Deep Learning for Medical Visual Question Answering

João Daniel Rua Ferreira Pires da Silva

Thesis to obtain the Master of Science Degree in

Information Systems and Computer Engineering

Supervisors: Prof. Bruno Emanuel da Graça Martins
Prof. João Miguel da Costa Magalhães

Examination Committee

Chairperson: Prof. Luís Manuel Antunes Veiga
Supervisor: Prof. Bruno Emanuel da Graça Martins
Members of the Committee: Prof. Allan Hanbury

November 2021

Acknowledgments

I would like to thank my professors, Prof. Bruno Martins and Prof. João Magalhães, for their regular feedback, support and guidance during the development of this M.Sc dissertation. In particular, thank you Prof. Bruno Martins for the constant help and insight throughout the various steps of this work and in reviewing many times all the documents written during this year. Thank you also Prof. João Magalhães for providing me access to the cluster of NOVA LINCS, which greatly improved my ability to test the different necessary methods during this work.

I would like to thank my parents, for teaching me the values of empathy and the joy of learning, and for doing their best so I can succeed and achieve my goals. Thank you to my friends, which are very dear to me and provided many moments of which I will keep with me. In particular, thank you Bruno and Francisco, for being my friends almost since day 1 and for all our moments together, both in academia and fun. Thank you João Barata, for your support and being a close friend. I know I can always rely on you. To Tiago Mesquita for your friendship and support during this last journey. To Alexandre Guerra for the bike rides. To Alexandre and Guilherme for being such wild friends. Thank you Daniel Sil, for your friendship and worry despite the distance in the later years. May we all also keep tight after Técnico. Thank you Henrique for your friendship and keeping in touch.

Thank you to my girlfriend Cecília, for the constant emotional support, partnership and love throughout all these years, and for our growth together. You have helped me so much.

Also, thank you to my extended family, which always believed in me and rooted for my best.

Thank you all.

Abstract

Models for Visual Question Answering (VQA) on medical images should answer diagnostically relevant natural language questions with basis on visual contents. A recent study in the area proposed MMBERT (Khare et al., 2021), a multi-modal encoder model that combines a ResNet backbone to represent images at multiple resolutions, together with a Transformer encoder. By pre-training the model over the Radiology Objects in COntext (ROCO) dataset of images+captions, using a masked language modeling objective that also considered the image features for attempting to reconstruct the masked tokens, the authors achieved state-of-the-art performance on the VQA-Med dataset of questions over radiology images, used in ImageCLEF 2019. Taking the source code provided by the authors, we first attempted to reproduce the results for MMBERT, afterwards extended the model in several directions: (a) using a stronger image encoder based on EfficientNetV2, (b) using a multi-modal encoder based on the RealFormer architecture, (c) extending the pre-training task with a contrastive objective, and (d) using a novel loss function for fine-tuning the model to the VQA task, that specifically considers class imbalance. Exactly reproducing the results published for MMBERT was met with some difficulties, and the default hyper-parameters given in the original source code resulted in a lower performance. Our experiments showed that aspects such as the size of the training batches can significantly affect the performance. Moreover, starting from baseline results corresponding to our reproduction of MMBERT, we also show that the proposed extensions can lead to improvements. The source code associated to our tests is given at `https://github.com/DannielSilva/MMBERT`.

Keywords

Medical Visual Question Answering; Transformer Encoders; Computer Vision; Natural Language Processing; Biomedical Informatics;

Resumo

Modelos para a tarefa de responder a perguntas sobre imagens (*Visual Question Answering*, em inglês) no domínio médico devem conseguir responder a perguntas relevantes com base nos conteúdos das imagens médicas. Um estudo recente na área propôs MMBERT (Khare et al., 2021), um modelo codificador com dados multimodais que combina uma ResNet para representar as imagens em múltiplas resoluções, juntando um codificador Transformer. Pre-treinando o modelo sobre o dataset Radiology Objects in COntext (ROCO), consistindo de imagens+legendas, com um objectivo de mascaramento de linguagem que também considera os conteúdos da imagem na tentativa de reconstrução dos tokens mascarados, os autores conseguiram atingir resultados estado-de-arte no dataset VQA-MED de questões sobre imagens de radiologia, usado no ImageCLEF 2019. A partir do código disponibilizado pelos autores, primeiramente tentámos reproduzir os resultados do MMBERT, e posteriormente avançamos em diversas direções: (a) um codificador de imagem mais forte com base na EfficientNetV2; (b) um codificador multi-modal com base na arquitetura RealFormer; (c) extender a tarefa de pre-treino com um objetivo de contraste, e (d) uma função de custo recente para afinar o modelo para a tarefa de VQA, que especificamente considera um desequilíbrio de classes. Foram encontradas algumas dificuldades em reproduzir exatamente os resultados do MMBERT, e os hiper-parâmetros pré-definidos no código original resultaram em resulados inferiores. Apartir dos resultados base da nossa reprodução do MMBERT, mostramos que as extensões ao modelo propostas resultam em melhorias. O código usado pode ser consultado em `https://github.com/DannielSilva/MMBERT`.

Palavras Chave

Resposta a Perguntas sobre Conteúdos Visuais no domínio médico; Transformadores; Processamento de Imagens; Processamento de Linguagem Natural; Informática Biomédica;

Contents

List of Figures

List of Tables

1 Introduction

Contents

1.1 Objectives

With the increasing interest in Artificial Intelligence (AI) to support clinical decision making and improve patient engagement, many opportunities related to automated medical image interpretation are currently being explored. Radiology is a medical domain which can greatly benefit from these approaches, to streamline workflows, to alleviate workload burdens, or to reallocate medical staff to more complex and needed tasks with higher priority. Visual Question Answering (VQA) is an exciting research problem that combines natural language processing and computer vision techniques, currently attracting a significant interest. Some previous efforts have looked into VQA in the context of radiology images (Ben Abacha et al., 2020, 2019), where systems can provide additional insights to clinicians, or help patients in the interpretation of their medical images.

A particular formulation for the VQA problem involves taking a medical image (e.g., a radiology image) accompanied by a clinically relevant question, and producing as output a relevant answer based on the image contents. The process is illustrated in Figure 1.1. State-of-the-art approaches for addressing the VQA problem are based on deep neural networks, with systems often following an architecture that features: a first encoder component corresponding to a pre-trained convolutional neural network that generates a representation for the image; a second encoder corresponding to a recurrent neural network or, more recently, a Transformer, that is used to generate a representation for the question; a fusion and classification component that combines both types of information and selects an answer from a set of candidates, thus treating the VQA problem as a multi-label classification task.

Approaches based on deep learning have obtained good results in VQA for the medical domain. This dissertation describes a model based on multi-modal Transformers capable of addressing VQA in the medical domain, in particular the VQA-MED Task of ImageCLEF 2019 (Ben Abacha et al., 2019). In particular, using as baseline a strong model developed by Khare et al. (2021), this work describes several extensions in order to further improve the results.

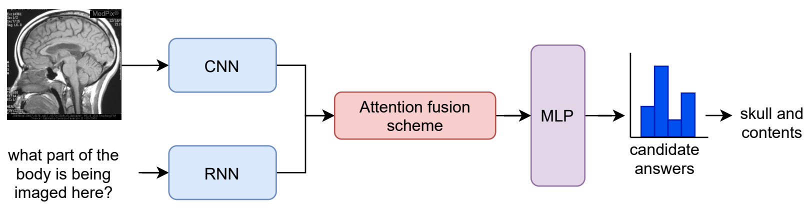


Figure 1.1: Graphical depiction of a traditional method for Visual Question Answering, in the medical domain, with an attention scheme to fuse the features from the image and question, resulting in a representation which is fed to a multi-layer perceptron that obtains an answer in a classification setting.

1.2 Methodology

Considering the aforementioned formulation for medical VQA, this work reproduces and extends the Multimodal BERT (MMBERT) study (Khare et al., 2021), reporting scores of 62.4% overall accuracy and 64.4% overall BLEU. In brief, Khare et al. (2021) proposed a multi-modal encoder that achieved state-of-the-art performance on the VQA-MED dataset from ImageCLEF 2019 (Ben Abacha et al., 2019). In this architecture, the visual representations are obtained with a ResNet backbone at multiple resolutions, which are then combined with text representations through a Transformer encoder. The authors leverage the Radiology Objects in COntext (ROCO) dataset (Pelka et al., 2018), containing medical images paired to textual captions, for model pre-training with a Masked Language Modeling (MLM) objective.

Using the original source code provided by the authors, we first attempted to reproduce the results for MMBERT. This task was met with some difficulties, namely due to the fact that hyper-parameters such as the batch size, the learning rate, or the stopping criteria, can have a strong influence on the results. After this, in order to further improve the results of MMBERT, we extended the model with several recent (all from years 2020 and 2021) state-of-the-art results methods:

- The initial ResNet image encoder was replaced by a stronger image encoder, namely the recent EfficientNetV2 (Tan and Le, 2021);
- The multi-modal encoder is now based on the RealFormer architecture (He et al., 2020), which extends the Transformer architecture Vaswani et al. (2017);
- The pre-training task was extended with a contrastive learning objective (Khosla et al., 2021), in addition to masked language modeling;
- The final model fine-tuning to the VQA task takes into account the class imbalancement problem, with a recently proposed asymmetric loss function that targets this issue (Ben-Baruch et al., 2021);

1.3 Contributions

The main contributions of this thesis can be summarized as follows:

- A thorough reproducibility of the original MMBERT study addressing medical Visual Question Answering, considering different values for batch size, patience and stopping criteria;
- Extensions over MMBERT, including architecture changes, new pre-training tasks, and a novel multi-class classification loss. The impact of these changes was also evaluated and the corresponding code can be accessed in a GitHub repository[1].

[1] https://github.com/DannielSilva/MMBERT

- A model combining the different extensions, which is able to improve the performance of the original MMBERT method. In particular, advancements were noticed with the use of contrastive learning and stronger image encoders, and also by using a larger batch size and a large value of patience. The best model obtained 62.80% overall accuracy and 64.32% in BLEU, respectively.
- A visualization method is also proposed, based on gradient-weighted class activation maps (Grad-CAM) (Selvaraju et al., 2019), to allow for a better interpretability of the model;

During the development of this dissertation, a paper was developed and submitted to the reproducibility track of ECIR 2022, with the title *Reproducing and Extending Multimodal Medical BERT for Visual Question Answering*.

1.4 Thesis Outline

The rest of this document is organized as follows: Chapter 2 introduces fundamental concepts to allow a better understanding of this dissertation to the reader. Chapter 3 presents work related to Visual Question Answering, in general and in the medical domain, while also mentioning different training methods and types of models. Chapter 4 describes the original MMBERT approach developed by Khare et al. (2021) and details the different extensions proposed in this work. Chapter 5 presents the experimental results. Finally, Chapter 6 presents the main conclusions and proposes directions for future work.

2

Fundamental Concepts

Contents

This chapter introduces fundamental concepts related to Deep Learning regarding neural network models for natural language processing and computer vision, which are necessary for an adequate understanding of the proposed approach of this dissertation. The section starts by describing the perceptron, multi layer perceptrons and more complex models for image processing, such as convolutional neural networks, or for text processing namely recurrent neural networks and Transformers.

2.1 Introduction to Deep Learning

Frank Rosenblatt's paper on the perceptron model (Rosenblatt, 1957) paved the way for the research and development of artificial neural networks. The inspiration for this model is from the biological neurons which receive stimuli and, if a threshold is reached, transmit a signal across the synapse to another cell. The perceptron receives an input vector of N dimensions and computes the dot product between the input and a weight vector. The result is then fed into an activation function, originally the sign function but nowadays other non linear functions like the sigmoid or the hyperbolic tangent. Mathematically, the perceptron model can be written as shown in the following equation:

$$y = f(x) = g\left(\sum_{i=1}^{n} x_i w_i + b\right) = g(\mathbf{x} \cdot \mathbf{w} + b)\,. \tag{2.1}$$

In the above equation y denotes the output prediction, $\mathbf{x} = (x_1, ..., x_n)$ is the input vector, $\mathbf{w}$ is weight vector, b is a bias term, and g is the activation function. Given a labelled training set consisting of inputs x and the corresponding outputs y, the perceptron adapts the weight vector to the optimal values. The perceptron learning rule consists by starting the vector w to random small values, and check at each iteration if the output y has the target value t. The weights are updated by the following expression:

$$w^{new} = w^{old} + \eta(t - o)x_i\,. \tag{2.2}$$

If y is correct, then $\eta(t - o)x_i = 0$, so the weights do not change. Otherwise, they are updated according the above mentioned formula, in which η represents the learning rate, which determines how much should the weights change for each wrongly predicted observation. The perceptron is a binary and linear classifier, meaning that it can only be applied to differentiate between two classes that are linearly separable. To cover this limitation and support more complex tasks, the multi layer perceptron (MLP) can be used, which is an extension with multiple neurons connected by layers.

The MLP consists of at least three layers of nodes: an input layer, a hidden layer and an output layer. The architecture of a MLP can consist of multiple hidden layers and multiple neurons in each layer. The MLP is a fully connected feed-forward network, with each neuron in one layer connected to all neurons in the next layers, permitting the information to flow from the input layers to the output layers without any cycles. It can be proved that a MLP with at least one hidden layer is a universal approximator, which means the model is able to approximate any bounded continuous function with an arbitrarily small error (Leshno et al., 1993). With more hidden layers, the MLP is able to learn complex relations from the hierarchical combination of multiple features, and thus create high-order features. The study of neural networks with many layers is called Deep Learning. A MLP can also be described as a succession of matrix multiplications fed into activation functions. A feed-forward network with a single hidden layer can be described by the following equation, where $\mathbf{W_1}, b_1$ and $\mathbf{W_2}, b_2$ represent the weights and biases of the hidden and output layers, respectively:

$$\mathbf{y} = f(x) = g_2\left(g_1\left(\mathbf{x} \cdot \mathbf{W_1} + b_1\right) \cdot \mathbf{W_2} + b_2\right) . \tag{2.3}$$

Learning is an optimization task concerned with minimizing a loss function by using iterative gradient descent methods. The loss function compares the output prediction of the neural network with the ground truth after the input is processed by the network during the forward phase. Common loss functions include the cross-entropy for classification or the mean squared error for regression. The best parameters are computed using the backpropagation algorithm developed by Rumelhart et al. (1986). In this algorithm during the forward phase, the input is processed and the error is computed by the loss function. The error is then propagated back through the network, one layer at the time, and the weights are updated according to the amount they contributed to the error. This is done by computing the partial derivatives of the loss function relative to the weights and biases of the various layers, through the chain rule of diferentiation. This allows the application of the gradient descent method of the loss function E, considering the parameters θ:

$$\theta = \theta - \eta \cdot \nabla_\theta E(\theta) . \tag{2.4}$$

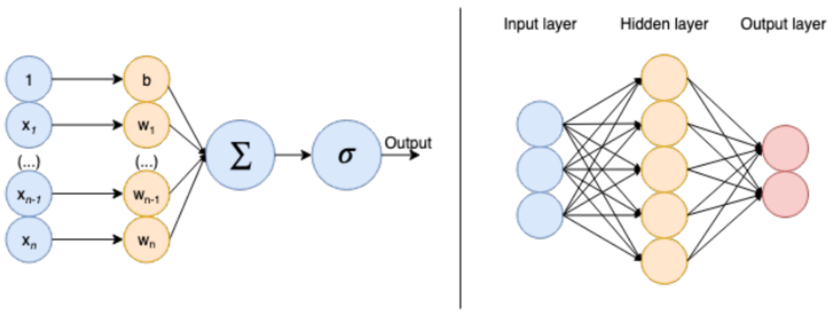


Figure 2.1: Perceptron model vs the multi-layer perceptron model.

2.2 Optimization Methods for Training Neural Networks

There are different variations on the gradient descent that can be used for model training, e.g. varying the amount of data that is used. There is a trade-off between the optimality of the parameter update and the necessary time to compute it. Batch training uses the whole training dataset to compute the gradient descent, which might be impossible for larger datasets because of the memory and time demanded for such one computation. In the opposite direction, there is Stochastic Gradient Descent (SGD) version where the gradient of the error is computed iteratively for a single training example, making learning faster and allowing online updates. However, SGD is prone to noise in the data, leading to updates with high variability and causing slower convergence rates. Mini-batch is a method that tries to achieve the advantages of former approaches, by selecting a fixed number of training examples.

Batch normalization is a method used to make the training of artificial neural networks faster and more stable through normalization of the input layer by re-centering and re-scaling. The method was originally developed by Ioffe and Szegedy (2015) to deal with internal covariate shift, a phenomenon causing the distribution of each layer's inputs to change during training, as the parameters of the previous layers change. Normalizing the input also prevents saturation of some activation functions like the sigmoid, prevents overflows and guarantees that the input values are in the same range so one feature doesn't dominate the prediction.

The value of the learning rate strongly influences training performance. A small value will lead to a slow convergence, while a learning rate that is too large can lead to suboptimal weights and in unstable training process. The following approaches address these problems to improve the training process and described in detail in a survey by Ruder (2017).

- Momentum leads to a faster gradient descent and reduced oscillation by adding a fraction γ of the update vector of the previous update to the current update vector. The current update is increased if both gradients point in the same direction and decreased when the gradient changes direction.

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta)\,, \\ \theta &= \theta - v_t\,. \end{aligned} \tag{2.5}$$

- Nesterov accelerated gradient also takes the idea of using the momentum term, but aims to prevent rushed jumps. The gradient is calculated not with respect to the current parameters θ, but with respect to the approximate future parameters already updated with the momentum term:

$$\begin{aligned} v_t &= \gamma v_{t-1} + \eta \nabla_\theta J(\theta - \gamma v_{t-1})\,, \\ \theta &= \theta - v_t\,. \end{aligned} \tag{2.6}$$

- Each individual parameter can be updated according to their importance. The Adagrad optimization algorithm applies this concept by adapting the learning rate to the parameters according to their update history, performing larger updates for infrequent parameters and smaller updates for more frequent ones being suitable when working with sparse datasets. Adadelt is an extension to the Adagrad algorithm that aims to prevent the agressive decreasing learning rate that can become infinitesimally small for frequently updated parameters, making it impossible for the algorithm to learn. Instead of inefficiently storing w previous squared gradients, in Adagrad we have that the sum of gradients is recursively defined as a decaying average of all past squared gradients. Adaptive Moment Estimation (Adam) keeps both the decaying average of past squared gradients like Adadelta and the decaying average of past gradients (although not squared).

Overfitting occurs when the error measured over the test set is much larger than that over the training set. This is a consequence of a model that is too complex and tries to fit the training data too closely in the presence of sampling noise, and is not able to generalize patterns from data. Some methods developed to reduce overfitting involve introducing weight penalties like L1 and L2 regularization: L1 regularization uses a term equal to the sum of the absolute values of the parameters while L2 regularization uses the sum of the squares of the parameters. In L1, weights can reach zero so the model can discard features (performing feature selection) that do not contribute in relations between data and the output (Ng, 2004).

Dropout is another type of regularization commonly used in the architecture of neural networks developed by Srivastava et al. (2014). This method approximates the goal of training an ensemble of a large number of networks to combine their predictions. This is achieved by randomly ignoring some nodes and their connections during training. If neurons are randomly dropped during training, other neurons will not overly depend on the information of others and their learning will be more robust.

