Konputazio Ingeniaritza eta
Sistema Adimentsuak Unibertsitate Masterra

Máster Universitario en Ingeniería Computacional
y Sistemas Inteligentes

Konputazio Zientziak eta Adimen Artifiziala Saila

Departamento de Ciencias de la Computación e Inteligencia Artificial

Master's Thesis

Answering questions about images that require outside knowledge

Imanol Miranda Martija

Advisors:

Gorka Azkune Galparsoro

HiTZ Basque Center for Language Technologies

Ixa NLP Group

Aitor Soroa Echave

HiTZ Basque Center for Language Technologies

Ixa NLP Group

Universidad del País Vasco Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Konputazio Ingeniaritza eta Sistema Adimentsuak Unibertsitate Masterra

Máster Universitario en Ingeniería Computacional y Sistemas Inteligentes

eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Master's Thesis

Master's Degree in Computational Engineering and Intelligent Systems

Answering questions about images that require outside knowledge

Imanol Miranda Martija

Advisors
Gorka Azkune Galparsoro
Aitor Soroa Echave

July 2023

Acknowledgments

Eskerrak eman nahi dizkiet nire gurasoei, beraien lan eta izerdiarekin bidea erein dutelako eta orain ni naizelako fruituak jasotzen ari dena.

Baita proiektu hau gauzatzen lagundu didaten Aitor, Gorka eta Anderri ere. Proiektu honetan lan egiteko aukera eman didatelako eta lehenengo momentutik edozein laguntza behar izan dudanean hor egon direlako. Bereziki Gorkari eskertuz txostena borobiltzeko orduan eskainitako laguntza.

Abstract

This document presents the research carried out by the student Imanol Miranda during his master's thesis.

The emergence of Transformer architectures, pretrained models and multimodal data problems have generated new challenges to solve. One of the most popular in recent years is the visual-linguistic task Visual Question Answering (VQA). Several variants of this task have emerged, one of them being the Outside Knowledge Visual Question Answering (OK-VQA) task, on which our research will focus. This task adds the complexity that the answer to the question does not appear explicitly in the image, and an external source of knowledge is needed to answer the question. Once the different proposals have been analyzed, the Caption Based Model (CBM) that will serve as the basis for the development is presented.

After the problem has been introduced, the proposals are presented, divided into two groups. On the one hand, a multilabel leveraging technique that can be used in multilabel tasks that have optimal and suboptimal solutions, improving model learning. This technique introduces the concept of balance between exploration and exploitation by means of a frequency distribution based on the proportion in which the solutions appear in the ground truth.

On the other hand, different image verbalization approaches are analyzed and compared. First, using an object detector, the objects and attributes that appear in an image are obtained. Thus, in addition to providing the CBM model with the image caption (where general image information is represented), we also provide object and attribute information (representing image details). In this way, the balance between general and detailed information is improved. Secondly, due to memory limitations, several reranking systems based on Sentence Similarity and Object bounding box area are presented. These systems seek to improve the quality of the information we pass to the model with respect to the question.

After several experiments, we conclude that the new multilabel leverage technique improves model learning by maintaining the number of optimal solutions and increasing the number of suboptimal solutions generated. Also, providing more information to the model improves the results, both by adding attributes to the objects, and by increasing the number of objects. The reranking system based on Object bounding box area gets the best results, reinforcing the idea that the questions focus on objects clearly represented in the image.

Keywords: Transformers, multimodal, OK-VQA, CBM, multilabel leverage technique, object detection, reranking system.

Laburpena

Dokumentu honetan Imanol Miranda ikasleak master amaierako lanean egindako ikerketa aurkezten da.

Transformer arkitekturak, aurrez entrenatutako ereduak eta datu multimodalak dituzten arazoen sorrerak erronka berriak sortu ditu konpontzeko. Azken urteotako ezagunenetako bat Visual Question Answering (VQA) ataza da. Bertan, irudi bat eta galdera bat emanda, erantzun egokia bilatu behar da. Ataza honen hainbat aldaera sortu dira, horietako bat Outside Knowledge Visual Question Answering (OK-VQA) izanik, non gure ikerketa oinarrituko den. Ataza honek konplexutasuna areagotzen du galderaren erantzuna irudian esplizituki ez baita agertzen, eta galderari erantzun ahal izateko kanpoko ezagutza iturri bat behar baita. Ataza ebazteko proposamen ezberdinak aztertu ondoren, gure garapenaren oinarri izango den Caption Based Model-a (CBM) aurkezten da.

Behin arazoa azalduta, bi taldetan banatuta aurkezten dira lan honetan egindako ekarpenak. Alde batetik, etiketa anitzeko aprobetxamendu teknika, soluzio optimoak eta azpi-optimoak dituzten etiketa anitzeko atazetan erabil daitekeena, ereduen ikaskuntza hobetuz. Teknika honek esplorazioaren eta esplotazioaren arteko orekaren kontzeptua erabiltzen du maiztasun-banaketa baten bidez, soluzioak agertzen diren proportzioan oinarrituta.

Bestalde, irudiak berbalizatzeko planteamendu desberdinak aztertu eta alderatzen dira. Lehenik eta behin, objektu detektore bat erabiliz, irudi batean agertzen diren objektuak eta atributuak lortzen dira. Horrela, CBM ereduari irudiaren goiburukoa emateaz gain (non irudiaren informazio orokorra errepresentatzen den), objektu eta atributuen informazioa ere ematen diogu (irudiaren xehetasunak errepresentatuz). Modu horretara informazio orokorraren eta zehatzaren arteko oreka hobetzen da. Bigarrenik, memoria murriztapenak direla eta, esaldiaren antzekotasunean eta objektuen kutxa mugatzaileen azaleran oinarritutako hainbat sailkapen-sistema aurkezten dira. Sistema hauek ereduari pasatzen diogun informazioaren kalitatea hobetzea dute helburu.

Hainbat esperimenturen ondoren, etiketa anitzeko aprobetxamendu teknika berriak ereduaren ikaskuntza hobetzen duela ondorioztatzen dugu, soluzio optimoen kopurua mantenduz eta sortutako soluzio azpi-optimoen kopurua handituz. Era berean, ereduari informazio gehiago emateak emaitzak hobetzen ditu, bai objektuei atributuak gehituz, baita objektu kopurua handituz ere. Objektuen kutxa mugatzaileen azaleran oinarritutako sailkapen-sistemak emaitza onenak lortzen ditu, galderak irudian argi irudikatutako objektuetan oinarritzen direlako ideia indartuz.

Hitz gakoak: Transformer, multimodala, OK-VQA, CBM, etiketa anitzeko aprobetxamendu teknika, objektu detektagailua, sailkapen-sistema.

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

The globalization of the Internet, the evolution of deep learning architectures and pretrained models have created new challenges for Artificial Intelligence. One of them is Multimodal Learning, a vibrant multidisciplinary field of growing importance and extraordinary potential [1]. These are problems with different types of input [1, 2, 3] such as text and images. This is where tasks like image captioning, text2image and Visual Question Answering (VQA) [4] are born.

We will focus on the VQA task, as it is the root of our research task. VQA task uses as input an image and a question in natural language to obtain an answer. Multiple variants arise from this task, such as Outside Knowledge Visual Question Answering (OK-VQA) [5], with the extra challenge of needing an external source of information to answer the question.

The OK-VQA task provides a dataset made up of questions and images to use as a benchmark. The main approaches are based on multimodal transformers [6, 7]. There are two main categories, single-stream and dual-stream [8]. Single-streams use object detectors to obtain the image features and then concatenate the visual and textual representations and provide them to the language transformer to generate a prediction, such as VisualBERT and OSCAR. On the other hand, in the dual-stream, two blocks of independent transformers are used, one for the visual part and the other for the linguistic part, such as ViLBERT and LXMERT. For both transformers to communicate, cross-attention is used.

The research on the Caption-Based Model (CBM) [9], observes that a text-only model, using the image caption and the knowledge gained by the language model during pretraining, equals the state-of-the-art. In our research, the model proposed in the CBM will be used as a basis. The contributions of this work can be divided into two parts: (i) a technique to leverage multilabel annotations, and (ii) a comparison of different image verbalization approaches. As the first contribution, a multilabel leverage technique is proposed. This technique seeks to improve the learning of models in multilabel problems where optimal and suboptimal solutions are available. To do this, a frequency distribution is used to introduce the concept of balance between exploration and exploitation when choosing an answer during training.

As the second contribution, two proposals related to image verbalization are analyzed. The first proposal continues with the philosophy proposed in CBM, seeking to complete the information provided to the model by adding an object detection system. This object detection system will return to the model the objects and attributes present in the image. The idea of adding an object detection system comes from two concepts: image captioning systems generate a general description of the image, without too much detail, and object detection instead, provides details of the image but without taking into account the relationship between them. The combination of both provides a better balance between general and detailed information. In the second proposal, since we had limited memory during the experiments that prevented us from passing all the information to the model, a reranking system is proposed. Two methods are presented, one based on sentence similarity between the question and image object-attributes, using FastText [10, 11] embeddings and cosine similarity [12, 13], and the second selecting the objects based on the bounding box area of the object from largest to smallest. The second proposal is based on the hypothesis that questions will refer to bigger objects that are clearly represented in the image.

With the experiments carried out, several conclusions have been reached. The first is that the proposed multilabel leverage technique is a simple method that improves the results of all the base models we have implemented. This improvement comes from better performance in answer generation, maintaining the number of optimal answers generated but increasing the suboptimal ones. The second conclusion shows that completing the information of the image caption through objects and attributes improves the base model too. This demonstrates the importance of a balance between general and detailed information. Finally, through the reranking system it is observed that the more information is better, both adding attributes to the objects, and when increasing the number of selected objects (k). Also, in this specific case, the reranking based on the area of the bounding box is the criterion that works best.

CHAPTER 2

Background

In this chapter, we will present the background needed to understand this work. To do so, we will start from the most general concepts, introducing Multimodal Learning and Typical multimodal tasks. In the latter, we will expose several examples, deepening with special emphasis on the Visual Question Answering task, introducing different variants of this task as is the case of OK-VQA. Finally, we will analyze the multimodal systems used for these tasks.

2.1 Multimodal Learning

Human beings are surrounded by inputs in different modalities, which are linked to our senses. Such as sight when observing the environment, smells, tastes, sounds we hear or touch with a surface. It is important to understand that these modalities are also linked to the way we communicate and understand our environment, for example natural language, signs or human contact [1].

Traditionally, Artificial Intelligence (AI) projects have been based on a single type of input, such as image classification or named entity recognition. For AI to progress in understanding the world around us, like humans, it must be able to interpret and reason about multimodal data. Increasingly there are problems where more than one type is needed [1, 2, 3], such as video (audio and images) [3, 14] (Figure 2.1), text and images, etc. The idea behind multimodal learning is to build models and techniques that can process and relate information from those different modalities.

Although multimodal learning has been studied over the past few decades, advances in Deep Learning (DL) in recent years have enabled breakthroughs in Multimodal Learning (MML). In particular, Transformer [7] architectures achieve very high performance in MML, which creates new opportunities and challenges, as well as large scale pretrained models [15]. In addition, the globalization of the Internet has generated new needs and challenges with multimodal data, making it a vibrant multidisciplinary field of growing importance and extraordinary potential [1]. This has generated the need to develop multimodal datasets such as Conceptual Captions [16] and VQA [4] with images and text, TIMIT [17] with audio and text, NYU Depth Dataset V2 [18] with images and depth, and so on. This last

dataset shows a new multimodality, different from the previously proposed idea of human senses. In this case, we have an image modality and a depth modality, but the idea is the same, through multimodality to better understand the environment.

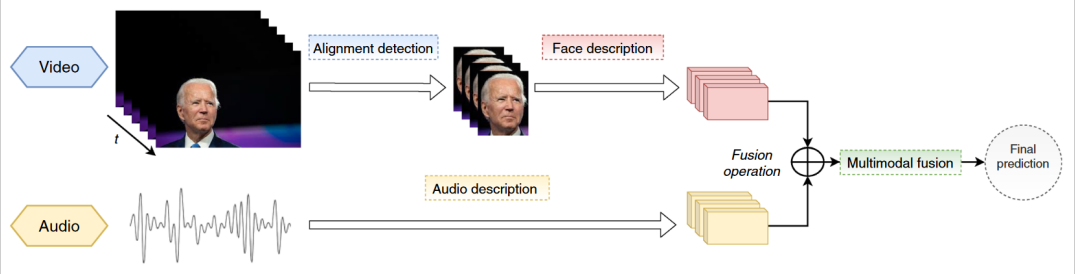

Figure 2.1: Example of a multimodal model with video and audio. It first obtains the representation of each input modality and then fuses both representations to obtain a final prediction [3].

Once we have introduced the concept of MML, we will focus on typical multimodal tasks.

2.2 Typical multimodal tasks

Multimodal tasks are those that use information from multiple modalities, as discussed above. Although there are numerous multimodal tasks, in this section we will focus on three related to the project developed: Image captioning, Text2Image and Visual Question Answering (VQA).

2.2.1 Image Captioning

Image captioning consists of generating a descriptive textual caption that accurately represents the visual content of an image [19, 20]. This task requires understanding both visual and textual information, since the system has to accurately describe the objects, actions, and relationships represented in the image using natural language.

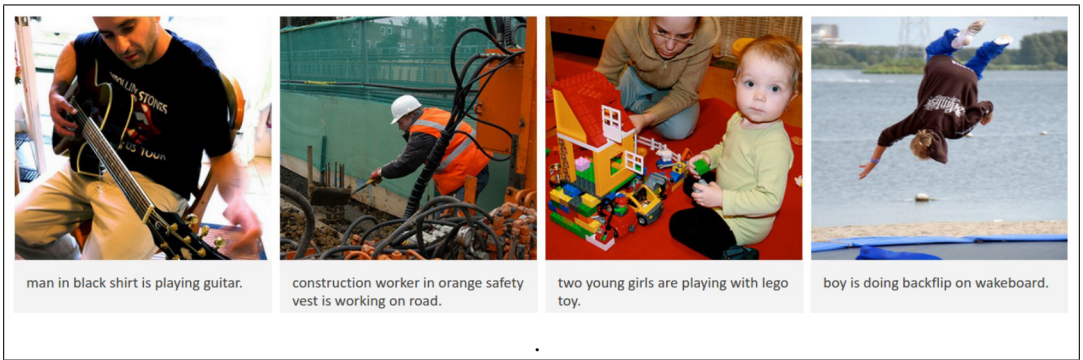

Figure 2.2: Example of images and corresponding caption generated by a multimodal RNN [19].

