Visual Question Answering

Sofiya Semenova

I. Introduction

The problem of Visual Question Answering (VQA) offers a difficult challenge to the fields of computer vision (CV) and natural language processing (NLP) – it is jointly concerned with the typically CV problem of *semantic image understanding*, and the typically NLP problem of *semantic text understanding*. In [29], Geman *et al.* describe a VQA-based "visual turing test" for computer vision systems, a measure by which researchers can prove that computer vision systems adequately "see", "read", and reason about vision and language.

The task itself is simply put - given an input image and a text-based query about the image (*"What is next to the apple?"* or *"Is there a chair in the picture"*), the system must produce a natural-language answer to the question (*"A person"* or *"Yes"*). A successful VQA system must understand an image semantically, understand natural language input, and construct a response given its visual, textual, and logical understanding of the input image and text. As the problem involves *image* understanding as well as *language* understanding, the VQA problem has a unique inherent challenge; VQA methods must combine the techniques used in the natural language processing communities and those used in the computer vision communities, two communities that have evolved separately and in parallel. Bridging both the lessons learned in CV and NLP research towards one common goal is no trivial task.

A. Problem Variations

The VQA problem specification does not formally require any particular question or answer type – a dataset or method sufficiently addresses VQA so long as it semantically answers questions about images. Thus, there is a large variety of and overlap between question/answer formats among datasets and methods. Several datasets, like the *VQA* dataset [1] and the *Visual 7W* dataset [20], provide questions in both multiple choice and open-ended formats. Several models, such as [8], provide several variations of their system to solve more than one question/answer type.

Binary. Given an input image and a text-based query, these VQA systems must return a binary, "yes/no" response. [29] proposes this kind of VQA problem as a "visual turing test" for the computer vision and AI communities. [5] proposes both a binary, abstract dataset and system for answering questions about the dataset.

Multiple Choice. Given an input image and a text-based query, these VQA systems must choose the correct answer out of a set of answers. [20] and [8] are two systems that have the option of answering multiple-choice questions.

"Fill in the blank". Given an input image and an incomplete sentence with blanks, these VQA systems must fill in the blanks in the sentnece. Unlike other VQA sub-problems, this one does not provide *one query*, it provides essentially a series of declarative questions (where each blank in the sentence is one question). *Visual MadLibs* creates a system for this kind of problem [6]. While this is a less popular VQA problem than the others, most VQA models could be adapted to fit this style of question.

Open-Ended. The open-ended, free-form VQA task, first proposed formally by Antol *et al.* in [1], is not a disjoint task from the previous tasks (binary, multiple choice, fill in the blank, visual dialog), but primarily refers to the open-endedness of the questions themselves – making the "natural" in the natural language inputs more prominent. Rather than reasoning about structured, templated questions and finite, discrete objects and relationships, the goal of open-ended VQA is to sufficiently answer *any* open-ended question. To this end, there has been as significant amount of work

towards crafting open-ended VQA datasets [1] [20] [23] [24] [46].

B. Related Research

Several problems in the broader semantic scene understanding area overlap with VQA: automatic image captioning [2], physical scene understanding [47] [48] [49], and scene classification [50]. Automatic image captioning covers the same intersection of natural language processing and computer vision as VQA does, but is a significantly easier task as the “problem" is known in advance – that is, the computer knows it must generate a human-readible description of an unknown input image, whereas in VQA the computer does not know the query before runtime. Physical scene understanding (reasoning about the physics of an image or video) is similar to VQA in that both require the computer to reason about the objects in a scene and their relationship to each other – however, in physical scene understanding (unlike in VQA), the problem is largely physics-based and doesn't require much, if any, natural language processing. Lastly, scene classification and location recognition often use similar techniques to some in VQA – in particular, scene graphs are very popular for scene classification, and several VQA approaches use scene graphs as well. However, the VQA problem requires more complex scene reasoning and natural language processing than scene classification generally does.

On the NLP side, the problem of text-based question answering is similar to the question-answering portion of VQA, but does not require any visual reasoning. In the robotics community, the problem of conversational robotic agents overlaps with that of VQA in that there is frequently a text-based and image-based reasoning component [52] [51]. However, research in that area has largely been domain-constrained in either the acceptable language inputs or the application.

II. Datasets

There are several VQA datasets, which range in input type (e.g., real-world vs. synthetic images) and response type (e.g., constrained vs. free-form). Earlier datasets (*DAQUAR* [21], *COCO-QA* [19]) constrict their question types to one of several categories – *DAQUAR* contains generated template-based questions and human-submitted questions about basic colors, numbers, or object categories; *COCO-QA* contains generated questions about object identification, number, color, and location. Later datasets (*Visual7W* [20], *VQA* [1]) broaden the types of questions that can be asked to be more free-form and natural. In parallel, some specialized datasets have also been released: *Tally-QA* [25] focuses on counting questions only, and *OK-VQA* [27] focuses on questions that require external knowledge to answer correctly. A few sample questions from these datasets can be seen in Figure 1.

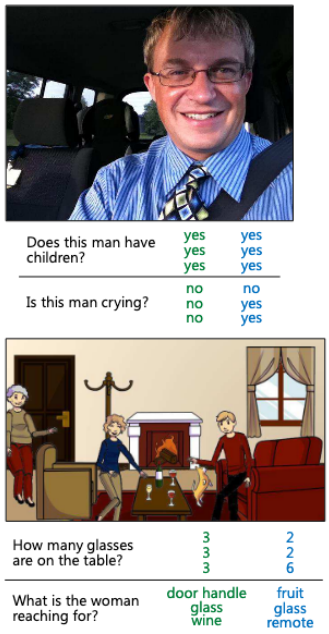
While the complexity of dataset questions grows, several authors point out an inherent bias found in VQA datasets that artificially inflate model accuracy and mislead authors into believing their models “understand" language and vision when they're just exploiting biases [22]. To this end, there has been some work in modifying existing datasets [23] [24] and creating new datasets [46] to combat the bias problem.

A. Towards Question Complexity

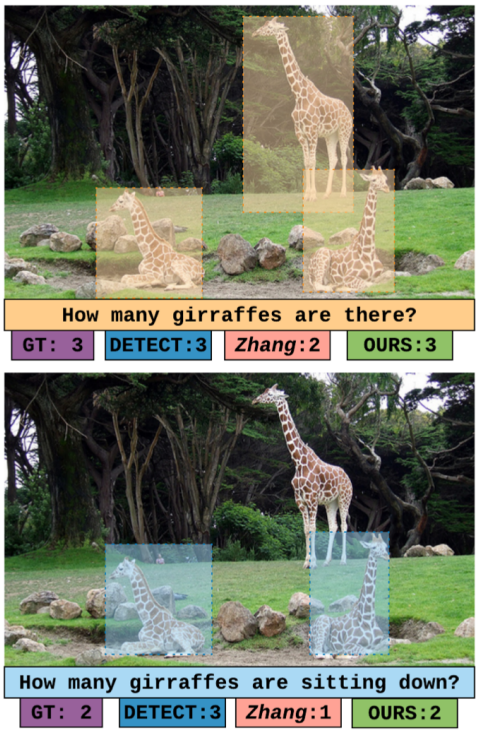
DAQUAR. The *DAQUAR* dataset [21] contains 6794 training and 5674 test question-answer pairs, where the image portion of each question-answer pair is sourced from the NYU-Depth V2 dataset, which contains roughly 1500 RGBD images of indoor scenes and corresponding annotated semantic segmantations with 894 object classes. *DAQUAR* contains two configurations – one where 894 object classes are used, as in the original NYU dataset, and one where 37 object classes are used, where the 37 object classes are created by the *DAQUAR* authors applying an image segmentation algorithm to the NYU dataset. The dataset contains two types of question-answer pairs – *synthetic* question-answer pairs, which are automatically generated from a set of question templates, and *human* question-answer pairs, which are generated by 5 participants instructed to provide questions and answers about either basic colors, numbers, or object categories.