2.3 Convolutional Neural Networks for Computer Vision

Convolutional Neural Networks (CNNs) are a type of deep neural network that is extensively applied in image processing. A problem for image processing using MLPs is the large dimensionality of the input. Even in images with low resolutions, the total amount of pixels and the weights (parameters) necessary for learning, such that there is a weight per pixel in each hidden layer, would be huge. CNNs use knowledge of the structure of the data, and assemble more complex patterns from smaller and simpler

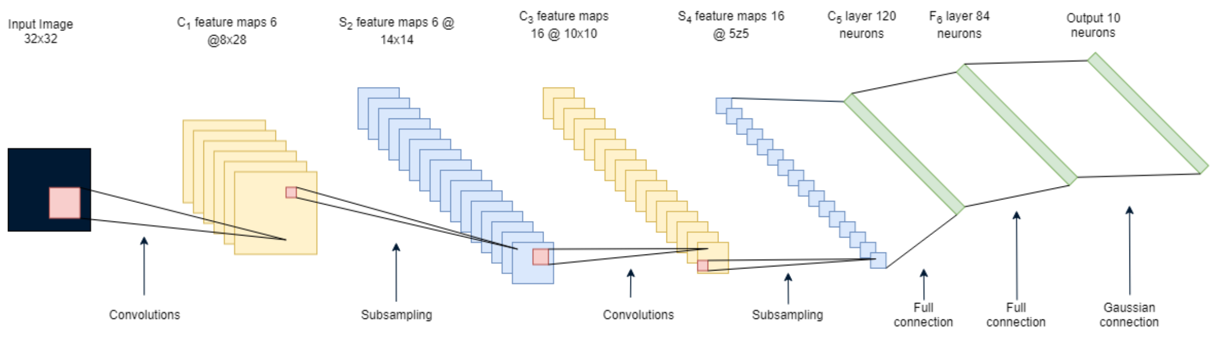


Figure 2.2: LeNet CNN architecture from Lecun et al. (1998).

ones. Thus, images can be reduced into a form which is easier to process, without losing features and important relations, taking into account spatial locality within images in a way that regular MLP cannot.

A CNN performs convolutions, non-linear projections and pooling operations over the input. The first layers recognize low-level features of image such as edges and corners while the last layers recognize high-level features dependent on the task. These operations act as automatic feature extractors and the results are then fed into a fully connected network which provides the final result of the network. A softmax function is usually used in the final layer resulting in a probability distribution across the possible labels for classification of the input.

The convolutional operation is done by a sliding window, named the kernel, over the input. It consists of a dot product between the input selected by the kernel and the weights of the kernel, resulting in a feature map. The number of pixels that the filter moves after each computation is called the Stride. A larger stride may skip some input positions, but this can be usefull to reduce the dimensionality of the feature map. To apply the convolutional operation to all input positions, padding can be used. This adds a frame of pixels around the image, usually with values equal to zeros (zero padding). Because images are usually comprised of more than one channel, the convolutional operation is applied to each channel.

A non-linear projection can be applied to the feature map using activation functions like the ReLU (Nair and Hinton, 2010) to capture non-linearities in data.

In turn, pooling operations also consist of a window sliding over the pixels, but a fixed operation is applied instead. Common pooling operations include max, min and average pooling. This process is done to decrease the resolution of the feature maps and remove noise, speeding up computations and allowing more generalization. However, if the resolution is too small, features can be lost too. Max pooling is commonly used, because it allows for edge detection.

The LeNet architecture was one of the first uses of CNNs for image classification, specifically for recognizing handwritten digits. It consists of two pairs of convolutional and pooling layers followed by two fully connected layers, with a softmax function in the last layer for the classification results.

Residual Connections and Denser Netwoks

Convolutional neural networks have consistently achieved state-of-the-art results over the last decade in image classification tasks. Much of this success has been credited for the addition of more layers, creating deeper networks. He et al. (2015) showed that by simply stacking more layers to a baseline model is not enough, as the deeper models often achieved worst performance. Vanishing gradients contribute to the problem, but He et al. (2015) hypothesize that if deeper models are worse than shallower, this is because they cannot map the identity function in the deeper layers and that this mapping is harder than mapping to 0. If multiple layers can map a general function $H(x)$, then they can also map $F(x) = H(x) - x$. Through the use of residual blocks, the final function to map becomes $H(x) = F(x) + x$, which makes the identity mapping possible by setting the residuals to 0. A residual block consists of a skip-connection between two layers, adding the input of the block to the output. To create even deeper networks a bottleneck design can also be used. This consists of a stack of three layers (instead of two) which are 1×1, 3×3 and 1×1 convolutions, respectively. The first 1×1 convolution allows for the computation of the 3×3 convolution in a smaller space, which improves performance. The output of the block is then restored to the original dimensions by the final 1×1 convolution, so the input can be added.

To further improve the information flow between layers a different connectivity pattern was proposed by Huang et al. (2017). DenseNet is a network architecture where each layer within a dense block is directly connected to every other layer in a feedforward fashion, so that a layer's input corresponds to the feature maps output from the previous layers. Each layer's output is propagated to every following layer within a dense block. Instead of summing the feature maps like in the case of a ResNet, the feature maps are aggregated with depth-concatenation, which prevents information loss. In each layer l, the input is fed to a function H_l that is a composition of three function: batch normalization, ReLU activation function and a 3×3 convolution.

The concatenation of the input and the architecture of these dense blocks leads to fewer parameters compared to traditional CNNs, as each layer contributes with small information to a global state instead

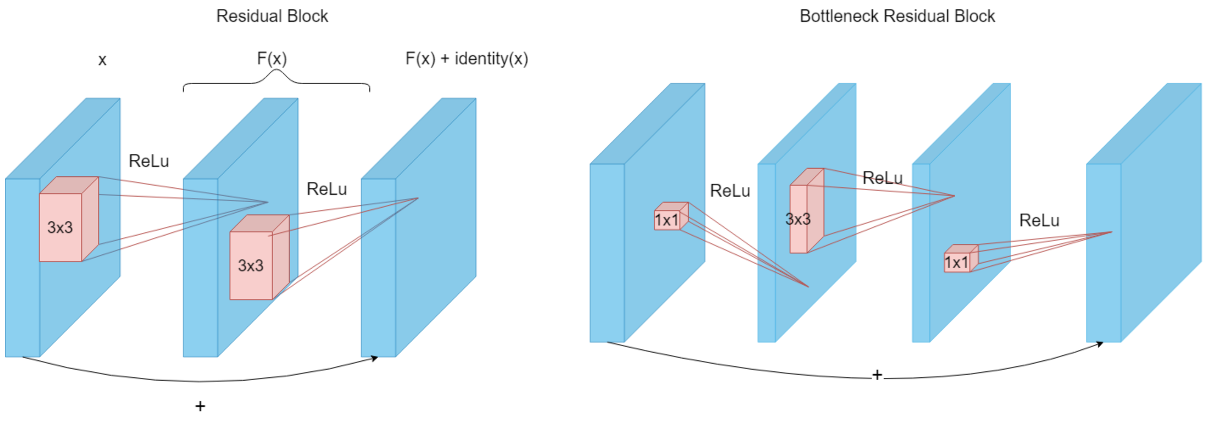


Figure 2.3: Residual blocks and bottleneck residual blocks used in ResNets.

of passing the state from a layer to another. The growth rate of the network is given by a new hyperparameter k which represents the number of feature maps that a layer produces; if a function H_l produces k feature maps, it follows that the l^{th} layer has $k_0 + k * (l - 1)$ input feature maps, where k_0 is the number of channels in the input layer. Between the dense blocks, there are transition blocks designed to reduce the feature maps by the use of bottleneck layers and compression layers. The bottleneck layer consists of a 1×1 convolution before each 3×3 convolution and the compression layer reduces the feature maps produced by a compression factor resulting in $\lfloor \theta m \rfloor$ feature maps, where $0 < \theta < 1$.

2.4 Recurrent Neural Networks

Sequential data, like time series or text, presents its own difficulties for processing with neural networks. Depending on the sequence problem to be solved, the input and/or the output can have varying lengths for each data record, e.g. many-to-one or many-to-many sequence analysis task.

A regular feedforward network has a fixed number of input neurons and so a sequence would have to be divided into groups. However, the length of each group would require a good knowledge of the domain problem, as a naïve division would miss trends or patterns that are relevant for a prediction. Another important aspect to consider in text data is the relation and order of words that cannot be addressed by a feedforward network. A Recurrent Neural Network (RNN) is a type of neural network that can process sequential data by adding loops in the neuron's connections. This allows the network to store a state containing information about the data that has been processed up to a given moment.

A RNN can be seen as an application of two functions. The first takes as input the input vector $\mathbf{x_t}$ and hidden state $\mathbf{h_{t-1}}$ producing a new state vector $\mathbf{h_t}$. Then, the current hidden state $\mathbf{h_t}$ is mapped to an output y_t. Considering this, together with parameters θ and an activation function $g()$, a possible RNN architecture can be formally described as:

$$\begin{aligned} \mathbf{h_t} &= g(\mathbf{W_{hh}} \cdot \mathbf{h_{t-1}} + \mathbf{W_{xh}} \cdot \mathbf{x_t} + \mathbf{b})\,, \\ \mathbf{y_t} &= \mathbf{W_{hy}} \cdot \mathbf{h_t} + \mathbf{c}\,. \end{aligned} \tag{2.7}$$

Because of the existence of the loops in these networks, the backpropagation algorithm has to be adapted. Backpropagation through time (BPTT) is backpropagation applied to the unrolled graph of the RNN so the error with respect to the loss is backpropagated throughout the network as if it was a feedforward network. An unrolled RNN is graphically depicted in Figure 2.4. However, this method suffers from vanishing and exploding gradients, in which the weights will either decay exponentially to zero, or grow exponentially fast, making it impossible for the network to capture long-term dependencies.

A type of recurrent neural network named Long Short-Term Memory (LSTM) was developed to address the aforementioned limitations of a regular RNN and is described in the next section.

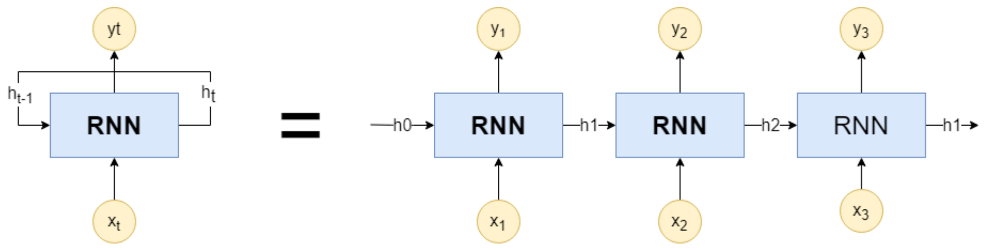


Figure 2.4: A simple RNN and the corresponding unrolled representation.

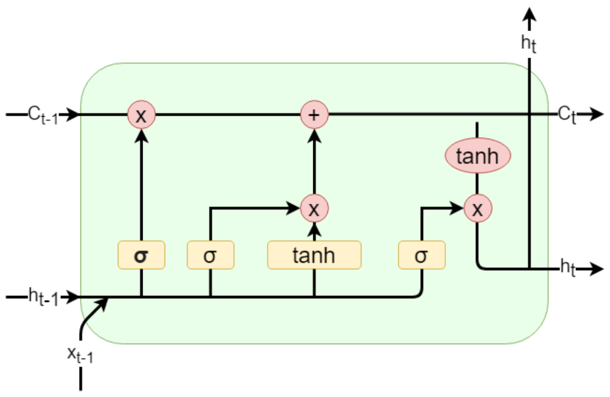


Figure 2.5: Representation of an LSTM cell with its gates.

Long Short-Term Memory Networks

Long Short-Term Memory (LSTM) networks were developed by Hochreiter and Schmidhuber (1997) to address the problems of unstable gradients and to capture long term dependencies. LSTM have internal mechanisms named gates that control the information to be stored or discarded. The cell state $\mathbf{C_t}$ is propagated over the network, acting like the memory of the network. Throughout the network, gates control which information is relevant or should be dropped during training. These gates are the forget gate, input gate and output gate.

In the beginning of each time step in the LSTM, the last hidden state $\mathbf{h_{t-1}}$ and the input $\mathbf{x_t}$ are concatenated. In the forget gate, it is decided which information of the cell state $\mathbf{C}_t$ is to be discarded or to be kept. The concatenation of the last hidden state and the input is fed to a sigmoid function which outputs values between 0 and 1, resulting in the forget vector $\mathbf{f}_t$. A value closer to 0 represents that information is going to be discarded and closer to 1 represents that it is to be kept.

The input gate decides what information is to be added to the cell state. In the sigmoid layer, the scaling values are commputed resulting in the input vector $\mathbf{i}_t$ while an hyperbolic tangent function creates a vector of candidate values $\tilde{\mathbf{C}}_t$ to be added to the state. Next, the candidate values are scaled by the input vector and then alongside the forget vector are used to update the cell state $\mathbf{C_{t-1}}$.

Finally, in the output gate, the new hidden state is computed. The newly-created cell state $\mathbf{C}_t$ is normalized by an hyperbolic tangent function and filtered by vector $\mathbf{o}_t$ from another sigmoid layer, resulting in the hidden state $\mathbf{h}_t$. In the end, both the cell state $\mathbf{C_t}$ and hidden state $\mathbf{h}_t$ are sent to the next timestep. The following equations formally represent the gate processing within LSTMs, where $[\mathbf{h_{t-1}}, \mathbf{x_t}]$ represents the concatenation of vectors $\mathbf{h_{t-1}}$ and $\mathbf{x_t}$, and $\odot$ represents the point-wise multiplication of two vectors (Hadamard product):

$$\begin{aligned}
\mathbf{f_t} &= \sigma\left(\mathbf{W_f} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_f}\right) , \\
\mathbf{i_t} &= \sigma\left(\mathbf{W_i} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_i}\right) , \\
\tilde{\mathbf{C}}_\mathbf{t} &= \tanh\left(\mathbf{W_C} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_c}\right) , \\
\mathbf{C_t} &= \mathbf{f_t} \odot \mathbf{C_{t-1}} + \mathbf{i_t} \odot \tilde{\mathbf{C}}_\mathbf{t} , \\
\mathbf{o_t} &= \sigma\left(\mathbf{W_o} \cdot [\mathbf{h_{t-1}}, \mathbf{x_t}] + \mathbf{b_o}\right) , \\
\mathbf{h_t} &= \mathbf{o_t} \odot \tanh\left(\mathbf{C_t}\right) .
\end{aligned} \tag{2.8}$$

Besides sentence classification or generation, LSTMs have been applied in several other ways: ELMo, by Peters et al. (2018), produces contextual embeddings for words. Regular word embeddings like Word2Vec by Mikolov et al. (2013) represent words as vectors and are able to capture semantic as well as syntathic relationships. Word embeddings can be compared with the cosine similarity, and used as inputs for neural models. However, Word2Vec embeddings are not able to capture polisemy, when the same word has different meanings depending on the context. ELMo produces contextual word embeddings by using bi-diectional LSTMs, with a forward and backward passes allowing for the model to read the entire sentence and gather its context. ELMo is trained for language modeling, which is the task of predicting the next word given a sequence of words. This task allows the model to train in large datasets in a semi-supervised learning, without the need of labels.

LSTMs have also been applied in encoder-decoder architectures. The sequence to sequence (seq2seq) model proposed by Sutskever et al. (2014) uses LSTM networks for both an encoder and decoder. The model takes a sequence of data and outputs another sequence, which is very suitable for machine translation task. Firstly, the enconder processes the input and creates a context vector, which is passed to the decoder that generates the final output sequence, one element at a time. The decoder takes the previous step's output as input when computing the current output, in an auto-regressive manner.

2.5 Transformers

Sequence to sequence models based on encoder and decoder architectures, using recurrent or convolutional neural networks, are limited by sequential operations (e.g. the computation of a RNN hidden state is dependent on the result of the previous hidden state). The Transformer architecture was proposed by Vaswani et al. (2017) and is based only on attention mechanisms, which also allows for parallelization making the training process much faster. Attention mechanisms are able to model dependencies independently of their distance in the input or output sequences.

The Transformer follows a encoder-decoder architecture. The encoder is composed by a stack of layers (originally the authors used $N = 6$) and each layer is composed by two sub-layers: a self-attention layer and a feed-forward neural network. A residual connection is applied around each of the two sub-layers, followed by normalization. The decoder is also composed by a stack of layers, in this case comprised by three sub-layers. Besides the two sub layers of the encoder, there is a third layer that performs attention over the output of the encoder stack. A residual connection is also applied around each sub-layer followed by normalization. The self-attention layer of the decoder is modified by the application of masking to prevent the decoder to take into consideration positions of the input following the position being processed.