Typically, these models generate a general description of the image without going into details (Figure 2.2).

2.2.2 Text2Image

Text2Image, also known as image synthesis, consists of generating images based on textual descriptions [21, 22, 23]. That is, something similar to the opposite of image captioning. Like the previous task, it requires a deep understanding of semantic and visual aspects to create meaningful and coherent visual representations.

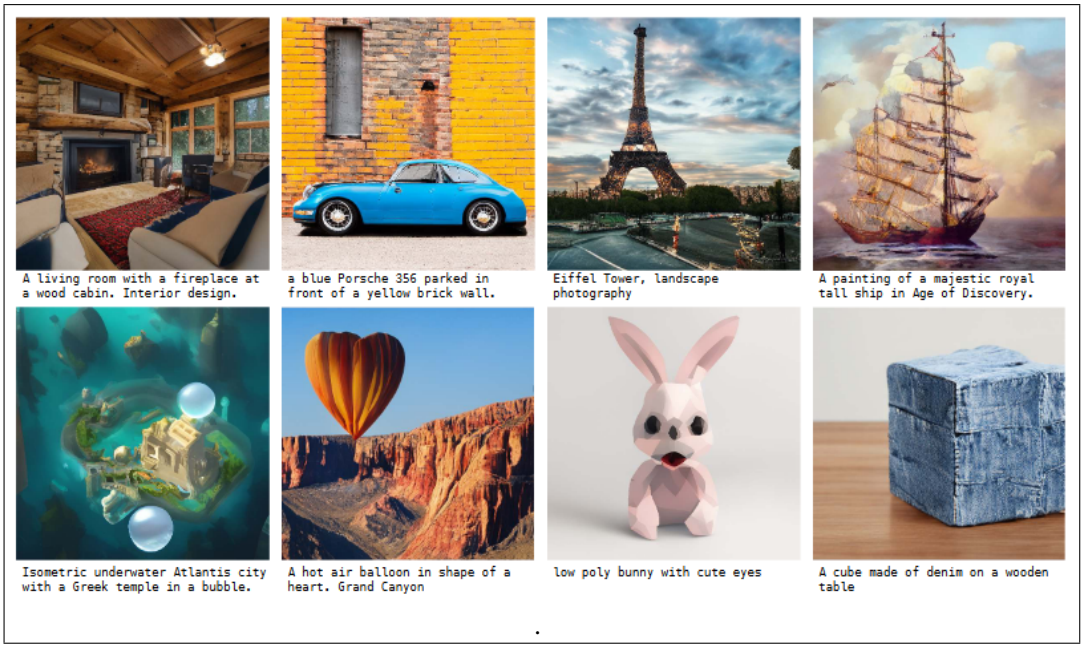

Figure 2.3: Example of images generated from descriptions by the GigaGAN model [23].

In this type of task, the prompt we pass to the model is very important (Figure 2.3). For example, it can happen that a word is ambiguous (Figure 2.4), generating an image that does not make much sense.

Figure 2.4: Ambiguity example of a street and a zebra (left) and a zebra on a gravel road (right) [24].

2.2.3 Visual Question Answering (VQA)

Visual Question Answering (VQA) [4] is a multimodal task that uses as input an image and a natural language question (Figure 2.5), and returns a natural language answer. In addition to being a multimodal task, it is a multidisciplinary task, as it mixes Computer Vision (CV), Natural Language Processing (NLP) and Knowledge Representation and Reasoning (KR).

Figure 2.5: Example of input image and various questions from the VQA dataset. As can be seen, all the answers can be answered only analyzing the image [4].

The most popular VQA dataset is the VQA V2 [25], which we will describe in detail in the next section.

2.2.3.1 VQA V2 datatset

VQA V2 [25] is an evolution of the original VQA dataset, currently being used as a benchmark for the VQA task. The dataset has 265,016 COCO images, where each image is associated with multiple candidate questions with at least 3 (5.4 questions on average) per image. These questions (1,105,904 questions in total) cover a wide range of topics and complexities, including questions that require reasoning, understanding relationships, counting, etc. Each of the questions has 10 ground truth answers.

Table 2.1: VQA V2 number of questions and images per split and total [25].

Split	Train		Val		Test		Total	
	Questions	Images	Questions	Images	Questions	Images	Questions	Images
Number	443,757	82,783	214,354	40,504	447,793	81,434	1,105,904	265,016

As can be seen in Table 2.1, VQA V2 is formed by three different splits, train, validation, and test.

2.2.3.2 VQA Score

The VQA [4] task proposes a standard metric called VQA Score (Equation 2.1). The idea behind this score, is to create a metric that is consistent with human variability in answer formulation, given that for the same question there may be discrepancies about the "correct" answer. Therefore, to be consistent with "human accuracies", the model answers are averaged over all 10 choose 9 sets of human annotators. That is, to calculate the VQA score, each annotation is compared with the other 9 annotations to calculate the metric and then the average of all the results is calculated as the final result.

This metric chooses the minimum between $\frac{x}{3}$ and 1, where x is the number of times our answer appears in the 9 annotations. Considering that, the answer of a model is **fully correct** when the answer (x) **appears at least three times** in the answers provided by the annotators.

$$acc = min(\frac{x}{3}, 1) \tag{2.1}$$

For example: If our model generates an answer "**dog**" and the 10 ground truth answers are: **dog, dog, cat, cat, cat, cat, cat, platypus, platypus, penguin.** We compare it to the 10 possible subsets of 9 annotators:

- _, dog, cat, cat, cat, cat, cat, platypus, platypus, penguin $\rightarrow \frac{1}{3}$
- dog, _, cat, cat, cat, cat, cat, platypus, platypus, penguin $\rightarrow \frac{1}{3}$
- dog, dog, _, cat, cat, cat, cat, platypus, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, _, cat, cat, cat, platypus, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, _, cat, cat, platypus, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, cat, _, cat, platypus, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, cat, cat, _, platypus, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, cat, cat, cat, _, platypus, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, cat, cat, cat, platypus, _, penguin $\rightarrow \frac{2}{3}$
- dog, dog, cat, cat, cat, cat, cat, platypus, platypus, _ $\rightarrow \frac{2}{3}$

Averaging all the results, we obtain the VQA score:

VQA Score $= \frac{(2\cdot(\frac{1}{3})+8\cdot(\frac{2}{3}))}{10} = 0.6$

2.2.3.3 VQA leaderboard

The most successful approaches for VQA V2 [26] are shown in Table 2.2. All of them are based on multimodal transformers (explained in Section 2.3) such as **PaLI-X** and **PaLI** [27], a combination of a vision and multilingual transformers, **BeiT-3** [28], multiway transformer, **mPlug** [29], Vision-Language Learning by Cross-modal Skip-connections, **CoCa** [30], image-text encoder-decoder model, and **Git2** [31], single image encoder and a

text decoder. We can see that the best **VQA Score is 86.06** obtained by the PaLI-X approach (Table 2.2), followed by a few approaches that obtain around 84.

Model	VQA Score
PaLI-X [27]	**86.06**
PaLI [27]	84.34
BeiT-3 [28]	84.18
mPlug [29]	84.08
CoCa [30]	82.33
Git2 [31]	81.92

Table 2.2: Overall results of top models of VQA task [26].

2.2.3.4 VQA variants

VQA task has been one of the most popular benchmarks in the CV and NLP community in recent years. In addition to having challenged research, it has opened up a new horizon. Since it has been seen that it is not only a task of visual recognition, but also of understanding the environment and incorporating knowledge about it. From this, new variants of this task are born, seeking to investigate different aspects.

These variations highlight different challenges and requirements, such as complex reasoning, text comprehension, external knowledge, and factual knowledge acquisition. They broaden the scope of VQA research and drive advances in areas such as visual comprehension, linguistic reasoning, and external knowledge. We list a few of these:

1. **GQA (Visual Genome Question Answering):** GQA [32] is a variant that focuses on more complex visual reasoning and understanding. It builds upon the Visual Genome dataset and contains questions that require detailed scene understanding, spatial relationships, object properties, and logical reasoning (Figure 2.6). GQA pushes the boundaries of VQA by emphasizing deep comprehension and reasoning abilities.

Figure 2.6: Example of images and questions in the GQA datasets [32]. As can be seen, the questions require understanding objects, relationships between objects, attributes, etc.

2. **TextVQA (Text-based VQA):** TextVQA [33] is a variant of VQA that requires answering questions based on text present in the image, such as signs, labels, or captions. Models must read and comprehend the textual information in the image to generate accurate answers (Figure 2.7).

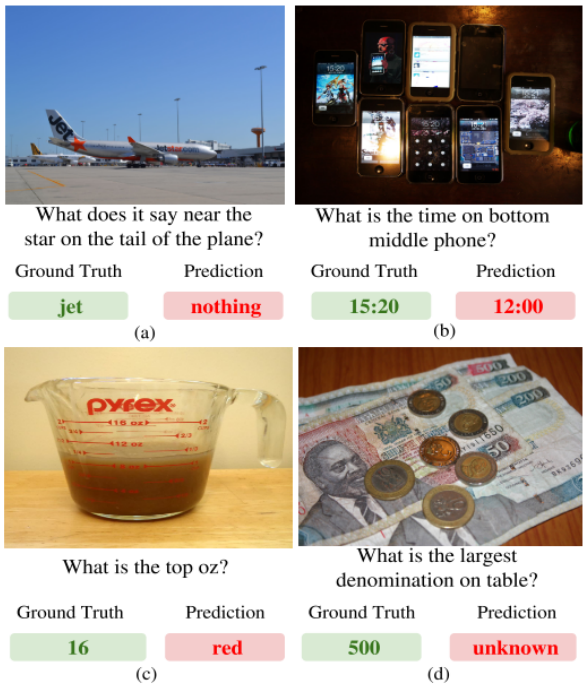

Figure 2.7: Example of images and questions in the Textvqa datasets [33].

3. **FVQA (Fact-based VQA):** FVQA [34] focuses on answering fact-based questions that require knowledge beyond what is visually depicted in the image. Models need to incorporate common sense reasoning to answer questions correctly (Figure 2.8).

Figure 2.8: Example of an image, question and common sense in the FVQA datasets [34].

4. **OK-VQA (Outside Knowledge VQA):** OK-VQA [5] inserts into the VQA task the challenge of reasoning about visual content, understanding the question, and applying external knowledge to answer questions about images where the answer is not explicitly stated (Figure 2.9).

Figure 2.9: Example of images and questions that need external knowledge to be answered from the OK-VQA datasets [5].

2.3 Multimodal systems

Multimodal systems refer to computer systems designed to perform multimodal tasks. There are different architectures used for multimodal problems: from more traditional ones such as multimodal RNNs [19], through Generative Adversarial Networks (GANs) [21, 23, 35], to Diffusion models [22], to multimodal transformers [1, 2, 3, 8, 36] and so on. Multimodal transformers have stood out for their ability to efficiently model and merge information from different modalities. These systems extend the transformer architecture [7], based on attention mechanisms, initially developed for translation.

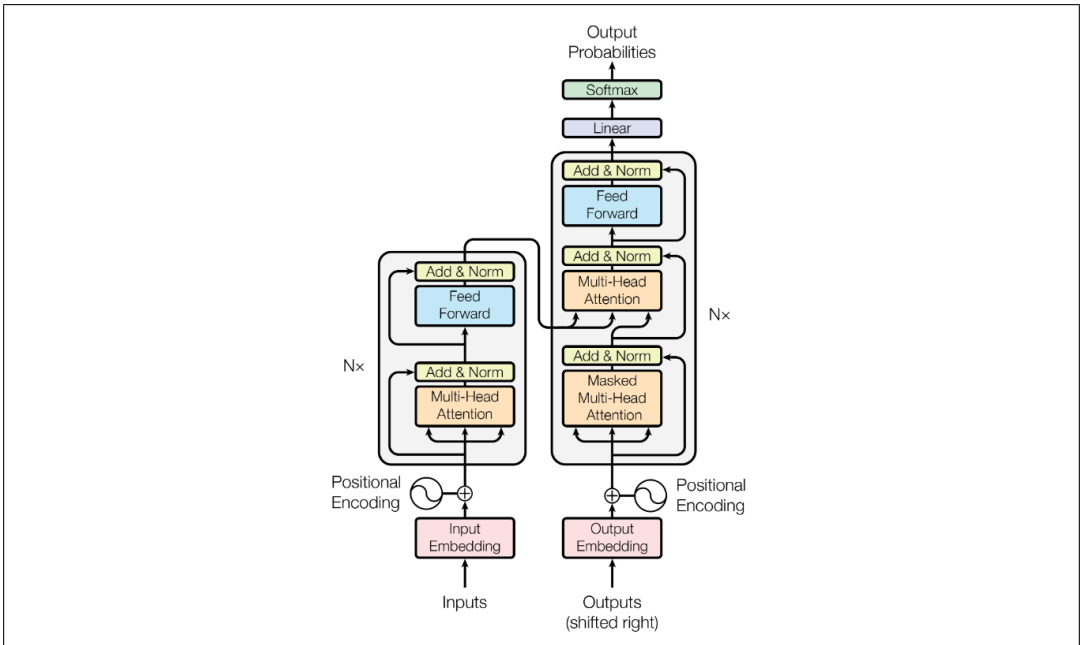

Figure 2.10: Transformer architecture consists of two parts: Encoder on the left and decoder on the right [7].

Transformer architecture consists of two parts: **Encoder** (left side of the Figure 2.10), receives an input and constructs a representation of its characteristics. **Decoder** (right side of the Figure 2.10), it uses the encoder representation with other inputs to generate a target sequence. Depending on the task, the parts can be used independently or in combination. In multimodal problems, there are two main approaches: many adopt the encoder-only architecture, with the intermodal representations being fed directly into an output layer. On the other hand, other models adopt an encoder-decoder architecture, in which the intermodal representations are first fed into a decoder and finally into an output layer [8].

