Multi-modal representation learning towards visual reasoning

Hedi Ben-Younes

THÈSE DE DOCTORAT DE
SORBONNE UNIVERSITÉ

Spécialité
Informatique
École Doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par

Hedi Ben-younes

Pour obtenir le grade de
DOCTEUR de SORBONNE UNIVERSITÉ

Sujet de la thèse :

Multi-modal representation learning towards visual reasoning

Apprentissage de représentation multi-modale et raisonnement visuel

soutenue le 20 mai 2019, devant le jury composé de :

M. Jakob Verbeek	INRIA Grenoble	Rapporteur
M. Christian Wolf	INSA de Lyon	Rapporteur
M. Vittorio Ferrari	Google AI – University of Edinburgh	Examinateur
M. Yann LeCun	Facebook AI Research – NYU	Examinateur
M. Patrick Pérez	Valeo AI	Examinateur
Mme Laure Soulier	Sorbonne Université – LIP6	Examinatrice
M. Nicolas Thome	CNAM – CEDRIC	Co-directeur de thèse
M. Matthieu Cord	Sorbonne Université – LIP6	Directeur de thèse

ABSTRACT

The quantity of images that populate the Internet is dramatically increasing. It becomes of critical importance to develop the technology for a precise and automatic understanding of visual contents. As image recognition systems are becoming more and more relevant, researchers in artificial intelligence now seek for the next generation vision systems that can perform high-level scene understanding.

In this thesis, we are interested in Visual Question Answering (VQA), which consists in building models that answer any natural language question about any image. Because of its nature and complexity, VQA is often considered as a proxy for visual reasoning. Classically, VQA architectures are designed as trainable systems that are provided with images, questions about them and their answers. To tackle this problem, typical approaches involve modern Deep Learning (DL) techniques. In the first part, we focus on developping multimodal fusion strategies to model the interactions between image and question representations. More specifically, we explore bilinear fusion models and exploit concepts from tensor analysis to provide tractable and expressive factorizations of parameters. These fusion mechanisms are studied under the widely used visual attention framework: the answer to the question is provided by focusing only on the relevant image regions. In the last part, we move away from the attention mechanism and build a more advanced scene understanding architecture where we consider objects and their spatial and semantic relations. All models are thoroughly experimentally evaluated on standard datasets and the results are competitive with the literature.

RÉSUMÉ

La quantité d'images présentes sur internet augmente considérablement, et il est nécessaire de développer des techniques permettant le traitement automatique de ces contenus. Alors que les méthodes de reconnaissance visuelle sont de plus en plus évoluées, la communauté scientifique s'intéresse désormais à des systèmes aux capacités de raisonnement plus poussées.

Dans cette thèse, nous nous intéressons au Visual Question Answering (VQA), qui consiste en la conception de systèmes capables de répondre à une question portant sur une image. Classiquement, ces architectures sont conçues comme des systèmes d'apprentissage automatique auxquels on fournit des images, des questions et leur réponse. Ce problème difficile est habituellement abordé par des techniques d'apprentissage profond. Dans la première partie de cette thèse, nous développons des stratégies de fusion multimodales permettant de modéliser des interactions entre les représentations d'image et de question. Nous explorons des techniques de fusion bilinéaire, et assurons l'expressivité et la simplicité des modèles en utilisant des techniques de factorisation tensorielle. Dans la seconde partie, on s'intéresse au raisonnement visuel qui encapsule ces fusions. Après avoir présenté les schémas classiques d'attention visuelle, nous proposons une architecture plus avancée qui considère les objets ainsi que leurs relations mutuelles. Tous les modèles sont expérimentalement évalués sur des jeux de données standards et obtiennent des résultats compétitifs avec ceux de la littérature.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

A-MPT	Arithmetic Mean of Per-Type accuracies
A-NMPT	Arithmetic Normalized Mean of Per-Type accuracies
AI	Artificial Intelligence
BAN	Bilinear Attention Network
CCA	Canonical Correlation Analysis
CLEVR	Compositional Language and Elementary Visual Reasoning
ConvNet	Convolutional Neural Network
CRF	Conditional Random Field
CP	Candecomp/Parafac
CV	Computer Vision
D-NMN	Dynamic Neural Module Network
DL	Deep Learning
DNN	Deep Neural Network
EEG	Electroencephalography
FCN	Fully Convolutional Network
FFT	Fast Fourier Transform
FiLM	Featurewise Linear Modulation
GAN	Generative Adversarial Network
GPU	Graphics Processing Unit
GRU	Gated Recurrent Unit
H-MPT	Harmonic Mean of Per-Type accuracies
H-NMPT	Harmonic Normalized Mean of Per-Type accuracies
HAN	Hard Attention Network
ILSVRC	ImageNet Large Scale Visual Recognition Challenge
IoU	Intersection over Union
LSTM	Long-Short Term Memory
MCB	Multimodal Compact Bilinear
MFB	Multi-modal Factorized Bilinear
MFH	Multi-modal Factorized Higher-order
MLB	Multimodal Low-rank Bilinear

MLP	Multi-Layer Perceptron
ML	Machine Learning
NMN	Neural Module Network
NMS	Non-Maximum Suppression
OE	Open-Ended
RAU	Recurrent Answering Units
ReLU	Rectified Linear Unit
RN	Relation Network
RNN	Recurrent Neural Network
RPN	Region Proposal Network
RoI	Region of Interest
SAN	Stacked Attention Network
SGD	Stochastic Gradient Descent
VRD	Visual Relationship Detection
VQA	Visual Question Answering

CHAPTER 1

INTRODUCTION

Contents

1.1 Context

Over the past decade, the field of Artificial Intelligence (AI) has experienced a growing interest. Machines are provided with more and more cognitive capacities, enabling applications like speech recognition, automatic translation or image understanding. These technological advances may have huge societal impacts on fields such as mobility (autonomous driving), health (automatic diagnosis), consumption (digital marketing), or even social interactions.

One of the most important field in AI is Computer Vision (CV), which aims at automatically understanding the visual content of images or videos. Until the 2010's, a first mandatory step to numerous CV tasks was the calculation of visual features that describe the image. Based on strong mathematical modeling, they are handcrafted to be invariant to several image transformations such as scale or lighting variations.

At the annual ImageNet classification challenge held in 2012, these traditional techniques have been outperformed by Deep Learning (DL) methods (Figure 1.1). Given enough training data, deep networks can learn a meaningful representation of the image together with the classification. Since then, DL-based models have shown outstanding results in various complex vision tasks such as object detection, semantic segmentation, image colorisation or even 3d shapes generation.

This opens the field for even more challenging applications involving multiple modalities, where deep networks can be designed to process images and texts. At the core of these multi-modal systems lies the question of modeling interactions between different sources, and more generally of automatically reasoning about the visual data. These complex problems are tackled by both academic and

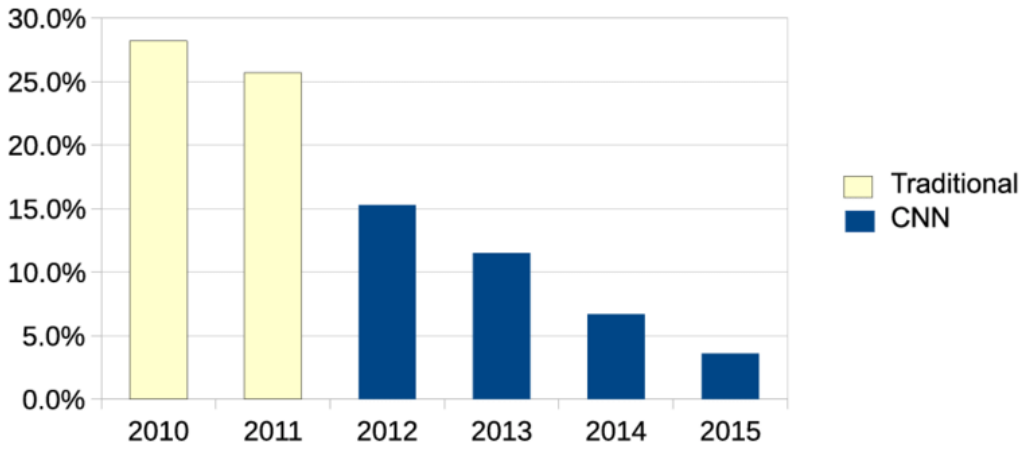


Figure 1.1 – **Historic error on ImageNet.** This plot shows the *top5* error obtained by the annual winner of ImageNet image classification challenge. A significant performance gap was achieved by a DL architecture (a ConvNet) in 2012 with respect to traditional CV approaches. The illustration was taken from (Bottou et al. 2016).

industrial actors, which makes this research domain a rich and extremely active field.

1.1.1 Joint image/text understanding

In the last decade, an increasing quantity of multi-modal data has become available, through web platforms such as e-commerce websites or social media. A lot of work has been done by the Machine Learning (ML) community to develop theoretical and practical principles that help analyse such heterogeneous content. In particular, studying the interactions between images and free-form texts has seen a growing interest.

One specific problem that appears in such setups is the multi-modal alignment. Given a corpus of visual data where each image is associated to a textual description of its content, multi-modal alignment aims at finding a common representation for both modalities. In this so-called multi-modal space, images and texts are comparable, and it is possible to obtain a quantitive measure of how similar an image and a text are. Early work for aligning image and text representation spaces are based on the statistical model of Canonical Correlation Analysis (CCA) (Hotelling 1936; Hardoon et al. 2004). Images and texts are seen as multivariate random variables, and CCA estimates a set of projections that maximize the correlation between true image/text pairs. Later on, CCA has been integrated into the DL framework in (F. Yan et al. 2015), where the Deep CCA is designed to be compliant with the efficient training of DL systems. Another line

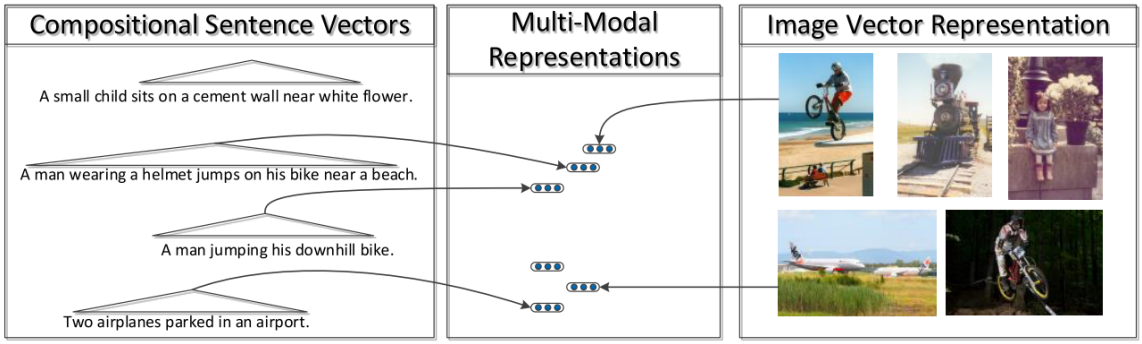


Figure 1.2 – **Illustration of a multi-modal representation space.** Image and sentence vectors are projected into the same space through the learning of a multi-modal metric. In this space, an image and its description are brought close together, and unrelated image/text pairs are separated. The illustration was taken from (Socher et al. 2014).

of work chooses to cast multi-modal alignment as a metric learning problem. In (Socher et al. 2014), a specific loss function called the *triplet loss* learns image and texts projections into a common vector space. This loss is used to bring closer together images and their textual descriptions, while pulling away unrelated image/text pairs. Interestingly, the language embedding is computed by a special type of neural network that operates over the text's Semantic Dependency Tree (see Figure 1.2). The same type of triplet loss is used in (Kiros et al. 2015), with a less explicitly structured language representation (the Skip-thought encoding, presented in Section 2.2.2).

Creating these common representation spaces between images and texts has opened the field for more challenging applications, such as automatic image captioning. It consists in learning to automatically generate a linguistically correct sentence that describes an image. Given a large corpus where each image is described by a sentence, a language model learns to maximize the likelihood of the description conditionned on the image. Learning to estimate this probability is often done in an autoregressive fashion, where the model tries to predict the next word given all previous words and the image that is passed as input (Kiros et al. 2014a). This caption generation task can even benefit from having a common multi-modal representation space, as it has been shown in (Kiros et al. 2014b) (see Figure 1.3). More ambitious, the reverse problem of generating images from textual descriptions has also been tackled, using the recently introduced Generative Adversarial Network (GAN) framework (Goodfellow et al. 2014; Reed et al. 2016; H. Zhang et al. 2017b).

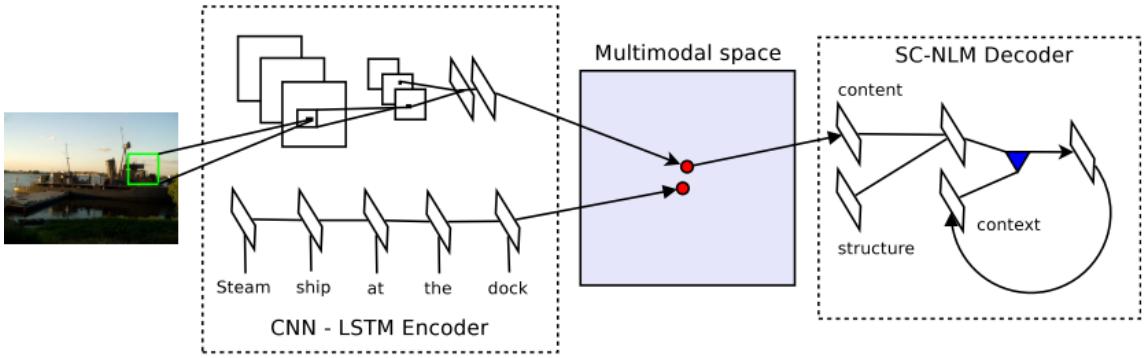


Figure 1.3 – **Illustration of a caption generation.** An image and its associated description are projected into the same representation space. Then, any vector from this space can be passed as the input of a neural language model that generates a sentence, one word at a time. The illustration was taken from (Kiros et al. 2014b).

1.1.2 Visual Question Answering

This trend of interlinking image and text modalities within a joint learning process, whether for alignment or data generation, has opened the perspective on deeper image and language understanding problems for ML researchers. In this thesis, we focus our efforts on what is among the most challenging vision and language applications: Visual Question Answering (VQA). It consists in building systems capable of answering any natural language question about any image (see Figure 1.4). VQA involves a high-level understanding of multiple modalities, in a context where neither image nor text can be considered independantly from one another. The system should model the relations between an image and a textual question, and extract meaningful interactions that can help provide an answer. Interestingly, VQA falls within the broader field of human-machine interactions, and is a first step to vision-based interactive systems.

The VQA problem was formulated for the first time, to our knowledge, in (Malinowski et al. 2014a). As it has been noticed later in (Malinowski et al. 2014b; Antol et al. 2015), answering questions about images involves addressing multiple non-trivial issues. The system should be able to understand a large and varied quantity of semantic concepts, from both perceptual and linguistic modalities. Moreover, as the quantity of understood concepts grows, semantic boundaries between some of those concepts may become ambiguous and fuzzy, which should be accounted for in the model. Besides, a VQA model is also expected to have commonsense knowledge about the world. This capacity may be necessary to answer questions about what use to make of an object. Other complex types of questions require high-level scene understanding or visual reasoning capacities (Johnson et al. 2017a). They include object detection, fine-

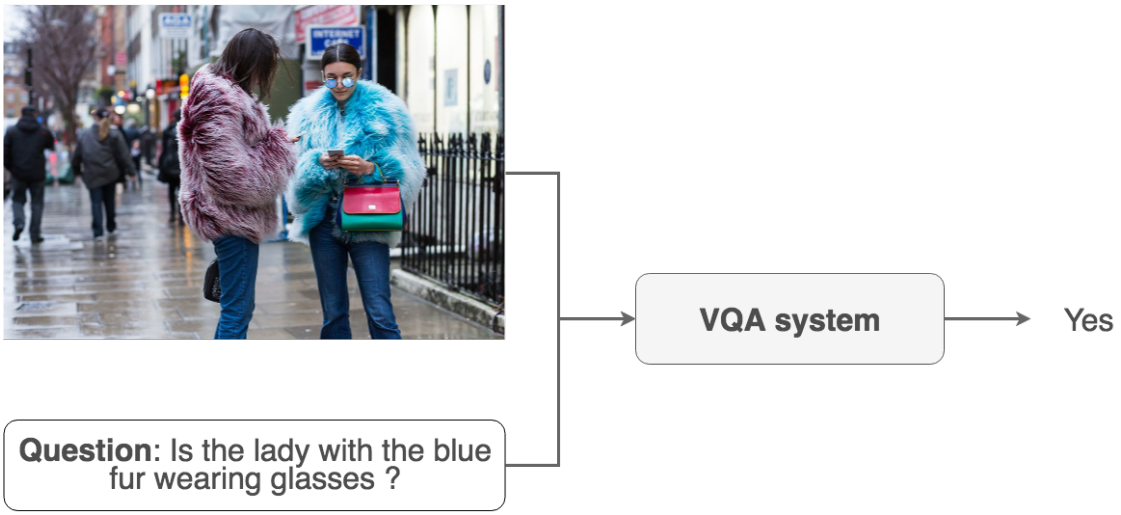


Figure 1.4 – **Illustration of the VQA task.** If a human was asked to answer this question, he would need to go through multiple steps. After precisely understanding the question, he would need to 1) find the ladies; 2) locate each one's fur; 3) find which fur is blue; 4) associate the blue fur with the lady who is wearing it; 5) find out if she is also wearing glasses; and finally 6) answer *yes*.

grain recognition, attribute identification, visual relationship recognition, counting, comparing objects with respect to certain caracteristics, or even performing logical operations. In the example shown in Figure 1.4 the question "*Is the lady with the blue fur wearing glasses?*" calls for these types of visual reasoning capacities. Finally, the problem of quantifying the performance of VQA models is not straightforward. As the goal is to mimic human response, it is necessary to deal with ambiguities, which can stem from many phenomena that are inherent to human judgement. For all these reasons, answering questions about images constitutes a major challenge for researchers in ML, and more generally in AI.

Most of the research conducted in VQA involves DL techniques, for their effectiveness and their ability to leverage large quantities of data. For each modality, representations are provided by powerful models, which may have been previously trained to understand the semantics behind the data they encode (see Section 2.2). For the image representation, a ConvNet provides a vector (or possibly many) that contains information about the image content, the different objects that are depicted in the picture, and the attributes each one carries. As for the question representation, a recurrent model reads the sentence and computes another vector that incorporates information about the words and their contexts. Designing a VQA model consists in finding the structures that are appropriate to understand each modality with respect to the other, learn the relevant interactions between image and question representations, and predict the correct answer.

1.2 Contributions

In this thesis, we tackle the problem of VQA from a DL perspective. We first attack the issue of learning a **multi-modal fusion** module, central to VQA, that merges vectors by extracting their relevant correlations. In particular, we focus our efforts on the powerful solution provided by bilinear models, and study them under a tensor viewpoint. It constitutes the work we present in Chapter 3 and Chapter 4. At a higher level, answering questions about images requires more than simple multi-modal fusions. The different objects, their visual appearance, how they interact with each other, the spatial layout in which they are disposed, *etc.*, should be understood by the model. In Chapter 5, we explore modeling the structure in the representation of the visual scene, and thus mimick some type of **visual reasoning** within the VQA system architecture itself.

This thesis is based on the material published in the following papers:

- Hedi Ben-Younes*, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2017). "MUTAN: Multimodal Tucker Fusion for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*;
- Hedi Ben-Younes, Rémi Cadène, Nicolas Thome, and Matthieu Cord (2019). "BLOCK: Bilinear Superdiagonal Fusion for Visual Question Answering and Visual Relationship Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*;
- Hedi Ben-Younes*, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2019). "MUREL: Multimodal Relational Reasoning for Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*

1.3 Industrial context

This thesis has been realized in collaboration with Heuritech, a french startup specialized in social media analysis for the fashion industry. It develops systems that extract information from user-generated posts, and aggregates this information into interactive dashboards for brands and retailers. Automatic image understanding is at the core of the company's technology, and constitutes its principal research focus.

Most of the social media content that is relevant for the fashion industry is visual. It mainly consists in images, posted by influencers or simple users, that are focused around pieces of garment. Analyzing these pictures involves detecting the fashion-related objects, understanding their nature, finding their attributes (such as color, texture, ...), and even sometimes identifying the exact brand and

model name of the product. DL systems are an appealing approach for their performance, robustness and flexibility. This is why the research effort on visual recognition at Heuritech is mostly turned towards DL.

Even if the visual modality provides the more direct information for fashion-related content, viewing a social media post only as a picture may be restrictive. A caption is often associated to the image, with user-defined hashtags. Other users can show interest through likes and comments. Sometimes meta-data can be retrieved, such as the geo-localisation or the time at which the picture was taken. All these other signals may carry information under the light of which the image content could be understood. For Heuritech, exploring and developing methods for merging several modalities within ML systems is of high interest, and has been studied in the work of this thesis.

CHAPTER 2

RELATED WORKS

Contents

As we stated in the introduction, deep multi-modal representations have been recently developped for numerous purposes. The work of this thesis is focused on Visual Question Answering (VQA), which consists in designing and training Machine Learning (ML) models to answer any free-form question, about any natural image. Architectures for VQA usually follow a generic template, depicted in Figure 2.1. Mono-modal encoders first provide high-level representations of visual and linguistic data. They constitute input modules to the actual VQA system, designed to fuse both modalities and reason about their interactions in order to provide an answer.

Learning to fuse both image and question representations to predict the answer is actually the core of Deep Learning (DL)-based VQA systems. Two functional components can be distinguished from each other, as they operate at different architectural levels. The final system performance heavily depends on these two components, and they constitute the research in VQA models almost exhaustively. First, the **multi-modal fusion** component operates at the vector level. It aims at learning an elementary function that takes two vectors as input and provides

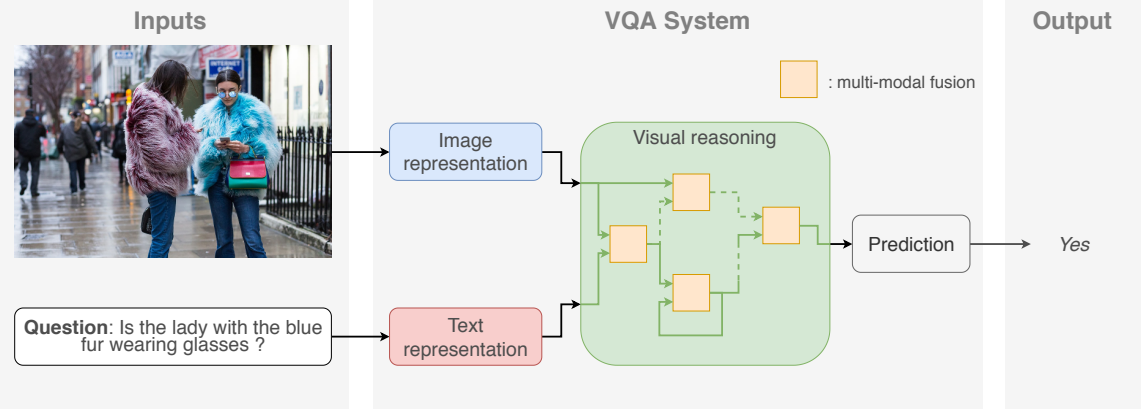


Figure 2.1 – **Big picture of a VQA system.** Two mono-modal understanding systems provide meaningful representations of image (in blue) and question (in red). These two representations are used as inputs to the visual reasoning architecture (in green), whose structure determines the high level capacities of the model. It is based on possibly many multi-modal fusion layers (in orange), each learning to extract the relevant interactions between its input. Finally, a representation of the image/question pair is passed to the output module (in gray) which provides an answer.

an output that extracts the relevant interactions between its inputs. The second important layer is the architectural design itself, which is often referred to as **visual reasoning**. It expresses the high-level capacities of the model, and conveys the inductive biases that the model can exploit using the training data. For instance, some models are designed to focus their attention on a subset of the image before predicting the answer. Others iterate multiple times over the image to refine their internal understanding of the scene, or can benefit from pairwise relations between objects, *etc.*

In this chapter, we review the related works that is relevant to study and build VQA systems. First, in Section 2.1 we give the generic setup for training a VQA architecture. As visual and linguistic representations constitue elementary modules used in all the VQA architectures, we review their design and training in Section 2.2. Then, we relate how the crucial problem of multi-modal fusion for VQA is tackled in Section 2.3. In Section 2.4, we review the different model structures, and how they induce behaviours that are akin to some types of reasoning processes. The different datasets that we use throughout this thesis are presented in Section 2.5. Finally, we expose our contributions and position them with respect to the existing literature in Section 2.6.

2.1 VQA architecture

In Figure 2.1 we show the architecture of a classical VQA system. An image representation (blue rectangle) is provided by a deep visual features extractor that yields one or many vectors. In parallel, a textual representation (red rectangle) is the output of a language model that goes through the question. Then, both these representations are merged (green rectangle), with possibly complex strategies based on multi-modal fusion and high-level reasoning schemes such as iterative processing or visual attention mechanisms. Finally, a prediction module (gray rectangle) provides its estimation of the answer to the question. The modules that compose the VQA system are usually designed to be end-to-end trainable on a dataset set $\mathcal{D} = \{(v_i, q_i, a_i)\}_{i=1:N}$. It contains ground-truth data where the question q_i on the image v_i has answer $a_i \in \mathcal{A}$.

A VQA model can be seen as a parametric function f_Θ that takes (image, question) pairs as input and yields an answer prediction. Using the training data $\mathcal{D}$, we can define an empirical loss function that quantifies how far the predictions of f_Θ are from the true answers:

$$\mathcal{L}(\Theta, \mathcal{D}) = \frac{1}{N} \sum_{i=1}^{N} l\left(f_\Theta\left(v_i, q_i\right), a_i\right) \tag{2.1}$$

where l measures the difference between the model prediction $f_\Theta(v_i, q_i)$ and the ground truth a_i.

As a result of the free-form answer annotation process, answers in $\mathcal{A}$ are possibly composed of multiple words. For this reason, early attempts (Malinowski et al. 2016; Gao et al. 2015) model the answer space as sentences, and learn to sequentially decodes each word of the true answer. However, the most widely adopted framework to represent the answer space is classification. In this setup, the scope of possible answers is fixed, each answer corresponds to a class, and the model computes a probability distribution over the set of classes given an image/question pair. As proposed in (M. Ren et al. 2015; Ma et al. 2016), the classes are obtained by taking the most frequent answers in the training set, regardless of whether they contain one or multiple words.

Following this setup, the VQA model outputs a probability distribution over possible answers $f_\Theta(v, q) \in [0, 1]^{|\mathcal{A}|}$, where each coordinate contains the estimated probability of the corresponding answer. To train the model, we use the cross-entropy loss function defined as:

$$l(f_\Theta(v_i, q_i), a_i) = -\log f_\Theta(v_i, q_i)[a_i] \tag{2.2}$$

The goal of the training stage is to find the parameters $\Theta^\star$ that minimize the empirical loss:

$$\Theta^\star = \underset{\Theta}{\operatorname{argmin}}\, \mathcal{L}(\Theta, \mathcal{D}) \tag{2.3}$$

In order to prevent the network from overfitting the training dataset $\mathcal{D}$, we use an external validation set $\mathcal{V}$ to apply the early-stopping strategy. It constists in learning on the training set until the empirical loss stops decreasing on $\mathcal{V}$. This technique, widely used in DL, acts as a regularizer.

2.2 Mono-modal representations

We aim at building models that are able to answer questions about images. Thus, a VQA model takes as input an image and a question, that are processed by mono-modal modules. The nature of these visual and textual representations will have a direct impact on the design and performance of a VQA system.

2.2.1 Image representation

Since the success of DL methods at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) (Russakovsky et al. 2015) challenge in 2012, these architectures keep improving the state-of-the-art on a large part of all the Computer Vision (CV)-related tasks: image classification, instance and semantic retrieval, semantic segmentation, object detection, and others. At the very basis of DL is the feedforward neural network. This model progressively maps raw inputs (*e.g.* image pixels) to outputs (*e.g.* a distribution over classes) through multiple transformation layers, usually consisting in an affine projection followed by a non-linear activation function. A model that stacks many layers one after the other is referred to as a Deep Neural Network (DNN), and is able to perform highly non-linear complex mappings. The transformation performed by each layer is parametrized, and their optimal value is obtained by minimizing a problem-dependant loss function. A DNN being differentiable, optimizing its parameters is usually done by gradient-based methods such as Stochastic Gradient Descent (SGD). Computing the gradient of the loss function with respect to each parameter of the DNN is almost exclusively done with the backpropagation algorithm (LeCun et al. 1989).

Convolutional Neural Networks (ConvNets) (Fukushima 1980; LeCun et al. 1989; Krizhevsky et al. 2012) are a special kind of neural networks where the linear operation within each layer is a convolution, which makes these architectures well suited to process image data. Indeed, convolutions provide an effective way to share parameters between local feature extractors that go through the whole image, taking into account spatial coherence of the visual content. Similarly to classical DNNs, deep ConvNets stack multiple convolution layers separated by non-linear activation functions such as Rectified Linear Unit (ReLU) (Krizhevsky

et al. 2012). Local pooling operations may also be integrated in the architecture, which make the representations invariant to local perturbations, and provide a control over their spatial size. As these models typically contain a huge number of parameters, large datasets are required if we want to train them. This is why these deep ConvNets are usually trained on ImageNet (Russakovsky et al. 2015), a dataset of 1 million images, each manually assigned to a label from a vocabulary of a thousand classes.

Collecting and labelling data is costly, which may limit the size of available data of a given task. However, a network trained on ImageNet (Russakovsky et al. 2015) has been shown to provide image representations that are generic enough to transfer to other tasks (Razavian et al. 2014; Azizpour et al. 2016). This property makes it possible to pre-train a ConvNet on ImageNet and slightly modify its weights (=*fine-tuning*) to adapt it to a new dataset, for which we have less labelled data.

In many VQA architectures, the image is represented using a pre-trained network as a features extractor. Each image is presented at the input of the network, and the forward pass is computed up until the penultimate layer, which outputs an internal representation that the ConvNet constructs for this image. This vector is used to characterize the image, and will be passed to the question answering system. In early VQA works, this single vector approch was very popular for its simplicity (Malinowski et al. 2016; Gao et al. 2015; M. Ren et al. 2015; Ma et al. 2016; Kim et al. 2016).

Incorporating spatial information. Unfortunately, information about spatial layout is hardly reachable from this single vector approach. Many questions in VQA may involve a fine understanding of the scene, and require to manipulate some spatial concepts such as *on top of*, *left*, *right*, *etc.* This is why modern VQA systems use more than a single vector to represent the image. One fairly simple technique is the Fully Convolutional Network (FCN) approach (Long et al. 2015; He et al. 2016). Instead of yielding a single vector that represents the whole image, an FCN preserves the spatial information throughout the network and provides a set of spatialy-grounded representation vectors organized in a 2-d grid (see the left image in Figure 2.2). All the vectors are computed simultaneously, in a single forward pass. As it has been done in (Long et al. 2015), one can easily transform a regular ConvNet into an FCN by increasing the size of input images and reshaping all the linear projections matrices into 1×1 convolutions. These FCN features have been extensively used in VQA as bags of vectors (Z. Yang et al. 2016; Fukui et al. 2016; Kim et al. 2017; Z. Yu et al. 2017). In rarer cases, the grid structure in the representation is leveraged by the VQA model to increase dependance of the output on the spatial layout of the image (Xiong et al. 2016; Z. Chen et al. 2017).

Since 2017, the standard in visual representations for VQA are the bottom-up features (Anderson et al. 2018). As illustrated in Figure 2.2, the fixed-grid structure

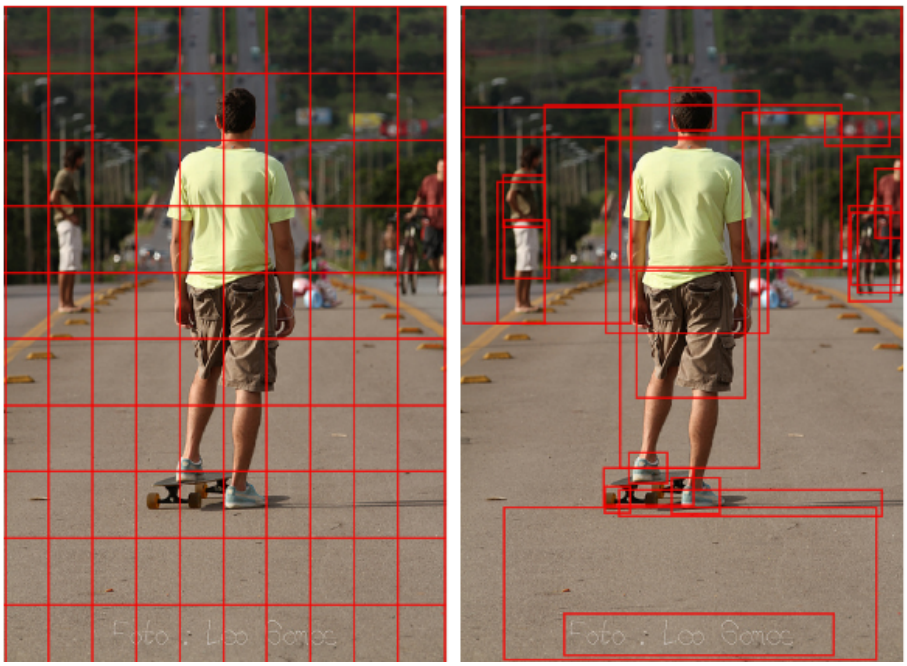


Figure 2.2 – **Bottom-up features** On the left, the fixed-grid illustrates the region that each feature vector in an FCN represents. On the right, each salient object in the image constitues an input region to the VQA system. The illustration was taken from (Anderson et al. 2018).

is replaced by a set of object-focused regions. The bottom-up mechanism, based on Faster-RCNN (S. Ren et al. 2015), proposes image regions and associates each one with a representation vector. This model is trained on a separate dataset to detect objects, predict their class but also their attributes such as a color, a texture, a size, *etc*. Objects are detected in two stages. First, a small network called Region Proposal Network (RPN) is slid over convolutional features at an intermediate level of the ConvNet, and predicts a class-agnostic objectness score for several anchor boxes. After a step of Non-Maximum Suppression (NMS) with Intersection over Union (IoU) threshold, the top boxes are kept to be used as input for the second stage. In this stage, features that correspond to each region are extracted with a method called Region of Interest (RoI) pooling, and these vectors are used to classify each box proposal. In both stages, a refinement of the bounding box coordinates is also learnt. At the time of writing, these features are the ones that provide the best results in VQA (Y. Zhang et al. 2018; Jiang et al. 2018; Kim et al. 2018). Moreover, using these representations is also time-efficient: the typical number of regions per image is 36 for the bottom-up features, whereas it can reach 196 for FCN grids of size 14×14.

