

# Towards Open-Ended VQA Models Using Transformers

Anonymous AAAI Submission

## Abstract

Visual Question Answering (VQA) is an open and active research task with many exciting practical applications. However, most prior work on this problem shares a common underlying limitation: the number of possible answers is restricted to a limited set of candidates, constraining the resulting models' power and flexibility. We address this shortcoming by introducing a novel Transformer-based architecture for VQA, designed to generate open-ended answers. This shift towards generating unconstrained text rather than selecting from discrete categories represents a substantial contribution towards developing more humanlike VQA systems. We show that our architecture's performance is competitive with traditional, classification-based VQA architectures, laying promising groundwork for future open-ended, generative VQA models.

## Introduction

Visual Question Answering (VQA) is a multidisciplinary research problem that synthesizes contributions from computer vision and natural language processing to interpret and respond to open-domain questions about images. This task is challenging because it requires both an ability to comprehend the question, and an ability to apply reasoning skills to the associated image to seek relevant information. If both tasks are performed adequately, the resulting VQA system should be capable of generating meaningful natural language answers that preserve both semantic and syntactic correctness.

Despite the difficulty of these interrelated tasks, current VQA architectures achieve strong results and can generalize reasonably well across different question-image pairs (Antol et al. 2015; Andreas et al. 2016; Lu et al. 2016; Hu et al. 2017; Tan and Bansal 2019). Still, they possess a shared limitation: the generated answers tend to be short and concise (e.g., "yes," "no," "blue"), with limited or no supplemental context. This is partially a side effect of how the VQA problem is commonly framed, with language generation modelled as a classification task rather than the open-ended exercise more reflective of how it is approached by humans. Because of this framing, answers are forced to lie within a limited, predefined set of candidates, creating a dichotomy between the human and machine scenarios; while humans enrich their answers with

supplemental content and arguments, VQA systems tend to rely on outputting single words as a performance-preserving mechanism.

We tackle the VQA task from a different perspective. Instead of distributing probabilities over a predefined set of possible candidate answers, we employ a state-of-the-art Transformer model (Radford et al. 2019) to generate image-relevant answers autoregressively, with the goal of generating longer, more descriptive answers rather than the commonly-seen one-word responses. In doing so, we seek to empirically assess the potential of Transformers for an established, inherently multimodal task; advance the current understanding of generative VQA models; and examine their utility as an alternative to more classical, discriminative approaches. Our contributions are as follows:

1. We leverage the Transformer architecture in a VQA system to develop a model capable of exploiting linguistic knowledge to generate more natural and open-ended answers.
2. We empirically demonstrate the feasibility of combining a Transformer language model with common visual feature extractors (e.g., convolutional neural networks), to condition the generation of each token not only on the textual modality but on visual input as well.
3. To this end, we introduce a multimodal variation of the Transformer architecture such that feature maps output from the image encoder are taken into consideration when generating tokens.
4. We show that our novel architecture is competitive with classical VQA models while producing qualitatively more descriptive answers, providing evidence that unconstrained, generative models are a viable approach for VQA and establishing a benchmark to stimulate further research in this area.

We describe our methods and evaluation procedures in more detail in the following sections.

## Related Work

Prior work in VQA has primarily shared a common structure (Antol et al. 2015; Andreas et al. 2016; Lu et al. 2016; Hu et al. 2017; Tan and Bansal 2019). First, encoders (e.g., long short-term memory (LSTM) or convolutional neural

network (CNN) models) extract features from visual and textual inputs. Then, the two modalities are joined in a common subspace to form a higher-level representation. Finally, a classifier distributes probabilities over a vocabulary of possible answers. A host of smaller variations to this common structure exist across architectures; we describe high-performing recent models in more detail.

**MCB.** Multimodal Compact Bilinear Pooling (MCB) (Fukui et al. 2016) is a robust architecture that exploits an outer product to combine the two modalities, to maximize the multiplicative interaction between their two vector representations. Fukui *et al.* (2016) devised this mechanism under the hypothesis that existing methods for combining the two modalities were insufficient for projecting them in the right space. In MCB, the vector representations for text and visual inputs are computed, respectively, by a two-layer LSTM and a Residual Network (He et al. 2016). Once the multimodal representations are encoded, they are pooled together. Traditional bilinear pooling results in high-dimensional vectors; however, MCB efficiently compresses the output for every modality by first projecting image and question into a higher-dimensional space using the Count Sketch method (Charikar, Chen, and Farach-Colton 2002), and then convolving the two representations via the element-wise product in the Fast Fourier Transform space (Gao et al. 2016). Given the attention output computed by MCB and the output of the LSTM, the final answer is generated using a classifier achieving 62.5% accuracy on the VQA dataset in (Antol et al. 2015).