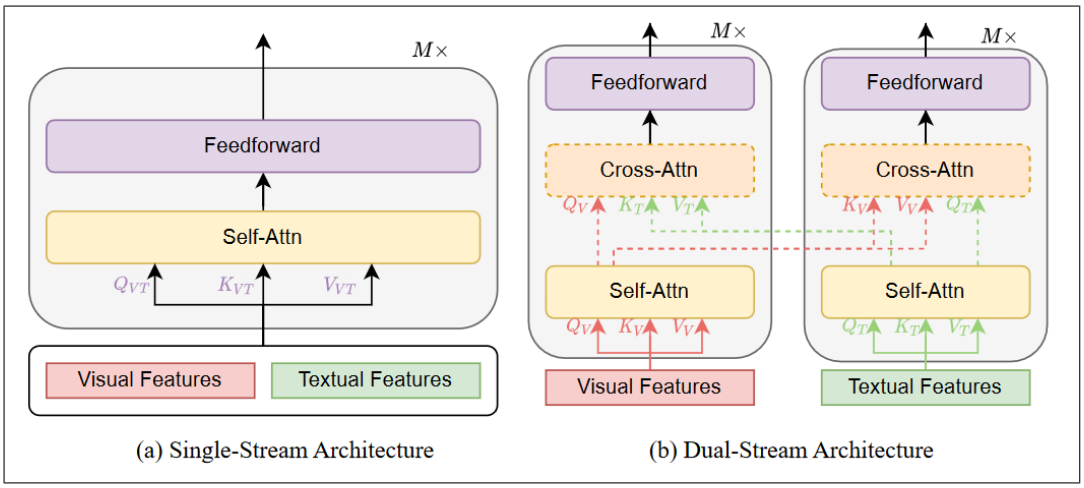

Figure 2.11: Illustration of the two multimodal transformers categories. (a) Single-stream and (b) Dual-stream [8].

Multimodal transformers can be categorized in **single-stream** and **dual-stream** (Figure 2.11) [2, 8, 36]. Single stream transformers need a model to extract the visual features of the image (usually a pretrained object detector which extracts object-region features). Once the visual and linguistic features of an image and text pair are obtained, they are concatenated and fed as input to the transformer (Figure 2.11 (a)). The best known single-stream multimodal transformers are based on the BERT architecture [37], such as VisualBERT [38], which uses Faster R-CNN [39] as its object detection model. Other examples are OSCAR [40], which is similar but uses different pretraining strategies, and OSCAR+ [41], an enhanced version of OSCAR that uses an improved object detector called VinVL.

Dual-stream transformers use a dedicated transformer for each modality, i.e., visual and textual features are not concatenated and are sent independently to each transformer. It is important to understand that these two transformers do not share parameters. Depending on whether we are looking for performance or efficiency, we will proceed differently: To achieve higher performance, cross-attention is used to allow cross-modal interaction (Figure 2.11 (b)). On the other hand, to achieve higher efficiency, there can be no cross-attention between the transformers. Example of this category are ViLBERT [42], LXMERT [43], and ERNIE-VIL [44].

2.4 Object Detection

As object detectors have been widely used for multimodal transformers and as we also use object detectors for image verbalization in this research, it is very important to understand how they work. Object detection is a fundamental technique for extracting information about the presence, location, and classification of objects within an image. It involves identifying and localizing specific objects within an image by drawing bounding boxes around them (Figure 2.12). Object detection algorithms employ various approaches, region-based methods like Faster R-CNN [39] or single-shot methods like YOLO [45].

Figure 2.12: Example of objects and attributes obtained by VinVL where we can see the bounding boxes of each one of them [41].

Currently, VinVL [41] model obtains state-of-the-art results. This object detection model is based in a ResNeXt-152 C4 (X152-C) architecture. It is larger, better designed for vision-language tasks, and has been pretrained with much larger training corpora than the predecessor models. Therefore, it can generate representations of a richer collection of visual objects, 1848 object categories and 524 attribute categories. The output can include the labels of detected objects, attributes, their locations, image dimensions and number of bounding boxes.

We can observe in Figure 2.13 output, that the objects “class” are ordered according to the confidence value "conf". The attributes of the objects “attributes” are also ordered by the confidence value “attr_scores”. The key “rect” refers to the $[x_1, y_1, x_2, y_2]$ coordinates of the object’s bounding box. In the last row, we have the image dimensions, image height “image_h” and image width “image_w”, and the number of bounding boxes “num_boxes”.

Input

Output

{"objects": [{"rect": [358.715, 181.724, 499.445, 258.509], "class": "shadow", "conf": 0.965,
"attributes": ["dark", "black", "cast", "long"], "attr_scores": [0.526, 0.134, 0.079, 0.056]},
{"rect": [22.803, 12.826, 140.126, 97.61], "class": "sun", "conf": 0.949, "attributes": ["bright",
"shining"], "attr_scores": [0.579, 0.32]},
{"rect": [207.581, 25.833, 401.316, 136.504], "class": "cloud", "conf": 0.877, "attributes":
["white", "fluffy"], "attr_scores": [0.731, 0.086]},
{"rect": [349.04, 116.223, 425.149, 228.069], "class": "backpack", "conf": 0.865, "attributes":
["black", "large", "gray"], "attr_scores": [0.678, 0.185, 0.052]},
"image_h": 333, "image_w": 500, "num_boxes": 32}

.

Figure 2.13: Example of objects and attributes obtained from an image of the COCO [46] dataset with VinVL and obtained output representation.

CHAPTER 3

Preliminaries

In this chapter, we will present the preliminary information necessary for the development of our research. We will present the task that is the focus of our research, Outside-Knowledge Visual Question Answering (OK-VQA), and the Caption Based Model (CBM) that we have used as a base model.

3.1 Outside-Knowledge Visual Question Answering

In this section, we will present the task that is the focus of our research, Outside-Knowledge Visual Question Answering (OK-VQA). We will name the differences with respect to the initial task VQA, and we will expose the different approaches and results in the state-of-the-art. We will then present and elaborate on the dataset used as a benchmark, called OK-VQA.

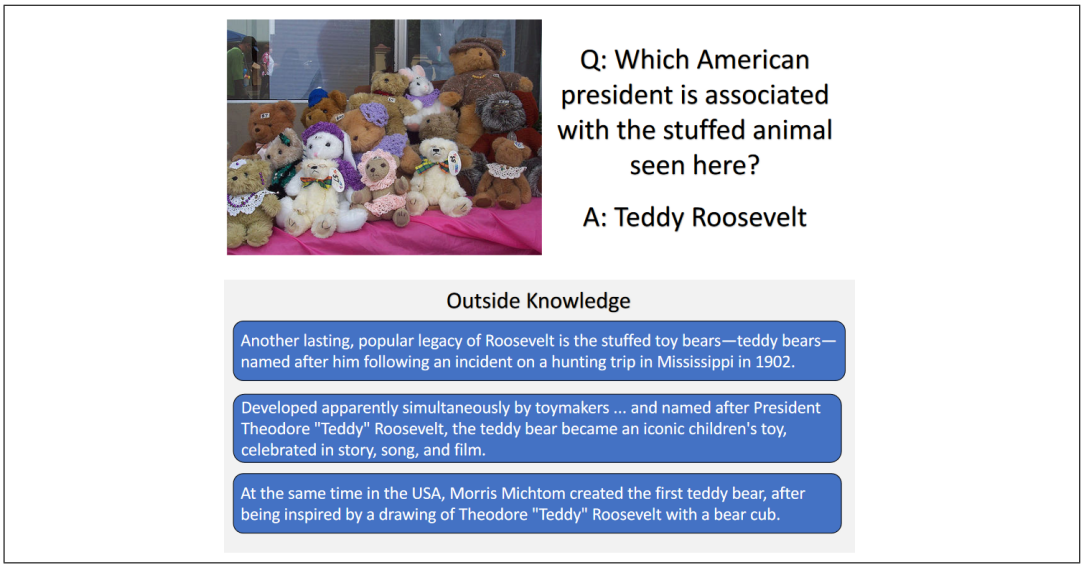

Figure 3.1: Example of an instance that requires external knowledge to be answered from OK-VQA [5]

3.1.1 Task description

OK-VQA inserts to the VQA task the challenge to understand the environment and apply external knowledge in order to answer the questions (Figure 3.1) [5].

To generate the response in the outside knowledge scenario, the model will need an external knowledge source. The knowledge source is typically categorized into two types [47]: Symbolic Knowledge and Implicit Knowledge.

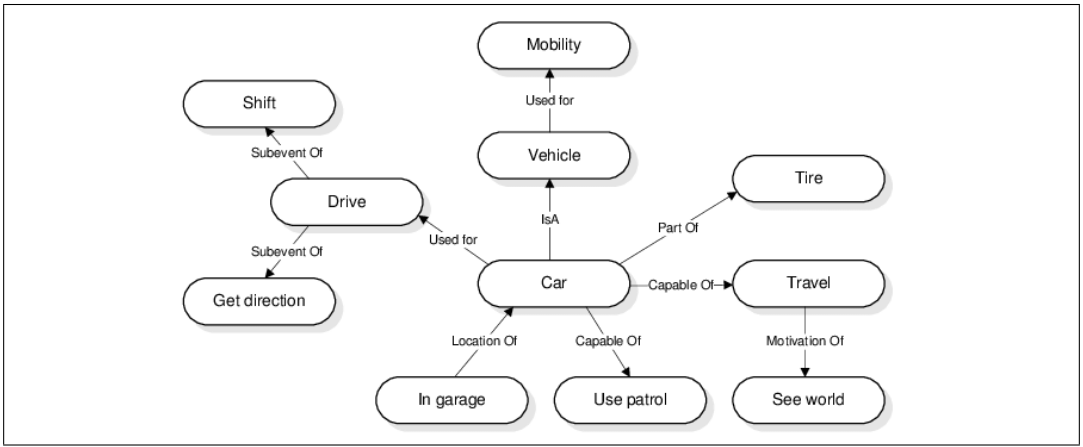

Figure 3.2: Example of concepts related to car in ConceptNet [48, 49]

Symbolic Knowledge is exemplified by ConceptNet [48], which connects natural language using labeled edges and gathers information from multiple sources, including expert input, purposeful games, and crowdsourcing (Figure 3.2). Its primary aim is to represent comprehensive implicit knowledge. On the other hand, **implicit knowledge** refers to the information embedded in the parameters of a model. This knowledge is derived from model training and is usually based on sources such as Wikipedia, Google Search, Google Images, concepts, captions, and so on. These sources provide the model with a wide range of implicit knowledge. This allows answering questions that refer to elements of the image, but in which the answer is not explicitly present (Figure 3.3).

Figure 3.3: Difference between VQA V2 and OK-VQA data [9].

As in the VQA task, the main results (Table 3.1) [6] are based on transformers as **Prophet** [50], Prompting Large Language Models with Answer Heuristics, **PromptCap** [51], Prompt-Guided Task-Aware Image Captioning, **REVIVE** [52], Regional Visual Representation method, **KAT** [53], Knowledge Augmented Transformer, **PICa** [54], Prompts GPT-3 via the

use of Image Captions, **CBM** [9], Caption Based (text only) Model, **MCAN** [55], Modular Co-Attention Networks, **UnifER** [56], Unified End-to-End Retriever-Reader Framework, **MAVEx** [57], Multi-modal Answer Validation using External knowledge, **KRISP** [47], Knowledge Reasoning with Implicit and Symbolic rePresentations and **ConceptBert** [58], BERT based in elements of the image and a Knowledge Graph.

Model	Overall accuracy
Prophet [50]	**61.11**
PromptCap [51]	60.4
REVIVE [52]	58.0
KAT [53]	54.41
PICa [54]	48.0
CBM [9]	47.9
MCAN [55]	44.65
UnifER [56]	42.13
MAVEx [57]	41.37
KRISP [47]	38.90
ConceptBert [58]	33.66

Table 3.1: Overall results of top models for OK-VQA task [6].

In terms of performance, we can see that the **best VQA Score is 61.11** obtained by the Prophet approach (Table 3.1), we can see that **there is still room to improve compared to the VQA task** (Table 2.2). Our starting point, the CBM model, is ranked sixth, tied with the fifth position. In Section 3.2, we will take a closer look at the CBM model.

3.1.2 Dataset description

To conduct the experiments, we will use the OK-VQA dataset [5] as benchmark with images, questions, and answers. These data are diverse, difficult and, require knowledge, proposing questions from multiple categories that require external knowledge to be answered (Figure 3.4).

Figure 3.4: Example questions and corresponding images and answers for each of the knowledge categories from the OK-VQA dataset [5].

The images belong to the COCO [46] dataset, having finally selected 14031 images. In total, the dataset has 9009 training questions and 5046 test questions, for a total of 14055 questions (Table 3.2).

Table 3.2: Number of questions and images per split and total.

Split	Train		Test		Total	
	Questions	Images	Questions	Images	Questions	Images
Number	9009	8998	5046	5033	14055	14031

Each of the questions has a list of 10 answers, but there are actually 5 ground truth answers per question (5 annotators). This is important to understand as the VQA score will be used as the reference metric and was originally proposed in the VQA task which had 10 ground truth answers. Therefore, we will use each answer twice, thus obtaining the 10 answers mentioned at the beginning (Figure 3.5).

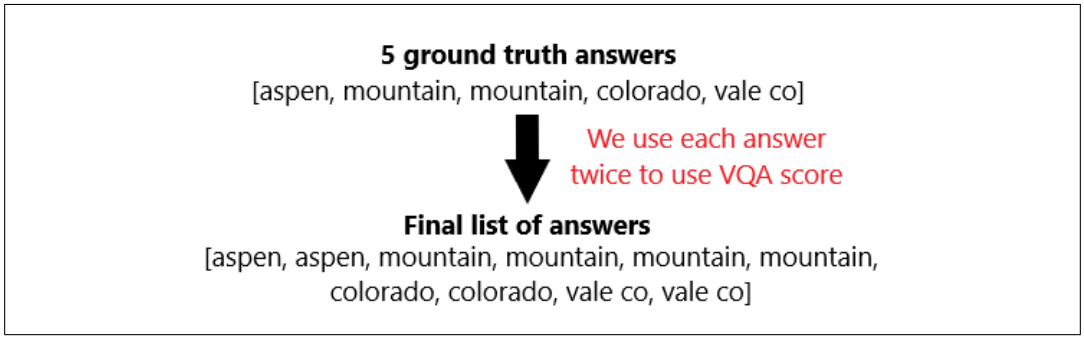

Figure 3.5: Explanation of the list of 10 answers of 5 ground truths.