Self-attention is an attention mechanism that allows the model to capture relationships between different parts of the input sequence. The attention is computed as a weighted sum of values, and the weights are computed by a function of a query and key. The authors named their attention method as scaled dot-product attention. To compute the attention, the input is comprised of the queries and keys, of dimensionality d_k, and values of dimensionality $\mathbf{d_v}$. To compute the weights, a dot product is applied between the query and all the keys, and then scaled by $\sqrt{d_k}$ followed by a softmax function. This scaling is applied to prevent the dot product to grow large in magnitude for large values of d_k, which would hinder the training process. The queries, keys, and values are packed in matrices $\mathbf{Q}$, $\mathbf{K}$ and $\mathbf{V}$. Formally, the matrix computation of attention is given by:

$$\mathrm{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \mathrm{softmax}\left(\frac{\mathbf{QK^T}}{\sqrt{d_k}}\right)\mathbf{V}\,. \tag{2.9}$$

The attention computation process is not done by just a single set of of query/key/value matrices but by different independent h sets, in a process called multi-head attention. Each of the sets is randomly initialized and, after training, projects the input into a different representation subspace. This training process of the different sets can be parallelized. Finally, the resulting projections of each attention head is concatenated and once again projected by $\mathbf{W}^O$, resulting in the final values.

$$\begin{aligned} \mathrm{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) &= \mathrm{Concat}(\mathrm{head}_1, ..., \mathrm{head}_h)\mathbf{W}^O\,, \\ \mathrm{head}_i &= \mathrm{Attention}(\mathbf{QW}_i^Q, \mathbf{KW}_i^Q, \mathbf{VW}_i^Q)\,. \end{aligned} \tag{2.10}$$

To make use of the order of the sequence, a positional encoding is added to the input token embeddings at the bottom of the encoder and decoder stacks. The authors used the sine and cosine functions

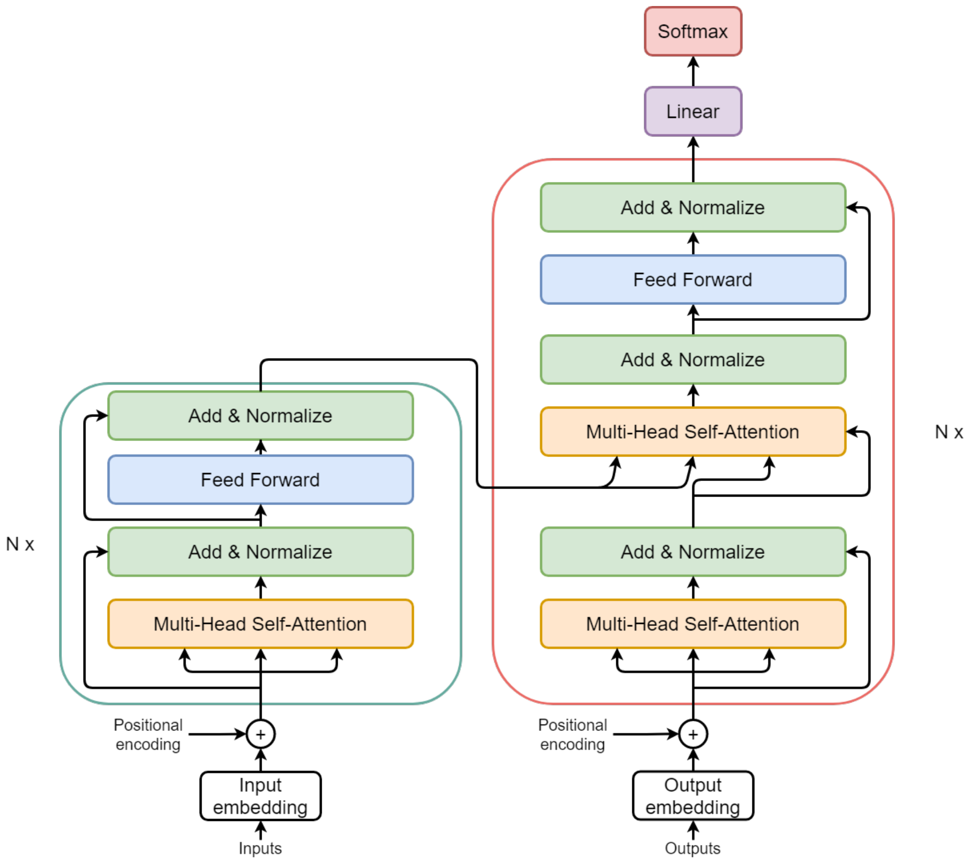


Figure 2.6: Transformer model architecture, with the encoder on the left and decoder on the right.

for positional encodings, where pos is the token position and i is the embedding dimension:

$$\begin{aligned} PE_{pos,2i} &= \sin(pos/10000^{2i/d_{model}})\,, \\ PE_{pos,2i+1} &= \cos(pos/10000^{2i/d_{model}})\,. \end{aligned} \tag{2.11}$$

The authors hypothesized that this positional encoding would allow the model to attend to relative positions and to extrapolate to sequence lengths longer than the ones encountered during training. The final output of the decoder is then fed into a fully connected layer that projects the vector returned by the decoder into a vector of the size of the vocabulary. This vector is then fed to a softmax function that returns the probability of each word.

Bidirectional Encoder Representations from Transformers

Unidirectional language models limit the choice of possible architectures to be implemented, because language models must not perceive the whole tokens of the sequence and consequently make trivial predictions. Approaches with such restrictions are sub-optimal for sentence-level tasks, such as question answering. ELMo improved the performance on these tasks, being able to capture context in word embeddings, but it still consisted of a shallow concatenation of independently trained left-to-right and right-to-left bidirectional LSTMs.

Bidirectional Encoder Representations from Transformers (BERT) developed by Devlin et al. (2019) uses an architecture based on Transformer encoders and creates deep bidirectional representations. The implementation of the model is done using semi-supevised pre-training on unlabelled data, and the model can then use supervised fine-tuning over a target task. The pre-training tasks are Masked Language Modeling (MLM) and Next Sentence Prediction (NSP). MLM consists on randomly masking a percentage of the input tokens, and training the model to predict those masked tokens. The model chooses 15% of the tokens, and these are replaced 80% of the times by the token `[MASK]`; 10% of the times by a random token; and 10% of the times the token remains unaltered. The NSP task allows the model to grasp understanding on the relationship between two sequences, which is important for downstream tasks such as question answering. In this case, the model is then trained to predict whether a sequence follows another: 50% of the times a sequence B follows a sequence A, and 50% of the times B is a random sentence.

The architecture of the model consists of a stack of Transformer encoders. The authors have tested two model sizes, namely BERT base (12 encoder layers) and BERT large (24 encoder layers). The feedforward networks are comprised of 768 and 1024 hidden units, respectively for each network size, and either 12 or 16 attention heads are used, respectively in each case.

A sentence is tokenized in the following way: a special classification token `[CLS]` is inserted in the first position, and sentence pairs are split into word pieces, packed and separated by a `[SEP]` token. Each word piece representation consists of the sum of three embeddings: a word; an embedding indicating whether it belongs to sentence `A` or `B`, to be used for the NSP task; and a positional embedding. The final output of the `[CLS]` token is C, which is used for classification tasks. Also, each input token i has a final output vector T_i.

BERT is easily finetuned to downstream tasks as the same architecture can be used in both pre-training and fine-tuning, apart from the output layers. This reduces the need for heavily-engineered task specific architectures. BERT can also unify the encoding of text pairs by the use of the self-attention mechanism. At the input, sentence `A` and sentence `B` from pre-training are analogous to text pairs used in tasks such as question answering.

The total number of parameters in BERT base is 110M, and for BERT large is 340M. This large number of parameters can constrain the application of these models due to the memory and time requirements. Some methods have been applied to reduce the size of the models while maintaining their high accuracy. For instance, quantization (Jacob et al., 2017) consists of approximating the weights of a network with smaller precision by encoding the weights with fewer bits; weights pruning (Han et al., 2015) consists on removing connections from the network.

Differently from the previous techniques, knowledge distillation involves training a small model (the student) to reproduce the behavior of a big and slow model (the teacher). The method was introduced by Bucilua et al. (2006) and generalized by Hinton et al. (2015). In supervised learning, a classification model is trained to predict the class of an instance by maximizing its probability. A model having a good performance will predict the correct class with high probability, and it will assign near-zero probabilities to the other classes. Still, the information about these near-zero probabilities is important, as it reflects the generalization capabilities of the model e.g. the next most likely classes should be similar to the predicted class. Thus, in teacher-student training, the student should mimic the full output distribution of the teacher.

Sanh et al. (2020) developed a student model called DistilBERT, consisting of a smaller version of BERT that achieves 97% of BERT's performance on 2 diverse set of tasks, while having about half of the parameters of BERT base and being 60% faster at inference time. DistilBERT has the same general architecture, but had the token-type embeddings and the pooler (used for the next sentence classification task) removed. The number of layers was also reduced by 2. The distillation loss used in DistilBERT consists of training the student with a cross-entropy over the soft targets (probabilities of the teacher for each class) instead of the hard targets (true labels):

$$L = -\sum_i t_i \times \log(s_i)\,. \tag{2.12}$$

In the previous equation, s_i and t_i are the probabilities estimated by the student and teacher, respectively. The complete training loss for DistilBERT is a linear combination of the distillation loss and the masked language modeling loss of the original BERT.

2.6 Question Anwering

Reading Comprehension, or the ability to read text and then answer questions about it, is a challenging task for machines, requiring both understanding of natural language and knowledge about the world. Question Answering can be seen as a restricted problem of Visual Question Answering, in which the models only attend the question. In this section the capabilities of approaches leveraging Transformer architectures for tackling Question Answering are described.

The Stanford Question Answering Dataset (SQuAD) (Rajpurkar et al., 2016) is a reading comprehension dataset for the evaluation of question answering models. The dataset consists of questions posed by crowdworkers on a set of Wikipedia articles, where the answer to every question is a segment of text, or span, from the corresponding reading passage. Models are expected to predict the beginning and ending in the span of the passage that answers the question. The SQuAD 2.0 (Rajpurkar et al., 2018) version extends the task to include the possibility that no answer exists in the paragraph.

Most of the top places occupied in the SQuAD leaderboard[1] use approaches based on BERT. When developing BERT, the authors also tested the model on SQuAD (Devlin et al., 2019). To extract the span, two parameters are trained, namely a start vector $\mathbf{S}$ and an end vector $\mathbf{E}$. The probability of word i being the start of the answer span is computed as a dot product between the final hidden vector for the i^{th} token $\mathbf{T}_i$ and start vector $\mathbf{S}$, followed by a softmax over all of the words in the paragraph:

$$P_i = \frac{e^{\mathbf{S}\cdot\mathbf{T}_i}}{\sum_j e^{\mathbf{S}\cdot\mathbf{T}_j}}. \tag{2.13}$$

The end of the answer span is similarly calculated. The final score of a candidate span from position i to position j is defined as $s = \mathbf{S}\cdot\mathbf{T}_i + \mathbf{E}\cdot\mathbf{T}_j$, and the candidate with the maximum scoring is used as the prediction. Questions that do not have an answer in the paragraph have the start and end vector at the position of the `[CLS]` token where the candidate score of a null answer is similarly calculated as $s_{\text{null}} = \mathbf{S}\cdot\mathbf{C} + \mathbf{E}\cdot\mathbf{C}$, where $\mathbf{C}$ is the output of the `[CLS]` token. The best non-null answer with the highest score is chosen if it surpasses a threshold over the null answer.

One of the BERT extensions that is widely adopted in the SQuAD task is ALBERT by Lan et al. (2020). ALBERT introduces two parameter reduction techniques that are able to achieve both lower memory requirements and speeding up the training process. The first parameter reduction technique is to decompose the large vocabulary embedding matrix into two small matrices, and thus separate the size of the hidden layer from the size of the vocabulary embedding. This separation makes it easier to grow the hidden size without significantly increasing the parameter size of the vocabulary embedding. The second memory reduction technique is to share all parameters, including feed-forward network and attention parameters, across all layers. Finally, the authors also replaced the NSP objective with Sentence Order Prediction (SOP), which consists on predicting the ordering of two consecutive segments of text. The authors argue that while NSP was designed to tackle coherence prediction, it also combines a topic prediction task which overlaps what is already being learned by MLM and thus the knowledge gathered by NSP is reduced. SOP forces the model to learn finer-grained distinctions about discourse-level coherence properties and removed the topic prediction component from this task. As a result of these approaches, ALBERT is able to achieve state-of-the-art results in downstreams tasks like Question Answering (+3.1% compared to BERT on SQuAD v2.0) despite being based on a Transformer architecture with fewer parameters. The authors used the same span extraction method as BERT for SQuAD v1 and for SQuAD v2.0 an additional classifier is used beforehand for predicting answerability.

[1]https://rajpurkar.github.io/SQuAD-explorer/

2.7 Overview

This chapter introduced different concepts which are be the basis of the approach proposed in this work, and for the different studies discussed in the following chapter. Different types of neural networks were discussed. In particular, the chapter discussed the use of convolutional neural networks as image encoders and the use of Transformers with the self-attention module.

3

Related Work

Contents

3.1 Visual Question Answering Methods

Visual Question Answering (VQA) is a computer vision task where a system is given a text question about an image, and an answer should be predicted. VQA encompasses many sub-problems in computer vision such as object recognition, activity recognition and scene classification. However, the aforementioned computer vision tasks are limited in scope and generality compared to VQA, where a system is often required to reason about the relationships between objects and the overall scene.

State-of-the-art approaches for VQA are based on deep neural networks (Srivastava et al., 2020), where an image encoder component corresponding to a pre-trained convolutional neural network generates a representation for the image, whereas a text encoder corresponding to a recurrent neural network generates a representation for the question. The image and question features can then be fused through various mechanisms. Answers can be provided using a decoder component that generates the answer word-by-word, although most previous studies treat VQA as a classification problem where each answer is treated as a distinct label. This leads to systems only being able to generate answers that are seen during training, which might be a limitation for some domains. Because the VQA-MED task of ImageCLEF can be tackled as a classification problem as well (Yan et al., 2019), the following studies use this approach rather than considering a task of text generation.

Agrawal et al. (2016) developed The VQA Dataset consisting mostly of images from the Microsoft Common Objects in Context (COCO) dataset (Lin et al., 2015). The authors provided a basic baseline and in their approach treated the task in a classification manner, choosing the top $K = 1000$ most frequent answers as classification labels, which covers 82.67% of the train+val possible answers. The image embedding is done using the ℓ_2 normalized activations from the last layer of VGGNet, producing 4096-dim image embedding. The questions are encoded using an LSTM with two hidden layers producing a 2048-dim embedding that is the concatenation of the the cell and the hidden states (each being 512-dim) from both layers. Both image and question features are projected to a 1024-dim space with non-linearity and a element-wise multiplication of the two vectors is performed that is then fed to a MLP. Finally, the answer is retrieved by a softmax distribution over the K answers.

The previous approach used global features of the input, which may introduce noisy information irrelevant to the given question. To address this, many studies have introduced attention mechanisms into VQA. The use of attention may significantly improve performance by creating region specific features. Most approaches only consider image attention, although a few explore incorporating attention in text (Srivastava et al., 2020). This way the models create region-specific features based on the idea that certain visual features and certain words in a question are more informative than the others.

The Focused Dynamic Attentiom Method developed by Ilievski et al. (2016) uses Edge Boxes to generate bounding boxes for region proposals, thus creating local features. Edge Boxes are based on the observation that the number of contours that are wholly contained in a bounding box is indicative of the likelihood of the box containing an object (Zitnick and Dollár, 2014). The question is encoded by an LSTM and image representations are given by a ResNet. Features are extracted for each proposed region (local features) and also the whole image (global features). Subsequently, Word2Vec is used to retrieve the similarity of the question words and all the object labels given by the ResNet from each region proposal. If a similarity score is greater than 0.5, the region is selected. Following the question word order, the retrieved features from the ResNet of each selected region are fed to an LSTM at each time-step. At the last time-step, the global features are also fed to the LSTM resulting in the final image embeddings. Thus, local and global features are combined with the use of a LSTM to create a joint representation. Finally, an element-wise multiplication is performed between the final image and question embeddings, which is then fed to a non-linear MLP with a softmax classifier in the end, to predict the answer. At the time it was submited, the model achieved state-of-the-art accuracy on the The VQA Dataset (Agrawal et al., 2016).

Yang et al. (2016) proposed Stacked Attention Networks (SANs) that use a semantic representation of a question as query to search for the regions in an image that are related to the answer. The authors argue that VQA requires multiple steps of reasoning so the image is queried multiple times to infer the answer progressively. VGGNet is used as image encoder, and the features from the last pooling layer are taken. Then, the features from each region are projected with a non-linearity to a vector with the same dimensionality d as the question vector $\mathbf{v}_Q$ retrieved from a LSTM.

The multi-step reasoning of SAN can be described as follows: given the image feature matrix $\mathbf{v}_I$ and the question vector $\mathbf{v}_Q$, the features are fed to a single layer neural network and then a softmax function generates the attention distribution over the regions of an image. This process is formally described as:

$$\begin{aligned} \mathbf{h}_A &= \tanh(\mathbf{W}_{I,A}\mathbf{v}_I \oplus (\mathbf{W}_{Q,A}\mathbf{v}_Q + \mathbf{b}_A))\,, \\ \mathbf{p}_I &= \mathrm{softmax}(\mathbf{W}_p\mathbf{h}_A + \mathbf{b}_P)\,. \end{aligned} \tag{3.1}$$

As previously noted, $\mathbf{v}_Q \in \mathbb{R}^d$ and $\mathbf{v}_I \in \mathbb{R}^{d\times m}$ where d is the image and question dimension and m is the number of image regions. Supposing $\mathbf{W}_{I,A}, \mathbf{W}_{Q,A} \in \mathbb{R}^{k\times d}$ it results that $\mathbf{W}_{I,A}\mathbf{v}_I \in \mathbb{R}^{k\times m}$ is a matrix and that $\mathbf{W}_{Q,A}\mathbf{v}_Q, \mathbf{b}_A \in \mathbb{R}^k$ is a vector. The symbol $\oplus$ corresponds to the addition of a matrix and a vector and this addition is performed by adding each column of the matrix by the vector. The following equation describes the calculation of $\tilde{\mathbf{v}}_I$ based on the attention distribution, by a weighted sum of the image vectors, each from region i. Subsequently, $\tilde{\mathbf{v}}_I$ is then combined with question vector $\mathbf{v}_Q$ to form a refined query vector $\mathbf{u}$:

$$\begin{aligned} \tilde{\mathbf{v}}_I &= \sum_i \mathbf{p}_i\mathbf{v}_i\,, \\ \mathbf{u} &= \tilde{\mathbf{v}}_I + \mathbf{v}_Q\,. \end{aligned} \tag{3.2}$$

Models using attention can construct a more informative $\mathbf{u}$, since higher weights are put on the visual regions that are more relevant to the question. The authors argue that for more complicated questions, more passes of the attention layer are needed. Therefore, the above query-attention process is iterated using multiple attention layers. For the k-th attention layer, the computation is as follows:

$$\begin{aligned} \mathbf{h}_A^k &= \tanh(\mathbf{W}_{I,A}^k \mathbf{v}_I \oplus (\mathbf{W}_{Q,A}^k \mathbf{u}^{k-1} + \mathbf{b}_A^k))\,, \\ \mathbf{p}_I^k &= \mathrm{softmax}(\mathbf{W}_p^k \mathbf{h}_A^k + \mathbf{b}_P^k)\,. \end{aligned} \tag{3.3}$$

Vector $\mathbf{u}^0$ is initialized as $\mathbf{v}_Q$. The refined query vector $\mathbf{u}^k$ is computed at every timestep, similarly to Equation 3.2. The process is repeated K times and then the final $\mathbf{u}^K$ is used to infer the answer:

$$\mathbf{p}_{ans} = \mathrm{softmax}(\mathbf{W}_u \mathbf{u}^K + \mathbf{b}_u)\,. \tag{3.4}$$

Anderson et al. (2018) achieved the first place in the 2017 Challenge on The VQA Dataset (Agrawal et al., 2016) by proposing a method combining bottom-up and top-down attention. The bottom-up attention is implemented with the use of a Faster R-CNN (Ren et al., 2016) to localize objects with bounding boxes and classify them. For each selected region i, $\mathbf{v}_i$ is the mean-pooled convolutional feature from this region with 2048 dimensions. Each question is encoded as the hidden state $\mathbf{q}$ of a gated recurrent unit (GRU). The authors propose a non-linear transformation through gated hyperbolic tangent activations which can be defined as follows:

$$\begin{aligned} \tilde{\mathbf{y}} &= \tanh(\mathbf{W}\mathbf{x} + \mathbf{b})\,, \\ \mathbf{g} &= \sigma(\mathbf{W}'\mathbf{x} + \mathbf{b}')\,, \\ \mathbf{y} &= \tilde{\mathbf{y}} \odot \mathbf{g}\,. \end{aligned} \tag{3.5}$$

The vector $\mathbf{g}$ acts multiplicatively as a gate on the intermediate activation $\tilde{\mathbf{y}}$. This formulation is inspired by similar gating operations within recurrent units such as LSTMs and GRUs.