COCO-QA. The *COCO-QA* dataset [19] contains 123,287 images and over 10,000 questions, using

(a) Visual7W sample questions, where the top two images show a multiple-choice question and the bottom two show open-ended questions



(b) Example VQA questions, where the top image is an example of a real photo and the bottom image is an example of a synthetic scene



(c) TallyQA sample questions, where the top question is an example of a "simple" counting question and the bottom is an example of a "complex" counting question

(d) OK-VQA sample questions, with corresponding knowledge categories

Fig. 1: Sample Questions from Earlier Datasets

images from the MS COCO dataset [44]. The answers are only one word, and rather than using human-created question-answer pairs, the authors generated question-answer pairs from image captions, which has the benefit of creating more organic, human-like questions and answers. Generated questions fall into one of four categories: object (identifying what an object is), number (identifying a quantity for an object), color, and location.

Visual7W. The *Visual7W* dataset [20] contains, in addition to an image and question-answer pairs, *object groundings* for objects in the image – a link between an object mentioned in a question-answer pair and a bounding box in an image. Including object grounding alleviates the problem of *coreference ambiguity* [45], wherein a question refers to one of multiple possible objects in the image. *Visual7W* questions fall into one of 7 "W"s: *what, where, when, who, why, how* and *which*. The dataset contains 327,939 question-answer pairs and 561,459 object groundings on 47,300 COCO images [25].

VQA. Larger and more open-ended than previous datasets, the *VQA* dataset [1] contains two parts – a real-world dataset with 207,721 images from the MS COCO dataset [44], and an abstract scene dataset with 50,000 scenes. Compared to previous datasets such as [29], *COCO-QA* [19], and *DAQAUR* [21], the *VQA* dataset does not draw questions from a fixed set of colors, object categories, object-to-object relationships, question templates, etc., but rather contains free-form questions and answers provided by humans. Each input image or scene has three corresponding input questions, and each question is answered by 10 human participants, leading to a total of over 750,000 questions and over 10 million answers.

Tally-QA. Compared to the previous datasets, the *Tally-QA* dataset [25] focuses only on counting-based questions. Questions are split into *Test-Simple* and *Test-Complex*, where the former focuses on simple counting questions (e.g., *"How many giraffes are there"*) and the latter focuses on complex counting questions (e.g., *"How many dogs are eating"*). The dataset contains 19,500 complex questions for 17,545 images, and provides more than twice as many complex counting questions than previous datasets.

OK-VQA. The *OK-VQA* [27] dataset contains only image/question pairs that require external knowledge to be able to answer. Compared to previous datasets, these questions are more complex to answer because there is not sufficient enough information to answer them in the query image. To answer these questions, the VQA system needs to learn what information is necessary to answer the question, determine how to retrieve that information from an outside source, and incorporate that information into a suitable answer. *OK-VQA* contains 14,000 questions that cover a variety of categories such as science and technology, sports and recreation, and geography, history, language, and culture.

B. Towards Eliminating Dataset Bias

The VQA problem itself is particularly susceptible to dataset bias. Several papers draw a comparison between VQA performance and a story about *Clever Hans*, a horse who seemed to understand and be capable of answering math problems, but who actually was just reacting to the subtle cues of his trainer and human observers [41] [42] [43]. Likewise, many VQA systems don't truly *understand* images and language, but are rather just exploiting inherent biases in their training datasets [42] [23] [22] [24] [46]. For example, "tennis" is the correct answer for 41% of *"What sport is.."* questions and "2" is the correct answer for 39% of counting questions [23]. Further, there is a strong visual priming bias in human-generated VQA datasets, where participants tend to only ask questions about objects in images that contain those objects (e.g., Answering "Yes" to *"Do you see..."* achieves an 87% accuracy) [23].

In [22], Jabri *et al.* show that a VQA system designed to exploit dataset biases perform comparatively with, if not better than, state-of-the-art approaches for multiple-choice tasks on the *Visual7W* dataset. Their system takes an (image, question, answer) triplet as inputs, represents the image as a feature vector computed by running the image through a pre-computed CNN, and represents the question and answers as average word2vec embeddings. Then, all the representations are concatenated and used to train a classification model

that predicts if the triplet is correct. They compare their model against state-of-the-art VQA models and find that, if given all three inputs (Answer, Question, and Image), their model achieves state-of-the-art performance in classifying the triplet as correct or not. If given just (Answer, Question) tuples, their model performs competitively by exploiting the most frequent Q-A pairs.

In [23], Goyal *et al.* propose *VQA v2*. This dataset builds on top of the original *VQA* dataset [1] to create a balanced dataset with reduced bias due to language priors. For each (I, Q, A) (Image, Quesion, Answer) triplet in their dataset, they ask a human participant to identify an image I' that is similar to I but where the answer to Q for that image is not A. By creating a uniform answer distribution across the dataset and providing counter-examples for each question, their dataset "forces" the computer to consider the visual content.

In [24], Agrawal *et al.* propose *VQA-CP v1* and *VQA-CP v2* (Visual Question Answering under Changing Priors), two splits of the *VQA v1* and *VQA v2* datasets, respectively. These split datasets are created by re-organizing the original training and evaluation datasets so that the distribution of answers for each question type is different in the test and train sets. Their intuition behind this change is that a good VQA model should be able to correctly answer a question from the test dataset even though the training dataset might have had a different language prior (e.g., "White" is the most common color in the train dataset, but "Black" is the correct answer in the test dataset). The authors also show that performance for several existing VQA models drop significantly when compared against the CP datasets, compared to the original VQA datasets.

In [46], Hudson *et al.* identify that the dataset proposed in [23] fails to address open questions, leading to an unbalanced dataset, and that the datasets proposed in [24] unfairly punish models for learning properties of the training data – they argue that "making an educated guess" about the data is in fact a desirable strategy for models to exhibit. Their dataset, *GQA*, focuses on real-world reasoning and compositional question answering. Each image is annotated with a dense scene graph that represents the objects, attributes, and relationships between objects that exist in the image. Each question is a program which lists the reasoning steps needed to arrive at the right answer. They argue that the enabling such strict structure is advantageous because they can have very tight control over their answer distribution and allow better assessment of VQA model performance.

Their dataset construction method is four-fold:

1) Normalize and augment the Visual Genome Scene Graph annotations, which are annotated with free-form natural language
2) Generate questions by iterating over 524 question "patterns" and the scene graph
3) Generate a functional representation of each question (e.g., *"What color is the apple on the white table?"* is equivalent to *"select: table, filter: white, relate(subject,on): apple, query:color"*)
4) Sample and balance from generated questions to create a balanced answer distribution across the dataset

III. Methods

VQA methods can generally be grouped into four categories: *language-image* (or joint) *embeddings*, *attention mechanisms*, *compositional models*, and *knowledge-based models*. Because VQA research has exploded in popularity in the past few years and a lot of work was completed in parallel, a lot of models use overlapping methods – the four groupings are not definitive.

A. Language-Image Embedding

Approaches in the language-image embedding category combine CNNs and RNNs to learn a joint embedding for both the input text and the input image in a shared feature space. Typically, a CNN pre-trained on an object recognition dataset is used to recognize objects, whereas an RNN is pre-trained on natural language inputs. By keeping the image representation and language representation in the same feature space, the VQA system can then create one classifier.

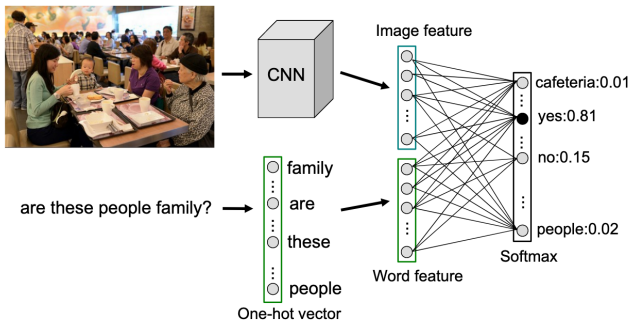


Fig. 2: iBOWIMG architecture

iBOWIMG. In [7], Zhou *et al.* propose a VQA baseline using a joint embedding approach. Their approach uses a bag of words for text features, and features from GoogLeNet as visual features. An overview of their network can be seen in Figure 2.

First, they convert the input words into one-hot vectors – a binary vector of length *dictionary_size* that has only one non-zero vector to identify the word. These vectors are piped into a word embedding layer to turn into word features. Word features are concatenated with image features and sent to a softmax layer to predict the answer class. Essentially, their model functions as a multi-class regression model.