It is important to understand this, given that when it comes to evaluating the results, everything will be peer-reviewed. That is, an answer can appear 0, 2, 4, 6, 8 or 10 times in the final list of answers.

3.2 Caption Based Model

In this section, we will present the Caption Based Model (CBM) that we have used as the basis for our research. To do so, we will explain its structure, operation, models used and results obtained.

We have reviewed the main approaches using transformer models for the OK-VQA task (Table 3.1). Due to our limited computational resources, specifically a small GPU, our goal was to find a model that could fit our constraints and would be a good basis for our research. During our analysis, we discovered that the authors of the CBM model had conducted a comparison study involving models of different sizes. Therefore, we have decided to adopt the CBM model as our starting point, due to its compatibility with smaller GPUs, aligning perfectly with our computing resources. This allows us to directly compare our results with those obtained from the CBM. Consequently, our proposal will build upon the approach and conclusions derived from the CBM research.

3.2.1 Description of CBM

CBM [9] showed that a text-only model, using the automatically generated image caption and the knowledge gained by the language model during pretraining, equals the previous state-of-the-art results. The model consists of two parts (Figure 3.6), a caption generation system that generates a description of an image and a language transformer model that takes as input the caption and the question, and generates an answer.

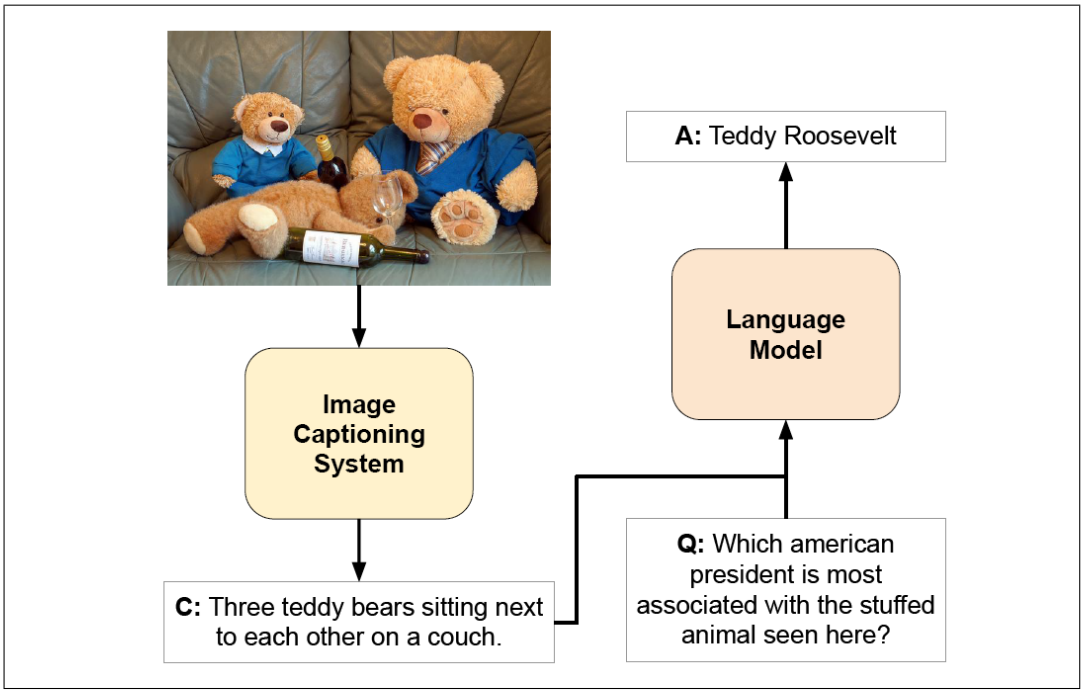

Figure 3.6: Structure of CBM with an example of image and question input and answer generation [9]. The model is composed of an Image Captioning System and a Language Model.

3.2.1.1 Caption generation system

The image captioning system is based on OSCAR [40], a pretrained multimodal transformer that produces state-of-the-art results in several multimodal tasks. Its goal is to generate descriptive textual captions that accurately represent the visual content of an image (Figure 3.7). For that, the model takes an input image and uses a pretrained object detector called Faster R-CNN [39] to obtain the region features of the images and their respective labels. This captures the visual features and representations of objects in the image. Additionally, the model incorporates textual information by encoding the caption using a language transformer. Then both representations are fused together using attention mechanisms to generate a joint representation. This joint representation is used to predict the next word in the caption sequence. During training, the Oscar captioning model is trained on a large dataset of images paired with their corresponding captions. It learns to associate the visual content of the image with the textual descriptions and generates captions that are semantically aligned with the image content.

Figure 3.7: Example of a caption obtained from an image from COCO dataset with OSCAR multimodal transformer. As can be seen, it is a general description of an image without much detail.

3.2.1.2 Language Model

Once the caption is obtained from the image, the question, and caption are passed to a language model. To facilitate learning the model, the prefixes 'question:' and 'caption:' are added. Here, we show an example input for the language model:

Input → *question: Can you guess the place where the man is playing? caption: a man riding a snowboard down a snow covered slope.*

For the language model, two pretrained Large Language Models (LLM) are compared (Table 3.3): Bert [37], a Deep Bidirectional Transformer, and T5 [59], a Text-To-Text Transformer.

Model	Score	Parameters
CBM_{Bert}	36.0	112M
CBM_{T5-11B}	**47.9**	11B

Table 3.3: Comparison between the Bert and T5 models implemented by the authors and the number of parameters of each one [9].

As can be seen, T5 obtains better results (Table 3.3), 11.9 points more, but the number of parameters is considerably higher also. Based on the T5 model, they test different sizes, and it is observed that the larger models obtain better results (Table 3.4). But larger models also need more computing power, and as mentioned above, we are limited to using a single GPU. So from now on we will use the T5_{Base} model as a reference and compare it with our proposal.

Model	Score	Parameters
$\text{CBM}_{T5-Small}$	29.2 $\pm$0.2	60M
$\text{CBM}_{T5-Base}$	36.1 $\pm$0.5	220M
$\text{CBM}_{T5-Large}$	40.8 $\pm$0.4	770M
CBM_{T5-3B}	44.0 $\pm$0.7	3B
CBM_{T5-11B}	**47.9** $\pm$0.2	11B

Table 3.4: CBM_{T5} performance of model range sizes and number of parameters [9].

3.2.2 T5 model

The T5 (Text-to-Text Transformer) [59, 60] model is a large pretrained language model that aims to overcome the limits of transfer learning in the field of natural language processing (NLP). It is a generative encoder-decoder transformer that obtains state-of-the-art results on text-only question answering tasks. As we have seen in the CBM comparison (Table 3.3), the model is available in different sizes, ranging from 60M parameters to 11B.

The model is trained using the teacher forcing technique, i.e., the model is provided with the correct target sequence (ground truth labels) at each training step. This helps the model learn the correct dependencies between input and output tokens and improve its ability to generate accurate output sequences. It can help the model converge more quickly during training and generate more accurate results.

It is important to keep in mind that, although the task we are going to face is multilabel, during training the model only receives one label for each question in each epoch.

CHAPTER 4

Leveraging Multilabel Annotations

In this chapter, we present our first contributions, called multilabel leverage. This is a new technique proposed to improve the learning process of a model in multilabel problems. Once the technique has been introduced, we will present the experiments performed and the results obtained.

4.1 Description of the approach

Our first contribution is a simple technique that takes advantage of multilabel annotations with optimal and suboptimal solutions or answers to improve model learning. It is based on the introduction of the concept of balance between exploration and exploitation by using a frequency distribution. In this particular case, the OK-VQA dataset contains a list of 10 answers for each question provided by 5 annotators, and an answer counts as fully correct (optimal) if at least 2 annotators (at least 4 answers out of 10) have given that answer. In the case of having only 1 annotator (2 answers out of 10) the answer is considered partially correct (suboptimal), and counts as 0.6.

In the original CBM implementation [9], a random answer is passed to the model during each training epoch, but this answer is randomly chosen only among those that are fully correct (Figures 4.1 and 4.2), i.e., answers that appear at least 4 times. This considerably reduces the number of possible answers that we can pass to the model during training, since at most we can have 2 possible answers that meet the condition of getting a score equal to 1. As can be seen (Figure 4.1), if we analyze the number of possible answers for the answers that score 1, 7611 of the instances (85.55%) have only one possible answer, and the rest, 1286 have 2 possible answers. On the other hand, if we analyze the distribution of All answers, we can observe that in 3425 instances (38.5%) we have 4 possible answers, in 2497 instances (28.07%) 3 possible answers, in 1931 instances (21.7%) 2 possible answers and finally 1044 instances (11.73%) have only one possible answer. The percentages of Answers with score = 1 are also important for our implementation, as they indicate that 85.55% of the instances have a single optimal answer, while 14.45% have two optimal answers. The technique proposed will take advantage of all these possible answers to improve the learning of the model.

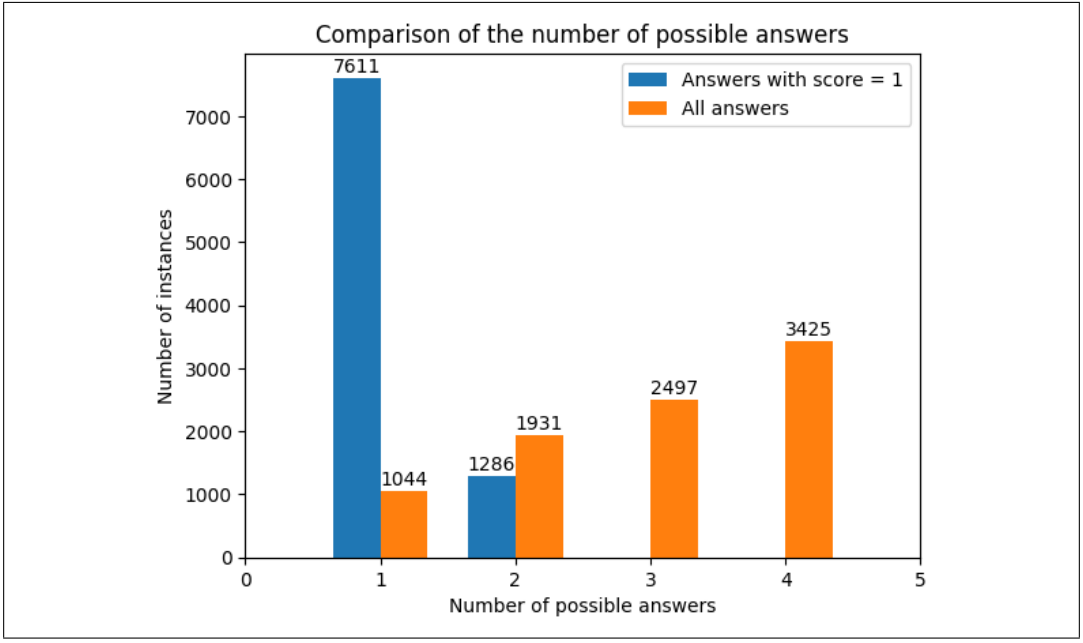

Figure 4.1: Comparison of the number of possible answers during training for the original implementation (Answers with score = 1) and multilabel leverage technique (All answers).

In our approach, we propose to use **all possible answers** (Figure 4.1), but we use a **frequency distribution** based on the proportion of the annotations (Figure 4.2). This is important, since the probability with which suboptimal answers are selected should be controlled and not too high, since it would then impair learning. Thus, we give more probability of being chosen to the answers that are fully correct, but also, the partially correct answers have a probability of being chosen. In this way we introduce the concept of **balance between exploration and exploitation**, in most cases we provide the optimal answer (exploitation) and with a small probability we provide suboptimal answers that enhance learning (exploration).

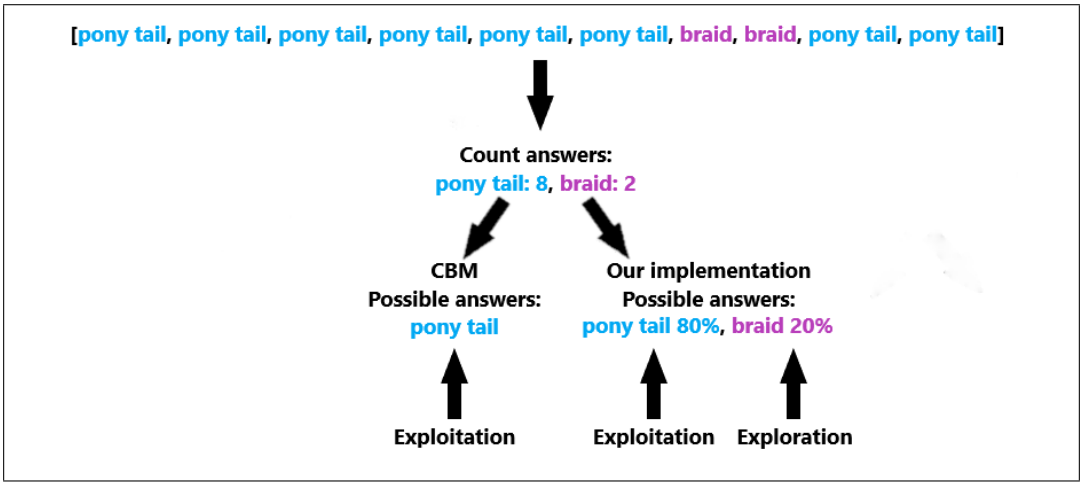

Figure 4.2: Comparison between CBM, with only one possible answer, and our proposed **multilabel leverage technique**, with all possible answers, their percentage of choice being equal to the proportion of times they appear in the list.

The hypothesis behind this idea is that this approach will maintain the correct answers given by CBM, but will also increase the number of partially correct answers. Exploration introduces a small variability into the model that enhances learning and increases the network of possible answers. This variability gives more "points of view" to the model than simply using fixed answers. This allows the model to maintain the number of fully correct answers, but substantially improves the accuracy of partially correct answers. Since with the previous implementation, the model was "limited" to a problem with a single solution for most training instances (Figure 4.1), but, instead, with several possible answers in the validation.