To compute a top-down attention mechanism, for each location i in the image, the vector $\mathbf{v}_i$ is concatenated with the question embedding $\mathbf{q}$, represented as $[\mathbf{v}_i, \mathbf{q}]$. This concatenation is passed by the gated non-linear activation function f_a and a linear layer, to obtain a scalar attention weigth $\alpha_{i,t}$ associated with that location. The attention weights are normalized over all locations with a softmax function. The image features from all locations are then weighted by the normalized values and summed to obtain a 2048 dimension vector $\hat{\mathbf{v}}$. This process can be formalized as follows:

$$\begin{aligned} a_i &= \mathbf{W}_a f_a([\mathbf{v}_i, \mathbf{q}])\,, \\ \boldsymbol{\alpha} &= \mathrm{softmax}(\mathbf{a})\,, \\ \hat{\mathbf{v}} &= \sum_{i=1}^{K} \alpha_i \mathbf{v}_i\,. \end{aligned} \tag{3.6}$$

Subsequently, the representations of the question and of the image are each passed through the gated non-linear activation function and combined with element-wise multiplication, producing a vector $\mathbf{h}$ of the joint embedding:

$$\mathbf{h} = f_q(\mathbf{q}) \odot f_v(\hat{\mathbf{v}})\,. \tag{3.7}$$

The joint representation $\mathbf{h}$ is fed through another gated non-linear layer and then through a linear transformation $\mathbf{W}_o$ to predict the score for each of the candidate answers:

$$\hat{s} = \sigma(\mathbf{W}_o f_o(\mathbf{h})). \tag{3.8}$$

3.2 Visual Question Answering Methods in VQA-MED ImageCLEF

Yan et al. (2019) achieved the first place in the VQA-MED task of ImageCLEF 2019 (Ben Abacha et al., 2019). The proposed approach consisted on four modules: image feature extraction, question semantic encoder, feature fusing with co-attention mechanisms, and answer prediction. This answer is produced by selecting from a set of possibilities i.e. through a classification objective.

The image feature extraction model is a VGG16 network pretrained on ImageNet. The fully-connected layers in the VGG16 network are removed and the convolution outputs corresponding to all the feature maps are concatenated after a global average pooling, forming a 1984 dimensional vector that represents the image. The authors showed that this method using global average pooling avoids over-fitting on the small dataset. The question encoder is a BERT base model that is used as feature extractor. To represent each sentence, the last and penultimate layers are averaged to obtain a 768 dimension question vector.

To combine the visual and question features, authors use Multi-modal Factorized Bilinear pooling (MFB) method by Yu et al. (2017), combining with co-attention learning. MFB pooling consists on two stages: first, in the expansion stage, a projection of the inputs is followed by an element-wise multiplication of the two vectors. To prevent over-fitting, a dropout layer is added after the element-wise multiplication layer. Next, in the squeeze stage the result of the previous stage is fed to a sum pooling function, which consists of a regular mean pooling but without the scaling. Because the magnitude of

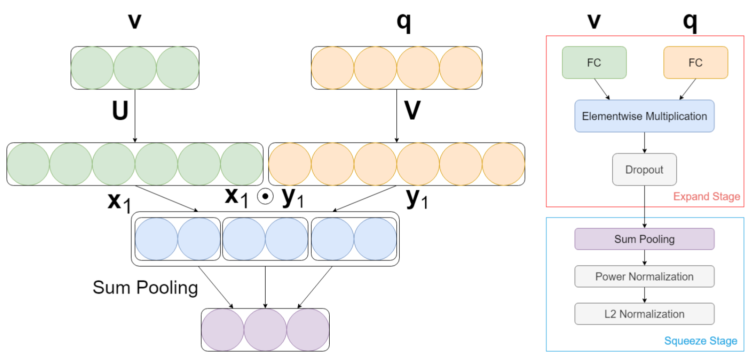


Figure 3.1: The flowchart of multi-modal factorized bilinear pooling method, together with the complete design of the MFB module.

the output neurons may vary dramatically due to the element-wise multiplication causing the model to converge to a suboptimal local minimum, power normalization ($\mathbf{z} \leftarrow sign(\mathbf{z})|\mathbf{z}|^{0.5}$) and l_2 normalization ($\mathbf{z} \leftarrow \mathbf{z}/\|\mathbf{z}\|$) procedures are performed after the MFB output. A graphical representation of the MFB process is shown in Figure 3.1 and a formal description follows, where $\mathrm{SumPooling}(\mathbf{x}, k)$ is a function using a one-dimensional non-overlapped window with the size k to perform sum pooling over $\mathbf{x}$, and where the involved parameters are $\mathbf{U} \in \mathbb{R}^{m \times k}$, $\mathbf{V} \in \mathbb{R}^{n \times k}$, the image features $\mathbf{v} \in \mathbb{R}^{m}$ and the question features $\mathbf{q} \in \mathbb{R}^{n}$:

$$z = \mathrm{SumPooling}(\mathbf{U}^T\mathbf{v} \odot \mathbf{V}^T\mathbf{q}, k)\,. \tag{3.9}$$

The authors use the co-attention mechanism of the original MFB pooling, consisting of both question attention and image attention, as graphically depicted in Figure 3.2. The question attention learns the attention weights as the contribution of each word. In image attention, the image is represented through a 14×14 spatial grid and each image grid cell is merged with the question features using MFB, followed by feature transformations (i.e., using a 1×1 convolution and a ReLU activation). Finally, softmax normalization to predict the attention weight for each image grid. For each image grid, a weighted sum is performed based on the attention weights. The results of each attention grid feature are concatenated to produce the final attentional image features. Finally, the attentional image features are merged with the question features, using MFB for the answer prediction.

Hoang Minh et al. (2019) instead used an ensemble approach with a variation of multiple VQA components. The authors used a ResNet-152 model, pretrained on ImageNet dataset, to extract visual features. Regarding question features extraction, the authors use Skip-Thought vectors (Kiros et al., 2015) and different BERT implementations within the `huggingface` Transformers library (Wolf et al., 2020). The skip-thought method extracts sentence features according to the semantic and syntactic properties of the sentence. It is based on an encoder-decoder model that was trained to reconstruct the previous s_{i-1} and next sentences s_{i+1} of a middle sentence s_i. Both the encoder and the decoder are based on RNNs and the embedding of the sentence s_i is fed to the decoder to generate a new sentence word-by-word. The loss function is the sum of the log probabilities of the surrounding sentence words.

Regading the attention mechanism, the authors use Multi-modal Low-rank Bilinear Pooling (MLB) (Kim et al., 2017) and Multimodal Tucker Fusion for Visual Question Answering (MUTAN) (Ben-younes et al., 2017), also introducing a novel bilinear transformation. The final ensemble contains a total 26 models. A description of these mechanisms follows.

MLB consists on the element-wise product of the two feature vectors in the common space with two low-rank projection matrices. A formal description is given next where the image features correspond to $\mathbf{v} \in \mathbb{R}^m$, the question features correspond to $\mathbf{q} \in \mathbb{R}^n$,$\mathbf{U} \in \mathbb{R}^{m\times k}$, and $\mathbf{V} \in \mathbb{R}^{n\times k}$:

$$\mathbf{z} = \mathrm{MLB}(x, y) = (\mathbf{U}^T\mathbf{v}) \odot (\mathbf{V}^T\mathbf{q})\,. \tag{3.10}$$

Regarding MUTAN, this fusion scheme is modeled by a tensor τ that combines the image and question features. Given the question features $\mathbf{q}$ and image feature $\mathbf{v}$, a vector $\mathbf{z}$ can be produced as:

$$\mathbf{z} = (\tau \times_1 \mathbf{q}) \times_2 \mathbf{v}\,. \tag{3.11}$$

The full tensor is $\tau \in \mathbb{R}^{d_\mathbf{q}\times d_\mathbf{v}\times|\mathcal{A}|}$, and the operator $\times_n$ represents the *mode-n* product between a tensor and a vector. A tensor can be seen as a collection of vectors, named the fibers. The *mode-n* product consists on mapping the tensor into *mode-n* fibers and compute all inner products of the *mode-n* fibers

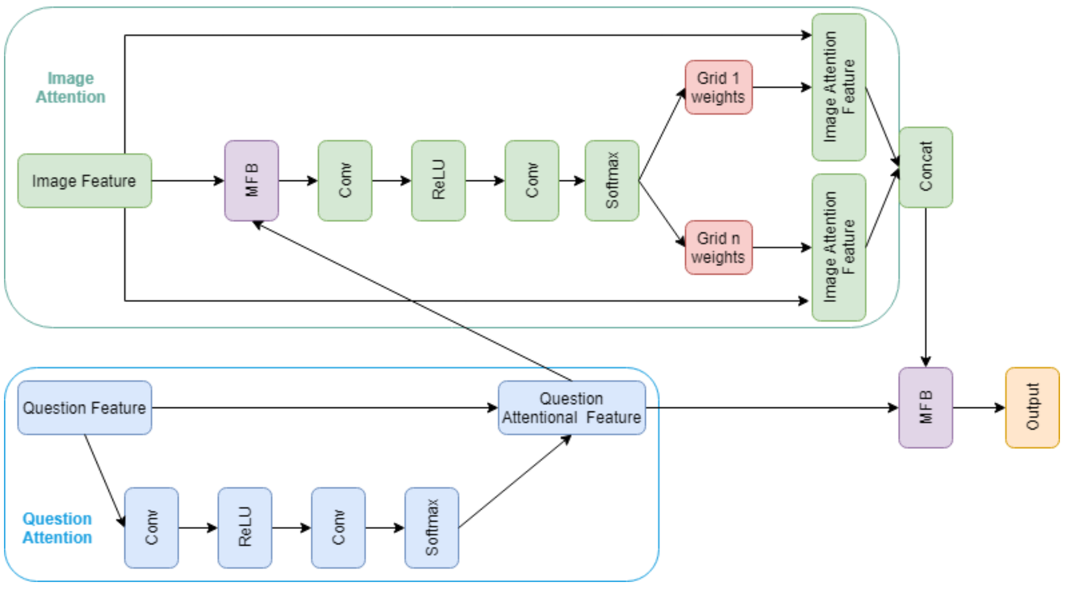


Figure 3.2: MFB with Co-Attention network architecture for VQA.

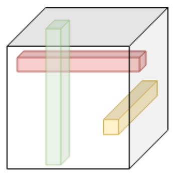


Figure 3.3: Mode-n fibers of a 3D-tensor: *mode*-1 in green, *mode*-2 in red, *mode*-3 in yellow.

with the vector. Considering a 3D-cube representing a tensor, *mode*-1 fibers divide the cube into multiple vectors in the bottom-up direction. *mode*-2 fibers divide the cube in the left-right direction, and*mode*-3 fibers divide the cube in the backwards-forwards direction.

The computation of the fusion scheme in aforementioned manner would quickly become intractable due to the large number of free parameters of τ. For common dimensions of textual, visual and answer spaces $d_{\mathbf{v}} \approx d_{\mathbf{q}} \approx 2048$, and $|\mathcal{A}| \approx 2000$, the total number of parameters in τ is $\sim 10^{10}$. To cover this, the MUTAN method relies on Tucker tensor-based decomposition, resulting in a factorization into three factor matrices $\mathbf{W_q}$, $\mathbf{W_v}$ and $\mathbf{W_o}$, and a core tensor $\tau_{\mathbf{c}}$:

$$\tau = ((\tau_{\mathbf{c}} \times_1 \mathbf{W_q}) \times_2 \mathbf{v_x}) \times_3 \mathbf{W_o} \,. \tag{3.12}$$

Considering the image features $\mathbf{v}$ and question features $\mathbf{q}$ and also the Tucker decomposition method, Equation 3.11 can be rewritten as:

$$\tau = ((\tau_{\mathbf{c}} \times_1 (\mathbf{q^T W_q})) \times_2 (\mathbf{v}^T \mathbf{W_v})) \times_3 \mathbf{W_o} \,. \tag{3.13}$$

The first place of the VQA-MED task in ImageCLEF2020 (Ben Abacha et al., 2020) was achieved by Liao et al. (2020). Due to the inherent structure of the questions on the task, the authors reduced the VQA task to a multi label image classification problem. The authors used Skeleton-based Sentence Mapping (SSM) to summarize questions with similar sentence structures into unified backbones, determining the question categories and infering the corresponding labels. Word-level edit distance (i.e., the Levenshtein distance) was applied to pairs of questions, and questions were then grouped according to the similarity. For each group, a backbone can be used to generate those questions. The authors found a total of 68 possible question backbones. A possible backbone is: `‘‘is ${this_alts} ${ct_alts} ${normal_alts}’’`, where `this_alts`, `ct_alts` and `normal_alts` are skeleton variables that can be replaced by a possible candidate. The backbones can be used for knowledge inference and produce 6 labels: i) fine imaging modalities; ii) coarse imaging modalities; iii) imaging plane; iv) organ systems; v) binary abnormality (yes/no answers) and vi) categorical abnormality. With these labels, the VQA problem can be reduced to multi label multiplication and all the six tasks are optimized via a mini-batch gradient method. The authors consider that the two main tasks are the binary (v) and categorical ab-

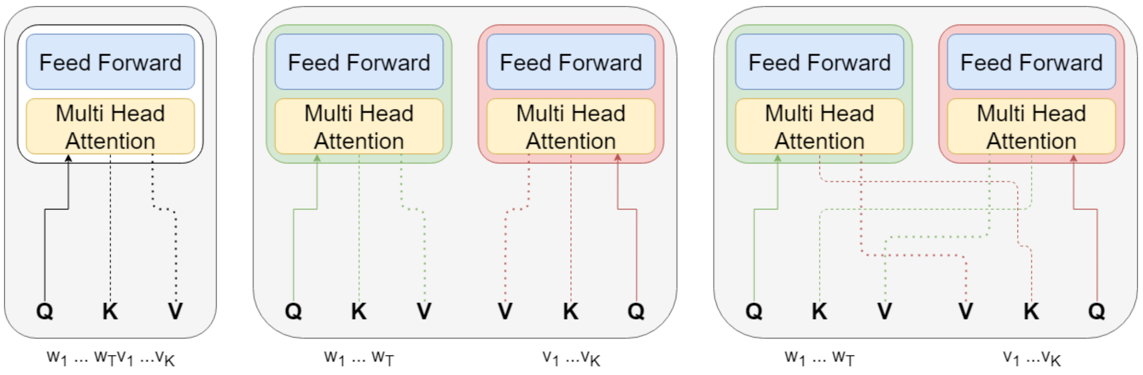


Figure 3.4: Graphical representation of single-stream, dual-stream intra-modal and dual-stream inter-modal, architectures for visual and language tasks.

normality (vi) classification tasks, while the rest four can be thought as regularization tasks. The authors also believe that all the tasks should have strong correlation to each other, i.e., the correct imaging modality and organ judgements should be strong prior knowledge for correct recognition of abnormality. Finally, the authors use an ensemble of different CNN architectures and image resize options. While very pragmatic, a drawback of this method of extracting backbones from the structure of the questions is that the questions in the dataset must follow this structure, and thus the approach has no flexibility to be used in other contexts/datasets.

3.3 Multimodal Work Unifying Vision and Language Through BERT

Approaches for unifying Vision and Language through the use of Transformers, particularly BERT, have been reported by several researchers. Given a set of word tokens $\{\mathbf{w}_1, ..., \mathbf{w}_T\}$ and of visual features $\{\mathbf{v}_1, ..., \mathbf{v}_K\}$, these V&L BERT models intend to produce cross-modal representations, and they can be grouped in two approaches single-stream and dual-stream. In single-stream approaches images and text are jointly processed by a single encoder. A BERT architecture is given a concatenation of the visual and language features as input. This allows for the attention mechanism to be made over both modalities. In dual-stream the inputs are encoded separately by Transformer layers before being jointly modeled in cross-modal Transformer layers.

Remembering the multi head attention of the Transformer architecture from Equation 2.10, the attention over the input is computed from the queries $\mathbf{Q}$, keys $\mathbf{K}$, and values $\mathbf{V}$. For the intra-modal layer, the attention of each modality is computed independently where as on the inter-modal layer the multi-head attention is computed over both modalities, which means the keys $\mathbf{K}$ and values $\mathbf{V}$ are from the complementary modality when computing the attention for a given modality. A graphical representation of single-stream and dual-stream interaction between the two modalities is depicted in Figure 3.4.

Bugliarello et al. (2020) developed a survey in which they compared different V&L BERT architectures in the same conditions, to empirically evaluate if there are performance differences between models in single and dual stream methods, as well as within these categories. The authors concluded that models achieve similar performance when trained on the same pretraining data and pretraining tasks. The claimed performance gaps narrow when models are pretrained on the same data. This is also true for single and dual streams models, which are on par. The authors also found that models show a significant variance between different runs in pretraining and finetuning, due to random initialization of the parameters. These differences between runs can be superior than 1 point. An interesting finding is that for the same architecture, the embedding layer plays a crucial role for downstream tasks. A compilation of V&L BERTs, corresponding to some of the models surveyed by Bugliarello et al. (2020) is provided in Table 3.1.

Chen et al. (2019) developed the UNiversal Image-TExt Representation (UNITER) approaches, which is an example of a single-stream model that uses the Transformer at its core model. UNITER is pre-trained in the following tasks: (i) Masked Language Modeling (MLM) conditioned on images; (ii) Masked Region Modeling (MRM) conditioned on text; (iii) Image-Text Matching (ITM); and (iv) Word-Region Alignment (WRA). The conditional masking tasks consist on masking some tokens from one modality for their value prediction (like in BERT) according to the surrounding tokens of that modality and all the other tokens from the complementary modality. The authors tested various types of masked region methods and concluded that masked object classification with KL-divergence achieved the best results, which consists on using the soft labels (i.e., the distribution of classes) of the object detector for supervised learning. In this way, MRC-kl uses a distillation approach to let UNITER learn the visual features. ITM consists on feeding the model with a sentence and a set of image regions as inputs, and asking for a binary classification as to weather the sentence corresponds to the image. For this task the output of the `[CLS]` token is used. For the WRA task, optimal transport is used to compute a matrix $\mathbf{T}$ to optimize the alignment between $\mathbf{w}$ and $\mathbf{v}$.

To create the image embeddings, a Faster-RCNN (Ren et al., 2016) is used to extract the visual features for each region. The locations of each region is encoded by a 7-dimensional tuple $[x_1, y_1, x_2, y_2, w, h, w * h]$. Both visual and location features are fed to a fully-connected layer to be projected in the same embedding space and finally they are added together. The text is embedded as in BERT, in which the input is first tokenized into WordPieces (Wu et al., 2016), and a position embedding is also summed. To use this model in VQA tasks, a muli-layer perceptron is added, which receives the output representation of the `[CLS]` token. This MLP is learned during finetuning for the VQA task as, given the formulation as a multilabel classification problem to minimize the cross-entropy over the ground-truth answers.