The model is trained and evaluated on the COCO dataset, on open-ended and multiple-choice questions. The authors show that, despite the simplicity of their model (it takes "10 lines of code in Torch" to create), their performance is comparable to previously proposed, more complex models.

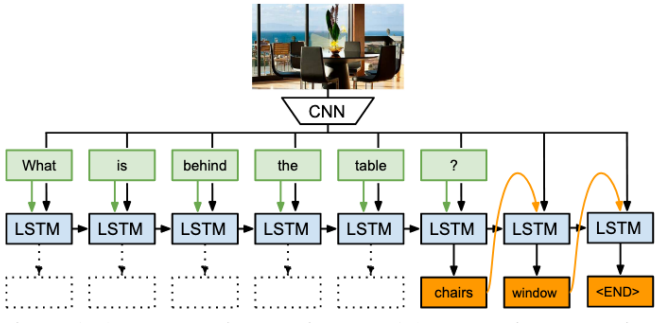


Fig. 3: Neural-Image-QA architecture

Neural-Image-QA. In [10], Malinowski *et al.* were inspired by the performance of CNNs for image classification tasks and that of LSTMs for sequence prediction tasks to create a CNN and LSTM-based joint embedding model for VQA. Their method analyses an image with a CNN and feeds a question and visual representation of the image into an LSTM network. Both the CNN and LSTM are trained together. An overview of their method can be seen in Figure 3.

In their problem scenario, each question can have a multiple-word answer, which informs the design of their system. First, they formulate the general VQA problem as predicting an answer $\boldsymbol{a}$ given an image $\boldsymbol{x}$ and a question $\boldsymbol{q}$:

$$a = \max_{a \in A} p(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{q}; \boldsymbol{\theta})$$

where $\boldsymbol{\theta}$ is a vector of all parameters to learn and $\boldsymbol{A}$ is the set of all possible answers. However, because their problem can have answers of more than one word, they modify the problem like:

$$a_t = \max_{a \in V} p(\boldsymbol{a}|\boldsymbol{x}, \boldsymbol{q}, \boldsymbol{A_{t-1}}; \boldsymbol{\theta})$$

where $\boldsymbol{a_t}$ are words from a vocabulary $\boldsymbol{V}$ and $\boldsymbol{A_{t-1}}$ is the set of previous predicted words. Thus, the re-formulates the problem as one of predicting an answer sequence from a vocabulary.

Their method first represents both question and answer vectors with a one-hot vector encoding. Then, question and image features are put through an "encoder" LSTM to get a feature vector of fixed-size, which is passed to a "decoder" LSTM that predicts the answer, one word per pass.

The approach used in this paper is frequently considered the true baseline result for VQA methods.

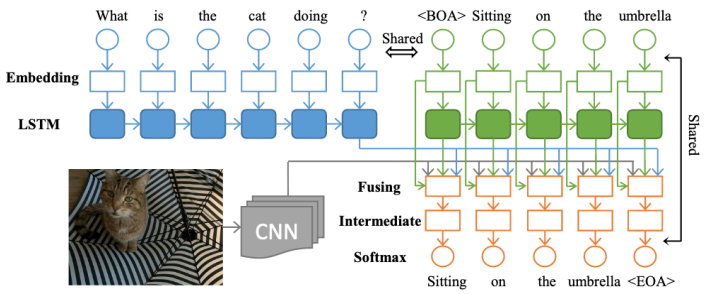


Fig. 4: MQA architecture

mQA. In [9], Gao *et al.* use an LSTM to extract a question representation, a CNN to extract visual representation, an LSTM for storing linguistic context in an answer, and a fusing component

to combine the first three into an answer. They also create a multilingual (Chinese and English) dataset, *FM-IQA*, to train and evaluate their model. An overview of their neural network can be seen in Figure 4.

Compared to *Neural-Image-QA*, *mQA*'s "encoder" and "decoder" LSTMs do not share weights, and learn distinct parameters. Their only shared component is the word embedding. The authors chose this network structure primarily because there are different textual properties (such as grammar) between the question and the answer.

Their network is structured as follows:

1) The first part of their network encodes a natural language sequence into a dense vector representation. First, they map a one-hot vector of each input question word into a dense semantic space by feeding the words into a 512-dimensional word embedding layer. Then, this dense semantic representation is piped into an "encoding" LSTM layer with 400 memory cells.
2) To capture the image features, *MQA* uses a CNN pre-trained on GoogLeNet.
3) A second word embedding layer and "encoder" LSTM which is structurally similar to the first LSTM encodes the information about the current word and previous words in the *answer* into dense semantic representations. The activation of the memory cells for the words in the generated answer and the word embeddings are then fed into the fusing component.
4) A fusing layer takes the outputs of the previous 3 layers to predict the next word in the answer.
5) Finally, an intermediate layer that maps the dense multimodal representation from the fusing layer back into a dense word representation. This is piped through a softmax layer to predict the probability distribution of the next word in the answer.

They jointly train the two LSTM layers and the fusing layer on their dataset, which contains 150k images, 200k Chinese and English question-answer pairs, and with no constrains on question types. In addition to the natural language dictionary, the authors also add the words "<BOA>" and "<EOA>" into the dictionary, corresponding to the *beginning* of the sentence and *end* of the sentence. To generate an answer, they start with a "<BOA>" and use the model to calculate the probability distribution of the next word.

B. Attention Mechanism

The language-image embedding approach generally contains a vision, question understanding, and answer generation portion. Generally, the vision part uses a CNN to extract visual features, while the question uses either a BoW or RNN in congruence with a word embedder to encode question semantics into a dense vector representation. Lastly, the answer generation portion generates an answer using the visual and textual representations of the input. While this approach works, it does not take advantage of the relationship between the image and the question/answer.

The intuition behind the attention mechanism approach is that using an attention mechanism in the VQA model should exploit the relationships found between the image and question/answer pairs by using a question to guide what parts of the image to "focus" on. The creators of these models posit that knowing where to look to answer a question correctly ignores the noise of irrelevant information in the image.