4.2 Experimental set up

In this section, we will present the main points of the implementation. To do so, we will expose the evaluation metric, the implementation details, such as the loss function, the optimizer, etc., and finally, we will present the main software and hardware used.

4.2.1 Evaluation metric

As evaluation metric, we will use the VQA Score (Equation 4.1) proposed in the VQA task [4] and presented in Section 2.2.3.2.

$$acc = min(\frac{x}{3}, 1) \tag{4.1}$$

In the **OK-VQA** case, as we have 5 duplicated ground truth answers, the answer needs to appear **at least 4 times** (4 of 10) in the answer list to be **fully correct, obtaining a score of 1**. If the answer appears **twice**, the answer is considered as **partially correct, obtaining a score of 0.6**, and if it does not appear it is incorrect, obtaining a score of 0.

4.2.2 Implementation details

In this section we will present the loss function, the optimizer and the hyperparameters used during the experiments. For direct comparison with CBM, **unless otherwise indicated, the hyperparameters used will be the same as those shown in this section and those proposed by the authors in CBM [9].**

4.2.2.1 Loss function

We will use the Cross Entropy Loss (Equation 4.2) for training the model. It is the default loss function of the T5 model [59].

$$\mathcal{L}_{CE} = -\sum_{c=1}^{M} y_{o,c} \log(p_{o,c}) \tag{4.2}$$

Where $y_{o,c}$ is the ground truth answer and $p_{o,c}$ is the Softmax probability of the c^{th} class.

4.2.2.2 Optimizer

We used the AdamW [61] optimizer proposed by the authors of CBM.

4.2.3 Hyperparameters

1. **Learning rate:** Constant $5 \cdot 10^{-5}$
2. **Batch size:** The batch size for training and validation is 56.
3. **Source and target maximum length:** The maximum length is 512.
4. **Training steps:** The number of training steps is 20k.

4.2.4 Implementation resources

In this section, we will present the software and hardware used during implementation.

4.2.4.1 Software

We will use Python as base language and Pytorch [62], Hugging Face Transformers and Datasets [63] as main libraries.

Pytorch: PyTorch is an open source machine learning framework used to develop and train artificial intelligence models, especially in the field of deep learning. It is one of the most popular libraries with an active and rich community.

It was developed by Facebook Artificial Intelligence Research (FAIR) group and has become very popular for its flexibility and ease of use. It provides a wide range of tools and functionalities to facilitate the development of machine learning models. In our case, it has made it easy for us to use the AdamW optimizer and the CrossEntropyLoss loss function, being called directly from the library. In the implementation, it has facilitated the manipulation of instances when loading and processing data to the model through the dataloader among others.

Hugging Face Transformers: Hugging Face Transformers is an open source library developed by the company Hugging Face. It provides a wide range of tools and state-of-the-art pretrained models for different tasks.

This library has provided us with the T5 model that is the basis of our proposal. These pretrained models can be used directly to perform tasks, or they can be adapted and tuned to specific datasets or tasks through the process of fine-tuning, as we have done. In addition, it provides us with all the tools related to the model, such as the Tokenizer or other tools like the DataCollater used together with the Pytorch dataloader to pass batch data to the model. This library is designed to integrate seamlessly with Pytorch and Tensorflow.

Hugging Face Datasets: The Hugging Face Datasets library, like the Transformers library, has been created by the Hugging Face company. It provides tools for loading and processing custom datasets, such as predefined datasets covering a wide variety of tasks. It allows reading data from different formats and facilitates data preparation. In our case,

it has been especially useful for converting Dataframes into Datasets, thus facilitating subsequent preprocessing and the use of tools such as the aforementioned Dataloader. Like the Transformers library, it is designed to integrate seamlessly with Pytorch and Tensorflow.

4.2.4.2 Hardware

To perform the experiments, we used the Graphics Processing Units (GPUs) of the IXA group, a research group of the Faculty of Informatics of the University of the Basque Country. For the implementation and the experiments of the $T5_{Base}$ we have used a single GPU with 12 GB of memory, **Nvidia Titan XP** or **Nvidia Titan V** depending on availability.

These GPUs have been useful for our experiments, but due to their memory size they have determined the development of the experiments.

4.3 Experiments and results

To perform the experiments with the new multilabel leverage technique, the first thing we did was to implement the CBM model to ensure that we were starting from a good base. We followed the same methodology defined in the paper [9], performing 3 runs with each model and calculating the mean VQA score and standard deviation, the results of each run can be found in Appendix A. As in the CBM implementation, we have removed the 112 instances that do not have a fully correct answer. We have done this to make the comparison as fair as possible, although our implementation can actually use these instances too. Once similar results to the original implementation were obtained (Table 4.1), we modified the same model by introducing the multilabel leverage technique, adding the concept of balance between exploration and exploitation, based in a frequency distribution when choosing the labels at each step.

Table 4.1: Mean VQA score and standard deviation results for three runs of the CBM_{T5Base} and our multilabel leverage implementation (All answers). The results of each run can be found in the Appendix A.

Model	Labels	Mean VQA score
CBM_{T5Base}	Answers with score = 1	**36.1** ± 0.5
Our CBM_{T5Base}	Answers with score = 1	**36.09** ± 0.426
Our CBM_{T5Base}	All answers	**37.01** ± 0.329

As can be seen in Table 4.1, this simple technique **improves the VQA Score** obtained by the original model by **almost one point (0.92) on average**. Our hypothesis in proposing this technique was that it would maintain the number of fully correct answers of the original CBM implementation, but improve the number of partially correct answers. So we analyzed the answers generated by both models (Figure 4.3), we calculate the VQA score of each answer and analyze the number of each.

Our implementation slightly improves the number of fully correct answers (+6), but significantly increases the number of partially correct answers by 39 (Figure 4.3).

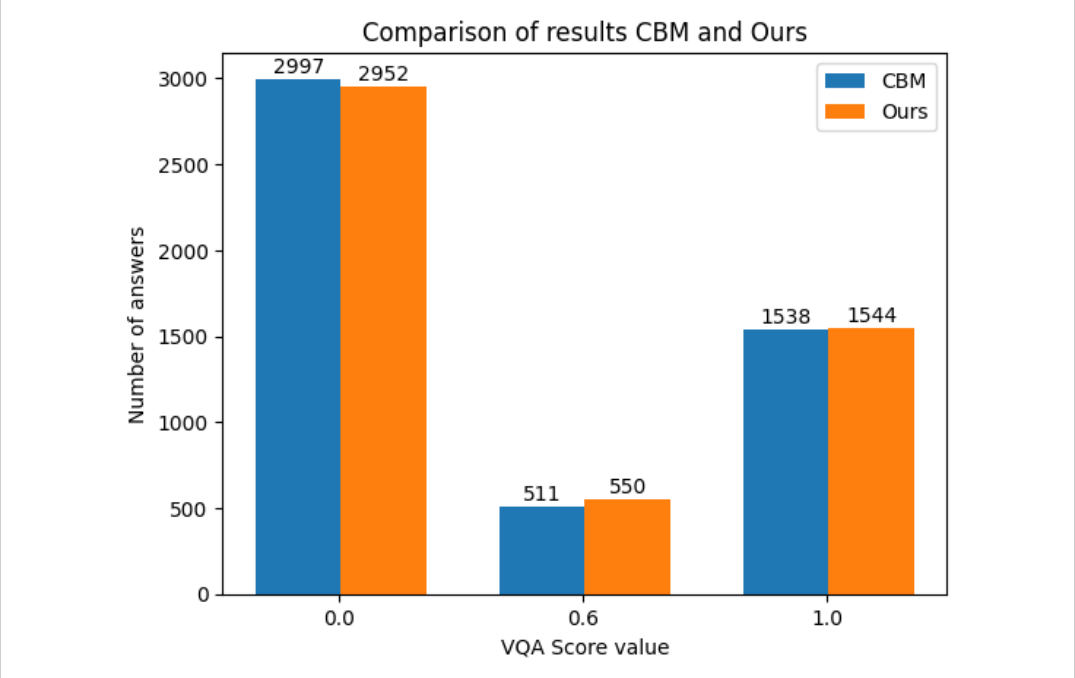

Figure 4.3: Comparison of the number of incorrect (0.0), partially correct (0.6) and fully correct (1.0) answers of the CBM and our implementation. Ours slightly improves the number of fully correct answers (+6), but increases by 39 partially correct answers.

The answers used for comparison come from runs that obtain a VQA score of 36.56 in the case of CBM and 37.14 in our implementation. We have a difference of 0.58 in favor of our implementation, with 0.46 coming from the increase in the number of partially correct answers versus 0.12 increase coming from the fully correct answers. This confirms the hypothesis that our multilabel leverage technique improves model learning by maintaining the number of fully correct answers and increasing partially correct answers. It should be noted that the average difference between models is greater than 0.58 (Table 4.1), assuming that on average the increase in partially correct answers is slightly higher.

In Appendix B we present more experiments where our technique is compared with the basic one, improving the results in all implementations. Thus confirming the improvement obtained in the CBM, and giving robustness to our proposal.

CHAPTER 5

Comparing Image Verbalization Approaches

In this chapter, we analyze and compare different image verbalization approaches. First, we use an object detector to obtain more detailed information from the image, such as objects and attributes. This new information is added to the input of the language model, complementing the information previously used by CBM, which was just an image caption. Second, since we have memory limitations and thus we cannot include all the objects and attributes extracted by the object detector in the input of the LM, a reranking system based on different approaches is presented to improve the quality of the information provided to the model. Once the techniques have been introduced, we will present the experiments carried out and the results obtained from each one.

5.1 Adding objects and attributes as input

As an addition to the caption, we propose to use an object detector to obtain the objects and attributes of the images. To do so, we select VinVL [41], a state-of-the-art object detector. As we have seen in figures 2.13 and 3.7, the object detection provides more details and the caption provides a good general description of the image. By combining both sources of information together, we seek a balance between general image information and details to provide the model with more complete information.

We have analyzed two approaches: **(i) only objects** and **(ii) objects and attributes**. Here, we provide two examples of how the input varies for each approach:

- **Input only objects** → question: Can you guess the place where the man is playing? caption: a man riding a snowboard down a snow covered slope. **objects:** shadow sun cloud backpack sky ski…
- **Input objects and attributes with template** → question: Can you guess the place where the man is playing? caption: a man riding a snowboard down a snow covered slope. **object shadow** *has attributes dark black cast long* **object sun** *has attributes bright shining* **object cloud** *has attributes white fluffy* **object backpack** *has attributes*

black large gray **object sky** *has attributes blue cloudy white* **object ski** *has attributes black blue...*

And we have also compared template entries (as in the previous example) and simple entries for objects and attributes. The idea of implementing a template is to make it easier for the model to understand and identify the information coming from the objects and the relationship with their attributes. But since we have limited memory, this limits the amount of information we can pass to the model, so we will implement a plain input and compare it with the template one.

- **Input objects and attributes plain** → question: Can you guess the place where the man is playing? caption: a man riding a snowboard down a snow covered slope. **objects: shadow** *dark black cast long* **sun** *bright shining* **cloud** *white fluffy* **backpack** *black large gray* **sky** *blue cloudy white* **ski** *black blue...*

5.1.1 Experiments and results

First, we use the VinVL [41] object detector to obtain the objects and attributes of all the images in the COCO [46] dataset. Once we have the objects, we match them with the questions using the image ID.

As noted in Chapter 4, the proposed multilabel leverage is beneficial to the model. From now on, it will be used in all experiments. Starting from the previous model as a basis, we have added the objects and attributes obtained with the VinVL model to the model inputs.

As mentioned above, we have made one implementation only with objects **in list format, objects: A B C D**, and another, with the objects and attributes using a **template, object A has attributes B C D**. Since we have limited memory capacity, we have reduced the **training batch size to 24** and **the maximum source and target length to 192**. This is not optimal, but we have tried to maintain a balance between execution time and model input length that allows us to experiment and get good results. Since we want to keep as much information as possible, reducing the input length further does not make sense, and increasing it to 256 for example forces us to reduce the batch size below 12 instances by increasing the runtime.

As in the previous experiments, we have performed 3 runs with each of the models and calculated the mean VQA score and standard deviation (Table 5.1), the results of each run can be found in the Appendix B.

Table 5.1: Mean VQA score and standard deviation results for three runs of our CBM$_{T5Base}$ and the implementations with object and object and attributes. The results of each run can be found in the Appendix B.

Model	Mean VQA score
Our CBM$_{T5Base}$	**37.01** $\pm$0.329
Caption + obj-attr	**39.09** $\pm$0.264
Caption + obj	**38.09** $\pm$0.278

We can see in Table 5.1 that adding more information improves the original model, especially in the case of **objects and attributes**, where it increases the average score by

more than **2 points** (2,08). In the case of **objects**, the improvement is **slightly lower**, 1.08. This confirms the idea that more information improves the model's ability to generate correct answers.

Once we see that adding objects and attributes improves the model performance considerably, we analyze whether the use of templates to form the input is beneficial. To do this, we analyze the length of the input and how much information is left out after truncation. First, we plot the distribution and quartiles of the training data set with the template.

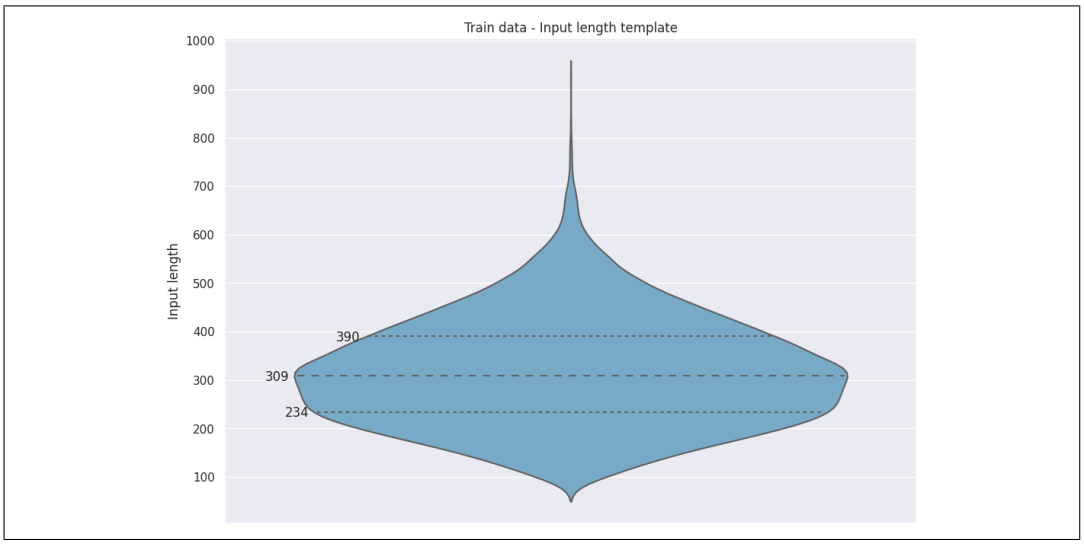

Figure 5.1: Violin plot of question and caption plus objects and attributes with template input length. Shows distribution and quartiles.