ViLBERT (Vision-and-Language BERT) is an example of a dual-stream model that achieved great

Type of V&L BERT Model	Name
Single-stream	UNITER (Chen et al., 2019)
	VisualBERT (Li et al., 2019a)
	VL-BERT (Su et al., 2019)
	Unicoder-VL (Li et al., 2019b)
Dual-stream	VilBERT (Lu et al., 2019)
	LXMERT (Tan and Bansal, 2019)

Table 3.1: Summarization of different V&L BERT models.

results (Lu et al., 2019). VilBERT consists of two parallel BERT models for separate vision and language processing. Each stream is a series of transformer blocks (TRM) that communicate through co-attentional transformer layers (Co-TRM), so information can be exchanged between modalities. In the Co-TRM layers, the keys and values from each modality are passed as input to the other modality's multi-head attention block, as described earlier. The input to VilBERT is represented as $\{\mathtt{IMG}, \mathbf{v}_1, ..., \mathbf{v}_K, \mathtt{CLS}, \mathbf{w}_1, ..., \mathbf{w}_T, \mathtt{SEP}\}$, where the output representations of IMG and CLS can be seen as holistic representation of images and text, respectively. Regarding pretraining tasks, VilBERT is trained for masked multi-modal modelling and image-text matching (ITM). The masked multi-modal task consists on randomly masking 15% of the input tokens of both modalities, and to reconstruct them given the remaining inputs. For the reconstruction of the visual features, VilBERT minimizes the KL-divergence of the distribution of classes predicted by it and the distribution of the object detector used to retrieve the visual features. The ITM task is the same that was described before. The authors use the outputs of the IMG and CLS tokens and perform an element-wise product that is fed to a linear layer, to learn whether the text describes the image. A Faster R-CNN model is used to extract the visual features and project both these and the location features are projected to the same embedding space to be summed. For text a standard BERT model is used. To fine-tune VilBERT on VQA, the task is formulated as a classification problem, and a two-layer MLP is learned on top of the element-wise product of the representations of the IMG and CLS tokens.

An intense test on the usage of visual features as input to BERT for Visual Question Generation was conducted by Scialom et al. (2020). This approach is graphically depicted in Figure 3.5. The authors compared the generated questions when using text only (the image caption), visual (the image) or multi-modal (both the caption and the image) as data input. The goal was to understand if the BERT abstractions are enough for the alignment of the visual and textual data, so the authors did not use some usual mechanisms such as multi-modal supervision as a pre-training step, on which models have access to text/image pairs as input to align the representations. In this method, the model is only fed a single modality, either text or an image. The authors also did not use image-specific losses, like Masked RoI or sentence-image prediction. The only loss that was used was the original MLM from BERT. The alignment between image and text was only made by a linear projection.

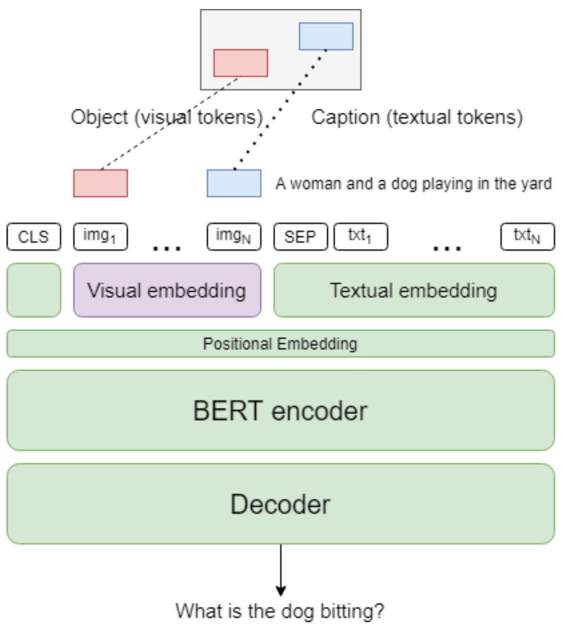


Figure 3.5: Overview of the model by Scialom et al. (2020). Captions are encoded via BERT embeddings, and visual embeddings are obtained via a linear projection layer.

To represent an input image in a textual form, the image needs to be converted into sequential data. To do that, an image X_{img} can be seen as a sequence of object regions $\text{img}_1, ..., \text{img}_N$. Images are processed by a Faster-RCNN which detects the $N = 36$ most salient regions. Each image is thus represented by $N = 36$ semantic embeddings $\mathbf{f}_1, ..., \mathbf{f}_N$ of dimensionality 2048, along with the corresponding box coordinates $\mathbf{b}_1, ..., \mathbf{b}_N$ of dimensionality 4. A final image region embedding $\mathbf{o}_j$ is the result of the concatenation of $\mathbf{f}_j$ and $\mathbf{b}_j$ of dimensionality $20148 + 4 = 2052$. Because the input dimensionality of BERT is 768, and the goal is to test if BERT can transfer knowledge beyond language, a simple cross-modal projection $\mathbf{W}$ of dimensionality 2052×4 is used. Finally, the image is represented as $X_{\text{img}} = (\mathbf{W} \cdot \mathbf{o}_1, ..., \mathbf{W} \cdot \mathbf{o}_n)$, and the BERT model does not distinguish between sequences of images or text tokens.

In the caption only mode, the model only had acess to textual input, where as in the image only mode, the model only had access to the input image. The BERT positional embedding of image token is added to the visual representation of image. The weight parameters of textual BERT are frozen and only the projection layer $\mathbf{W}$ is learnable. The authors also used a full model which has access to both the image and caption inputs. The two inputs are concatenated and separated by the special BERT token SEP. In this step, the model used the parameters learned by BERT in the text only mode and the projection matrix $\mathbf{W}$ from the image only mode.

The authors found that the model performs well using only images as input, and that it further improves when unfreezing the BERT weights. The multi modal experiments of images + captions gives the best results, leveraging the information encoded in both modalities. With these findings, it can be concluded that the abstractions encoded in a pre-trained BERT can generalize beyond text, with the use of a simple linear projection.

3.4 Contrastive Learning Methods

Self-supervised methods are learning techniques to create a supervised problem from unlabeled data, by exploring relationships between data to create new labels. Regarding NLP tasks, the masked language modelling in BERT is an example of a self-supervised method. However, on the vision domain these methods usually consisted on complex changes of the architecture of the models, or of the training procedure, which prevented wider adoption (Chen et al., 2020). SimCLR, developed by Chen et al. (2020), is a simple method for self-supervising learning which learns generic representations of images from an unlabeled dataset which then can be fine-tuned with a small labeled dataset for a classification task. These generic representations are learned by maximizing the agreement between different transformations of the same image, while minimizing the agreement between transformations of different images. This learning process is called contrastive learning.

Each retrieved image is subject to simple augmentations to create two transformed images: random cropping (with flip and resize), a random color distortion and Gaussian blur. The image encoder can be any suitable model, and the authors used a ResNet. The encoded results are fed into a MLP that computes a non-linear projection. The authors found that this non-linear projection improves the generalization capabilities of the model. A contrastive loss is defined to update both the CNN and MLP parameters. Considering a minibatch of N examples the data augmentation techniques produces two transformation of each image resulting in $2N$ data points. Considering a transformation pair from the same original image as a positive pair, the remaining $2(N-1)$ pairs are considered negative samples. Given a similariy function between two vectors (i.e, the cosine similarity $\mathrm{sim}(\mathbf{u},\mathbf{v}) = \mathbf{u}^T\mathbf{v}/(\|u\|\|v\|)$), the loss function for a positive pair (i,j) is defined as:

$$\mathcal{L}_{i,j} = -\log \frac{\exp(\mathrm{sim}(\mathbf{z_i},\mathbf{z_j})/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\mathrm{sim}(\mathbf{z_i},\mathbf{z_j})/\tau)} \, . \tag{3.14}$$

In the previous equation $\mathbb{1}_{[k\neq i]} \in \{0,1\}$ is an indicator function that evaluates to 1 iff $k \neq i$. The parameter τ is a temperature parameter. Chen et al. (2020) term this loss as the normalized temperature-scaled cross entropy loss (NT-Xent).

SimCLR shows great results when finetuning a very low amount of data. When using only 1% of the labels, it achieves 85.8 % top-5 accuracy on ImageNet, outperforming AlexNet with 100$\times$ fewer labels. The authors made three important findings in their research. First, the augmentation techniques allowed the network to produce good representations. In cropping, the relations between the two patches can be one of two, namely a patch inside another (global and local view) or between adjacent patches (adjacent view). Also, the color distortion is important so that the network does not make trivial generalizations, as different patches of the same image have similar color histograms and the network could distinguish images by considering them. No single augmentation technique suffices to achieve good result, and the composition of these techniques is necessary. Second, on what regards the projection head, it was found that using the input that is fed to the projection layer is better for downstream tasks than using the final output. The authors hypothesized that the final projection can lose useful features for downstream tasks. Finally, the authors also noted that larger batch sizes, deeper networks, and training for longer amounts of time lead to significant improvements. These improvements seem larger when compared to supervised learning. This is because a larger batch size and training longer provide more negative examples, which facilitate convergence and more robust representations.

Contrastive learning is an attractive method for the medical domain, due to the much smaller size of this type of datasets. Most previous work relies on transfer learning of convolutional neural networks pretrained in the ImageNet dataset, which is suboptimal due to the domain differences, image characteristics and high inter-class similarity. Zhang et al. (2020) developed a framework for learning visual representations by exploiting the naturally occurring pairing of images and textual data, which they called Contrastive VIsual Representation Learning from Text (ConVIRT). ConVIRT learns visual representations by maximizing the agreement between true image-text pairs versus random pairs, via a bidirectional contrastive objective between the image and text modalities.

An overview of ConVIRT is shown in Figure 3.6 and can be described as follows: given a paired input of $(\mathbf{x_v}, \mathbf{x_u})$, where $\mathbf{x_v}$ represents an image and $\mathbf{x_u}$ a text sequence that describes the visual information in $\mathbf{x_v}$, the method first converts these representations into d-dimensional representations $\mathbf{v}$ and $\mathbf{u}$, respectively. Then, a transformation t_v is applied at the input image, creating a view $\tilde{\mathbf{x}}_v$ which is fed to an image decoder f_v that outputs a vector $\mathbf{h}_v$ which in turn is fed to a non-linear projection function g_v. This results in a final representation $\mathbf{v} \in \mathbb{R}^d$:

$$\mathbf{v} = g_v(f_v(\tilde{\mathbf{x}}_v)) \,. \tag{3.15}$$

A similar process is made to obtain the text representation $\mathbf{u} \in \mathbb{R}^d$:

$$\mathbf{u} = g_u(f_u(\tilde{\mathbf{x}}_u)) \,. \tag{3.16}$$

The projection functions g_v and g_u project the representations of both modalities to the same d-dimensional space for constrastive learning. During training, given a minibatch of N input pairs $(\mathbf{x_v}, \mathbf{x_u})$, the representation pairs $(\mathbf{v}, \mathbf{u})$ are computed. The training loss of ConVIRT is composed by two parts,

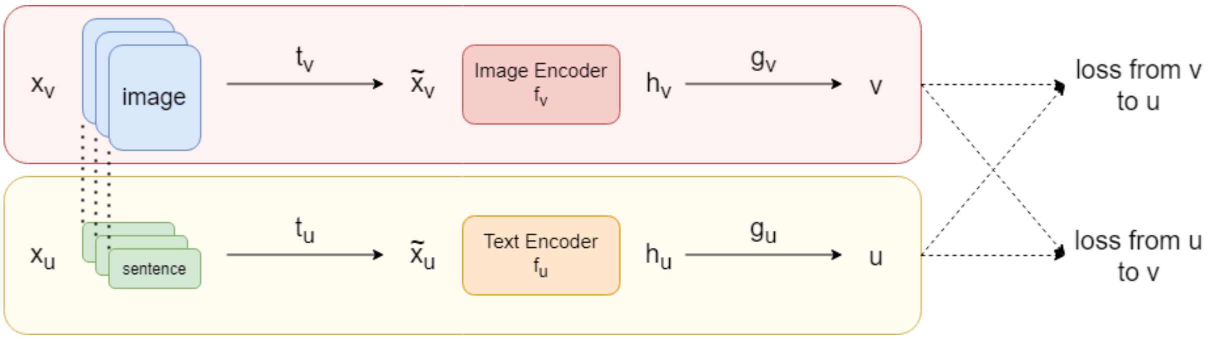


Figure 3.6: ConVIRT framework by Zhang et al. (2020): the image pipeline is shown in the top, and the text pipeline is shown in bottom. The method tries to maximize the agreement between the image-text representations with the bidirectional losses $l^{(v\to u)}$ and $l^{(u\to v)}$.

the first being an image-to-text constrastive loss for the i-th pair:

$$\ell_i^{(v\to u)} = -\log \frac{\exp(\mathrm{sim}(\mathbf{v_i}, \mathbf{u_i})/\tau)}{\sum_{k=1}^{N} \exp(\mathrm{sim}(\mathbf{v_i}, \mathbf{u_i})/\tau)}, \tag{3.17}$$

where $\mathrm{sim}()$ is the cosine similarity function between two vectors and τ is a temperature parameter. A similar text-to-image contrastive loss is defined as follows:

$$\ell_i^{(u\to v)} = -\log \frac{\exp(\mathrm{sim}(\mathbf{u_i}, \mathbf{v_i})/\tau)}{\sum_{k=1}^{N} \exp(\mathrm{sim}(\mathbf{u_i}, \mathbf{v_i})/\tau)}. \tag{3.18}$$

The final training loss is a weighted combination of the two previous functions, determined by $\lambda \in \{0, 1\}$ as a scalar weight, averaged over all positive image-text pairs in each minibatch:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left(\lambda \ell_i^{(v\to u)} + (1-\lambda) \ell_i^{(u\to v)} \right). \tag{3.19}$$

For the image encoder f_v, the authors use a ResNet, while for the text encoder f_u they used BERT. The image transformations t_v are cropping, horizontal flipping, affine transformation, color jittering and Gaussian blur. In color jittering, only brightness and contrast adjustments were applied given the monochromatic characteristic of the images. For the text transformation t_v, a simple sentence sampling of the input document is performed. The authors do not use a more aggressive approach so that the semantic meaning of the transformation is preserved. Models pre-trained with ConVIRT required only 10% as much labeled training data as an ImageNet initialized counterpart, to achieve better or comparable performance in medical imaging classification tasks.

3.5 Scaling Convolutional Neural Networks

Since the development of AlexNet, convolutional neural networks demonstrated to be highly suitable for vision tasks. New state-of-art results were subsequentely achieved by new wider and deeper networks, or using larger input image resolutions for training and evaluation. The conventional method is to scale one of these dimensions, which requires a tedious manual tuning and still often leads to sub-optimal results. Tan and Le (2020) made two important observations: scaling up any dimension of the network (width, depth, image resolution) improves accuracy but the accuracy gain diminishes for bigger models. Also, the different dimensions are not independent. A higher resolution input demands an increase of the network depth, such that similar features can include more pixels. The width should also be increased for higher resolutions in order to capture more fine-grained patterns with more pixels. Because of this relation between the dimensions, all of them should be considered and balanced in order to achieve better accuracy and efficiency. Tan and Le (2020) proposed a new compound scaling method, which uses a compound coefficient ϕ to uniformly scale the network width, depth and resolutions:

$$\begin{aligned} \text{depth}: \quad & d = \alpha^{\phi}\,, \\ \text{width}: \quad & w = \beta^{\phi}\,, \\ \text{resolution}: \quad & r = \gamma^{\phi}\,, \\ \text{s.t.} \quad & \alpha \cdot \beta^2 \cdot \gamma^2 \approx 2\,, \\ & \alpha \geq 1, \beta \geq 1, \gamma \geq 1\,. \end{aligned} \tag{3.20}$$

The constants α, γ, and β can be determined by a small grid search. The ϕ coefficient represents the amount of resources that are available for model scaling, while α, γ, and β specify how to assign the resources for the network depth, width, and resolution. The resource for optimization is FLOPS (Floating

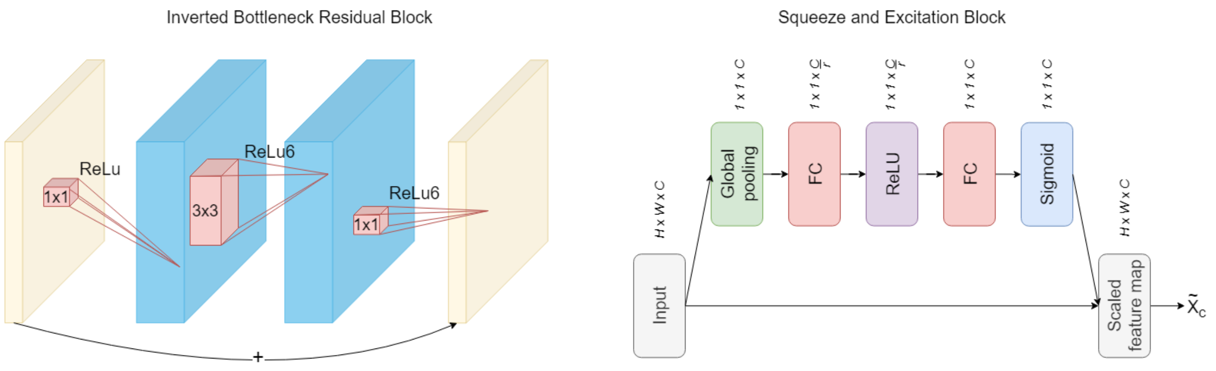


Figure 3.7: Graphical representation of an inverted residual block, on the left, and for a squeeze and excitation block on the right.

Stage i	Operator $\hat{\mathcal{F}}_i$	Resolution $\hat{H}_i \times \hat{W}_i$	#Channels $\hat{C}_i$	#Layers $\hat{L}_i$
1	Conv3×3	224 $\times$ 224	32	1
2	MBConv1, k3×3	112 $\times$ 112	16	1
3	MBConv6, k3×3	112 $\times$ 112	24	2
4	MBConv6, k5×5	56 $\times$ 56	40	2
5	MBConv6, k3×3	28 $\times$ 28	80	3
6	MBConv6, k5×5	14 $\times$ 14	112	3
7	MBConv6, k5×5	14 $\times$ 14	192	4
8	MBConv6, k3×3	7 $\times$ 7	320	1
9	Conv1x1 & Pooling & FC	7 $\times$ 7	1280	1

Table 3.2: EfficientNet-B0 baseline network – Each row describes a stage i with $\hat{L}_i$ layers, with input resolution $\langle \hat{H}_i \times \hat{W}_i \rangle$ and output channels $\hat{C}_i$.

Point Operations) and the expression $\alpha \cdot \beta^2 \cdot \gamma^2$ represents how each dimension growth affects the total FLOPS. The constraint $\alpha \cdot \beta^2 \cdot \gamma^2 \approx 2$ represents that for every new ϕ, the total FLOPS demanded by the network will approximately increase by 2^ϕ.

To test compound scaling, the authors implement a family of CNN networks which they called EfficientNet, and whose main building block is inverted bottleneck (Sandler et al., 2019) block with squeeze-and-excitation optimization (Hu et al., 2019). These blocks are shown in Figure 3.7. The authors use the swish activation function, corresponding to $x \cdot \sigma(\beta x)$, which addresses the lack of gradient in negative values in ReLU and smooths the gradient around 0. Given a EfficientNet-B0 baseline, described in Table 3.2, the authors apply the compound scaling method by setting $\phi = 1$, which assumes that there are twice more resources available, and do a small grid search for α, γ, β. Given the values of α, γ, β, the baseline can be scaled by a new value of ϕ according to the resources available, creating new networks from the scaling of the baseline.

Regarding the inverted residual blocks proposed by Sandler et al. (2019), when compared to the residual blocks of ResNet, it can be seen that the convolutional operations are switched and we also have the bottleneck operations. The input is low-dimensional and a 1×1 convolution is used to expand it to a high-dimensional space. In this high-dimensional space, the 3×3 convolution is performed and finally the feature map is projected to the low-dimensional space to be added to the input. Thus, the residuals are connected by the lower dimension bottlenecks. Because of the projections in the low-dimension space, the authors found that using linear projections instead of non-linear prevents information loss and leads to better results.