**BUTD.** Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering (Anderson et al. 2018) is a recent architecture that achieves an accuracy of 70.3% on the Visual Question Answering V2 (VQAv2) dataset (Goyal et al. 2017). It combines bottom-up and top-down attention mechanisms that enable attention to be calculated both at the level of objects and other salient image regions. The bottom-up attention mechanism accounts for determining interesting regions in the input image, whereas the top-down mechanism determines how to weight each region. The final classification model distributes probabilities over a fixed set of candidate answers.

**LXMERT.** Tan and Bansal’s (2019) architecture is the current state-of-the-art on the VQAv2 dataset, achieving a 72.5% accuracy. Similarly to our approach, LXMERT incorporates a Transformer architecture, although it does so for a different purpose. The large-scale model consists of three encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. To train the model to connect vision and language, authors pre-train the model with large amounts of image-and-sentence pairs. The model does not generate open-ended answers, with the authors instead exploiting Transformers as more powerful encoders. Ultimately, the cross-modal intermediate output is fed to a classifier, which still distributes probabilities over a predefined vocabulary of answers.

## Architecture

In this work, we introduce a multimodal generative VQA model (VGGPT-2) that employs attention mechanisms at multiple levels and focuses on specific portions of the image using information contained in the question. This information is also employed to mask out irrelevant portions of the question, facilitating the generation of accurate answers. Four components work together to process the visual and textual modalities and subsequently form natural, unconstrained answers. We describe these components in the subsections below, and provide a high-level overview of the architecture in Figure 1.

### Image Encoder

The first component, an image encoder, consists of a pre-trained VGGNet-11<sup>1</sup> model (Simonyan and Zisserman 2014). We employ this model to extract visual features from the images associated with the input questions. Since, in our context, we do not need VGGNet to act as a classifier, but rather as a feature extractor, we follow common practice and drop the last three fully-connected layers, retaining only the output of the last average pooling layer. This way, given an image, the encoder produces a set of maps containing high-level features.

Specifically, letting  $c_i$  be three different channels (red, blue, and green) of width  $w_i = 224$  and height  $h_i = 224$ , the visual input consists of an image  $\mathbf{I} \in \mathbb{R}^{c_i \times w_i \times h_i}$ . The output of the image encoder then consists of a set of maps  $\mathbf{M} \in \mathbb{R}^{c_o \times w_o \times h_o}$ , where  $c_o = 512$  different channels of width  $w_o = 7$  and height  $h_o = 7$ .  $\mathbf{M}$  then serves as one of the two inputs to our attention mechanism.

### Language Model

We use OpenAI’s GPT-2 (small, 117M) Transformer (Radford et al. 2019) as our base language model, refining their pre-trained architecture in order to integrate it into our system. GPT-2 is a powerful, Transformer-based language model that relies entirely on attention mechanisms to condition the generation of tokens on the given context and the tokens generated at previous time steps. It consists of two core blocks: (1) the Transformer, which is responsible for computing the multi-head attention discussed previously; and (2) the final classifier which, for each token processed by the Transformer, produces a distribution of probabilities over the vocabulary of words.<sup>2</sup>

Since our goal is for the language model to interact with the image encoder, we split apart these two blocks and use them separately in our model. Given a sequence of  $n$  input tokens  $\mathbf{S} \in \mathbb{R}^{n \times 1}$ , the output of the Transformer  $\mathbf{T} \in \mathbb{R}^{n \times \text{hidden}}$  consists of a sequence of  $n$  hidden vectors of size  $\text{hidden} = 768$ . We use  $\mathbf{T}$  as the second input to our attention mechanism.

<sup>1</sup>We do not experiment with deeper vision architectures here since our primary focus is on enhancing language output.

<sup>2</sup>Recall that this is a language model that exploits attention, which means that the probability  $p(w_i | t_i, t_{i-1}, \dots, t_{i-k})$  of generating the word  $w_i$  at time step  $i$  is conditioned on the input tokens  $t_i, t_{i-1}, \dots, t_{i-k}$  up to time step  $i$ .