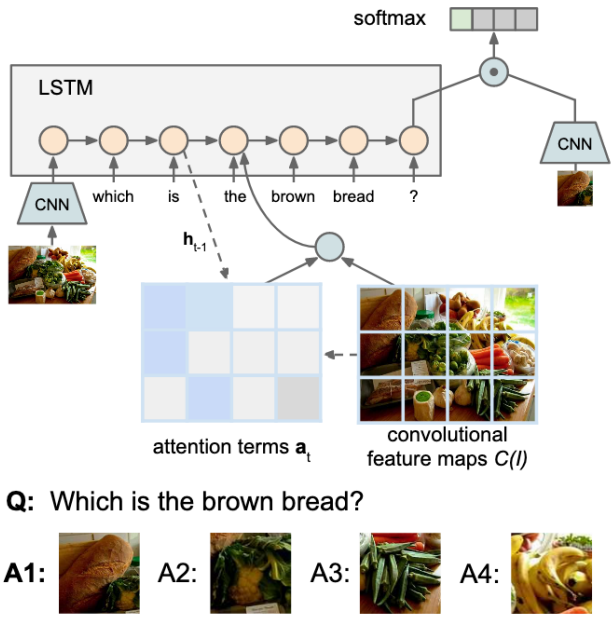


Fig. 5: Visual7W architecture for "pointing" task

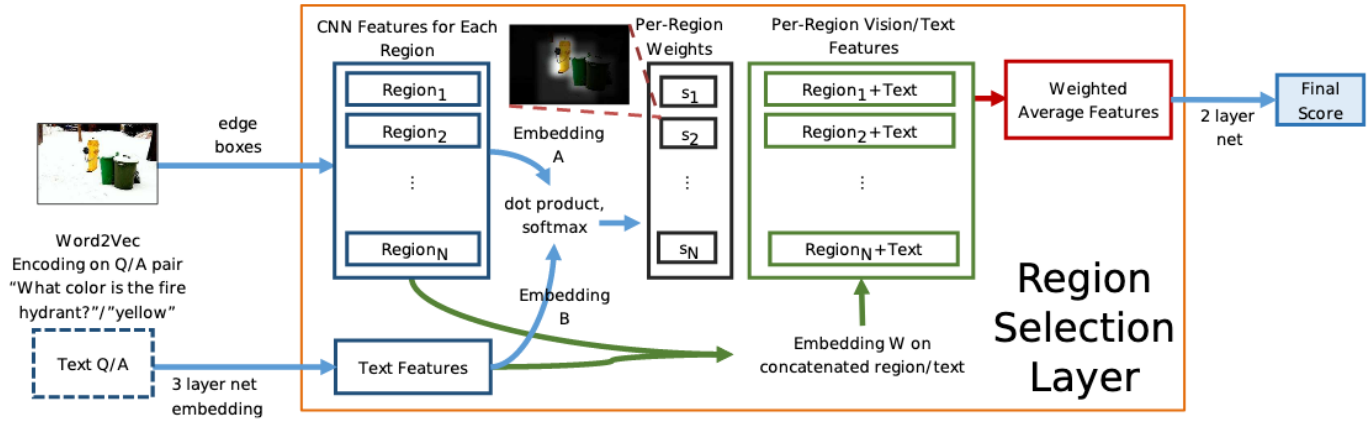


Fig. 6: "Where To Look" architecture

Visual7W. In addition to the *Visual 7W* dataset, [20] provides an attention-based model for grounded VQA, by encorporating attention into the typical LSTM-based model. Their model follows the familiar two-stage process: an "encoding" step and "decoding" step. A visual overview of their architecture can be seen in Figure 5.

At the encoding stage, the model reads the image and the question tokens word by word and memorizes the image and question into a hidden state vector (in an LSTM). For each word, it computes an *attention term* based on the previous hidden state and the features, which indicates what region to focus on. In the decoding step, their model selects an answer from the multiple choices based on its memory (the softmax layer).

They encorporate attention into the standard LSTM model with the r_t term below:

$$
\begin{aligned}
\boldsymbol{i_t} &= \sigma(W_{vi}v_t + W_{hi}\boldsymbol{h_{t-1}} + W_{ri}\boldsymbol{r_t} + b_i) \\
\boldsymbol{f_t} &= \sigma(W_{vf}v_t + W_{hf}\boldsymbol{h_{t-1}} + W_{rf}\boldsymbol{r_t} + b_f) \\
\boldsymbol{o_t} &= \sigma(W_{vo}v_t + W_{ho}\boldsymbol{h_{t-1}} + W_{ro}\boldsymbol{r_t} + b_o) \\
\boldsymbol{g_t} &= \sigma(W_{vg}v_t + W_{hg}\boldsymbol{h_{t-1}} + W_{rg}\boldsymbol{r_t} + b_g) \\
\boldsymbol{c_t} &= \boldsymbol{f_t} \odot \boldsymbol{c_{t-1}} + \boldsymbol{i_t} \odot \boldsymbol{g_t} \\
\boldsymbol{h_t} &= \boldsymbol{o_t} \odot \phi(\boldsymbol{c_t})
\end{aligned}
$$

where $\sigma()\dot{}$ is the sigmoid function, $\phi()\dot{}$ is the tanh function, and $\odot$ is the element-wise multiplication operator. The r_t term is the attention mechanism, which is just a weighted average of features that is created using the LSTM's previous hidden state and question text features.

Where To Look. In [12], Shih *et al.* propose an image-region selection mechanism that identifies which region of an image is relevant to answering a question, and a learning framework with a margin-based loss to solve multiple choice VQA problems. An overview of their system can be seen in Figure 6. The input to their model is a question, a potential answer, and image features from a set of automatically selected candidate attention regions. Then, their model processes the input as follows:

1) Parse question and answer encoded using word2vec and a three-layer network
2) Encode the visual features for each region using the top 2 layers of a CNN trained on ImageNet
3) Embed language and vision features are and compare with a dot product
4) Softmax to create a per-region relevance weighting
5) Using the weights, create a concatenated vision and language feature vector
6) Send weighted average features to a 2 layer network for a final confidence score

The "Region selection layer" is every layer of their network but the last step. This layer selectively combines incoming test features with image features from relevant regions of the image. The model determines relevance by projecting the image and text features into a shared space and computing an inner product between each question-answer pair and all available regions. The inner product forces the model to compute region/question relevance as if it were computing

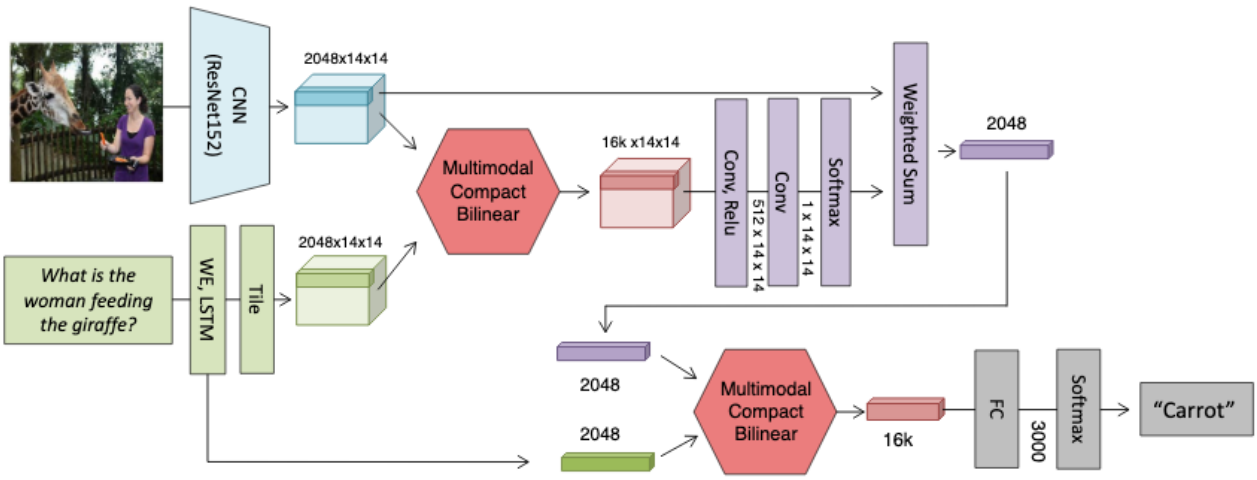


Fig. 7: Multimodal Compact Bilinear Pooling with Attention architecture

similarity.

Lastly, another key feature of this system is the *margin-based* loss function. The authors explicitly define the goal for their VQA system to maximize a margin between correct and incorrect choices in a structured way. Their reasoning behind this is their evaluation dataset (*VQA*) has 10 answers for each question, which may not all be the same; for example, *"What color is the scarf?"* could have "blue" or "purple" answers. To take this discrepency into account, they create a *margin-based loss* – they change their loss function to require that the score of the correct answer is at least some margin above the score from the incorrect choices. For example, if 6/10 of the annotators answer an answer a and 4 annotators answer b, then the answer a should outscore b by at least 40%.

Multimodal Compact Bilinear Pooling with Attention. The previous approaches to combining two-vector representations for image and text rely on concatenating vectors or applying an element-wise sum or product. This generates a joint representation, but the authors in [8] claim that it is not expressive enough to represent the complex relationships between the input image and text. Instead, the authors propose a multimodal compact bilinear pooling (*MCB*) based model to get a joint representation. An overview of their method can be seen in Figure 7.

Bilinear pooling computes the outer product between two vectors (vs. an elementwise product), which creates a multiplicative interaction between all elements of both image and text vector. Their approach is based on research on bilinear pooling models that show that it is a beneficial method for fine-grained classification on vision-only tasks. Despite its performance, a problem with bilinear pooling is its high dimensionality, so not many models use it. To combat this, the authors incorporate previous research about compressing bilinear pooling into their model.

The model extracts representations for the image and the question, pools the vectors using MCB, and gets the answer as a multi-class classification problem with 3k classes. Images features are extracted by a 152-layer residual network that is pretrained on the ImageNet dataset.

To incorporate spatial information, they incorporate *soft attention* into their MCB pooling. For each spatial location in the visual representation, their method use MCB pooling to merge the visual feature with the language representation. After pooling, they use two convolutional layers to get the attention weight for each location.

In this paper, the authors also propose three VQA architectures – one for VQA with attention (discussed here), another for VQA with attention and answer encoding (a solution for VQA problems with multiple choice questions), and another for VQA with visual grounding (where the goal is to predict a bounding box that corresponds to the question).