The squeeze and excitation mechanism was originally developed by Hu et al. (2019) for modeling the information and inter-dependencies of the channels of an image. With this method, the information of each channel can be decoupled and used for building better representations. First, given a feature map $\mathbf{U} \in \mathbb{R}^{H \times W \times C}$, in the squeeze phase we have that a global average pooling is applied to each channel $\mathbf{u_c}$, to create a vector that gathers the understanding of each channel, $\mathbf{z_c}$. Then, the excitation phase captures channel-wise dependencies, the vector $\mathbf{z}$ is fed to two fully connected layers to capture non-linear and non-mutually-exclusive interactions between channels. This is achieved by using a gating mechanism with a sigmoid activation, where $\mathbf{W_1} \in \mathbb{R}^{\frac{C}{r} \times C}$ and $\mathbf{W_2} \in \mathbb{R}^{C \times \frac{C}{r}}$:

$$\mathbf{s} = \sigma(\mathbf{W_2}\,\mathrm{ReLU}(\mathbf{W_1 z}))\,. \tag{3.21}$$

To reduce the model complexity, the two fully connected layers form a bottleneck to reduce the dimensional space with a ratio r. The matrix $\mathbf{W_2}$ is a dimension-increasing layer, so that the dimensions of the output return to the channel dimensions of $\mathbf{U}$. Then, the final feature map $\mathbf{U}$ is rescaled with the activations in s, according to:

$$\tilde{\mathbf{x}}_c = \mathbf{s_c u_c}\,,\,. \tag{3.22}$$

3.6 Overview

This chapter presented different methods to address Visual Question Answering (VQA), in general and in the particular case of medical images. From the work surveyed here, it can be seen that most proposals treat VQA as a classification task. Different methods were presented based on bottom-up or top-down attention (Anderson et al., 2018). In particular, for the medical domain, different methods proposed for the VQA-MED task of ImageCLEF were presented.

Studies addressing methods to unify vision and language modalities, with the use of Transformers, were also surveyed. These type of approach is the basis of the architecture of the MMBERT model. The use of contrastive learning techniques as pre-training tasks to further improve the performance of models, and a strong image encoder named EfficientNet, were also detailed.

4

Extending the Multimodal Medical BERT (MMBERT) Model

Contents

4.1 The Original MMBERT

The Multimodal Medical BERT (MMBERT) proposed by Khare et al. (2021) is a multi-modal Transformer architecture designed to address medical VQA. A ResNet152 encoder, pre-trained on ImageNet, is used to capture image representations at 5 different resolutions, while a shallow Transformer is used to model the interaction between visual features and the question. Convolution and global average pooling operations are performed over the results from five different blocks of the ResNet architecture, to obtain 5 visual tokens with the same dimensionality of the Transformer. To obtain text representations, BERT (Devlin et al., 2019) embeddings are first used. The text is tokenized into WordPieces and each token is represented with the sum of token, position, and segment embeddings from a pre-trained BERT model. The tokens of each modality are concatenated into a representation $\{\text{CLS}, \text{img}_1, ..., \text{img}_5, \text{SEP}, c_1, ..., c_N, \text{SEP}\}$, where $\text{img}_1, ..., \text{img}_5$ corresponds to the visual features and $c_1, ..., c_N$ represents the text features. The Transformer encoder model has 4 layers, a hidden state size of 768, and 12 attention heads, using the ReLU activation function (Nair and Hinton, 2010) after the feed-forward layers.

The authors used the Radiology Objects in COntext (ROCO) dataset (Pelka et al., 2018) of medical images and their corresponding captions, for model pre-training. This dataset contains over 81k images of several medical imaging modalities. Each image is accompanied by its caption and, additionally, with keywords extracted from the caption. MMBERT is pre-trained with a Masked Language Modeling (MLM) objective, which consists on predicting the original tokens of masked positions based on the remaining text and image features. To improve the performance on medical data, only medical keywords are masked and common words are left untouched, leveraging the keyword information of the dataset.

For fine-tuning to the VQA-MED task, instead of taking the standard approach of using the `[CLS]` token representation from the last layer of the Transformer, the representations of each token in the last layer are averaged and fed to a feed-forward component to obtain an answer classification. The authors report an accuracy of 62.4% and a BLEU score of 64.2% with the aforementioned approch, at the same time also reporting that the results can be improved by having 5 independent models for each type of question category, in this case achieving 67.2% in terms of overall accuracy and 69.0% for BLEU.

4.2 The Proposed Extensions over MMBERT

Besides reproducibility, this study also tried to advance over MMBERT in several directions. For instance, regarding the image encoder, we replaced the ResNet152 in the original architecture with EfficientNetV2 (Tan and Le, 2021), which is a recent CNN architecture that achieved state-of-the-art performance in several image classification tasks. In the multi-modal encoder part of the model, instead of a Transformer architecture, we tested the use of a RealFormer (He et al., 2020), which is an extension

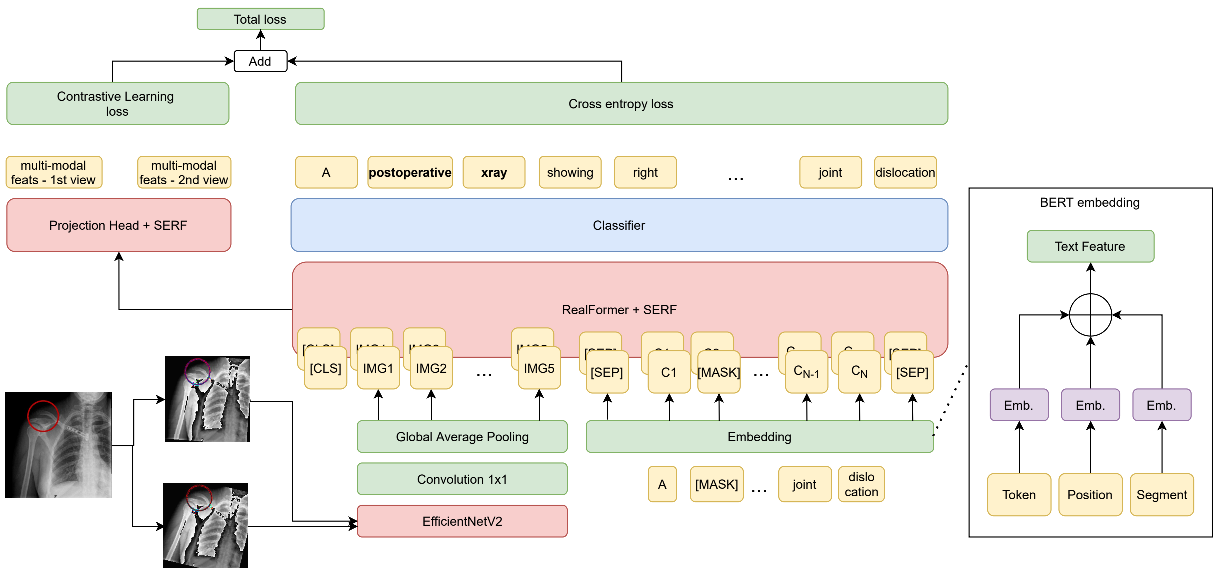


Figure 4.1: Graphical depiction for the MMBERT architecture, together with our extensions and featuring the MLM and SimCLR-based contrastive pre-training tasks. Items coloured red indicate extensions over the original architecture.

of the Transformer architecture (Vaswani et al., 2017) that creates a residual path between the attention scores of each layer. The ReLU activation function was also replaced by SERF (Nag and Bhattacharyya, 2021), which addresses some drawbacks of the ReLU and achieved promising results on various types of tasks with different models. Each of these methods were recently introduced in the last 2 years, reporting better results compared to their counterparts in different tasks, and so were considered as possible extensions to further the performance in VQA task when combined, to achieve better results over the original MMBERT. A further detailed description of them is provided in the following sections.

Following the MMBERT approach, the visual features are captured at five different depths of the EfficientNetV2, resulting in five visual tokens representing different resolutions. As for the text features, we leverage BERT (Devlin et al., 2019) to first tokenize the text into WordPieces and then to create embeddings for the tokens.

Contrastive learning is an attractive method for the medical domain, due to the much smaller size of this type of datasets. Most previous work relies on transfer learning of convolutional neural networks pretrained in the ImageNet dataset, which is suboptimal due to the domain differences, image characteristics and high inter-class similarity (Zhang et al., 2020). Also motivated by the results reported in Zhang et al. (2020), described in Section 3.4, the model is pre-trained with the same masked language modeling objective as the original MMBERT, although combined with a new pre-training task that leverages contrastive learning. With this method, the model is given a batch of image+caption pairs where each pair has 2 views. Each view is the result of data augmentation from the same original data: image transformations are used to produce 2 views of the image, and back-translation is used to augment the

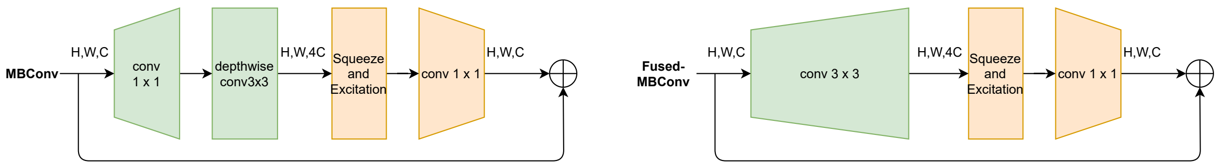


Figure 4.2: Graphical depiction of the MBConv (Sandler et al., 2019) and Fused-MBConv (Gupta and Akin, 2020) blocks used in EfficientNetV2 (Tan and Le, 2021) architecture.

caption text. The representations to be contrasted are extracted at the end of the multi-modal Real-Former, by averaging the representations for each token and passing them to a feed-forward layer with the SERF activation function, to produce features of size 128. The training objective involves pulling together the multi-modal representations from the same original sample, and pushing apart the views from different samples. We experimented with the use of the loss function from SimCLR (Chen et al., 2020), and also with a supervised contrastive loss (Khosla et al., 2021). Given the loss for the contrastive task $\mathcal{L}^{con}$ and the cross-entropy loss for the masked language modeling $\mathcal{L}^{MLM}$, the final pre-training loss $\mathcal{L}^{tot}$ is a simple addition of the two:

$$\mathcal{L}^{tot} = \mathcal{L}^{con} + \mathcal{L}^{MLM}. \tag{4.1}$$

For the downstream task of VQA, we also experimented with an asymmetric loss (Ben-Baruch et al., 2021) which addresses the class imbalance present in our classification task. Figure 4.1 shows a graphical depiction for the pre-training process.

4.2.1 The EfficientNetV2 Image Encoder

EfficientNetV2 (Tan and Le, 2021) is a family of convolutional neural networks that improved upon EfficientNet (Tan and Le, 2020) with faster training speed and better parameter efficiency. Neural architecture search was used to build these models, in which the key components are the MBConv (Sandler et al., 2019) and Fused-MBConv (Gupta and Akin, 2020) blocks depicted in Figure 4.2.

MBConv blocks consist on applying a 1×1 convolution to increase the number of channels in an input representation and then a depthwise 3×3 convolution in the high dimensional space. Then, a 1×1 convolution is used to reduce the number of channels to the original size, so that this final feature map can be added to the input in a residual connection. Depthwise convolutions have fewer parameters and FLOPs than regular convolutions, but cannot be processed on modern accelerators in their full capacity. The authors proposed the Fused-MBConv block to address this limitation, replacing the depthwise 3×3 convolution and expansion 1×1 convolution, in MBConv, with a single regular 3×3 convolution. Neural architecture search is used to automatically search for the best combination of these two building blocks, together with squeeze-and-excitation operations (Hu et al., 2019). The authors also proposed to train the model with a progressive learning technique, which consists on gradually increasing the image

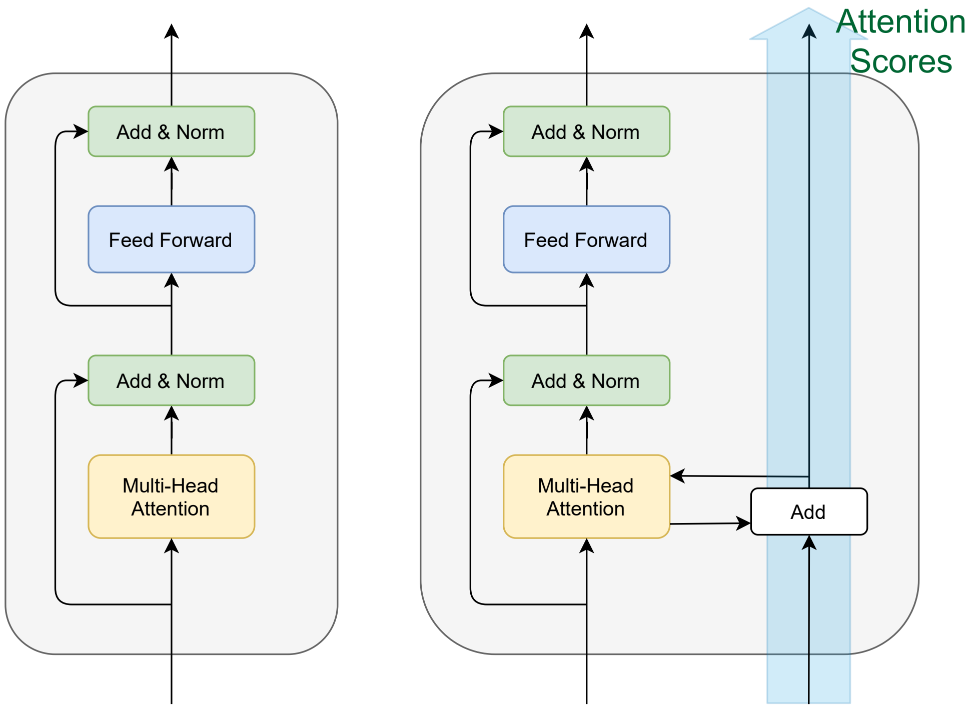


Figure 4.3: Left - Widespread Post-LN layer used in (e.g.) BERT. Right - Realformer (He et al., 2020) with a direct path to propagate attention scores using a residual connection.

size as the training progresses, while also increasing the strength of regularization techniques such as dropout (Srivastava et al., 2014) and image augmentations.

EfficientNetV2 was chosen because of the higher performance in image classification tasks compared to the ResNet152 used in MMBERT. The EfficientNetV2 used in our experiment has a medium size, with $\approx 54M$ parameters, which is similar to the number of parameters of ResNet152. The model was pre-trained on ImageNet-1k with progressive learning (Tan and Le, 2021). The model and pre-trained weights were provided in the `timm` Python package[1].

4.2.2 The RealFormer Multi-Modal Encoder

The original Transformer architecture (Vaswani et al., 2017) follows an encoder-decoder structure based on self-attention. The encoder is composed by a stack of layers (originally $N = 6$) and each layer is composed by a stack of sub-layers, consisting of a multi-head attention module and a feed-forward module. A residual connection is also applied around each sub-layer, followed by normalization.

The attention is computed with a scaled dot-product attention operation, which corresponds to a weighted sum of values in V (i.e. token representations), where the weights are computed by a function

[1] https://github.com/rwightman/pytorch-image-models#may-14-2021

of queries Q and keys K:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right). \quad (4.2)$$

In the previous equation, $\frac{QK^T}{\sqrt{d_k}}$ represents the raw attention scores for each (query,key) pair, while d_k represents the dimensionality of queries and keys.

The attention module performs an aggregation of the attention calculation over multiple heads. The resulting computation of all heads is concatenated and linearly transformed by a matrix W^O. This process can be formally defined as:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_i, ..., \text{head}_h)W^O, \quad (4.3)$$

where $\text{head}_i = \text{Attention}(QW_i^Q, KW_i^K, VW_i^V)$, Q and K are matrices with dimensionality d_k, and V is a matrix with dimensionality d_v. The parameters W_i^Q, W_i^K, and W_i^V are matrices that linearly project queries, keys, and values into the attention space of the *i*-th head.

In the feed-forward module, non-linear transformations are performed, using activation functions like ReLU (Nair and Hinton, 2010). Layer normalization modules are also used above each sub-layer, to stabilize training.

Residual Connections in RealFormer

The RealFormer (He et al., 2020) follows the same general design, adding skip edges to connect multi-head attention modules in adjacent layers. The pre-softmax attention scores from a previous layer are additional inputs to the current one:

$$\text{ResidualMultiHead}(Q, K, V, \text{Prev}) = \text{Concat}(\text{head}_1, ..., \text{head}_h)W^O. \quad (4.4)$$

The variables head_i correspond to $\text{ResidualAttention}(QW_i^Q, KW_i^K, VW_i^V, \text{Prev}_i)$ operations, and Prev_i is a slice of the previous layer corresponding to head_i. A residual attention operation adds the residual score on top of $Prev_i$ and then the regular weighted sum is computed as usual:

$$\text{ResidualAttention}(Q', K', V', \text{Prev}') = \text{softmax}\left(\frac{Q'K'^T}{\sqrt{d_k}} + \text{Prev}'\right)V'. \quad (4.5)$$

Finally, the pre-softmax attention scores of the current layer, corresponding to $\frac{Q'K'^T}{\sqrt{d_k}} + \text{Prev}'$, are passed over to the next layer.

Regarding the parameters of the architecture used in our tests, the model has 4 encoder layers, a hidden state size of 768, and 8 heads for the multi-head attention module.

SERF Activation Function

The original ReLU activation function was replaced by SERF (Softplus ERror activation Function) (Nag and Bhattacharyya, 2021). Activation functions are important for their role in introducing non-linearity to neural networks. Rectified Linear Unit (ReLU) (Nair and Hinton, 2010) activation function is widely used for its simplicity and effectiveness, showing better generalization and convergence properties compared to the $\mathrm{sigmoid}$ and $\tanh$ functions. However, ReLU is not differentiable which can introduce inconsistencies in the training process and leads to gradient information loss caused by transforming the negative inputs to zero. SERF is a recently proposed activation function that is smooth, non-monotonic, and fully differentiable, avoiding issues with the gradient-based optimization process. Given the Gauss error function $\mathrm{erf}()$, SERF can be formally defined as:

$$\mathrm{f}(x) = x \times \mathrm{erf}(\ln(1 + e^x)) \,. \tag{4.6}$$

Because of numerical stabilization issues in the calculation of the $\log$ and exponential functions, the value of x is clipped if it surpasses a threshold before the calculation of the exponential, to prevent overflow. It is also noted that for:

$$c \gg 1, \mathrm{f}(c) = c \times \mathrm{erf}(\ln(1 + e^c)) \approx c \times \mathrm{erf}(c) \approx c \times 1 = c \,.$$

4.2.3 Contrastive Pre-Training

Contrastive learning is an attractive method for initializing models operating with medical data, due to the much smaller size of the available annotated datasets, when compared to other domains. In this work, we experimented with two contrastive strategies for model pre-training, namely SimCLR (Chen et al., 2020) and SupCon (Khosla et al., 2021).