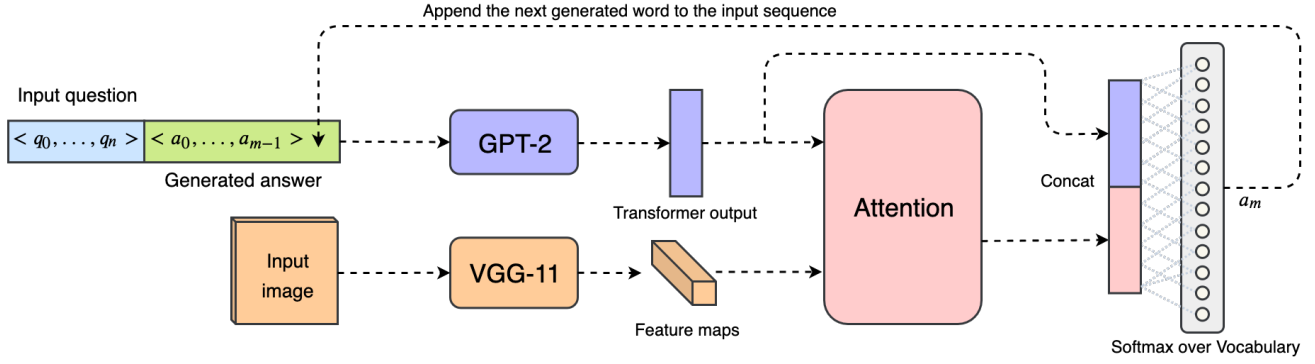


Figure 1: VGGPT-2 architecture overview, highlighting the answer generation process beginning from a question/image pair.

## Attention

A crucial component of our architecture is a novel co-attention mechanism, which is responsible for combining features from the two different sub-spaces into a single representation. In designing this mechanism, we use the question to drive the focus on the image and vice versa. To formally define this component, we recall that it leverages the following inputs:

- **Image Encoder** output:  $\mathbf{M} \in \mathbb{R}^{512 \times 7 \times 7}$
- **Language Model** output:  $\mathbf{T} \in \mathbb{R}^{n \times 768}$

Our model first flattens  $\mathbf{M}$  into a more convenient representation  $\mathbf{M}_{\text{flat}} \in \mathbb{R}^{512 \times 49}$ . Then, both  $\mathbf{M}_{\text{flat}}$  and  $\mathbf{T}$  are fed to two separate linear layers ( $\mathbf{Lin}_M$  and  $\mathbf{Lin}_T$ ) that bring the two vectors into a common subspace of dimensionality  $\text{AttDim} = 512$ :

- $\mathbf{Lin}_M$ : Maps  $\mathbf{M}_{\text{flat}} \in \mathbb{R}^{512 \times 49}$  to  $\mathbf{M}_{\text{att}} \in \mathbb{R}^{\text{AttDim} \times 49}$
- $\mathbf{Lin}_T$ : Maps  $\mathbf{T} \in \mathbb{R}^{n \times 768}$  to  $\mathbf{T}_{\text{att}} \in \mathbb{R}^{n \times \text{AttDim}}$

At this point, for every element  $\mathbf{T}_{\text{att}}^i \in \mathbb{R}^{\text{AttDim}}$  in  $\mathbf{T}_{\text{att}}$ , with  $i \in \mathbb{N} \wedge i \in [0, n)$ , we repeat the following set of operations:

1. We sum  $\mathbf{T}_{\text{att}}^i$  with each element  $\mathbf{M}_{\text{att}}^k \in \mathbb{R}^{\text{AttDim}}$  of  $\mathbf{M}_{\text{att}}$ , with  $k \in \mathbb{N} \wedge k \in [0, 49)$ , obtaining the first multi-modal attention representation  $\mathbf{A}_1^i \in \mathbb{R}^{\text{AttDim} \times 49}$  for the  $i^{\text{th}}$  token in the input sequence.
2. We rectify  $\mathbf{A}_1^i$  using a *ReLU* unit and obtain an intermediate attention representation  $\mathbf{A}_2^i \in \mathbb{R}^{\text{AttDim} \times 49}$ .
3. Using a linear layer ( $\mathbf{Lin}_{\text{soft}}$ ) we compress  $\text{AttDim}$  down to 1. In other words,  $\mathbf{Lin}_{\text{soft}}$  brings  $\mathbf{A}_2^i \in \mathbb{R}^{\text{AttDim} \times 49}$  into  $\mathbf{A}_3^i \in \mathbb{R}^{1 \times 49}$ .
4. To obtain a softmax that highlights relevant image regions given a specific word at position  $i$ , we pass  $\mathbf{A}_3^i$  through a *Softmax* function and obtain  $\mathbf{A}_4^i \in \mathbb{R}^{1 \times 49}$ , where  $\mathbf{A}_4^{i,h} \in (0, 1)$  with  $h \in \mathbb{N} \wedge h \in [0, 49)$ . At this point every pixel in  $\mathbf{A}_4^i$  has an associated value between 0 and 1, which indicates how important that pixel is to answer the question.
5. We perform a **point-wise multiplication** between every element  $\mathbf{M}_{\text{flat}}^k \in \mathbb{R}^{512 \times 49}$  in the original flattened maps  $\mathbf{M}_{\text{flat}}$  and  $\mathbf{A}_4^i$ , obtaining a new vector  $\mathbf{A}_5^i \in \mathbb{R}^{512 \times 49}$ .