In our work, we propose several deep models based on ConvNet representations for their simplicity and compactness, and on FCN features to compare against

leading methods (Chapter 3). As the work on bottom-up features (Anderson et al. 2018) was published in 2017, we use them in our contributions that came after and that are presented in Chapter 4 and Chapter 5.

2.2.2 Textual embedding

In classical VQA systems, a sentence encoder provides an algebraic representation of the question. This representation should encode precise and fine-grain semantic information about questions with variable length. Multiple models exist for such encoders, with different complexity and expressivity. Their choice and design hold a critical position in question understanding, and in the final performance of the VQA model.

To manipulate texts in natural language, we first need to define the atomic linguistic element. We could consider characters, words, bi-grams of words, *etc.* In the context of VQA, the usual atomic linguistic unit is words. Before representing arbitrarily long sentences, we need to define how words can be processed by ML models.

Word representation. The simplest way to represent a word is by its **one-hot encoding**. Given a finite list of words that constitute a vocabulary $\mathcal{W}$, each word w is assigned to an integer index i_w. The one-hot encoding of a word w in the vocabulary $\mathcal{W}$ is a binary vector $\boldsymbol{v}_w$ whose size is the same as $\mathcal{W}$, and where the k-th dimension is defined as follows:

$$\boldsymbol{v}_w[k] = \begin{cases} 1 \text{ if } k = i_w \\ 0 \text{ else} \end{cases} \tag{2.4}$$

This very high-dimensional vector is usually substituted by the more compact **word embeddings**, which provide a learnable representation of words. Each word w is assigned to a vector of parameters $\boldsymbol{x}_w \in \mathbb{R}^d$, referred to as the *embedding of* w. The dimension d is a hyperparameter, whose typical value is between 50 and 500. These vectors are initialized randomly, or using pre-trained models such as Word2Vec (Mikolov et al. 2013) or Glove (Pennington et al. 2014). Depending on the task on which these vectors have been trained, semantic and syntactic properties of words can be captured in their associated embedding. In particular, the euclidean distance between two embedding vectors reflect some type of semantic similarity between their associated words.

Bag of words. One of the simplest ways to represent an arbitrarily long sentence as a fixed-size vector is to view the sentence as a bag of words. The first step is to tokenize the text into a list of elementary language units (in our case, they are words): $q = [w_1, ..., w_T]$. Then the bag of words representation simply averages the word embeddings as follows:

$$\boldsymbol{q} = \frac{1}{T}\sum_{i=1}^{T} \boldsymbol{x}_{w_i} \tag{2.5}$$

This representation has been used in early works of VQA (Antol et al. 2015; H. Xu et al. 2016; Zhou et al. 2015). In (Shih et al. 2016), a variant of this model separates different parts of information by splitting the question into 4 bins: the first bin contains the first two words of the question, the second bin contains the nominal subject, the third is composed of all other noun words, and the last one is made of all the remaining words. Word vectors are averaged within each bin, and all the 4 vectors are concatenated.

Recurrent networks. While these bag of word models are easy to implement, their effectiveness is limited by the fact that word order is not taken into account. More elaborate models are required to learn a fixed-dimensional representation of variable-length sequences. Recurrent Neural Networks (RNNs) (Elman 1990; Bodén 2001) have been developped to model time dependencies in sequences. In particular, they have been used to represent sentences (Mikolov et al. 2010) as they provide order-dependant representations. These networks operate over an input space, *e.g.* the word vectors, and an internal state that summarizes what has been processed by the network so far. Given a sequence of word vectors $[\boldsymbol{x}_1, ..., \boldsymbol{x}_T]$, the RNN iteratively updates its internal hidden state $\boldsymbol{s}$ using a simple transformation:

$$\boldsymbol{h}_t = f(\boldsymbol{W}_{h\to h}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{x\to h}\boldsymbol{x}_t) \tag{2.6}$$

where f is a non-linear activation function. Additionally, an output layer can provide predictions for each timestep $\boldsymbol{y}_t = g(\boldsymbol{W}_{h\to y}\boldsymbol{h}_t)$, where g is a problem-dependant activation function. The parameters of an RNN are trainable end-to-end with backpropagation. The output vectors $[\boldsymbol{y}_1, ..., \boldsymbol{y}_T]$ can be used to calculate a loss with respect to some ground-truth value, or they can also be forwarded to another neural network. In Figure 2.3, we show different possible types of input-output designs summarized in (Karpathy 2015).

In practice, the classical RNN exhibits some problems regarding the propagation of gradients during learning, and seems to be unable to handle long-term dependencies. These phenomena, referred to as vanishing and exploding gradients, have been studied in (Bengio et al. 1994; Pascanu et al. 2013). To circumvent these issues, more elaborate recurrent models have been developped. In particular, the Long-Short Term Memory (LSTM) network was proposed in (Hochreiter et al. 1997a), and made popular by (Greff et al. 2015; Christopher Olah 2015). The core idea of this model is the cell state $\boldsymbol{c}_t$, that stores information from previous timesteps. The network can choose to remove information from the cell, update its value using the input, and output its content towards the hidden state $\boldsymbol{h}_t$. Mathematically, three *gating* operators are computed as functions of the input $\boldsymbol{x}_t$

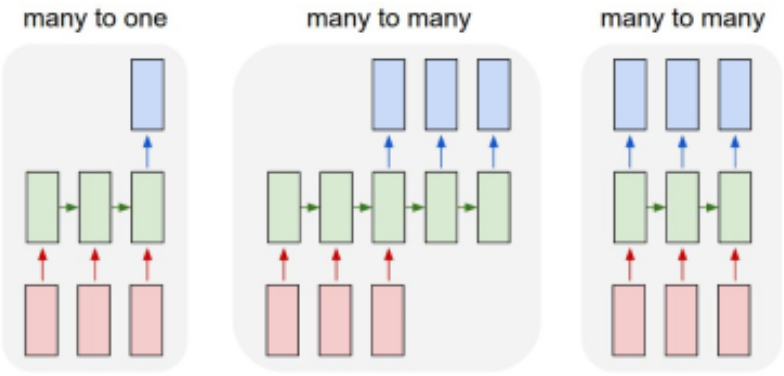


Figure 2.3 – **Input-output designs of RNNs.** On the left, the many-to-one architecture may be used for sentence classification. On the center, the many-to-many scheme is often chosen for neural langage translation models, where it is required to have full understanding of the whole sentence before starting to predict its traduction. Finally on the right, this setup provides a representation for each input token, useful for tasks such as part-of-speech tagging. Source: (Karpathy 2015)

and the previous hidden state $\boldsymbol{h}_{t-1}$: the input gate $\boldsymbol{i}_t$, the forget gate $\boldsymbol{f}_t$ and the output gate $\boldsymbol{o}_t$:

$$\boldsymbol{i}_t = \sigma\left(\boldsymbol{W}_i[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_i\right) \tag{2.7}$$
$$\boldsymbol{f}_t = \sigma\left(\boldsymbol{W}_f[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_f\right) \tag{2.8}$$
$$\boldsymbol{o}_t = \sigma\left(\boldsymbol{W}_o[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_o\right) \tag{2.9}$$

where σ is the sigmoid function, whose output is in $[0, 1]$. The network proposes a cell vector $\tilde{\boldsymbol{c}}_t$ in the form

$$\tilde{\boldsymbol{c}}_t = f\left(\boldsymbol{W}_c[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t] + \boldsymbol{b}_c\right) \tag{2.10}$$

and this vector is used to update the network's cell and compute $\boldsymbol{c}_t$ following the equation

$$\boldsymbol{c}_t = \boldsymbol{f}_t * \boldsymbol{c}_{t-1} + \boldsymbol{i}_t * \tilde{\boldsymbol{c}}_t \tag{2.11}$$

Intuitively, $\boldsymbol{f}_t$ selects the components of $\boldsymbol{c}_{t-1}$ that should be kept ($\boldsymbol{f}_t$ close to 1) and those that should be forgotten ($\boldsymbol{f}_t$ close to 0). Similarly, $\boldsymbol{i}_t$ chooses the components of $\tilde{\boldsymbol{c}}_t$ that should be passed on to $\boldsymbol{c}_t$. Finally, following the same gating mechanism, the internal hidden state $\boldsymbol{h}_t$ is updated by

$$\boldsymbol{h}_t = \boldsymbol{o}_t * \tanh(\boldsymbol{c}_t) \tag{2.12}$$

Other recurrent models have been developped using similar gating processes. Among them, the Gated Recurrent Unit (GRU) (Cho et al. 2014; Chung et al. 2014) is one the most popular, certainly because its performs as well as LSTMs for less

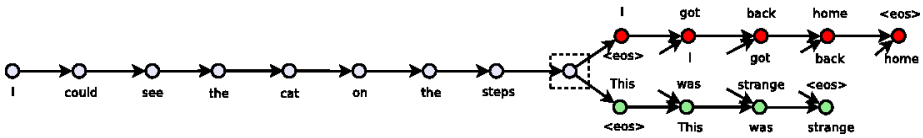


Figure 2.4 – **Skip-thought model.**

parameters. In this simplified model, the cell state $\boldsymbol{c}$ is removed and the input and forget gates are merged into a single update gate. The equations of the GRU are as follows:

$$\boldsymbol{z}_t = \sigma\left(\boldsymbol{W}_z[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]\right) \tag{2.13}$$
$$\boldsymbol{r}_t = \sigma\left(\boldsymbol{W}_r[\boldsymbol{h}_{t-1}, \boldsymbol{x}_t]\right) \tag{2.14}$$
$$\tilde{\boldsymbol{h}}_t = f\left(\boldsymbol{W}_z[\boldsymbol{r}_t * \boldsymbol{h}_{t-1}, \boldsymbol{x}_t]\right) \tag{2.15}$$
$$\boldsymbol{h}_t = \boldsymbol{z}_t * \boldsymbol{h}_{t-1} + (1 - \boldsymbol{z}_t) * \tilde{\boldsymbol{h}}_t \tag{2.16}$$

The vast majority of VQA models use either LSTM or GRU to encode variable length sentences in a vectorial form. They keep only the last hidden state, or they use more elaborate models to aggregate the list of all output vectors, one for each timestep (Z. Yu et al. 2017; Z. Yu et al. 2018).

As we saw in Section 2.2.1, image models can be pre-trained on the ImageNet dataset to learn to extract relevant visual feature vectors. Similar pre-training schemes exist for language models. In particular, the Skip-thought encoder (Kiros et al. 2015) learns the weights of a GRU neural network using a large quantity of unlabelled textual data. As is illustrated in Figure 2.4, a GRU first encodes a sentence into a unique vector, which is supposed to contain all the information about the sentence. This vector is then fed to a second GRU that will try to decode the previous sentence and the next sentence occuring in the text. This self-supervised model, inspired by Word2Vec (Mikolov et al. 2013), offers an effective pre-training scheme for sentence representations, and is often used to encode the question in VQA models (Kim et al. 2017; Z. Yu et al. 2017).

In this thesis, the question is systematically represented using the last internal state of GRU network pretrained on the Skip-thought task. Our models could surely benefit from the very recent advances in sentence representation such as ELMo (Peters et al. 2018) or Transformer-based architectures (Devlin et al. 2018; Radford et al. 2019).

2.3 Multi-modal fusion

2.3.1 Fusion in VQA

Multi-modal fusion is a critical component of VQA systems. Whether it consists in the whole system or a sub-part of it, we often need to build a learnable function that takes as input two vectors and outputs a single representation. Moreover, this representation is required to account for complex interactions between both modalities. This is why the VQA task has been a fertile playground for researchers on multi-modal fusion. More formally, given the question embedding $\boldsymbol{q} \in \mathbb{R}^{d_q}$ and an image representation $\boldsymbol{v} \in \mathbb{R}^{d_v}$, how do we design a learnable function f, parametrized by θ, that provides an output $\boldsymbol{y} \in \mathbb{R}^{d_o}$ such that $\boldsymbol{y} = f(\boldsymbol{q}, \boldsymbol{v}; \theta)$?

Early works have modeled interactions between multiple modalities with first-order models. The IMG+BOW model in (M. Ren et al. 2015) is the first to use a concatenation to merge a global image representation with a question embedding, obtained by summing all the learned word embeddings from the question. In (Shih et al. 2016), (image, question, answer) triplets are scored in an attentional framework. Each local feature is given a score corresponding to its similarity with textual features. These scores are used to weight region multimodal embeddings, obtained from a concatenation between the region's visual features and the textual embeddings. The hierarchical co-attention network (J. Lu et al. 2016), after extracting multiple textual and visual features, merges them with concatenations and sums. To improve the expressive power in the multi-modal fusion, (Jabri et al. 2016) place a succession of fully-connected layers behind the concatenation of textual and visual features.

However, most of the recent work on multi-modal fusion is focused on bilinear models, as they are a core component of many state-of-the-art VQA models. They express each coordinate in the output as a function of pairwise products between dimensions of the two input vectors. In (Fukui et al. 2016), they use the fact that a bilinear model can be seen as a linear model whose inputs are all the possible products between dimensions of $\boldsymbol{q}$ and $\boldsymbol{v}$: $\boldsymbol{y} = \boldsymbol{W}[\boldsymbol{q} \otimes \boldsymbol{v}]$, where $\otimes$ is the outer product, and $[.]$ corresponds to the vectorization operation. The Multimodal Compact Bilinear (MCB) pooling is introduced to make the model tractable, and the calculation of the outer product avoided thanks to count-sketching techniques (Charikar et al. 2002). However, the best performing methods tackle this complexity issue from a tensor analysis viewpoint. In the recent Multimodal Low-rank Bilinear (MLB) pooling (Kim et al. 2017), the number of parameters in the bilinear model is controlled using the concept of *tensor rank*. They write the interaction between image and text vectors as a Hadamard product (or element-wise multiplication) between vectors: $\boldsymbol{y} = \boldsymbol{W}^o \left((\boldsymbol{W}^q \boldsymbol{q}) * (\boldsymbol{W}^v \boldsymbol{v}) \right)$. More recently, the Multi-modal Factorized Bilinear (MFB) fusion (Z. Yu et al. 2017)

writes each output dimension as a scalar function of the shape $\boldsymbol{y}[k] = \boldsymbol{q}^\top \boldsymbol{W}_k \boldsymbol{v}$, and reduce the model complexity by constraining the rank of each matrix $\boldsymbol{W}_k$.

As we can see, the task of VQA provides an attractive application for developping effective and efficient multi-modal fusions. In the following, we review other significant works that use bilinear models and tensor structurations for other purposes than VQA.

2.3.2 Bilinear models

In all the aforementionned contributions, the tensors we manipulate correspond to parameters of a model that we want to learn using standard DL tools, such as SGD optimizers and back-propagation. In this context, we focus on the representation power and computational complexity reduction. It is worth mentionning that a long line of work on tensor structurations aims at reducing tensors that correspond to data. In these cases, inferring an interpretable structure through the decomposition is often desired. In the next paragraph, we briefly review some of these contributions.

Tensor decompositions in data analysis In the last century, multi-way tensor analysis has been an active field of research. In many problems where the data is acquired directly as different views, multilinear algebra provides an efficient framework for analyzing and understanding the complex underlying phenomena. In fluorescence spectroscopy analysis (Andersen et al. 2003), a low-rank model is used to understand complex chemometrics data, where different samples are measured at several emission wavelengths for several excitation wavelength, thus forming a three-way array. Tensor decomposition are also used to analyze Electroencephalography (EEG) data, in the form of a three-way array of size *time samples* $\times$ *frequency* $\times$ *channel* (Miwakeichi et al. 2004). An extensive review on tensor analysis for chemometrics is provided in (Bro 2006), where they cover other applications such as kinetics, magnetic resonance or chromatography. Blind source separation of statistically independant signals can be solved using a low-rank decomposition (Comon 2014), whereas more complex models can match the structure of correlated sources (Cichocki et al. 2015). In (Goovaerts et al. 2015), a low-rank model is used to detect irregular heartbeats in a three-way array of size *channels* $\times$ *time steps* $\times$ *heartbeats*. Tensor decompositions have also been used as data compression tools, as in (Wang et al. 2008) where a video is compressed using a Tucker model (that we present later in Chapter 3) on the third-order tensor of size *height* $\times$ *width* $\times$ *frames*. The image classification problem has also been viewed through the lens of tensor decompositions. In (Vasilescu et al. 2002), the tensor framework is used to analyse face images of different persons under varying viewpoints, expressions and illumnations. Later, (S. Yan et al. 2007) propose a tensor-based multilinear discriminant analysis algorithm to classify face

images. Finally, web data analysis has also benefitted from the development of tensor decompositions, as these data are often intrinsically multi-modal (Evrim Acar et al. 2005; Bader et al. 2007; Sun et al. 2005; G. Kolda et al. 2005). More details about the above references can be found in (Cichocki et al. 2015; Mørup 2011; E. Acar et al. 2009).

Compressing learning modules An important family of work uses multi-linear algebra and tensor structures to reduce the complexity of DL models. In (Lebedev et al. 2014), tensor decompositions are used to simplify and compress convolution kernels. The convolution filter banks of a trained ConvNet are decomposed into their canonical components using a low-rank approximation. After this architectural change, the network remains differentiable, which allows fine-tuning after the parameter compression. This technique shows important speed-up, under low performance drop. In (Y. Yang et al. 2017a), an application of tensor decompositions for efficient parameter sharing in multi-task learning is presented. The matrix weights of different task-specific networks are considered as elements of a decomposition of the same underlying tensor, which allows an implicit sharing of trainable parameters and improves performance over having multiple single task independant learnings. More recently, (Ye et al. 2018) simplify the large linear projections $\boldsymbol{W}.\boldsymbol{x}$ contained in an LSTM network (Hochreiter et al. 1997b) by reshaping the matrix $\boldsymbol{W}$ and the input vector $\boldsymbol{x}$ into multi-way arrays $\mathcal{W}$ and $\mathcal{X}$. The tensor $\mathcal{W}$ is then compressed using the block-term decomposition, introduced in (De Lathauwer 2008). Compared to the classical LSTM, the compressed architecture converges faster, to a more accurate model, and with less parameters. In Chapter 4, we develop a multi-modal fusion module based on this tensor decomposition. All these works on compressing deep architectures through tensor reduction confirm the intuition that modern DL models (ConvNets, LSTMs and multi-task networks) are over-parametrized. Tensor decompositions constitute a promising research path towards lighter and more efficient models.

Multi-modal fusion Another research track on tensor decomposition for DL is focused on multi-modal fusion, in other contexts than VQA. Given two inputs, represented as vectors $\boldsymbol{x} \in \mathbb{R}^x$ and $\boldsymbol{y} \in \mathbb{R}^y$, what type of trainable function f, parametrized by θ, can be used to merge them in a single output vector $\boldsymbol{z} = f(\boldsymbol{x}, \boldsymbol{y}; \theta) \in \mathbb{R}^z$? One popular family of functions are bilinear models, which can be characterized by a three-way tensor $\mathcal{T} \in \mathbb{R}^{x \times y \times z}$. Each entry $\mathcal{T}_{i,j,k}$ is a learnable parameter that weights a specific product $\boldsymbol{x}_i \boldsymbol{y}_j$ for an output dimensions $\boldsymbol{z}_k$. While very expressive, the number of parameters grows cubically with the number of dimensions in the inputs and output vectors. A naive implementation of such models is thus intractable for most deep learning applications. Leveraging the power of bilinear models for DL systems is however possible if the tensor $\mathcal{T}$ is simplified through efficient structuration. In (Kiros et al. 2014a),

a conditional language model is trained to generate the textual description of an image, represented by the vector $\boldsymbol{x}$. More precisely, the model predicts the next word w_n as a low-rank bilinear fusion between the $\hat{\boldsymbol{r}}$, which represent the previous words $(w_1, ..., w_{n-1})$, and $\boldsymbol{x}$. Another type of tensor decomposition is used in (G. Hu et al. 2017) for face recognition. In this work, a bilinear model merges a visual attributes vector with face recognition features, thus leveraging the robustness of attributes prediction to improve face authentication. The tensor fusion framework has also impacted multi-domain and multi-task learning. In (Y. Yang et al. 2017b) a bilinear fusion between an input vector and a task indicator is used to express a task-specific linear model. To share parameters between tasks or domains, multiple tensor decompositions are evaluated and compared, always showing better performance than training each domain independently. Recently, (Rose Yu et al. 2017) use p-order arrays of parameters in recurrent networks to model interactions between multiple consecutive hidden states, and thus perform long-term forecasting on traffic and climate time-series.

A major part of the work in this thesis falls within this line of research. We study bilinear models to learn the relevant correlations between image and question representations. We develop techniques based on decompositions of higher order tensors to make bilinear models tractable in a DL context. Moreover, we show the links and the interconnections between our methods and previous work on second order fusions. As we delve deeper into tensor decompositions for multi-modal fusion, we explore other applicative contexts than VQA where this type of fusion proves to be beneficial.

2.4 Towards visual reasoning

We saw in Section 2.2 that the input modules of a VQA system often take the form of a DL representation extractor. In particular, an image may be represented as a single vector, or a bag of vectors where each embedding is spatially grounded in the image. Then we introduced the different types of fusion layers in Section 2.3, trained to provide a representation that models the interaction between two vector spaces. In this section, we are interested in how different VQA architectures induce different structures in the model. These structures can be seen as providing visual reasoning capacities to the systems, and are crucial to their performance.

2.4.1 Visual attention

One of the most widely used architecture for VQA is the question-guided visual attention mechanism. This technique was introduced in (Bahdanau et al. 2015) for neural machine translation and in (K. Xu et al. 2015) for image caption generation. When the input is presented as a set of vectors $\{\boldsymbol{v}_i\}_{i=1..N}$, the attention mechanism

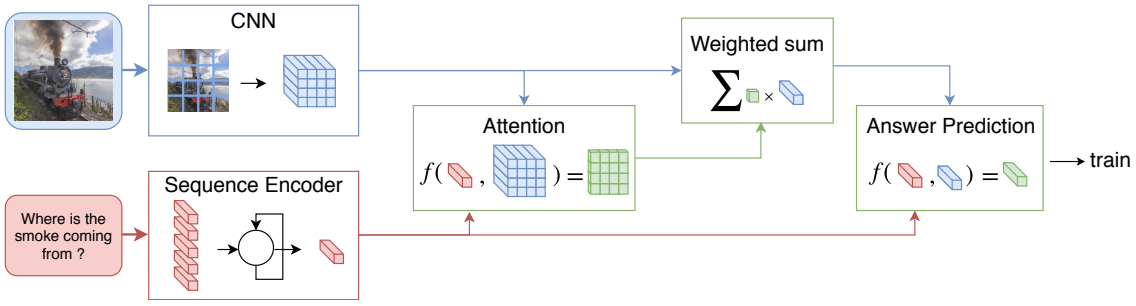


Figure 2.5 – **Visual attention for VQA** Each image vector is fused with the question embedding to provide a scalar value. These scores are used to weight the image region vectors and form the final question-dependant image representation.

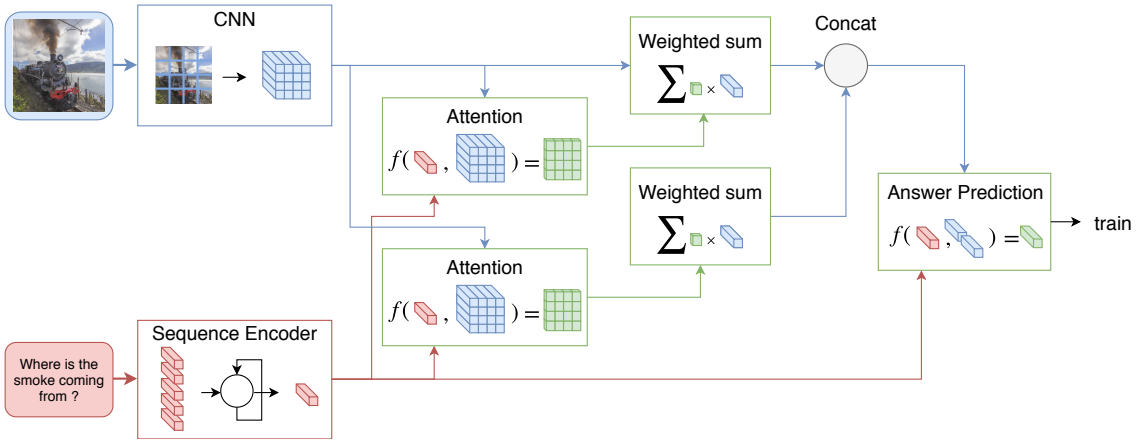


Figure 2.6 – **Multi-glimpse attention for VQA** Multiple attention maps (here 2 glimpses) are computed in parallel and are used to pool the region vectors independantly. The pooled vectors are concatenated and fused with the question to predict the answer.

learns to discard some of these vectors that are irrelevant to complete a certain task, with respect to a *context* vector $\boldsymbol{c}$. For each vector $\boldsymbol{v}_i$, an attention score a_i is computed as a function of $\boldsymbol{v}_i$ and $\boldsymbol{c}$, which accounts for how relevant the input i is, given the context $\boldsymbol{c}$. In the most common and simple case of soft-attention, these scores are used to weight-sum pool the input vectors and provide the final context-dependant representation. In Figure 2.5, we show the attention-based architecture for VQA, where the attention is computed over the set of image region vectors, and the context is provided by a question embedding.

A *multi-glimpse* version of this soft-attention is used in (Fukui et al. 2016), where K attention maps are computed in parallel. Each parametrizes a different weighted sum pooling to provide a per-glimpse image representation. The final question-dependant image representation is the concatenation of the K glimpses. This *multi-glimpse* soft-attention, depicted in Figure 2.6, is used in (Fukui et al. 2016; Kim et al. 2017; Z. Yu et al. 2017; Z. Yu et al. 2018).

The different glimpses can also be computed in an iterative fashion, as in the Stacked Attention Network (SAN) proposed in (Z. Yang et al. 2016). At step k, a query vector is computed with the recurrence law $\boldsymbol{u}^k = \tilde{\boldsymbol{v}}^k + \boldsymbol{u}^{k-1}$. In this equation, $\tilde{\boldsymbol{v}}^k$ represents the attended visual representation at step k over the set of region vectors $\{\boldsymbol{v}_i\}$. The weights that are used to do the sum pooling $\boldsymbol{p}^k$ are a function of the previous query vector $\boldsymbol{u}^{k-1}$. The final query vector $\boldsymbol{u}^K$ is used to predict the answer with a classification layer. A similar method is proposed in (H. Xu et al. 2016), where the iterative visual attention is based on similarities between question words and image regions. All these methods rely on the soft version of attention mechanisms, which is the most widely used because of its simplicity. In the recent work of (Malinowski et al. 2018) they demonstrate with the Hard Attention Network (HAN) that this soft-attention mechanism can efficiently be replaced by a hard selection of multiple regions in the image that are relevant for a given question. They exploit a phenomenon studied in (Chris Olah et al. 2018) which states that the revelance of a region vector is correlated with its L2-norm.

2.4.2 Image/question attention

Besides computing attention over image regions, some work also consider the same type of attention over question words. In (Z. Yu et al. 2017; Z. Yu et al. 2018), attention over question words is computed independantly from the image. An LSTM network provides a representation per word. This sequence of vector passes through a one-dimensional ConvNet that outputs a scalar value per word. Thanks to the structure in the ConvNet, the score for a word depends on its neighbours. The question embedding is defined as a sum of the word representations weighted by their attention scores. In (J. Lu et al. 2016), they propose to use a co-attention network, where an attention over the words is guided by the image, and an attention over image regions is guided by the question. This co-attention is done

for multiple hierarchical levels of semantics in the question representation. This idea of two modalities that mutually condition each other's attention maps is also used in (Nam et al. 2017), where an iterative process progressively refines these soft-attention processes. At a given step, image and question vectors are the output of two attention processes conditionned on the same context vector, which is itself a function of image and question vectors at the previous step. In the recent work of (Kim et al. 2018), the Bilinear Attention Network (BAN) refines even more the attention process as it defines a saliency score between each question word and each image region. The attention score between a word token and an image region is expressed as a bilinear function of their respective representations, and a single vector encodes simultaneously both input modalities. This vector is updated multiple times in an iterative fashion.

2.4.3 Exploiting relations between regions

In the aformentionned methods, all the regions are usually considered independantly from their context. This could prevent the network from learning to answer questions related to the spatial layout of objects, or to the semantic relations between them. This is why methods that exploit relations between objects for VQA constitute a growing line of research. In (Z. Chen et al. 2017), a structured attention mechanism is used to encode cross-region relations. They remark that the weighted sum used in classical attention schemes can be seen as computing the expectation of a region selection process. They model the joint probability of this process with a grid-structured Conditional Random Field (CRF) which considers a region's 4-neighbourhood. For each region, a unary potential is computed as a score that measures the likelihood of selecting it, with respect to a question. Similarly, pairwise potentials are calculated for each pair of neighbouring regions. Finally, approximate inference algorithms take as input these unary and pairwise scores to compute the marginal probability of selecting each region.

It is however not straightforward to adapt this type of local propagation methods to representations that have a non-regular spatial arrangement. In particular, modern VQA architectures rely on the powerful bottom-up features (see Section 2.2.1), in which region vectors are associated to bounding boxes, whose position and size vary from an image to another. The relational modeling has been adapted to these object detection features by (Norcliffe-Brown et al. 2018). They develop a method that generalizes the convolution to features that are not disposed in a regular grid but in a graph, where each node corresponds to a region. First, the bottom-up representation of each region is fused with the question embedding to provide a per-region multi-modal representation. This set of vectors defines a semantic neighbourhood structure, and the value of the edge between two regions corresponds to the scalar product between their representations. Over this graph,

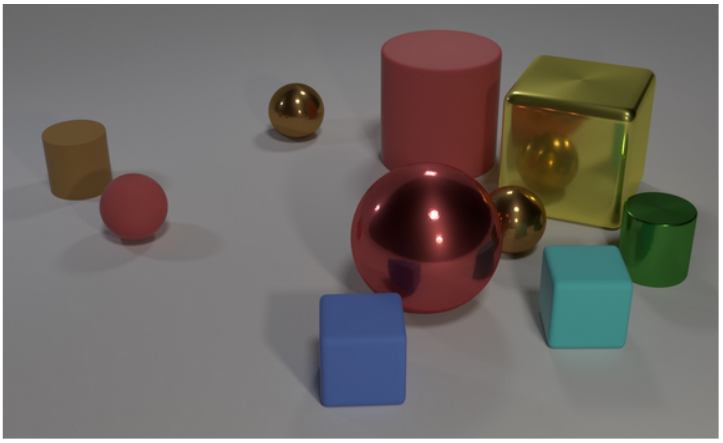
spatial convolutions based on gaussian kernels are used to propagate the node information and make each region vector aware of its local context.

Modeling relations in VQA can also be done in a vectorial manner, as in (Santoro et al. 2017). Given a set of region vectors, the Relation Network (RN) computes a representation for each pair of objects. This vector encodes the ways in which two objects are related, with respect to a specific question. All these pair embeddings are summed to provide an aggregated image-level representation, which is based on pairwise relations of objects. Finally, this aggregated relational information is given to an MLP that predicts the answer.

2.4.4 Composing neural architectures

An interesting idea has seen a growing interest within the VQA community. It states that to answer a question, we should first re-write this question as a program that takes as input the image, and that returns the answer. This program is composed of modules, able to accomplish some elementary perceptual tasks. Not only do we need to learn each perceputal module, we also want to learn to assemble these modules into programs. In (Andreas et al. 2016b), the Neural Module Network (NMN) is proposed as a general architecture for discretly composing heterogeneous, jointly-trained neural modules into deep networks. Each module, corresponding to a composable vision primitive, manipulates attention maps over the image. They are associated with concepts such as `find[c]` which takes as input the image and yields an attention map that locates the argument c (*e.g.* dog, red, ...), or `transform[c]` which applies the transformation c to an attention map (*e.g.* above, ...). The network is assembled using a parsed version of the question. The parser transforms a sentence like " *What color is the tie?* " into a network with the structure `describe[color](find[tie])`. By the same authors, the Dynamic Neural Module Network (D-NMN) (Andreas et al. 2016a) generates multiple network layout candidates for a question. These layouts are scored according to the question with a neural network f, and the best scoring layout is selected to perform the forward-backward pass on the image. Selecting a network is a non-differentiable operation, which implies that the layout scorer f cannot be learned by simple backpropagation. Here, tools from the reinforcement learning community are used to compute the gradients on the layout scorer's parameter, and efficiently learn to choose a layout from candidate layouts, given a question.

To support this line of research, the CLEVR Dataset (Johnson et al. 2017a) has been conceived. In this dataset, very complex questions are asked about images coming from a simple visual world. In the example we show in Figure 2.7, the visual scene is composed of simple objects such as cubes, spheres and cylinders. They exist in a discrete and limited set of colors, shapes and textures. Opposed to this very simple visual world, the questions require strong reasoning capacities,



Q: Are there an **equal number** of **large things** and **metal spheres**?
Q: **What size** is the **cylinder** **that is left of** the **brown metal** thing **that is left of** the **big sphere**?
Q: There is a **sphere** with the **same size as** the **metal cube**; is it **made of the same material as** the **small red sphere**?
Q: **How many** objects are **either** **small cylinders** or **red** things?

Figure 2.7 – **CLEVR Dataset** In this example, the questions require visual reasoning capacities such as attribute identification, counting, comparison, spatial relationships, and logical operations.

such as spatial relationship, logical operations or attribute identifications. Importantly, each one of these natural language questions is associated to a functional program that can be executed on the scene graph representing an image, yielding the answer to the question. This allowed the development of methods that rely on such program annotations to train VQA models.

In (Johnson et al. 2017b), these programs are used as a supervision to an LSTM sequence-to-sequence model. It reads the question and learns to generate the associated program. Each instruction in this program corresponds to a module, similarly to the NMN. However, in contrast to (Andreas et al. 2016b), the different modules have the same generic architecture, instead of using different handcrafted computations for each module. Even though they use ground-truth program annotation at training time, they also backpropagate the error from answer prediction down to the LSTM that generates the network layout, using the same technique as (Andreas et al. 2016a). Similar work was done by (R. Hu et al. 2017), where an LSTM sequence-to-sequence network reads the question, and generates a sequence of instructions. For each decoded instruction, it also generates an attention of the question words, which acts as attributes to the instruction. In the work of (Mascharka et al. 2018), they use the same program generator as the one provided by (Johnson et al. 2017b). However, they improve the design of each specific module by making them manipulate almost exclusively attention maps. This property makes the network's predictions interpretable and

transparent. Finally, in the recent work of (Yi et al. 2018), both the question and the image are parsed to provide structured representations. Similar to previous works, the question is transformed into a functional program using an encoder-decoder LSTM. Moreover, the image is represented as a set of objects along with their attributes (size, color, shape, texture) and their 3D position using a Mask-RCNN segmenter (He et al. 2017). Instead of assigning each instruction from the question to a trainable neural module, the program executor is implemented as a collection of deterministic functional modules that are not trained.