We can see in Figure 5.1, that with a truncation of 192 and using a template for objects and attributes we are receiving the complete information in less than 25% of the cases. We are going to make the same plot but without using the template, i.e., plain input.

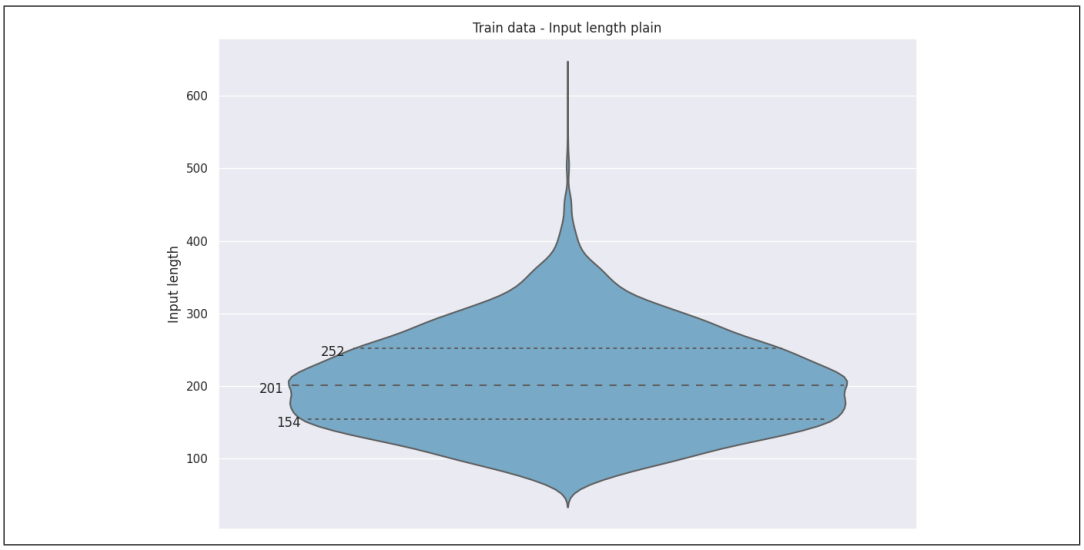

Figure 5.2: Violin plot of question and caption plus objects and attributes plain input length. Shows distribution and quartiles.

We can see that in the case of plain input (Figure 5.2), we are using complete information in almost 50% of the instances. This is a considerable increase in information compared to the previous model, ensuring that almost 50 percent of the instances receive all available information.

If we obtain the statistics of both models (Table 5.2), template and plain, we can observe that the average length of the template is 50% greater than that of the plain.

Table 5.2: Input length statistics of template and plain models.

Model	Mean	Standard Deviation	Minimum	25%	50%	75%	Maximum
Template	317.37	111.74	86.00	234.00	309.00	390.00	923.00
Plain	206.51	70.08	56.00	154.00	201.00	252.00	625.00

We have implemented the above model but replacing the template with a plain object and a list of attributes. As before, we performed 3 runs and calculated the mean VQA score and standard deviation, the results of each run can be found in Appendix C.

Table 5.3: Mean VQA score and standard deviation results for three runs of the Object attributes implementations. The results of each run can be found in the Appendix C.

Model	Mean VQA score
Caption + obj-attr with template	**39.09** ± 0.264
Caption + obj-attr plain	**39.23** ± 0.354

We can see in Table 5.3 that the plain information slightly improves the model with template, but the improvement is not significant. With those results, we reach several conclusions: first, that in case of limited memory, the template does not improve the learning of the model, and second, that increasing the information slightly improves the results.

In Section 5.2, we will try to improve the results by increasing the quality of the selected objects and attributes through a reranking system, and we will analyze the influence on the results of the number of selected objects.

5.2 Object reranking system

Since we have memory limitations and cannot pass all objects and attributes, our third proposal is to maximize the "quality" of the objects and attributes we pass to the model. The idea is to sort and filter the objects and attributes with a reranking system.

5.2.1 Proposed methods

To improve results in cases of limited memory where we cannot use all objects and attributes, we propose two different methods: **(i) Sentence similarity** and **(ii) Object bounding box area**. The first one is based on FastText embeddings and cosine similarity to measure sentence similarity between questions and objects. The second system is based on selecting the objects whose bounding box has the largest area.

5.2.1.1 Sentence similarity

Our first reranking approach consists of analyzing the similarity between sentences. We propose to use FastText embeddings [10, 11], originally developed by the Facebook Artificial Intelligence Research (FAIR) team as a text classification and representation tool, together with cosine similarity [12, 13] to measure and analyze sentence similarity. Using this technique, we can obtain information about the semantic and contextual similarities between the question and each of the objects and attributes detected in the image.

First, we have removed punctuation marks, such as question marks, and we utilize FastText's embeddings to convert each word of a sentence into a 300-dimensional vector representation. The idea of these embeddings is to capture the meaning and context of individual words. Then we calculate the average of the word embeddings of all the words in the sentence, obtaining a vector representation that represents the overall semantic information of the sentence.

Once we have the embeddings of the question and each of the objects and attributes related to the image referenced by the question, we calculate the cosine similarity of each of the pairs, i.e., question with first object and attributes, question with second object and attributes and so on. The cosine similarity measures the cosine of the angle between two vectors, returning a value in the range of -1 and 1.

$$S_C(A, B) = cosine(\theta) = \frac{A \cdot B}{||A|| \cdot ||B||} \tag{5.1}$$

As we can see in Equation 5.1, to calculate the cosine similarity, we first calculate the dot product of the average embeddings of the sentences and divide it by the product of their magnitudes. If we get a value of 1, it means that the sentences are identical, and -1 that are dissimilar. What we obtain as output is a list of objects with each object and attributes and each cosine similarity value.

Objects $\rightarrow$ [(object1, similarity value), (object2, similarity value), ...]

5.2.1.2 Objects bounding box area

Our second reranking proposal is based on the hypothesis that, in most of the cases, the questions will refer to the most prominent objects of the image. That is, objects with a bounding box that occupy a considerable area, and not small objects that even humans would have difficulty recognizing. Therefore, using the rectangle coordinates provided by the VinVL [41] object detection model (Figure 2.13), we have calculated the area of each object (Equation 5.2).

$$Area = (x_2 - x_1) \cdot (y_2 - y_1) \tag{5.2}$$

What we obtain as output is a list of objects with each object and attributes and each bounding box area value.

Objects $\rightarrow$ [(object1, area), (object2, area), ...]

5.2.2 Influence of the number of objects

Based on the idea of providing quality information to the model, we also want to analyze the influence of the amount of information, i.e., the number of objects provided. Thus, we analyze how the number of objects affects the performance of the model in generating answers. We will analyze the input length and, consequently, we will define different values of number of objects (k), being aware of the memory limitation and maximizing the amount of information in as many instances as possible.

5.2.3 Experiments and results

For this purpose, we experiment with both systems and with a different number of objects to analyze how the number of objects influences the results. As in the multilabel leverage experiments, the **training batch size is 24 and the maximum source and target length are 192.** First, we analyze the inputs to define a maximum number of objects. For that, we have analyzed the length of question plus caption (Figure 5.3).

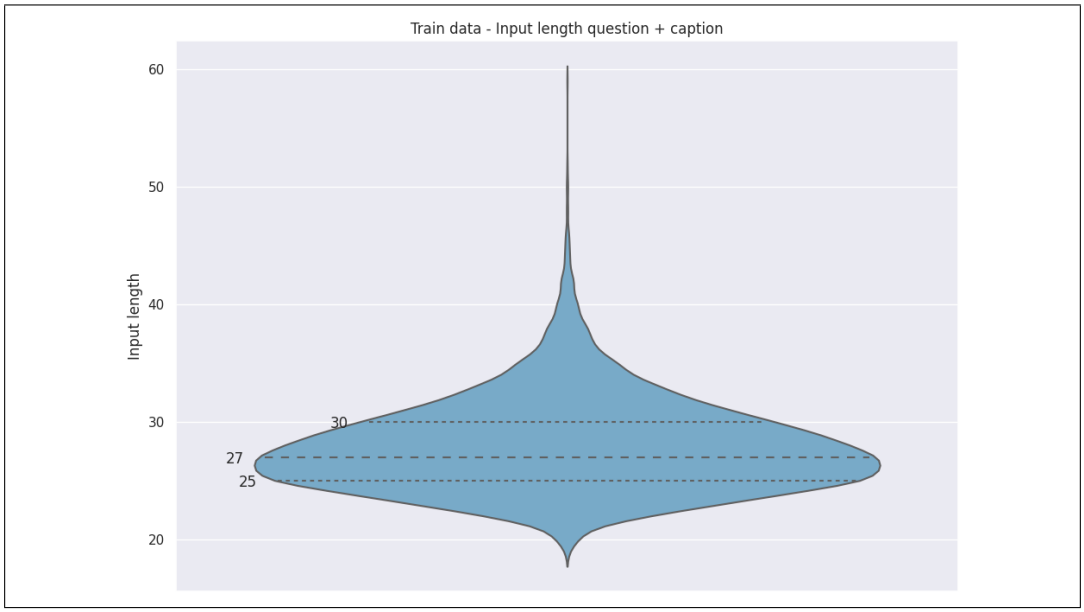

Figure 5.3: Violin plot of question plus caption input length. Shows distribution and quartiles.

We can see in Figure 5.3 that 75% of instances has a length under 30, and if we calculate 90% we obtain that the length is less than 33 tokens, so we can have around 160 tokens to add objects and attributes. We have calculated the average number of words for each object and its attributes, obtaining that the training instances have **4.23 tokens per object** (object included). We **round that number to 5 tokens per object** to have a margin. This tells us that if we **select 30 objects** we will need about 150 tokens on average, **plus 30 tokens of question and caption, getting 180 tokens on average**. This tells us that on average we will fall below our truncation of 192 tokens. Thus, we will analyze the distribution of the number of tokens once 30 objects have been selected (Figure 5.4).

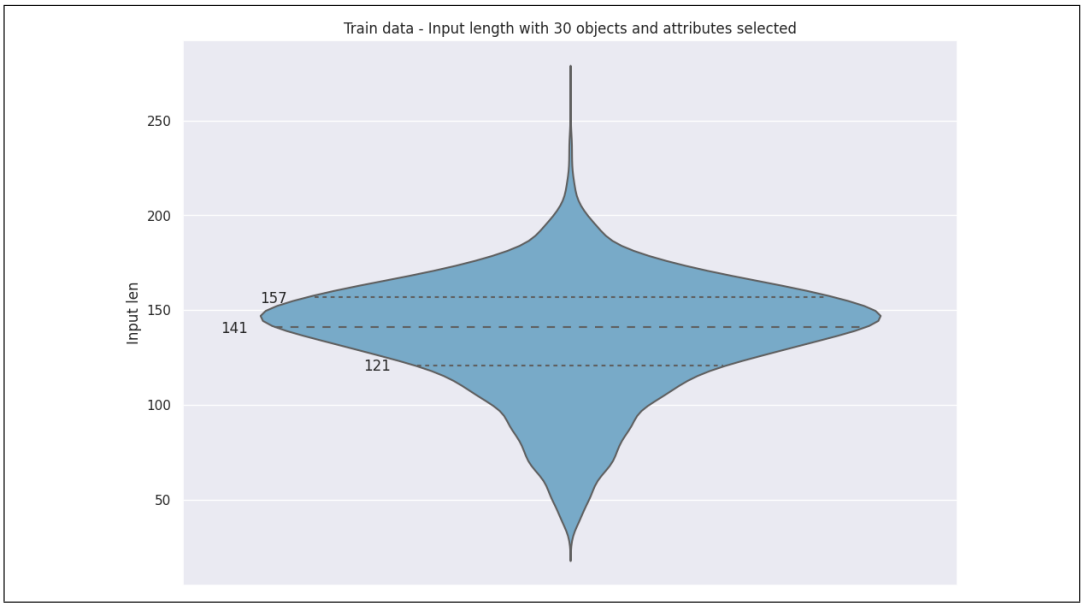

Figure 5.4: Violin plot showing the input length after selecting 30 objects with their attributes. Shows the distribution and quartiles.

We can see in Figure 5.4 that the length of the objects and attributes of the 75% of the instances is less than 157 tokens, and if we calculate the percentage of instances that have a length equal to or less than 192 tokens, we obtain that it is 98%. Therefore, we accept that the maximum number of 30 objects is adequate. On this basis, we experimented with different numbers of objects (k), from 5 to 30, and defined the previous model presented in Section 5.1.1 of objects and attributes as a baseline. We implemented 4 models with 2 different reranking systems, three models based on the **sentence similarity**, objects, objects and attributes and the mix of both, and a fourth model based on the **bounding box area** of the object.

For the sentence similarity models, we compute the cosine similarity for both objects and objects and attributes, using each of them respectively with its model. Then we prepare the mixed model, choosing for each instance (question) between objects or objects and attributes (Table 5.4), depending on which of the two has a higher cosine similarity value. This model tries to find a model that takes advantage of the best of the two previous ones. For the bounding box area model, we calculate the area for each of the objects based on the coordinates provided by the object detector.

Table 5.4: Selected percentage of objects and objects and attributes for different number of items (k) for Train and Test splits in the mixed model.

	Train		**Test**	
k	**Obj**	**Obj-attr**	**Obj**	**Obj-attr**
5	2,586	97.414	3,686	96.314
10	1,165	98.835	1,684	98.316
20	0,455	99.545	0,495	99.505
30	0,233	99.767	0,198	99.802

As can be seen in Table 5.4, the information containing objects and attributes is selected

for most of the instances, above 96% in the worst case. It is interesting to note that the more objects we select, the more the percentage of selection in favor of objects and attributes increases, converging to 100%.