ABC-CNN. In [14], Chen *et al.* propose an attention-based configurable CNN (ABC-CNN) to

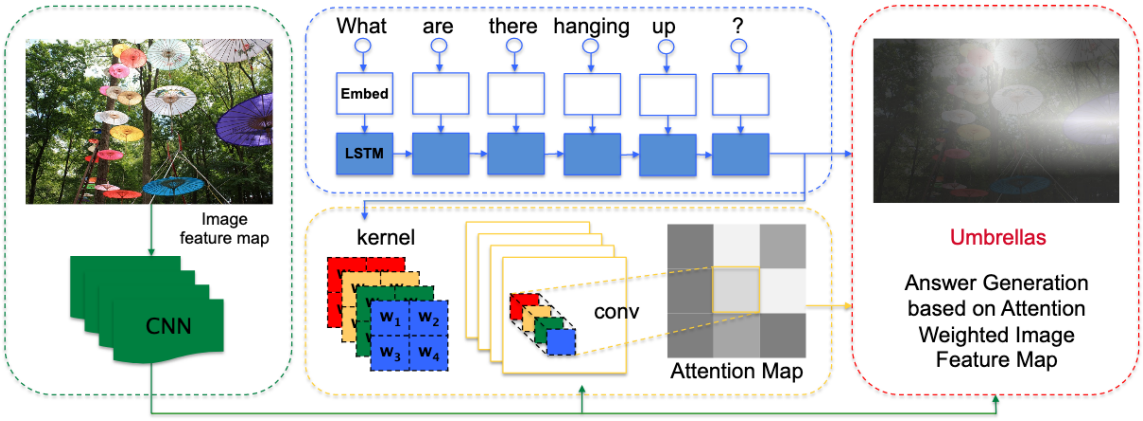


Fig. 8: ABC-CNN architecture

aid in *question-guided attention*, where the input question determines a region of interest within the image. Their model explicitly contains "a vision part", "a question understanding part", "an answer generation part", and "an attention extraction part". An overview of their system can be seen in Figure 8.

For the vision part, *ABC-CNN* uses the deep CNN VGG-19, to extract visual features into a spatial feature map, I. This model is pre-trained on a 1000-class ImageNet classification challenge dataset and a fully convolutional segmentation neural network pre-trained on the PASCAL 2007 segmentation dataset.

For the question understanding part, they use an LSTM to learn the dense question embedding vector, s, to encode semantic information.

The attention extraction part encodes question-based attention information as a *question-guided attention map* (QAM), which is generated by searching for visual features in the spatial feature map that correspond to the input query's semantics. This is done using a CNN and convolving the visual feature map with a *configurable* kernel. The kernel is generated by transforming the question embeddings from semantic space into visual space, which contain the visual information determined by the intent of the question. More intuitively, the question *"What color is the umbrella?"* should generate a kernel that only selects for "umbrella" visual features.

For the answer generation, they generate single-word answers with a multi-class classifier with softmax activation. The multi-class classifier is based on the original feature map, the question embedding, and the attention-weighted feature map. The weighted feature map is generated by an element-wise product between the feature map, I, and the attention map, m. While their method focuses on single-word answer generation, they can extend their method to generate full sentences by using an RNN decoder.

Compositional Memory. In [15], Jiang *et al.* propose a *Compositional Memory* for an end-to-end training framework to fuse local, region-based visual features with linguistic information, to combat the loss of fine-grained local features in traditional, holistic VQA methods. They base their approach off the idea of "visual facts" – local visual evidence that can be used to answer a question – and seek to unearth them through modeling the interactions between vision and language. As an overview, their method traverses the words in a question one-by-one and explicitly combines the linguistic information from the question with visual information from the image to create *episodes* for each *time step* (a "place" in processing the input question). Episodes are then fed into an answer generation component, along with contextual visual and linguistic information.

An overview of their method can be seen in Figure 9.

Critical to their method is adding a *compositional memory* block to the general LSTM-based approach for VQA, and a second LSTM which parses the input question and provides input to

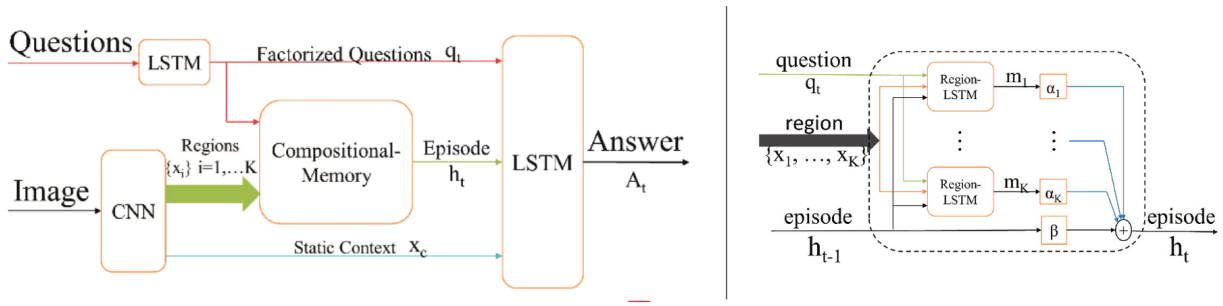


Fig. 9: Compositional Memory architecture, where the left hand side of the figure is an architectural diagram of the entire network and the right hand side is a close-up look at the internals of the *Compositional Memory* block

both the compositional memory block and the question-answering LSTM.

The compositional memory block consists of several Region-LSTM and α gates. The Region-LSTMs share the same parameters and their results are combined to generate an *episode* – a block of fused visual and linguistic features at a time-step t for a visual region x_i. The Region-LSTMs process input image region contents in parallel, and dynamically generate visual-linguistic information for each region. The authors' implemention of Region-LSTMs is mostly similar to an implementation of an LSTM; the primary difference is that all LSTM parameters (W_{q*}, W_{h*}, W_{x*}, and b_*) are shared across different regions, although each Region-LSTM retains its own gate and cell state.

The α gates in the compositional memory block are conditioned on a previous episode h at time $t-1$, a vector of features for a region X_k, and the current language feature q_t. These gates are used to generate a weighted combination of region messages into one episode, where each episode is then dynamically pooled into image-level information. At each time t and region k, an α gate generates the value α_k^t. To summarize region-level information into an image-level feature, they use a modified pooling mechanism.

SMem-VQA. In [13], Xu *et al.* argue that a major problem with existing VQA models is that they have no explicit understanding of an object's position, while the VQA task necessarily requires reasoning about multiple objects and their relative position to each other. Their approach, *SMem-VQA* (Spatial Memory VQA) is based on memory networks, which combine text embeddings with an attention mechanism and multi-step inference. Normal memory networks store information about *text*, whereas for VQA they should be storing information about an *image*, so the authors adapt the memory network to store the CNN outputs from different receptive fields. Further, their model can update the proposed answer based on several attention stops, which they call "hops" ... this is equivalent to a person looking at multiple areas of an image and gathering evidence from all of them before forming the answer. Thus, *SMem-VQA* is a multi-hop memory network with attention. An overview of their method can be seen in Figure 10.

The input to *SMem-VQA* is a variable-length question and an image. First, they represent each word as a one-hot vector and embed it into a real-valued word vector, V, which is dependent on the maximum number of words in the question and the dimensionality of the embedding space. The image is converted into image features using a CNN, and features are mapped to a grid of spatial locations, S.

Then, the image feature vectors at each spatial location and the word vectors are embedded into a common semantic space, using the "evidence" visual embedding, W_A. The authors use W_A to project each visual feature vector, s_i in S, so that its combination with the embedded question words generates the attention weight, W_{att} at each location i in the visual grid. They do this through a process they call "word-guided attention", which predicts the attention given a question word that has the maximum correlation between the embedded visual features at each location.