While very appealing, these approches that rely on assembling neural modules based on the question have two major downsides. First, their performance strongly depends on whether or not program annotations are used to learn the program generator. In real-world VQA data, it can be very expensive to generate this type of annotations. Second, their performance can be matched or surpassed by much simpler models that implicitly learn to reason about images, without requiring program annotations. In particular, the very simple Featurewise Linear Modulation (FiLM) proposed in (Perez et al. 2018) modifies the visual feature map with an affine transformation whose parameters are a function of the question. This affine modulation is done at multiple levels to allow complex visual reasoning. For both reasons, these methods based on compositional reasoning remain challenging to use in real-data contexts.

Visual reasoning constitutes the second focus of the work presented in this thesis. We use the multi-glimpse attention as a baseline architecture to support our contributions on multi-modal fusion. Over this model, we propose several enhancements that are related to the region selection process (Section 2.4.1), the interaction modeling between objects (Section 2.4.3) and the iterative reasoning scheme (Section 2.4.4).

2.5 Datasets for VQA

We evaluate our propositions on the main VQA benchmarks. As this field is rather new and changes rapidly, all our contributions may not be evaluated on the same datasets.

- **VQA Dataset**
  Proposed in (Antol et al. 2015), this large scale dataset is built over images from MS-COCO (Tsung-Yi Lin et al. 2014), where each picture is manually annotated with 3 questions. These questions are then answered by 10 distinct annotators, yielding a list of 10 ground-truth answers. The answers are free-form open-ended text, where it is possible to answer anything in natural language. The dataset is composed of 248,349 (image,question) pairs for training, 121,512 for validation and 244,302 for testing. The ground-truth

answers are publicly available for *train* and *val* splits, but not for the *test* split. The *test* score are evaluated on a remote server that compares the predictions to the hidden ground-truths.

To evaluate the performance on this dataset, the authors proposed to use the Open-Ended (OE) Accuracy. This metric compares an answer $\hat{a}$, proposed by the model, to the list of 10 ground-truth answers $[a_1, ..., a_{10}]$ given by annotaters. It is defined as

$$OE(\hat{a}, [a_1, ..., a_{10}]) = \min\left(1, \sum_{i=1}^{10} \frac{\mathbb{1}_{a_i,\hat{a}}}{3}\right) \tag{2.17}$$

where $\mathbb{1}_{i,j}$ is 1 for $i = j$, and 0 otherwise. This metric assesses whether the model answer is on par with at least three annotators out of ten.

One problem with this dataset has been identified as the **textual bias** issue, and has been explained as follows:

> *Inherent structure in our world and bias in our language tend to be a simpler signal for learning than visual modalities, resulting in models that ignore visual information, leading to an inflated sense of their capability.* (Goyal et al. 2017).

As an example, in the VQA Dataset, questions that begin with the words "What sport is" have answer "tennis" in 41% of the examples. This phenomena implies that models can already achieve good performance simply through linguistic pattern matching, without even looking at the visual modality.

- **VQA 2.0 Dataset**
  Built over the VQA Dataset, this second version was proposed in (Goyal et al. 2017) to correct problems related to textual biases. This dataset is specifically designed to be harder than the previous version for models that do not take into account the visual modality. More precisely, it contains pairs of instances where both images are *similar*, on which the *same question* is asked, and for which the two answers are *different*. It contains 443,757 (image,question) pairs on the *train* set, 214,354 on the *val* set and 447,793 on the *test* set. Similarly to the first version, answer annotations for the *test* split are unavalaible, and evaluation is done on an external server.

- **TDIUC**
  Proposed in (Kafle et al. 2017), this dataset is composed of 1,654,167 questions asked about 167,437 images. Though the answer space is also open-ended, it only contains 1,618 unique classes. The particularity of this dataset is that the questions are divised into explicitly-defined question types, which allows to measure the strenghts and weaknesses of a VQA system. Metrics that accompagny TDIUC are devised to assess for a model's robustness

with respect to answer imbalance, as well as to account for perfromance homogeneity across multiple question types.

- **VQA Changing Priors**
  The more recent dataset on which we evaluate our work is VQA-CP (Agrawal et al. 2018). It tackles the problem of generalization by proposing a setting where for every question-type, *train* and *test* have different answer distributions. For example, questions starting by *What color...* are often answered by *white, red, blue* in the *train* set, and by *black, pink, gray* in the *test* set. Two versions are proposed: VQA-CP v1 built from re-arranging *train* and *val* splits from the VQA Dataset, VQA-CP v2 similarly from the VQA 2.0 Dataset.

2.6 Outline and contributions

Falling within the related works presented in this chapter, our work is an attempt to address some of the main problems that arise when building VQA models. Throughout this thesis, we rely on high quality vision and language feature extractor to form the input modules of our system (blue and red blocks in Figure 2.1). The first axis of this work is focused on designing multi-modal fusion modules (orange blocks in Figure 2.1) that learn to extract and represent the relevant correlations between two vector spaces. We explore the powerful solution provided by bilinear models for fusion between modalities, relying on the widely used visual attention framework. This attentional architecture is an entry point to the second axis of our work, where we challenge the idea of explicit region selection induced by the question-guided attention model.

- Chapter 3 MUTAN: MULTIMODAL TUCKER FUSION FOR VQA
  In this chapter, we present the framework of bilinear models to learn a multi-modal fusion of vector inputs. Our method differs from MCB (Fukui et al. 2016) as it is not based on random projections but on the decomposition of tensors. It also differs from MLB (Kim et al. 2017), where the tensor is factorized into low-rank elements. Instead, we rely on the Tucker structure (Tucker 1966), which explicitly assigns trainable parameters to pairwise products between dimensions. We insert this merging function into the commonly used multi-glimpse visual attention for VQA, which is depicted in Figure 2.6. Taking as input the feature maps of an FCN, each region is scored to estimate how relevant it is to answer a given question. This score is then used to aggregate the visual features and provide a question-guided image representation.

- Chapter 4 BLOCK: BILINEAR SUPERDIAGONAL FUSION FOR VQA AND VRD
  Building upon MUTAN, we develop a more advanced bilinear fusion strategy that leverages the notion of *block-term ranks* for higher-order tensors. This algebraic concept generalizes both notions of *rank* and *mode-ranks*, at the heart of low-rank and Tucker decompositions. We constrain the tensor of learnable weights using its block-term ranks, which imposes a block-diagonal structure on the core three-way array that parametrizes the interactions between modalities. We show how this structure brings the best of both MLB and MUTAN fusion techniques: it enables to model pairwise interactions between high dimensional mono-modal projections without exploding the number of parameters. To demonstrate the effectiveness of this bilinear merging strategy, we integrate BLOCK into two DL systems for complex visual tasks. The first is a question-based attention VQA architecture, similar to the one showed in Figure 2.6 except it takes as input the more efficient bottom-up features (see Section 2.2.1). The second one is a simple yet effective architecture for Visual Relationship Detection (VRD), where the features of two boxes are fused through a BLOCK module to predict the relationship predicate that links them.

- Chapter 5 MUREL: MULTIMODAL RELATIONAL REASONING FOR VQA
  Finally, we move away from the classical question-guided visual attention framework with MuRel. While the focus of the two previous contributions was on modeling the interactions between two vector spaces, this work tackles the problem of designing VQA architectures for visual reasoning. Departing from the attentional model of Figure 2.5, the interactions between each region and a question are represented as vectors instead of scalar saliency weights. Moreover, following the line of work presented in Section 2.4.3, we exploit pairwise relations between regions to make each representation aware of its spatial and semantic context. This multi-modal fusion between the question embedding and content-aware region vectors is embedded into an iterative process that refines the representations, and allows to efficiently predict the answer.

Finally, even if the major part of our work is focused on VQA, the contributions proposed in this thesis may apply to broader contexts. In particular, our work on multi-modal fusion between vectors can be applied to any context where we want to fuse information from multiple sources within a DL model. As an example, we show in Chapter 4 an application of our fusion techniques to the task of VRD.

CHAPTER 3

MUTAN: MULTIMODAL TUCKER FUSION FOR VQA

Contents

Chapter abstract

Learning to merge information from two vector spaces is a critical component of Visual Question Answering (VQA) systems. While many Deep Learning (DL) models handle this problem by concatenating the two vectors, we are interested in more powerful solutions.

In this chapter, we tackle the problem of multi-modal fusion using bilinear models. After introducing their general form, we present our approach that combines Tucker decomposition of tensors and matrix rank sparsity. We show how the Tucker decomposition framework generalizes some of the latest bilinear fusions used for VQA, and evaluate our approach on the VQA Dataset.

The work in this chapter, at equal contribution with Rémi Cadène, has led to the publication of a conference paper:

- Hedi Ben-Younes*, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2017). "MUTAN: Multimodal Tucker Fusion for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

3.1 Introduction

To process images and questions, Visual Question Answering (VQA) models rely on high-level semantic representations of the data. As we saw in Section 2.2, information from raw images and texts can be extracted by Deep Learning (DL) models, designed to provide a vector encoding of their input. VQA models are tailored to learn answering questions about images using these high-level mono-modal representations. These models involve at least two well-identified challenges that need to be met, which are detailed in Chapter 2. First, we should develop efficient and expressive **multi-modal fusion** modules (see Section 2.3) that learn to merge two vector representations by modeling potentially complex interaction between features. These modules are then assembled together to form a **visual reasoning** architecture (see Section 2.4) that defines high-level capacities of the model. The focus of this chapter is on the first problem of learning a fusion module between vector spaces. As we detailed in Section 2.3, this fusion can be achieved by simple linear operators, potentially followed by deep networks. However, most recent approaches tackle this problem using bilinear models.

In this chapter, we present the general formulation of bilinear fusion models. These powerful approaches express the fusion between two vectors as a second order function where the coefficients are learnable parameters. With this formulation, the fusion explicitly encodes interactions as it parametrizes pairwise products between vector dimensions. Though these methods are appealing, the number of parameters in these models prevents their use in a naive way, especially in DL contexts where we manipulate vector spaces of relatively high dimensions. Motivated by this complexity concern, we develop MUTAN, a bilinear fusion scheme where the interaction modeling is structured based on the Tucker decomposition of tensors. This structure helps us model rich interactions between image and textual modalities while controlling the model tractability. We also explore combining this tensor-based structure with a matrix-based rank sparsification, to reduce even more the complexity.

In Section 3.2, we present our fusion strategy based on the Tucker decomposition of tensors, and discuss its links to other bilinear fusions developped for VQA. In particular, we show how the framework of Tucker decompositions allows to generalize previous work on multi-modal fusion. We experimentally validate our proposition in Section 3.3 and compare our performance to previous state-of-the-art on the VQA Dataset.

3.2 Bilinear models

As commonly done in VQA, images v and questions q are firstly embedded into vectors. In this chapter, the image is represented using a ResNet-152 (He et al.

2015), which is a specific Convolutional Neural Network (ConvNet) architecture. The resulting representation is the vector $\boldsymbol{v} \in \mathbb{R}^{d_v}$. As for the question, we use the last state of a Gated Recurrent Unit (GRU) network, as described in Section 2.2.2, which provides $\boldsymbol{q} \in \mathbb{R}^{d_q}$. Vision and language representations $\boldsymbol{v}$ and $\boldsymbol{q}$ are then fused using a bilinear fusion operator to produce a vector $\boldsymbol{y} \in \mathbb{R}^{d_o}$, with $d_o = |\mathcal{A}|$ the number of possible answers. This vector encodes the predicted score for each of them. Following the notations in Section 2.1, the final output of the system is a normalized version of $\boldsymbol{y}$:

$$f_\Theta(v, q) = \text{softmax}\left(\boldsymbol{y}\right) \tag{3.1}$$

Each dimension in $f_\Theta(v, q)$ is the probability that the model assigns to a possible answer. The index that provides the maximal value in $f_\Theta(v, q)$ is considered as the answer predicted by the system.

Linear fusion. Prior to studying the bilinear model, we first write the linear model for fusion between $\boldsymbol{q}$ and $\boldsymbol{v}$. It is completely defined by a matrix $\boldsymbol{W} \in \mathbb{R}^{d_o \times (d_q + d_v)}$, and operates the fusion through the following transformation:

$$\boldsymbol{y} = \boldsymbol{W}[\boldsymbol{q}; \boldsymbol{v}] \tag{3.2}$$

where $[\boldsymbol{q}; \boldsymbol{v}]$ denotes the concatenation of both vectors. This equation states that each output dimension k is a weighted sum of all the input dimensions:

$$\boldsymbol{y}[k] = \sum_{i=1}^{d_q} \boldsymbol{W}[k, i]\boldsymbol{q}[i] + \sum_{j=1}^{d_v} \boldsymbol{W}[k, d_q + j]\boldsymbol{v}[j] \tag{3.3}$$

The main drawback of this formulation is that each modality is viewed independantly from the other. The model does not have the structural capacity to consider the visual content in its interactions with the question asked about it.

Bilnear model. Just as a linear model is defined by a matrix, a bilinear model is defined by a third-order tensor $\mathcal{T} \in \mathbb{R}^{d_q \times d_v \times d_o}$. This multi-way array of learnable weights is used to parametrize the fusion as follows:

$$\boldsymbol{y} = (\mathcal{T} \times_1 \boldsymbol{q}) \times_2 \boldsymbol{v} \tag{3.4}$$

where $\times_i$ designates the *i-mode* product between tensors. We can make explicit the calculation performed by these mode products by writing the output coordinate k of $\boldsymbol{y}$:

$$\boldsymbol{y}[k] = \sum_{i=1}^{d_q} \sum_{j=1}^{d_v} \mathcal{T}[i, j, k]\boldsymbol{q}[i]\boldsymbol{v}[j] \tag{3.5}$$

Each output coordinate is a weighted sum over all possible pairwise products of dimension couples in $\boldsymbol{q}$ and $\boldsymbol{v}$.

The principal issue that we face when working with these functions is that they quickly become intractable when dimensions d_q, d_v and d_o are high. The number of free parameters in the tensor is defined as

$$|\mathcal{T}| = d_q d_v d_o \tag{3.6}$$

Under common dimensions for image and language representations, and with a reasonably large number of possible answers, this complexity is prohibitive. In our case, $d_v = 2048$ as the output of the ResNet-152, our GRU provides vectors of dimension $d_q = 2400$, and the we have $d_o = 2000$ possible answers. This leads to $|\mathcal{T}| \sim 9,8.10^9$ free parameters. Such a huge number of parameters is a problem both for learning and for Graphics Processing Unit (GPU) memory consumption: storing 9.8 billion float32 scalars requires approximately 39Go, while top-grade GPU hold about 24Go. Naively applying a bilinear fusion on image and question representations is not manageable, which is why we need to find ways to reduce the complexity.

To circumvent this issue, we attack the problem of complexity from a tensor decomposition perspective. As we develop in Section 3.2.1, we shrink the number of free parameters in $\mathcal{T}$ by imposing a structure on its weights. In particular, we use exploit the Tucker decomposition of tensors to simplify the model and make it tractable.

3.2.1 Tucker decomposition

The Tucker decomposition (Tucker 1966) of a 3-way tensor $\mathcal{T} \in \mathbb{R}^{d_q \times d_v \times d_o}$ expresses $\mathcal{T}$ as a product between *factor matrices* $\boldsymbol{W}_q, \boldsymbol{W}_v$ and $\boldsymbol{W}_o$, and a *core tensor* $\mathcal{D}$ in such a way that:

$$\mathcal{T} = \left(\left(\mathcal{D} \times_1 \boldsymbol{W}_q\right) \times_2 \boldsymbol{W}_v\right) \times_3 \boldsymbol{W}_o \tag{3.7}$$

with $\boldsymbol{W}_q \in \mathbb{R}^{d_q \times t_q}$, $\boldsymbol{W}_v \in \mathbb{R}^{d_v \times t_v}$ and $\boldsymbol{W}_o \in \mathbb{R}^{d_o \times t_o}$, and $\mathcal{D} \in \mathbb{R}^{t_q \times t_v \times t_o}$. In this model, t_q, t_v and t_o are called mode-1, mode-2 and mode-3 ranks (see Figure 3.1). Interestingly, Equation 3.7 states that the weights in $\mathcal{T}$ are inter-dependant: they are obtained through combinations of a restricted number of parameters. More precisely, $\forall i \in [1, d_q], j \in [1, d_v], k \in [1, d_o]$:

$$\mathcal{T}[i,j,k] = \sum_{l=1}^{t_q} \sum_{m=1}^{t_v} \sum_{n=1}^{t_o} \mathcal{D}[l,m,n] \boldsymbol{W}_q[i,l] \boldsymbol{W}_v[j,m] \boldsymbol{W}_o[k,n] \tag{3.8}$$

This decomposition helps us reducing the size of the model. With this parametrization based on the Tucker decomposition, the number of parameters becomes

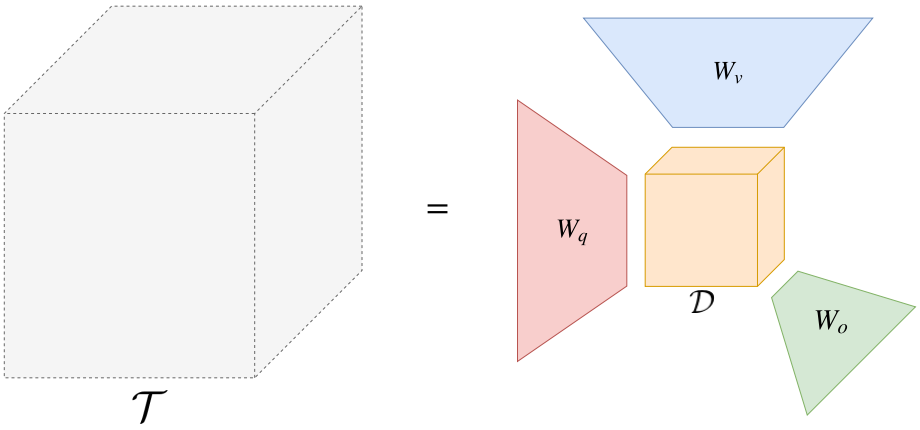


Figure 3.1 – **Tucker decomposition** The Tucker decomposition of a three-way array factorizes the tensor as a product between three matrices (one for each mode) and a smaller tensor.

$$|\mathcal{T}| = d_q t_q + d_v t_v + d_o t_o + t_q t_v t_o. \tag{3.9}$$

This number can be controlled through the hyperparameters t_q, t_v and t_o.

In more traditional uses of this type of decomposition-based methods, such as the ones reviewed in the first paragraph of Section 2.3.2, the multi-way array $\mathcal{T}$ does not correspond to a tensor of trainable parameters, but is actually provided as *data*. This data tensor could for instance represent observations acquired through sensors that measure multiple phenomena, and for which the multi-way array structure is well suited. In this context, tensor decompositions are actually used to find the optimal elements or factors, such that they explain best the tensor $\mathcal{T}$. In particular, the Tucker decomposition is employed to understand, reduce or compress the information stored in $\mathcal{T}$. Note that in our case, the Tucker decomposition is not used to *reduce* an existing tensor. Instead, we use the structure implied by the decomposition to avoid explicitly computing the tensor, as we directly define it from its Tucker elements.

3.2.2 Multimodal Tucker Fusion

As we parametrize the weights of the tensor $\mathcal{T}$ with its Tucker decomposition of the Equation 3.7, we can rewrite the bilinear model of Equation 3.4 using the matrices $\boldsymbol{W}_q, \boldsymbol{W}_v, \boldsymbol{W}_o$ and the tensor $\mathcal{D}$. First, input vectors $\boldsymbol{q}$ and $\boldsymbol{v}$ are transformed following:

$$\tilde{\boldsymbol{q}} = \boldsymbol{q}^\top \boldsymbol{W}_q \tag{3.10}$$

$$\tilde{\boldsymbol{v}} = \boldsymbol{v}^\top \boldsymbol{W}_v \tag{3.11}$$

These mono-modal projections are of size t_q for $\tilde{\boldsymbol{q}}$ and t_v for $\tilde{\boldsymbol{v}}$. Then, a bilinear fusion parametrized by $\mathcal{D}$ is applied on these two projections:

$$\boldsymbol{z} = (\mathcal{D} \times_1 \tilde{\boldsymbol{q}}) \times_2 \tilde{\boldsymbol{v}} \in \mathbb{R}^{t_o} \tag{3.12}$$

$\boldsymbol{z}$ can be seen as a multi-modal representation of the input pair (q, v). This vector sums up the correlations between image and question vectors that are relevant to predict the correct answer. $\boldsymbol{z}$ is then projected into the prediction space using the matrix $\boldsymbol{W}_o$:

$$\boldsymbol{y} = \boldsymbol{z}^\top \boldsymbol{W}_o \in \mathbb{R}^{d_o} \tag{3.13}$$

Eventually, this Tucker structure on the interaction tensor comes down to having a full bilinear interaction between *lower-dimensional projections* of $\boldsymbol{q}$ and $\boldsymbol{v}$. Finally, the output $f_\Theta(v, q)$ is obtained by the softmax the normalization on $\boldsymbol{y}$ as shown in Equation 3.1.

To increase the modeling capacities of our tensor-based fusion, we include non-linearities in the model. More specifically, inspired by (Kim et al. 2017), we use

$$\tilde{\boldsymbol{q}} = \tanh(\boldsymbol{q}^\top \boldsymbol{W}_q) \tag{3.14}$$

$$\tilde{\boldsymbol{v}} = \tanh(\boldsymbol{v}^\top \boldsymbol{W}_v) \tag{3.15}$$

In our experiments, using these non-linearities provides slightly better results. As we will see in Chapter 4, they can be replaced by more efficient power normalizations on the multi-modal vector $\boldsymbol{z}$.

Interpretation Using the Tucker decomposition, the interaction modeling weights $\mathcal{T}$ are factorized into four components, each having a specific role. Matrices $\boldsymbol{W}_q$ and $\boldsymbol{W}_v$ project the question and the image vectors into spaces of respective dimensions t_q and t_v. These dimensions directly impact the modeling complexity that will be allowed for each modality. High values of t_q and t_v enable to learn rich mono-modal projections. Then, the tensor $\mathcal{D}$ models interactions between components of $\tilde{\boldsymbol{q}}$ and $\tilde{\boldsymbol{v}}$. It learns to project the array of all possible products $\tilde{\boldsymbol{q}}[i]\tilde{\boldsymbol{v}}[j]$ into a vector $\boldsymbol{z}$ of size t_o. This dimension controls the complexity allowed for the modeling of interactions between modalities. Finally, the matrix $\boldsymbol{W}_o$ performs the final classification and scores this pair embedding $\boldsymbol{z}$ for each class in $\mathcal{A}$.

3.2.3 MUTAN fusion

The parametrization provided by the Tucker decomposition enables a control of the model complexity, according to Equation 3.9. In this equation, the limiting factor is the size of the core tensor $\mathcal{D}$:

$$|\mathcal{D}| = t_q t_v t_o, \tag{3.16}$$

which invites us to have low values for t_q, t_v and t_o. However, reducing too much these values would result in coarse mono-modal projections $\tilde{\boldsymbol{q}}$ and $\tilde{\boldsymbol{v}}$, lying in very low-dimensional spaces and causing a bottleneck in the modeling.

To further balance between expressivity and number of parameters, we need to break the complexity in $\mathcal{D}$ with respect to the projection dimensions t_q, t_v and t_o. When we perform the t_o bilinear combinations between $\tilde{\boldsymbol{q}}$ and $\tilde{\boldsymbol{v}}$ of Equation 3.12, each dimension $k \in [\![1, t_o]\!]$ in $\boldsymbol{z}$ can be written as:

$$\boldsymbol{z}[k] = \tilde{\boldsymbol{q}}^\top \mathcal{D}[:, :, k] \tilde{\boldsymbol{v}} \tag{3.17}$$

According to this equation, the correlations between elements of $\tilde{\boldsymbol{q}}$ and $\tilde{\boldsymbol{v}}$ are weighted by the parameters of $\mathcal{D}[:, :, k]$. Reducing the number of parameters could possibly be achieved by sparsifying $\mathcal{D}[:, :, k]$. A first solution consists in adding an L1 penalty to these matrices. This would force some parameters to be set to exactly 0 as training goes by. Unfortunately, this solution requires to consider all elements in $\mathcal{D}$ as free trainable parameters, which will eventually be removed after training, but that we must keep in our model during the learning phase.

Instead of a raw sparsity on the parameters, we choose to solve this problem from a structure perspective. More precisely, we state that each $\mathcal{D}[:, :, k]$ is generated by summing a small number of very simple elementary matrices. This corresponds to defining $\mathcal{D}[:, :, k]$ as a sum of rank-one elements:

$$\mathcal{D}[:, :, k] = \sum_{r=1}^{R} \boldsymbol{m}_r^k . \boldsymbol{n}_r^{k\top} \tag{3.18}$$

where $\boldsymbol{m}_r^k \in \mathbb{R}^{t_q}$ and $\boldsymbol{n}_r^k \in \mathbb{R}^{t_v}$. This formulation implies that for each $k \in [1, d_o]$:

$$\text{Rank}(\mathcal{D}[:, :, k]) \leq R \tag{3.19}$$

with equality when the set of matrices $\{\boldsymbol{m}_r^k . \boldsymbol{n}_r^{k\top}\}_{r=1:R}$ is linearly independant. Under this structured sparsity constraint on slices of $\mathcal{D}$, Equation 3.17 becomes:

$$\boldsymbol{z}[k] = \tilde{\boldsymbol{q}}^\top \left(\sum_{r=1}^{R} \boldsymbol{m}_r^k . \boldsymbol{n}_r^{k\top} \right) \tilde{\boldsymbol{v}}, \tag{3.20}$$

which can equivalently be written as

$$\boldsymbol{z}[k] = \sum_{r=1}^{R} \left(\tilde{\boldsymbol{q}}^\top \boldsymbol{m}_r^k \right) \left(\tilde{\boldsymbol{v}}^\top \boldsymbol{n}_r^k \right). \tag{3.21}$$

For each r, we can stack the vectors $\boldsymbol{m}_r^k$ into a matrix $\boldsymbol{M}_r \in \mathbb{R}^{t_q \times t_o}$ such that $\boldsymbol{M}_r[:, k] = \boldsymbol{m}_r^k$. Similarly, we can build $\boldsymbol{N}_r \in \mathbb{R}^{t_v \times t_o}$ such that $\boldsymbol{N}_r[:, k] = \boldsymbol{n}_r^k$. Equation 3.21 can then be re-written as

$$z[k] = \sum_{r=1}^{R} \left(\tilde{q}^\top M_r\right)[k]. \left(\tilde{v}^\top N_r\right)[k], \tag{3.22}$$

where $\tilde{q}^\top M_r \in \mathbb{R}^{t_o}$ and $\tilde{v}^\top N_r \in \mathbb{R}^{t_o}$. Finally, the structured sparsity constraint defined in Equation 3.18 implies that z corresponds to as a sum over R vectors:

$$z = \sum_{r=1}^{R} z_r \tag{3.23}$$

such as for each $r \in [1, R]$,

$$z_r = (\tilde{q}^\top M_r) * (\tilde{v}^\top N_r) \tag{3.24}$$

where $*$ stands for element-wise vector multiplication.

Interpretation Adding this rank constraint on $\mathcal{D}$ leads to expressing the output vector z as a sum over R vectors z_r. To obtain each of these vectors, we project $\tilde{q}$ and $\tilde{v}$ into a common space and merge them with an elementwise product. We can interpret z as encoding an OR interaction over multiple AND gates between projections of $\tilde{q}$ and $\tilde{v}$. $z[k]$ can described in terms of logical operators as:

$$z_r[k] = \left(\tilde{q} \text{ similar to } m_r^k\right) \text{ AND } \left(\tilde{v} \text{ similar to } n_r^k\right) \tag{3.25}$$

$$z[k] = z_1[k] \text{ OR ... OR } z_R[k] \tag{3.26}$$

This decomposition gives an insight of how the fusion is carried out in our MUTAN model. In our experiments, we show that all the vectors z_r for r in $[\![1, R]\!]$ behave differently depending on the type of question. Particularly, we exhibit some cases where some r's specialize over specific question types.

3.2.4 Model unification and discussion

We show how two fusion models that were state-of-the-art at the time of publication of this work, MLB and MCB, can be seen as special cases of our MUTAN fusion. As reviewed in Section 2.3, each of these models uses a different type of bilinear interaction between q and v, hence instantiating a specific parametrization of the weight tensor $\mathcal{T}$. These parametrizations actually consist in a Tucker decomposition with specific constraints on the elements $\mathcal{D}, W_q, W_v$ and W_o. More importantly, when we cast MCB and MLB into the framework of Tucker decompositions, we show that the structural constraints imposed by these two models state that some parameters are fixed, while they are free to be learnt in our full Tucker fusion. This is illustrated in Figure 3.2, Figure 3.3, Figure 3.4 and Figure 3.5, where we show in color the learnable parameters.

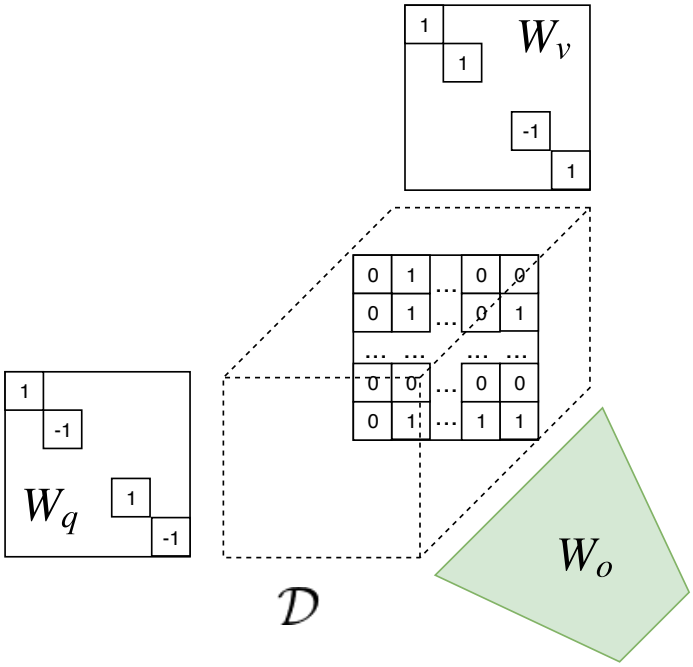


Figure 3.2 – **Multimodal Compact Bilinear (MCB).** W_q and W_v are fixed diagonal matrices, $\mathcal{D}$ is a sparse fixed tensor, only the output factor matrix W_o is learnt.

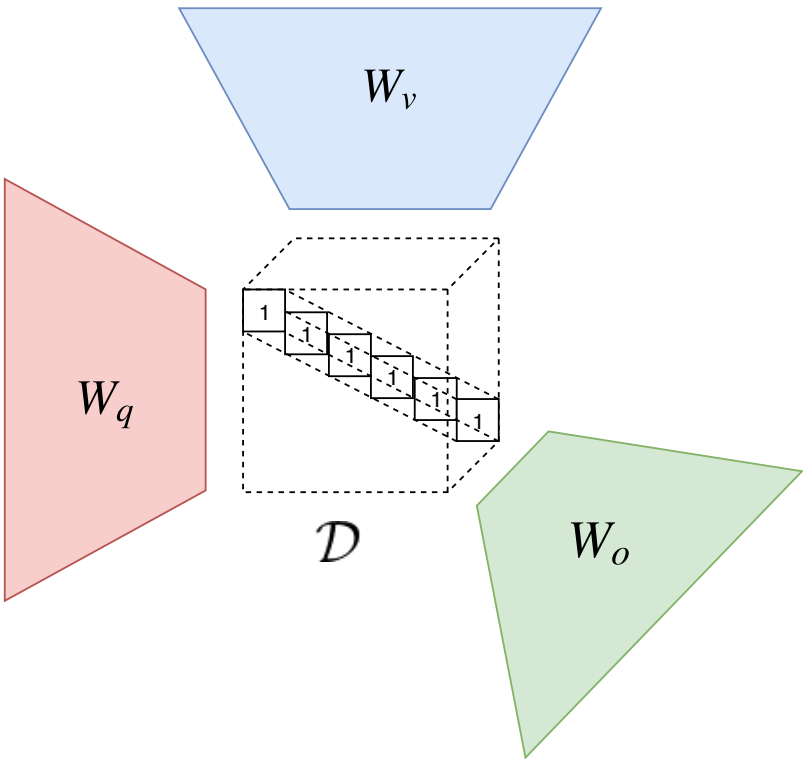


Figure 3.3 – **Multimodal Low-rank Bilinear (MLB).** The 3 factor matrices are learnt but the core tensor is $\mathcal{D}$ set to identity.

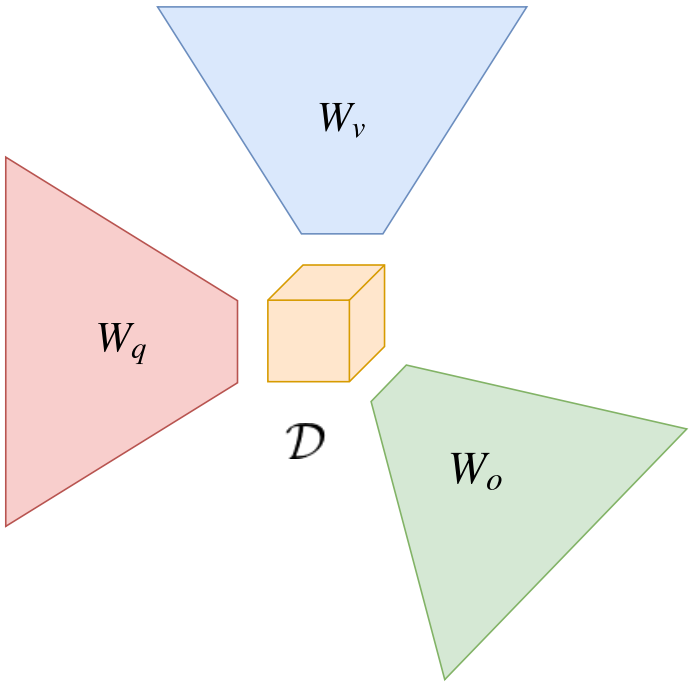


Figure 3.4 – **Tucker.** $\boldsymbol{W}_q$, $\boldsymbol{W}_v$, $\boldsymbol{W}_o$ and $\mathcal{D}$ are learnt. The interaction modeling tensor $\mathcal{D}$ is full.