In this step we perform the **question-to-image attention**, since we exploit the softmax computed at the previous step to mask out irrelevant portions of the image.

6. We sum, for each channel in  $\mathbf{A}_5^i$ , all the masked pixels together, reducing the dimension from 49 to 1. In other words, we go from  $\mathbf{A}_5^i \in \mathbb{R}^{512 \times 49}$  to  $\mathbf{A}_6^i \in \mathbb{R}^{512 \times 1}$  by summing all the elements together on the pixel axis.
7. We bring  $\mathbf{A}_6^i$  back to the Transformer space through a final linear layer ( $\mathbf{Lin}_{\text{out}}$ ), which expands  $\mathbf{A}_6^i \in \mathbb{R}^{512 \times 1}$  to  $\mathbf{A}_7^i \in \mathbb{R}^{768}$ .
8. We compute the **image-to-question attention** through a final **point-wise multiplication** between  $\mathbf{A}_7^i$  and every element  $\mathbf{T}_i \in \mathbb{R}^{768}$  of  $\mathbf{T}$ , which is the original output of the Transformer before this layer. The final output of the attention mechanism for every embedded token  $\mathbf{T}_i$  and the maps  $\mathbf{M}$  coming from the image encoder is the vector  $\mathbf{A}_{\text{out}}^i \in \mathbb{R}^{768}$ .

We described how the attention layer works considering one single embedded token  $\mathbf{T}_i$  at a time. However, it can be trivially scaled to a sequence of  $n$  elements by including an additional dimension for each vector that accounts for the token under consideration. The output  $\mathbf{A}_{\text{out}}$  of the attention mechanism is then combined back with the output of the language model to generate the answer.

## Answer Generator

We generate the final answer by once again exploiting the language model. This last component diverges from classical VQA systems—while most existing work has fed the output of the attention mechanism to a classifier, thereby distributing probabilities over a predefined set of answers, we condition the generation of words using the output of the attention layer.

Instead of feeding the output  $\mathbf{T}$  of the Transformer directly to its head, we first take the attention output  $\mathbf{A}_{\text{out}}$  and concatenate it with the Transformer output  $\mathbf{T}$  into a new vector  $\mathbf{O} \in \mathbb{R}^{n \times 1536}$ . Then, with the help of a new linear layer ( $\mathbf{Lin}_{\text{Classifier}}$ ), we distribute probabilities over the original vocabulary of words from the pre-trained GPT-2 model. In order not to lose the valuable information contained in the

weights of the pre-trained model **C** (or **head**), we do the following:

1. We initialize the first  $768 \times 50254$  weights of **Lin**<sub>Classifier</sub> with the ones present in the original **head** of GPT-2.
2. We initialize the last  $768 \times 50254$  weights, accounting for the attention layer output, to 0.

This allows the model to start training from a state in which it is already able to generate natural sequences of words, since all weights relevant to the attention mechanism are initialized to 0, and the only ones that influence the output are coming from **T**. This last linear layer consists of roughly 77.1M parameters ( $1536 \times 50254$ ), so it is imperative to perform such an initialization to train the system in a reasonable amount of time.

Afterward, we feed **O**, the **Logits** vector, to a *Softmax* function that distributes probabilities over our vocabulary. To generate the output word  $w^i$  corresponding to the input token at position  $i$ , we pool the most probable word in a greedy fashion,<sup>3</sup> dynamically generating each word and conditioning it on the previously-generated words. At evaluation time, this allows the system to generate arbitrarily long answers in the following fashion:

1. Given a question of  $n$  words and an image, we first let the system generate the initial word in the answer, conditioned only on the question and the image.
2. After generating the first word, we append it to the original question, which becomes a sequence of  $n + 1$  words. The next word in the answer is generated conditioned on the new sequence.
3. The procedure continues until the system outputs the `<eos>` token or a maximum length threshold is reached.

## Dataset

Following other recent work on VQA, we train and evaluate our model using the VQAv2 dataset (Goyal et al. 2017), a large collection of real-world question-image pairs. Although the corpus is primarily meant for standard classifier-driven VQA models, its use facilitates direct comparison with existing models. Such a comparison is key to demonstrating the effectiveness of open-ended, generative VQA models such as that designed here.

Importantly, the dataset is based on real-world images rather than synthetic ones, allowing for learning better generalization capabilities. It consists of more than 200K images, 1M questions and 10M ground truth answers, manually annotated. There are at least 3 questions (5.4 on average) per image, and each question/image pair has 10 different annotated answers. To disincentivize answers based purely on linguistic bias, some questions have complementary image pairs. Additionally, answers were balanced for different question types in order not to have a predominant value.