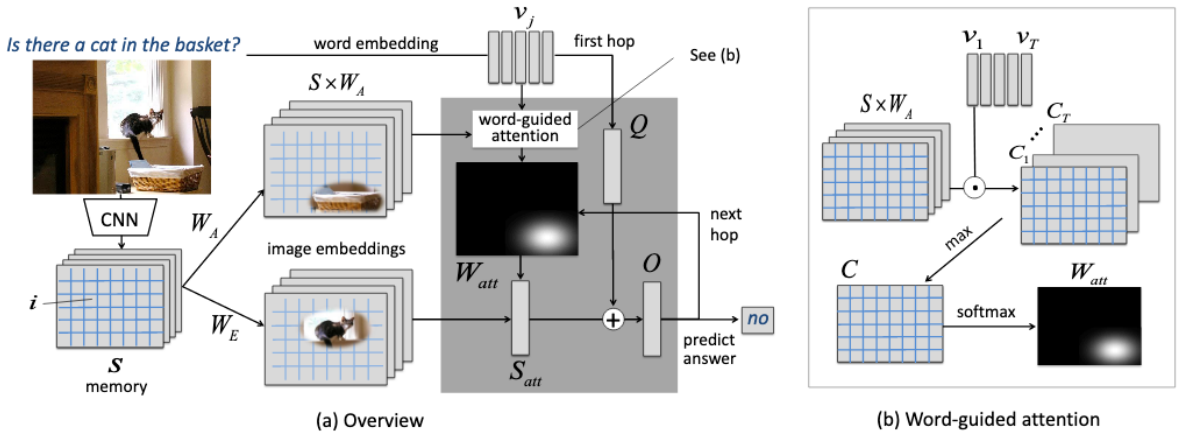
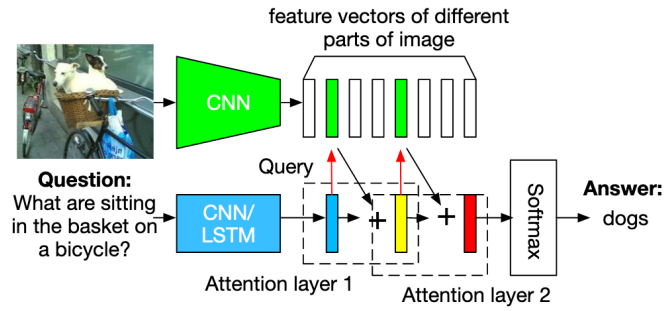


Fig. 10: SMem-VQA architecture, where the left side of the figure shows an overview of the whole method, and the right side shows the word-guided attention method

They also introduce a second embedding, W_E – the "evidence" embedding. W_E detects the presence of semantic objects and is used to compute a weighted sum over the visual features by multiplying W_E by the W_{att} attention weights vector gleaned from the "word-guided attention" process. The resultant vector is the evidence vector, S_{att}.

Lastly, the "evidence" vector S_{att} is combined with the question representation, Q, to predict an answer for the given question-image pair.

This describes one "hop" in the "multi-hop" method ... additional hops can repeat the process to gather more evidence.



(a) Stacked Attention Network for Image QA

Fig. 11: Stacked Attention Networks architecture

Stacked Attention Networks. In [17], Yang *et al.* notice that answering a question often requires multi-step reasoning. They propose *stacked attention networks* (SANs) that allow for multi-step reasoning. A stacked attention comprises of three components: an image model, a question model, and a stacked attention model. The image model uses a CNN to extract high level features, the question model uses a CNN or LSTM to extract a semantic vector of the question, and the stacked attention model locates the regions in the image that are relavant for the given question. An overview of their method can be see in Figure 11.

Their method can be summarized as follows:

1) Use a CNN to extract high level features from the image
2) Use either a CNN or LSTM (the authors test both implementations) to extract a semantic vector of the question
3) Generate an attention distribution over the regions of the image by feeding the image and question feature vectors through a single layer neural network and a softmax function
4) Based on the attention distribution, calculate the weighted sum of the image vectors across all image regions
5) Combine the weighted sum with the question vector to form a refined query vector. This vector encodes *only* the visual and linguistic information that is relevant to the answer

6) Iterate on the above query vector by using multiple attention layers, where each layer extracts more fine-grained visual attention information

C. Compositional

So far, all approaches to VQA have used what Andreas *et al.* call "monolithic" methods – regardless of the network structure and the components therein (e.g., RNNs, CNNs, LSTMs, BoWs, word embeddings), previous approaches are necessarily hindered in their non-dynamic structure. Approaches in the compositional area focus on *modular*, *changing* architectures that adapt themselves to fit the particular VQA problem. The intuition behind this approach is that essentially, different problems (question-image pairs) will require different machine thought processes.

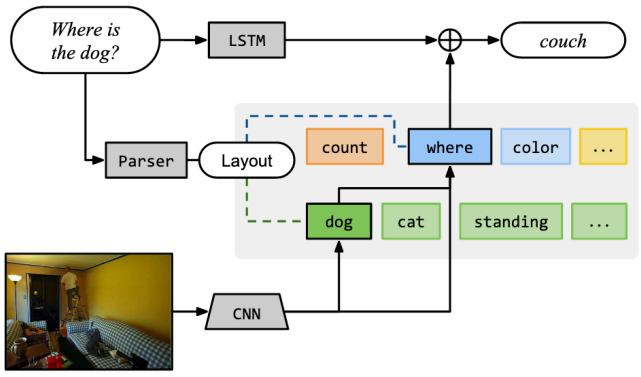


Fig. 12: Neural Module Networks architecture, with an example modular network in the light grey box

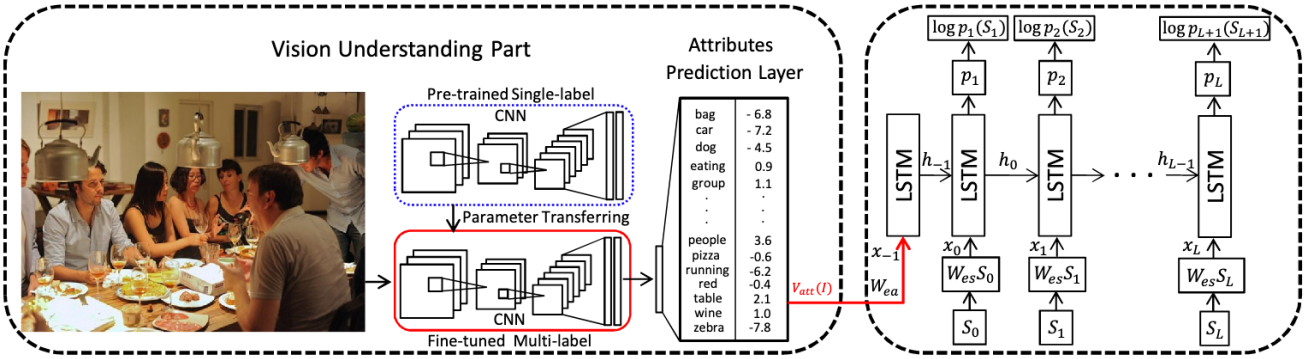
Neural Module Networks. Andreas *et al.* propose the idea of a *neural module network* (NMN) in [16]. Inspired by the use of RNNs to encode a question and train a classifier on an encoded question-image pair in common VQA literature, and the use of semantic parsers to decompose questions into logistical expressions in common textual and image QA literature, the authors develop a technique for combining the pros of RNNs with the flexibility and compositionability of symbolic approaches to semantics. The rationale behind their approach is that combining both methods avoids the pitfalls of RNNs (a monolothic network structure that is invariant despite changing inputs) and the pitfalls of semantic parsers (limited in their reasoning ability when merely applied to reasoning about truth values). Thus, their approach provides a dynamic, specialized network that reasons about the inputs in the visual and attention domain. An overview of their architecture with a sample modular network can be seen in Figure 12.