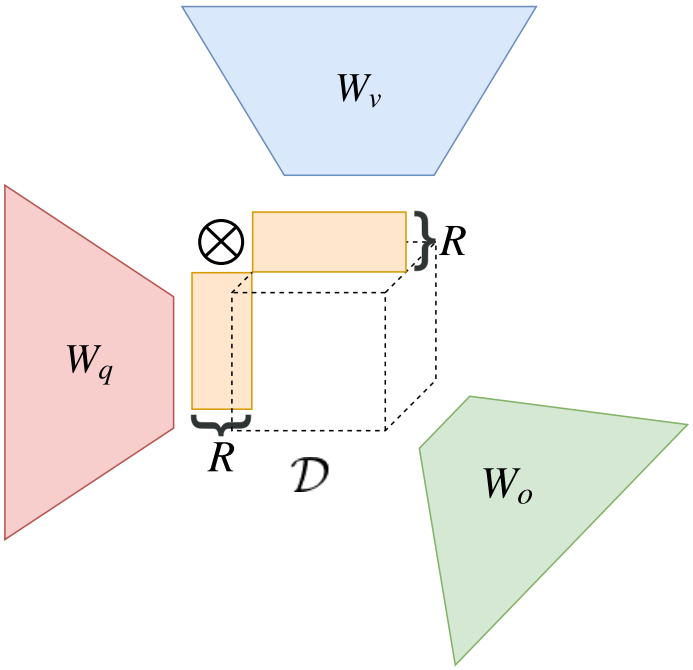


Figure 3.5 – **MUTAN.** $\boldsymbol{W}_q$, $\boldsymbol{W}_v$, $\boldsymbol{W}_o$ and $\mathcal{D}$ are learnt. The interaction modeling tensor $\mathcal{D}$ is simplified and structured with a rank-R decomposition on its mode-3 slices.

MCB We can show that the MCB pooling (Fukui et al. 2016) can be written as a bilinear model where the weight tensor $\mathcal{T}^{mcb}$ is decomposed into its Tucker decomposition, with specific structures on the decompositions' elements. The intramodal projection matrices $\boldsymbol{W}_q^{mcb}$ and $\boldsymbol{W}_v^{mcb}$ are diagonal matrices where the non-zero coefficients take their values in $\{-1;1\}$: $\boldsymbol{W}_q^{mcb} = \textbf{Diag}(\boldsymbol{s}_q)$ and $\boldsymbol{W}_v^{mcb} = \textbf{Diag}(\boldsymbol{s}_v)$, where $\boldsymbol{s}_q \in \mathbb{R}^{d_q}$ and $\boldsymbol{s}_v \in \mathbb{R}^{d_v}$ are random vectors sampled at the instanciation of the model but kept fixed afterwards. The core tensor $\mathcal{D}$ is sparse and its values follow the rule: $\mathcal{D}^{mcb}[i,j,k] = 1$ if $h(i,j) = k$ (and 0 else), where $h : [\![1,d_q]\!] \times [\![1,d_v]\!] \rightarrow [\![1,d_o]\!]$ is randomly sampled at the beginning of training and no longer changed. Please note that this is a mathematical formulation of MCB, and that the actual implementation leverages properties of the functions defined by $h, \boldsymbol{s}_q$ and $\boldsymbol{s}_v$ that allow efficient computation *via* convolutions and Fast Fourier Transform (FFT).

As was noticed in (Kim et al. 2017), all the learnt parameters in MCB are located *after* the fusion. The combinations of dimensions from $\boldsymbol{q}$ and from $\boldsymbol{v}$ that are supposed to interact with each other are randomly sampled beforehand (through h). To compensate for the fact of fixing the parameters $\boldsymbol{s}_q, \boldsymbol{s}_v$ and h, they must set a very high t_o dimension (typically 16,000). This set of combinations is taken as a feature vector for classification.

MLB The low-rank bilinear interaction used in (Kim et al. 2017) defines $\mathcal{T}$ as a sum of R rank-1 tensors. This corresponds to a Candecomp/Parafac (CP) decomposition of the tensor $\mathcal{T}$ such as its rank is equal to R. It is well-known that the low-rank decomposition of a tensor is a special case of the Tucker decomposition, such as $\mathcal{T}^{mlb} = [\![\mathcal{I}_R; \boldsymbol{W}_q, \boldsymbol{W}_v, \boldsymbol{W}_o]\!]$ where $t_q = t_v = t_o = R$, and $\mathcal{I}_R$ is the identity tensor of size $R \times R \times R$. Two major constraints are imposed when reducing Tucker decomposition to low-rank decomposition. First, the three dimensions t_q, t_v and t_o are structurally set to be equal. The dimension of the space in which a modality is projected (t_q and t_v) quantifies the model's complexity. Our intuition is that since the image and language spaces are different, they may require to be modeled with different levels of complexity, hence different projection dimensions. The second constraint is on the core tensor, which is set to be the identity. A dimension k of $\tilde{\boldsymbol{q}}^{mlb}$ is only allowed to interact with the same dimension of $\tilde{\boldsymbol{v}}^{mlb}$, which might be restrictive. We will experimentally show the beneficial effect of removing these constraints.

We would like to point out the differences between MLB and the structured sparsity per slice presented in Section 3.2.3. There are two main differences between the two approaches. First, our rank reduction is made on the core tensor of the Tucker decomposition $\mathcal{D}$, while in MLB they constrain the rank of the global tensor $\mathcal{T}$. This lets us keep different dimensionalities for the projected vectors $\tilde{\boldsymbol{q}}$ and $\tilde{\boldsymbol{v}}$. The second difference is we do not reduce the tensor on the third mode, but only on the first two modes corresponding to the image and question modalities.

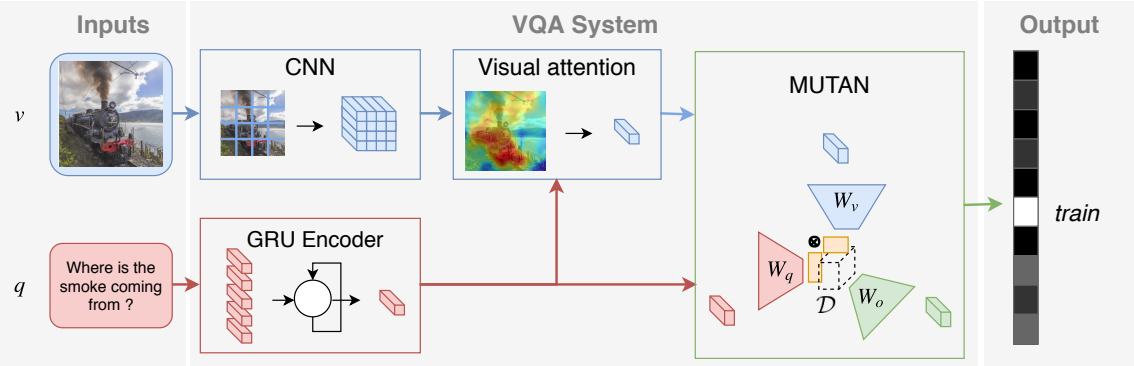


Figure 3.6 – **Attention-based MUTAN architecture for VQA** The question is processed by a GRU encoder to provide a single vector, and the image passes through a fully convolutional ConvNet to yield a $14 \times 14 \times d_v$ features map. Both these representations are fused with a MUTAN bilinear model, within an attentional framework.

The implicit parameters in $\mathcal{D}$ are correlated *inside* a mode-3 slice but independent *between* the slices.

3.2.5 MUTAN architecture

Multi-glimpse attention. We embed our MUTAN fusion module in the multi-glimpse attention mechanism, presented in Section 2.4.1. It takes as input a question embedding $\boldsymbol{q} \in \mathbb{R}^{d_q}$ and the visual representation $\boldsymbol{V} \in \mathbb{R}^{h \times w \times d_v}$ provided by a Fully Convolutional Network (FCN) network (see Section 2.2.1). An attentional system with G glimpses, computes G independant attention maps $\boldsymbol{\alpha}^g \in \mathbb{R}^{h \times w}$. Each coefficient $\boldsymbol{\alpha}^g[i,j]$ is obtained by a MUTAN fusion between the question representation and the corresponding region vector. More precisely:

$$\boldsymbol{\alpha}^g[i,j] = MUTAN^g(\boldsymbol{q}, \boldsymbol{V}[i,j]) \tag{3.27}$$

where $\boldsymbol{V}[i,j] \in \mathbb{R}^{d_v}$ is the representation at the grid position (i,j). Please note that we have one MUTAN fusion module for each of the the G glimpses. Within each glimpse, the attention map is normalized with a softmax activation function:

$$\boldsymbol{s}^g[i,j] = \frac{\exp(\boldsymbol{\alpha}^g[i,j])}{\sum_{k=1,l=1}^{h,w} \exp(\boldsymbol{\alpha}^g[k,l])} \tag{3.28}$$

These normalized scalar maps are used to aggregate the visual features, such that

$$\boldsymbol{v}^g = \sum_{i=1}^{h} \sum_{j=1}^{w} \boldsymbol{s}^g[i,j] \boldsymbol{V}[i,j] \tag{3.29}$$

The vectors outputed by each glimpse are finally concatenated such that

$$\boldsymbol{v} = [\boldsymbol{v}^1, ..., \boldsymbol{v}^G] \in \mathbb{R}^{Gd_v} \tag{3.30}$$

This representation serves as input to the last MUTAN module that outputs the answer prediction:

$$\boldsymbol{y} = MUTAN(\boldsymbol{q}, \boldsymbol{v}) \tag{3.31}$$

An attentional architecture with $G = 1$ is shown in Figure 3.6. Later in Chapter 5, we discuss the relevance and effectiveness of this architecture, and propose an alternative way of interlinking image and language representations.

Learning The transformations performed by the MUTAN fusion is a composition of linear projections, element-wise multiplications and vectorial additions. The model is differentiable, and trainable by backpropagation. All the parameters in the MUTAN architecture are trained end-to-end: the MUTAN fusion module between $\boldsymbol{v}$ and $\boldsymbol{q}$, but also the MUTAN fusions that generate the attention weights.

As we recall in Section 3.2.1, the traditional use of these tensor decompositions methods is, given a **tensor of data** $\mathcal{T}$, find the elements of the decomposition $\mathcal{D}, \boldsymbol{W}_1, \boldsymbol{W}_2$ and $\boldsymbol{W}_3$ that best reconstruct $\mathcal{T}$. In this context, a classical practice is to impose an orthogonality constraint on the matrices. This improves interpretability of the decomposition and makes the optimization process easier. In our application, we do not decompose a tensor but we compose it as the elements of its factorization. In that respect, the orthogonality constraints are not considered in our work.

3.3 Experiments

We now evaluate our multi-modal fusion framework MUTAN on the task of VQA. To do so, we use the most widely benchmarked dataset available at the time when we conducted this work, which is the VQA Dataset (Antol et al. 2015). For details about the dataset and the evaluation metrics, please report to Section 2.5.

Image pre-processing As in (Fukui et al. 2016) and (Kim et al. 2017), we pre-process the images before training our VQA models as follow. We load and resize the image such as the smaller side has size 448, keeping the proportions of the image. Then, we crop at the center and obtain a region of size 448×448. We feed it to a ResNet-152 (He et al. 2015), pre-trained on ImageNet (Krizhevsky et al. 2012). We extract the features before the last Rectified Linear Unit (ReLU), which produces feature maps of size $14 \times 14 \times 2048$.

Question pre-processing We use a similar preprocessing as (Fukui et al. 2016) or (Kim et al. 2017) for the questions. We keep the questions which are associated

to the 2000 most frequent answers. These questions are lower-cased and we remove all the ponctuation marks. We use the space character to split the raw string sentences into word sequences. Then, we replace all the words which are not in the vocabulary of our pre-trained GRU by a special "unknown" word ("UNK"). Finally, we pad all the sequences of words with zero-padding to match the maximum sequence length of 26 words. We use TrimZero as in (Kim et al. 2017) to avoid the zero values from the padding.

Optimization We use the classical implementation of Adam (Kingma et al. 2014) with a learning rate of 10^{-4}, without learning rate decay. Parameters in the GRU are fine-tuned, whereas the image network is kept fixed. During the optimization process, we use a batch size of 100 for experiments in Section 3.3.1, and 512 for those in Section 3.3.2. We use early stopping as a regularizer: during training, we save the model parameters after each epoch. To evaluate our model on the evaluation server, we choose the best epoch according to the accuracy computed on the *val* split when available. As in (Fukui et al. 2016; Kim et al. 2017), some models are trained on the union of *train* and *val* splits (*trainval* split). In this setup, we use the *test-dev* split as a validation set and are obliged to submit several times on the evaluation server. Note that we are limited to 10 submissions per day. In practice, we submit 3 to 4 times per models for epochs associated to training accuracies between 63% to 70%.

3.3.1 Comparison with leading methods

To compare the performance of our proposed approach to other leading works, we associate the MUTAN fusion with recently introduced techniques for VQA, which are described below.

Answer sampling Each (image,question) pair in the VQA dataset is annotated with 10 ground truth answers, corresponding to the different annotators. Each time our model sees an (image,question) pair, we randomly choose an answer among the labels as ground-truth. We keep only the answers occuring more than 3 times within those 10.

Data augmentation We use Visual Genome (Krishna et al. 2017) as a data augmentation to train our model, keeping only the examples whose answer is in our vocabulary $\mathcal{A}$ extracted from the VQA Dataset. This triples the size of our training set.

Ensembling MUTAN(3) is made of a MUTAN trained on the *trainval* split with 2 glimpses, an other MUTAN with 3 glimpses and a third MUTAN with 2 glimpses trained on the *trainval* split with the Visual Genome data augmentation. All

three have been trained with the same hyper-parameters besides the number of glimpses. MUTAN(5) is made of the three same MUTAN models of MUTAN(3) and two MLB models which can be viewed as a special case of our MUTAN. The first MLB has 2 glimpses and was trained on the *trainval* split. The second MLB has 4 glimpses and was trained on the *trainval* split with the visual genome data augmentation. The final results of both ensembles are obtained by averaging the logits, extracted before the final softmax activation of all their models.

Results State-of-the-art comparison results are gathered in Table 3.1. First, we can notice that attention-based bilinear models (MCB and MLB) have a strong edge over other methods with a less powerful fusion scheme. MUTAN outperforms all the previous methods on *test-dev* and *test-std*. This validates the relevance of the proposed fusion scheme, which models precise interactions between modalities. The good performances of MUTAN (5) also confirms its complementarity with MLB, MLB learns informative mono-modal projections, wheras MUTAN is explicitly devoted to accurately model multi-modal interactions.

Finally, we see that MUTAN (3), an ensemble of 3 MUTAN-based models (without MLB) also outperforms state-of-the-art results at the moment of publication. We can point out that this improvement is reached with is an ensembling of 3 models, which is smaller than the previous state-of-the-art MLB results containing an ensembling of 7 models. Please note that MLB(7) and MCB(7) also use the Visual Genome data augmentation.

3.3.2 Further analysis

Experimental setup In this section, we examine under different aspects the fusion between $\boldsymbol{q}$ and $\boldsymbol{v}$ with the Tucker decomposition of tensor $\mathcal{T}$. For practical considerations, we only consider a global visual vector, computed as the average of the 14×14 region vectors given by the ResNet-152. We also do not use the answer sampling: we train the models to always predict the most frequent answer among the 10 ground-truth responses. Neither do we apply the model ensembling technique explained in Section 3.3.1. All the models are trained on the VQA *train* split, and the scores are reported on *val*.

Impact of a plain tensor The goal is to see how important are all the parameters in the core tensor $\mathcal{D}$, which models the correlations between projections of $\boldsymbol{q}$ and $\boldsymbol{v}$. We train multiple Tucker, where we fix all projection dimensions to be equal to each other: $t_q = t_v = t_o = t$ and t ranges from 20 to 220. In Figure 3.7, we compare these Tucker with a model trained with the same projection dimension, but where

	test-dev				*test-std*
	Y/N	No.	Other	All	All
SMem 2-hop (H. Xu et al. 2016)	80.87	37.32	43.12	57.99	58.24
AYN (Malinowski et al. 2016)	78.39	36.45	46.28	58.39	58.43
SAN (Z. Yang et al. 2016)	79.3	36.6	46.1	58.7	58.9
D-NMN (Andreas et al. 2016a)	81.1	38.6	45.5	59.4	59.4
ACK (Wu et al. 2016)	81.01	38.42	45.23	59.17	59.44
MRN (Kim et al. 2016)	82.28	38.82	49.25	61.68	61.84
HieCoAtt (J. Lu et al. 2016)	79.7	38.7	51.7	61.8	62.1
MCB (7) (Fukui et al. 2016)	83.4	39.8	58.5	66.7	66.5
MLB (7) (Kim et al. 2017)	84.57	39.21	57.81	66.77	66.89
MUTAN (3)	84.54	39.32	57.36	67.03	66.96
MUTAN (5)	**85.14**	**39.81**	**58.52**	**67.42**	**67.36**

Table 3.1 – **State-of-the-art comparison** We evaluate MUTAN against other methods from the literature. We train on the VQA Dataset *train+val*, and add the data from VisualGenome (Krishna et al. 2017). Performance is reported on *test-dev* and *test-std* splits; (n) designates an ensemble of n models.

$\mathcal{D}$ is replaced by the identity tensor [1]. One can see that Tucker gives much better results than identity tensor, even for very small core tensor dimensions. This shows that Tucker is able to learn powerful correlations between modalities [2].

Impact of rank sparsity We want to study the impact of introducing the rank constraint in the core tensor $\mathcal{D}$. We fix the input dimensions $t_q = 210$ and $t_v = 210$, and vary the output dimension t_o for multiple rank constraints R. As we can see in Figure 3.8, controlling the rank of slices in $\mathcal{D}$ allows to better model the interactions between the unimodal spaces. The different colored lines show the behavior of MUTAN for different values of R. Comparing $R = 60$ (blue line) and $R = 20$ (green line), we see that a lower rank allows to reach higher values of t_o without overfitting. The number of parameters in the fusion is lower, and the accuracy on the *val* split is higher.

Fusion scheme comparison To point out the performance variation due to the fusion modules, we first compare MUTAN to leading methods for bilinear

1. This is strictly equivalent to MLB (Kim et al. 2017) without attention. However, we are fully aware that it takes between 1000 and 2000 dimensions of projection to be around the operating point of MLB. With our experimental setup, we just focus on the effect of adding parameters to our fusion scheme.

2. Notice that for each t, Tucker has t^3 parameters. For instance, for $t = 220$, Tucker adds 10.6M parameters over identity.

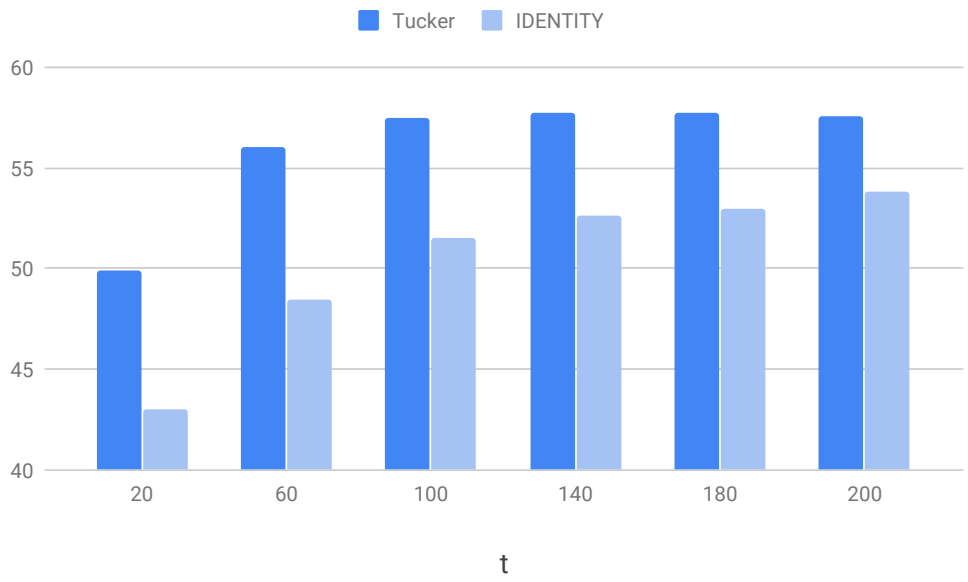


Figure 3.7 – **Comparing Tucker and CP structures** For different number of projection dimensions, we observe that a model trained using the Tucker decomposition performs better than a model with the CP decomposition. Models are trained on VQA Dataset *train* split and performance is reported on *val*.

fusion, under the same experimental framework. We use the same visual features, language models, text pre-processing and optimizers. This allows us to isolate the influence of multi-modal fusion. We compare multiple merging scheme in Table 3.2:

- Concat: a baseline where v and q are merged by simply concatenating them;
- MCB: we choose an output dimension of 16,000, as indicated in their article;
- MLB: we choose an output dimension of 1,200, as indicated in their article;
- Tucker: MUTAN model without the rank sparsity constraint, *i.e.* with a full tensor as $\mathcal{D}$. We choose all the projection dimensions to be equal to each other: $t_q = t_v = t_o = 160$. These parameters are chosen considering the results on *val* split.
- MUTAN: the full Tucker decomposition with rank sparsity strategy on the matrix slices of $\mathcal{D}$. We choose all the projection dimensions to be equal to each other: $t_q = t_v = t_o = 360$, and a rank $R = 10$. These values were chosen so that MUTAN and Tucker have the same number of parameters.

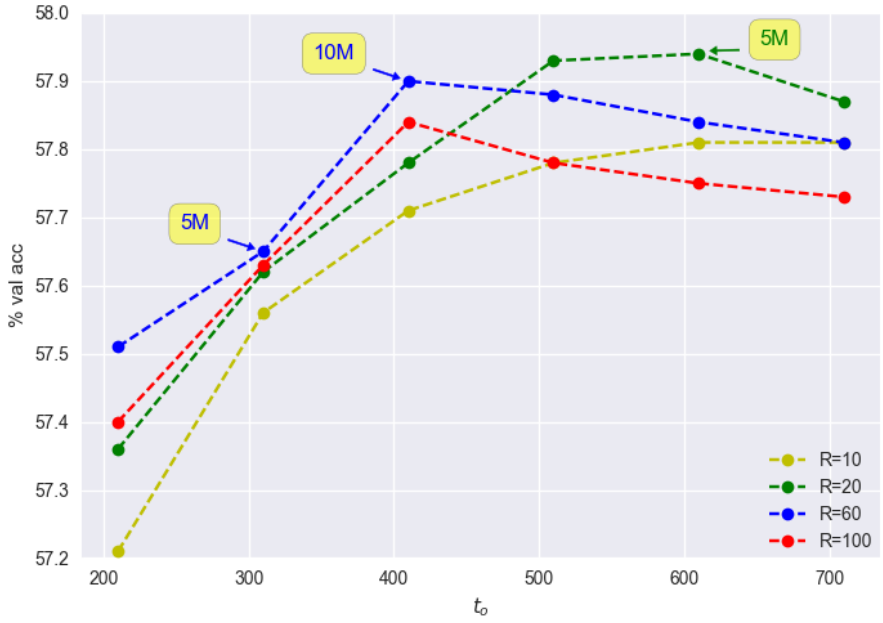


Figure 3.8 – **Impact of rank sparsity** We show performance variations with respect to t_o, for different values of R. The yellow labels indicate the number of parameters in the fusion. Models are trained on VQA Dataset *train* split and performance is reported on *val*.

To implement MCB and MLB, we use the code made available by the authors at `https://github.com/jnhwkim/cbp` and `https://github.com/jnhwkim/MulLowBiVQA`. This allows us to use their model on top of our mono-modal features, which are slightly different than theirs. Moreover, the code for our models can be found at `https://github.com/cadene/vqa.pytorch`.

As we can see in Table 3.2, Tucker performs slightly better than MLB, which validates the fact that modeling full bilinear interactions between low dimensional projections is relevant, compared with having strong mono-modal transformations with a simple fusion scheme (element-wise product). With the structured sparsity constraint, MUTAN obtains the best results, validating our intuition of having a nice tradeoff between the projection dimensions and a reasonable number of useful bilinear interaction parameters in the core tensor $\mathcal{D}$. Finally, a naive late fusion MUTAN+MLB further improves performances (about +1pt on *test-dev*). It validates the complementarity between the two types of tensor decomposition, confirming the results of the experiments in Section 3.3.1 with the attentional setup.

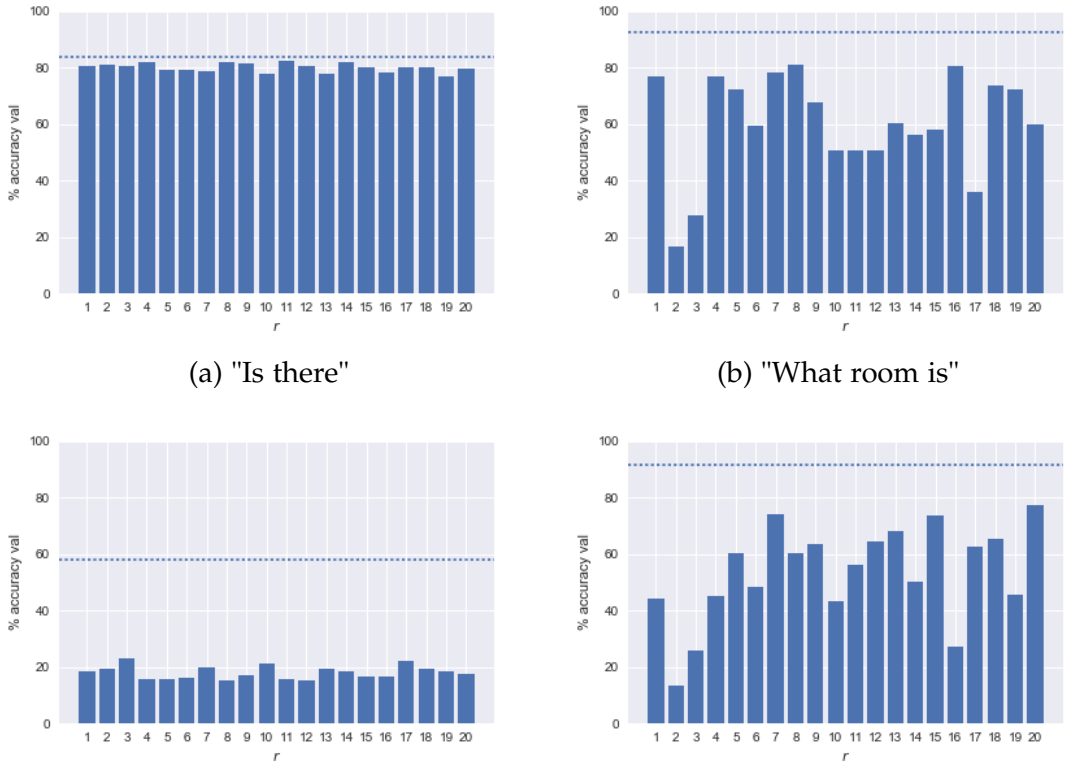
Qualitative observations In MUTAN, the vector z, which encodes the (image,question) pair, is expressed as a sum over R vectors z_r (see Equation 3.17).

		test-dev				*val*
Model	$\|\theta\|$	Y/N	No.	Other	All	All
Concat	8.9	79.25	36.18	46.69	58.91	56.92
MCB	32	80.81	35.91	46.43	59.40	57.39
MLB	7.7	**82.02**	36.61	46.65	60.08	57.91
Tucker	4.9	81.44	36.42	46.86	59.92	57.94
MUTAN	4.9	81.45	**37.32**	**47.17**	**60.17**	**58.16**
MUTAN+MLB	17.5	82.29	37.27	48.23	61.02	58.76

Table 3.2 – **Fusion scheme comparison**. We evaluate different fusion mechanisms under the same setup. Models are trained on the VQA Dataset *train* split, and results are reported on the *test-dev* and *val* splits. $|\theta|$ indicates the number of learnable parameters (in million).

We want to study the R different latent projections that have been learnt during training, and assess whether the representations have captured different semantic properties of inputs. We quantify the differences between each of the R spaces using the VQA question types. We first train a model on the *train* split, with $R = 20$, and measure its performance on the *val* set. Then, we set to zero all of the z_r vectors except one, and evaluate this ablated system on the validation set. In Figure 3.9, we compare the full system to the R ablated systems for 4 different question types. The dotted line shows the accuracy of the full system, while the different bars show the accuracy of the ablated system for each R. Depending on the question type, we observe 3 different behaviors of the ranks. When the question type's answer support is small, we observe that each rank has learnt enough to reach almost the same accuracy as the global system. This is the case for questions starting by "Is there", whose answer is almost always "yes" or "no". Other question types require information from all the latent projections, as in the case of "What is the man". This leads to cases where all projections perform equally and significantly worst when taken individually than when combined to get the full model. At last, we observe that specific projections contribute more than others depending on the question type. For example, latent variable 16 performs well on "what room is", and is less informative to answer questions starting by "what sport is". The opposite behavior is observed for latent variable 17.

We run the same kind of analysis for the MUTAN fusion in the attention mechanism. We train an attentional MUTAN with $G = 1$ glimpse. In Figure 3.10, we show for two images the different attentions that we obtain when turning off all the projections but one. For the first image, we can see that a projection focuses on the elephant, while another focuses on the woman. Both these visual informations are necessary to answer the question "Where is the woman ?". The



(a) "Is there"

(b) "What room is"

(c) "What is the man"

(d) "What sport is"

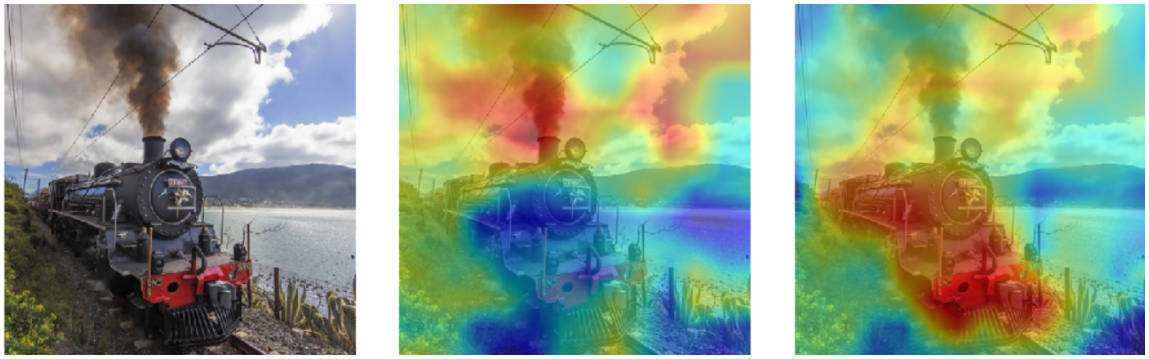
Figure 3.9 – **Per-rank performance of ablated system on multiple question types** Visualizing the performances of ablated systems according to the R variables. Full system performance is showed in dotted line.

same behavior is observed for the second image, where a projection focuses on the smoke while another gives high attention to the train.

3.4 Conclusion

In this chapter, we introduced our first strategy (MUTAN) developped for the VQA task. We propose a module that learns to represent interactions between visual and textual information using a bilinear model. It is built on the Tucker decomposition, which factorizes the interaction tensor into interpretable elements. The hyperparameters of the model are the tensor *mode-ranks*, through which we affect the complexity of the bilinear interaction. We also design a low-rank matrix constraint to increase the efficiency of the model. Interestingly, we show how the Tucker decomposition framework generalizes previous fusion models developped for VQA.

(a) Question: Where is the woman ? - Answer: on the elephant



(b) Question: Where is the smoke coming from ? - Answer: train

Figure 3.10 – **Visualization of per-rank saliency maps** The original image is shown on the left. The center and right images show heatmaps obtained when turning off all the projections but one, for two different projections. Each projection focuses on a specific concept needed to answer the question.

We evaluate our approach on the VQA Dataset, which was the most widely used dataset at the time of the submission of this work. We show how the MUTAN-based attentional architecture compares favorably to leading VQA methods, providing highly competitive results in the ensembling setup. Besides, we validate the influence of structural parameters of our model such as the tensor mode-ranks or the low-rank sparsity constraint. We also run a comparison of MUTAN to other fusion methods. This experiment shows that we can reach more accurate models with less parameters, thus demonstrating the strength of our parametrization.

While the VQA Dataset provides a way to evaluate an overall performance of VQA systems, other datasets allow for more a precise analysis. In particular, the TDIUC dataset separates questions into subtasks, enabling a fine-grained assessment of the strengths and weaknesses of a model. Quantitative evaluation, in the remainder of this thesis, is also carried on these newer and more detailed datasets.

In the next chapter, we further investigate bilinear models through tensor structuration. We study in Chapter 4 an efficient factorization that breaks the complexity in the core tensor while ensuring expressivity of the model.

CHAPTER 4

BLOCK: BILINEAR SUPERDIAGONAL FUSION FOR VQA AND VRD

Contents

Chapter abstract

Bilinear models are powerful approaches for multi-modal fusion in a Deep Learning (DL) setup. However, they come with computational issues due to the tensorial nature of their parameters. As we saw in Chapter 3, Tucker decompositions provide a convenient framework for tensor reduction. They factorize the interaction by introducing a smaller core tensor that explicitly models correlations between mono-modal projections. Unfortunately, this methods projects each modality in a representation space whose dimension must be low if we want to keep the fusion tractable, which can cause a bottleneck in the model.

In this chapter, we delve deeper into tensor decompositions for learning powerful and tractable bilinear fusion models. We develop a fusion module based on the block-term decomposition, whose expression encompasses the Tucker formulation, as well as other tensor structures. This fusion module is embedded into a multi-glimpse attentional architecture for Visual Question Answering (VQA), enhanced with the latest advances in terms of visual representations. We

demonstrate competitive performance with this system on two widely used datasets. Additionally, we showcase the interest of these fusion models for another challenging Computer Vision (CV) problem which is Visual Relationship Detection (VRD). We design a simple architecture for this task based on multi-modal fusion and provide promising results.

The work in this chapter has led to the publication of a conference paper:

- Hedi Ben-Younes, Rémi Cadène, Nicolas Thome, and Matthieu Cord (2019). "BLOCK: Bilinear Superdiagonal Fusion for Visual Question Answering and Visual Relationship Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)*.

4.1 Introduction

Simplifying the expression of third order tensors is essential to make bilinear fusion models tractable in Deep Learning (DL) applications. In Chapter 3, we showed that the Tucker decomposition could factorize the full interaction by defining a bilinear fusion between lower-dimensional projections. Doing so, it separates the model complexity from its dependance on input and output dimensions. However, as we saw in Section 3.2.3, the presence of this core tensor imposes to have very small dimensions for the mono-modal projections, which could harm the system performance and cause a bottleneck in the model. To circumvent this issue, we simplified the core tensor with a structured sparsity constraint based on low-rank matrix factorization.