<sup>3</sup>We also experimented with beam search, finding that with beam sizes greater than one, the model generated shorter answers more characteristic of those produced by traditional classification-based VQA systems.

	Train	Test
Split size	$10^6$	$10^5$
Sequence length	24	22
Question + Answer	yes	no
Min. question length	5	6
Max. question length	21	22
Avg. question length	8.45	9.48
Min. answer length	3	-
Max. answer length	18	-
Avg. answer length	3.89	-
Avg. <code>&lt;pad&gt;</code> count	11.65	-
# of unique images	71450	32822

Table 1: Structure of the processed dataset.

## Preprocessing

We apply a variety of preprocessing steps to the text and images in order to most productively make use of the corpus. We summarize the structure of the preprocessed dataset in Table 1, and elaborate further on our preprocessing steps below.

**Grayscale removal.** We discarded all samples with non-RGB images, retaining only colored ones.

**Answer length.** Since we were interested in building an architecture capable of generating open-ended answers, we prioritized longer samples and gave less importance to those with concise answers.

**Answer variability.** We randomly selected up to four annotations for each question, ensuring that at most four identical question/image pairs were fed to the model, with four different answers.

**Data formatting.** Given a question/answer pair for a given image, we concatenated the pair to form a sequence of words. This sequence was then fed to the model at training time. For the test data, we dropped all answers and kept only the question/image pairs. All textual inputs were byte-pair encoded (Shibata et al. 1999) using the pre-trained GPT-2 vocabulary.

**Custom tokens.** We introduce four new tokens: `<bos>`, `<eos>`, `<sep>`, and `<pad>` to indicate the beginning and end of the sequence, the separation between question and answer, and finally the padding token. In the training set, we make use of all four tokens while encoding the input sequence. In the test set, we only use `<bos>` and `<sep>` since no answer is present. Since we introduce these new tokens, we set the language model’s weights to be trainable in order to update the embedding matrix and the attention layers.

## Training

To train our architecture we employ a technique called Teacher Forcing (Lamb et al. 2016): given a question with its associated answer, we combine the two to form a sequence of tokens that are fed to the model. Exploiting a *CrossEntropy* loss function, we train our system to predict the input sequence shifted by 1 time step, effectively telling the model to condition the probability of generating a new token on the previously seen context. The loss function ignores the  $\langle pad \rangle$  tokens and focuses only on portions of the sequence that contain meaningful information. We block weight updates in our image encoder, using it solely as a visual feature extractor. However, we set our language model’s weights to be trainable. This is necessary for the model to learn both the nature of the task and the meaning of the newly introduced tokens. We train the system for 20 epochs using batches of 20 elements.<sup>4</sup> The learning rate is initialized to  $5 \times 10^{-5}$  and Adam is employed as the optimizer. The overall architecture consists of 202M parameters, since we fine-tune GPT-2 (117M) alongside the attention mechanism (8M) and the final classifier (77M).

## Evaluation

To assess the performance of our architecture, we compare it against three different baselines designed to disentangle the performance of our model relative to exclusively vision- or text-conditioned alternatives, as well as a standard classification-based VQA approach:

- **Captioning baseline:** The first baseline that we consider is a captioning system (Xu et al. 2015): given an image, this model tries to generate a meaningful description about what it sees. Our goal was to understand the extent to which the textual modality (i.e., the question) was useful to generate answers. The captioning baseline exploits a VGGNet-11 image encoder to extract visual features that are later combined with the outputs of an LSTM. The joint representation then passes through an attention layer, which focuses on specific parts of the image and updates the hidden states of the LSTM. The caption is generated one token at a time and the process goes on until an  $\langle eos \rangle$  token is produced. We train this model using Teacher Forcing and a loss computed with a doubly stochastic attention regularization that updates the gradient using *CrossEntropy* on the output sequence, with a regularization factor computed starting from the attention maps.
- **Answering baseline:** To check if our architecture was making use of the information contained in the input images, we considered another baseline starting from a pre-trained instance of GPT-2 (Small). We fine-tuned the system to learn how to answer the questions exploiting only the linguistic bias present in the question, without conditioning the generation of the answer on the images.
- **VQA baseline:** For a direct comparison with a standard, publicly-available VQA system, we consider a third baseline (Kazemi and Elqursh 2017) that uses both question

<sup>4</sup>Training took less than 60 hours on an NVidia RTX2070 GPU.