As before, we performed 3 runs with each model and calculated the mean VQA score and the standard deviation. All the results can be found in Appendix D.

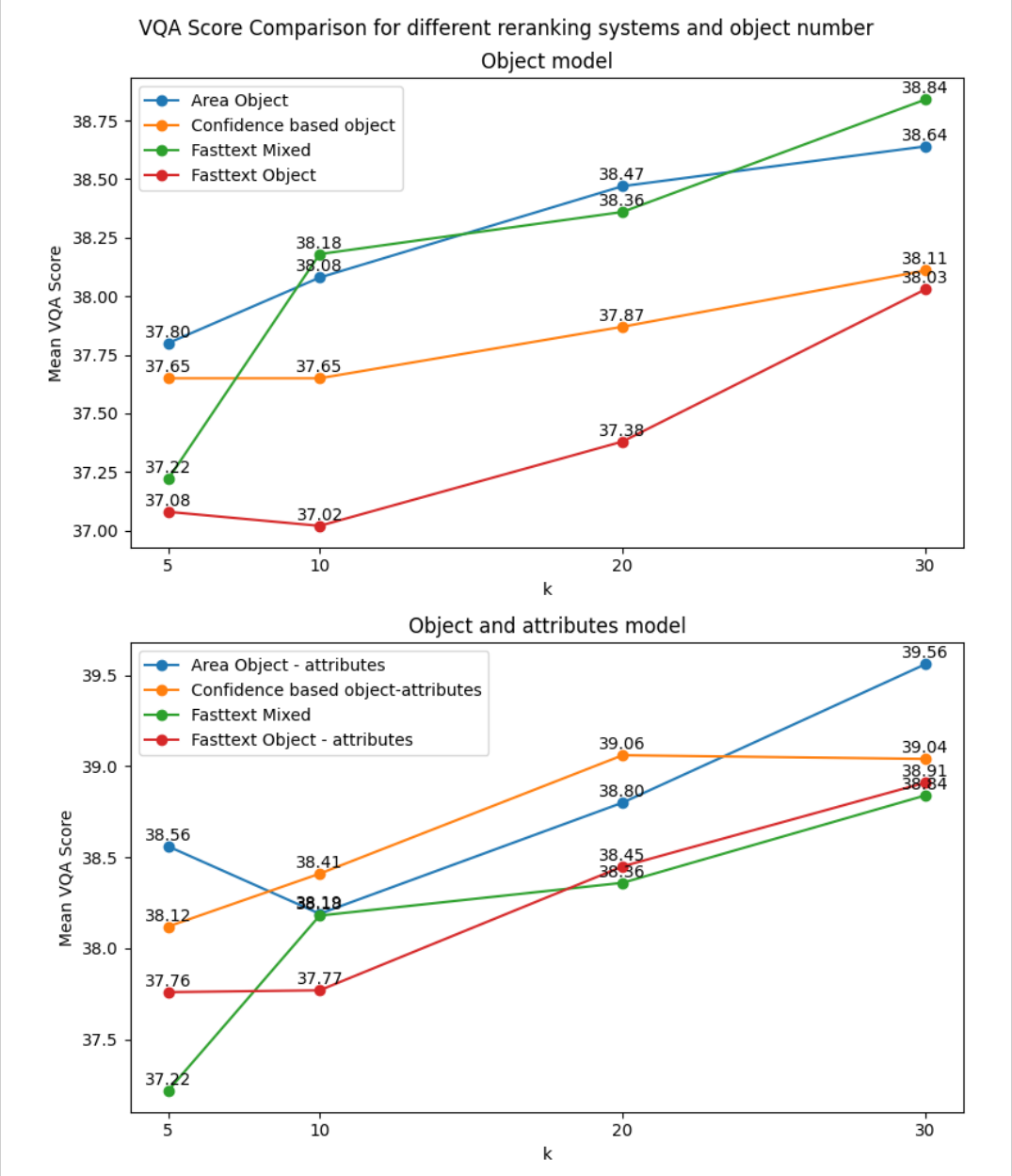

Figure 5.5: Plots showing, objects (top) and objects and attributes (below), the mean VQA score obtained in three runs with each of the proposed reranking systems and the model used as reference ordered according to the confidence value of the objects. The x-axis indicates the number of selected objects (k). The model that obtains the best results is the model of objects and attributes with reranking of bounding box area and $k = 30$, with a mean VQA score of 39.56. On the other hand, the worst is the model of objects with FastText reranking and $k = 10$, obtaining a mean VQA score of 37.02. The results of each run can be found in Appendix D.

Several conclusions can be drawn from the results (Figure 5.5):

- **Increasing the number of objects improves the results**, except in the case of the area object and attributes model with $k = 10$. For the future it would be interesting to analyze what happens to the objects when we select 10 since in several cases it does not improve.
- Related to the previous one, the model that integrates **objects and attributes obtains the best results**. A mean VQA score of 39.56.
- **The FastText reranking system does not perform well compared to the other proposals with a low number of objects**; as the number of objects increases, the different approaches converge to the same result.
- We can conclude that of the two reranking systems, sentence similarity and objects bounding box area, **the selection of objects with the largest area is the one that works best**. In the case of the objects model the FastText mixed model obtains a better result, but it is misleading, since this model uses mostly object and attribute information (Table 5.4).

Taking into account the results obtained (Figure 5.5), we present our final model (Figure 5.6): Based on the CBM enhanced by the multilabel leverage technique and extended by the object detection and reranking systems.

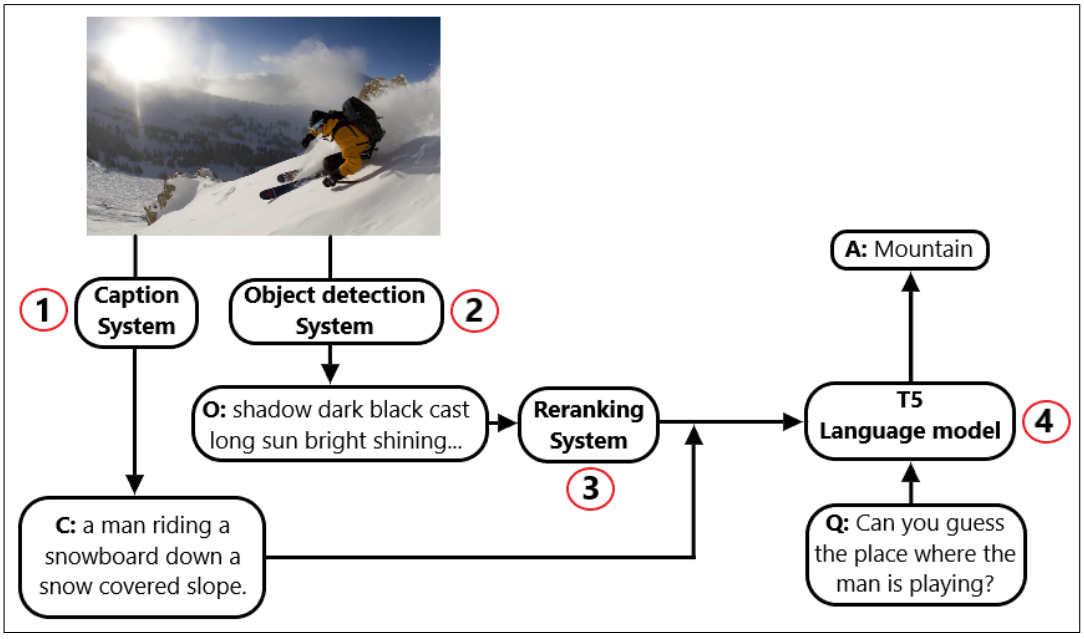

Figure 5.6: Where **Q** indicates question, **C** caption, **O** object and attributes and **A** answer.

As can be seen (Figure 5.6), our model is formed with 4 systems:

1. **Caption system:** Oscar model that performs the description of the input image. This system is replicated from the CBM model.
2. **Object detection system:** VinVL object detector that extracts object-attributes from the input image. It provides objects and attributes to the reranking system.
3. **Reranking system:** System that reranks the objects and attributes provided by the object detection system from highest to lowest according to the area of the bounding box and selects the largest 30 objects.
4. **Language model:** T5 model that receives as input the question, caption, and the objects and attributes, and generates the answer. Having as a training novelty the **multilabel leverage technique** that improves the learning.

Here we present the improvement obtained with respect to the original CBM of our final model (Table 5.5), this improvement has been obtained by combining our three proposals.

Table 5.5: Mean VQA score and standard deviation results of the original CBM$_{T5Base}$ and our final model.

Model	Mean VQA score
CBM$_{T5Base}$	**36.1** ± 0.5
Our final model	**39.56** ± 0.15

As can be seen (Table 5.5), **our model improves the mean score of the initial proposal by 3.46 points with a lower standard deviation**, indicating greater consistency and accuracy in each of the runs. We can conclude that the model not only performs better on average, but also produces more consistent results across runs or evaluations. These factors contribute to increased confidence in the efficacy and reliability of our contributions.

CHAPTER 6

Conclusions and future work

In this document we have exposed the proposals, implementations and results obtained in the OK-VQA [5] task. For that, we selected CBM [9] as our base model, a text-only model, using the image caption and the implicit knowledge integrated in the parameters of a T5 [59] generative language model. CBM has been tested for different sizes of the language model, in our case using the results obtained with the $T5_{Base}$ as a reference for our research. We divided the research into two parts: leveraging multilabel annotations and comparing image verbalization approaches.

In the first part, Leveraging Multilabel Annotations, we conclude that our contribution is a simple method that slightly improves the results in multilabel problems with optimal and suboptimal solutions. The introduction of the concept of balance between exploration and exploitation improves the learning of the model, maintaining the number of fully correct answers and increasing the number of partially correct answers.

In the second part, we analyze and compare different image verbalization approaches. We conclude that completing the image caption information through objects and attributes improves the information provided to the model. As with the concept of balance between exploration and exploitation, something similar occurs when passing information to the model. It is important to strike a balance between generalized and detailed information: since the information in the image referred to in the question will not be very obvious, but neither will it be a detail that even humans find difficult to recognize. Related to this, it is observed that adding a template with prefixes in the case of limited memory does not improve the understanding of the model. This is because adding prefixes excludes a lot of information when truncating, obtaining better results with plain information of objects and attributes.

Once it is observed that adding objects and attributes improves the base model, a reranking system is defined to improve the quality of these objects. In addition, the influence of the number of objects on the results is analyzed. By means of the reranking system, it is observed that the more information is better, both by adding attributes to the objects and by increasing the number of selected objects (k). For this task, the reranking system based on the bounding box area is the one that obtains the best results, confirming the hypothesis that the questions are about objects that are clearly represented in the image.

As future work, we could also divide it into two parts. On the one hand, it would be interesting to implement the proposed multilabel leverage technique on other tasks that meet the casuistry presented above. Performing experiments on other tasks could consolidate the technique as a simple way to improve the previously obtained results without major changes in the implementation or the need of any modification in the model architecture. Another experiment that could be carried out is to modify the balance between exploration and exploitation. Instead of using a frequency distribution based on the percentage of occurrences of the answers, we could define fixed percentages. For example, 90% of the time the optimal answers are chosen and the other 10% of the time the suboptimal ones. It would be interesting to analyze the training of the model with the proposed technique to see how it affects learning, since it is possible that it modifies the behavior of the model. This could help us to optimize the training and improve the results. In addition, it would be intriguing to see how this technique behaves with larger models, since it improves learning, and the improvement could be more considerable in models with a larger number of parameters.

On the other hand, it would be interesting to test our first two proposals in combination using GPUs with larger memory. This way we could pass all the information to the model and squeeze it to the maximum. With our experiments we have observed that more information is better, but we could investigate if there exists a number of objects for which more information does not bring extra performance, interfering with the learning of the model.

In addition, other model enhancements related to the OK-VQA task have emerged that could be interesting for the future. For example, generating conditional image captions to the question [50, 51], thus generating a specific description of the image that directly relates to the question. This would reduce the problem of balancing generalized and detailed information, since it will be specific to each question. Another option could be to generate multiple answers and implement a model that selects the best one based on the question [50].

Appendix

Appendix A Multilabel leveraging results

In this appendix, we present the results of each run and the mean VQA score and standard deviation of the original CBM and our first implementations. Our first implementation (Answers with score = 1) is our replication of the CBM, which will be the basis of our subsequent implementations, and the second is the implementation with our proposed multilabel leverage technique (All answers). As can be seen, our first implementation obtains very similar results to the original CBM, and in the second one, the implementation with our proposed technique improves by 0.92 the score.

A.1 VQA score of each run and the mean VQA score and standard deviation of the CBM$_{T5Base}$ and our multilabel leverage technique implementation (All answers).

Model	Labels	Run	VQA score	Mean VQA score
CBM$_{T5Base}$	Answers with score = 1	1	-	**36.1** ± 0.5
		2	-	
		3	-	
Our CBM$_{T5Base}$	Answers with score = 1	1	35.88	**36.09** ± 0.426
		2	35.81	
		3	36.58	
Our CBM$_{T5Base}$	All answers	1	37.15	**37.01** ± 0.329
		2	36.63	
		3	37.24	

Appendix B Caption plus object and attributes results

In this appendix, we present the results of each run and the mean VQA score and standard deviation of the previous implementations and the experiments developed with objects and attributes. We compare the implementations with captions and with captions plus objects and attributes, in addition to comparing each model with the original label selection (Answers with score = 1) and with our proposed multilabel leverage technique (All answers). We observe how the addition of objects and of objects and attributes improves the results, especially in the case of objects and attributes. Moreover, in all cases the multilabel leverage technique improves the original selection, confirming that it is a robust technique and improves model learning. It is curious how in the case of the object model (mean improvement 1.57), our technique improves considerably more than in the object and attribute model (mean improvement 1.12). This may be something interesting to investigate in the future. The average improvement of the combination of our two proposals over our CBM replica for objects and attributes is 3 points, and for objects 2 points.

B.1 VQA score of each run and the mean VQA score and standard deviation of the previous implementations and the implementations with caption plus object and attributos.

Model	Labels	Run	VQA score	Mean VQA score
CBM$_{T5Base}$	Answers with score = 1	1	-	**36.1** ± 0.5
		2	-	
		3	-	
Our CBM$_{T5Base}$	Answers with score = 1	1	35.88	**36.09** ± 0.426
		2	35.81	
		3	36.58	
Our CBM$_{T5Base}$	All answers	1	37.15	**37.01** ± 0.329
		2	36.63	
		3	37.24	
Caption + obj-attr	Answers with score = 1	1	37.84	**37.97** ± 0.112
		2	38.01	
		3	38.05	
Caption + obj-attr	All answers	1	39.38	**39.09** ± 0.264
		2	38.86	
		3	39.04	
Caption + obj	Answers with score = 1	1	36.98	**36.52** ± 0.482
		2	36.02	
		3	36.57	
Caption + obj	All answers	1	38.05	**38.09** ± 0.278
		2	38.39	
		3	37.84	

Appendix C Template vs. plain object and attributes

In this appendix, we present the results of each run and the mean VQA score and standard deviation of the experiments with caption plus objects and attributes, comparing the template and plain inputs presented in Section 5.1. The first implementation is the one with the original label selection, the second is with a template and our multilabel leverage technique, and the last is with a plain input and our multilabel leverage technique. We can see how the implementation with a plain entry slightly improves the results (mean improvement of 0.14), but nothing significant.

C.1 VQA score of each run and the mean VQA score and standard deviation of the caption plus objects and attributes implementations, comparing template and plain inputs.

Model	Labels	Run	VQA score	Mean VQA score
Caption + obj-attr template	Answers with score = 1	1	37.84	**37.97** ± 0.112
		2	38.01	
		3	38.05	
Caption + obj-attr template	All answers	1	39.38	**39.09** ± 0.264
		2	38.86	
		3	39.04	
Caption + obj-attr plain	All answers	1	38.93	**39.23** ± 0.354
		2	39.14	
		3	39.62	

Appendix D Reranking results

In this appendix, we present the results of the confidence based implementations used as reference and the reranking systems, dividing the results into tables by system. In each of the tables, we present the results of each run and the mean VQA score and standard deviation for a different number of selected objects (k).

D.1 Confidence based implementation of **objects model without** reranking.

k	Run	VQA score	Mean VQA score
5	1	37.27	**37.65** ± 0.33
	2	37.87	
	3	37.81	
10	1	37.90	**37.65** ± 0.291
	2	37.72	
	3	37.33	
20	1	37.79	**37.87** ± 0.072
	2	37.92	
	3	37.91	
30	1	37.77	**38.11** ± 0.356
	2	38.48	
	3	38.09	

D.2 Confidence based implementation of **objects and attributes model without** reranking.

k	Run	VQA score	Mean VQA score
5	1	38.18	**38.12** ± 0.081
	2	38.16	
	3	38.03	
10	1	38.32	**38.41** ± 0.123
	2	38.36	
	3	38.55	
20	1	38.98	**39.06** ± 0.08
	2	39.08	
	3	39.14	
30	1	39.37	**39.04** ± 0.39
	2	38.61	
	3	39.14	

D.3 Implementation of **objects model with FastText cosine similarity** reranking.

k	Run	VQA score	Mean VQA score
5	1	37.20	**37.08** ± 0.199
	2	36.85	
	3	37.19	
10	1	37.33	**37.02** ± 0.3
	2	36.73	
	3	37.01	
20	1	37.72	**37.38** ± 0.356
	2	37.40	
	3	37.01	
30	1	38.22	**38.03** ± 0.413
	2	37.56	
	3	38.32	

D.4 Implementation of **objects and attributes model with FastText cosine similarity** reranking.

k	Run	VQA score	Mean VQA score
5	1	38.06	**37.76** ± 0.271
	2	37.54	
	3	37.67	
10	1	37.30	**37.77** ± 0.596
	2	37.57	
	3	38.44	
20	1	38.45	**38.45** ± 0.006
	2	38.45	
	3	38.46	
30	1	39.39	**38.91** ± 0.419
	2	38.64	
	3	38.69	

D.5 Implementation of **mixed model with FastText cosine similarity** reranking.

k	Run	VQA score	Mean VQA score
5	1	37.46	**37.22** ± 0.24
	2	37.21	
	3	36.98	
10	1	38.08	**38.18** ± 0.203
	2	38.41	
	3	38.04	
20	1	38.72	**38.36** ± 0.317
	2	38.14	
	3	38.21	
30	1	39.14	**38.84** ± 0.263
	2	38.73	
	3	38.65	

D.6 Implementation of **objects model with descending area** reranking.

k	Run	VQA score	Mean VQA score
5	1	37.53	**37.80** ± 0.265
	2	38.06	
	3	37.80	
10	1	38.54	**38.08** ± 0.401
	2	37.85	
	3	37.84	
20	1	38.65	**38.47** ± 0.605
	2	38.97	
	3	37.80	
30	1	38.96	**38.64** ± 0.325
	2	38.65	
	3	38.31	

D.7 Implementation of **objects and attributes model with descending area** reranking.

k	Run	VQA score	Mean VQA score
5	1	38.0	**38.56** ± 0.5
	2	38.96	
	3	38.72	
10	1	37.81	**38.19** ± 0.340
	2	38.28	
	3	38.47	
20	1	39.48	**38.80** ± 0.761
	2	37.98	
	3	38.95	
30	1	39.48	**39.56** ± 0.15
	2	39.46	
	3	39.73	

References

[1] Tadas Baltrušaitis, Chaitanya Ahuja, and Louis-Philippe Morency. Multimodal machine learning: A survey and taxonomy. 2017.

[2] Peng Xu, Xiatian Zhu, and David A Clifton. Multimodal learning with transformers: A survey. 2022.

[3] Khaled Bayoudh, Raja Knani, · Fayçal Hamdaoui, · Abdellatif Mtibaa, B Khaled Bayoudh, Fayçal Hamdaoui, and Abdellatif Mtibaa. A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. *The Visual Computer 2021 38:8*, 38:2939–2970, 6 2021.

[4] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C. Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering, 2015.

[5] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. pages 3195–3204, 2019.

[6] Ok-vqa leaderboard.

[7] Ashish Vaswani, Google Brain, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. 2017.

[8] Feilong Chen, Duzhen Zhang, Minglun Han, Xiuyi Chen, Jing Shi, Shuang Xu, and Bo Xu. Vlp: A survey on vision-language pre-training. 2022.

[9] Ander Salaberria, Gorka Azkune, Oier Lopez De Lacalle, Aitor Soroa, and Eneko Agirre. Image captioning for effective use of language models in knowledge-based visual question answering. 2022.

[10] Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. Bag of tricks for efficient text classification. *15th Conference of the European Chapter of the Association for Computational Linguistics, EACL 2017 - Proceedings of Conference*, 2:427–431, 7 2016.

[11] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146, 7 2016.

[12] Teruaki Kitasuka, Masayoshi Aritsugi, and Faisal Rahutomo. Semantic cosine similarity. 2012.

[13] Pinky Sitikhu, Kritish Pahi, Pujan Thapa, and Subarna Shakya. A comparison of semantic similarity methods for maximum human interpretability. 2019.

[14] Bin Zhao, Maoguo Gong, and Xuelong Li. Hierarchical multimodal transformer to summarize videos. 2021.

[15] Xiao Wang, Guangyao Chen, Guangwu Qian, Pengcheng Gao, Xiao-Yong Wei, Yaowei Wang, Yonghong Tian, and Wen Gao. Large-scale multi-modal pre-trained models: A comprehensive survey. 2023.

[16] Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. pages 2556–2565.

[17] William M. Fisher Jonathan G. Fiscus David S. Pallett Nancy L. Dahlgren Victor Zue John S. Garofolo, Lori F. Lamel. Timit acoustic-phonetic continuous speech corpus - linguistic data consortium.

[18] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. 2012.

[19] Andrej Karpathy and Li Fei-Fei. Deep visual-semantic alignments for generating image descriptions. 2014.

[20] Zakir Hossain. A comprehensive survey of deep learning for image captioning. 2018.

[21] Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. Generative adversarial text to image synthesis. *33rd International Conference on Machine Learning, ICML 2016*, 3:1681–1690, 5 2016.

[22] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. 2021.

[23] Minguk Kang, Jun-Yan Zhu, Richard Zhang, Jaesik Park, Eli Shechtman, Sylvain Paris, and Taesung Park. Scaling up gans for text-to-image synthesis. 2023.

[24] Royi Rassin, Shauli Ravfogel, and Yoav Goldberg. Dalle-2 is seeing double: Flaws in word-to-concept mapping in text2image models. 2022.

[25] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. 2016.

[26] Vqa challenge 2021 - leaderboard - evalai.

[27] Xi Chen, Xiao Wang, Soravit Changpinyo, A J Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, Alexander Kolesnikov, Joan Puigcerver, Nan Ding, Keran Rong, Hassan Akbari, Gaurav Mishra, Linting Xue, Ashish Thapliyal, James Bradbury, Weicheng Kuo, Mojtaba Seyedhosseini Chao, Jia Burcu, Karagol Ayan, Carlos Riquelme, Andreas Steiner, Anelia Angelova, Xiaohua Zhai, Neil Houlsby, and Radu Soricut. Pali: A jointly-scaled multilingual language-image model. 9 2022.

[28] Wenhui Wang, Hangbo Bao, Li Dong, Johan Bjorck, Zhiliang Peng, Qiang Liu, Kriti Aggarwal, Owais Khan Mohammed, Saksham Singhal, Subhojit Som, and Furu Wei. Image as a foreign language: Beit pretraining for all vision and vision-language tasks. 8 2022.

[29] Chenliang Li, Haiyang Xu, Junfeng Tian, Wei Wang, Ming Yan, Bin Bi, Jiabo Ye, Hehong Chen, Guohai Xu, Zheng Cao, Ji Zhang, Songfang Huang, Fei Huang, Jingren Zhou, and Luo Si. mplug: Effective and efficient vision-language learning by cross-modal skip-connections. 5 2022.

[30] Jiahui Yu, Zirui Wang, Vijay Vasudevan, Legg Yeung, Mojtaba Seyedhosseini, Yonghui Wu, and Google Research. Coca: Contrastive captioners are image-text foundation models. 5 2022.

[31] Jianfeng Wang, Zhengyuan Yang, Xiaowei Hu, Linjie Li, Kevin Lin, Zhe Gan, Zicheng Liu, Ce Liu, and Lijuan Wang. Git: A generative image-to-text transformer for vision and language. 5 2022.

[32] Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering visualreasoning.net. 2019.

[33] Amanpreet Singh, Vivek Natarajan, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. 2019.

[34] Peng Wang, Qi Wu, Chunhua Shen, Anthony Dick, and Anton Van Den Hengel. Fvqa: Fact-based visual question answering. 2016.

[35] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. 2014.

[36] Emanuele Bugliarello, Ryan Cotterell, Naoaki Okazaki, and Desmond Elliott. Multimodal pretraining unmasked: A meta-analysis and a unified framework of vision-and-language berts. 2020.

[37] Jacob Devlin, Ming Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*, 1:4171–4186, 10 2018.

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

[39] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39:1137–1149, 6 2015.

[40] Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, Yejin Choi, and Jianfeng Gao. Oscar: Object-semantics aligned pre-training for vision-language tasks. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12375 LNCS:121–137, 4 2020.

[41] Pengchuan Zhang, Xiujun Li, Xiaowei Hu, Jianwei Yang, Lei Zhang, Lijuan Wang, Yejin Choi, and Jianfeng Gao. Vinvl: Revisiting visual representations in vision-language models. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 5575–5584, 1 2021.

[42] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. 2019.

[43] Hao Tan. Lxmert: Learning cross-modality encoder representations from transformers. 2019.

[44] Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graphs. 2021.

[45] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. 2015.

[46] Tsung Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 8693 LNCS:740–755, 5 2014.

[47] Kenneth Marino, Xinlei Chen, Devi Parikh, Abhinav Gupta, and Marcus Rohrbach. Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based vqa. 2020.

[48] Robyn Speer, Joshua Chin, and Catherine Havasi. Conceptnet 5.5: An open multilingual graph of general knowledge. 2016.

[49] Stephan Bloehdorn, Marko Grobeinik, Peter Mika, and Thanh Tran Duc. Organization workshop organizers. 2008.

[50] Zhenwei Shao, Zhou Yu, Meng Wang, and Jun Yu. Prompting large language models with answer heuristics for knowledge-based visual question answering. 2023.

[51] Yushi Hu, Hang Hua, Zhengyuan Yang, Weijia Shi, Noah A Smith, and Jiebo Luo. Promptcap: Prompt-guided task-aware image captioning. 2022.

[52] Yuanze Lin, Yujia Xie, Dongdong Chen, Yichong Xu, Chenguang Zhu, and Lu Yuan. Revive: Regional visual representation matters in knowledge-based visual question answering. 2022.

[53] Liangke Gui, Borui Wang, Qiuyuan Huang, Alex Hauptmann, Yonatan Bisk, and Jianfeng Gao. Kat: A knowledge augmented transformer for vision-and-language.

[54] Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. 2021.

[55] Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. Deep modular co-attention networks for visual question answering. pages 6281–6290, 2019.

[56] Yangyang Guo, Liqiang Nie, Yongkang Wong, Yibing Liu, Zhiyong Cheng, and Mohan Kankanhalli. A unified end-to-end retriever-reader framework for knowledge-based vqa. pages 2061–2069, 6 2022.

[57] Jialin Wu, Jiasen Lu, Ashish Sabharwal, and Roozbeh Mottaghi. Multi-modal answer validation for knowledge-based vqa. 2022.

[58] François Gardères, Maryam Ziaeefard, Baptiste Abeloos, Thales Montreal, and Canada Freddy Lecue. Findings of the association for computational linguistics conceptbert: Concept-aware representation for visual question answering. 2020.

[59] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 10 2019.

[60] T5 - documentation.

[61] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. 2017.

[62] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury Google, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf Xamla, Edward Yang, Zach Devito, Martin Raison Nabla, Alykhan Tejani, Sasank Chilamkurthy, Qure Ai, Benoit Steiner, Lu Fang Facebook, Junjie Bai Facebook, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. 2019.

[63] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick Von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M Rush. Transformers: State-of-the-art natural language processing. 2020.