In this chapter, we explore an alternative structuration of the fusion parameters where we fix all values of the core tensor to be exactly 0 outside a pre-defined block diagonal. This structure breaks the complexity of the bilinear fusion as it models interactions between *groups of dimensions* from input projections, thus enabling expressive interaction modeling while keeping high dimensional mono-modal projections. Interestingly, this structure corresponds to the block-term decomposition (De Lathauwer 2008), where tensor complexity is defined by the notion of *block-term ranks*. This notion encapsulates both concepts of *rank* and *mode ranks*, at the basis of Candecomp/Parafac (CP) and Tucker decompositions. We capitalize on this complexity analysis to provide a new way to control the trade-off between the expressiveness and complexity of the fusion model.

In Section 4.2, we present the BLOCK model and show how it is related to previous work on bilinear fusion. Compared to MUTAN in Chapter 3, we show how BLOCK gives a more general framework of tensor factorizations. In particular, it allows to model rich multi-modal fusions between high dimensional projections. In Section 4.3, we demonstrate the practical effectiveness of BLOCK on the Visual Question Answering (VQA) task. Our fusion module is embedded into a multi-glimpse attentional architecture that leverages the power of the recently

introduced bottom-up features (see Section 2.2.1). Furthermore, we explore in Section 4.4 the challenging problem of Visual Relationship Detection (VRD), for which we propose a simple and effective system based on the BLOCK fusion model.

4.2 BLOCK fusion model

In this section, we present the BLOCK fusion strategy and discuss its connection to other bilinear fusion methods from the literature. We quickly recall the general form of bilinear models that we presented in Section 3.2. Please note that we slightly change some the notations in this chapter in comparison with Chapter 3. We remove the notations $\boldsymbol{q} \in \mathbb{R}^{d_q}$ and $\boldsymbol{v} \in \mathbb{R}^{d_v}$, as the fusion module we develop now is not used exclusively used for merging a question $\boldsymbol{q}$ and an image $\boldsymbol{v}$.

In its general form, a bilinear fusion model takes as input two vectors $\boldsymbol{x}^1 \in \mathbb{R}^I$ and $\boldsymbol{x}^2 \in \mathbb{R}^J$, and projects them to a K-dimensional space with tensor products:

$$\boldsymbol{y} = \mathcal{T} \times_1 \boldsymbol{x}^1 \times_2 \boldsymbol{x}^2 \tag{4.1}$$

where $\boldsymbol{y} \in \mathbb{R}^K$. Each component of $\boldsymbol{y}$ is a quadratic form of the inputs: $\forall k \in [1, K]$,

$$\boldsymbol{y}[k] = \sum_{i=1}^{I} \sum_{j=1}^{J} \mathcal{T}[i, j, k].\boldsymbol{x}^1[i].\boldsymbol{x}^2[j] \tag{4.2}$$

A bilinear model is completely defined by its associated tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$, the same way as a linear model is defined by its associated matrix.

4.2.1 BLOCK model

In order to reduce the number of parameters and constrain the complexity of the model, we express $\mathcal{T}$ using the block-term decomposition. As we discuss later, it generalizes the CP and Tucker decompositions, which are used in previous work to make bilinear models tractable. The decomposition of $\mathcal{T}$ in rank (L,M,N) terms is defined as:

$$\mathcal{T} = \sum_{r=1}^{R} \mathcal{D}_r \times_1 \boldsymbol{A}_r \times_2 \boldsymbol{B}_r \times_3 \boldsymbol{C}_r \tag{4.3}$$

where $\forall r \in [1, R]$, $\mathcal{D}_r \in \mathbb{R}^{L \times M \times N}$, $\boldsymbol{A}_r \in \mathbb{R}^{I \times L}$, $\boldsymbol{B}_r \in \mathbb{R}^{J \times M}$ and $\boldsymbol{C}_r \in \mathbb{R}^{K \times N}$. This decomposition is called *block-term* because it can be written as

$$\mathcal{T} = \mathcal{D}^{bd} \times_1 \boldsymbol{A} \times_2 \boldsymbol{B} \times_3 \boldsymbol{C} \tag{4.4}$$

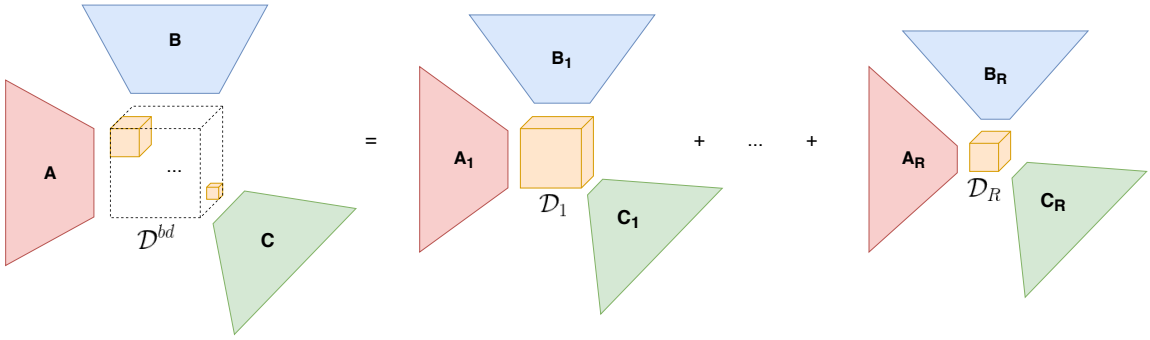


Figure 4.1 – **BLOCK framework.** The third-order interaction tensor is decomposed in R rank-(L,M,N) terms. We give here the two equivalent representations of the block-term decomposition, presented in this paper. On the left, we show the formulation with the block-superdiagonal tensor decomposition that corresponds to 4.4. On the right, we express it as a sum of small decompositions, as written in Equation 4.3. Through the R number of blocks and the dimensions of the $\boldsymbol{A}$, $\boldsymbol{B}$ and $\boldsymbol{C}$ projections, we can handle the trade-off between model complexity and expressivity.

In this equation, matrices $\boldsymbol{A} \in \mathbb{R}^{I \times LR}$, $\boldsymbol{B} \in \mathbb{R}^{J \times MR}$ and $\boldsymbol{C} \in \mathbb{R}^{K \times NR}$ are defined as

$$\boldsymbol{A} = [\boldsymbol{A}_1, ..., \boldsymbol{A}_R] \tag{4.5}$$
$$\boldsymbol{B} = [\boldsymbol{B}_1, ..., \boldsymbol{B}_R] \tag{4.6}$$
$$\boldsymbol{C} = [\boldsymbol{C}_1, ..., \boldsymbol{C}_R] \tag{4.7}$$

and the tensor $\mathcal{D}^{bd} \in \mathbb{R}^{LR \times MR \times NR}$ is the block-diagonal concatenation of the tensors $\mathcal{D}_r$:

$$\mathcal{D}^{bd} = \text{diag}\left(\mathcal{D}_1, ...\mathcal{D}_R\right). \tag{4.8}$$

This equality between the two expressions is illustrated in Figure 4.1. In this figure, we display different sizes for each block to show the freedom allowed in the model choice. However, early experiments on variable block sizes did not show any practical interest compared to fixed-sized blocs.

Similarly to what we did in Section 3.2.2 with the Tucker decomposition, we now parametrize the tensor $\mathcal{T}$ with its block-term decomposition defined in Equation 4.3. We can rewrite the bilinear model in Equation 4.1 using the matrices $\{\boldsymbol{A}_r\}_{r=1..R}$, $\{\boldsymbol{B}_r\}_{r=1..R}$, $\{\boldsymbol{C}_r\}_{r=1..R}$ and the tensors $\{\mathcal{D}_r\}_{r=1..R}$. First, input vectors $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ are transformed following:

$$\hat{\boldsymbol{x}}^1 = \boldsymbol{x}^{1\top} \boldsymbol{A} \in \mathbb{R}^{LR} \tag{4.9}$$
$$\hat{\boldsymbol{x}}^2 = \boldsymbol{x}^{2\top} \boldsymbol{B} \in \mathbb{R}^{MR} \tag{4.10}$$

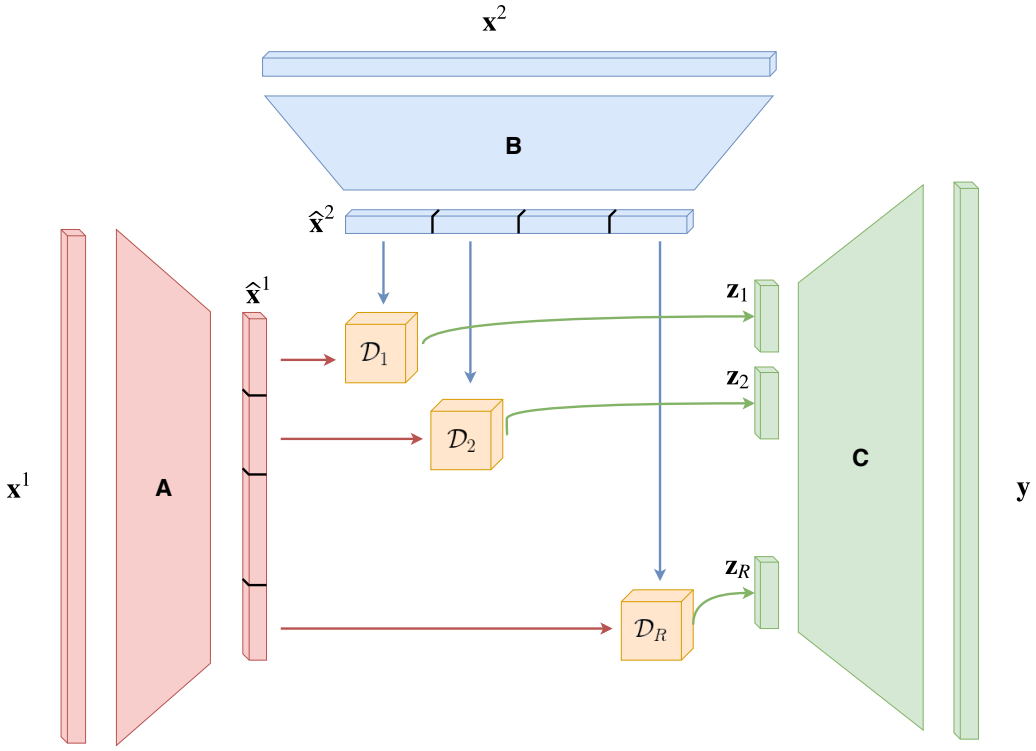


Figure 4.2 – **BLOCK multi-modal fusion.** First, $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ are projected with matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ to produce $\hat{\boldsymbol{x}}^1$ and $\hat{\boldsymbol{x}}^2$. Then, the block-sparsity enforced in $\mathcal{D}^{bd}$ allows to model efficiently the interactions between both projections, and to finally produce $\boldsymbol{y}$.

These two projections are merged with a fusion parametrized by the block-superdiagonal tensor $\mathcal{D}^{bd}$. Each block $\mathcal{D}_r \in \mathbb{R}^{L \times M \times N}$ in $\mathcal{D}^{bd}$ parametrizes a fusion between *chunks* of size L from $\hat{\boldsymbol{x}}^1$ and of size M from $\hat{\boldsymbol{x}}^2$. Each chunk-vs-chunk fusion produces a vector of size $\boldsymbol{z}_r \in \mathbb{R}^N$ such that:

$$\boldsymbol{z}_r = \mathcal{D}_r \times_1 \hat{\boldsymbol{x}}^1[rL : (r+1)L] \times_2 \hat{\boldsymbol{x}}^2[rM : (r+1)M] \tag{4.11}$$

Finally, all the $\boldsymbol{z}_r$ vectors are concatenated to form

$$\boldsymbol{z} = [\boldsymbol{z}_1, ..., \boldsymbol{z}_R] \in \mathbb{R}^{NR} \tag{4.12}$$

The final prediction vector is $\boldsymbol{y} \in \mathbb{R}^K$, which consists in a projection of the multi-modal representation $\boldsymbol{z}$ with the matrix $\boldsymbol{C}$:

$$\boldsymbol{y} = \boldsymbol{C}\boldsymbol{z} \in \mathbb{R}^K \tag{4.13}$$

The whole computation of $\boldsymbol{y}$ with respect to $\boldsymbol{x}^1$ and $\boldsymbol{x}^2$ is depicted in Figure 4.2.

Notably, we take inspiration from (Tsung-Yu Lin et al. 2015) to add a normalization of $\boldsymbol{z}$ that consists in a combination of signed square-root and L2-normalization.

More precisely, instead of Equation 4.12, the multi-modal representation z is computed as:

$$z \leftarrow [z_1, ...z_R] \in \mathbb{R}^{NR} \tag{4.14}$$

$$z \leftarrow \text{sign}(z)\sqrt{|z|} \tag{4.15}$$

$$z \leftarrow \frac{z}{\|z\|_2} \tag{4.16}$$

This normalization was firstly introduced in (Perronnin et al. 2010) to reduce sparsity in high-dimensional vectors, as absolute values that are close to 0 are increased by the square root. While we did not conduct experiments to measure the sparsity in z, we found this normalization to be effective in practice.

Discussion. When working with a linear model whose input and output dimensions are large, a usual technique to limit the number of parameters is to restrict the hypothesis space by constraining the rank of its associated matrix. The formal notion of *matrix rank* quantifies how complex a linear model is allowed to be. However, when it comes to imposing a structural constraint on the complexity of a bilinear model, multiple algebraic concepts can be used. We give two examples that are related to the block-term decomposition.

The CP decomposition (Carroll et al. 1970; Harshman et al. 2001) of a tensor $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ expresses it as the linear combination of rank-1 terms

$$\mathcal{T} = \sum_{r=1}^{R} a_r \otimes b_r \otimes c_r \tag{4.17}$$

where $\otimes$ denotes the outer product, and the vectors $a_r \in \mathbb{R}^I$, $b_r \in \mathbb{R}^J$ and $c_r \in \mathbb{R}^K$ represent the elements of the decomposition. With this formulation, each coefficient of $\mathcal{T}$ can be expressed as

$$\mathcal{T}[i, j, k] = \sum_{r=1}^{R} a_r[i] b_r[j] c_r[k] \tag{4.18}$$

For a given tensor, its *rank* is defined as the minimal number of vector triplets that can generate it as their sum. Thus, restricting the hypothesis space for $\mathcal{T}$ to the set of tensors defined by Equation 4.17 guarantees that the rank is upper-bounded by R. Applying this constraint on $\mathcal{T}$, Equation 4.1 is simplified into

$$y = C\left(\left(x^{1\top} A\right) * \left(x^{2\top} B\right)\right) \tag{4.19}$$

where $*$ denotes element-wise multiplication and

$$A = [a_1, ..., a_R] \in \mathbb{R}^{I \times R} \tag{4.20}$$

$$B = [b_1, ..., b_R] \in \mathbb{R}^{J \times R} \tag{4.21}$$

$$C = [c_1, ..., c_R] \in \mathbb{R}^{K \times R} \tag{4.22}$$

This decomposition can be seen as a special case of the block-term decomposition where $L = M = N = 1$, reducing $\mathcal{D}^{bd}$ to a super-diagonal identity tensor.

The rank-(L,M,N) **Tucker decomposition** of $\mathcal{T} \in \mathbb{R}^{I \times J \times K}$ expresses it as a product between a tensor and three matrices:

$$\mathcal{T} = \mathcal{D} \times_1 \boldsymbol{A} \times_2 \boldsymbol{B} \times_3 \boldsymbol{C} \tag{4.23}$$

where $\mathcal{D} \in \mathbb{R}^{L \times M \times N}$, $\boldsymbol{A} \in \mathbb{R}^{I \times L}$, $\boldsymbol{B} \in \mathbb{R}^{J \times M}$ and $\boldsymbol{C} \in \mathbb{R}^{K \times N}$. With this formulation, each coefficient of $\mathcal{T}$ can be explicited:

$$\mathcal{T}[i,j,k] = \sum_{l=1}^{L} \sum_{m=1}^{M} \sum_{n=1}^{N} \mathcal{D}[l,m,n] \boldsymbol{A}[i,l] \boldsymbol{B}[j,m] \boldsymbol{C}[k,n] \tag{4.24}$$

As detailed in (De Lathauwer 2008), the Tucker decomposition assumes a constraint on the three *matricizations* of $\mathcal{T}$, which are the matrices that result from the concatenation of all tensor slices along a certain axis. More formally, they are defined as $\mathcal{T}_{JK \times I} \in \mathbb{R}^{JK \times I}$, $\mathcal{T}_{KI \times J} \in \mathbb{R}^{KI \times J}$ and $\mathcal{T}_{IJ \times K} \in \mathbb{R}^{IJ \times K}$ such that for each index triplet $i \in 1..I, j \in 1..J$ and $k \in 1..K$,

$$\mathcal{T}_{JK \times I}[(j-1)K + k, i] = \mathcal{T}[i,j,k] \tag{4.25}$$
$$\mathcal{T}_{KI \times J}[(k-1)I + i, j] = \mathcal{T}[i,j,k] \tag{4.26}$$
$$\mathcal{T}_{IJ \times K}[(i-1)J + j, k] = \mathcal{T}[i,j,k] \tag{4.27}$$

Restricting the hypothesis space of $\mathcal{T}$ to the set of tensors defined by Equation 4.23 guarantees that

$$\text{Rank}\left(\mathcal{T}_{JK \times I}\right) \leq L \tag{4.28}$$
$$\text{Rank}\left(\mathcal{T}_{KI \times J}\right) \leq M \tag{4.29}$$
$$\text{Rank}\left(\mathcal{T}_{IJ \times K}\right) \leq N \tag{4.30}$$

These ranks are called the *mode ranks* of $\mathcal{T}$. Applying this constraint to $\mathcal{T}$, Equation 4.1 can be re-written as:

$$\boldsymbol{y} = \boldsymbol{C}\left(\mathcal{D} \times_1 \left(\boldsymbol{x}^{1\top} \boldsymbol{A}\right) \times_2 \left(\boldsymbol{x}^{2\top} \boldsymbol{B}\right)\right) \tag{4.31}$$

This decomposition can be seen as a special case of the block-term decomposition where there is only $R = 1$ block in the core tensor.

As studied in (De Lathauwer 2008), the notion of tensor complexity should be expressed not only in terms of rank or mode ranks, but using the number of blocks and the mode ranks of each block. It appears that CP and Tucker decompositions are two extreme cases, where only one of the two quantities is used. For the CP,

the number of blocks corresponds to the rank, but each block is of size (1,1,1). The mono-modal projections can be high-dimensional and thus integrate rich transformation of the input, but the interactions between both projections is relatively poor as a dimension from one projection is only allowed to interact with a single dimension from the other projection. For the Tucker decomposition, there is only one block of size (L,M,N). The interaction modeling is very rich since all inter-correlations between feature dimensions of the different modalities are considered. However, this quantity of possible interactions limits the dimensions of the projected space, which can cause a bottleneck in the model that we have already identified in Section 3.2.3.

BLOCK being built on the block-term decomposition, we constrain the tensor using a combination of both concepts, which provides a richer modeling of the interactions between modalities. This richness is ensured by the R tensors $\mathcal{D}_r$, each parametrizing a bilinear function that takes as inputs chunks of $\tilde{\boldsymbol{x}}^1$ and $\tilde{\boldsymbol{x}}^2$. As this interaction modelling is done by chunks and not for every possible combination of components in $\tilde{\boldsymbol{x}}^1$ and $\tilde{\boldsymbol{x}}^2$, we can reach high dimensions in the projections $\boldsymbol{A}$ and $\boldsymbol{B}$ without exploding the number of parameters in $\mathcal{D}^{bd}$. This property of having a fine interaction modeling between high dimensional projections is very desirable in our context where we need to model complex interactions between high-level semantic spaces. As we show in the experiments, performance of a bilinear model depends on both the number and the size of the blocks $\mathcal{D}_r$ that parametrize the system.

4.3 BLOCK fusion for VQA task

Now that we developed our BLOCK fusion strategy, we evaluate its performance and analyze its behaviour on the task of VQA. In Section 4.3.1 we present the overall architecture in which we embed the BLOCK fusion module. In Section 4.3.2, we quantitively compare BLOCK with other fusion models. Finally, in Section 4.3.3, we show that our BLOCK-based system performs favorably against leading VQA systems.

4.3.1 VQA architecture

We build our VQA model based on the classical attentional architecture used in Chapter 3, enriched by our proposed merging scheme. Instead of the Fully Convolutional Network (FCN) representations used in Chapter 3, we take advantage of the new bottom-up image vectors provided by (Teney et al. 2018), consisting of a set of detected objects and their representation (see Section 2.2.1 for more details). For the question representation, we use a similar Gated Recurrent Unit (GRU) encoder than the one we used in Chapter 3, except that we keep

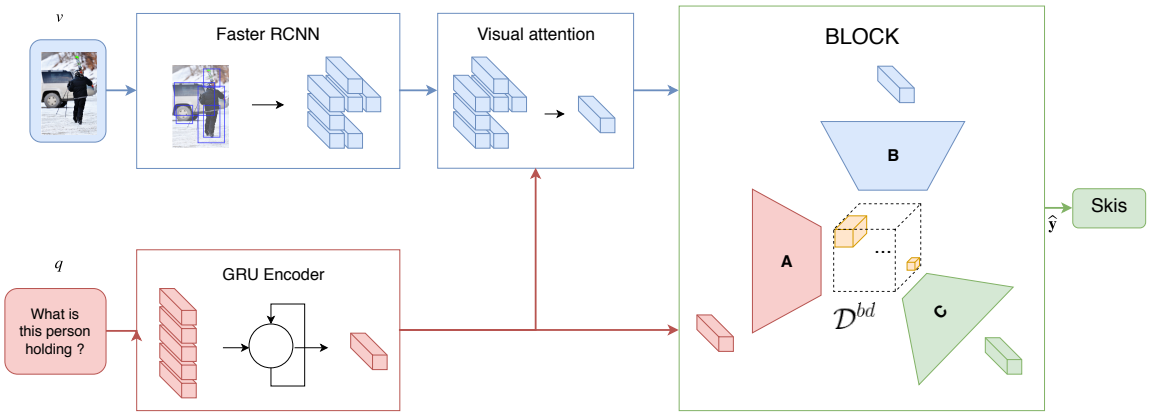


Figure 4.3 – **Architecture for VQA.** Architecture for VQA that embeds the BLOCK bilinear fusion. To make this system more efficient, we integrate the fusion in an attentional framework.

all the output vectors (one for each word). All the vectors are pooled with the recent self-attention mechanism proposed in (Z. Yu et al. 2018). This question vector is used as a context to guide the visual attention over the bounding box representations. Saliency scores are produced using a BLOCK fusion between each region vector and the question embedding. Finally, another BLOCK fusion module merges the question with the aggregated image representation to produce a distribution over possible answers. The architecture is depicted in Figure 4.3.

Details For the BLOCK layers, we set $L = M = N = 80, R = 20$ and constrain the rank of each mode-3 slices of each block to be less than 10 with the technique we developped in Section 3.2.3. We found these hyperparameters with a a cross-validation on the *val* set. As we detailed in Section 2.1, we consider the 3000 most frequent answers as independant classes and train our system with a cross-entropy loss. We use the answer sampling technique explained in Section 3.3.1. We optimize the parameters of our model using Adam (Kingma et al. 2014) with a learning rate of $1e^{-4}$, without learning rate decay or gradient clipping, and with a batch size of 200. We early stop the training of our models according to their accuracy on a holdout set.

4.3.2 Fusion analysis

In Table 4.1, we compare BLOCK to 8 different fusion schemes available in the literature on the commonly used VQA 2.0 Dataset (Goyal et al. 2017), presented in Section 2.5. We train all the models on *trainval* minus a small subset used for early-stopping, and report the performance on *test-dev* set. For each fusion strategy, we run a grid search over its hyperparameters and keep the model that performs best on our validation set. We report the size of the model, corresponding to the

number of parameters in the last fusion module: the one that lies between the attended image features, the question embedding, and the answer prediction. We briefly describe the different fusion schemes used for the comparison:

– (1) the two vectors are projected on a common space, and their summation is projected to predict the answer;

– (2) the vectors are concatenated and passed at the input of a 3-layer Multi-Layer Perceptron (MLP);

– (3) a bilinear interaction based on a count-sketching technique that projects the outer product between inputs on a multimodal space, as in (Fukui et al. 2016) (MCB);

– (4) a bilinear interaction where the tensor is expressed as a Tucker decomposition, as in Chapter 3;

– (5) a bilinear interaction where the tensor is expressed as a CP decomposition, as in (Kim et al. 2017) (MLB);

– (6) a bilinear interaction where each 3rd mode slice matrix of the tensor is constrained by its rank, as in (Ruichi Yu et al. 2017) (MFB);

– (7) a bilinear interaction where the tensor is expressed as a Tucker decomposition, and where its core tensor has the same rank constraint as (6), as in Chapter 3 (MUTAN);

– (8) a higher order fusion composed of cascaded (6), as in (Z. Yu et al. 2018) (MFH);

– (9) our BLOCK fusion.

From the results in Table 4.1, we see that the simple sum fusion (1) provides a very low baseline. We also note that the MLP (2) does not provide the best results, despite its non-linear structure. The MLP has practical difficulties to reach a solution that looks for interactions between modalities, as this prior is not induced by its structure.

Instead, top performing methods are based on a bilinear model. The structure imposed on the parameters highly influences the final performance. We can see that (3), which simplifies the bilinear model using random projections, has efficiency issues due to the count-sketching technique.

These issues are alleviated by the other bilinear methods, which use the tensor decomposition framework to practically implement the interaction. Our BLOCK method (9) gives the best results. As we saw, the block-term decomposition generalizes both CP and Tucker decompositions, which is why it is not surprising to see it surpass them. Interestingly, it even surpasses (8) which is based on a higher-order interaction modeling, while using 30M less parameters. This strongly indicates that controlling a bilinear model through its block-term ranks provides an efficient trade-off between modeling capacities and number of parameters.

To further validate this hypothesis, we evaluate a BLOCK fusion with only 3M parameters. This model reaches an Open-Ended (OE) Accuracy of 64.91%. Unsurprisingly, it does not surpasses all the methods against which we compare.

	Description	$\|\Theta\|$	All	Yes/no	Number	Other
(1)	Sum	8M	58.48	71.89	36.56	52.09
(2)	Concat+MLP	13M	63.85	81.34	43.75	53.48
(3)	B + count-sketching	32M	61.23	79.73	39.13	50.45
(4)	B + Tucker decomp.	14M	64.21	81.81	42.28	54.17
(5)	B + CP decomp.	16M	64.88	81.34	43.75	53.48
(6)	B + 3rd mode rank	24M	65.56	82.35	41.54	56.74
(7)	(4) and (6)	14M	65.19	82.22	42.1	55.94
(8)	Higher order fusion	48M	65.72	82.82	40.39	56.94
(9)	B + Block-term decomp.	18M	**66.41**	**82.86**	**44.76**	**57.3**

Table 4.1 – **Fusion scheme comparison on VQA.** Comparison of the fusion schemes on VQA 2.0 Dataset *test-dev* set. $|\Theta|$ is the number of parameters learned in the fusion modeling. *All* is the overall Open Ended accuracy (higher is better). *Yes/no*, *Numbers* and *Others* are subsets that correspond to answers types. In the descriptions, the letter B corresponds to a bilinear model.

However, it obtains competitive results, improving over 5 out of 8 methods that all use far more parameters. This result shows the efficiency of the parametrization offered by the block-term decomposition.

4.3.3 Comparison to leading VQA methods

We compare our model with state-of-the-art VQA architecture on two datasets: the widely used VQA 2.0 Dataset and TDIUC. TDIUC is currently the biggest VQA dataset in terms of number of images, with a training set of 1,115,299 image-question-answer triplets and a testing set of 538,868 triplets. The Arithmetic Mean of Per-Type accuracies (A-MPT) and Harmonic Mean of Per-Type accuracies (H-MPT) metrics account for how well the model behaves across the different question types, and their normalized versions Arithmetic Normalized Mean of Per-Type accuracies (A-NMPT) and Harmonic Normalized Mean of Per-Type accuracies (H-NMPT) assess the robustness of the model with respect to answer imbalance.

As we show in Table 4.2, our model is able to outperform the preceding ones on TDIUC by a large margin for every metrics, especially those which account for bias in the data. We notably report a gain of +1.7 in accuracy, +3.95 in A-MPT, +5.05 in H-MPT, +16.12 in A-NMPT, +15.45 in H-NMPT, over the best scoring model in each metric. The high results in the harmonic metrics (H-MPT and H-NMPT) suggest that BLOCK performs well across all question types, while the high scores

Model	Accuracy	A-MPT	H-MPT	A-NMPT	H-NMPT
Ques. only (Kafle et al. 2017)	62.74	39.31	25.93	21.46	8.42
NMN* (Andreas et al. 2016b)	79.56	62.59	51.87	34.00	16.67
MCB* (Fukui et al. 2016)	81.86	67.90	60.47	42.24	27.28
RAU* (Noh et al. 2016)	84.26	67.81	59.00	41.04	23.99
BLOCK	**85.96**	**71.84**	**65.52**	**58.36**	**39.44**

Table 4.2 – **Comparison to previous work on TDIUC.** Models are trained on the TDIUC training set, and results are reported on the test set. * scores reported from (Kafle et al. 2017).

Model	VQA2 Test-dev				VQA2 Test-std			
	All	Yes/no	Num.	Other	All	Yes/no	Num.	Other
MCB*	-	-	-	-	62.27	78.82	38.28	53.36
Bottom-up	65.32	81.82	44.21	56.05	65.67	82.20	43.90	56.26
MFH	65.80	-	-	-	-	-	-	-
Counter	**68.09**	83.14	**51.62**	**58.97**	**68.41**	83.56	**51.39**	**59.11**
BLOCK	67.58	**83.6**	47.33	58.51	67.92	**83.98**	46.77	58.79

Table 4.3 – **Comparison to previous work on VQA 2.0.** The models were trained on the union of VQA 2.0 *trainval* split and VisualGenome (Krishna et al. 2017) train split. *All* is the overall OpenEnded accuracy (higher is better). *Yes/no*, *Numbers* and *Others* are subsets that correspond to answers types. Only single model scores are reported. * scores reported from (Goyal et al. 2017)

in the normalized metrics (A-NMPT and H-NMPT) denote that our model is robust to answer imbalance type of bias in the dataset.

In Table 4.3, we see that our fusion model obtains competitive results on VQA 2.0 Dataset compared to previously published methods. We are outperformed by Counter (Y. Zhang et al. 2018), whose proposition rely on a completely different architecture. Still, our model performs better than Bottom-up (Teney et al. 2018) and Multi-modal Factorized Higher-order (MFH) (Z. Yu et al. 2018), with whom we share the global VQA architecture. In further details, we point out that BLOCK surpasses (Z. Yu et al. 2018) reaching a +1.78 improvement in the overall accuracy on *test-dev*, even though the latter encompasses the current state-of-the-art fusion scheme. Furthermore, we use the same image features than (Teney et al. 2018) and are able to achieve a +2.26 gain on *test-dev* and +2.25 on *test-std*.

4.4 BLOCK fusion for VRD task

To demonstrate the genericity and the effectiveness of BLOCK, we evaluate its performance on a simple model for Visual Relationship Detection (VRD). The task of VRD aims at predicting triplets of the type *"subject-predicate-object"* where *subject* and *object* are localized objects, and *predicate* is a label corresponding to the relationship that links them (for example: "man-riding-bicycle", "woman-holding-phone"). To predict this relationship, multiple types of information are available, for both the subject and object regions: classes, bounding box coordinates, visual features, *etc.* However, this context being more recent than VQA, fusion techniques are less formalized and more *ad-hoc*. In (H. Zhang et al. 2017a), the relation is predicted by a substractive fusion between subject and object representations, each consisting in a linear function of relative coordinates, class distributions and visual features. (Y. Li et al. 2017) predicts the relationship by a complex message passing structure between subject and object representations, and (Dai et al. 2017) uses a formulation inspired from Conditional Random Fields to perform joint recognition between the subject, object and predicate classes. We adopt in the following a very simple architecture, to put emphasis on the fusion module between different information sources.

We train and evaluate our model on widely the VRD Dataset (C. Lu et al. 2016). It is composed of 5,000 images with 100 object categories and 70 predicates. It contains 37,993 relationships with 6,672 unique triplets and an average of 24.25 predicates per object category. The dataset is divided between 4,000 images for training and 1,000 for testing. Three different settings are commonly used to evaluate a model on VRD:

(1) **Predicate prediction**: the coordinates and class labels are given for subject and object regions in both training and evaluation phases, and the goal is to predict the predicate. This setup assesses the ability of the model to predict a relationship, regardless of the object detection stage;

(2) **Phrase detection**: no bounding boxes are given in the evaluation phase. A predicted triplet $<subject, predicate, object>$ is said to match a ground-truth if the three labels match and if the union region of its bounding boxes matches the union region of the ground-truth triplet, with IoU above 0.5;

(3) **Relationship detection**: more challenging than (2), a predicted triplet is said to match a ground-truth if the predicted subject matches with the ground-truth subject and the predicted object matches with the ground-truth object.

An illustration for each of these evaluation settings is provided in Figure 4.4. For each of these settings, performance is usually measured with Recall@50 and Recall@100.

In Section 4.4.1, we present our BLOCK-based architecture for VRD. In Section 4.4.2, we use this architecture to compare BLOCK to all the fusion modules we listed in Section 4.3.2. We also conduct an intrinsic study of the behaviour of

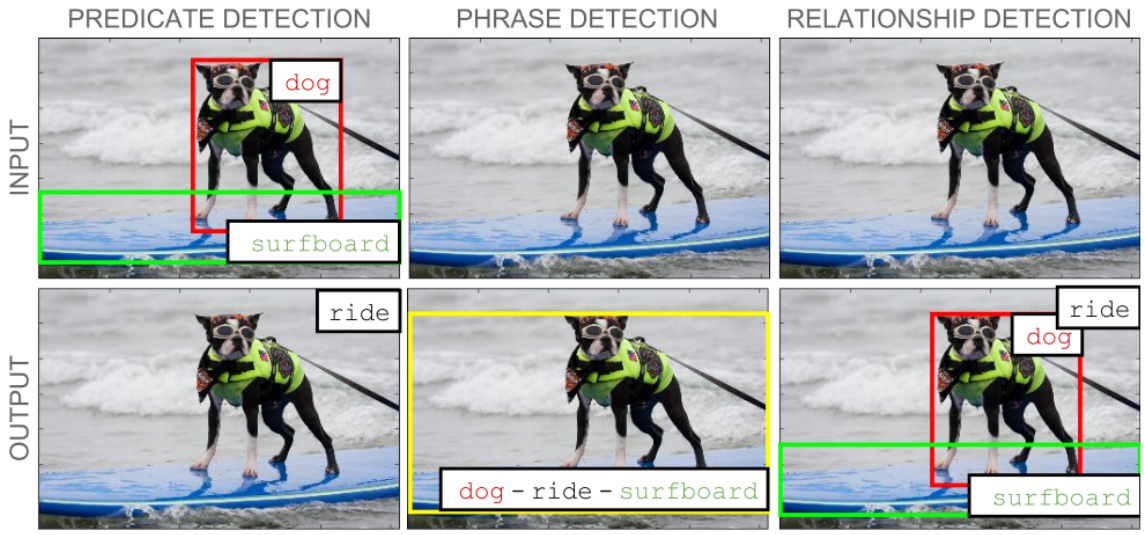


Figure 4.4 – **VRD settings.** Illustration of the different evaluation settings in VRD. This illustration was taken from (C. Lu et al. 2016).

BLOCK with respect to hyperparameter variations. Finally in Section 4.4.3, we show that our system performs well against other leading VRD methods.

4.4.1 VRD Architecture

Our VRD architecture is shown in Figure 4.5. It takes as inputs a subject and an object bounding box. Each of them is represented as their 4-dimensional box spatial coordinates $\boldsymbol{x}_s^s$ and $\boldsymbol{x}_o^s$ (normalized between 0 and 1), their object class embeddings $\boldsymbol{x}_s^c$ and $\boldsymbol{x}_o^c$, and their semantic visual features $\boldsymbol{x}_s^f$ and $\boldsymbol{x}_o^f$. To predict the relationship predicate, we use one fusion module for each type of features following Equation 4.32.

$$\boldsymbol{x} = [f^s\left(\boldsymbol{x}_s^s, \boldsymbol{x}_o^s\right), f^c\left(\boldsymbol{x}_s^c, \boldsymbol{x}_o^c\right), f^f\left(\boldsymbol{x}_s^f, \boldsymbol{x}_o^f\right)] \tag{4.32}$$

where f can be implemented as BLOCK, or any other multimodal fusion. Each fusion module outputs a vector of dimension d, all concatenated into a $3d$-dimensional vector that will serve as an input to a linear layer predictor $\boldsymbol{y} = \boldsymbol{W}\boldsymbol{x}$. The system is trained with back-propagation on a binary-crossentropy loss.

Another important component of the VRD system is the object detector. We first train a Faster-RCNN on the object boxes of the VRD Dataset. For Predicate prediction, we use it as a features extractor to obtain the $\boldsymbol{x}^f$ vectors for ground truth bounding boxes. For Phrase detection and Relationship detection, we use it to extract the bounding boxes with their associated features.

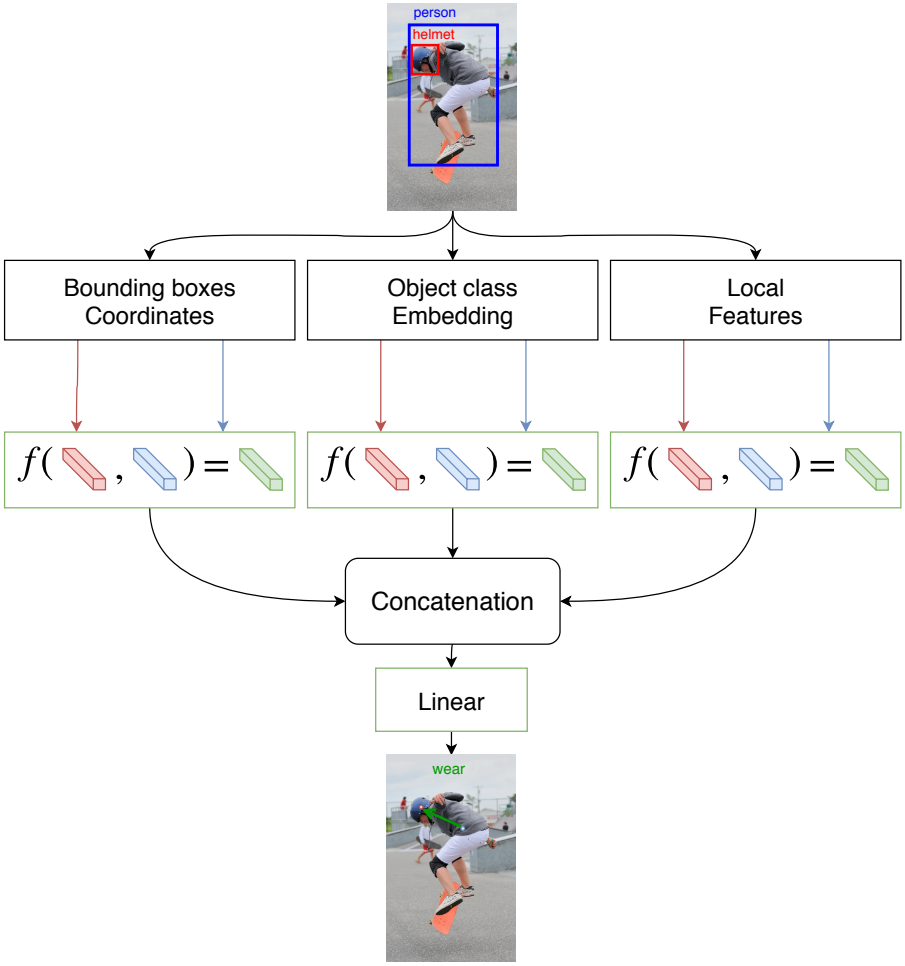


Figure 4.5 – **Architecture for VRD.** Our very simple architecture for VRD classifies a pair of regions by representing three types of interaction between regions: spatial coordinates, visual features and object class. For each of these three types of signals, interactions are modeled with a BLOCK layer.

Negative pairs. The VRD Dataset is composed of localized pairs of subject-object regions, associated to a class relationship label. Importantly, the annotations do not contain *negative pairs* of objects that are not linked together by any relation. For the task of Predicate prediction, the test data consists in *true* input pairs, for which we ask the model to select a predicate among the classes. Thus, the absence of negative pairs in training is not a problem. However, for Phrase and Relationship detection, the input regions in the test phase are predictions of an object detector. In this case, the VRD model has to predict a predicate for all the pairs of detected objects. For these two tasks, it is important that the model learns also on negative pairs. This is why for each train image, we randomly sample

	Description	$\lvert\Theta\rvert$	Predicate Prediction		Phrase Detection		Relationship Detection	
			R@50	R@100	R@50	R@100	R@50	R@100
(1)	Linear	2.5M	82.99	89.68	14.44	16.94	9.73	11.34
(2)	Non-linear	2M	84.47	91.6	21.9	24.69	15.79	17.83
(3)	B + count-sketching	2M	82.23	89.07	13.42	15.8	9.17	10.79
(4)	B + Tucker decomp.	3M	83.25	89.77	11.23	14.09	7.37	9.00
(5)	B + CP decomp.	4M	85.96	91.66	23.67	26.50	16.41	18.59
(6)	B + 3rd mode rank	15M	85.21	91.06	25.31	28.03	17.83	19.77
(7)	(4) and (6)	30M	85.65	91.33	25.77	28.65	18.53	20.38
(8)	Higher order fusion	16M	85.58	91.3	26.09	28.73	18.81	20.63
(9)	Block-term decomp.	5M	**86.58**	**92.58**	**26.32**	**28.96**	**19.06**	**20.96**

Table 4.4 – **Fusion scheme comparison on VRD.** Comparative study of the different multimodal fusion strategies on the VRD test-set. The reported metrics are the Recall@K in %.

half of all possible pairs that are not assigned to any label and assign them an to all-zeros label vectors. We add this set of *negative pairs* to the positive data pairs.

4.4.2 Fusion analysis

To show the effectiveness of the BLOCK bilinear fusion, we run the same type of experiment we did in the previous section. For each fusion technique, we use the architecture described in Equation 4.32 where we replace f by the corresponding function. We cross-validate the hyperparameters of each fusion technique and keep the best model each time. In Table 4.4, we see that BLOCK still outperforms all previous methods on each of the three tasks. We can remark that for this task, the non linear MLP perform relatively well compared to the other methods. It is likely that an MLP can model the interactions at stake for VRD more easily than those for VQA. However, we can improve over this strong baseline using a BLOCK fusion.

In the next experiments, we validate the power of our BLOCK fusion, and analyze how it behaves under different setups. We randomly split the training set into three train/val sets, and plot the mean and standard deviation of the Recall@50 calculated over them. In Figure 4.6a, we freeze the dimension of the block-superdiagonal tensor to $RL = RM = RN = 500$ and vary the number of blocks used to fill this tensor. When $R = 1$, which corresponds to the Tucker decomposition, the number of parameters in the core tensor is equal to $500^3 = 125M$, making the system arduously trainable on our dataset. On the opposite, when

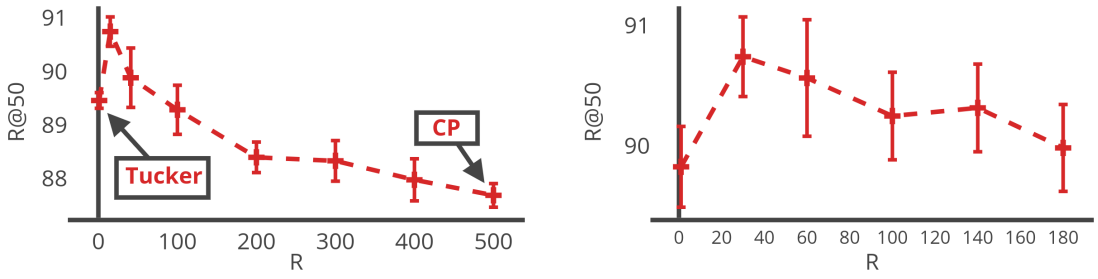


Figure 4.6 – **Performance of analysis of BLOCK.** We analyze how the performance of BLOCK is impacted as we change some structural parameters of the super-diagonal tensor. Models are trained on the VRD Dataset *train*, minus a holdout validation set over which we report the scores. Figure 4.6a: The size of the core tensor is fixed to $RL = RM = RN = 500$. Figure 4.6b: The total number of parameters in the block-diagonal tensor is fixed to $555K$.

$R = 500$, the number of parameters is controlled, but the mono-modal projections are only allowed to interact through an element-wise multiplication, which makes the interaction modeling relatively poor. The block-term decomposition provides an in-between working regime, reaching an optimum when $R \approx 20$.

In Figure 4.6b, we keep the number of parameters fixed. As the number of chunks increases, the dimensions of the mono-modal projections also increases. Once again, an optimum is reached when $R \approx 20$. These results confirm our hypothesis that the way the parameters are distributed within the tensor, in terms of size and number of blocks, has a real impact on the system's performance.

4.4.3 Comparison to leading VRD methods

In Table 4.5, we compare our system to the state-of-the-art methods on VRD. On predicate prediction, our fusion outperforms all previous methods on R@50, including (Ruichi Yu et al. 2017) that uses external data. On R@100, the BLOCK fusion is only marginally outperformed by (Ruichi Yu et al. 2017), but we perform better than all methods that don't use extra data. These results validate the efficiency of the block-term decomposition to predict a predicate by fusing information coming from ground truth subject and object boxes. On phrase detection, our BLOCK fusion achieves better results than all previous models in R@50. Notably, the scores obtained for phrase detection are lower than for predicate prediction, since the ground truth regions are not provided in this setup. Finally, on relationship detection, BLOCK surpasses all previous methods without extra data in R@50, and gives similar performance than (Dai et al. 2017) in R@100.

Model	External data	Predicate Prediction		Phrase Detection		Relationship Detection	
		R@50	R@100	R@50	R@100	R@50	R@100
(Ruichi Yu et al. 2017)	✓	85.64	94.65	26.32	29.43	22.68	31.89
(Y. Li et al. 2017)	✗	-	-	22.78	27.91	17.32	20.01
(Liang et al. 2017)	✗	-	-	21.37	22.60	18.19	20.79
(H. Zhang et al. 2017a)	✗	44.76	44.76	19.42	22.42	14.07	15.20
(C. Lu et al. 2016)	✗	47.87	47.87	16.17	17.03	13.86	14.70
(Peyre et al. 2017)	✗	52.6	52.6	17.9	19.5	15.8	17.1
(Dai et al. 2017)	✗	80.78	81.90	19.93	23.45	17.73	20.88
BLOCK	✗	**86.58**	**92.58**	**26.32**	**28.96**	**19.06**	**20.96**

Table 4.5 – **Comparison to previous work on VRD Dataset.** Results are reported on the VRD test set.

The scores for relationship detection are lower than for phrase detection: in this setup, a prediction is positive if both subject and object boxes match the ground truth. On contrary, in phrase detection, the comparison between prediction and ground truth is done on the union between subject and object regions. Lastly, unlike some of the methods reported in Table 4.5, we do not fine-tune or adapt the detection network to the visual relationship tasks.

4.5 Conclusion

In this chapter, we present our second strategy (BLOCK) for multi-modal fusion. In the continuity of our work in Chapter 3, we develop a bilinear model structured using the block-term decomposition. Through its block-diagonal structure, it offers the possibility to extract the relevant correlations between high dimensional mono-modal projections. Moreover, BLOCK optimizes the trade-off between complexity and modeling capacities through the *block-term ranks* of the tensor, which explicitly controls the sparsity in the core tensor. Interestingly, this algebraic concept encompasses both notions of *rank* and *mode-ranks*, thus combining the strengths of the CP and Tucker decompositions.

Building on the visual attention framework and on the recently introduced bottom-up image representation, our system compares favorably with leading VQA methods. To further validate our fusion, we experimentally compare BLOCK to 8 different multi-modal fusion models, showing high performance and parameter efficiency.

Besides, we demonstrate the genericity of our method by designing a simple yet effective architecture for VRD which is based on multi-modal fusion. We show that BLOCK compares well to other fusion methods in VRD, in a similar way as in VQA. Additionaly, we experimentally analyze the dependancy of BLOCK with respect to structural parameters of the decomposition.

In the next chapter, we question the relevance of the multi-glimpse attention architecture that learns multiple question-guided region selection functions in parallel. We develop an iterative model that progressively learns to refine each local representation, mimicking some type of multi-step reasoning.

CHAPTER 5

MUREL: MULTIMODAL RELATIONAL REASONING FOR VQA

Contents

Chapter abstract

In this chapter, we focus on the visual reasoning problem in Visual Question Answering (VQA). We move away from the classical multi-glimpse attention architecture, proposed in Chapter 3 and Chapter 4, to build an iterative scheme that reasons about the visual scene. Our architecture, called MuRel, relies on our work on multi-modal fusion to merge question embedding with each local visual representation. Additionally, we model pairwise relations between regions to make their representation context-aware. We evaluate our model on three recent and challenging datasets for VQA.

The work in this chapter, at equal contribution with Rémi Cadène, has led to the publication of a conference paper:

- Hedi Ben-Younes*, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2019). "MUREL: Multimodal Relational Reasoning for Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).*

5.1 Introduction

In the previous chapters, we studied how image and language representations could be fused together in an end-to-end learning fashion. Using tensor decompositions, we developped efficient multi-modal techniques capable of representing the relevant correlations between two vector spaces, corresponding to visual and textual embeddings. In addition to these fusion problems, the visual reasoning architecture is also crucial for Visual Question Answering (VQA). As developped in Section 2.4, a successful VQA model needs to extract the relevant information from the image, understand the objects in presence, the attributes that characterize them, the relations between them, their spatial layout, *etc*. This analysis has to be conducted under the light of a given natural language question, and guided towards the goal of providing an answer.

There are two widely used approaches on which we focus for this thesis. The first one is the multi-glimpse attention, which computes several question-guided visual attention maps in parallel, independantly from each other (see Figure 5.1). This is the approach we used to build the VQA architectures in Chapter 3 and Chapter 4. In this chapter, we are interested in the second type of approaches, where the visual reasoning is built iteratively. Indeed, complex questions may require that image and text modalities interact multiple times, each step providing intermediate results that condition the behaviour of the next. An illustration of such iterative process is shown in Figure 5.2, where the attention modules are structured in a chain. Each module attends over the image regions with respect to a *context* vector, which is provided by the previous attention layer. The first context vector is set to be the question embedding. Passing attention information from layer to layer makes each attention module aware of what has *already been seen* by the model. This type of multi-step reasoning scheme is at the basis of the Stacked Attention Network (SAN) that we detail in Section 2.4.1.

In this chapter, we develop a VQA architecture that falls within this family of iterative visual reasoning. We first present the **MuRel cell**, an atomic reasoning primitive that enables to represent rich interactions between a question and a set of image regions. It incorporates the question information into an ensemble of localized visual representations using a bilinear fusion scheme. Moreover, we model pairwise relations between region vectors to make each region aware of its context. We then embed this MuRel cell into the **MuRel network**, an iterative reasoning process which progressively refines the internal network representations to answer the question. The rationale of MuRel is illustrated in Figure 5.3: for the question "what is she eating", our model focuses on two main regions (the head and the donut) with important visual cues and semantic relations between them to provide the correct answer "donut". The visual reasoning process that our MuRel system embodies is formed by this multi-step relational module that discards useless information to focus on the relevant regions.

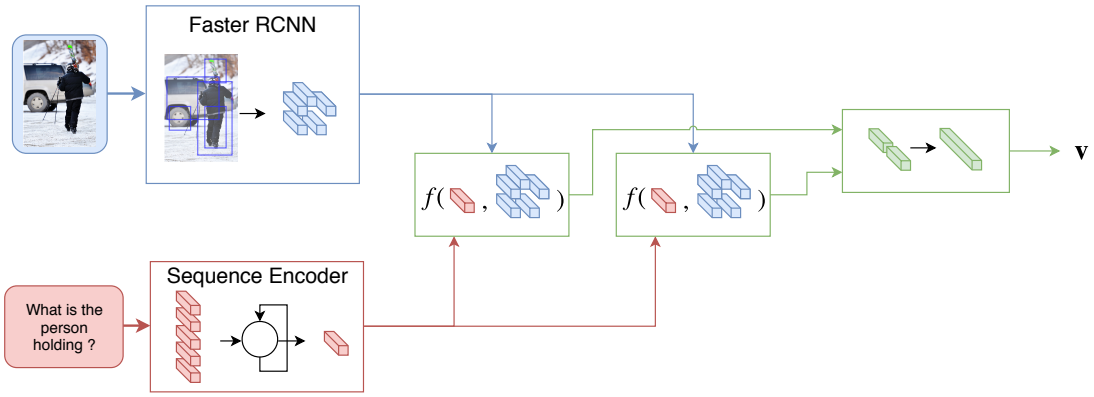


Figure 5.1 – **Multi-glimpse attention.** In this figure, two attention modules process the set of region vectors with respect to the question. Each of these glimpses provides a question-based image representation, both computed in parallel. Finally, the outputs of the glimpses are concatenated to provide the final image representation.

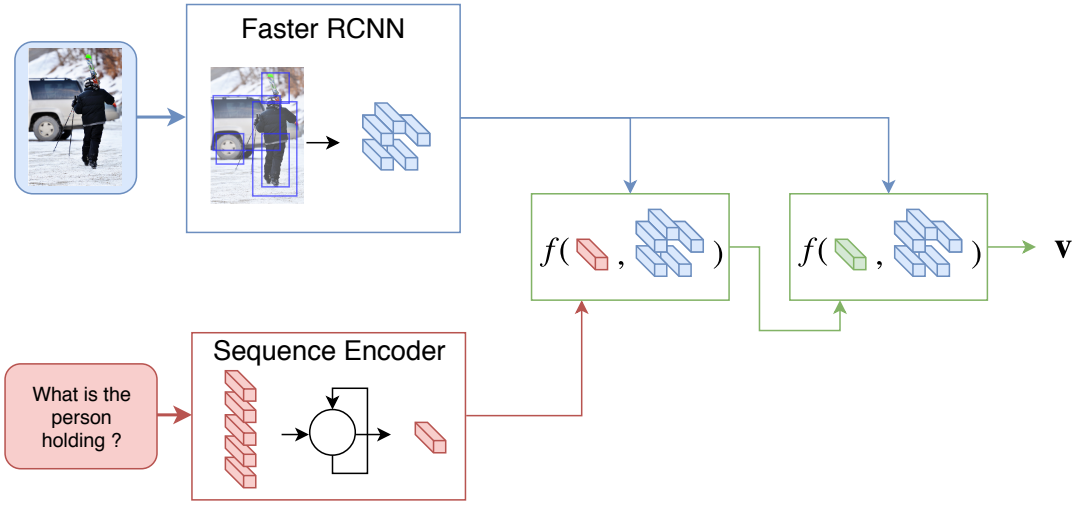


Figure 5.2 – **Stacked attention.** Here, the two modules are inter-dependant. The first module conditions its attention on the question embedding, As for the following ones, their context vector is directly the output of the previous module. This allows the system to perform iterative reasoning, each time looking at the image while knowing what has already been seen.

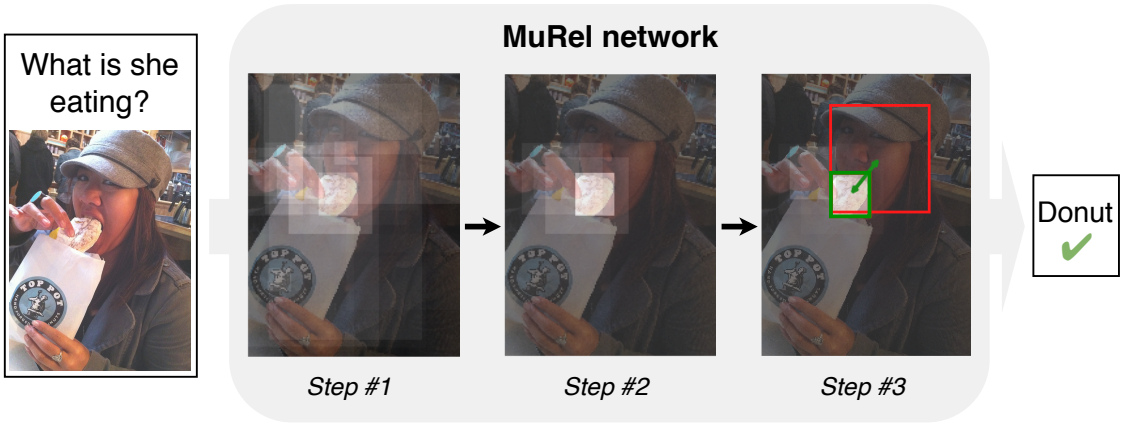


Figure 5.3 – **Visualization of the MuRel approach.** Our MuRel network for VQA is an iterative process based on a rich vectorial representation between the question and visual information explicitly modeling pairwise region relations. MuRel is thus able to express complex analysis primitives beyond attention maps: here the two regions corresponding to the head and the donuts are selected based on their visual cues and semantic relations to properly answer the question "what is she eating?".

In Section 5.2 we introduce the different components of our visual reasoning architecture for VQA. We present the MuRel cell in Section 5.2.1, a neural module that learns to perform elementary reasoning operations by blending question information into the set of spatially-grounded visual representations. Next, in Section 5.2.2, we leverage the power of this cell using the MuRel network, a VQA architecture that iterates through a MuRel cell to reason about the scene with respect to a question. Finally, we experimentally demonstrate the effectiveness of the MuRel approach in Section 5.3, both quantitatively and qualitatively.

5.2 MuRel approach

The MuRel approach is complementary to the points explored in the previous chapters. We do not focus on the multi-modal fusion of vectors but on exploiting these fusion modules to build a higher level architecture. In particular, in Chapter 3 and Chapter 4, we adopted the standard multi-glimpse attention approach as it had already demonstrated positive results in previous work. The approach developped in this chapter is based on an iterative modeling of the visual reasoning for VQA.

Notations. The image is represented by a set of vectors $\{\boldsymbol{v}_i\}_{i\in[1,N]}$, where each $\boldsymbol{v}_i \in \mathbb{R}^{d_v}$ corresponds to the visual representation of a detected object.

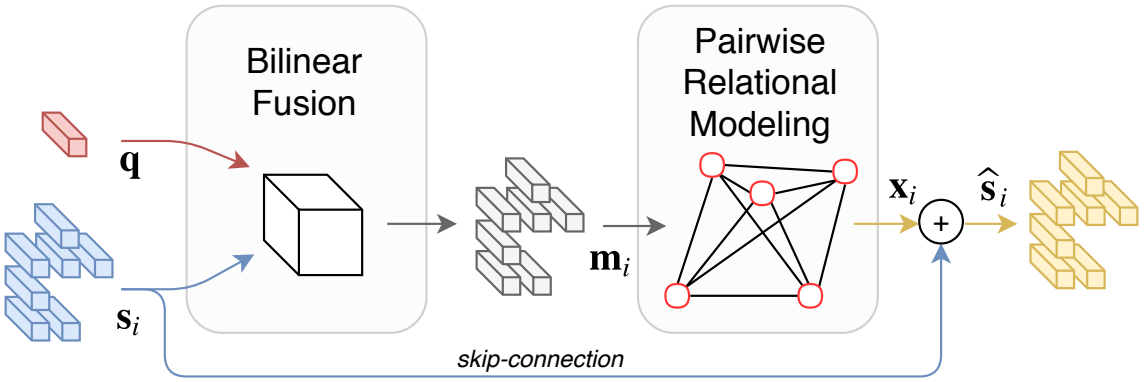


Figure 5.4 – **MuRel cell.** In the MuRel cell, the bilinear fusion represents rich and fine-grained interactions between question and region vectors $\boldsymbol{q}$ and $\boldsymbol{s}_i$. All the resulting multi-modal vectors $\boldsymbol{m}_i$ pass through a pairwise modeling block to provide a context-aware embedding $\boldsymbol{x}_i$ per region. The cell's output $\hat{\boldsymbol{s}}_i$ is finally computed as a sum between $\boldsymbol{s}_i$ and $\boldsymbol{x}_i$, acting as residual function of $\boldsymbol{s}_i$.

These representations are provided by the bottom-up visual features, presented in Section 2.2.1. Additionally, we use the spatial coordinates of each region $\boldsymbol{b}_i = [x, y, w, h]$, where (x, y) are the coordinates of the top-left point of the box, and h and w correspond to the height and the width of the box. Note that x and w (respectively y and h) are normalized by the width (resp. height) of the image. For the question representation, we the same encoder as in Chapter 4, which is based on a Gated Recurrent Unit (GRU) network with self-attention. It provides a sentence embedding $\boldsymbol{q} \in \mathbb{R}^{d_q}$.

5.2.1 MuRel cell

The MuRel cell takes as input a bag of N visual features $\boldsymbol{s}_i \in \mathbb{R}^{d_v}$, along with their bounding box coordinates $\boldsymbol{b}_i$. The link between these visual features $\boldsymbol{s}_i$ and the representations given by the object detector $\boldsymbol{v}_i$ will be made clear in Section 5.2.2. As shown in Figure 5.4, it is a residual function consisting of two modules. First, an efficient bilinear fusion module merges question and region feature vectors to provide a local multi-modal embedding. This fusion is directly followed by a pairwise modeling component, designed to update each multi-modal representation with respect to its own spatial and visual context.

Multi-modal fusion The first layer of our MuRel cell is designed to merge the question information within each visual representation $\boldsymbol{s}_i$, and to model the relevant interactions between both modalities. To do so, we use the efficient bilinear fusion strategy developped in Chapter 4, which is based on the block-

term decomposition of tensors. This bilinear fusion model learns to focus on the relevant correlations between input dimensions, while keeping a relatively low number of parameters. Each input vector $\boldsymbol{s}_i$ is fused with the question embedding $\boldsymbol{q}$ using the same BLOCK module:

$$\boldsymbol{m}_i = \mathrm{B}\left(\boldsymbol{s}_i, \boldsymbol{q}; \Theta\right) \tag{5.1}$$

where Θ are the trainable parameters of the fusion module. We set the number of dimensions in $\boldsymbol{m}_i$ to d_v to facilitate the use of residual connections throughout our architecture.

Each dimension m of $\boldsymbol{m}_i$ can be written as a bilinear function of the form

$$\boldsymbol{m}_i[m] = \sum_{s,q} \mathcal{T}[s,q,m]\boldsymbol{s}_i[s]\boldsymbol{q}[q] \tag{5.2}$$

In this bilinear model, the tensor $\mathcal{T}$ is factorized into the list of parameters Θ that consist in the mono-modal projections and the block-sparse third-order tensor, which are the elements of the block-term decomposition. Please refer to Chapter 4 for more details.

In classical attention models, the fusion between image region $\boldsymbol{s}_i$ and question features $\boldsymbol{q}$ learns to predict a saliency score, which encodes whether or not a region is relevant with respect to a given question. On contrary, the MuRel cell represents local multi-modal information within a richer vectorial form $\boldsymbol{m}_i$ which can encode more complex correlations between both modalities. This allows to store more specific information about what precise characteristic of a particular region is important in a given textual context.

Pairwise interactions Answering certain types of question may require analyzing multiple objects and their mutual interactions. This is necessary, for example, to answer questions about relative spatial positioning or semantic relationships. Modeling relations between objects within a VQA architecture is generating an increasing interest among the research community (see Section 2.4.3). A region should be represented by a vector that is aware of the spatial and semantic contexts in which it appears. As we use the bottom-up features, which are structured as a bag of localized vectors (see Section 2.2.1), modeling the visual context of each region is not straightforward. Similarly to the recent work of (Norcliffe-Brown et al. 2018), we opt for a pairwise relationship modeling where each region receives a message based on its relations to its neighbours. In their work, the neighbours of a region correspond to the K most similar regions, whereas in the MuRel cell the neighbourhood is composed of every region in the image. Besides, instead of using scalar pairwise attention and graph convolutions with Gaussian kernels as they do, we merge spatial and semantic representations to build relationship vectors.

For every region i, we compute a context vector $\boldsymbol{e}_i$ as an aggregation of all the pairwise links $\boldsymbol{r}_{i,j}$ coming into i. We define it as $\boldsymbol{e}_i = \max_j \boldsymbol{r}_{i,j}$, where $\boldsymbol{r}_{i,j}$ is a vector containing information about the content of both regions, but also about their relative spatial positioning. We use the element-wise max operator in the aggregation function to reduce the noise that can be induced by average or sum poolings, which oblige all the regions to interact with each other. To encode the relationship vector, we use the following formulation:

$$\boldsymbol{r}_{i,j} = \mathrm{B}\left(\boldsymbol{b}_i, \boldsymbol{b}_j; \Theta_b\right) + \mathrm{B}\left(\boldsymbol{m}_i, \boldsymbol{m}_j; \Theta_m\right) \tag{5.3}$$

Note that this formulation is very close to the one we use in Equation 4.32 to detect visual relationships. Through the $\mathrm{B}(.,.;\Theta_b)$ operator, the cell is free to learn spatial concepts such as *on top of, left, right, etc.* In parallel, $\mathrm{B}(.,.;\Theta_s)$ encodes correlations between multi-modal vectors $(\boldsymbol{s}_i, \boldsymbol{s}_j)$, each corresponding to a fusion between locally-grounded semantic representations and the question embedding. By summing up both spatial and semantic fusions, the network can learn high-level relational concepts such as *wear, hold, etc.*

For every region, the vector $\boldsymbol{e}_i$ encodes the context in which the region appears as an aggregation of messages $\boldsymbol{r}_{i,j}$ provided by its neighbours. It additively updates the multi-modal vector $\boldsymbol{m}_i$ to provide a local context-aware multi-modal representation $\boldsymbol{x}_i$:

$$\boldsymbol{x}_i = \boldsymbol{m}_i + \boldsymbol{e}_i \tag{5.4}$$

This formulation of the pairwise modelling is close to the Graph Networks (Battaglia et al 2018), where the notion of relational inductive biases is formalized.

Finally, the output of the MuRel cell is computed as a residual function of its input, to avoid the vanishing gradient problem. Each visual features vector $\boldsymbol{s}_i$ is updated following the rule: $\hat{\boldsymbol{s}}_i = \boldsymbol{s}_i + \boldsymbol{x}_i$.

The chain of operations that updates the set of localized region embeddings $\{\boldsymbol{s}_i\}_{i\in[1,N]}$ using the multi-modal fusion with $\boldsymbol{q}$ and the pairwise modeling operator is noted:

$$\{\hat{\boldsymbol{s}}_i\} = \mathrm{MurelCell}\left(\{\boldsymbol{s}_i\}; \{\boldsymbol{b}_i\}, \boldsymbol{q}\right) \tag{5.5}$$

5.2.2 MuRel network

To mimick a simple form of multi-step reasoning, we embed the MuRel cell into an iterative process which merges the question information into context-aware visual embeddings. As we can see in Figure 5.5, a MuRel cell passes multiple times over the region vectors $\{\boldsymbol{s}_i\}$, each time refining the representations with contextual and question information. For each step $t = 1..T$ where T is the total number of steps fixed beforehand, a MuRel cell processes and updates the state vectors following Equation 5.6:

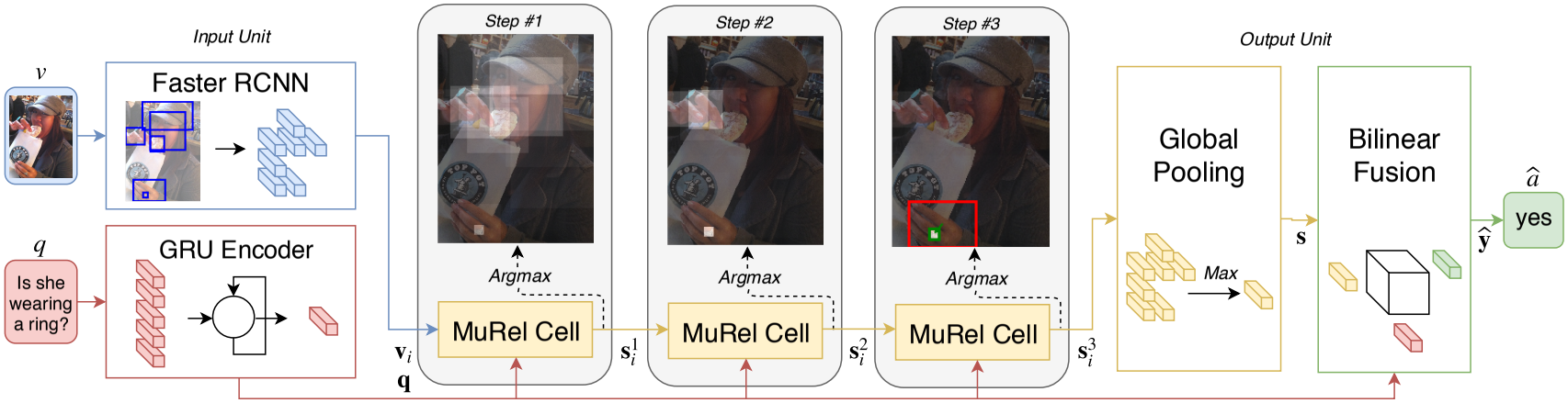


Figure 5.5 – **MuRel network.** The MuRel network merges the question embedding $\boldsymbol{q}$ into spatially-grounded visual representations $\{\boldsymbol{v}_i\}$ by iterating through a single MuRel cell. This module takes as input a set of localized vectors $\{\boldsymbol{s}_i\}$ and updates their representation using a multi-modal fusion component. Moreover, it models all the possible pairwise relations between regions by combining spatial and semantic information. To construct the importance map at step t, we count the number of time each region provides the maximal value of $\max_i\{\boldsymbol{s}_i^t\}$ (over the 2048 dimensions).

$$\{\boldsymbol{s}_i^t\} = \text{MurelCell}\left(\{\boldsymbol{s}_i^{t-1}\}; \{\boldsymbol{b}_i\}, \boldsymbol{q}\right) \tag{5.6}$$

The state vectors are initialized with the features coming from the feature extraction step. For each region i, $\boldsymbol{s}_i^0 = \boldsymbol{v}_i$.