Model	Overall	Yes/No	Number	Other
Captioning	0.00	0.00	0.00	0.00
Answering	0.11	0.20	0.20	0.14
VQA	<b>0.46</b>	0.71	<b>0.27</b>	<b>0.41</b>
<b>VGGPT-2</b>	0.34	<b>0.73</b>	0.23	0.24

Table 2: Accuracy scores of baselines and VGGPT-2.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
Captioning	0.100	0	0.100	0
Answering	0.293	0.149	0.074	0.027
VQA	<b>0.549</b>	0.187	0.066	0.011
<b>VGGPT-2</b>	0.463	<b>0.223</b>	<b>0.101</b>	<b>0.036</b>

Table 3: BLEU- $k$  scores of baselines and VGGPT-2.

and image to generate the answer. This is a strong baseline specifically designed for VQA tasks, and a sound comparative architecture.<sup>5</sup> The system consists of an image encoder (ResNet-152 (He et al. 2016)), alongside an LSTM which extracts textual features. A compact, but powerful, attention layer brings the two modalities into a common subspace where they are combined to generate attention maps over the image. Finally, the attention output is concatenated back with the LSTM hidden state and the resulting vector is fed to a classifier which distributes probabilities over 3000 possible answers. We do not re-train or fine-tune this architecture, and download the authors’ pre-trained instance on the VQAv2 dataset. This architecture follows the standard VQA approach of addressing the task in a classification manner, thereby generating short and concise answers.

## Quantitative Results

We evaluate our models using three different quantitative metrics: Accuracy, BLEU (Papineni et al. 2002), and Word Mover’s Distance (Kusner et al. 2015). We describe each briefly here.

**Accuracy.** Nearly all existing VQA work reports accuracy as the primary metric, computed as the percentage of times an answer exactly matches *one of* the ground truths. Even though our architecture does not generate answers that span across a fixed set of possible choices, we report these results as a comparative means for traditional VQA approaches. It is

<sup>5</sup>At the time of writing, the state of the art for the VQAv2 dataset is LXMERT (Tan and Bansal 2019), which is optimized for the classic, classification-based VQAv2 task. Since the point of our work is not to build the best-performing system for the VQAv2 task, but rather to tackle the more challenging problem of generating open-ended responses to VQA questions, we decided it a distraction to compare directly with LXMERT; hence, we selected Kazemi and Elqursh’s well-known, strong VQA baseline.

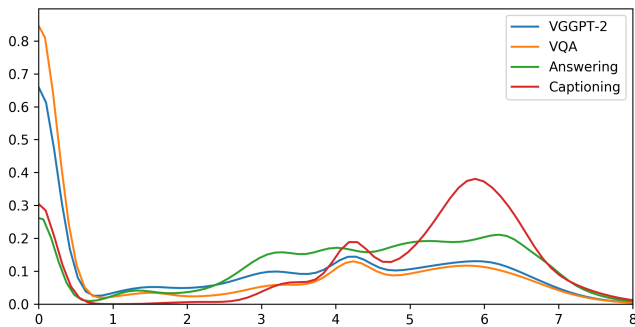


Figure 2: Word Mover's Distance distribution.

critical to note that if a ground truth is, say, “sunny,” and our system outputs “it is sunny,” the accuracy would be 0. Since our architecture’s goal is to generate longer, open-ended answers, this problem occurs very frequently. Nevertheless, we find that our model’s accuracy is comparable to that of the VQA baseline (Table 2). Interestingly, when we compare the shortest answers (i.e., binary outputs), VGGPT-2 achieves a higher accuracy than the VQA baseline; it appears that when the model does output short answers, it does so with relatively high precision. For other question types, the VQA baseline performs better, as expected given the preponderance of short, concise answers in both the gold standard and VQA baseline’s output.

**BLEU.** A commonly used evaluation metric for generative language tasks is BLEU (Papineni et al. 2002), and thus we incorporated it as an additional evaluation metric here. We compute a corpus-level BLEU score by comparing each output with its respective 10 ground truth answers from the VQAv2 corpus. In Table 3 we report BLEU- $k$  scores.<sup>6</sup> From these results we highlight a couple interesting findings:

1. As we consider longer  $n$ -gram overlaps (BLEU-2, -3 and -4), VGGPT-2 performs better relative to the VQA baseline. Thus, our architecture is superior when it comes to generating longer answers, while the VQA baseline is strongest with single word outputs.
2. The two VQA models outperform the captioning<sup>7</sup> and (particularly importantly) answering baselines, confirming that multimodality is key to answer quality.

**Word Mover’s Distance.** To check the semantic similarity between the answers generated by our models and the ground truths, we also evaluated our results using a metric that exploits pre-trained word embeddings, known as Word Mover’s Distance (WMD) (Kusner et al. 2015). We employ WMD with 100-dimensional pre-trained GloVe embeddings (Pennington, Socher, and Manning 2014), finding evidence

<sup>6</sup>To compute BLEU- $k$  we equally weight  $n$ -gram matches from 1 through  $k$ .

<sup>7</sup>In general, we observe that the image captioning baseline failed to properly learn the task, as evidenced by dismal scores for both accuracy and BLEU.




Question & Model	Image & Answer
<p>(a): <i>What is in the plate?</i> VQA-baseline VGGPT-2</p>	 Salad Toast, orange and salad
<p>(b): <i>What color is the train?</i> VQA-baseline VGGPT-2</p>	 Red Red and white
<p>(c): <i>What are the animals doing?</i> VQA-baseline VGGPT-2</p>	 Drinking Walking in water

Table 4: Qualitative results on real-world images.

that VGGPT-2 indeed generates semantically high-quality answers, further supporting the trend of high performance observed with both accuracy and BLEU metrics.

In Figure 2 we plot the WMD distribution for each model. The plot demonstrates that VQA architectures generate a larger amount of answers whose WMD is small (sharper peak around 0), indicating that the latter models are making good use of both the visual and textual modality to generate the outputs. If we compare the VQA baseline with VGGPT-2, we see a strong correlation between the two models since they both tend to produce answers which are semantically close to the ground truths. Overall, the WMD distribution effectively confirms the general trend seen with our other evaluation metrics, and supports the notion that an open-ended, generative architecture offers a feasible alternative to traditional means of addressing the VQA task.



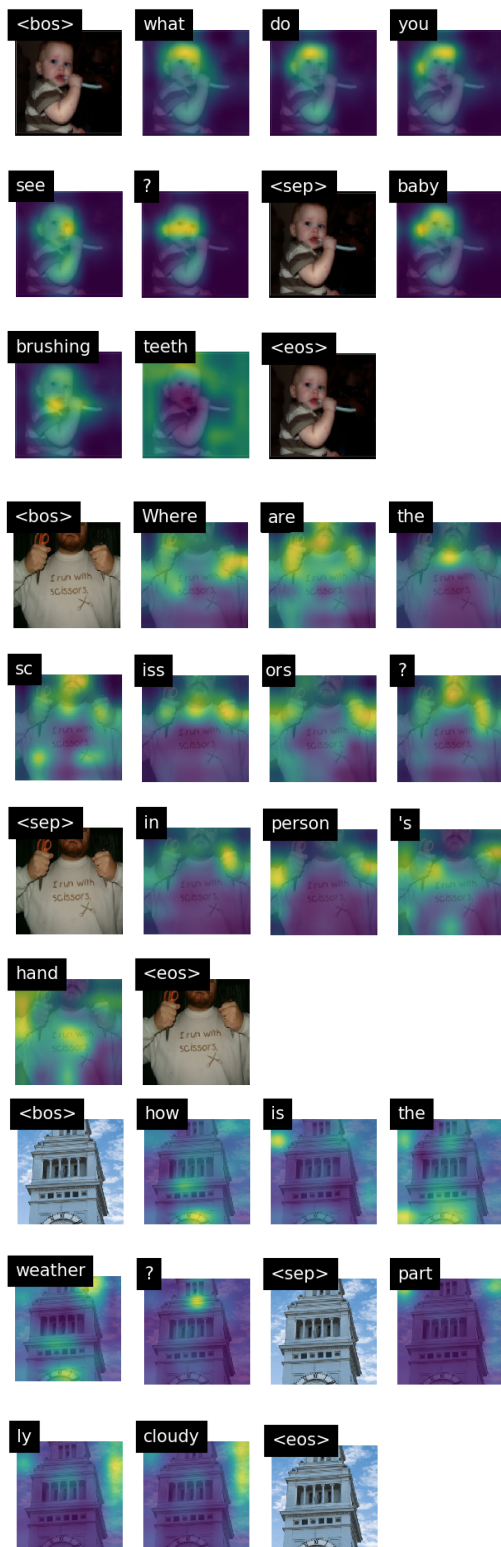


Figure 3: Examples of questions and automatically generated responses for three different images. Attention maps for each question/answer token are shown. Special tokens `<bos>`, `<sep>` and `<eos>` delimit the start and end of the question and answer.

## Qualitative Results

In addition to the quantitative results, we perform a qualitative analysis of our architecture’s performance. In Table 4, we compare answers for specific question/image pairs between VGGPT-2 and the VQA baseline. We discover that our model generates, most of the time, longer and more explanatory answers, exploiting the autoregressive behavior of the underlying language model. In contrast, the VQA baseline generates shorter and more concise outputs. This issue becomes particularly evident when the question requires the output to be descriptive, such as the first two questions in Table 4, where the baseline is forced to pick only one of the objects present on the plate, or one of the colors of the train. Finally, the third question demonstrates the language model’s ability to generate more specific action descriptions than are possible with the classification-based approach.

In Figure 3, we show the answers produced by VGGPT-2 for three other images (a baby brushing its teeth, a man holding scissors, and a clock tower). The attention mask inferred by the model for each question and answer token is shown, demonstrating the ability of the model to identify salient aspects of the images in order to answer the question correctly.

The case studies presented in Table 4 and Figure 3 also underscore the necessity of the spatial-attention mechanism for generating descriptive answers while maintaining answer correctness. Responses successfully communicating multiple concepts and/or their relations (e.g., “Toast, orange and salad,” “Walking in water,” or “in person’s hand”) would not be possible without spatial attention guiding the effective synthesis of linguistic and visual cues.

## Conclusion

In this work, we proposed an elegant Transformer-based architecture for visual question answering that combines textual and visual modalities to generate unconstrained, open-ended answers. We empirically verify the feasibility of this architecture by framing the problem in a way that diverges from the standard classification perspective, by building a system that does not distribute probabilities over a fixed set of possible answers but generates them dynamically one token at a time. We provide ample quantitative and qualitative results to justify the effectiveness and competitiveness of this novel, open-ended approach.

In future work we plan to experiment with deeper image encoders, such as Residual Networks (He et al. 2016), and will experiment with a multi-head variation of our attention mechanism. We will also examine deeper Transformer architectures in this context.<sup>8</sup> It is our hope that the work reported here stimulates additional interest in open-ended models in the VQA domain, and to that end we will also release our model and source code online to facilitate ease of replication.<sup>9</sup>

<sup>8</sup>For example, GPT-2 (Large) or perhaps even a variant of GPT-3 (Brown et al. 2020), a 175 billion parameter model not yet available for public use.

<sup>9</sup>Link in camera-ready paper.

## References

- Anderson, P.; He, X.; Buehler, C.; Teney, D.; Johnson, M.; Gould, S.; and Zhang, L. 2018. Bottom-Up and Top-Down Attention for Image Captioning and Visual Question Answering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 6077–6086. IEEE.
- Andreas, J.; Rohrbach, M.; Darrell, T.; and Klein, D. 2016. Neural module networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 39–48.
- Antol, S.; Agrawal, A.; Lu, J.; Mitchell, M.; Batra, D.; Lawrence Zitnick, C.; and Parikh, D. 2015. VQA: Visual Question Answering. In *The IEEE International Conference on Computer Vision (ICCV)*.
- Brown, T. B.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; Agarwal, S.; Herbert-Voss, A.; Krueger, G.; Henighan, T.; Child, R.; Ramesh, A.; Ziegler, D. M.; Wu, J.; Winter, C.; Hesse, C.; Chen, M.; Sigler, E.; Litwin, M.; Gray, S.; Chess, B.; Clark, J.; Berner, C.; McCandlish, S.; Radford, A.; Sutskever, I.; and Amodei, D. 2020. Language Models are Few-Shot Learners. *arXiv preprint arXiv:2005.14165*.
- Charikar, M.; Chen, K.; and Farach-Colton, M. 2002. Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, 693–703. Springer.
- Fukui, A.; Park, D. H.; Yang, D.; Rohrbach, A.; Darrell, T.; and Rohrbach, M. 2016. Multimodal Compact Bilinear Pooling for Visual Question Answering and Visual Grounding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 457–468. Austin, Texas: Association for Computational Linguistics. doi:10.18653/v1/D16-1044. URL <https://www.aclweb.org/anthology/D16-1044>.
- Gao, Y.; Beijbom, O.; Zhang, N.; and Darrell, T. 2016. Compact bilinear pooling. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 317–326.
- Goyal, Y.; Khot, T.; Summers-Stay, D.; Batra, D.; and Parikh, D. 2017. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*.
- He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep residual learning for image recognition. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 770–778.
- Hu, R.; Andreas, J.; Rohrbach, M.; Darrell, T.; and Saenko, K. 2017. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the Conference on Computer Vision and Pattern Recognition (CVPR)*, 804–813.
- Kazemi, V.; and Elqursh, A. 2017. Show, Ask, Attend, and Answer: A Strong Baseline For Visual Question Answering. URL <https://arxiv.org/abs/1704.03162>.
- Kusner, M.; Sun, Y.; Kolkin, N.; and Weinberger, K. 2015. From word embeddings to document distances. In *International Conference on Machine Learning*, 957–966.
- Lamb, A. M.; Goyal, A. G. A. P.; Zhang, Y.; Zhang, S.; Courville, A. C.; and Bengio, Y. 2016. Professor forcing: A new algorithm for training recurrent networks. In *Advances In Neural Information Processing Systems*, 4601–4609.
- