and global features of the image, and the hierarchical processing enables the model to handle high-resolution images more efficiently. We utilized SWIN-base-patch4-window7, which is the most extensive configuration of the SWIN transformer, which consists of 6 stages and 2 SWIN transformer blocks per stage. This model has 197.0 million parameters. The process of feature extraction using the SWIN transformer is described below.

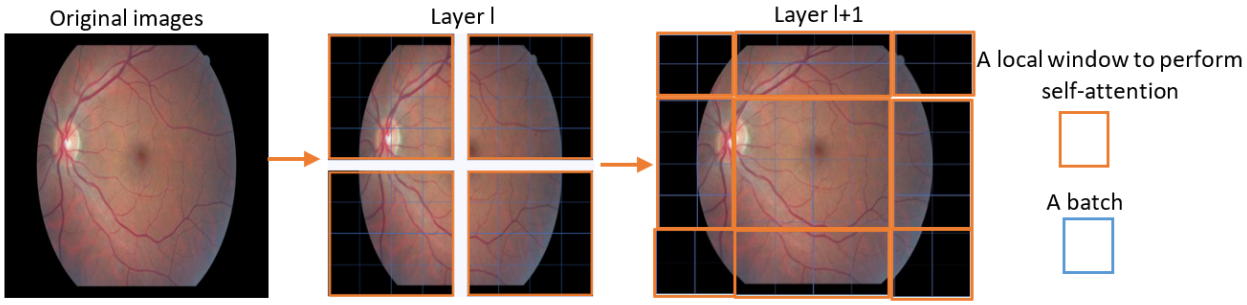


Figure 3. The SWIN Shift window Approach.

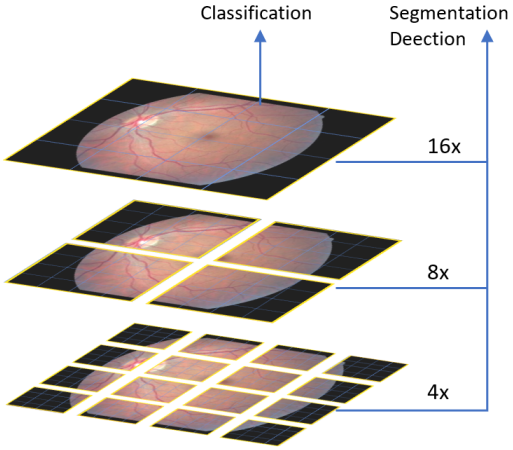


Figure 4. The SWIN Model Architecture.

- PatchEmbedding: The input image is divided into non-overlapping patches, which are flattened into vectors and passed through ainearayer to obtain a set of patch embeddings. The patch embeddings are denoted as X.

$$X = \text{PatchEmbed}(I) \tag{1}$$

- SWIN Transformer Blocks: The SWIN transformer block processes the feature maps of the input patches using a set of shifted windows, producing a set of output feature maps, which are then processed by a SWIN transformerayer that combines the feature maps across the patches and produces a set of feature maps at a higher resolution.

$$X_{i,j} = \text{Swin-L Block}(X_{i,j-1}, X_{i,j+1}, X_{i-1,j}, X_{i+1,j}) \tag{2}$$

  where $X_{i,j}$ denotes to the patch in row i and column j.
- SWIN Transformerayers: The output feature maps from the SWIN transformer blocks are processed by a SWIN transformerayer that combines the feature maps across the patches and produces a set of feature maps at a higher resolution.

$$X'i, j = \text{Swinayer}(Xi, j) \tag{3}$$

  where $X_{i,j}$ denotes to the patch in row i and column j.
- Global Average Pooling: The output feature maps from the final SWIN transformerayer are passed through a global average poolingayer, which computes the average of each feature map across the spatial dimensions and produces a final feature vector.

$$V = \text{GlobalAvgPool}(X'_{i,j}) \tag{4}$$

This final feature vector V size is 2048.

3.2. ELECTRA Transformer

The proposed model utilizes ELECTRA, a textual pre-trained transformer model trained on aarge corpus of text for textual featurization [77]. The main advantage of ELECTRA is designing a novel pre-training approach called contrastive estimation. Unlike other pre-training approaches, such as maskedanguage modeling used in BERT, ELECTRA trains the model to distinguish between real and replaced tokens in an input sentence. This method encourages the model toearn more effectively from the surrounding context and improves its ability to capture the nuances of theanguage,eading to better performance on downstream tasks. Furthermore, the ELECTRA transformer offers several advantages over other transformer models. Apart from improving efficiency, it has faster times, robustness to noise and adversarial attacks, and is adaptable to a wide range of naturalanguage processing tasks [77]. Figure 5 shows the replaced token detection overview. The contrastive estimation can be expressed with the following equations:

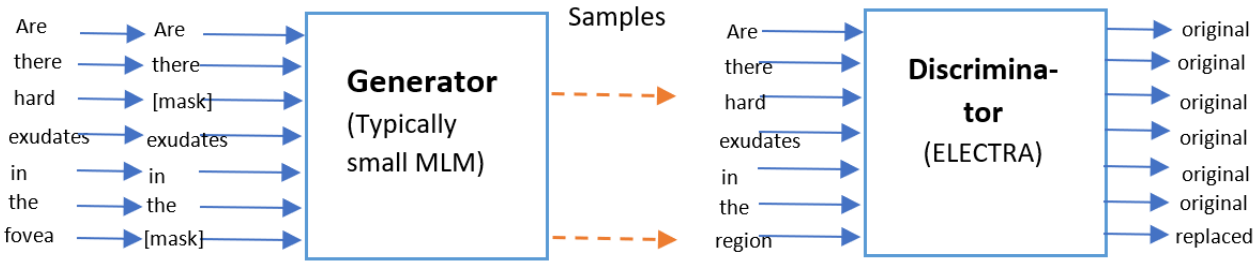


Figure 5. Replacement of the token detection overview.

Given an input sentence x, a set of replaced sentences x', and a binaryabel y (where $y = 1$ if x' is a true replacement of x, and $y = 0$ otherwise), the contrastiveoss function, which is a type of cross-entropy (CE), is defined as:

$$_{CE} = -\frac{1}{N}\sum_{i=1}^{N} y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i) \tag{5}$$

where N, y_i, and $\hat{y}_i$ are the number of training examples, the trueabel of example i, and the predictedabel of example i, respectively. The predictedabel $\hat{y}_i$ is computed as:

$$\hat{y}_i = \frac{\exp(f(x_i, x_i'))}{\exp(f(x_i, x_i')) + \sum_{j \neq i} \exp(f(x_i, x_j'))} \tag{6}$$

where the similarity score between the input sentence x_i and the corresponding replaced sentence is x_i' $f(x_i, x_i')$. The similarity score is computed using a neural network based on the concatenated input, replaced sentences as input, and produces a scalar output. The denominator in the predictedabel expression normalizes the scores across all replaced sentences for a given input sentence. During training, the model is presented with pairs of input sentences containing real tokens and replaced tokens. The model is trained to predict whether the tokens replaced in the second sentence were replaced based on the surrounding context. This method encourages the model to effectivelyearn from theanguage and improve its understanding of the nuances of theanguage.

The ELECTRA transformer consists of 12ayers in its transformer encoder. Eachayer contains 12 attention heads and a hidden size of 768. The model total parameters is 110 million. The input to the ELECTRA-base transformer is a sequence of tokens, which are first passed through an embeddingayer to obtain a sequence of dense vectors. The transformer encoderayers then process these vectors to extract contextualized representations of the input text. Theength of the features produced by the ELECTRA-base transformer is 512. The feature extraction processes are described as follows:

- Token Embedding: The input text T is tokenized, and each token is mapped to its corresponding embedding vector. The token embeddings are denoted as X.

$$X = \text{TokenEmbedding}(T) \tag{7}$$

- Encoderayers: The ELECTRA transformer uses a stack of encoderayers to process the token embeddings and extract contextualized representations of the input text. Each encoderayer consists of a self-attention mechanism followed by a feedforward network.

$$X' = \text{Encoder}(X) \tag{8}$$

- Maskedanguage Modeling: During pre-training, a subset of the input tokens are randomly masked, and the model is trained to predict the original tokens based on the masked tokens. The masked tokens are denoted as M.

$$M = \text{Mask}(T) \tag{9}$$

- Masked Token Prediction:
  The output of the encoderayers is used to predict the original tokens based on the masked tokens. The masked token prediction can be expressed as follows:
  - First, the output of the encoderayers is passed through ainearayer to obtain a set ofogits for each token position.

$$= \text{Linear}(X') \tag{10}$$

  - Then, theogits corresponding to the masked token positions are selected and passed through a softmax function to obtain a probability distribution over the vocabulary.

$$P = \text{Softmax}(L_M) \tag{11}$$

  - Finally, the model is trained to maximize theog-likelihood of the original tokens given the masked tokens and the predicted probability distribution.

$$\mathcal{L} = \sum_{i=1}^{n} \log(P_{i,t_i}) \tag{12}$$

    where n and t_i are the number of masked tokens and the index of the original token at position i, respectively.

3.3. MLP Fusion

A Multilayer Perceptron (MLP) is a feedforward neural network consisting of multipleayers of perceptrons (also called neurons) [78]. Each perceptron in the network computes a weighted sum of its input features and applies an activation function to produce an output. The output of each perceptron is then fed into the nextayer of perceptrons until the finalayer, which produces the final output of the network. The proposed model consists of two layers.

The equation for the output of a single perceptron can be expressed as:

$$y = \sigma(\sum_{i=1}^{n} w_i x_i + b) \tag{13}$$

where y is the output of the perceptron, x_i is the ith input feature, w_i is the weight associated with the ith input feature, b is the bias term, and σ is the activation function.

The output of each perceptron in aayer is computed in parallel, and the output of the entireayer is a vector of outputs. The output of aayer is then fed as input to the next layer.

The MLP network can have one or more hiddenayers, where each hiddenayer consists of multiple perceptrons. The number of hiddenayers and their perceptrons are hyperparameters that can be tuned to optimize the performance of the network.

The equation for the output of an MLP network with one hiddenayer can be expressed as:

$$y = \sigma_2(\sum_{i=1}^{h} w_{2,i}\sigma_1(\sum_{j=1}^{n} w_{1,j,i}x_j + b_{1,i}) + b_2) \tag{14}$$

where y is the final output of the network, x_j is the j-th input feature, $w_{1,j,i}$ is the weight associated with the connection between the j-th input feature and the i-th perceptron in the hiddenayer, $w_{2,i}$ is the weight associated with the connection between the i-th perceptron in the hiddenayer and the final output, $b_{1,i}$ is the bias term associated with the i-th perceptron in the hiddenayer, b_2 is the bias term associated with the final output, σ_1 is the activation function for the hiddenayer, and σ_2 is the activation function for the final output.

The MLP network can be trained using backpropagation, which is an iterative algorithm that adjusts the weights and biases of the network to minimize aoss function that measures the difference between the predicted output and the true output. The weights and biases are updated using the gradient of theoss function with respect to the weights and biases.

The proposed model consists of two hiddenayers with theeak-ReLu activation function [79]. This is followed by dropayer with a rate of 0.1 and a normalizationayer. The firstayer contains 128 nodes, whereas the secondayer includes 5 nodes, which is the number of classes. Theeak-ReLu activation is calculated using the equation:

$$f(x) = \begin{cases} x & \text{if } x \geq 0 \\ \alpha x & \text{otherwise} \end{cases} \tag{15}$$

where α is a small positive constant that determines the slope of the function for negative input values. The default value of α is 0.01.

The cross-entropyoss that measures the difference between the predicted probabilities of the classes and the trueabels is used in the designed model. The cross-entropyoss function is defined as:

$$(y, \hat{y}) = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{C} y_{ij}\log(\hat{y}_{ij}) \tag{16}$$

where y_{ij} is the trueabel for the j-th class of the i-th sample (either 0 or 1), $\hat{y}_{ij}$ is the predicted probability of the j-th class for the i-th sample, N is the number of samples, and C is the number of classes.

The first sum in the equation computes theoss for each sample, while the second sum computes theoss for each class. When the predicted probability is close to 1 and the trueabel is 1, theoss is close to 0. Theoss is significant when the predicted probability is close to 0 and the trueabel is 1. Theog function is used to penalize the model more heavily for predictions that are far from the actualabel.

The softmax function, which is used to transform the output of a neural network into a probability distribution over the classes, is used. The softmax function is defined as:

$$\sigma(\mathbf{z})_j = \frac{e^{z_j}}{\sum_{k=1}^{K} e^{z_k}} \tag{17}$$

where $\mathbf{z}$ is the output of the neural network for a given input, K is the number of classes, and $\sigma(\mathbf{z})_j$ is the predicted probability of the j-th class.

Weight decay is a common regularization technique in machineearning that penalizesarge weights and encourages the model toearn more superficial data representations. The model utilized Adam with a weight decay regularization (AdamW) optimization function [80], an improvement over the popular Adam optimization algorithm. The

AdamW algorithm is similar to the original Adam algorithm but with the addition of weight decay regularization. In the original Adam algorithm, weight decay is applied to both the weights and theearning rate. Therefore, this canead to suboptimal performance, especially in deep-learning models with manyayers. The AdamW algorithm solves this problem by applying weight decay only to the weights and not to theearning rate. The AdamW algorithm is implemented by adding a weight decay term to the gradient update rule:

$$w_t = w_{t-1} - \eta_t \times (m_t/(sqrt(v_t) + eps) + \lambda \times w_t) \quad (18)$$

where w_t is the weight at time t, η_t is theearning rate at time t, m_t and v_t are the first and second moments of the gradients, eps is a small constant to prevent division by zero, and λ is the weight decay coefficient.

By applying weight decay only to the weights, the AdamW algorithm can improve the performance of deepearning models, especially in cases where weight decay significantly impacts the model's performance.

3.4. SOUPS Fusion Method

- **Greedy soups**
  The greedy soups technique is used to fuse the parameters of multiple fine-tuned models, where the model is trained, and validation accuracy is calculated several times during training; the final parameters will be the average of the best k models' parameters based on the fine-tuned model weights that significantly improve the validation performance by averaging the weights of the new fine-tuned model with theist of previous models weights. The proposed model utilizes this technique to maximize its performance.
  Let $M = \{m_1, m_2, \ldots m_n\}$ and $\theta = \{\theta_0, \theta_1, \ldots, \theta_n\}$ denote the set of models and their parameters, respectively.et $sg = \{\theta_0, \theta_1, \ldots, \theta_k\}$ and $Mk = \{m_{k1}, m_{k2}, \ldots, m_{kk}\}$ be soup ingredients or the parameters for the considered Mk models. Each time i the validation is computed, the model m_i is considered if $valAcc(m_i) > min(m_{k1}, m_{k2}, \ldots, m_{kk})$. The Mk models are sorted decreasingly.
  From the Mk models and their sg parameters, the model is considered for fusion if, for each time i to k, the $valAcc(sg_{i-1} \cup \{\theta_i\})$ is greater than $valAcc(sg_{i-1})$.et $c - sg$ denote the considered j models. The final model parameters θ' are the average of the $c - sg$ models' parameters, and they are calculated using the equation:

$$\theta' = \frac{1}{j} \sum_{i=1}^{j} \theta_i \quad (19)$$

  The proposed model is set to be based on the best three models, $k = 3$, where the validation is calculated in the middle and end of each epoch. Algorithm 1 shows the algorithm of greedy soup to fuse the three models with different hyper-parameters (number of steps). Figure 6 presents the overall structure of the greedy soup technique to fuse three models.
- **The best value**
  The fine-tuning based on the best accuracy value selects the model that achieves the highest score. It is represented using the equation:

$$f(x, argmax_i ValAcc(\theta_i)) \quad (20)$$

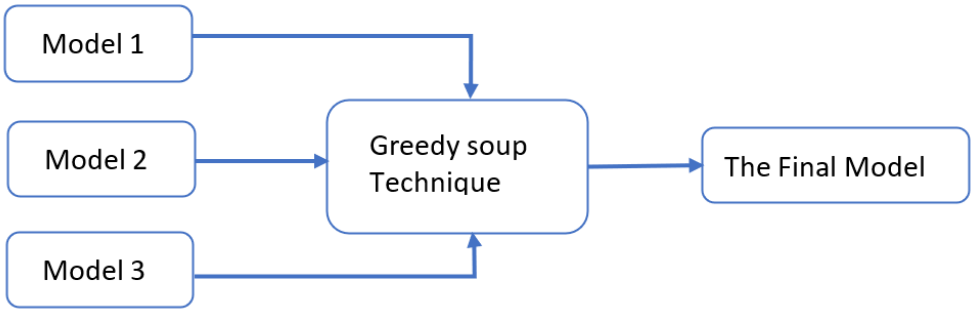


Figure 6. The Greedy Soup Model.

Algorithm 1 Greedy_Soup_Model

Require: number of epoch *epoch* and greedy soup grade k
Ensure: model parameters θ

$valAcc \leftarrow \{\}$
$ingredients \leftarrow \{\}$
for $i \leftarrow 1$ to *epoch* **do**
  if not mid(*epoch*) **then**
    train model
  end if
  Compute the $val - accuracy(model)$
  train model
  if length$(ingredients) < 3$ **then**
    $ingredients \leftarrow ingredients \cup \{model(\theta)\}$
    $valAcc \leftarrow valAcc \cup val - accuracy(model)$
  else if $val - accuracy(model) > min(valAcc)$ **then**
    remove$(valAcc, min(valAcc))$
    remove$(ingredients, min(model(\theta)))$
    $ingredients \leftarrow ingredients \cup \{model(\theta)\}$
  end if
end for
decreasing order of $ValAcc$ with its appropriate *ingredients*
$ingredients\_soup \leftarrow \{\}$
for $i \leftarrow 1$ to k **do**
  if $ValAcc(\text{average}(ingredients_soup \cup \{ingredients_i\})) \geq ValAcc(\text{average}(ingredients\_soup))$ **then**
    $ingredients\_soup \leftarrow ingredients\_soup \cup \{ingredients_i\}$
  end if
end for
return average(*ingredients_soup*)

4. Experiment

4.1. Environment Setup

The models are trained on a Google Colab premium account with NVIDIA A100-SXM4-40 GB (Nvidia Corporation, Santa Clara, CA, USA) and 80 GB RAM. For the optimization function, AdamW is used with 1.0×10^{-3} and 0.9 weight decay andearning rate decay, respectively. The model optimized to select the best batch size andearning rate among the values 1.0×10^{-3}, 3.0×10^{-3}, 2.0×10^{-3}, 1.0×10^{-4}, 9.0×10^{-5}, 8.0×10^{-5}, 1.0×10^{-5} and 16, 32, 64, 128, respectively. The model was trained for ten epochs in all experiments and evaluated using the validation set twice per epoch. Furthermore, it sets up to stop training if there is no enhancement for ten sequential times.

4.2. Dataset

The Diabetic Macular Edema (DME) dataset [5] is generated automatically from the Indian Diabetic Retinopathy Image Dataset (IDRiD) [81], and the e-Ophta dataset [82] is used. The dataset has 679 images with 13,470 question–answer pairs distributed into 433, 112, 134 images with 9779, 2380, and 1311 for the train, validation, and testing dataset, respectively. The dataset has questions about hard exudates, optic discs, and exudates' grades. The distinction between grades is based on the presence of a hard exudate at various sites on the retina. Specifically, grade 0 means no hard exudate at all, grade 1 means there is a hard exudate in the periphery of the retina (i.e., central fovea and radius one papilla diameter), and grade 2 if there is a hard macular exudate. The dataset has the original images and masks that specify a specific region of the image. The masks have to be applied to the images before training. Figure 7 shows examples of images and related question–answer pairs from the dataset. Table 1 shows the distributed of classes in the train, validation, and testing datasets.

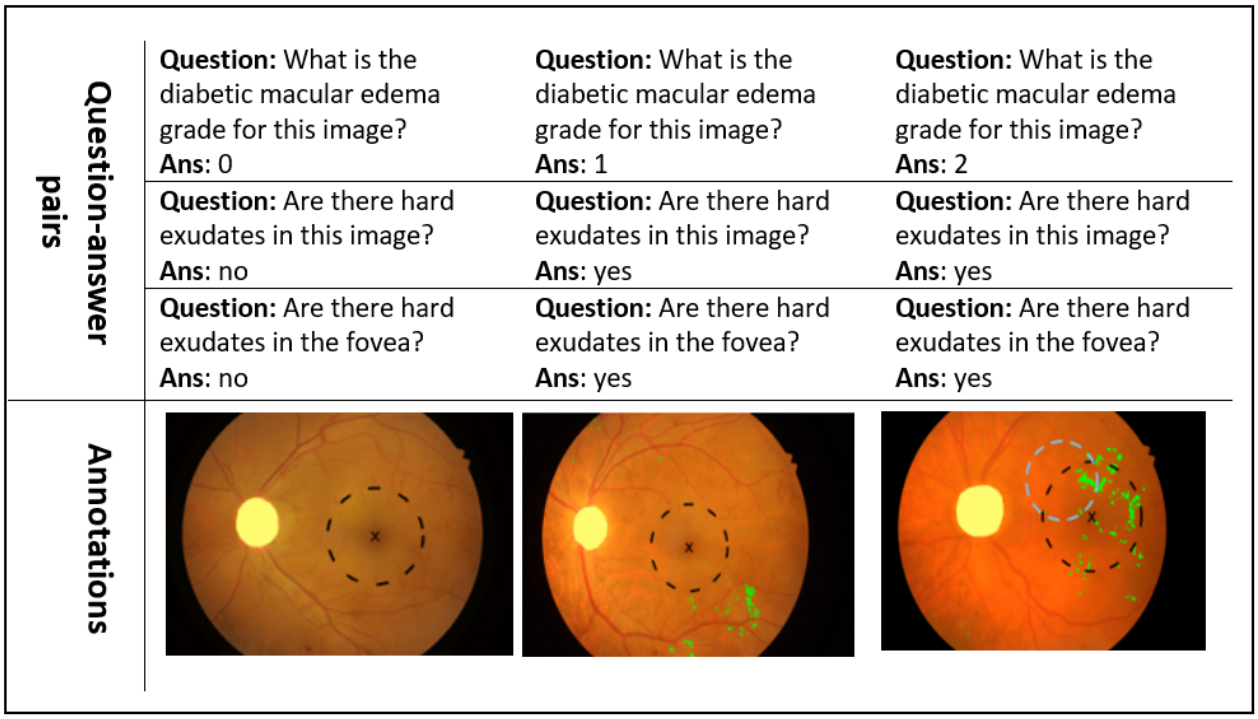


Figure 7. Examples of DME dataset images and related QA pairs, where the yellow circle is an optic disk, the cutine circle is the macula, and x is the fovea- the center of the macula.

Table 1. Number of instances per answer for each part of DME dataset.

	Yes	No	0	1	2	Total
Train	4713	4639	166	41	220	9779
Validation	1151	1123	39	8	59	2380
Test	530	650	49	15	67	1311
Total	6394	6412	254	64	346	13,470

4.3. Evaluation Metrics

We calculate the model accuracy, precision, recall, F1-score, macro accuracy average, and weighted accuracy average to measure the model performance and compare it with the SOTA models. The equation of each metric is shown in the following:

- **Accuracy:** Accuracy is a commonly used performance metric for classification problems that measures the proportion of correctly classified samples out of the total number of samples. It is calculated using the equation:

$$\text{Accuracy} = \frac{\text{True Positives + True Negatives}}{\text{True Positives + False Positives + True Negatives + False Negatives}} \quad (21)$$

where True Positives are the number of positive samples that are correctly identified as positive by the model, True Negatives are the number of negative samples that are correctly identified as negative by the model, False Positives are the number of negative samples that are incorrectly identified as positive by the model, and False Negatives are the number of positive samples that are incorrectly identified as negative by the model.

- **Precision:** Precision, also known as a positive predictive value, measures the proportion of the predicted true positive samples out of all predicted positive samples. The precision metric's importance is giving an indication of whether the model has aow false positive rate based on a high precision score. It is calculated using the equation:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \quad (22)$$

- **Recall or Sensitivity:** Recall, also known as sensitivity or true positive rate, measures the proportion of the predicted positive samples out of the actual positive samples. The recall metric is critical in applications such as medical diagnosis, where it is essential to minimize false negatives. It is calculated using the equation:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \quad (23)$$

- **F1-score:** The F1 -score is the harmonic mean of precision and recall that measures the model's accuracy in correctly identifying positive samples. The F1-score is a valuable metric for evaluating classification models, especially when the dataset is imbalanced, and the goal is to ensure that the model performs well on both positive and negative samples. It is calculated using the equation:

$$F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}} \quad (24)$$

- **Macro-averaged precision:** The macro-averaged precision is calculated by taking the average of the precision scores for each class. The equation for the macro-averaged precision can be expressed as:

$$\text{Macro Avg Precision} = \frac{1}{n} \sum_{i=1}^{n} \frac{TP_i}{TP_i + FP_i} \quad (25)$$

where n, TP_i, and FP_i are the total number of classes, the number of true positives for class i, and the number of false positives for class i, respectively.

- **Macro-averaged recall:** The macro-averaged recall is calculated by taking the average of the recall scores for each class. InaTeX, the equation for the macro-averaged recall can be expressed as:

$$\text{Macro Avg Recall} = \frac{1}{n} \sum_{i=1}^{n} \frac{TP_i}{TP_i + FN_i} \quad (26)$$

where n, TP_i, and FN_i are the total number of classes, true positives for class i, and false negatives for class i, respectively.

- **Macro-averaged F1-score:** The macro-averaged F1-score is calculated by taking the average of the F1-scores for each class. The equation for the macro-averaged F1-score can be expressed as :

$$\text{Macro Avg F1-score} = \frac{1}{n}\sum_{i=1}^{n}\frac{2 \times \text{precision}_i \times \text{recall}_i}{\text{precision}_i + \text{recall}_i} \tag{27}$$

  where n is the total number of classes, and precision_i and recall_i are the precision and recall values for class i, respectively.
- **Weighted-average precision:** The weighted-average precision is calculated by taking the weighted average of the precision scores for each class. The equation for the weighted-average precision can be expressed as :

$$\text{Weighted Avg Precision} = \frac{\sum_{i=1}^{n} w_i \times \frac{TP_i}{TP_i+FP_i}}{\sum_{i=1}^{n} w_i} \tag{28}$$

  where n, TP_i, FP_i, and w_i are the total number of classes, the number of true positives for class i, the number of false positives for class i, and the weight for class i, respectively.
- **Weighted average recall:** The weighted average recall is calculated by taking the weighted average of the recall scores for each class. The equation for the weighted-average precision can be expressed as :

$$\text{Weighted Avg Recall} = \frac{\sum_{i=1}^{n} w_i \times \frac{TP_i}{TP_i+FN_i}}{\sum_{i=1}^{n} w_i} \tag{29}$$

  where n,TP_i, FN_i, and w_i are the total number of classes, the number of true positives for class i, the number of false negatives for class i, and the weight for class i, respectively.
- **Weighted-average F1-score:** The weighted-average F1-score is calculated by taking the weighted average of the F1-scores for each class. InaTeX, the equation for the weighted-average F1-score can be expressed as :

$$\text{Weighted Avg F1-score} = \frac{\sum_{i=1}^{n} w_i \times \frac{2\times\text{precision}_i\times\text{recall}_i}{\text{precision}_i+\text{recall}i}}{\sum i = 1^n w_i} \tag{30}$$

  where n, precision_i, recall_i, and w_i are the total number of classes, the precision values for class i, recall values for class i, and the weight for class i, respectively.

4.4. Results and Analysis

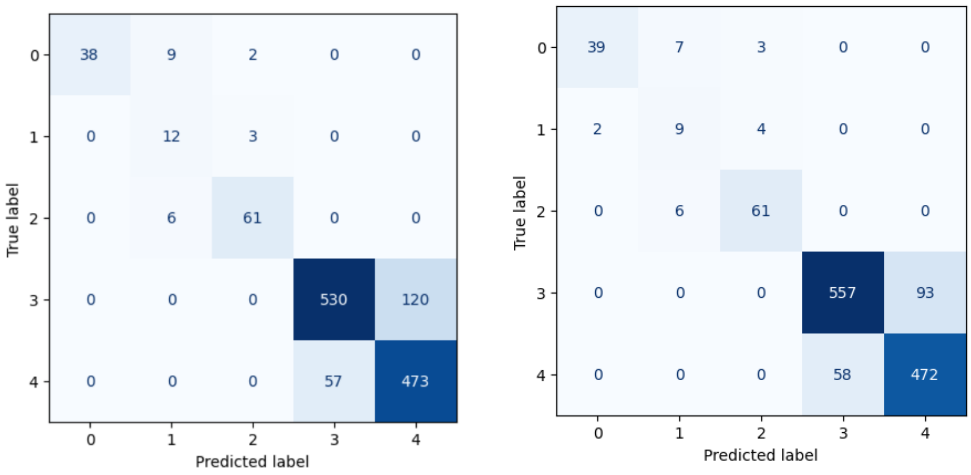
In this paper, we proposed two models that are based on ELCTRA-base, SWIN-base, and MLP for textual, visual, and fusion VQA phases. The first model parameters were selected based on the best validation model performance in the training. The second model is a fusion of three of the first model with different hyper-parameters based on the greedy soup method. The proposed models are compared to the SOTA models proposed by Tascon et al [5]. Tascon et al. [5] designed a VQA model that utilized ResNet101, LSTM, and multi-glimpse attention for visual, textual, and fusion, respectively. They designed a consistency loss function that penalizes the model if it answers the related questions with inconsistent answers. They trained the model without attention, with attention, with attention with SQuINT [83] configuration, and with attention and consistency loss. They trained the models with the Adam optimization function with a learning rate of 1.0×10^{-4} and 64 batch size for 100 epochs with an early stop if there is no validation accuracy enhancement for 20 epochs.

The proposed model is configured with a learning rate value of 0.0001, the default value for the AdamW optimization function, and a batch size value of 32, which is usually used in the field. In both techniques, the model outperforms the SOTA performance, which has an accuracy of 83.49% , where it achieves an accuracy of 84.97% and 86.8% for the two models, respectively. Table 2 presents the precision, recall, F1-score, macro average, and weighted average accuracy for each class in the testing dataset for both models. In addition,

they present the overall accuracy. The result appears that the model with the greedy soup fine-tuning technique outperforms the traditional one. According to the confusion matrix for both models in Figure 8, it can be clearly seen that the greedy soup model outperforms the best-value-based model in predicting '0' and 'no' answers, whereas the best-value-based model outperforms it in predicting '1' and 'yes' answers. For the answer of grade equal to '0', although the greedy-soup-based model predicts correct answers more than the best-value-based model, the latter achieves 100% precision, whereas the greedy-soup-based model achieves 95.12% precision. This result shows that the best-value-based model can predict that other answers are not '1', but at the same time, it can not always predict the answer '1' as '1', and that is why the recall of the answer '1' in the greedy-soup-based model is higher than that in the best-value-based model.

Table 2. The result of models using the greedy soup fine-tuning—batch size = 32, lr = 1.0×10^{-4}.

Answer	Precision	Recall	F1-Score	Instances No.
0	0.9512	0.7959	0.8667	49
1	0.4091	0.6000	0.4865	15
2	0.8971	0.9104	0.9037	67
no	0.9057	0.8569	0.8806	650
yes	0.8354	0.8906	0.8621	530
accuracy			0.8680	1311
macro avg	0.7997	0.8108	0.7999	1311
weighted avg	0.8729	0.8680	0.8693	1311



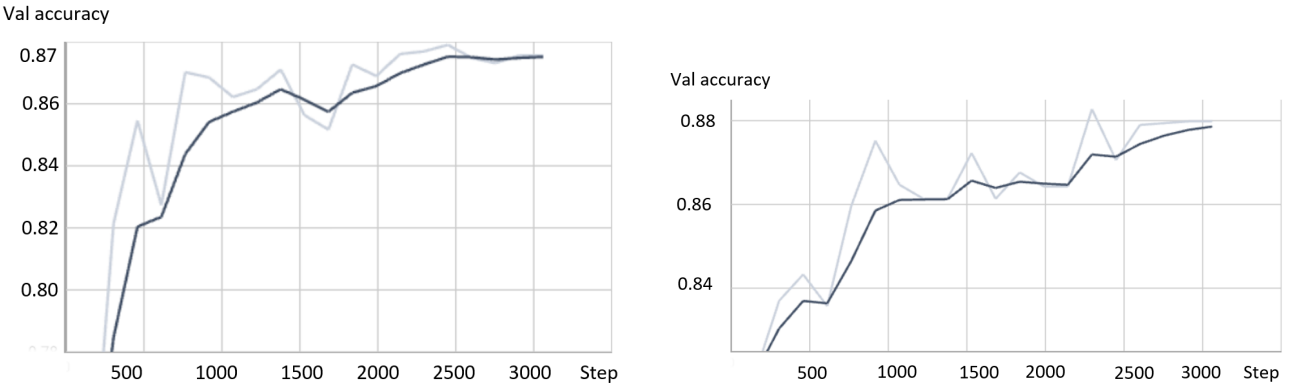
(**a**) The best validation accuracy (**b**) The greedy soup

Figure 8. The model confusion matrix for models with different fine-tuning method.

Overall, since the dataset is unbalanced, the macro average and weighted average performance are more precise than those metrics, which compute the performance ignoring the variance between the answers numbers in the dataset. Therefore, comparing the two methods based on the macro average and weighted average performance is considered. The average macro precision, recall, and F1-score are 81.38 vs. 79.97, 83.88 vs. 81.08, and 8123 vs. 79.99 for the best-value-based and greedy-soup-based models, respectively. On the other hand, the average weighted precision, recall, and F1-score are 85.98 vs. 87.29, 84.97 vs. 86.80, and 85.15 vs. 86.93 for the best-value and greedy-soup-based models, respectively. Figure 9 presents the validation performance for both model fine-tuning techniques. The best-value-based model achieves higher average macro performance than the greedy-soup-based models, whereas the greedy-soup-based model achieves higher weighted average performance and overall accuracy than the first model. Therefore, the greedy-soup-based model is more significant than the best-value-based model for two reasons. The first reason is that the weighted average metric considers the unbalanced

dataset and gives each class weight based on how much that class appears. The second reason is that accuracy is the metric computed on the SOTA med-VQA.

The model is optimized based on learning rate and batch size hyper-parameters. Since no rule indicates the best values assigned for the hyper-parameters, we select the best learning rate from the learning rate set 1.0×10^{-5}, 8.0×10^{-5}, 9.0×10^{-5}, 1.0×10^{-4}, 2.0×10^{-4}, 3.0×10^{-4}, and 1.0×10^{-3}. This set is selected based on the default value of AdamW and experiments. The AdamW default value is 1.0×10^{-4}. The selecting technique we follow starts from the default learning rate of AdamW then decreases the value by 0.00001 and trains the model. We repeat decreasing the value until there is no model performance enhancement for two sequential values compared to the model performance in the default value learning rate. We follow the same method by increasing the learning rate value by 0.0001 until there is no significant model performance for two sequential values. This technique produces 8.0×10^{-5}, 9.0×10^{-5}, 1.0×10^{-4}, 2.0×10^{-4}, and 3.0×10^{-4}. Then the edge values are considered as well, which are 1.0×10^{-5} and 1.0×10^{-3} . The model is trained for all learning rates with a batch size of 32. The selection of the best learning rate is based on overall accuracy. The model accuracies for the seven experiments with learning rates of 1.0×10^{-5}, 8.0×10^{-5}, 9.0×10^{-5}, 1.0×10^{-4}, 2.0×10^{-4}, 3.0×10^{-4}, and 1.0×10^{-3} are 85.74%, 86.04%, 85.81%, 86.8%, 84.97%, 66.67%, and 64.61%, respectively. The learning rate of value 1.0×10^{-4} is selected because it achieved the highest performance. The results of the other metrics for all experiments are presented in Tables 3–9. The confusion matrices present the number of each answer predicted correctly or mistakenly, which are shown in Figure 10. The accuracy chart for validation is shown in Figure 11, whereas the validation loss and Train loss is in Figures A1 and A2.



(**a**) The best validation accuracy (**b**) The greedy soup

Figure 9. The model validation accuracy for models with different fine-tuning methods. The lighter line in the graph represents the validation accuracy, whereas the bold line is the smoothed validation accuracy.

Table 3. The result of models using the greedy soup fine-tuning—batch size = 32, lr = 9.0×10^{-5}.

Answer	Precision	Recall	F1-Score	Instances No.
0	0.8723	0.8367	0.8542	49
1	0.4286	0.4000	0.4138	15
2	0.8714	0.9104	0.8905	67
no	0.9012	0.8415	0.8703	650
yes	0.8202	0.8868	0.8522	530
accuracy			0.8581	1311
macro avg	0.7787	0.7751	0.7762	1311
weighted avg	0.8604	0.8581	0.8582	1311

Table 4. The result of models using the greedy soup fine-tuning—batch size = 32, lr = 8.0×10^{-5}.

Answer	Precision	Recall	F1-Score	Instances No.
0	0.9048	0.7755	0.8352	49
1	0.3500	0.4667	0.4000	15
2	0.8841	0.9104	0.8971	67
no	0.8942	0.8585	0.8760	650
yes	0.8345	0.8755	0.8545	530
accuracy			0.8604	1311
macro avg	0.7735	0.7773	0.7725	1311
weighted avg	0.8637	0.8604	0.8614	1311

Table 5. The result of models using the greedy soup fine-tuning—batch size = 32, lr = 2.0×10^{-4}.

Answer	Precision	Recall	F1-Score	Instances No.
0	0.8936	0.8571	0.8750	49
1	0.4615	0.4000	0.4286	15
2	0.8732	0.9254	0.8986	67
no	0.8738	0.8523	0.8629	650
yes	0.8242	0.8491	0.8364	530
accuracy			0.8497	1311
macro avg	0.7853	0.7768	0.7803	1311
weighted avg	0.8497	0.8497	0.8495	1311

Table 6. The result of models using the greedy soup fine-tuning—batch size = 32, lr = 3.0×10^{-4}.

Answer	Precision	Recall	F1-Score	Instances No.
0	0.6622	1.0000	0.7967	49
1	0.0000	0.0000	0.0000	15
2	0.9636	0.7910	0.8689	67
no	0.7727	0.5231	0.6239	650
yes	0.5822	0.8151	0.6792	530
accuracy			0.6667	1311
macro avg	0.5961	0.6258	0.5937	1311
weighted avg	0.6925	0.6667	0.6581	1311

Table 7. The result of the best-value model with batch size = 32 and lr = 1.0×10^{-3}.

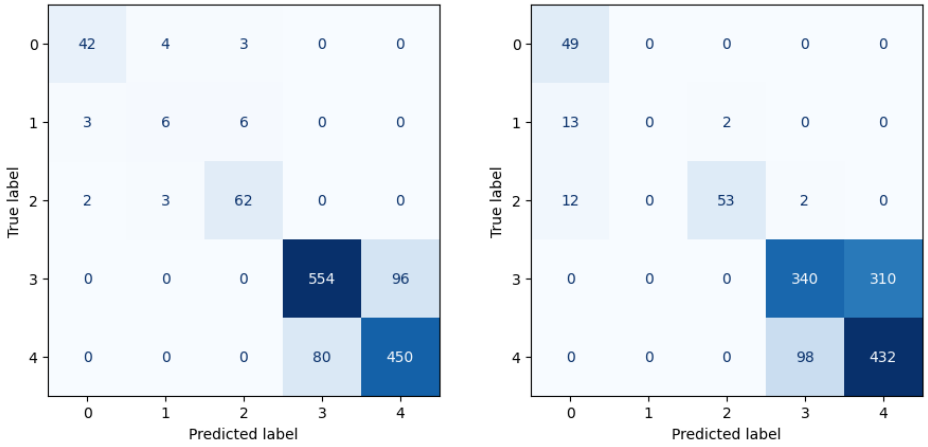
Answer	Precision	Recall	F1-Score	Instances No.
0	0.7119	0.8571	0.7778	49
1	0.0000	0.0000	0.0000	15
2	0.8194	0.8806	0.8489	67
no	0.7634	0.4815	0.5906	650
yes	0.5623	0.8170	0.6662	530
accuracy			0.6461	1311
macro avg	0.5714	0.6073	0.5767	1311
weighted avg	0.6743	0.6461	0.6346	1311

Table 8. The result of the best-value model with batch size = 32 and lr = 1.0×10^{-5}.

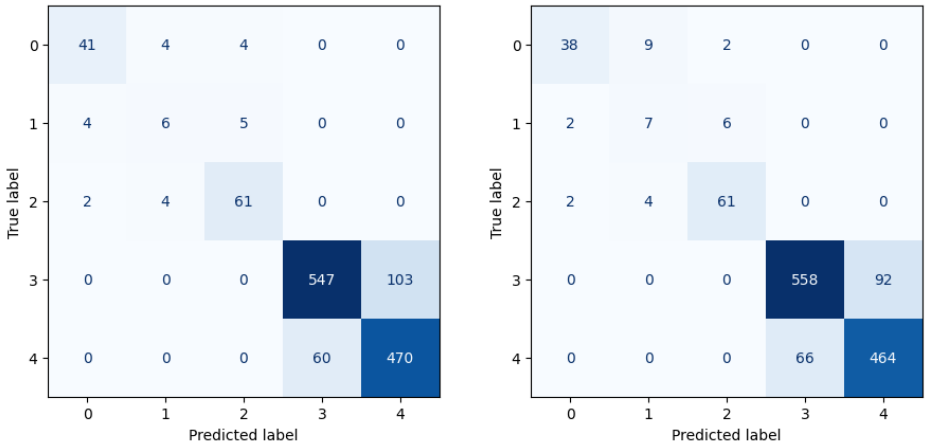
Answer	Precision	Recall	F1-Score	Instances No.
0	0.8000	0.8980	0.8462	49
1	0.8000	0.2667	0.4000	15
2	0.8592	0.9104	0.8841	67
no	0.8819	0.8615	0.8716	650
yes	0.8349	0.8585	0.8465	530

Table 8. *Cont.*

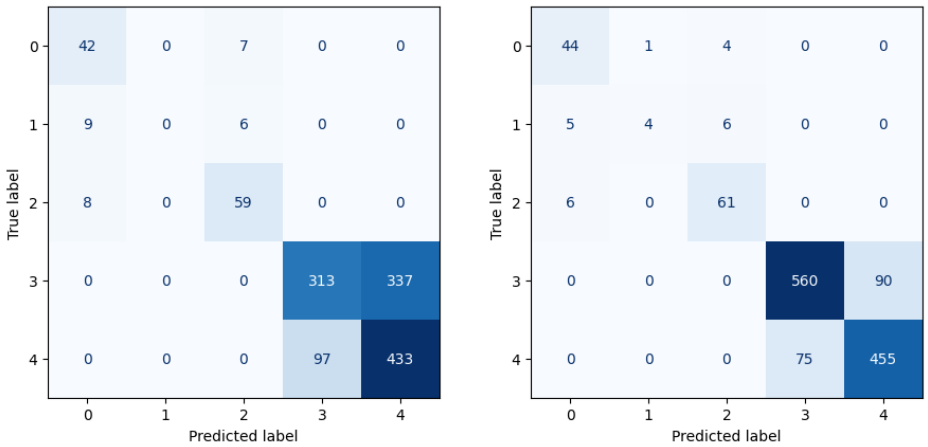
Answer	Precision	Recall	F1-Score	Instances No.
accuracy			0.8574	1311
macro avg	0.8352	0.7590	0.7697	1311
weighted avg	0.8577	0.8574	0.8557	1311



(**a**) learning rate = 2.0×10^{-4} (**b**) learning rate = 3.0×10^{-4}



(**c**) learning rate = 9.0×10^{-5} (**d**) learning rate = 8.0×10^{-5}

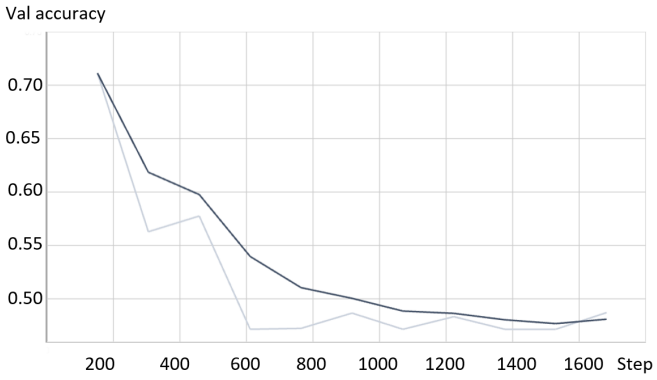


(**e**) learning rate = 1.0×10^{-3} (**f**) learning rate = 1.0×10^{-5}

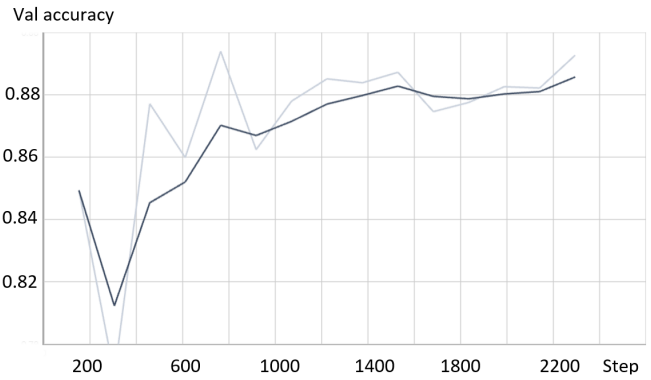
Figure 10. The confusion matrices of models with different learning rates and batch size = 32.

Table 9. The result of the best-value model fine-tuned based on the best validation accuracy—batch size = 32, lr = 1.0×10^{-4} weight.

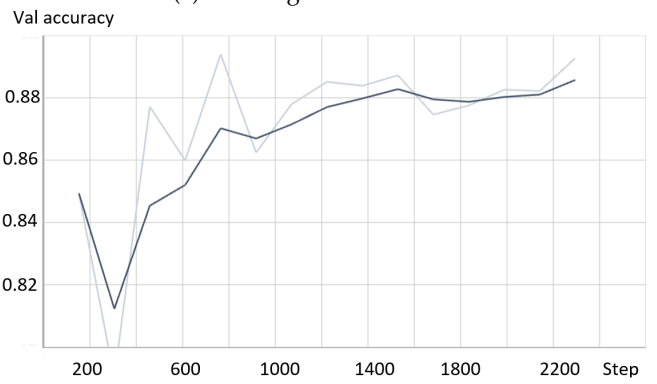
Answer	Beat-Value-Based Model			Greedy-Soup-Based Model			Samples No.
	Precision	Recall	F1-Score	Precision	Recall	F1-Score	
0	1.000	0.7755	0.8736	0.9512	0.7959	0.8667	49
1	0.4444	0.8000	0.5714	0.4091	0.6000	0.4865	15
2	0.9242	0.9104	0.9173	0.8971	0.9104	0.9037	67
no	0.9029	0.8154	0.8569	0.9057	0.8569	0.8806	650
yes	0.7976	0.8925	0.8424	0.8354	0.8906	0.8621	530
Accuracy			0.8497			0.8680	1311
macro avg	0.8138	0.8388	0.8123	0.7997	0.8108	0.7999	1311
weighted avg	0.8598	0.8497	0.8515	0.8729	0.8680	0.8693	1311



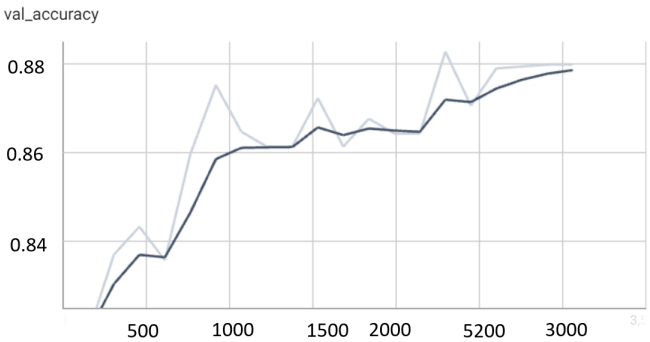
(**a**) learning rate = 1.0×10^{-3}



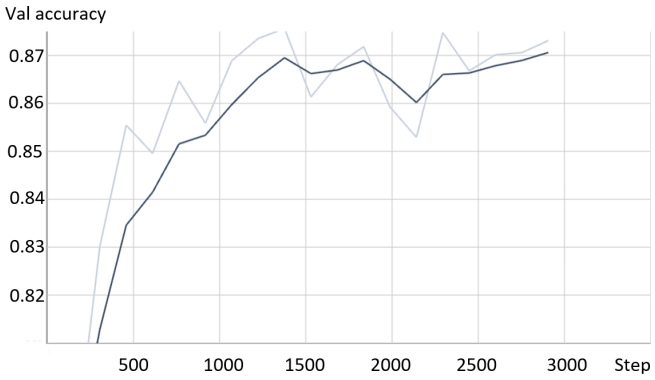
(**b**) learning rate = 3.0×10^{-4}



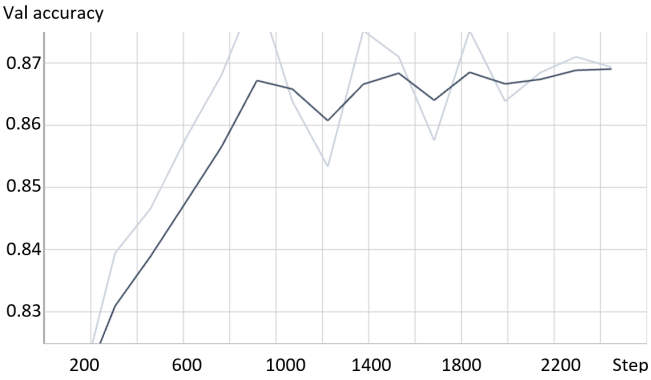
(**c**) learning rate = 2.0×10^{-4}



(**d**) learning rate = 1.0×10^{-4}

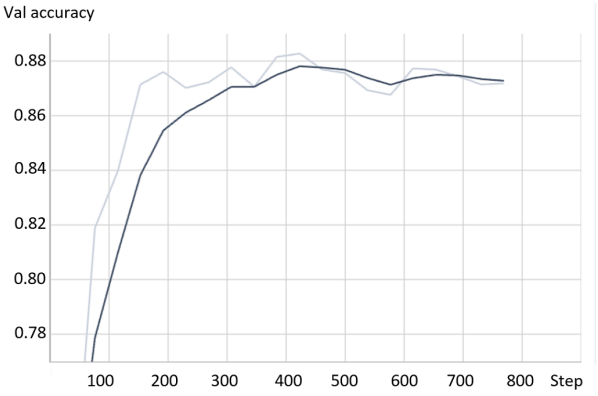


(**e**) learning rate = 9.0×10^{-2}



(**f**) learning rate = 8.0×10^{-2}

Figure 11. *Cont.*



(**g**) learning rate = 1.0×10^{-5}

Figure 11. The model optimization using different learning rates and batch size = 32. The lighter line in the graph represents the validation accuracy, whereas the bold line is the smoothed validation accuracy.

The model is optimized for batch size based on the learning rate (1.0×10^{-4}), allowing the model to achieve the highest accuracy in the previous experiments. The model is trained with various batch sizes, 16, 32, 64, 128. The selection of the best batch size is based on overall accuracy. The model accuracies for the four experiments with a batch size of 16, 32, 64, and 128 are 87.41%, 86.8%, 83.45%, and 84.13%, respectively. The batch size of value 16 is selected. The result of other metrics for all experiments is presented in Tables 3 and 10–12. The confusion matrices present the number of each answer predicted correctly or mistakenly, which are shown in Figure 12. The accuracy chart for validation is shown in Figure 13, whereas the validation loss and Train loss is in Figures A3 and A4.

Table 10. The result of the the model with batch size = 16 and lr = 1.0×10^{-4}.

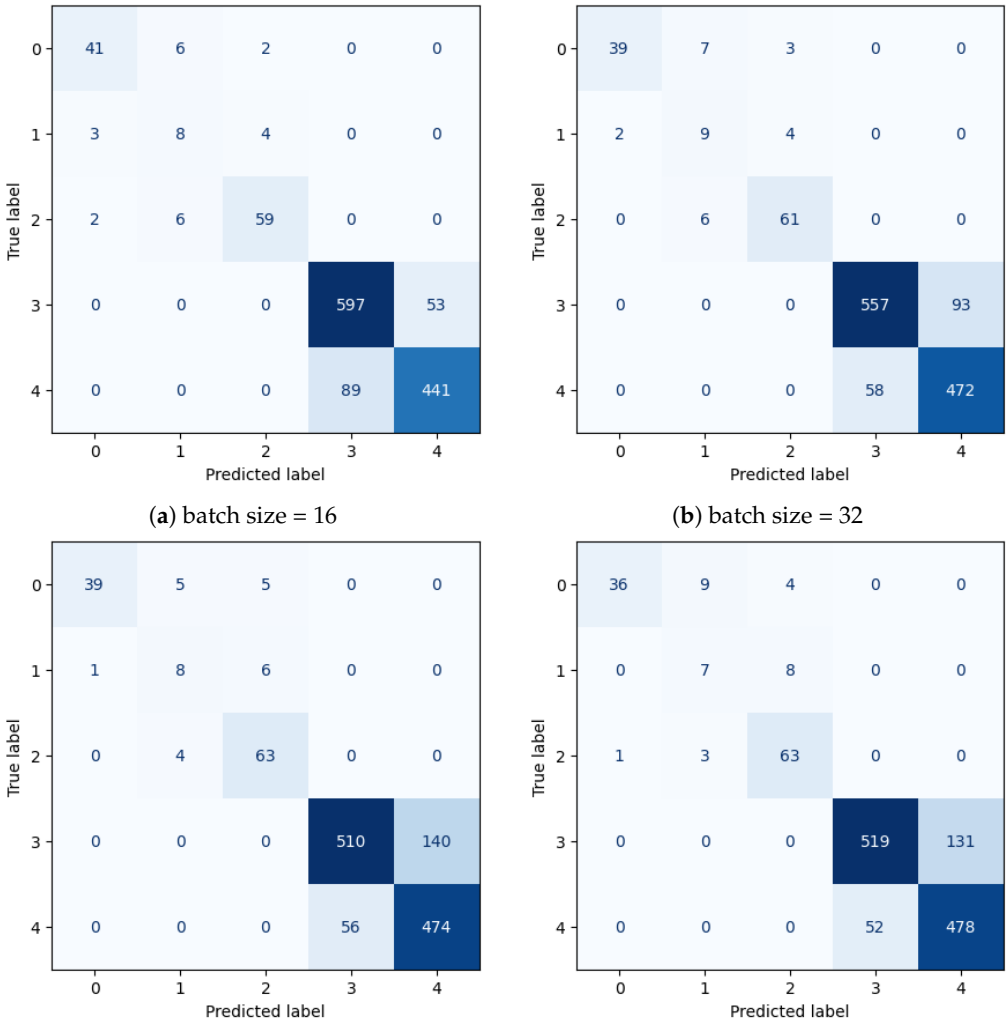
Answer	Precision	Recall	F1-Score	Samples No.
0	0.8913	0.8367	0.8632	49
1	0.4000	0.5333	0.4571	15
2	0.9077	0.8806	0.8939	67
no	0.8703	0.9185	0.8937	650
yes	0.8927	0.8321	0.8613	530
accuracy			0.8741	1311
macro avg	0.7924	0.8002	0.7939	1311
weighted avg	0.8767	0.8741	0.8745	1311

Table 11. The result of the the model with batch size = 64 and lr = 1.0×10^{-4}.

Answer	Precision	Recall	F1-Score	Samples No.
0	0.9750	0.7959	0.8764	49
1	0.4706	0.5333	0.5000	15
2	0.8514	0.9403	0.8936	67
no	0.9011	0.7846	0.8388	650
yes	0.7720	0.8943	0.8287	530
accuracy			0.8345	1311
macro avg	0.7940	0.7897	0.7875	1311
weighted avg	0.8442	0.8345	0.8350	1311

Table 12. The result of the the model with batch size = 128 and lr = 1.0×10^{-4}.