The MuRel network represents each region regarding the question, but also using its own visual context. This representation is done iteratively, through multiple steps of a MuRel cell. The residual nature of this module makes it possible to align multiple cells without being subject to gradient vanishing. Moreover, the weights of our model are shared across the cells, which enables compact parametrization and good generalization.

At the final step $t = T$, the representations $\{\boldsymbol{s}_i^T\}$ are aggregated with a element-wise max pooling operation to provide a single vector

$$\boldsymbol{s} = \max_i \boldsymbol{s}_i^T \in \mathbb{R}^{d_v} \tag{5.7}$$

This scene representation contains information about the objects and their relations, spatial and semantic, with respect to a given question. A score for every possible answer is computed as a fusion between $\boldsymbol{s}$ and the question vector $\boldsymbol{q}$: $\hat{\boldsymbol{y}} = \text{B}\left(\boldsymbol{s}, \boldsymbol{q}; \Theta_y\right)$. This vector is then normalized using a softmax, which provides a distribution $f_\Theta(v, q)$ over possible answers, conditionned on an image and a question. Finally, $\hat{a}$ is the answer with maximum score in $\hat{\boldsymbol{y}}$.

Visualizing MuRel network. We can use the architecture of our model to define visualiztion schemes that are finer than mere attention maps. At the end of the processing done by the MuRel network, the region features $\{\boldsymbol{s}_i^T\}$ are aggregated using a max operation, yielding a d_v-dimensional vector $\boldsymbol{s}$. Associated to this aggregation, we compute what we call a *contribution map* that measures to which extent each region contributes to the final vector. In parallel to the max, we also keep the region that provides this maximal value for each dimension. This information is stored in $\boldsymbol{c}^T = \text{argmax}_i\{\boldsymbol{s}_i^T\} \in [1, N]^{d_v}$. We can then measure the occurrence frequency of each region in this vector $\boldsymbol{c}$. This frequency provides an estimation of the contribution of each reagion to the final vector $\boldsymbol{s}$. Interestingly, the vector that contains argmax values can be computed at each step t, and not exclusively at the last one. Intuitively, calculating $\boldsymbol{c}^t$ for $t < T$ measures what the contribution map would have been if the iterative process had stopped at this point. In other words, it indicates which regions are considered important by the model at a given reasoning step. As we can see in Figure 5.3,Figure 5.5 and Figure 5.6, these relevance scores match human intuition and can be used to explain the decisions of the model. Importantly, we recall that the network has not been trained with any kind of selection mechanism, and that these visualization are inferred from the internal activations of the model.

We develop a similar technique to visualize the pairwise relationships that are involved in the predictions of the model. At a given reasoning step t, we first identify the region $i^\star$ which is the most impacted by the pairwise modeling. We define it as the region with maximal $\|\frac{\boldsymbol{e}_i}{\boldsymbol{x}_i}\|_2$ (cf. Equation 5.4). In all our visualizations, this bounding box is shown in green. We then measure the contribution of every other vector to this region $i^\star$, again using occurrence frequencies in a vector of argmax values $\boldsymbol{c}_{i^\star} = \text{argmax}_j \boldsymbol{r}_{i^\star,j}$. We show in red the regions whose contribution to $i^\star$ is above a certain threshold (0.2 in our visualizations). If there is no such region, the green box is not shown.

Connection to FiLM network. We can draw a comparison between our MuRel network and the FiLM network proposed in (Perez et al. 2018). Beyond the fact that their model is built for the synthetic CLEVR Dataset (Johnson et al. 2017a) and ours processes real data, some connections can be found between both models. In their work, the image passes through multiple residual cells with different parameters, whereas we only have one cell through which we iterate. In FiLM, the multi-modal interaction is modeled with a feature-wise affine modulation, while we use the BLOCK strategy that we developped in Chapter 4, which is more efficient. Finally, both MuRel and FiLM leverage the spatial structure of the image representation to model the relations between regions. In FiLM, the image local features are disposed in a fixed spatial grid, as the visual representation is provided with a Fully Convolutional Network (FCN). This structure on image features encourages to model relations between regions through the neighbourhood structure induced by the spatial grid. In FiLM, this is done by having 3×3 convolution inside each of their residual blocks. The representation of each region then depends on its closest neighbours, defined by the locally-connected graph of the grid structure. In our MuRel network, the image is represented as a set of localized features. Vectors are not disposed in a fixed grid but in arbitrary positions that vary from an image to another. This makes the relational modeling not trivial. As we want to model relations between regions that are potentially far apart, we consider that the set of regions forms a complete graph, where each region is connected to all the others.

5.3 Experiments

5.3.1 Experimental setup

Datasets Our experiments are conducted on three recent datasets The most commonly used is certainly the VQA 2.0 Dataset (Goyal et al. 2017). As explained in Section 2.5, it comes with a *train* set, a *val* set and an online *test* set. For the fine grained analysis of MuRel, we train the models on the *train* split and provide

the results on the *val* split, whereas. To compare MuRel against state-of-the-art approaches, we train on the concatenation of *train* and *val* and report the results on the *test* set. To measure the generalization capacities of MuRel, we evaluate its performance on VQA Changing Priors v2 (VQA-CP) (Agrawal et al. 2018). VQA-CP v2 uses the same data as the VQA 2.0 Dataset, with a different *train/test* split. In particular, both splits are explicitly set to have different conditionnal distributions of answer on question-types, which makes it a very challenging dataset. Finally, we use the TDIUC dataset (Kafle et al. 2017) to construct a detailed analysis of the behaviour of MuRel on 12 well-defined types of question. TDIUC is currently the biggest dataset for VQA.

Setup The architecture setup is similar to the one we use in Chapter 4. The image is represented as the set of 36 bottom-up features provided by (Anderson et al. 2018), and the question by a pre-trained GRU on the Skip-thought task. Inspired by the recent work of (Jiang et al. 2018), we use Adam as optimizer (Kingma et al. 2014) with a warm-up learning scheduler. Unless we explicitly state otherwise, the number of iterations through the MuRel cell is set $T = 3$.

5.3.2 Qualitative results

Following the method we explain in Section 5.2.2, we are able to visualize the behaviour of a MuRel network. Here, we train a MuRel network with 3 cells on the *train* split of VQA 2.0 Dataset and visualize it on examples taken from the unseen *val* set. In Figure 5.6, we highlight the regions that contribute the most to the global scene representation, and show in green and red those that are the most involved in the pairwise modeling. Both region contributions and pairwise links match human intuition. On the first line, the most relevant relations according to our model are between the player's hand, which contains the WII controller, and the screen, which contains information about the true answer. In the third line, the model answered *kite* using the relation between the man's hand and the kite he is holding. Finally, these visualizations help understand some failure cases of the model. In the last line, the feature extraction gives two boxes around the green jacket on the left. Our model is confused by this double detection and mistakenly believes that "the jacket on the right" is the rightest detection of the left jacket. This type of problem could be alleviated if we modified our model by taking inspiration from (Y. Zhang et al. 2018), which learns to correct double detections.

The visualization shown in Figure 5.7 illustrates the behaviour of MuRel for multiple questions on a single image. As we can see, the contribution maps heavily depend on the textual modality. The common point of all the questions is that they involve objects related to the woman. Each time, MuRel highlights the important object, as well as the most relevant pairwise link.

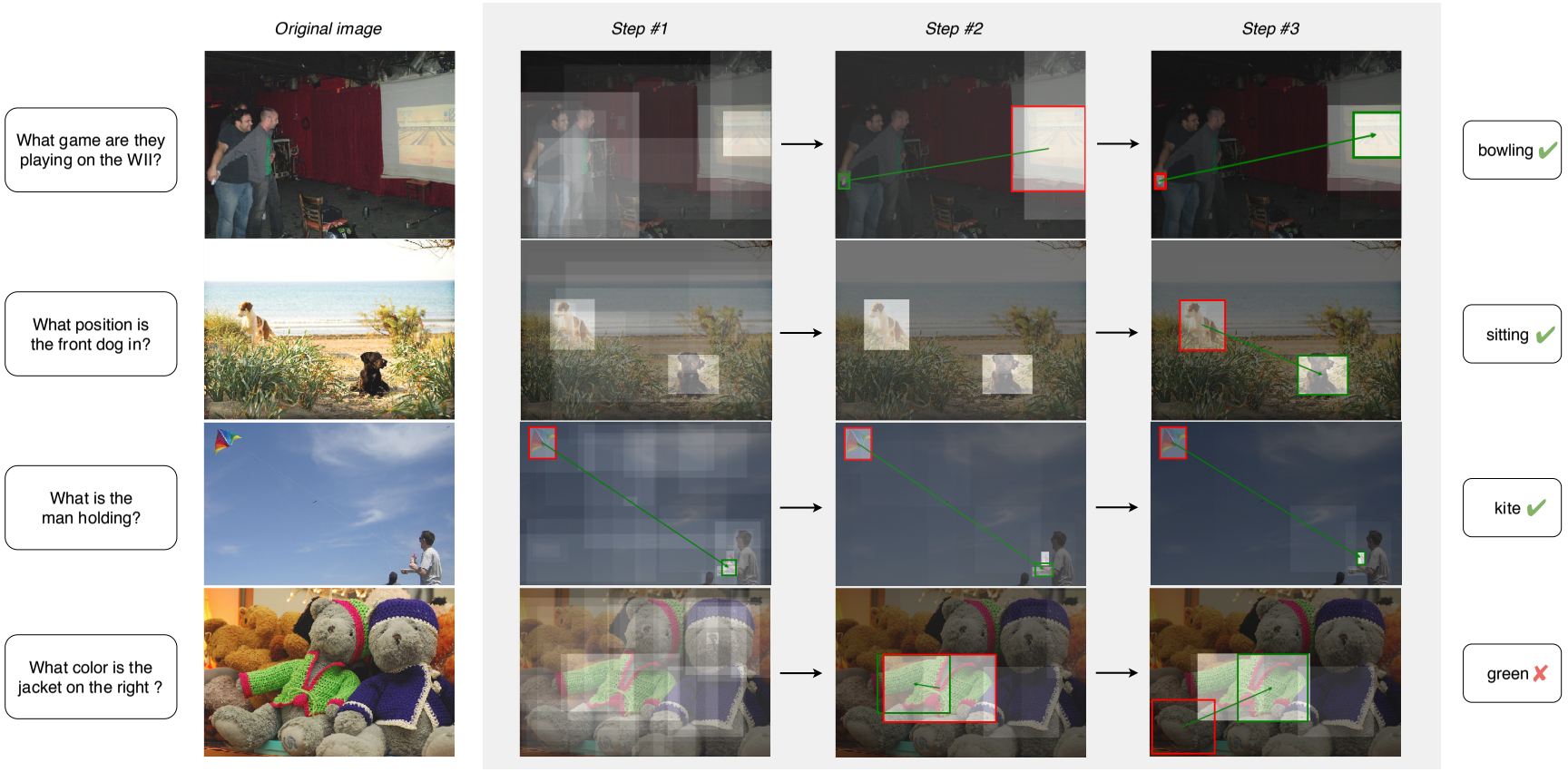


Figure 5.6 – **Qualitative evaluation of MuRel**. Visualization of the importance maps with colored regions related to the relational mechanism. As in Figure 5.5, the most selected regions by the implicit attentional mechanism are shown in brighter. The green region is the most impacted by the pairwise modeling, while the red regions impact the green regions the most. These colored regions are only represented if they are greater than a certain threshold.

Figure 5.7 – **Impact of the question on contribution maps**. We observe that for a given image, different questions assign different contribution maps for regions and pairwise relations. They are on par with an intuitive idea of what could be useful to answer each question.

Not only does the model provide an answer, it also gives information about objects and pairs of regions it found the most helpful to provide this answer. As VQA models are often subject to linguistic bias (Goyal et al. 2017; Agrawal et al. 2018), this type of visualization help increasing the reliability of a prediction, and is a step towards more interpretable models.

Entropy of the implicit attention. Interestingly, we remark in Figure 5.6 that iterations through the MuRel cell tend to gradually discard regions, keeping only the most relevant ones. We want to quantify this phenomenon, and study the behaviour of the MuRel network regarding its implicit region selection process. During training, we compute the contribution maps of every instance of the dataset. We recall that this map provides a score for each region bounded between 0 and 1, such that they all sum to 1. Thus, it can be viewed as a probability distribution over regions. As we want to measure to which extent the contribution maps are focused around a few regions, we choose to compute the entropy of the distributions in the regions contribution. Low values mean that the contribution process is peaked around very few regions, and high values denote a uniform repartition of the contribution over regions. In Figure 5.8, we plot the mean entropy of the region contributions in the *train* and *val* splits during training at

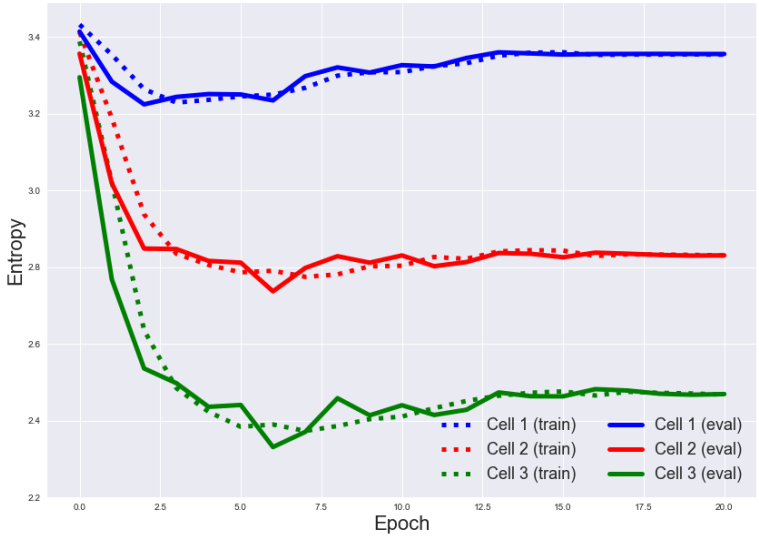


Figure 5.8 – **Entropies of the implicit attention of each cell during training.**

each epoch. We observe that entropies converge to different levels depending on the cell. The value of this level is similar for both *train* and *val* sets. Moreover, we see that the higher entropy is reached by the first cell, and it decreases as we iterate through the cell. Intuitively, the network is more and more selective as the representations flow towards the top of the model. It could be interesting to deeper study the effects of this phenomenon. In particular, we did not measure whether the regions selected by a cell are a subset of the previous ones or if they are different. Besides, it is not clear to which extent this increasing selectivity is desirable, if we should encourage this behaviour, or on contrary prevent it from happening.

5.3.3 Model validation

The experiments conducted in this subsection aim at validating our contributions. We first compare the MuRel approach to the attention-based architecture used in Chapter 4. Then, we run an ablation study on the different components of MuRel. Finally, we evaluate how the number of steps over a MuRel cell impacts the performance of the VQA system.

Comparison to Attention-based model In Table 5.1, we compare MuRel against a strong attentional model, based on the bilinear fusion approach developped in Chapter 4. For a fair comparison, both models are trained on the same *train* set, on top of the same bottom-up features, and have an equivalent amount of

Model	VQA 2.0	VQA CP v2	TDIUC
Attention baseline	63.44	38.04	86.96
MuRel	**65.14**	**39.54**	**88.20**

Table 5.1 – **Comparing MuRel to Attention.** Comparison of the MuRel strategy against a strong Attention-based model on the VQA 2.0 *val*, VQA-CP v2 and TDIUC datasets. We report the OE-Accuracy for VQA 2.0 and VQA-CP v2, and the classical accuracy for TDIUC. Both models have an equivalent number of parameters (∼60 million) and are trained on the same features following the same experimental setup.

learned parameters(∼60 millions including those from the GRU encoder). MuRel reaches a higher accuracy on the three datasets. We report a significant gain of +1.70 on VQA 2.0 and +1.50 on VQA CP v2. Not only these results validate the ability of MuRel to better model interactions between the question and the image, but also to generalize when the distribution of the answers per question-types are completely different between the training and test, as in VQA CP v2. The gain of +1.24 on TDIUC demonstrates the richer modeling capacity of MuRel in a fine-grained context of 12 well delimited question types.

Ablation study In Table 5.2, we evaluate the gains provided by different component of MuRel. In the first line, we show the result of a vanilla MuRel. It performs a single iteration through the MuRel cell ($T = 1$) and no pairwise module (Equation 5.4 becomes $\boldsymbol{x}_i = \boldsymbol{m}_i$). In the second line, the model still performs a single MuRel cell iteration, but we activate the pairwise module. As we can see, this leads to higher accuracy on all the datasets. In fact, between line 1 and 2, we report a gain of +0.44 on VQA 2.0, +0.24 on VQA CP v2 and +0.36 on TDIUC. In the third line, we remove the pairwise module and activate the iterations through the cell. Between line 1 et 3, we report a gain of +0.59 on VQA 2.0, +0.49 on VQA CP v2 and +0.42 on TDIUC. Please note that this modification does not add any parameter to the model. We simply iterate multiple times over a single MuRel cell. Finally, the pairwise module and the iterative process are added to obtain the complete MuRel network. This instance in line 4 reaches the highest accuracy on the three datasets, indicating the relevance of our method.

Number of iterations In Figure 5.9, we analyze the behaviour of the iterative process performed by the MuRel network. We train four different MuRel networks on the VQA 2.0 *train* split, each with a different number of iterations over the MuRel cell. Performance is reported on *val* split. Again, we remind that the four networks have exactly the same amount of parameters. The performance varies only because of the number of times we iterate over a single cell. We report an

Pairwise	Iter.	VQA 2.0	VQA CP v2	TDIUC
✗	✗	64.13	38.88	87.50
✓	✗	64.57	39.12	87.86
✗	✓	64.72	39.37	87.92
✓	✓	**65.14**	**39.54**	**88.20**

Table 5.2 – **Ablation study of MuRel**. Experimental validation of the pairwise module and the iterative processing on the VQA 2.0 *val*, VQA-CP v2 and TDIUC datasets.

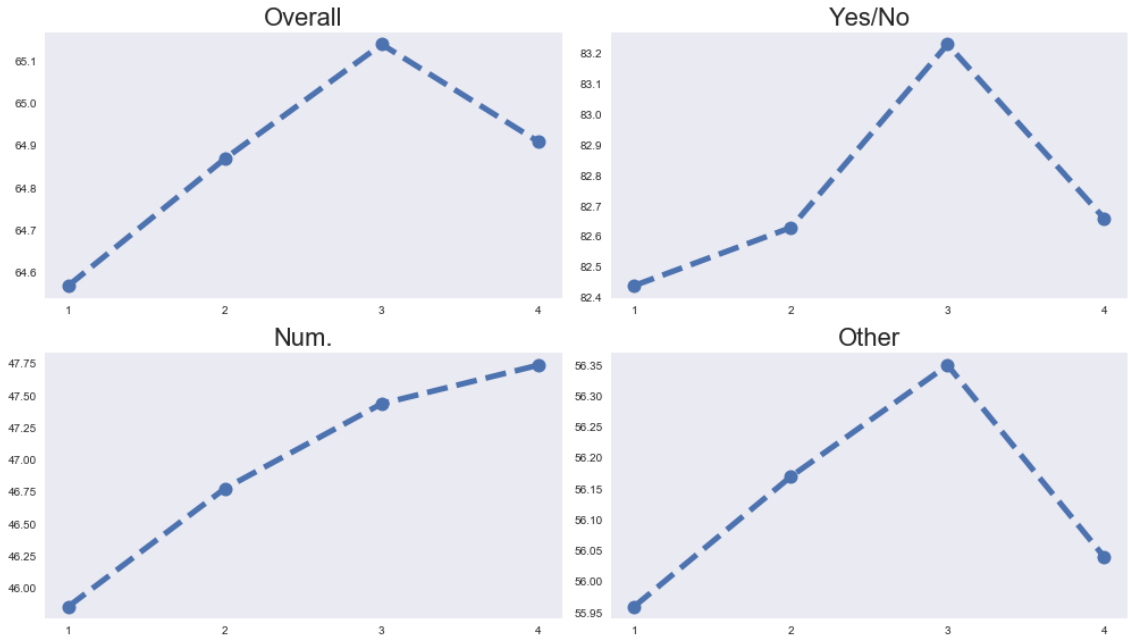


Figure 5.9 – **Number of iterations**. Impact of the number of steps in the iterative process on the different question types of VQA 2.0 *val*.

increase of performance until 3 steps on the global (Overall accuracy), yes/no questions and "other" answers. Interestingly, the performance on questions whose answer is a number keeps increasing. Counting is a challenging task for an end-to-end Deep Learning (DL) system. Not only does it need to detect every occurence of the counted object, it also needs to keep an information about quantity in the final image representation s. The complexity of this question may require deeper relational modeling, and thus benefit from a higher number of iterations over the MuRel cell.

5.3.4 State of the art comparison

We now compare our MuRel network to state-of-the-art approaches on three commonly used datasets for VQA: the VQA 2.0 Dataset, VQA-CP v2 and TDIUC.

VQA 2.0 In Table 5.3, we compare MuRel to the most recent contributions on the VQA 2.0 Dataset. For fairness considerations, all the scores correspond to models trained on the VQA 2.0 *train+val* split, using the bottom-up visual features (Anderson et al. 2018). Interestingly, our model surpasses both MUTAN, presented in Chapter 3 and trained by (Bai et al. 2018), and Multimodal Low-rank Bilinear (MLB) (Kim et al. 2017), which correspond to some of the most powerful approaches involving visual attention and bilinear models. This tends to indicate that VQA models can benefit from retaining local information in multi-modal vectors instead of scalar coefficients. Moreover, our model greatly improves over the recent method proposed in (Norcliffe-Brown et al. 2018). In this work, pairwise attention scores between regions define a structure, over which a set of spatial graph convolutions updates local representations depending on their neighbours. This result shows the strength of our spatial-semantic pairwise modeling between all possible pairs of regions. Even though we did not extensively tune the hyperparameters of our model, our overall score on the *test-dev* split is highly competitive with state-of-the-art methods. In particular, we are comparable to Pythia (Jiang et al. 2018) who won the VQA Challenge 2018. Please note that they improve their overall scores up to 70.01% when they include multiple types of visual features and more training data. Our performance is also on par with (Y. Zhang et al. 2018), whose model is specifically tailored for answering counting questions. We would like to mention that their model is trained with a different version of the bottom-up features where each image is described by a variable number of regions (10 to 100). This setup, as measured (Teney et al. 2018), provides better results than the fixed-size 36 regions we use. Implementing the support for a variable number of regions would possibly improve our results. Also, we did not report in Table 5.3 the score of 69.52% obtained by Bilinear Attention Network (BAN) (Kim et al. 2018) as they train their model on extra data from the Visual Genome dataset (Krishna et al. 2017).

TDIUC One of the core aspect of VQA models is their ability to address various tasks. The TDIUC dataset enables a detailed analysis of the strengths and limitations of a model by measuring its performance on different types of question. We show in Table 5.4 a detailed comparison of recent models to our MuRel. We obtain state-of-the-art results on the Overall Accuracy and the Arithmetic Mean of Per-Type accuracies (A-MPT), and surpass by a significant margin the second best model proposed by (Shi et al. 2018). Interestingly, we improve over this model even though it uses a combination of bottom-up and fixed-grid features, as well

Model	*test-dev* Yes/No	Num.	Other	All	*test-std* All
Bottom-up (Anderson et al. 2018)	81.82	44.21	56.05	65.32	65.67
Graph Att. (Norcliffe-Brown et al. 2018)	-	-	-	-	66.18
MUTAN† (Chapter 3)	82.88	44.54	56.50	66.01	66.38
MLB† (Kim et al. 2017)	83.58	44.92	56.34	66.27	66.62
DA-NTN (Bai et al. 2018)	84.29	47.14	57.92	67.56	67.94
Pythia (Jiang et al. 2018)	-	-	-	68.05	-
Counter (Y. Zhang et al. 2018)	83.14	**51.62**	**58.97**	**68.09**	**68.41**
MuRel	**84.77**	49.84	57.85	68.03	**68.41**

Table 5.3 – **State-of-the-art comparison on the VQA 2.0 dataset.** Results on *test-dev* and *test-std* splits. All these models were trained on the same training set (VQA 2.0 *train+val*), using the Bottom-up features provided by (Anderson et al. 2018). No ensembling methods have been used. † have been trained by (Bai et al. 2018).

as a supervision on the question types (hence its 100% result on the *Absurd* task). MuRel notably surpasses all previous methods on the Positional reasoning (+5.9 over Multimodal Compact Bilinear (MCB)), Counting (+8.53 over QTA) questions. These improvements are likely due to the pairwise structure induced within the MuRel cell, which makes the answer prediction depend on the spatial and semantic relations between regions. The effectiveness of our per-region context modelling is also demonstrated by our the improvement on Scene recognition questions. For these questions, representing the image as a collection of independent objects shows lower performance than grounding each of them in its spatial and semantic context. Interestingly, our results on the Harmonic Mean of Per-Type accuracies (H-MPT) are lower than state-of-the-art. For MuRel, this harmonic metric is significantly harmed by our low score of 21.43% on the *Utility and Affordances* task. These questions require to understand the possible usages and possibilities given by objects depicted in the scene. An example of such questions is *Can you eat the yellow object?*. It is quite unlikely to tackle this task properly only through a Computer Vision (CV) approach, without including some

	RAU* (Noh et al. 2016)	MCB* (Fukui et al. 2016)	QTA (Shi et al. 2018)	MuRel
Bottom-up	✗	✗	✓	✓
Scene Reco.	93.96	93.06	93.80	**96.11**
Sport Reco.	93.47	92.77	95.55	**96.20**
Color Attr.	66.86	68.54	60.16	**74.43**
Other Attr.	56.49	56.72	54.36	**58.19**
Activity Reco.	51.60	52.35	60.10	**63.83**
Pos. Reasoning	35.26	35.40	34.71	**41.19**
Object Reco.	86.11	85.54	86.98	**89.41**
Absurd	96.08	84.82	**100.00**	99.8
Util. and Afford.	31.58	**35.09**	31.48	21.43
Object Presence	94.38	93.64	94.55	**95.75**
Counting	48.43	51.01	53.25	**61.78**
Sentiment	60.09	**66.25**	64.38	60.65
Overall A-MPT	67.81	67.90	69.11	**71.56**
Overall H-MPT	59.00	**60.47**	60.08	59.30
Overall Accuracy	84.26	81.86	85.03	**88.20**

Table 5.4 – **State-of-the-art comparison on the TDIUC dataset.** * trained by (Kafle et al. 2017).

kind of external knowledge about the objects and their utility. For this reason, we advocate that this performance drop on *Utility and Affordances* is acceptable.

VQA-CP v2 This dataset has been proposed to evaluate and reduce the question-oriented bias in VQA models. In particular, the distributions of answers with respect to question types differ from *train* to *test* splits. In Table 5.5, we report the scores of two recent baselines (Agrawal et al. 2018; Malinowski et al. 2018), on which we improve significantly. In particular, we demonstrate an important gain over GVQA (Agrawal et al. 2018), which embeds a multi-hop attention scheme within an architecture that behaves differently whether it sees a Yes/No question or not. However, since both methods do not use the powerful bottom-up features, the fairness of the comparison can be questioned. So we also train an attention model with a BLOCK fusion (Chapter 4) using these bottom-up region representation. We observe that MuRel provides a substantial gain over this strong attention baseline. Given the distribution mismatch between *train* and *test* splits, models that learn linguistic biases instead of reasoning about the image are systematically penalized on their *test* scores. This property of VQA-CP v2 implies that the

Model	Bottom up	Yes/No	Num.	Other	All
HAN (Malinowski et al. 2018)	✗	52.25	**13.79**	20.33	28.65
GVQA (Agrawal et al. 2018)	✗	**57.99**	13.68	22.14	31.30
Attention	✓	41.56	12.19	43.29	38.04
MuRel	✓	42.85	13.17	**45.04**	**39.54**

Table 5.5 – **State-of-the-art comparison on the VQA-CP v2 dataset.** The Attention model was trained by us using the Bottom-up features.

pairwise iterative structure of MuRel is less prone to question-based overfitting than classical attention architectures.

5.4 Conclusion

In this chapter, we focused on the problem of visual reasoning for VQA. Our system (MuRel) is based on rich representations of visual image regions that are progressively merged with the question embedding. We make the representation of each region dependant of its context through non-local pairwise modeling of relations. Moreover, internal activations of the MuRel network provide a basis to define visualization schemes, helpful to interpret the decisions of the model.

We validate our approach on three challenging datasets: VQA 2.0, VQA-CP v2 and TDIUC. We exhibite various ablation studies, clearly demonstrating the gain of our contributions. We show how the vector-based architecture of MuRel competes strongly against to the widely used attention framework. We also demonstrate the relevance of using pairwise relations between regions, and performing multiple iterations over a cell. Our final MuRel network is very competitive and performs favorably against state-of-the-art models, consistently over the three datasets.

CHAPTER 6

CONCLUSION

Contents

6.1 Summary of Contributions

In this thesis, we tackle the recent and challenging problem of Visual Question Answering (VQA). We took an approach based on Deep Learning (DL), and identified several limitations of classical models. Our contribution can be organized in two points detailed below.

Multi-modal fusion. The problem of modeling the interactions between two modalities is fundamental for VQA systems. Even if images and questions are analyzed by cutting-edge encoders, with the latest and most efficient DL techniques, it is still necessary to learn how both modalities should be fused together to answer the question. Following recent literature on the topic, we develop multi-modal fusion strategies that are based on bilinear models.

In Chapter 3, we remark the tensorial structure that these models induce in their parameters and point out the complexity issues raised by these second order strategies. We propose MUTAN, an multi-modal fusion layer where the three-way array of trainable parameters is compressed and simplified using the Tucker decomposition, which structurally restricts the *mode ranks* of the tensor. This leads to an efficient calculation of the fusion between two vectors under a reduced number of learnable weights. To show the effectiveness of MUTAN, we incorporate it into the widely used visual attention architecture for VQA, which learns to smoothly select the image regions that are useful to answer a specific question.

We delve deeper into bilinear fusion models in Chapter 4, where we leverage the general block-term decomposition of tensors to learn even more expressive fusion models. This decomposition imposes a block-superdiagonal structure of the third-order tensor, which breaks down the complexity while improving the model capacities. In particular, it allows to represent multi-modal interactions between

high-dimensional projections of mono-modal vectors. Similarly to Chapter 3, we demonstrate the strength of BLOCK in VQA by integrating it into the attention setup. We also show that bilinear fusion models, and especially BLOCK, can be used for other applications that require learning a fusion module. In particular, we design a very simple yet effective architecture for Visual Relationship Detection (VRD) based on BLOCK, and show how it surpasses previous methods.

Visual reasoning. Both contributions in Chapter 3 and Chapter 4 are focused on designing trainable fusion modules that are able to learn and extract the relevant correlations between two input vectors. Their effectiveness in the context of VQA is highlighted by the visual attention model, which learns the contribution of each region as a fusion between the question embedding and the corresponding local representation. In Chapter 5, we move away from this widely used architecture and question its relevance to attack problems that require strong spatial and semantic reasoning capacities. In MuRel, we model the fusion between each region and the question with a bilinear module that outputs a vector instead of a single scalar. Moreover, these local multi-modal representations are designed to be dependant on context informations that we model by pairwise representations of regions. This process is encapsulated in an iterative process, that progressively updates each local representation and finally provides an answer prediction.

6.2 Perspectives for Future Work

At the end of our work, it seems that multiple directions are promising research paths to improve VQA systems.

Data. Previous to designing and learning models, a training dataset must be constructed. For VQA, the data collection is far from being an easy task. When working on real data, in an open domain for both questions and answers, a manual annotation process is often required. It has multiple drawbacks, which constitute interesting research problems.

- The annotation process can produce undesirable effects on the resulting dataset. A lot of work on datasets for VQA is related to the problem of **textual bias**, which has lead to the creation of VQA 2.0 Dataset and VQA-CP (see Section 2.5). In particular, the recent VQA-CP explicitly sets a difference between train and test answer distributions, conditionned on the question type. This very hard setup may be too extreme, and harm the system learnability. This unsolved problem of detecting, characterizing and overcoming bias in the data is more general than VQA, and attacking this problem would benefit many real-world applications.

- Asking annotators to provide questions about images without restriction is likely to result in a huge variety of examples. Using this raw data, it can be difficult to measure the performance of a model for different sub-tasks. To answer the question "*What color is the second car on the left*", a model should have the ability to detect cars, differentiate left from right, count from the left until the second and extract its color. A VQA system designed to be aware of objects and their contexts is expected to behave well on this type of examples. But how could we expect it to answer questions like "*What is the name of the person at the right of the US president*", which requires face recognition modules and extra knowledge about politics? Unfortunately, many questions in modern datasets for VQA require this type of non-visual awareness, which make Computer Vision (CV)-based models difficult to evaluate.

  Some recent work propose to narrow down the domain of images and/or questions, in order to develop and evaluate specific VQA capacities. For instance, the KVQA dataset (Sanket Shah et al. 2019) contains questions with named entities related to politics (people, locations, occupations, *etc.*), and is associated to a knowledge graph composed of relational triplets (*e.g.* <Elizabeth Warren - dateOfBirth - 22 June 1949>). With this dataset, we can develop models that are specifically tailored to reason about images and world knowledge. Another example of specific datasets is TallyQA (Acharya et al. 2019), focused on the problem of counting objects in an image. A particular interest of this dataset around complex counting questions, such as *"How many people are wearing glasses? "* , which requires to filter the people in the image with respect to a specific condition before counting them.

  In parallel to designing datasets that focus on specific model capacities, other work are related to specific input domains. We can cite for instance the VizWiz dataset (Gurari et al. 2018), designed to help blind and visually impaired people in their daily life, or DVQA (Kafle et al. 2018) which asks questions about charts and data visualization plots.

  I believe that building VQA datasets with a well-defined, delimited and precise area of interest could help build and benchmark systems that could serve in real world applications.

- To evaluate specific abilities of VQA model, one possible solution is to build **synthetic datasets**. The most famous example is certainly the CLEVR dataset (Johnson et al. 2017a), where the questions focus on spatial and relational reasoning capacities. Supposedly, achieving high scores on this dataset requires the capacity to reason about the image, the objects, the spatial layout, store and access an internal memory and perform logical operations. This is why an important part VQA models that attack this dataset are complex architectures designed for these specific purposes. However, as these methods can be outperformed by simpler models (Perez et al. 2018), we are invited

to relativise the difficulty of this CLEVR task. I believe more work is to be done on the hard problem of building synthetic datasets for VQA, and even more to build models that can transfer from synthetic to real visual reasoning application.

Models. Following the contributions of this thesis on multi-modal fusion and visual reasoning, we now present possible future works on models for VQA.

- Under the light of the work presented in this thesis, particularly the contributions in Chapter 3 and Chapter 4, the framework of bilinear functions seems to be relevant to learn a merging module between several modalities. These bilinear models are defined by a third order tensor, where the parameters of the function are stored. To obtain simple tractable functions, we impose a structure on this tensor. Many types of structures exist and have been explored in our work. It is possible to restrict the tensor rank (Kim et al. 2017), the mode-ranks (MUTAN, Chapter 3), or the more elaborate block-term ranks (BLOCK, Chapter 4). For each of these ways of constraining the hypothesis space, the idea is roughly similar: it consists in breaking down the complexity and reducing the number of parameters by replacing the large tensor by a factorization composed of simpler elements. I believe that deeper study is required about learning in this context of factorized parameters. For example, in the case of the Tucker decomposition

$$\mathcal{T} = \mathcal{D} \times_1 \boldsymbol{A} \times_2 \boldsymbol{B} \times_3 \boldsymbol{C}, \tag{6.1}$$

  we can postmultiply $\boldsymbol{A}$ by nonsingular matrix $\boldsymbol{F}$, $\boldsymbol{B}$ by nonsingular matrix $\boldsymbol{G}$, $\boldsymbol{C}$ by nonsingular matrix $\boldsymbol{H}$ and replace $\mathcal{D}$ by $\mathcal{D} \times_1 \boldsymbol{F}^{-1} \times_2 \boldsymbol{G}^{-1} \times_3 \boldsymbol{H}^{-1}$ without changing the resulting tensor. While these two decompositions are considered equivalent from a signal analysis perspective (De Lathauwer 2008), this non-uniqueness property of the decomposition could have some effects on the optimization and learning algorithms, which may worth investigation. Besides, recent work started to propose fusion modules based on third or fourth order functions (Z. Yu et al. 2018). In their general form, these models involve *more than third*-order tensors, with even greater complexity issues than our bilinear models. I think these higher order models could benefit from the framework of tensor decompositions.

- Apart from increasing modeling capacities of multi-modal fusion layers, there is still room for improvement in designing VQA systems that can reason about their inputs. I believe that providing systems with spatial and relational reasoning capacities is crucial for answering complex visual questions. In Chapter 5, we designed MuRel as a step towards better global scene understanding, as it models objects and their pairwise relations with respect to a given question. However, there is still room for improvement in this area. VQA

systems could benefit from smarter structure, where a region is only linked to a certain subset of other regions. Building this graph-structured representation and using it in a VQA context is an exciting problem. Graph-based methods could improve performance on spatial and relational questions, but also increase the system interpretability.

- In some cases, accessing an external knowledge base of facts and links between concepts could help the VQA process. For instance, say we have in image that depicts a table with multiple objects on it. Answering the question *"Which object on the table is used for cutting? "* is hardly answerable without some type of access to the information "knife is used to cut". To the best of our knowledge, this complex problem is currently tackled only by a small number of articles like (Wu et al. 2016; Sanket Shah et al. 2019). I believe more work is to be done on providing VQA system with the ability to query structured knowledge bases, from which it could retrieve facts that could help its answering process.

As these problems are being attacked, real-world VQA systems are becoming more and more relevant and reliable. In my opinion, the next step in the field of human-machine interaction systems is Visual Dialog (Das et al. 2017). More complex than VQA, this setup does not involve a single question about an image but a full dialog between a human user and the machine. Not only does the system need to understand the question with respect to an image, it should also have a notion of history regarding which the system would answer the question. Obtaining powerful dialog systems would open the door to fascinating applications. More generally than VQA, machines that *mesure* their environment and understand its semantics would be able to interact with users under minimal friction, thus providing humans with new interpretation capacities of the world.

BIBLIOGRAPHY

Acar, E. and B. Yener (2009). "Unsupervised Multiway Data Analysis: A Literature Survey". In: *IEEE Transactions on Knowledge and Data Engineering* 21.1, pp. 6–20 (cit. on p. 21).

Acar, Evrim, Seyit A. Çamtepe, Mukkai S. Krishnamoorthy, and Bülent Yener (2005). "Modeling and Multiway Analysis of Chatroom Tensors". In: *Proceedings of the 2005 IEEE International Conference on Intelligence and Security Informatics*. ISI'05. Atlanta, GA: Springer-Verlag, pp. 256–268. URL: http://dx.doi.org/10.1007/11427995_21 (cit. on p. 21).

Acharya, Manoj, Kushal Kafle, and Christopher Kanan (2019). "TallyQA: Answering Complex Counting Questions". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on p. 97).

Agrawal, Aishwarya, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi (2018). "Don't Just Assume; Look and Answer: Overcoming Priors for Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 30, 85, 87, 93, 94).

Andersen, C. M. and R. Bro (2003). "Practical aspects of PARAFAC modeling of fluorescence excitation-emission data". In: *Journal of Chemometrics* 17.4, pp. 200–215. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cem.790. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cem.790 (cit. on p. 20).

Anderson, Peter, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang (2018). "Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 13–15, 85, 91, 92).

Andreas, Jacob, Marcus Rohrbach, Trevor Darrell, and Dan Klein (2016a). "Learning to Compose Neural Networks for Question Answering". In: *NAACL HLT 2016, San Diego California, USA, June 12-17, 2016*, pp. 1545–1554. URL: http://aclweb.org/anthology/N/N16/N16-1181.pdf (cit. on pp. 26, 27, 48).

Andreas, Jacob, Marcus Rohrbach, Trevor Darrell, and Dan Klein (2016b). "Neural module networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 39–48 (cit. on pp. 26, 27, 66).

Antol, Stanislaw, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh (2015). "VQA: Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 4, 16, 28, 45).

Azizpour, Hossein, Ali Sharif Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson (2016). "Factors of Transferability for a Generic ConvNet Rep-

resentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)* 38.9, pp. 1790–1802 (cit. on p. 13).

Bader, B. W., R. A. Harshman, and T. G. Kolda (Oct. 2007). "Temporal Analysis of Semantic Graphs Using ASALSAN". In: *Seventh IEEE International Conference on Data Mining (ICDM 2007)*, pp. 33–42 (cit. on p. 21).

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2015). "Neural Machine Translation by Jointly Learning to Align and Translate". In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://arxiv.org/abs/1409.0473 (cit. on p. 22).

Bai, Yalong, Jianlong Fu, Tiejun Zhao, and Tao Mei (Sept. 2018). "Deep Attention Neural Tensor Network for Visual Question Answering". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 91, 92).

Battaglia et al, Peter W (2018). "Relational inductive biases, deep learning, and graph networks". In: *CoRR* abs/1806.01261 (cit. on p. 81).

Ben-Younes, Hedi, Rémi Cadène, Nicolas Thome, and Matthieu Cord (2019). "BLOCK: Bilinear Superdiagonal Fusion for Visual Question Answering and Visual Relationship Detection". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on pp. 6, 56).

Ben-Younes*, Hedi, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2017). "MUTAN: Multimodal Tucker Fusion for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 6, 33).

Ben-Younes*, Hedi, Rémi Cadène*, Nicolas Thome, and Matthieu Cord (2019). "MUREL: Multimodal Relational Reasoning for Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 6, 75).

Bengio, Yoshua, Patrice Simard, and Paolo Frasconi (1994). "Learning Long-Term Dependencies with Gradient Descent is Difficult". In: *IEEE Transactions on Neural Networks* 5.2, pp. 157–166. URL: http://www.iro.umontreal.ca/~lisa/pointeurs/ieeetrnn94.pdf (cit. on p. 16).

Bodén, Mikael (2001). *A Guide to Recurrent Neural Networks and Backpropagation* (cit. on p. 16).

Bottou, Léon, Frank E. Curtis, and Jorge Nocedal (June 2016). "Optimization Methods for Large-Scale Machine Learning". In: *SIAM Review* 60 (cit. on p. 2).

Bro, Rasmus (Dec. 2006). "Review on Multiway Analysis in Chemistry—2000–2005". In: *Critical Reviews in Analytical Chemistry* 36, pp. 279–293 (cit. on p. 20).

Carroll, J. Douglas and Jih-Jie Chang (Sept. 1970). "Analysis of individual differences in multidimensional scaling via an n-way generalization of "Eckart-Young" decomposition". In: *Psychometrika* (cit. on p. 60).

Charikar, Moses, Kevin Chen, and Martin Farach-Colton (2002). "Finding Frequent Items in Data Streams". In: *International Colloquium on Automata, Languages*

and Programming, pp. 693–703. URL: http://dl.acm.org/citation.cfm?id=646255.684566 (cit. on p. 19).

Chen, Zhu, Zhao Yanpeng, Huang Shuaiyi, Tu Kewei, and Ma Yi (2017). "Structured Attentions for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 13, 25).

Cho, Kyunghyun, Bart Van Merriënboer, Çağlar Gülçehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). "Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Doha, Qatar: Association for Computational Linguistics, pp. 1724–1734. URL: http://www.aclweb.org/anthology/D14-1179 (cit. on p. 17).

Chung, Junyoung, Çağlar Gülçehre, Kyunghyun Cho, and Yoshua Bengio (2014). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. Tech. rep. Arxiv report 1412.3555. Presented at the Deep Learning workshop at NIPS2014. Université de Montréal (cit. on p. 17).

Cichocki, Andrzej, Danilo Mandic, Lieven De Lathauwer, Guoxu Zhou, Qibin Zhao, Cesar Caiafa, and Huy Anh Phan (2015). "Tensor decompositions for signal processing applications: From two-way to multiway component analysis". In: *IEEE Signal Processing Magazine* 32.2, pp. 145–163 (cit. on pp. 20, 21).

Comon, P. (May 2014). "Tensors : A brief introduction". In: *IEEE Signal Processing Magazine* 31.3, pp. 44–53 (cit. on p. 20).

Dai, Bo, Yuqi Zhang, and Dahua Lin (2017). "Detecting Visual Relationships with Deep Relational Networks". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 67, 71, 72).

Das, Abhishek, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José M.F. Moura, Devi Parikh, and Dhruv Batra (2017). "Visual Dialog". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 99).

De Lathauwer, L. (2008). "Decompositions of a Higher-Order Tensor in Block Terms — Part II: Definitions and Uniqueness". In: *SIAM J. Matrix Anal. Appl.* 30.3, pp. 1033–1066 (cit. on pp. 21, 56, 61, 98).

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2018). "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv preprint arXiv:1810.04805* (cit. on p. 18).

Elman, Jeffrey L. (1990). "Finding structure in time". In: *COGNITIVE SCIENCE* 14.2, pp. 179–211 (cit. on p. 16).

Fukui, Akira, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach (2016). "Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding." In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. The Association

for Computational Linguistics (cit. on pp. 13, 19, 24, 30, 43, 45, 46, 48, 64, 66, 93).

Fukushima, Kunihiko (1980). "Neocognitron: a Self Organizing Neural Network Model for a Mechanism of Pattern Recognition Unaffected by Shift in Position". In: *Biological Cybernetics* 36.4, pp. 193–202 (cit. on p. 12).

G. Kolda, Tamara, Brett W. Bader, and Joseph P. Kenny (Jan. 2005). "Higher-Order Web Link Analysis Using Multilinear Algebra." In: pp. 242–249 (cit. on p. 21).

Gao, Haoyuan, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu (2015). "Are You Talking to a Machine? Dataset and Methods for Multilingual Image Question Answering". In: *Advances in Neural Information Processing Systems (NIPS)*. NIPS'15. Montreal, Canada: MIT Press, pp. 2296–2304. URL: http://dl.acm.org/citation.cfm?id=2969442.2969496 (cit. on pp. 11, 13).

Goodfellow, Ian, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio (2014). "Generative adversarial nets". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 2672–2680 (cit. on p. 3).

Goovaerts, G., O. De Wel, B. Vandenberk, R. Willems, and S. Van Huffel (Sept. 2015). "Detection of irregular heartbeats using tensors". In: *2015 Computing in Cardiology Conference (CinC)*, pp. 573–576 (cit. on p. 20).

Goyal, Yash, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh (2017). "Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 29, 63, 66, 84, 87).

Greff, Klaus, Rupesh Kumar Srivastava, Jan Koutnık, Bas R. Steunebrink, and Jürgen Schmidhuber (2015). "LSTM: A Search Space Odyssey". In: *CoRR* abs/1503.04069. URL: http://arxiv.org/abs/1503.04069 (cit. on p. 16).

Gurari, Danna, Qing Li, Abigale J. Stangl, Anhong Guo, Chi Lin, Kristen Grauman, Jiebo Luo, and Jeffrey P. Bigham (2018). "VizWiz Grand Challenge: Answering Visual Questions from Blind People". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 97).

Hardoon, David R., Sandor R. Szedmak, and John R. Shawe-taylor (Dec. 2004). "Canonical Correlation Analysis: An Overview with Application to Learning Methods". In: *Neural Comput.* 16.12, pp. 2639–2664. URL: http://dx.doi.org/10.1162/0899766042321814 (cit. on p. 2).

Harshman, Richard A., Peter Ladefoged, Hans Reichenbach, Robert I. Jennrich, Dale Terbeek, Lee Cooper, Andrew Comrey, P. M. Bentler, Jeanne Yamane, Diane Vaughan, and Bill Jahnke (2001). "Foundations of the Parafac Procedure: Models and Conditions for an "explanatory" Multimodal Factor Analysis". In: (cit. on p. 60).

He, Kaiming, Georgia Gkioxari, Piotr Dollár, and Ross Girshick (2017). "Mask R-CNN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 28).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2015). "Deep Residual Learning for Image Recognition". In: *arXiv preprint arXiv:1512.03385* (cit. on pp. 34, 45).

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun (2016). "Deep Residual Learning for Image Recognition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 13).

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997a). "Long Short-Term Memory". In: *Neural Comput.* 9.8, pp. 1735–1780. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735 (cit. on p. 16).

Hochreiter, Sepp and Jürgen Schmidhuber (Nov. 1997b). "Long Short-Term Memory". In: *Neural Comput.* 9.8, pp. 1735–1780. URL: http://dx.doi.org/10.1162/neco.1997.9.8.1735 (cit. on p. 21).

Hotelling, Harold (1936). "Relations Between Two Sets of Variates". In: *Biometrika* 28.3-4, pp. 321–377. eprint: http://biomet.oxfordjournals.org/content/28/3-4/321.full.pdf+html. URL: http://biomet.oxfordjournals.org/content/28/3-4/321.short (cit. on p. 2).

Hu, Guosheng, Yang Hua, Yang Yuan, Zhihong Zhang, Zheng Lu, Sankha S. Mukherjee, Timothy M. Hospedales, Neil M. Robertson, and Yongxin Yang (Oct. 2017). "Attribute-Enhanced Face Recognition With Neural Tensor Fusion Networks". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 22).

Hu, Ronghang, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko (2017). "Learning to Reason: End-to-End Module Networks for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 27).

Jabri, Allan, Armand Joulin, and Laurens van der Maaten (2016). "Revisiting Visual Question Answering Baselines". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Cham: Springer International Publishing, pp. 727–739 (cit. on p. 19).

Jiang, Yu, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh (2018). *Pythia vo.1: The Winning Entry to the VQA Challenge 2018*. https://github.com/facebookresearch/pythia (cit. on pp. 14, 85, 91, 92).

Johnson, Justin, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross B. Girshick (2017a). "CLEVR: A Diagnostic Dataset for Compositional Language and Elementary Visual Reasoning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1988–1997 (cit. on pp. 4, 26, 84, 97).

Johnson, Justin, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick (2017b). "Inferring and Executing Programs for Visual Reasoning". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 27).

Kafle, Kushal and Christopher Kanan (Oct. 2017). "An Analysis of Visual Question Answering Algorithms". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 29, 66, 85, 93).

Kafle, Kushal, Brian Price, Scott Cohen, and Christopher Kanan (2018). "DVQA: Understanding Data Visualizations via Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 97).

Karpathy, Andrej (2015). *The Unreasonable Effectiveness of Recurrent Neural Networks*. URL: http://karpathy.github.io/2015/05/21/rnn-effectiveness/ (cit. on pp. 16, 17).

Kim, Jin-Hwa, Jaehyun Jun, and Byoung-Tak Zhang (2018). "Bilinear Attention Networks". In: *Advances in Neural Information Processing Systems (NIPS)*. Ed. by S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett. Curran Associates, Inc., pp. 1571–1581. URL: http://papers.nips.cc/paper/7429-bilinear-attention-networks.pdf (cit. on pp. 14, 25, 91).

Kim, Jin-Hwa, Sang-Woo Lee, Donghyun Kwak, Min-Oh Heo, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang (2016). "Multimodal Residual Learning for Visual QA". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 361–369 (cit. on pp. 13, 48).

Kim, Jin-Hwa, Kyoung Woon On, Woosang Lim, Jeonghee Kim, Jung-Woo Ha, and Byoung-Tak Zhang (2017). "Hadamard Product for Low-rank Bilinear Pooling". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on pp. 13, 18, 19, 24, 30, 38, 43, 45, 46, 48, 64, 91, 92, 98).

Kingma, Diederik P. and Jimmy Ba (2014). "Adam: A Method for Stochastic Optimization". In: *CoRR* abs/1412.6980. URL: http://arxiv.org/abs/1412.6980 (cit. on pp. 46, 63, 85).

Kiros, Ryan, Ruslan Salakhutdinov, and Rich Zemel (22–24 Jun 2014a). "Multimodal Neural Language Models". In: ed. by Eric P. Xing and Tony Jebara. Proceedings of Machine Learning Research. Bejing, China: PMLR, pp. 595–603. URL: http://proceedings.mlr.press/v32/kiros14.html (cit. on pp. 3, 21).

Kiros, Ryan, Ruslan Salakhutdinov, and Richard S Zemel (2014b). "Unifying visual-semantic embeddings with multimodal neural language models". In: *arXiv preprint arXiv:1411.2539* (cit. on pp. 3, 4).

Kiros, Ryan, Yukun Zhu, Ruslan Salakhutdinov, Richard S. Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler (2015). "Skip-thought Vectors". In: *Advances in Neural Information Processing Systems (NIPS)*. Montreal, Canada, pp. 3294–3302. URL: http://dl.acm.org/citation.cfm?id=2969442.2969607 (cit. on pp. 3, 18).

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1097–1105 (cit. on pp. 12, 45).

Krishna, Ranjay, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. (2017). "Visual genome: Connecting language and vision using crowdsourced dense image annotations". In: *International Journal of Computer Vision (IJCV)* 123.1, pp. 32–73 (cit. on pp. 46, 48, 66, 91).

Lebedev, Vadim, Yaroslav Ganin, Maxim Rakhuba, Ivan Oseledets, and Victor Lempitsky (2014). *Speeding up convolutional neural networks using fine-tuned CP-decomposition*. arXiv preprint 1412.6553. URL: http://arxiv.org/abs/1412.6553 (cit. on p. 21).

LeCun, Yann, Bernhard Boser, John Denker, Donnie Henderson, Richard Howard, Wayne Hubbard, and Lawrence Jackel (1989). "Backpropagation Applied to Handwritten Zip Code Recognition". In: *Neural computation* 1.4, pp. 541–551 (cit. on p. 12).

Li, Yikang, Wanli Ouyang, Xiaogang Wang, and Xiao'ou Tang (July 2017). "ViP-CNN: Visual Phrase Guided Convolutional Neural Network". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 67, 72).

Liang, Xiaodan, Lisa Lee, and Eric P. Xing (July 2017). "Deep Variation-Structured Reinforcement Learning for Visual Relationship and Attribute Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 72).

Lin, Tsung-Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick (Jan. 1, 2014). "Microsoft COCO: Common Objects in Context". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*. URL: /se3/wp-content/uploads/2014/09/coco_eccv.pdf,%20http://mscoco.org (cit. on p. 28).

Lin, Tsung-Yu, Aruni RoyChowdhury, and Subhransu Maji (2015). "Bilinear CNN Models for Fine-grained Visual Recognition". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 59).

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440 (cit. on p. 13).

Lu, Cewu, Ranjay Krishna, Michael Bernstein, and Li Fei-Fei (2016). "Visual Relationship Detection with Language Priors". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 67, 68, 72).

Lu, Jiasen, Jianwei Yang, Dhruv Batra, and Devi Parikh (2016). "Hierarchical Question-Image Co-Attention for Visual Question Answering". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 289–297. URL: http://papers.nips.cc/paper/6202-hierarchical-question-image-co-attention-for-visual-question-answering (cit. on pp. 19, 24, 48).

Ma, Lin, Zhengdong Lu, and Hang Li (2016). "Learning to Answer Questions from Image Using Convolutional Neural Network". In: *Proceedings of the AAAI*

Conference on Artificial Intelligence (AAAI). AAAI'16. Phoenix, Arizona: AAAI Press, pp. 3567–3573. URL: http://dl.acm.org/citation.cfm?id=3016387.3016405 (cit. on pp. 11, 13).

Malinowski, Mateusz, Carl Doersch, Adam Santoro, and Peter Battaglia (Sept. 2018). "Learning Visual Question Answering by Bootstrapping Hard Attention". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 24, 93, 94).

Malinowski, Mateusz and Mario Fritz (2014a). "A Multi-World Approach to Question Answering about Real-World Scenes based on Uncertain Input". In: *Advances in Neural Information Processing Systems (NIPS)*. Ed. by Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger. Curran Associates, Inc., pp. 1682–1690. URL: http://papers.nips.cc/paper/5411-a-multi-world-approach-to-question-answering-about-real-world-scenes-based-on-uncertain-input.pdf (cit. on p. 4).

Malinowski, Mateusz and Mario Fritz (2014b). "Towards a Visual Turing Challenge". In: *Learning Semantics* (cit. on p. 4).

Malinowski, Mateusz, Marcus Rohrbach, and Mario Fritz (Jan. 1, 2016). "Ask Your Neurons: A Deep Learning Approach to Visual Question Answering". In: *arXiv:1605.02697*. published (cit. on pp. 11, 13, 48).

Mascharka, David, Philip Tran, Ryan Soklaski, and Arjun Majumdar (June 2018). "Transparency by Design: Closing the Gap Between Performance and Interpretability in Visual Reasoning". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 27).

Mikolov, Tomas, Martin Karafiát, Lukás Burget, Jan Cernocký, and Sanjeev Khudanpur (2010). "Recurrent neural network based language model." In: *INTERSPEECH*. Ed. by Takao Kobayashi, Keikichi Hirose, and Satoshi Nakamura. ISCA, pp. 1045–1048. URL: http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10 (cit. on p. 16).

Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean (2013). "Distributed representations of words and phrases and their compositionality". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 3111–3119 (cit. on pp. 15, 18).

Miwakeichi, Fumikazu, Eduardo Martınez-Montes, Pedro A. Valdés-Sosa, Nobuaki Nishiyama, Hiroaki Mizuhara, and Yoko Yamaguchi (2004). "Decomposing EEG data into space–time–frequency components using Parallel Factor Analysis". In: *NeuroImage* 22.3, pp. 1035–1045. URL: http://www.sciencedirect.com/science/article/pii/S1053811904001958 (cit. on p. 20).

Mørup, Morten (2011). "Applications of tensor (multiway array) factorizations and decompositions in data mining". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 1.1, pp. 24–40. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.1. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/widm.1 (cit. on p. 21).

Nam, Hyeonseob, Jung-Woo Ha, and Jeonghee Kim (2017). "Dual Attention Networks for Multimodal Reasoning and Matching". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE Computer Society, pp. 2156–2164 (cit. on p. 25).

Noh, Hyeonwoo and Bohyung Han (2016). "Training recurrent answering units with joint loss minimization for vqa". In: *arXiv preprint arXiv:1606.03647* (cit. on pp. 66, 93).

Norcliffe-Brown, Will, Efstathios Vafeias, and Sarah Parisot (2018). "Learning Conditioned Graph Structures for Interpretable Visual Question Answering". In: (cit. on pp. 25, 80, 91, 92).

Olah, Chris, Arvind Satyanarayan, Ian Johnson, Shan Carter, Ludwig Schubert, Katherine Ye, and Alexander Mordvintsev (2018). "The Building Blocks of Interpretability". In: *Distill*. https://distill.pub/2018/building-blocks (cit. on p. 24).

Olah, Christopher (2015). *Understanding LSTM Networks*. URL: http://colah.github.io/posts/2015-08-Understanding-LSTMs/ (cit. on p. 16).

Pascanu, Razvan, Tomas Mikolov, and Yoshua Bengio (2013). "On the difficulty of training recurrent neural networks." In: vol. 28. JMLR Proceedings. JMLR.org, pp. 1310–1318. URL: http://dblp.uni-trier.de/db/conf/icml/icml2013.html#PascanuMB13 (cit. on p. 16).

Pennington, Jeffrey, Richard Socher, and Christopher Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1532–1543 (cit. on p. 15).

Perez, Ethan, Florian Strub, Harm de Vries, Vincent Dumoulin, and Aaron C. Courville (2018). "FiLM: Visual Reasoning with a General Conditioning Layer". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on pp. 28, 84, 97).

Perronnin, Florent, Jorge Sanchez, and Thomas Mensink (2010). "Improving the Fisher Kernel for Large-Scale Image Classification". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*.

Peters, Matthew E., Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer (2018). "Deep contextualized word representations". In: (cit. on p. 18).

Peyre, Julia, Ivan Laptev, Cordelia Schmid, and Josef Sivic (2017). "Weakly-supervised learning of visual relations". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 72).

Radford, Alec, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever (2019). "Language Models are Unsupervised Multitask Learners". In: (cit. on p. 18).

Razavian, Ali Sharif, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson (2014). "CNN Features off-the-shelf: an Astounding Baseline for Recognition".

In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshop* (cit. on p. 13).

Reed, Scott, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee (2016). "Generative Adversarial Text to Image Synthesis". In: ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, pp. 1060–1069 (cit. on p. 3).

Ren, Mengye, Ryan Kiros, and Richard S. Zemel (2015). "Exploring Models and Data for Image Question Answering". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 2953–2961. URL: http://papers.nips.cc/paper/5640-exploring-models-and-data-for-image-question-answering (cit. on pp. 11, 13, 19).

Ren, Shaoqing, Kaiming He, Ross Girshick, and Jian Sun (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 91–99 (cit. on p. 14).

Russakovsky, Olga, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander Berg, and Li Fei-Fei (2015). "ImageNet Large Scale Visual Recognition Challenge". In: *International Journal of Computer Vision (IJCV)* 115.3, pp. 211–252 (cit. on pp. 12, 13).

Sanket Shah Anand Mishra, Naganand Yadati and Partha Pratim Talukdar (2019). "KVQA: Knowledge-Aware Visual Question Answering". In: *Proceedings of the AAAI Conference on Artificial Intelligence (AAAI)* (cit. on pp. 97, 99).

Santoro, Adam, David Raposo, David G Barrett, Mateusz Malinowski, Razvan Pascanu, Peter Battaglia, and Timothy Lillicrap (2017). "A simple neural network module for relational reasoning". In: *Advances in Neural Information Processing Systems (NIPS)*. Ed. by I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett. Curran Associates, Inc., pp. 4967–4976. URL: http://papers.nips.cc/paper/7082-a-simple-neural-network-module-for-relational-reasoning.pdf (cit. on p. 26).

Shi, Yang, Tommaso Furlanello, Sheng Zha, and Animashree Anandkumar (Sept. 2018). "Question Type Guided Attention in Visual Question Answering". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)* (cit. on pp. 91, 93).

Shih, Kevin J., Saurabh Singh, and Derek Hoiem (2016). "Where To Look: Focus Regions for Visual Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 16, 19).

Socher, Richard, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng (2014). "Grounded Compositional Semantics for Finding and Describing Images with Sentences". In: *Transactions of the Association for Com-*

putational Linguistics (TACL) 2, pp. 207–218. URL: https://tacl2013.cs.columbia.edu/ojs/index.php/tacl/article/view/325 (cit. on p. 3).

Sun, Jian-Tao, Hua-Jun Zeng, Huan Liu, Yuchang Lu, and Zheng Chen (2005). "CubeSVD: A Novel Approach to Personalized Web Search". In: *Proceedings of the 14th International Conference on World Wide Web*. WWW '05. Chiba, Japan: ACM, pp. 382–390. URL: http://doi.acm.org/10.1145/1060745.1060803 (cit. on p. 21).

Teney, Damien, Peter Anderson, Xiaodong He, and Anton van den Hengel (June 2018). "Tips and Tricks for Visual Question Answering: Learnings From the 2017 Challenge". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 62, 66, 91).

Tucker, Ledyard R. (1966). "Some mathematical notes on three-mode factor analysis". In: *Psychometrika* 31.3, pp. 279–311. URL: http://dx.doi.org/10.1007/BF02289464 (cit. on pp. 30, 36).

Vasilescu, M. Alex O. and Demetri Terzopoulos (2002). "Multilinear Analysis of Image Ensembles: TensorFaces". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*. Ed. by Anders Heyden, Gunnar Sparr, Mads Nielsen, and Peter Johansen. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 447–460 (cit. on p. 20).

Wang, Hongcheng and Narendra Ahuja (Mar. 2008). "A Tensor Approximation Approach to Dimensionality Reduction". In: *International Journal of Computer Vision (IJCV)* 76.3, pp. 217–229. URL: https://doi.org/10.1007/s11263-007-0053-0 (cit. on p. 20).

Wu, Q., P. Wang, C. Shen, A. Dick, and A. van den Hengel (2016). "Ask me anything: free-form visual question answering based on knowledge from external sources". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 48, 99).

Xiong, Caiming, Stephen Merity, and Richard Socher (2016). "Dynamic Memory Networks for Visual and Textual Question Answering". In: ICML'16. New York, NY, USA: JMLR.org, pp. 2397–2406. URL: http://dl.acm.org/citation.cfm?id=3045390.3045643 (cit. on p. 13).

Xu, Huijuan and Kate Saenko (2016). "Ask, Attend and Answer: Exploring Question-Guided Spatial Attention for Visual Question Answering". In: *Proceedings of the IEEE European Conference on Computer Vision (ECCV)*, pp. 451–466. URL: http://dx.doi.org/10.1007/978-3-319-46478-7_28 (cit. on pp. 16, 24, 48).

Xu, Kelvin, Jimmy Lei Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard S. Zemel, and Yoshua Bengio (2015). "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". In: ICML'15. Lille, France: JMLR.org, pp. 2048–2057. URL: http://dl.acm.org/citation.cfm?id=3045118.3045336 (cit. on p. 22).

Yan, Fei and Krystian Mikolajczyk (June 2015). "Deep Correlation for Matching Images and Text". In: (cit. on p. 2).

Yan, S., D. Xu, Q. Yang, L. Zhang, X. Tang, and H. Zhang (Jan. 2007). "Multilinear Discriminant Analysis for Face Recognition". In: *IEEE Transactions on Image Processing* 16.1, pp. 212–220 (cit. on p. 20).

Yang, Yongxin and Timothy M. Hospedales (2017a). "Deep Multi-task Representation Learning: A Tensor Factorisation Approach". In: *Proceedings of the International Conference on Learning Representations (ICLR)* (cit. on p. 21).

Yang, Yongxin and Timothy M. Hospedales (2017b). "Unifying Multi-domain Multitask Learning: Tensor and Neural Network Perspectives". In: *Domain Adaptation in Computer Vision Applications*. Cham: Springer International Publishing, pp. 291–309 (cit. on p. 22).

Yang, Zichao, Xiaodong He, Jianfeng Gao, Li Deng, and Alexander J. Smola (2016). "Stacked Attention Networks for Image Question Answering". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 21–29. URL: http://dx.doi.org/10.1109/CVPR.2016.10 (cit. on pp. 13, 24, 48).

Ye, Jinmian, Linnan Wang, Guangxi Li, Di Chen, Shandian Zhe, Xinqi Chu, and Zenglin Xu (June 2018). "Learning Compact Recurrent Neural Networks With Block-Term Tensor Decomposition". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on p. 21).

Yi, Kexin, Jiajun Wu, Chuang Gan, Antonio Torralba, Pushmeet Kohli, and Josh Tenenbaum (2018). "Neural-Symbolic VQA: Disentangling Reasoning from Vision and Language Understanding". In: *Advances in Neural Information Processing Systems (NIPS)*, pp. 1039–1050. URL: http://papers.nips.cc/paper/7381-neural-symbolic-vqa-disentangling-reasoning-from-vision-and-language-understanding (cit. on p. 28).

Yu, Rose, Stephan Zheng, Anima Anandkumar, and Yisong Yue (2017). "Long-term forecasting using tensor-train RNNs". In: *arXiv preprint arXiv:1711.00073* (cit. on p. 22).

Yu, Ruichi, Ang Li, Vlad I. Morariu, and Larry S. Davis (Oct. 2017). "Visual Relationship Detection With Internal and External Linguistic Knowledge Distillation". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 64, 71, 72).

Yu, Zhou, Jun Yu, Jianping Fan, and Dacheng Tao (2017). "Multi-modal Factorized Bilinear Pooling with Co-Attention Learning for Visual Question Answering". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on pp. 13, 18, 19, 24).

Yu, Zhou, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao (2018). "Beyond Bilinear: Generalized Multi-modal Factorized High-order Pooling for Visual Question Answering". In: *IEEE Transactions on Neural Networks and Learning Systems* (cit. on pp. 18, 24, 63, 64, 66, 98).

Zhang, Hanwang, Zawlin Kyaw, Shih-Fu Chang, and Tat-Seng Chua (2017a). "Visual Translation Embedding Network for Visual Relation Detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (cit. on pp. 67, 72).

Zhang, Hanwang, Zawlin Kyaw, Jinyang Yu, and Shih-Fu Chang (2017b). "PPR-FCN: Weakly Supervised Visual Relation Detection via Parallel Pairwise R-FCN". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)* (cit. on p. 3).

Zhang, Yan, Jonathon Hare, and Adam Prügel-Bennett (2018). "Learning to Count Objects in Natural Images for Visual Question Answering". In: *Proceedings of the International Conference on Learning Representations (ICLR)*. URL: https://openreview.net/forum?id=B12Js_yRb (cit. on pp. 14, 66, 85, 91, 92).

Zhou, Bolei, Yuandong Tian, Sainbayar Sukhbaatar, Arthur Szlam, and Rob Fergus (2015). "Simple baseline for visual question answering". In: *arXiv preprint arXiv:1512.02167* (cit. on p. 16).