In both cases, the transformations used for image augmentation include cropping, affine transformations, and color jittering. Back-translation was applied to augment the text, where the captions are first translated to Spanish, German and French, and then translated back to English. Transformer models based on Marian-NMT were used in pre-processing to obtain the translated text[2]. The back-translation method provides a simple method of augmentation, preserving the context of the sentence with word changes. Each batch consists of N pairs of image+caption, where each pair has 2 views. For informing the supervised contrastive loss, a similarity matrix can be computed with basis on the captions. Different similarity functions were tested, namely the Jaccard similarity between word tokens, or a semantic metric based on the use of sentence-transformers (Reimers and Gurevych, 2019), specifically the model named `all-mpnet-base-v2`[3].

In brief, SimCLR (Chen et al., 2020), is a simple method for self-supervising learning maximizing the agreement between different transformations of the same input, while minimizing the agreement

[2] https://huggingface.co/transformers/model_doc/marian.html
[3] https://huggingface.co/sentence-transformers/all-mpnet-base-v2

between transformations of different inputs.

The representations resulting from the last RealFormer layer, before the classifier, are processed with average pooling and fed into a feed-forward layer using the SERF that computes a non-linear projection, producing features of size 128. Considering a mini-batch of N examples, the data augmentations produce two transformations of each instance resulting in $2N$ data points. Considering a transformation pair from the same original inputs as a positive pair, the remaining $2(N-1)$ pairs are considered negative samples. Given a similariy function between two vectors (i.e, the cosine similarity $\mathrm{sim}(u,v) = u^T v/(\|u\|\|v\|)$) and considering $i \in I \equiv \{1, ..., 2N\}$ as the index of an arbitrary augmented sample and j the index of the other augmented sample originating from the same source sample, the loss function $\mathcal{L}^{self}$ is defined as follows:

$$\mathcal{L}^{self} = -\sum_{k=1}^{2N} \log \frac{\exp(\mathrm{sim}(z_i, z_j)/\tau)}{\sum_{k=1}^{2N} \mathbb{1}_{[k\neq i]} \exp(\mathrm{sim}(z_i, z_j)/\tau)} \,. \tag{4.7}$$

In the previous equation $\mathbb{1}_{[k\neq i]} \in \{0, 1\}$ is an indicator function that evaluates to 1 iff $k \neq i$. The parameter τ is a temperature parameter.

Supervised Contrastive (SupCon) learning, developed by Khosla et al. (2021), extends methods like SimCLR to leverage available label information, in the form of classes associated to instances or similarities between instances. In SimCLR, only one positive is extracted to pull data augmentations of the same sample together. In SupCon, normalized representations for instances from the same class (or instances known to be similar) are pulled closer together than representations from different classes. The label information allows the use of many more positives per anchor, in addition to the many negatives. The training method is similar to that of SimCLR, where data augmentation is used to obtain two views of the batch, which are then fed to the model to obtain representations that are re-projected with a feed-forward layer. The loss function is designed to include the information or similarity between instances, generalizing for an arbitrary number of positives. The supervised contrastive loss function is defined as:

$$\mathcal{L}^{sup} = \sum_{i\in I} -\frac{1}{2N} \sum_{p\in I} \log \frac{\mathrm{label}(p,i)\cdot \exp(\mathrm{sim}(z_i, z_p)/\tau)}{\sum_{k\in I} \mathbb{1}_{[k\neq i]} \cdot \mathrm{label}(k,i) \cdot \exp(\mathrm{sim}(z_i, z_k)/\tau)} \,. \tag{4.8}$$

Here, $i \in I \equiv \{1, ..., 2N\}$ is the index of an arbitrary augmented sample, $\mathrm{label}(k,i) \in [0, 1]$ encodes the similarity between samples k and i (e.g., derived from the textual captions), and $\mathbb{1}_{[k\neq i]} \in \{0, 1\}$ is an indicator function that evaluates to 1 iff $k \neq i$. The parameter τ is again a temperature parameter, set to 0.07 in our experiments.

SupCon allows every positive in the batch (from augmentation or the other similar instances within the batch) to contribute to the numerator in Equation 4.8, resulting in a more robust clustering of the representation space. It also preserves the summation over negatives in the contrastive denominator, wherein the ability to discriminate between signal and noise (negatives) can be improved by adding more negative examples. Authors, through gradient analysis, also demonstrate that this loss has the ability to perform hard positive/negative mining.

4.2.4 Asymmetric Loss for VQA Fine-Tuning

A recent study on image classification proposed an asymmetric loss function (Ben-Baruch et al., 2021) for imbalanced multi-label or multi-class classification problems. For multi-class problems, like ours, the main idea is to decouple the processing of the positive and negative cases, assigning them different exponential decay factors.

$$\begin{cases} \mathcal{L}_{+} = (1-p)^{\gamma_{+}} \log(p) \\ \mathcal{L}_{-} = p^{\gamma_{-}} \log(1-p) \end{cases} . \tag{4.9}$$

In the previous equation, $\mathcal{L}_{+}$ and $\mathcal{L}_{-}$ are the positive and negative parts of the loss, respectively, p is the networks's output probability vector, and γ_{+} and γ_{-} are the positive and negative focusing parameters, respectively.

For $\gamma_{-} > 0$, the contribution of the negative part is down-weighted when the probability is low $(p \ll 0.5)$. The contribution of the positive part can be emphasized by setting $\gamma_{-} > \gamma_{+}$, to help the network learn more meaningful representations. In our tests, we used $\gamma_{-} = 4$ and $\gamma_{+} = 0$, which are the values used by the authors for single-label classification, and given by default in the original source code[4].The final loss is obtained by combining $\mathcal{L}_{+}$ and $\mathcal{L}_{-}$:

$$\mathcal{L} = -y\mathcal{L}_{+} - (1-y)\mathcal{L}_{-}. \tag{4.10}$$

Label smoothing is also performed, replacing the multi-class hard 0/1 classification targets, with targets of $\frac{\epsilon}{k-1}$ and $1-\epsilon$, respectively. Our tests used $\epsilon = 1e^{-8}$.

4.3 Overview

This chapter detailed the original MMBERT model and the proposed extensions addressing Visual Question Answering (VQA). In summary, a Transformer architecture is fed visual and text tokens. The visual tokens are extracted with a ResNet backbone at multiple resolutions, and the text tokens correspond to BERT embeddings. These tokens are concatenated and fed to the model, which obtains an answer with a classification objective. This model was pre-trained with medical data, namely through a masked language modeling task that also considers the image contents.

Section 4.2 details the proposed extensions over the original MMBERT approach. These include (a) using a stronger image encoder based on EfficientNetV2, (b) using a multi-modal encoder based

[4]https://github.com/Alibaba-MIIL/ASL

on the RealFormer architecture, (c) extending the pre-training task with a contrastive objective, and (d) using a novel loss function for fine-tuning the model to the VQA task, that specifically considers class imbalance.

5 Experimental Evaluation

Contents

This chapter details the experimental evaluation of the work produced for this dissertation. Firstly, in Section 5.1 a description of both the VQA-MED Task of ImageCLEF2019 and the ROCO datasets is provided, together with the experimental methodology applied and a description of the evaluation metrics. Section 5.2 presents the obtained results in the reproducibility study of the original MMBERT and of the proposed extensions. Section 5.3 describes a visualization mechanism, allowing for model interpretability. Finally, Section 5.4 provides an overview of the chapter.

5.1 Datasets and Experimental Methodology

5.1.1 Datasets

We tested the original MMBERT model, as well as our extensions, on the ImageCLEF 2019 VQA-MED dataset (Ben Abacha et al., 2019). The training split contains 12792 question-answer pairs and 3200 medical images for training, together with 2000 question-answer pairs and 500 images for validation. There are also 500 question-answer pairs and 500 images for testing. Each question-answer pair belongs to one of 4 different categories: modality, plane, organ system, and abnormality. Due to the high frequency of answers ”yes” and ”no”, we followed the approch in (Khare et al., 2021) and added a new category binary for these types of answers. In the training split, the number of different possible answers for each category is as follows: 44 for modality; 15 for plane; 10 for organ system; and 1485 for abnormality. There are 149 questions and 55 for the test validation and test splits, respectively, whose answers do not exist on the training split. This difference of answers happens mainly for the abnormality category. Since we are addressing the VQA problem as a classification task, these are answers that the model will never be able to produce.

The dataset used for pre-training the proposed model was Radiology Objects in COntext (ROCO) (Pelka et al., 2018), which is comprised of images from publications available on the PubMed Central Open Access FTP mirror. It contains over 81k radiology images with several medical imaging modalities. Each image is accompanied by its caption and, additionally, with keywords extracted from the image caption, the corresponding UMLS Semantic Types (SemTypes) and UMLS Concept Unique Identifiers.

5.1.2 Implementation Details

The chosen hyper-parameters follow the original MMBERT source code. For pretraining and fine-tuning, images are resized to 224×224 pixels. To avoid distortion, first the smaller edge is matched to 224, and then a center crop of size 224×224 is performed. Image crops, rotations, and color jittering are used for image augmentation.

For pre-training, the optimizer is Adam and the learning rate is $2e^{-5}$, which is reduced if the validation loss does not improve for 5 consecutive epochs. As per the source code, models are pre-trained for 10 epochs and the probability of masking a given medical token is 0.15. The medical keywords provided in the ROCO dataset allowed for the model to consider only medical words when pre-trained in the masked language modeling. These keywords were retrieved and compiled into a single Python dictionary and saved to a file to facilitate the access in pre-training. To provide text augmentation for the contrastive learning pre-training task for the proposed method, back-translation was applied, where the captions are first translated to Spanish, German and French, and then translated back to English. Transformer models based on Marian-NMT were used in pre-processing to obtain the translated text[1].

For fine-tuning, the Adam optimizer is also used, with a learning rate of $1e^{-4}$ which is reduced by a factor of 0.1 if validation loss does not improve for 10 consecutive epochs. The batch size was set to 16, but we also tested the use of larger batches with 48 instances. We use the loss or the accuracy in the validation split as early stopping criteria, with default value as 20, unless stated differently.

For both pre-training and fine-tuning, the whole model was trained in an end-to-end manner. All components of the architecture of the model (multi-modal Transformer backbone + image encoder CNN) were updated in both steps.

5.1.3 Evaluation Metrics

VQA-MED used two primary evaluation metrics, namely strict accuracy and BLEU (Papineni et al., 2002). Strict accuracy considers the exact matching of a provided answer and the ground-truth answer, corresponding to the percentage of correct classifications. BLEU is not as strict, and is used to capture the word overlap similarity between a system-generated answer and the ground-truth answer. While BLEU is more commonly used for the evaluation of text generation, the ImageCLEF organizers considered BLEU even when treating VQA as a classification task, mostly due to the hierarchical relationship of the candidate answers. For example, given a ground-truth answer `ct scan with contrast` and a prediction of `ct scan`, the strict accuracy would evaluate to 0 but BLEU would consider the fact that the system got two words correctly.

The main idea in BLEU is to compare n-grams of the generated text with the n-grams of the reference ground-truth, and count the number of matches. BLEU relies on a modified precision metric, calculated by counting up the number of candidate words which occur in any reference text, and then dividing by the total number of words in the candidate text. However, simply doing this would lead to evaluating sentences that are only a repetition of a single word that is present in the reference as having a high quality. To counter this, the authors use a modified unigram precision by firstly counting the maximum number of times a word occurs in the reference text, and clipping the total count of each candidate word

[1]https://huggingface.co/transformers/model_doc/marian.html

Table 5.1: Results on the VQA-MED 2019 dataset for MMBERT models pre-trained with masked language modeling. The highlighted row reports the result for our run with the default hyper-parameters in the original source code. Accuracy and BLEU means overall metric across categories.

Batch Size	Patience	Stop Criteria	Accuracy	BLEU
MMBERT	-	-	62.4	64.20
		Reproduced results		
16	40	Loss	48.2	50.62
48	40	Loss	55.6	57.65
48	20	Loss	56.0	58.36
16	**20**	**Accuracy**	**58.8**	**60.74**
16	40	Accuracy	58.9	60.88
48	20	Accuracy	59.6	61.36
48	40	Accuracy	59.8	61.59
48	80	Accuracy	59.2	61.27

by its maximum reference count.

$$p_n = \frac{\sum_{\mathcal{C}\in\{\text{Candidates}\}} \sum_{n\text{-gram}\in\mathcal{C}} \text{Count}_{\text{clip}}(n\text{-gram})}{\sum_{\mathcal{C}'\in\{\text{Candidates}\}} \sum_{n\text{-gram}\in\mathcal{C}'} \text{Count}(n\text{-gram}')} \,. \tag{5.1}$$

To combine the modified precisions for the various n-gram sizes, the authors observed that modified n-gram precision decays roughly exponentially with n. As such, to compute a weighted average w_n that takes this exponential decay into account, the authors use the geometric mean of the modified n-gram precisions. Regarding sentence length, the modified precision n-gram already penalizes larger sentences than intended, due to the clipping of the counting. Considering generated short sentences, if these had few tokens but all part of the reference text, this situation would lead to high scores for the modified n-gram precision. To approach this, the authors include a brevity penalty to penalize shorter sentences.

$$\text{BP} = \begin{cases} 1, & \text{if } c > r\,, \\ \exp(1 - \frac{r}{c}), & \text{if } c \leq r\,. \end{cases} \tag{5.2}$$

Finally, BLEU is computed as the geometric average of the modified n-gram precisions, p_n, using n-grams up to length N and positive weights w_n summing to one:

$$\text{BLEU} = \text{BP} \cdot \exp\left(\sum_{n=1}^{N} w_n \log(p_n)\right) \,. \tag{5.3}$$

5.2 Experimental Results

This section is divided in two. Firstly, a description of the results obtained for the reproducibility study is detailed. Then, the results of the different extensions and their support on improving the performance of the original MMBERT in the VQA task is discussed.

Table 5.2: Results on VQA-MED 2019 dataset for different configurations and pre-training objectives. The first row is the original architecture of MMBERT. SupCon-J represents the use of the SupCon loss when computing the similarity between sentences with the Jaccard similarity. SupCon-SB represents the SupCon loss with the cosine similarity between the sentence-BERT encodings. Accuracy and BLEU means overall metric across categories.

Image Encoder	Architecture	Activation	Loss	Pretraining task	Accuracy	BLEU
ResNet152	Transformer	ReLU	CE	MLM	58.80	60.74
Effic.NetV2	Transformer	ReLU	CE	MLM	59.40	61.36
Effic.NetV2	RealFormer	ReLU	CE	MLM	59.20	61.52
Effic.NetV2	RealFormer	SERF	CE	MLM	60.00	62.39
Effic.NetV2	RealFormer	SERF	ASL	MLM	59.80	61.55
Effic.NetV2	RealFormer	SERF	ASL	MLM + SimCLR	59.80	61.50
Effic.NetV2	RealFormer	SERF	ASL	MLM + SupCon-J	60.20	62.50
Effic.NetV2	RealFormer	SERF	ASL	MLM + SupCon-SB	60.60	62.98
Effic.NetV2	RealFormer	SERF	ASL	MLM + SupCon-SB	61.60†	63.72†
Effic.NetV2	RealFormer	SERF	ASL	MLM + SupCon-SB	62.80†*	64.32†*

† represents a model where the batch size was set to 48 (vs 16 in the rest), and * represents a model where the patience was set to 80.

5.2.1 Reproducibility Results

Results were not met using the default hyper-parameters present in the original source code, resulting in a lower performance compared to the results reported in (Khare et al., 2021). Table 5.1 shows these results, where the 4-th row is a reproduction with these hyper-parameters, obtaining 58.80% in overall accuracy and 60.74% in overall BLEU. It can be seen that using the loss in the validation split as stop condition in training leads to poor performance. This can be explained by the high over-fitting of the model and the mismatch of answers between training and test splits. Using a larger batch size or training for more epochs (i.e., using a higher value for patience) helps to increase the performance, reaching 59.80% in overall accuracy and 61.59% in overall BLEU with batch size 48 and patience value of 40. However, the original results of MMBERT with 62.40% in overall accuracy and 64.20% in overall BLEU were never met. Although the MMBERT paper does not mention the values for the aforementioned hyper-parameters, the authors claim to be using a NVIDIA RTX 2080Ti GPU with 11 GB of memory, and hence the batch size used in their tests was likely less than 48.

Table 5.3: Results per category on the VQA-MED 2019 dataset, for two different configurations. MMBERT represents the results for the original architecture and default hyper-parameters. MMBERT-SupCon-SB†* is detailed in the last row of Table 5.2.

Method	Modality		Plane		Organ		Abnormality		Binary		Overal	
	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU	Acc.	BLEU
MMBERT	70.8	77.2	80.8	80.8	68.8	71.8	0.9	2.4	81.3	81.3	58.2	60.2
MMBERT-SupCon-SB†*	76.4	81.2	80.8	80.8	72.0	74.8	13.2	13.8	82.8	82.8	62.8	64.3

5.2.2 Results for the Proposed Extensions

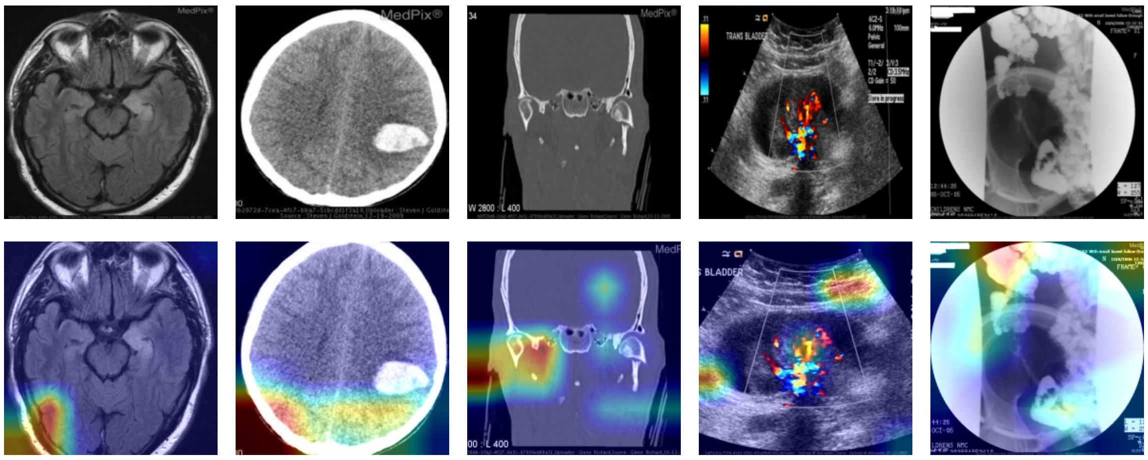
Table 5.2 shows results for the proposed extensions, also assessing the performance of different pre-training tasks based on contrastive learning. The results are presented in an incremental manner, firstly changing the image encoder, the architecture of the multi-modal Transformer, the activation function used, loss for fine-tuning, and pre-training task. The results reported are measured by the overall accuracy and BLEU across categories.

The first row corresponds to the reproduction of the architecture with the original parameters of the source code, achieving a baseline result of 58.80% in overall accuracy and 60.74% in overall BLEU. It can be seen that while maintaining the rest of the architecture as the original, the use of a stronger encoder for the images, as the EfficientNetV2, improves performance ($+0.60\%$ and $+0.62\%$ accuracy and BLEU, respectively). On top of it, the use of RealFormer as multi-modal backbone slightly decreased the accuracy (-0.12%) but improved BLEU ($+0.16\%$). The use of SERF activation function also helped to further performance in both metrics, $+0.80\%$ in accuracy and $+0.87\%$ in BLEU. The use of the asymetric loss did not help to improve the results (-0.20% in accuracy, -0.84% in BLEU).

Regarding the pre-training task, compared to a model with the same architecture but employing only the MLM objective, the use of contrastive learning with SimCLR in addition with the MLM objective does not benefit performance. A reason that this might happen is because SimCLR only approximates representations from the same original sample of image+caption, while the rest are pulled apart, which might not be useful if there are other similar data points in the batch, either because of their image contents or the content of their captions.

Using contrastive supervised loss with Jaccard similarity (SupCon-J) for the captions improves the performance compared to using SimCLR ($+0.40\%$ in accuracy and $+1.00\%$ in BLEU, comparatively), which shows the usefulness of also leveraging the information of the captions. Using instead cosine similarity for the embeddings of sentence transformers (SupCon-SB) furthers the results, due to the increased capacity for measuring similarity between the captions compared to a shallow method as Jaccard ($+0.40\%$ and $+0.48\%$ in accuracy and BLEU, respectively).

All but the last two rows in Table 5.2 used the same hyper-parameters given in the original source code. The performance of a model pre-trained with the same training procedure as SupCon-SB was assessed with higher values for batch size and patience. As suggested by various studies on contrastive learning (Chen et al., 2020), increasing the batch size leads to a higher performance, as did training the model with more epochs. Using a higher batch size increased the performance by $+1.00\%$ in accuracy and $+0.74\%$ in BLEU. This results are further improved by using a higher value for patience and thus training the model for a higher number of epochs, achieving the best score of 62.80% in accuracy and 64.32% in BLEU. This results improve our reproducibility results of the original MMBERT with $+4.00\%$ in accuracy and $+3.58\%$ in BLEU. It also improves the results originally reported in (Khare et al., 2021) of



C: Organ	**C**: Plane	**C**: Binary	**C**: Modality	**C**: Abnormality
Q: What organ is this image of?	**Q**: What is the plane of the image?	**Q**: Is this an mri image?	**Q**: What imaging method was used?	**Q**: What is the primary abnormality in this image?
GT: skull and contents	**GT**: axial	**GT**: no	**GT**: us-d - doppler ultrasound	**GT**: crohn's disease
O: **skull and contents**	**O**: **axial**	**O**: **no**	**O**: **us - ultrasound**	**O**: **malrotation**
B: **skull and contents**	**B**: **axial**	**B**: **no**	**B**: **us - ultrasound**	**B**: **hiatal hernia**

Figure 5.1: Graphical depiction of Grad-cam activation maps. **B** corresponds to MMBERT, GT to ground-truth, and **O** corresponds to the model marked with † in Table 5.2.

62.4% and 64.20% in BLEU, respectively, by $+0.40\%$ in accuracy and $+0.12\%$ in BLEU.

Table 5.3 presents results for different question categories. Our reproduction of the original MMBERT architecture had poor results in the abnormality category, while getting reasonably high results on the other categories. The full model is able to improve results over all categories, and approximately by 13 times in the abnormality category, although this result is still significantly lower compared to the others, due to the nature of the questions being open-ended. The categories that the model achieves the best performance are modality (76.4% in accuracy and 81.2% in BLEU) and binary (82.8% in accuracy and in BLEU).

5.3 Qualitative Analysis

I

Figure 5.1 illustrates the results obtained with the original MMBERT and with our extensions. The top row shows the original images, and the second row shows the results of a method named gradient-weighted class activation map (Grad-CAM) (Selvaraju et al., 2019) to assess the contribution of different image regions. The gradients of the final EfficientNetV2 block were used to construct the localization map emphasizing the important regions for a specific classification. In the organ, plane, and binary categories, the model attends the bony and tissue parts surrounding the skull, as well as the brain, to produce an answer. For the abnormality example, both methods fail to retrieve the correct answer. Still, our method provides an answer with an issue related to the organ of the ground truth (intestines), while the answer of the simpler model is related with stomach issues. These visualization methods can be useful for medical experts in their diagnostics and for an in-domain evaluation of the models.

5.4 Summary

This chapter detailed the experiments performed for the evaluation of the proposed multi-modal Transformer architecture, extending on MMBERT (Khare et al., 2021). First, a description of the datasets used in the VQA-MED task at ImageCLEF2019, and ROCO dataset, was provided. A description of the hyper-parameters used in the tests, such as the learning rate and batch size, was also detailed in this section. Second, the evaluation metrics used were presented and defined. Third, the experimental results were reported. The reproduction of the results reported for the original model was met with some difficulties, given that the default hyper-parameters present in the original source code resulted in a lower performance. Results with models using different architectures and techniques were reported, which it can be concluded that the full model achieved the best performance with 62.80% and 64.32% overall accuracy and BLEU, respectively. Finally, a qualitative analysis is provided using visualization methods based on visual attention to allow for interpretability to the obtained results.

6

Conclusion

Contents

The use of multi-modal Transformer encoders was studied to tackle the VQA-MED Task of ImageCLEF. These Transformer encoders are able to unify vision and language producing cross-modal representations. In this work, a reproducibility study was performed on MMBERT and different extensions over it were proposed. This chapter summarizes the main conclusions of this dissertation and provides some prospects for future work in medical Visual Question Anwering.

6.1 Conclusions

This work reports a reproducibility study of a recent multimodal Transformer model for visual question answering in the medical domain, leveraging pre-training with in-domain data, developed by Khare et al. (2021). The originally reported results were not met, which indicates the existence of an additional preprocessing method that was not reported both in the written report and source code. The performance of the original MMBERT can change considerably according to hyper-parameter choices. In particular, increasing the batch size and the value of patience for the model to wait for early stopping, and thus train longer, provides better results. Given the original hyper-parameters in the source code, we were able to achieve 58.8% and 60.74% overall accuracy and BLEU, respectively, compared to the 62.4% and 64.2% originally reported. With a batch size of 48 and using a patience value of 40, the results were increased to 59.8% and 60.74% overall accuracy and BLEU, respectively.

A number of possible extensions to the original model was assessed, showing that they can lead to improvements. A study of the impact on the performance of each of these extensions was studied in an incremental manner, starting from the image encoder, multi-modal Transformer architecture, activation function, loss for fine-tuning, and the pre-training task. Each of these methods were recently introduced in the last 2 years, reporting better results compared to their counterparts in different tasks, and so were considered as possible extensions to further the performance in VQA task when combined, to achieve better results over the original MMBERT. Following this incremental setup, it can be concluded that improving the image encoder with EfficientNetV2 ($+0.60\%$ in accuracy and $+0.62\%$ BLEU), the use of SERF activation function ($+0.80\%$ in accuracy and $+0.87\%$ BLEU) and introducing supervised contrative learning as a pre-training task helped the most to further the performance of the model. RealFormer helped to improve a bit the performance on BLEU ($+0.16\%$) while slightly decreasing accuracy (-0.12%). The asymmetric loss did not improve the results (-0.20% in accuracy, -0.84% in BLEU). Regarding contrastive learning, it can be concluded that SimCLR did not improve from just using masked language modeling but using supervised contrastive learning helped. Also, it can be concluded that a correct and strong similarity between the captions of the images plays an important role for good results for the multi-modal supervised contrastive loss. From a setup with masked language modeling as pre-training task, a similarity score based on Jaccard increased the performance by $+0.40\%$ in accuracy and $+0.95\%$

in BLEU, while using the cosine similarity of the encodings of sentence-BERT models provided better results than with $+0.80\%$ in accuracy and $+1.43\%$ in BLEU. This performance was further improved by using an larger batch size of 48 and patience value of 80 and thus training for a higher number of epochs, achieving a final result of 62.80% in accuracy and 64.32% in BLEU.

Thus, this dissertation provides a final model using (a) an EfficientNetv2 as image encoder, (b) RealFormer as multi-modal Transformer architecture, (c) a combination of pre-training tasks consisting of a masked language modeling with supervised contrastive learning and (d) fine-tuned with assymetric loss function that addresses class imbalacement, obtaining a final result of 62.80% and 64.32% overall accuracy and BLEU, respectively.

This dissertation complements the findings of the original MMBERT (Khare et al., 2021) in demonstrating that multi-modal Transformers are capable of being used in Visual Question Answering, in particular for the medical domain, and, in general, that Transformers are an adequate method to unify vision and language. The performance of the multi-modal Transformer also benefits from masked language modeling and contrastive learning as pre-training tasks, by using strong image encoder such as EfficientNetV2 and stable activation function as SERF.

In methods regarding the medical domain, it is important to provide techniques for the interpretability of the model. A method based on Grad-Cam was provided, which can allow for a thorough evaluation by medical experts and produce a justification on the results obtained.

6.2 Future Work

For future work, we can consider additional extensions and/or model changes despite the interesting results. These include other methods for visual token extraction: in this method, the outputs at different depths of a EfficientNetV2 were used as visual tokens with different resolutions. Other methods can be used, such as dividing the images into patches and with this setup introduce new pre-training tasks such as masked image modeling (Bao et al., 2021). Other Transformer extensions can also be considered such as the architecture of the multi-modal Transformer encoder, for e.g., Shleifer et al. (2021) proposes an architecture named NormFormer, which addresses gradient magnitude mismatch where gradients at early layers are much larger than at later layers.

When combining the pre-training tasks of masked language modeling and contrastive learning, an addition between the losses of the two pre-training tasks is performed, but the use of an weighted sum could also be explored. Also, when the model is used for Visual Question Answering, the answer is obtained through a classification objective but a generative procedure can also be considered to generate the answer on top of the proposed model, which might obtain interesting results.

Bibliography

Yash Khare, Viraj Bagal, Minesh Mathew, Adithi Devi, U Deva Priyakumar, and C V Jawahar. MMBERT: Multimodal BERT Pretraining for Improved Medical VQA. In *2021 IEEE 18th International Symposium on Biomedical Imaging (ISBI)*, pages 1033–1036, 2021.

Y Lecun, L Bottou, Y Bengio, and P Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Thomas Scialom, Patrick Bordes, Paul-Alexis Dray, Jacopo Staiano, and Patrick Gallinari. BERT Can See Out of the Box: On the Cross-modal Transferability of Text Representations, 2020.

Yuhao Zhang, Hang Jiang, Yasuhide Miura, Christopher D Manning, and Curtis P Langlotz. Contrastive Learning of Medical Visual Representations from Paired Images and Text. *arXiv:2010.00747*, 2020.

Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. MobileNetV2: Inverted Residuals and Linear Bottlenecks. *arXiv:1801.04381*, 2019.

Suyog Gupta and Berkin Akin. Accelerator-aware Neural Network Design using AutoML. *arXiv:2003.02838*, 2020.

Mingxing Tan and Quoc V Le. EfficientNetV2 Smaller Models and Faster Training. *arXiv:2104.00298*, 2021.

Ruining He, Anirudh Ravula, Bhargav Kanagal, and Joshua Ainslie. RealFormer: Transformer Likes Residual Attention. *arXiv:2012.11747*, 2020.

Asma Ben Abacha, Vivek V Datla, Sadid A Hasan, Dina Demner-Fushman, and Henning Müller. Overview of the VQA-MED Task at ImageCLEF 2020: Visual Question Answering and Generation in the Medical Domain. In *Proceedings of the Cross Language Evaluation Forum*, 2020.

Asma Ben Abacha, Sadid A Hasan, Vivek V Datla, Joey Liu, Dina Demner-Fushman, and Henning Müller. VQA-MED: Overview of the Medical Visual Question Answering Task at ImageCLEF 2019. In *Proceedings of the Cross Language Evaluation Forum*, 2019.

Obioma Pelka, Sven Koitka, Johannes Rückert, Felix Nensa, and Christoph M Friedrich. Radiology Objects in COntext (ROCO): A Multimodal Image Dataset. In Danail Stoyanov, Zeike Taylor, Simone Balocco, Raphael Sznitman, Anne Martel, Lena Maier-Hein, Luc Duong, Guillaume Zahnd, Stefanie Demirci, Shadi Albarqouni, Su-Lin Lee, Stefano Moriconi, Veronika Cheplygina, Diana Mateus, Emanuele Trucco, Eric Granger, and Pierre Jannin, editors, *Intravascular Imaging and Computer Assisted Stenting and Large-Scale Annotation of Biomedical Data and Expert Label Synthesis*, pages 180–189. Springer International Publishing, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need. *arXiv:1706.03762*, 2017.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised Contrastive Learning. *arXiv:2004.11362*, 2021.

Emanuel Ben-Baruch, Tal Ridnik, Nadav Zamir, Asaf Noy, Itamar Friedman, Matan Protter, and Lihi Zelnik-Manor. Asymmetric Loss For Multi-Label Classification. *arXiv:2009.14119*, 2021.

Ramprasaath R Selvaraju, Michael Cogswell, Abhishek Das, Ramakrishna Vedantam, Devi Parikh, and Dhruv Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. *arXiv:1610.02391*, 2019.

Frank Rosenblatt. *The Perceptron, a Perceiving and Recognizing Automaton Project Para*. Cornell Aeronautical Laboratory, 1957.

Moshe Leshno, Vladimir Ya Lin, Allan Pinkus, and Shimon Schocken. Multilayer Feedforward Networks With a Nonpolynomial Activation Function Can Approximate Any Function. *Neural Networks*, 6:861–867, 1993.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Sergey Ioffe and Christian Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv:1502.03167*, 2015.

Sebastian Ruder. An overview of gradient descent optimization algorithms. *arXiv:1609.04747*, 2017.

Andrew Y Ng. Feature Selection, L1 vs. L2 Regularization, and Rotational Invariance. In *Proceedings of the International Conference on Machine Learning*, 2004.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted Boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, 2010.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep Residual Learning for Image Recognition. *arXiv:1512.03385*, 2015.

Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q Weinberger. Densely Connected Convolutional Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

Sepp Hochreiter and Jürgen Schmidhuber. Long Short-term Memory. *Neural computation*, 9:1735–1780, 1997.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. *arXiv:1802.05365*, 2018.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781*, 2013.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to Sequence Learning with Neural Networks. *arXiv:1409.3215*, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv:1810.04805*, 2019.

Benoit Jacob, Skirmantas Kligys, Bo Chen, Menglong Zhu, Matthew Tang, Andrew Howard, Hartwig Adam, and Dmitry Kalenichenko. Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference. *arXiv:1712.05877*, 2017.

Song Han, Jeff Pool, John Tran, and William J Dally. Learning both Weights and Connections for Efficient Neural Networks. *arXiv:1506.02626*, 2015.

Cristian Bucilua, Rich Caruana, and Alexandru Niculescu-Mizil. Model compression. In *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2006.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531*, 2015.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *arXiv:1910.01108*, 2020.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *arXiv:1606.05250*, 2016.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know What You Don't Know: Unanswerable Questions for SQuAD. *arXiv:1806.03822*, 2018.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A Lite BERT for Self-supervised Learning of Language Representations. *arXiv:1909.11942*, 2020.

Yash Srivastava, Vaishnav Murali, Shiv Ram Dubey, and Snehasis Mukherjee. Visual Question Answering using Deep Learning: A Survey and Performance Analysis. *arXiv:1909.01860*, 2020.

Xin Yan, L Li, Chulin Xie, Jun Xiao, and Lin Gu. Zhejiang University at ImageCLEF 2019 Visual Question Answering in the Medical Domain. In *Proceedings of the Cross Language Evaluation Forum*, 2019.

Aishwarya Agrawal, Jiasen Lu, Stanislaw Antol, Margaret Mitchell, C Lawrence Zitnick, Dhruv Batra, and Devi Parikh. VQA: Visual Question Answering. *arXiv:1505.00468*, 2016.

Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C Lawrence Zitnick, and Piotr Dollár. Microsoft COCO: Common Objects in Context. *arXiv:1405.0312*, 2015.

Ilija Ilievski, Shuicheng Yan, and Jiashi Feng. A Focused Dynamic Attention Model for Visual Question Answering. *arXiv:1604.01485*, 2016.

C Lawrence Zitnick and Piotr Dollár. Edge Boxes: Locating Object Proposals from Edges. In *Proceedings of the European Conference on Computer Vision*, 2014.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked Attention Networks for Image Question Answering. *arXiv:1511.02274*, 2016.

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *arXiv:1506.01497*, 2016.

Zhou Yu, Jun Yu, Jianping Fan, and Dacheng Tao. Multi-modal Factorized Bilinear Pooling with Co-Attention Learning for Visual Question Answering. *arXiv:1708.01471*, 2017.

Vu Hoang Minh, Raphael Sznitman, Tufve Nyholm, and Tommy Löfstedt. Ensemble of Streamlined Bilinear Visual Question Answering Models for the ImageCLEF 2019 Challenge in the Medical Domain. In *Proceedings of the Cross Language Evaluation Forum*, 2019.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. Skip-Thought Vectors. *arXiv:1506.06726*, 2015.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online, 2020. Association for Computational Linguistics.

Jin-Hwa Kim, Kyoung-Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang. Hadamard Product for Low-rank Bilinear Pooling. *arXiv:1610.04325*, 2017.

Hedi Ben-younes, Rémi Cadene, Matthieu Cord, and Nicolas Thome. MUTAN: Multimodal Tucker Fusion for Visual Question Answering. *arXiv:1705.06676*, 2017.

Zhibin Liao, Qi Wu, Chunhua Shen, Anton Van Den Hengel, and Johan Verjans. AIML at VQA-Med 2020: Knowledge Inference via a Skeleton-based Sentence Mapping Approach for Medical Domain Visual Question Answering. In *Proceedings of the Cross Language Evaluation Forum*, 2020.

Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. Multimodal Pretraining Unmasked: Unifying the Vision and Language BERTs. *arXiv:2011.15124*, 2020.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. UNITER: UNiversal Image-TExt Representation Learning. *arXiv:1909.11740*, 2019.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *arXiv:1609.08144*, 2016.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. *arXiv:1908.02265*, 2019.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. VisualBERT: A Simple and Performant Baseline for Vision and Language. *arXiv:1908.03557*, 2019a.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. *arXiv:1908.08530*, 2019.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, Daxin Jiang, and Ming Zhou. Unicoder-VL: A Universal Encoder for Vision and Language by Cross-modal Pre-training. *arXiv:1908.06066*, 2019b.

Hao Tan and Mohit Bansal. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. *arXiv:1908.07490*, 2019.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A Simple Framework for Contrastive Learning of Visual Representations. *arXiv:2002.05709*, 2020.

Mingxing Tan and Quoc V Le. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks, 2020.

Jie Hu, Li Shen, Samuel Albanie, Gang Sun, and Enhua Wu. Squeeze-and-Excitation Networks. *arXiv:1709.01507*, 2019.

Sayan Nag and Mayukh Bhattacharyya. SERF: Towards better training of deep neural networks using log-Softplus ERror activation Function. *arXiv:2108.09598*, 2021.

Nils Reimers and Iryna Gurevych. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. *arXiv:1908.10084*, 2019.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, 2002.

Hangbo Bao, Li Dong, and Furu Wei. BEiT: BERT Pre-Training of Image Transformers. *arXiv:2106.08254*, 2021.

Sam Shleifer, Jason Weston, and Myle Ott. NormFormer: Improved Transformer Pretraining with Extra Normalization. *arXiv:2110.09456*, 2021.