Their approach can be summarized as:

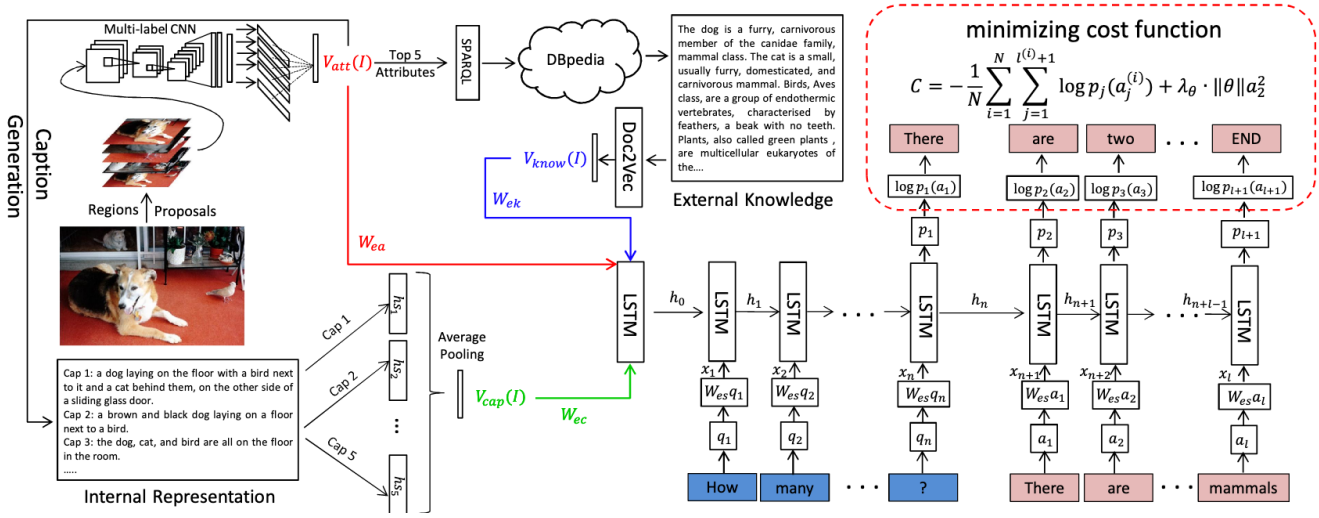
1) Extract image features using a CNN
2) Read the question using an LSTM
3) Analyze the question with a semantic parser and use this to determine: 1) the basic computational units needed to answer the question, and 2) the relationship of each computational unit to the others
4) Combine the output from the neural module network with predictions from a simple LSTM question encoder

They define several different kinds of modules, each with their own types of input and output messages. Depending on the structure, the messages may be raw image features, attentions, or classification decisions. Further, all modules are independent and composable. They define the module types as:

1) **Find[c]**. This module convolves every position in the input image with a weight vector to produce a heatmap (also known as an *unnormalized attention*), where c is the item they want to find.
2) **Transform[c]**. This module is a multilayer perceptron with ReLUs, which perform a fully-connected mapping from one attention to another. For example, where c = *"above"*, this module will take an attention and shift the region of greatest activation upward, and where c = *"not"*, the module moves attention away from the active regions.
3) **Combine[c]**. This module merges two attentions into a single attention. For example, when c = *"and"*, the resultant attention is only active in both inputs, whereas when c = *"or"*, the resultant attention is only active where the first input is active and the second is inactive.
4) **Describe[c]**. This module takes an attention and an input image and maps both to a distribution over labels. For example, when c = *"color"*, the module should return a



(a) The architecture for the image caption generation model, where the left side shows the conversion of an image into a vector of image attributes, and the right side shows the use of an LSTM to generate an appropriate caption



(b) The architecture for the VQA model, that allows for the encorporation of external knowledge. $V_{att}(I)$, in red, is the attribute representation of an image that was generated in 13a. $V_{cap}(I)$, in green, is a vector representation of the generated caption from 13a. $V_{know}(I)$, in blue, is the encorporation of image-relevant external knowledge. The LSTM that all three vectors feed into generates a single representation of an image, which is then pipelined along with a question (*"How", "many" ... "?"*) into a series of LSTMs that predict the answer.

Fig. 13: The external knowledge and attributes-based architecture in [33]

representation of the colors in the region of attention.

5) **Measure[c]**. This module take an attention and maps it to a distribution over labels, and is mostly used for deciding whether an object exists or counting a set of objects.

To create a network of modules, they first parse each question with a Stanford Parser to obtain a universal dependency representation. This both provides an abstraction of the original sentence and reduces complexity by bucketing similar words together (for example, “kites" into “kite"). Then, they express the primary part of the sentence’s meaning in a symbolic form – for example, *“Is there a circle next to a square"* becomes *is(circle, next-to(square)*. Lastly, they convert the parsed symbolic representation into neural modules through a static mapping based on the structure of the parse.

The final output of their module network is combined with a simple LSTM question encoder for two reasons: 1) It allows them to model underlying syntactic regularities in the data that cannot be captured in their neural module network because the NMN’s parser simplifies the question too aggressively, and 2) It allows them to capture semantic regularities in the face of missing or low-quality image data.

D. Knowledge-Based

The previous three methods have been focused particularly on question-image pairs where the image *contains sufficient information to answer the question*. However, these methods are useless when external knowledge is required, as a network can only draw from the (insufficient) information given in the input question and image.

Attributes and External Knowledge. In [33], Wu *et al.* propose a CNN and LSTM based model for the image captioning task, but modify the network to encorporate external knowledge and common sense for the VQA task. Intuitively, they do this by fusing an automatically generated caption of the image with information extracted from an external knowledge base, where both are represented as text. An overview of their original image caption network and their modified network can be seen in Figure 13a and Figure 13b, respectively.

First, the authors build a network to solve the image captioning task. Their network contains an image analysis portion and a caption generation portion. For the former, they train a deep CNN to solve the multi-label classification problem, where the labels are a set of attributes commonly found in image captions. These attributes can be any part of speech, as they are generated from image captions and not hard-coded to be nouns or attributes. The authors represent an image in terms of its attributes by creating a fixed-length vector $V_{att}(I)$, where the length is the size of the attribute set, and each dimension contains the probability that that attribute exists in the image. For the caption generation portion, they train a caption generation model by maximizing the probability of the correct description for a given $V_{att}(I)$. They use an LSTM to do this part. Figure 13a shows a visual overview of this method.

Then, they modify their caption generation model to encorporate external knowledge and solve the VQA task. They use their image caption generation model to generate $V_{att}(I)$, a representation of the given image's attributes. Then, they create a vector representation, $V_{cap}(I)$, of the generated image caption by average pooling the five hidden states of the LSTM in the caption generation part of their old model, within which is contained the representation of the generated caption. Lastly, they encorporate relevant external knowledge, $V_{know}(I)$, by querying DBpedia, a structured database of information from Wikipedia, for the top 5 most strongly predicted attributes for an image and converting the query results into a fixed-length feature representation with Doc2Vec. These three vectors ($V_{att}(I)$, $V_{cap}(I)$, and $V_{know}(I)$), are combined into one single representation with an LSTM, which is then used as input along with a question to an LSTM-based VQA model. This can be seen in Figure 13b.

KVQA. In [26], Shah *et al.* propose the following dilineation between VQA questions: conventional VQA, commonsense knowledge-enabled VQA, and world knowledge-aware VQA. The first answers questions that can be trivially solved with no knowledge, not even complex reasoning – questions such as *"What color is the ball?"* or *"How many people are in the image?"*. The second requires knowledge about common nouns, such as knowing that *"a microphone"* is *"the item in the image that amplifies sound"*. Lastly, the third requires information about named entities in the image, to answer questions such as *"Who is to the left of Barack Obama?"* or *"Do all the people in the image have the same occupation?"*. They propose a VQA dataset, *KVQA*, to deal with the third type of question.

IV. Future Work

Future areas of research for VQA are split into primarily two areas. Some of the latest work has pivoted into defining a narrower, more specific subset or variation of the original VQA problem, and propose models to solve these more domain-constrained problems. [38], published in 2020, focuses on the problem of *video* question answering – a problem which provides an extra dimensionality of complexity. [36] proposes a VQA model and dataset that can read and reason about *text* in the provided image. [35] proposes a VQA model that both reads text in an image and encorporates external knowledge about the text. While the previous work focuses on narrowing the scope of the problem to unexplored dimensionalities, other work focuses on perfecting the techniques we currently have for standard VQA. [39] augments the VQA task with semantic information to achieve

better performance, while [37] focuses on *explaining* the reasoning behind a VQA model's decision. Further, the problem of external knowledge-based VQA is still a new sub-area, and work in that area has not sufficiently exhausted its possibilities and problems yet.

On the other hand, as the numerous authors in section II prove, early results for VQA models have overestimated their performance because the datasets they work on are not only biased, but the entire area of VQA is particularly prone to dataset bias if the datasets are not carefully constructed. Further, as more and more specific subproblems emerge, specific datasets must be made to accomodate them, such as *KVQA* for questions that require named entity external knowledge, *TextVQA* for questions that require reading text in the image, and *imSituVQA* for imSitu verb semantic annotations [26] [36] [39].

REFERENCES

[1] Antol, Stanislaw, et al. "Vqa: Visual question answering." Proceedings of the IEEE international conference on computer vision. 2015.

[2] Y. Feng and M. Lapata, "Automatic Caption Generation for News Images," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 4, pp. 797-812, April 2013.

[3] Srivastava, Yash, et al. "Visual Question Answering using Deep Learning: A Survey and Performance Analysis." arXiv preprint arXiv:1909.01860 (2019).

[4] Wu, Qi, et al. "Visual question answering: A survey of methods and datasets." Computer Vision and Image Understanding 163 (2017): 21-40.

[5] Zhang, Peng, et al. "Yin and yang: Balancing and answering binary visual questions." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[6] Yu, Licheng, et al. "Visual madlibs: Fill in the blank image generation and question answering." arXiv preprint arXiv:1506.00278 (2015).

[7] Zhou, Bolei, et al. "Simple baseline for visual question answering." arXiv preprint arXiv:1512.02167 (2015).

[8] Fukui, Akira, et al. "Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding." Conference on Empirical Methods in Natural Language Processing. ACL, 2016.

[9] Gao, Haoyuan, et al. "Are you talking to a machine? dataset and methods for multilingual image question." Advances in neural information processing systems. 2015.

[10] Malinowski, Mateusz, Marcus Rohrbach, and Mario Fritz. "Ask your neurons: A neural-based approach to answering questions about images." Proceedings of the IEEE international conference on computer vision. 2015.

[11] Ma, Lin, Zhengdong Lu, and Hang Li. "Learning to answer questions from image using convolutional neural network." Thirtieth AAAI Conference on Artificial Intelligence. 2016.

[12] Shih, Kevin J., Saurabh Singh, and Derek Hoiem. "Where to look: Focus regions for visual question answering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[13] Xu, Huijuan, and Kate Saenko. "Ask, attend and answer: Exploring question-guided spatial attention for visual question answering." European Conference on Computer Vision. Springer, Cham, 2016.

[14] Chen, Kan, et al. "Abc-cnn: An attention based convolutional neural network for visual question answering." arXiv preprint arXiv:1511.05960 (2015).

[15] Jiang, Aiwen, et al. "Compositional memory for visual question answering." arXiv preprint arXiv:1511.05676 (2015).

[16] Andreas, Jacob, et al. "Neural module networks." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2016.

[17] Yang, Zichao, et al. "Stacked attention networks for image question answering." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[18] Lu, Jiasen, et al. "Hierarchical question-image co-attention for visual question answering." Advances in neural information processing systems. 2016.

[19] Ren, Mengye, Ryan Kiros, and Richard Zemel. "Exploring models and data for image question answering." Advances in neural information processing systems. 2015.

[20] Zhu, Yuke, et al. "Visual7w: Grounded question answering in images." Proceedings of the IEEE conference on computer vision and pattern recognition. 2016.

[21] Malinowski, Mateusz, and Mario Fritz. "A multi-world approach to question answering about real-world scenes based on uncertain input." Advances in neural information processing systems. 2014.

[22] Jabri, Allan, Armand Joulin, and Laurens Van Der Maaten. "Revisiting visual question answering baselines." European conference on computer vision. Springer, Cham, 2016.

[23] Goyal, Yash, et al. "Making the V in VQA matter: Elevating the role of image understanding in Visual Question Answering." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[24] Agrawal, Aishwarya, et al. "Don't just assume; look and answer: Overcoming priors for visual question answering." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018.

[25] Acharya, Manoj, Kushal Kafle, and Christopher Kanan. "TallyQA: Answering complex counting questions." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. 2019.

[26] Shah, Sanket, et al. "Kvqa: Knowledge-aware visual question answering." Proceedings of the AAAI Conference on Artificial Intelligence. Vol. 33. 2019.

[27] Marino, Kenneth, et al. "Ok-vqa: A visual question answering benchmark requiring external knowledge." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[28] Jiang, Yu, et al. "Pythia v0. 1: the winning entry to the vqa challenge 2018." arXiv preprint arXiv:1807.09956 (2018).

[29] Geman, Donald, et al. "Visual turing test for computer vision systems." Proceedings of the National Academy of Sciences 112.12 (2015): 3618-3623.

[30] Bigham, Jeffrey P., et al. "VizWiz: nearly real-time answers to visual questions." Proceedings of the 23nd annual ACM symposium on User interface software and technology. 2010.

[31] Das, Abhishek, et al. "Visual dialog." Proceedings of the IEEE

Conference on Computer Vision and Pattern Recognition. 2017.

[32] Das, Abhishek, et al. "Learning cooperative visual dialog agents with deep reinforcement learning." Proceedings of the IEEE international conference on computer vision. 2017.

[33] Wu, Qi, et al. "Image captioning and visual question answering based on attributes and external knowledge." IEEE transactions on pattern analysis and machine intelligence 40.6 (2017): 1367-1381.

[34] Johnson, Justin, et al. "Inferring and executing programs for visual reasoning." Proceedings of the IEEE International Conference on Computer Vision. 2017.

[35] Singh, Amanpreet, et al. "Towards vqa models that can read." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[36] Singh, Ajeet Kumar, et al. "From Strings to Things: Knowledge-enabled VQA Model that can Read and Reason." Proceedings of the IEEE International Conference on Computer Vision. 2019.

[37] Li, Qing, et al. "Vqa-e: Explaining, elaborating, and enhancing your answers for visual questions." Proceedings of the European Conference on Computer Vision (ECCV). 2018.

[38] Yang, Zekun, et al. "BERT representations for Video Question Answering." The IEEE Winter Conference on Applications of Computer Vision. 2020.

[39] Alizadeh, Mehrdad, and Barbara Di Eugenio. "Augmenting Visual Question Answering with Semantic Frame Information in a Multitask Learning Approach." 2020 IEEE 14th International Conference on Semantic Computing (ICSC). IEEE, 2020.

[40] Kafle, Kushal, et al. "Answering Questions about Data Visualizations using Efficient Bimodal Fusion." The IEEE Winter Conference on Applications of Computer Vision. 2020.

[41] Pfungst, Oskar. *Clever Hans:(the horse of Mr. Von Osten.) a contribution to experimental animal and human psychology.* Holt, Rinehart and Winston, 1911.

[42] Johnson, Justin, et al. "Clevr: A diagnostic dataset for compositional language and elementary visual reasoning." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2017.

[43] Kafle, Kushal, Robik Shrestha, and Christopher Kanan. "Challenges and Prospects in Vision and Language Research." arXiv preprint arXiv:1904.09317 (2019).

[44] Lin, Tsung-Yi, et al. "Microsoft coco: Common objects in context." European conference on computer vision. Springer, Cham, 2014.

[45] Kong, Chen, et al. "What are you talking about? text-to-image coreference." Proceedings of the IEEE conference on computer vision and pattern recognition. 2014.

[46] Hudson, Drew A., and Christopher D. Manning. "GQA: A new dataset for real-world visual reasoning and compositional question answering." Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2019.

[47] Zheng, B., Zhao, Y., Yu, J. et al. Scene Understanding by Reasoning Stability and Safety. Int J Comput Vis 112, 221–238 (2015). https://doi.org/10.1007/s11263-014-0795-4

[48] Battaglia, Peter W., Jessica B. Hamrick, and Joshua B. Tenenbaum. "Simulation as an engine of physical scene understanding." Proceedings of the National Academy of Sciences 110.45 (2013): 18327-18332.

[49] Mottaghi, Roozbeh, et al. "“What happens if...” Learning to Predict the Effect of Forces in Images." European conference on computer vision. Springer, Cham, 2016.

[50] Hoiem, Derek, et al. "Guest editorial: Scene understanding." International Journal of Computer Vision 112.2 (2015): 131-132.

[51] Cantrell, Rehj, et al. "Robust spoken instruction understanding for HRI." 2010 5th ACM/IEEE International Conference on Human-Robot Interaction (HRI). IEEE, 2010.

[52] Matuszek, Cynthia, et al. "A joint model of language and perception for grounded attribute learning." arXiv preprint arXiv:1206.6423 (2012).