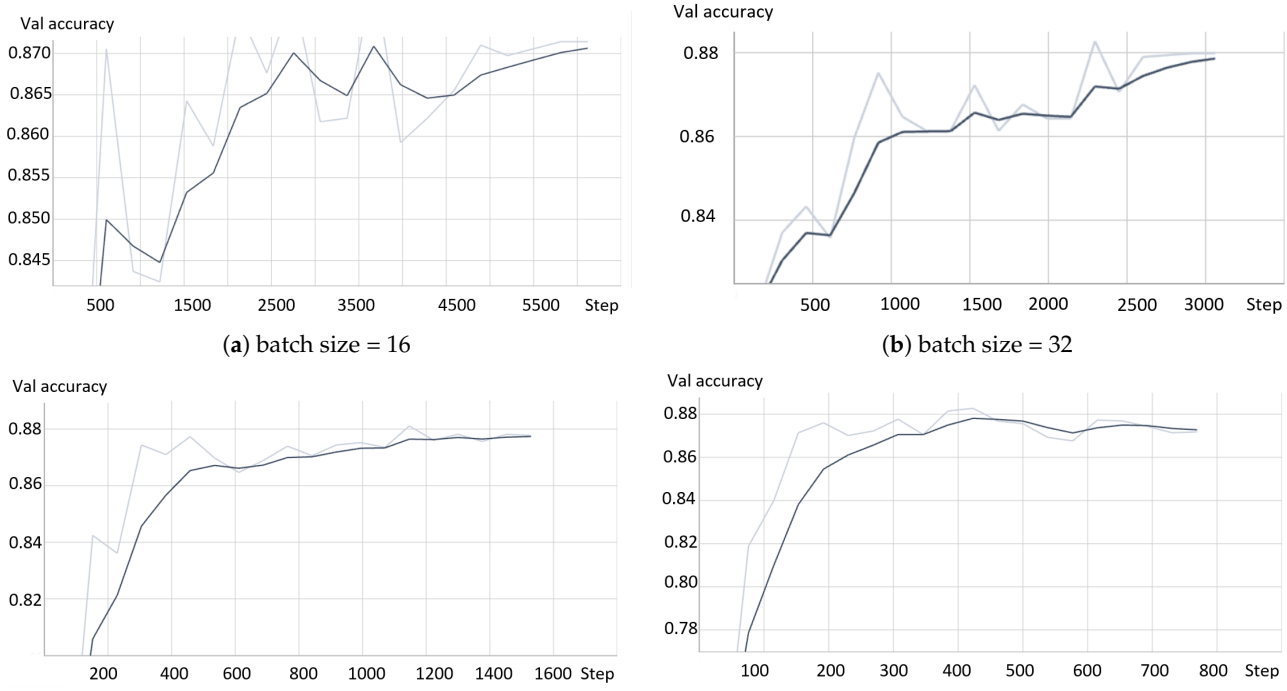
Answer	Precision	Recall	F1-Score	Samples No.
0	0.9730	0.7347	0.8372	49
1	0.3684	0.4667	0.4118	15
2	0.8400	0.9403	0.8873	67
no	0.9089	0.7985	0.8501	650
yes	0.7849	0.9019	0.8393	530
accuracy			0.8413	1311
macro avg	0.7750	0.7684	0.7652	1311
weighted avg	0.8515	0.8413	0.8422	1311



(**a**) batch size = 16 (**b**) batch size = 32

(**c**) batch size = 64 (**d**) batch size = 128

Figure 12. The confusion matrices of models with different batch sizes and learning rate = 1.0×10^{-4}.



(**a**) batch size = 16

(**b**) batch size = 32

(**c**) Validation accuracy of the model with batch size = 64

(**d**) Validation accuracy of the model with batch size = 128

Figure 13. The model optimization using different batch sizes and learning rate = 1.0×10^{-4}. The lighter line in the graph represents the validation accuracy, whereas the bold line is the smoothed validation accuracy.

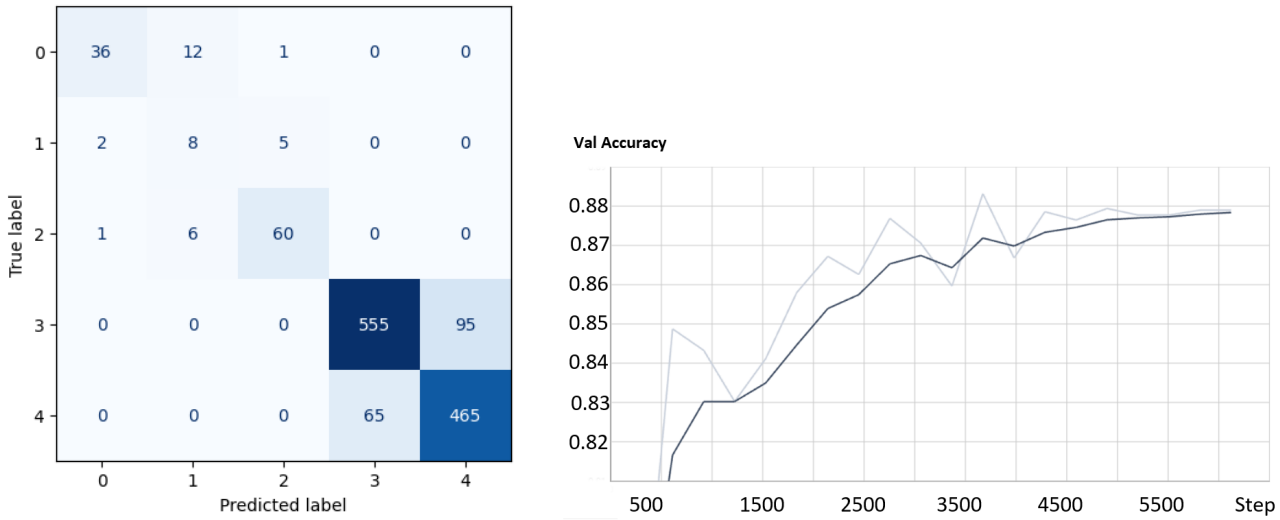
An experiment of the best-value model with the same hyper-parameters (16 batch size and 1.0×10^{-4} learning rate) of the best model based on the fusion of three models with the same seed (42) is conducted to show the effectiveness of the model based on the weights fusion of multiple fine-tuned models. Table 13 shows the performance of the best-value model with batch size 16 and learning rate 1.0×10^{-4} . In addition, Figure 14 shows the confusion matrix and validation accuracy. The metrics used for the comparison are accuracy, which is the metric usually used in medical VQA, and weighted average, precision, recall, and F1-score, as a result of the unbalanced dataset with a massive variance of the example number in each class. Table 14 shows the results of four models, two best-value models and two fused models. Figure 15 presents the comparison chart showing how the greedy soup model can produce more significant results than the model based on the best validation performance with the same computation cost $O(1)$ for the investigated problem. The greedy soup model with batch size 32 and learning rate 1.0×10^{-4} enhances the performance of the best-value model with the same batch size and learning rate values by 1.83%, 1.31%, 1.83%, and 1.78% for accuracy, weighted average precision, weighted average recall, and weighted average F1-recall, respectively. The models trained with batch size 16 and learning rate 1.0×10^{-4} have enhanced accuracy, weighted average precision, weighted average recall, and weighted average F1-recall. The greedy soup model with batch size 16 outperforms the best-value model with the same 16 batch size by 1.67%, 1.27%, 1.67%, and 1.51% for accuracy, weighted average precision, weighted average recall, and weighted average F1-recall, respectively.

Table 13. The result of the the best-value model with batch size = 16 and lr = 1.0×10^{-4}.

Answer	Precision	Recall	F1-Score	Samples No.
0	0.9231	0.7347	0.8182	49
1	0.3077	0.5333	0.3902	15
2	0.9091	0.8955	0.9023	67
no	0.8952	0.8538	0.8740	650
yes	0.8304	0.8774	0.8532	530
accuracy			0.8574	1311
macro avg	0.7731	0.7790	0.7676	1311
weighted avg	0.8640	0.8574	0.8594	1311

Table 14. A comparison between the best-value model and the greedy soup model in different batch size values.

Model	Accuracy	Weights Avg Precision	Weights Avg Recall	Weights Avg F1-Score
bv-32-1.0×10^{-4}	84.97	85.98	84.97	85.15
bv-16-1.0×10^{-4}	85.74	86.40	85.74	85.94
gs-32-1.0×10^{-4}	86.80	87.29	86.80	86.93
gs-16-1.0×10^{-4}	87.41	87.67	87.41	87.45



(**a**) Confusion Matrix (**b**) Validation Accuracy

Figure 14. The best-value model with batch size = 16 and learning rate = 1.0×10^{-4}.

According to all experiments, seven models exceed the SOTA accuracy, which is 83.49%. Those models are the best-value-based model with batch size 32 and learning rate of 1.0×10^{-4} (84.97%), the best-value-based model with batch size 16 and learning rate of 1.0×10^{-4} (85.74%), the greedy-soup-based model with batch size 16 and learning rate of 1.0×10^{-4} (87.41%), the greedy-soup-based model with batch size 32 and learning rate of 9.0×10^{-5} (85.81%), the greedy-soup-based model with batch size 32 and learning rate of 8.0×10^{-5} (86.04%), the greedy-soup-based model with batch size 32 and learning rate of 2.0×10^{-4} (84.97%), the greedy-soup-based model with batch size 32 and learning rate of 1.0×10^{-4} (86.8%), the greedy-soup-based model with batch size 128 and learning rate of 1.0×10^{-4} (84.13%), and greedy-soup-based model with batch size 32 and learning rate of 1.0×10^{-5} (85.74%). The greedy-soup-based model with batch size 64 and a learning rate of 1.0×10^{-4} achieves an accuracy of 83.45%, which is not far from the SOTA accuracy. Table 15 compares the results of the 12 models based on overall accuracy, average macro precision, average macro recall, average macro F1-score, weighted average precision, weighted average recall, and weighted average F1-score. Figures 11, 13, and 14 show the

chart of the 12 models' validation accuracy. Tables 14–16 refer to the best-value-based and greedy-soup-based models with bv-batch-lr and gs-batch-lr, respectively, where batch and lr are replaced with batch size and learning rate values. For example, the model gs-128-4 is the model that is fine-tuned with the greedy soup method and training in batch size 128 and a learning rate of 1.0×10^{-4} .

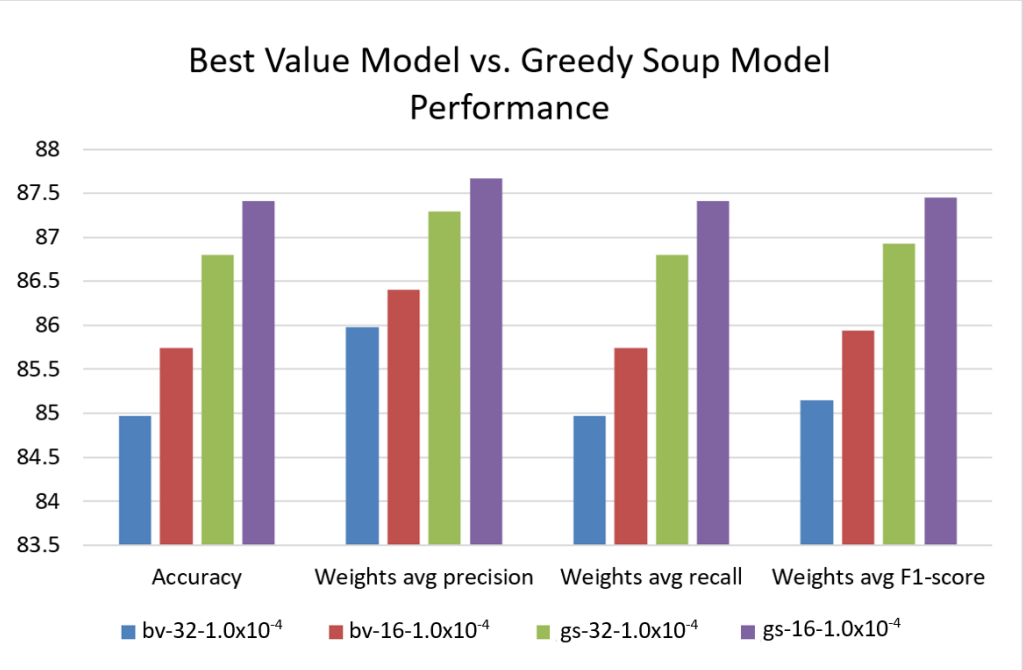


Figure 15. The comparison between best-value model and greedy soup model with different batch size (16 and 32).

Table 15. The result of the model with different batch sizes and learning rates.

Model	Accuracy	Avg. Macro Precision	Avg. Macro Recall	Avg. Macro F1-Score	Weighted Avg. Precision	Weighted Avg. Recall	Weighted Avg. F1-Score
bv-32-1.0 × 10^{-4}	0.8497	0.8138	**0.8388**	**0.8123**	0.8598	0.8497	0.8515
bv-16-1.0 × 10^{-4}	0.8574	0.7731	0.7790	0.7676	0.8640	0.8574	0.8594
gs-32-1.0 × 10^{-4}	0.868	0.7997	0.8108	0.7999	0.8729	0.8680	0.8693
gs-32-9.0 × 10^{-5}	0.8581	0.7787	0.7751	0.7762	0.8604	0.8581	0.8582
gs-32-8.0 × 10^{-5}	0.8604	0.7735	0.7773	0.7725	0.8637	0.8604	0.8614
gs-32-2.0 × 10^{-4}	0.8497	0.7853	0.7768	0.7803	0.8497	0.8497	0.8495
gs-32-3.0 × 10^{-4}	0.6667	0.5961	0.6258	0.5937	0.6925	0.6667	0.6581
gs-32-1.0 × 10^{-3}	0.6461	0.5714	0.6073	0.5767	0.6743	0.6461	0.6346
gs-32-1.0 × 10^{-5}	0.8574	**0.8352**	0.7590	0.7697	0.8577	0.8574	0.8557
gs-16-1.0 × 10^{-4}	**0.8741**	0.7924	0.8002	0.7939	**0.8767**	**0.8741**	**0.8745**
gs-64-1.0 × 10^{-4}	0.8345	0.7940	0.7897	0.7875	0.8442	0.8345	0.8350
gs-128-1.0 × 10^{-4}	0.8413	0.7750	0.7684	0.7652	0.8515	0.8413	0.8422

Table 16. The result comparison with the SOTA.

Model	Overall	Grade	Whole	Macula	Region
SOTA [5]	83.49	80.69	84.96	87.18	83.16
bv-32-1.0 $\times 10^{-4}$	84.97	**84.73**	90.84	85.29	83.22
bv-16-1.0 $\times 10^{-4}$	85.74	79.39	90.84	83.21	86.27
gs-128-1.0 $\times 10^{-4}$	84.13	80.92	90.08	88.55	83.12
gs-32-1.0 $\times 10^{-5}$	85.74	83.21	90.08	87.79	85.19
gs-32-1.0 $\times 10^{-4}$	86.8	83.21	**92.37**	**90.84**	85.95
gs-16-1.0 $\times 10^{-4}$	**87.41**	82.44	88.55	87.02	**88.02**
gs-32-2.0 $\times 10^{-4}$	84.97	83.97	87.02	89.31	84.20
gs-32-9.0 $\times 10^{-5}$	85.81	82.44	89.31	88.55	85.40
gs-32-8.0 $\times 10^{-5}$	86.04	80.92	89.31	89.32	85.84

Tascon et al. [5] evaluated the SOTA models with different question categories: overall, which is the whole dataset; grade, which refers to the questions about the diabetic macula edema grade; whole, which is a question about the existence of exudates in the whole image; macula, which refers to the fovea in the dataset; and region, which is a question about specific region in the image. The research follows the same evaluation to make a fair comparison with the SOTA model. Table 16 compares the SOTA and the nine models outperforming it based on the accuracy evaluation metric. All models based on greedy soup outperform the SOTA performance in terms of overall, whole, grade, and macula. The SOTA achieved higher region accuracy than the greedy soup model trained in 128 batch size and 1.0×10^{-4} learning rate with a slight difference of 0.04, where the SOTA model achieved 83.12%, and the gs-128-1.0×10^{-4} model achieved 83.12%. Therefore, seven fusion models based on greedy soup exceed the SOTA model. Even the bv-32-1.0×10^{-4} model achieved the highest grade accuracy; its overall accuracy is less than five fusion models based on greedy soup, which are gs-16-1.0×10^{-4}, gs-32-1.0×10^{-4}, gs-32-9.0×10^{-5}, gs-32-8.0×10^{-5}, and gs-32-1.0×10^{-5}, and it has the same accuracy as gs-32-2.0×10^{-4}. Figure 16 shows the comparison of the SOTA model and the proposed nine models in terms of overall, whole, grade, region, and macula.

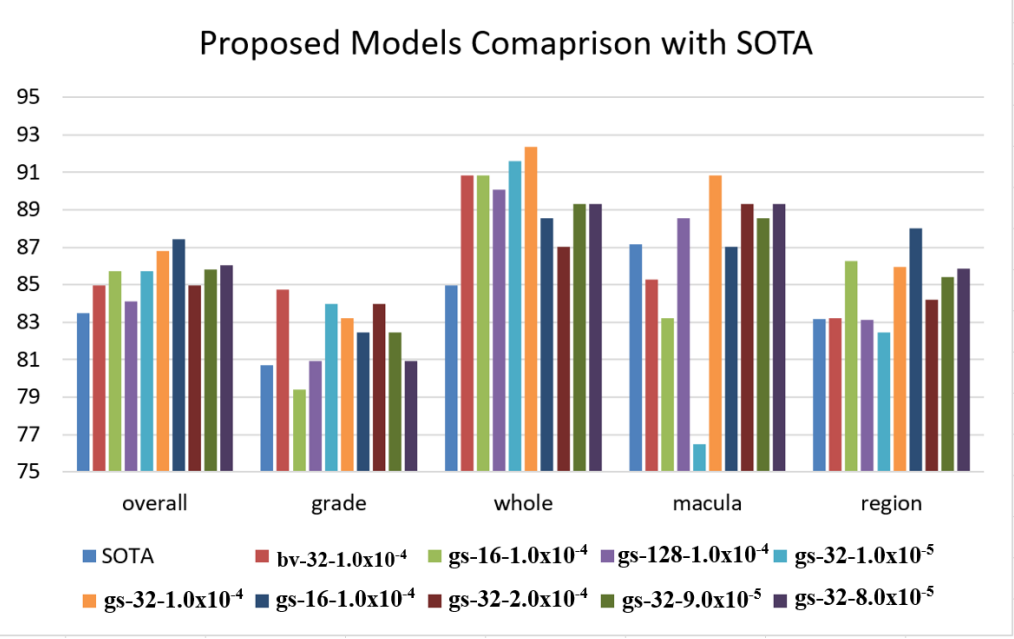


Figure 16. The comparison between the SOTA model and the proposed models.

5. Conclusions

Visual question answering (VQA) is a task that involves generating or predicting an answer to a question about visual images in human language. This task combines two branches of AI, namely, NLP and computer vision, and is an active field of research. However, in the medical field, VQA is still in its early stages and requires extensive efforts and exploration to become practically useful.

This paper presents two VQA models in the medical field. The models utilized the most current transformers for vision and language. Using a transformer instead of the visual CNN pre-trained model reduces the variation between the textual and visual features. ELECTRA-base and SWIN-base are utilized for textual and visual feature extraction. The model enhances the SOTA performance. The first model, the best-value-based model, fine-tuned the parameters based on the model that achieved the highest validation accuracy during training. The second model, the greedy-soup-based model, fine-tuned the parameters based on the greedy soup technique, where the parameters are the average of parameters that significantly impact the validation accuracy during the training. On the other hand, the model fused three best-value-based models with different training step numbers, leading to different models' weights. The best-value-based model achieved 84.97% accuracy, which outperformed the SOTA accuracy, which is 83.49%. Using greedy soup to select the model parameters enhances the model performance from 84.97% to 86.8%. Since the dataset is unbalanced, weighted average precision, recall, and F1-score were calculated. The greedy-soup-based model outperformed the best-value-based model in all calculated performances.

The greedy-soup-based model was optimized for batch size and learning rate with values 16, 32, 64, 128 and 8.0×10^{-5}, 9.0×10^{-5}, 1.0×10^{-4}, 2.0×10^{-4}, 3.0×10^{-4}, respectively. The best model performance was achieved when the batch size was 16 and the learning rate was 1.0×10^{-4}, with an accuracy of 87.41%. The other two models that exceeded the SOTA accuracy were the model with batch size 32 and learning rate 1.0×10^{-5} and the model with batch size 128 and learning rate 1.0×10^{-4}, with accuracy values of 85.74 and 84.13, respectively. Furthermore, the accuracy for question types and locations were calculated separately and compared with the SOTA.

Since there is no rule for hyper-parameters values, an optimization technique, such as a genetic algorithm, can significantly improve the model performance by selecting the optimal ones. Collecting large datasets in the medical field for VQA is complex and requires a long time because specialists are needed to validate the question-answer pairs and the images. In future work, we aim to collect and validate vast data with the help of specialists.

Author Contributions: Conceptualization S.A.-H.; methodology, S.A.-H. and M.E.B.M.; software, S.A.-H.; validation, S.A.-H. and A.A.; formal analysis, S.A.-H.; investigation, S.A.-H.; resources, S.A.-H. and A.A.; data curation, S.A.-H.; writing—original draft preparation, S.A.-H.; writing—review and editing, M.E.B.M.; visualization, S.A.-H.; supervision, M.E.B.M. and S.A.-A.; project administration, A.A.; funding acquisition, A.A. All authors have read and agreed to the published version of the manuscript.

Funding: This work was funded by the Deanship of Scientific Research at King Saud University through the initiative of DSR Graduate Students Research Support (GSR).

Institutional Review Board Statement: Not applicable.

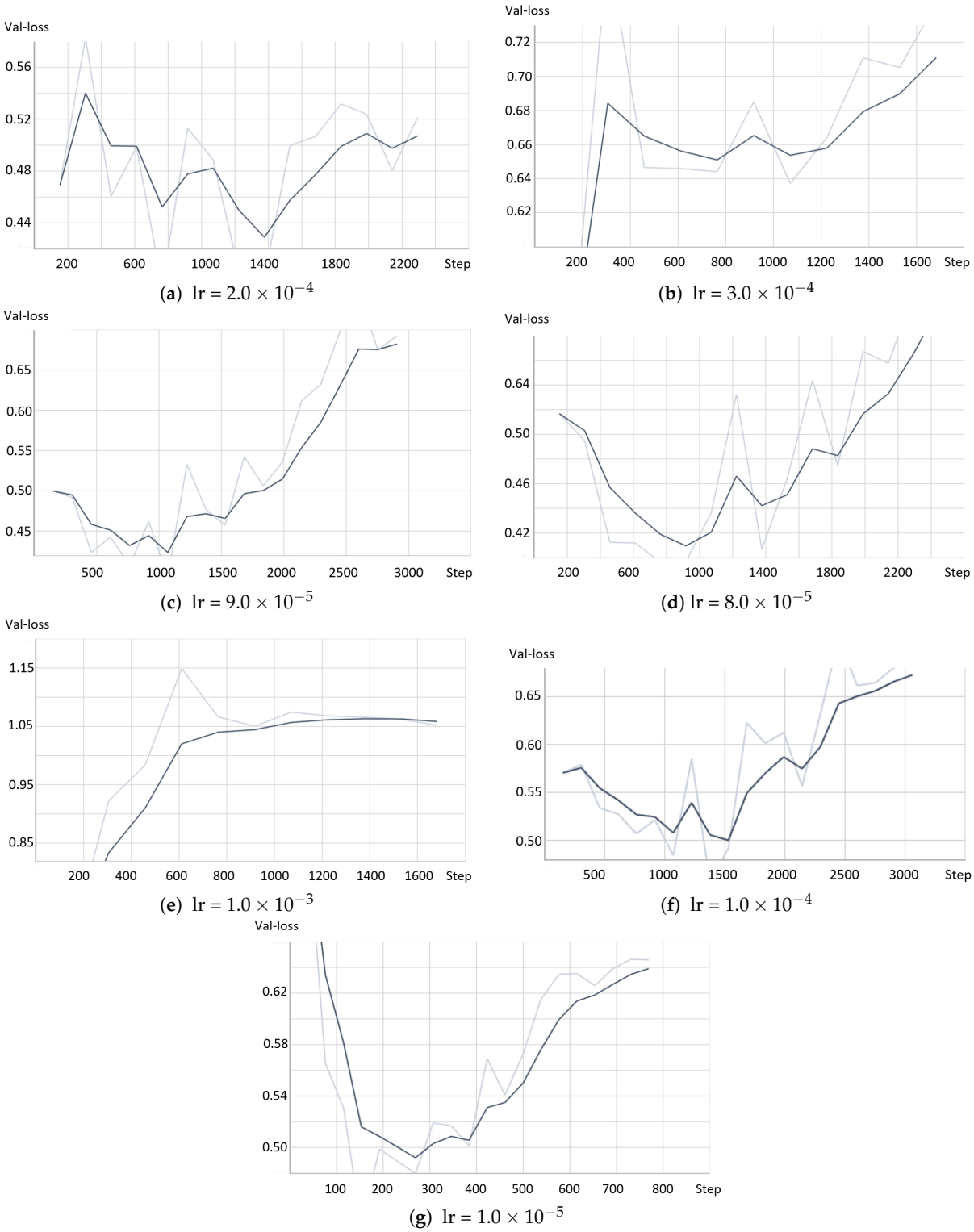
Informed Consent Statement: Not applicable.

Data Availability Statement: The dataset we used in the experiment is available at https://zenodo.org/record/6784358 (accessed on 1 April 2023).

Acknowledgments: The authors would like to thank King Saud University and the College of Computer and Information Sciences. Additionally, the authors would like to thank the Deanship of Scientific Research at King Saud University for funding and supporting this research through the initiative of DSR Graduate Students Research Support (GSR). This work was supported in part by KSU and the Center for Complex Engineering Systems (jointly between MIT and KACST).

Conflicts of Interest: The authors declare no conflict of interest.

Appendix A



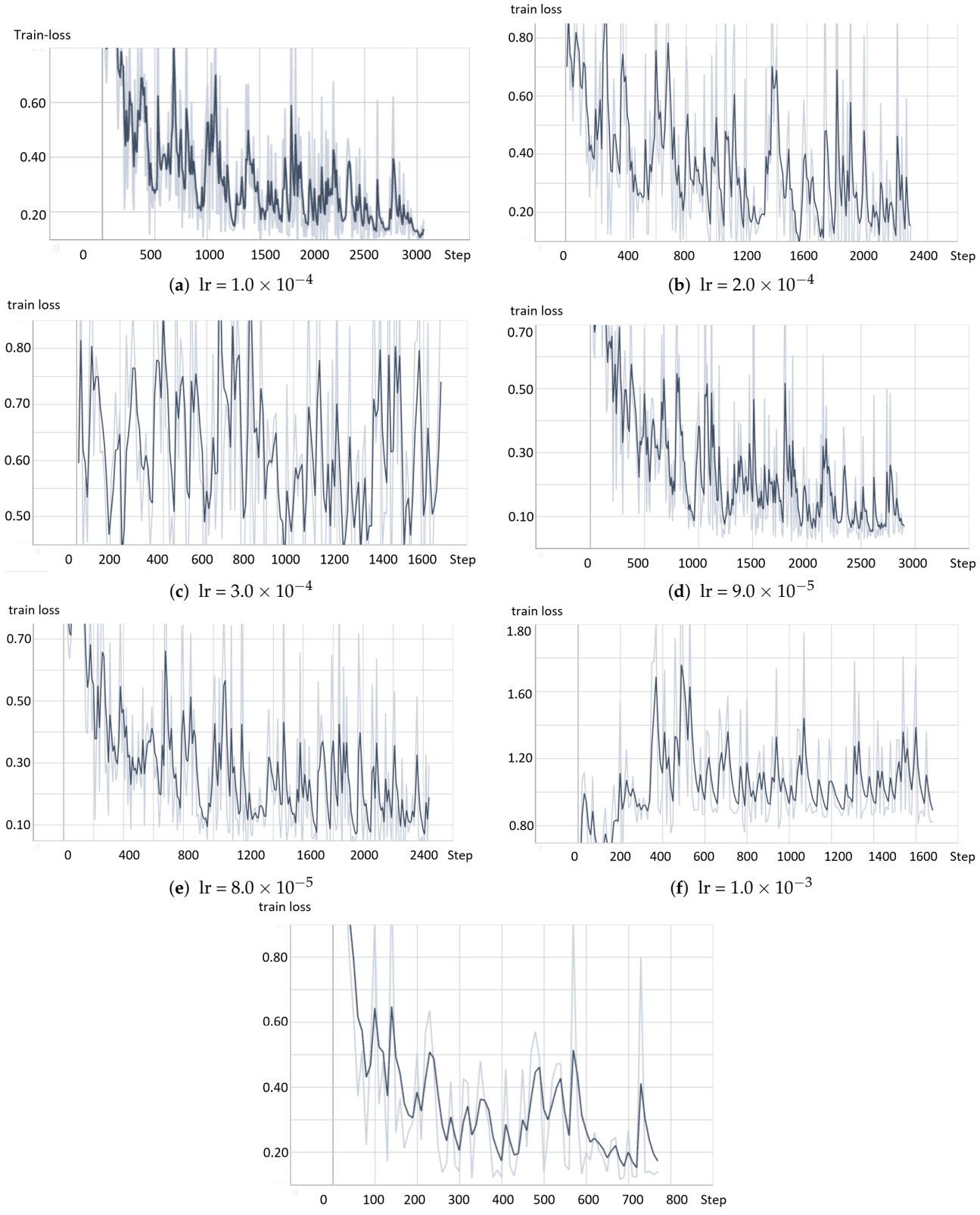
(**a**) lr = 2.0×10^{-4} (**b**) lr = 3.0×10^{-4}

(**c**) lr = 9.0×10^{-5} (**d**) lr = 8.0×10^{-5}

(**e**) lr = 1.0×10^{-3} (**f**) lr = 1.0×10^{-4}

(**g**) lr = 1.0×10^{-5}

Figure A1. The model validation loss using different learning rates and batch size = 32.



(**a**) lr = 1.0×10^{-4}

(**b**) lr = 2.0×10^{-4}

(**c**) lr = 3.0×10^{-4}

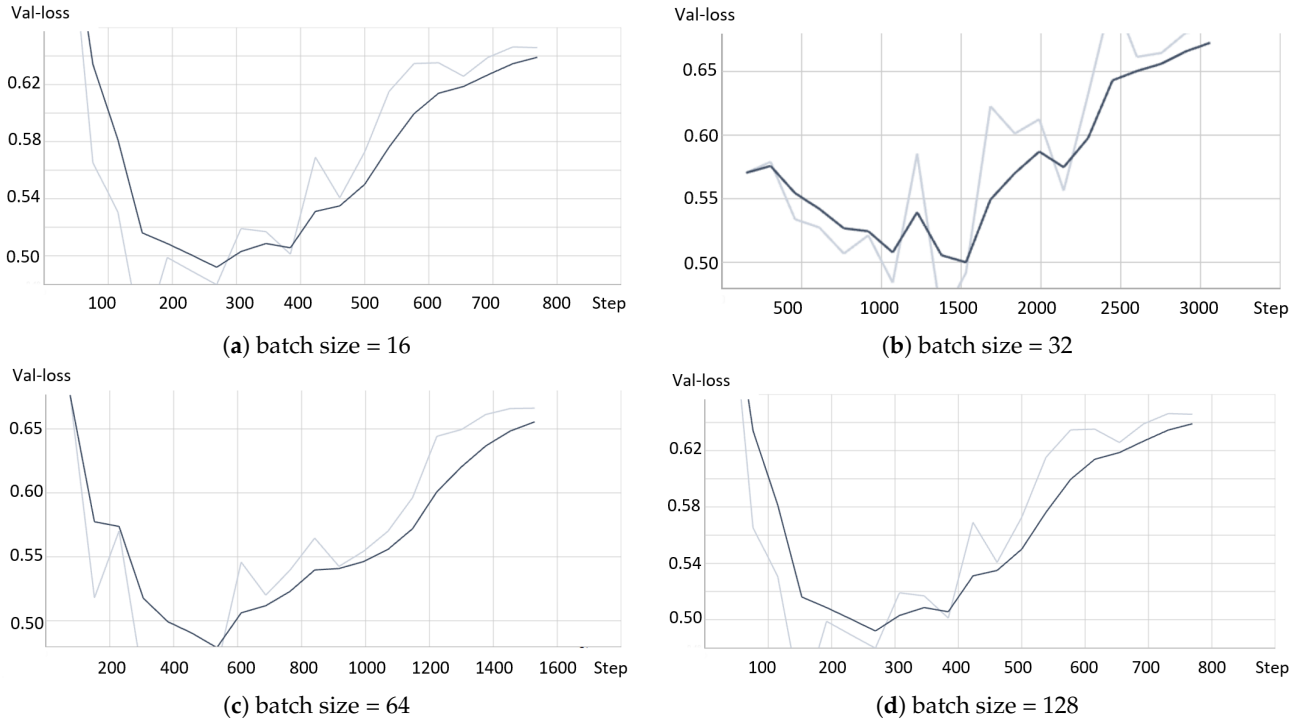
(**d**) lr = 9.0×10^{-5}

(**e**) lr = 8.0×10^{-5}

(**f**) lr = 1.0×10^{-3}

(**g**) lr = 1.0×10^{-5}

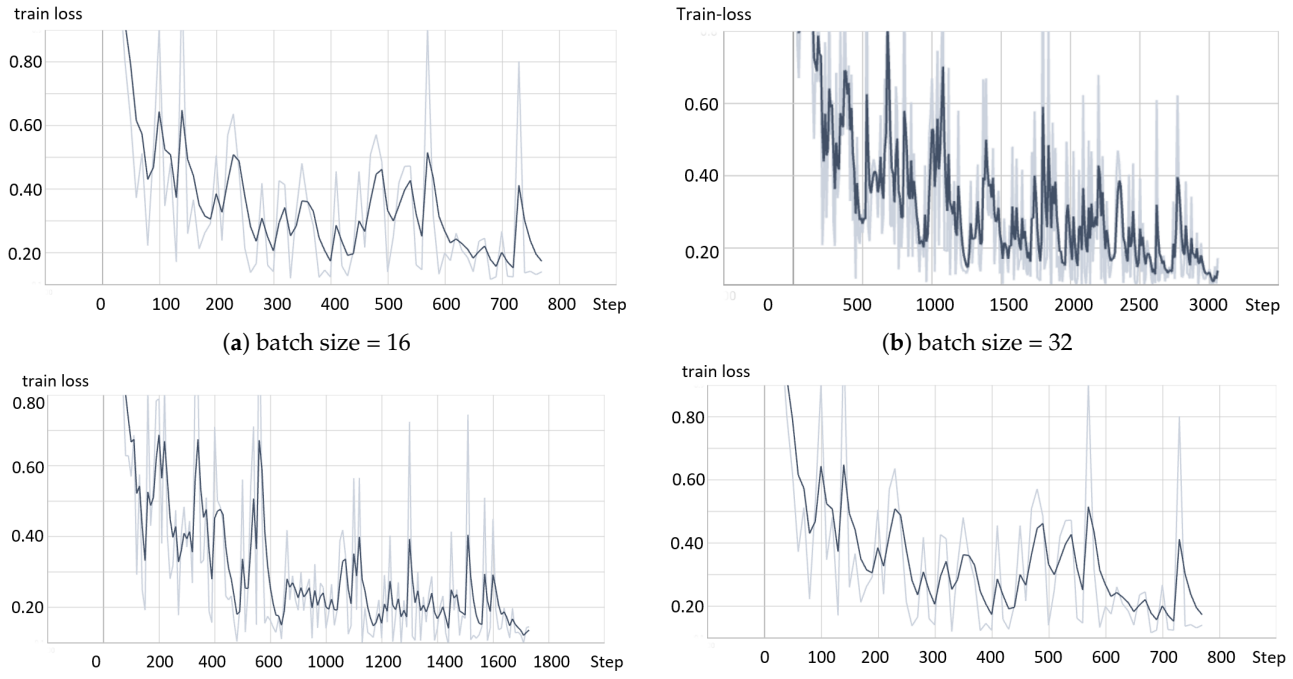
Figure A2. The model training loss using different learning rates and batch size = 32.



(**a**) batch size = 16 (**b**) batch size = 32

(**c**) batch size = 64 (**d**) batch size = 128

Figure A3. The model validation loss using different batch sizes and lr = 1.0×10^{-4}.



(**a**) batch size = 16 (**b**) batch size = 32

(**c**) batch size = 64 (**d**) batch size = 128

Figure A4. The model training loss using different batch sizes and lr = 1.0×10^{-4}.

References

1. Zhu, Y.; Groth, O.; Bernstein, M.; Fei-Fei, L. Visual7w: Grounded question answering in images. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–28 June 2016; pp. 4995–5004.
2. Abacha, A.B.; Hasan, S.A.; Datla, V.V.; Liu, J.; Demner-Fushman, D.; Müller, H. VQA-Med: Overview of the Medical Visual Question Answering Task at ImageCLEF 2019. In proceeding of Working Notes of CLEF 2019, Lugano, Switzerland, 9–12 September 2019.

3. Abacha, A.B.; Datla, V.V.; Hasan, S.A.; Demner-Fushman, D.; Müller, H. Overview of the VQA-Med Task at ImageCLEF 2020: Visual Question Answering and Generation in the Medical Domain. In Proceedings of the CLEF 2020—Conference and Labs of the Evaluation Forum , Thessaloniki, Greece, 22–25 September 2020; pp. 1–9.
4. Liu, B.; Zhan, L.M.; Xu, L.; Ma, L.; Yang, Y.; Wu, X.M. SLAKE: A Semantically-Labeled Knowledge-Enhanced Dataset for Medical Visual Question Answering. In Proceedings of the 2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI), Nice, France, 13–16 April 2021; pp. 1650–1654.
5. Tascon-Morales, S.; Márquez-Neila, P.; Sznitman, R. Consistency-Preserving Visual Question Answering in Medical Imaging. In *Medical Image Computing and Computer Assisted Intervention—MICCAI 2022, Proceedings of the 25th International Conference, Singapore, 18–22 September 2022*; Part VIII; Springer: Cham, Switzerland, 2022; pp. 386–395.
6. Ren, M.; Kiros, R.; Zemel, R. Image question answering: A visual semantic embedding model and a new dataset. *Proc. Adv. Neural Inf. Process. Syst.* **2015**, *1*, 5.
7. Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Zitnick, C.L.; Parikh, D. VQA: Visual question answering. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 2425–2433.
8. Malinowski, M.; Rohrbach, M.; Fritz, M. Ask your neurons: A neural-based approach to answering questions about images. In Proceedings of the IEEE International Conference on Computer Vision, Santiago, Chile, 7–13 December 2015; pp. 1–9.
9. Jiang, A.; Wang, F.; Porikli, F.; Li, Y. Compositional memory for visual question answering. *arXiv* **2015**, arXiv:1511.05676.
10. Chen, K.; Wang, J.; Chen, L.C.; Gao, H.; Xu, W.; Nevatia, R. ABC-CNN: An Attention Based Convolutional Neural Network for Visual Question Answering. *arXiv* **2015**, arXiv:1511.05960v2.
11. Ilievski, I.; Yan, S.; Feng, J. A focused dynamic attention model for visual question answering. *arXiv* **2016**, arXiv:1604.01485.
12. Andreas, J.; Rohrbach, M.; Darrell, T.; Klein, D. Neural module networks. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–28 June 2016; pp. 39–48.
13. Song, J.; Zeng, P.; Gao, L.; Shen, H.T. From pixels to objects: Cubic visual attention for visual question answering. *arXiv* **2022**, arXiv:2206.01923.
14. Andreas, J.; Rohrbach, M.; Darrell, T.; Klein, D. Learning to compose neural networks for question answering. *arXiv* **2016**, arXiv:1601.01705.
15. Xiong, C.; Merity, S.; Socher, R. Dynamic memory networks for visual and textual question answering. In Proceedings of the International Conference on Machine Learning, PMLR, New York City, NY, USA, 20–22 June 2016; pp. 2397–2406.
16. Kumar, A.; Irsoy, O.; Ondruska, P.; Iyyer, M.; Bradbury, J.; Gulrajani, I.; Zhong, V.; Paulus, R.; Socher, R. Ask me anything: Dynamic memory networks for natural language processing. In Proceedings of the International Conference on Machine Learning, PMLR, New York City, NY, USA, 20–22 June 2016; pp. 1378–1387.
17. Noh, H.; Han, B. Training recurrent answering units with joint loss minimization for VQA. *arXiv* **2016**, arXiv:1606.03647.
18. Gao, L.; Zeng, P.; Song, J.; Li, Y.F.; Liu, W.; Mei, T.; Shen, H.T. Structured two-stream attention network for video question answering. *Proc. AAAI Conf. Artif. Intell.* **2019**, *33*, 6391–6398. [CrossRef]
19. Wang, P.; Wu, Q.; Shen, C.; Hengel, A.v.d.; Dick, A. Explicit knowledge-based reasoning for visual question answering. *arXiv* **2015**, arXiv:1511.02570.
20. Wang, P.; Wu, Q.; Shen, C.; Dick, A.; Van Den Hengel, A. FVQA: Fact-based visual question answering. *IEEE Trans. Pattern Anal. Mach. Intell.* **2017**, *40*, 2413–2427. [CrossRef]
21. Wu, Q.; Wang, P.; Shen, C.; Dick, A.; Van Den Hengel, A. Ask me anything: Free-form visual question answering based on knowledge from external sources. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Las Vegas, NV, USA, 27–28 June 2016; pp. 4622–4630.
22. Simonyan, K.; Zisserman, A. Very Deep Convolutional Networks for Large-Scale Image Recognition. *arXiv* **2015**, arXiv:1409.1556v6.
23. He, K.; Zhang, X.; Ren, S.; Sun, J. Deep Residual Learning for Image Recognition. In Proceedings of the CVPR 2016, Las Vegas, NV, USA, 27–28 June 2016; pp. 770–778.
24. Nguyen, B.D.; Do, T.T.; Nguyen, B.X.; Do, T.; Tjiputra, E.; Tran, Q.D. Overcoming Data Limitation in Medical Visual Question Answering. In *Proceedings of the International Conference on Medical Image Computing and Computer-Assisted Intervention*; Springer: Cham, Switzerland, 2019; pp. 522–530.
25. Do, T.; Nguyen, B.X.; Tjiputra, E.; Tran, M.; Tran, Q.D.; Nguyen, A. Multiple Meta-model Quantifying for Medical Visual Question Answering. *arXiv* **2021**, arXiv:2105.08913.
26. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
27. Schuster, M.; Paliwal, K.K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
28. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding *arXiv* **2019**, arXiv:1810.04805.
29. Peng, Y.; Liu, F.; Rosen, M.P. UMass at ImageCLEF Medical Visual Question Answering (Med-VQA) 2018 Task. In Proceedings of the CEUR Workshop, Avignon, France, 10–14 September 2018.
30. Lu, J.; Yang, J.; Batra, D.; Parikh, D. Hierarchical question-image co-attention for visual question answering. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 289–297.
31. Shi, Y.; Furlanello, T.; Zha, S.; Anandkumar, A. Question Type Guided Attention in Visual Question Answering. In Proceedings of the ECCV 2018, Munich, Germany, 8–14 September 2018; pp. 151–166.

32. Li, L.H.; Yatskar, M.; Yin, D.; Hsieh, C.J.; Chang, K.W. VisualBERT: A simple and performant baseline for vision and language. *arXiv* **2019**, arXiv:1908.03557.
33. Lu, J.; Batra, D.; Parikh, D.; Lee, S. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Adv. Neural Inf. Process. Syst.* **2019**, *32*, 1–11 .
34. Chen, Y.C.; Li, L.; Yu, L.; El Kholy, A.; Ahmed, F.; Gan, Z.; Cheng, Y.; Liu, J. Uniter: Universal image-text representation learning. In Proceedings of the Computer Vision—ECCV 2020, 16th European Conference, Glasgow, UK, 23–28 August 2020; pp. 104–120.
35. Radford, A.; Kim, J.W.; Hallacy, C.; Ramesh, A.; Goh, G.; Agarwal, S.; Sastry, G.; Askell, A.; Mishkin, P.; Clark, J.; et al. Learning transferable visual models from natural language supervision. In Proceedings of the International Conference on Machine Learning, PMLR, Virtual only, 18–24 July 2021; pp. 8748–8763.
36. Cong, F.; Xu, S.; Guo, L.; Tian, Y. Caption-Aware Medical VQA via Semantic Focusing and Progressive Cross-Modality Comprehension. In Proceedings of the 30th ACM International Conference on Multimedia, Lisbon, Portugal, 10–14 October 2022; pp. 3569–3577.
37. Liu, Z.; Lin, Y.; Cao, Y.; Hu, H.; Wei, Y.; Zhang, Z.; Lin, S.; Guo, B. Swin transformer: Hierarchical vision transformer using shifted windows. In Proceedings of the IEEE/CVF International Conference on Computer Vision, Montreal, BC, Canada, 10–17 October 2021; pp. 10012–10022.
38. Wortsman, M.; Ilharco, G.; Gadre, S.Y.; Roelofs, R.; Gontijo-Lopes, R.; Morcos, A.S.; Namkoong, H.; Farhadi, A.; Carmon, Y.; Kornblith, S.; et al. Model soups: Averaging weights of multiple fine-tuned models improves accuracy without increasing inference time. In Proceedings of the International Conference on Machine Learning, PMLR, Baltimore, MA, USA, 17–23 July 2022; pp. 23965–23998.
39. Lowe, D.G. Object recognition from local scale-invariant features. In Proceedings Piscataway, NJ, USA of the Seventh IEEE International Conference on Computer Vision, Kerkyra, Greece, 20–27 September 1999, Volume 2, pp. 1150–1157.
40. Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 20–25 June 2005; Volume 1, pp. 886–893.
41. Lienhart, R.; Maydt, J. An extended set of Haar-like features for rapid object detection. In Proceedingsof the IEEE International Conference on Image Processing, Rochester, NY, USA, 22–25 September 2002; pp. 900–903. [CrossRef]
42. Krizhevsky, A.; Sutskever, I.; Hinton, G.E. ImageNet Classification with Deep Convolutional Neural Networks. *Commun. ACM* **2017**, *60*, 84–90. [CrossRef]
43. Zhang, D.; Cao, R.; Wu, S. Information fusion in visual question answering: A Survey. *Inf. Fusion* **2019**, *52*, 268–280. [CrossRef]
44. Abacha, A.B.; Gayen, S.; Lau, J.J.; Rajaraman, S.; Demner-Fushman, D. *NLM at ImageCLEF 2018 Visual Question Answering in the Medical Domain*; In Proceedings of Working Notes of CLEF 2018, Avignon, France, 10–14 September 2018.
45.]Verma, H.; Ramachandran, S. HARENDRAKV at VQA-Med 2020: Sequential VQA with Attention for Medical Visual Question Answering. In Proceedings of Working Notes of CLEF 2018, Thessaloniki, Greece, 22–25 September 2020.
46. Bounaama, R.; Abderrahim, M.E.A. Tlemcen University at ImageCLEF 2019 Visual Question Answering Task. In Proceedings of Working Notes of CLEF 2018, Thessaloniki, Lugano, Switzerland, 9–12 September 2019..
47. Szegedy, C.; Liu, W.; Jia, Y.; Sermanet, P.; Reed, S.; Anguelov, D.; Erhan, D.; Vanhoucke, V.; Rabinovich, A. Going deeper with convolutions. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, Boston, MA, USA, 5–12 June 2015; pp. 1–9.
48. Fukui, A.; Park, D.H.; Yang, D.; Rohrbach, A.; Darrell, T.; Rohrbach, M. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, Austin, TC, USA, 1–5 November 2016; pp. 457–468.
49. Kim, J.H.; On, K.W.; Lim, W.; Kim, J.; Ha, J.W.; Zhang, B.T. Hadamard Product for Low-rank Bilinear Pooling. In Proceedings of the 5th International Conference on Learning Representations, ICLR Toulon, France, 24–26 April 2017 .
50. Ben-Younes, H.; Cadene, R.; Cord, M.; Thome, N. MUTAN: Multimodal Tucker Fusion for Visual Question Answering. In Proceedingsof the IEEE International Conference on Computer Vision, Venice, Italy, 22–29 October 2017; pp. 2612–2620.
51. Huang, J.; Chen, Y.; Li, Y.; Yang, Z.; Gong, X.; Wang, F.L.; Xu, X.; Liu, W. Medical knowledge-based network for Patient-oriented Visual Question Answering. *Inf. Process. Manag.* **2023**, *60*, 103241. [CrossRef]
52. Haridas, H.T.; Fouda, M.M.; Fadlullah, Z.M.; Mahmoud, M.; ElHalawany, B.M.; Guizani, M. MED-GPVS: A Deep Learning-Based Joint Biomedical Image Classification and Visual Question Answering System for Precision e-Health. In Proceedings of the ICC 2022—IEEE International Conference on Communications, Seoul, Republic of Korea, 15–18 August 2022; pp. 3838–3843.
53. Kovaleva, O.; Shivade, C.; Kashyap, S.; Kanjaria, K.; Wu, J.; Ballah, D.; Coy, A.; Karargyris, A.; Guo, Y.; Beymer, D.B.; et al. Towards Visual Dialog for Radiology. In Proceedings of the 19th SIGBioMed Workshop on Biomedical Language Processing, Online, 9 July 2020; pp. 60–69. [CrossRef]
54. Liao, Z.; Wu, Q.; Shen, C.; van den Hengel, A.; Verjans, J. AIML at VQA-Med 2020: Knowledge Inference via a Skeleton-based Sentence Mapping Approach for Medical Domain Visual Question Answering. In Proceedings of Working Notes of CLEF 2020, Thessaloniki, Greece, 22–25 September 2020.
55. Gong, H.; Huang, R.; Chen, G.; Li, G. SYSU-Hcp at VQA-MED 2021: A data-centric model with efficient training methodology for medical visual question answering. In Proceedings of the Working Notes of CLEF 2021, Bucharest, Romania, 21–24 September 2021; Volume 201.

56. Wang, H.; Pan, H.; Zhang, K.; He, S.; Chen, C. M2FNet: Multi-granularity Feature Fusion Network for Medical Visual Question Answering. In Proceedings of the PRICAI 2022: Trends in Artificial Intelligence, 19th Pacific Rim International Conference on Artificial Intelligence, PRICAI 2022, Shanghai, China, 10–13 November 2022; Part II; Springer: Cham, Switzerland, 2022; pp. 141–154.
57. Wang, M.; He, X.; Liu, L.; Qing, L.; Chen, H.; Liu, Y.; Ren, C. Medical visual question answering based on question-type reasoning and semantic space constraint. *Artif. Intell. Med.* **2022**, *131*, 102346. [CrossRef] [PubMed]
58. Manmadhan, S.; Kovoor, B.C. Visual question answering: A state-of-the-art review. *Artif. Intell. Rev.* **2020**, *53*, 5705–5745. [CrossRef]
59. He, X.; Zhang, Y.; Mou, L.; Xing, E.; Xie, P. PathVQA: 30.000+ questions for medical visual question answering . *arXiv* **2020**, arXiv:2003.10286.
60. Allaouzi, I.; Benamrou, B.; Benamrou, M.; Ahmed, M.B. Deep Neural Networks and Decision Tree Classifier for Visual Question Answering in the Medical Domain. In Proceedings of Working Notes of CLEF 2018, Avignon, France, 10–14 September 2018.
61. Zhou, Y.; Kang, X.; Ren, F. Employing Inception-Resnet-v2 and Bi-LSTM for Medical Domain Visual Question Answering. In Proceedings of Working Notes of CLEF 2018, Avignon, France, 10–14 September 2018.
62. Talafha, B.; Al-Ayyoub, M. JUST at VQA-Med: A VGG-Seq2Seq Model. In Proceedings of Working Notes of CLEF 2018, Avignon, France, 10–14 September 2018.
63. Vu, M.H.; Lofstedt, T.; Nyholm, T.; Sznitman, R. A Question-Centric Model for Visual Question Answering in Medical Imaging. *IEEE Trans. Med. Imaging* **2020**, *39*, 2856–2868. . [CrossRef] [PubMed]
64. Kiros, R.; Zhu, Y.; Salakhutdinov, R.; Zemel, R.S.; Torralba, A.; Urtasun, R.; Fidler, S. Skip-Thought Vectors. *Adv. Neural Inf. Process. Syst.* **2015**, *28*, 3294–3302.
65. Yang, Z.; Dai, Z.; Yang, Y.; Carbonell, J.; Salakhutdinov, R.; Le, Q.V. XLNet: Generalized Autoregressive Pretraining for Language Understanding. In Proceedings of the 33rd Advances in Neural Information Processing Systems, Vancouver, BC, Canada, 8–14 December 2019; pp. 5753–5763.
66. Eslami, S.; de Melo, G.; Meinel, C. Teams at VQA-MED 2021: BBN-orchestra for long-tailed medical visual question answering. In Proceedings of the Working Notes of CLEF 2021, Bucharest, Romania, 21–24 September 2021; pp. 1211–1217.
67. Schilling, R.; Messina, P.; Parra, D.; Lobel, H. Puc Chile team at VQA-MED 2021: Approaching VQA as a classfication task via fine-tuning a pretrained CNN.In Proceedings of the Working Notes of CLEF 2021, Bucharest, Romania, 21–24 September 2021; pp. 346–351.
68. Zhou, Y.; Jun, Y.; Chenchao, X.; Jianping, F.; Dacheng, T. Beyond Bilinear: Generalized Multimodal Factorized High-Order Pooling for Visual Question Answering. *IEEE Trans. Neural Netw. Learn. Syst.* **2018**, *29*, 5947–5959.
69. Malinowski, M.; Rohrbach, M.; Fritz, M. Ask Your Neurons: A Deep Learning Approach to Visual Question Answering. *Int. J. Comput. Vis.* **2017**, *125*, 110–135. . [CrossRef]
70. Kuniaki, S.; Andrew, S.; Yoshitaka, U.; Tatsuya, H. Dualnet: Domain-invariant network for visual question answering. In Proceedings of the the IEEE International Conference on Multimedia and Expo (ICME) 2017, Hong Kong, 10–14 July 2017; pp. 829–834.
71. Noh, H.; Seo, P.H.; Han, B. Image Question Answering Using Convolutional Neural Network with Dynamic Parameter Prediction. In Proceedings of the CVPR, Las Vegas, NV, USA, 27–30 June 2016; pp. 30–38.
72. Kim, J.H.; Lee, S.W.; Kwak, D.; Heo, M.O.; Kim, J.; Ha, J.W.; Zhang, B.T. Multimodal residual learning for visual QA. *Adv. Neural Inf. Process. Syst.* **2016**, *29*, 361–369 .
73. Mingrui, L.; Yanming, G.; Hui, W.; Xin, Z. Cross-modal multistep fusion network with co-attention for visual question answering. *IEEE Access* **2018**, *6*, 31516–31524.
74. Bai, Y.; Fu, J.; Zhao, T.; Mei, T. Deep Attention Neural Tensor Network for Visual Question Answering. In Proceedings of the ECCV 2018, Munich, Germany, 8–14 September 2018; pp. 20–35.
75. Narasimhan, M.; Schwing, A.G. Straight to the Facts: Learning Knowledge Base Retrieval for Factual Visual Question Answering. In Proceedings of the ECCV 2018, Munich, Germany, 8–14 September 2018; pp. 451–468.
76. Chen, L.; Yan, X.; Xiao, J.; Zhang, H.; Pu, S.; Zhuang, Y. Counterfactual Samples Synthesizing for Robust Visual Question Answering. In Proceedings of the Computer Vision and Pattern Recognition, CVPR 2020, Seattle, WA, USA, 14–19 June 2020; pp. 10800–10809.
77. Clark, K.; Luong, M.T.; Le, Q.V.; Manning, C.D. Electra: Pre-training text encoders as discriminators rather than generators. *arXiv* **2020**, arXiv:2003.10555.
78. Rumelhart, D.E.; Hinton, G.E.; McClelland, J.L. A general framework for parallel distributed processing. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition*; MIT Press: Cambridge, MA, USA, 1986; Volume 1, pp. 45–76. p. 26.
79. Maas, A.L.; Hannun, A.Y.; Ng, A.Y. Rectifier nonlinearities improve neural network acoustic models. In Proceedings of the ICML, Atlanta, GA, USA, 16–21 June 2013; Volume 30, p. 3.
80. Loshchilov, I.; Hutter, F. Decoupled weight decay regularization. *arXiv* **2017**, arXiv:1711.05101.
81. Porwal, P.; Pachade, S.; Kamble, R.; Kokare, M.; Deshmukh, G.; Sahasrabuddhe, V.; Meriaudeau, F. Indian diabetic retinopathy image dataset (IDRiD): A database for diabetic retinopathy screening research. *Data* **2018**, *3*, 25. [CrossRef]

82. Decenciere, E.; Cazuguel, G.; Zhang, X.; Thibault, G.; Klein, J.C.; Meyer, F.; Marcotegui, B.; Quellec, G.; Lamard, M.; Danno, R.; et al. TeleOphta: Machine learning and image processing methods for teleophthalmology. *IRBM* **2013**, *34*, 196–203. [CrossRef]
83. Selvaraju, R.R.; Tendulkar, P.; Parikh, D.; Horvitz, E.; Ribeiro, M.T.; Nushi, B.; Kamar, E. Squinting at VQA models: Introspecting vqa models with sub-questions. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, Seattle, WA, USA, 14–19 June 2020; pp. 10003–10011.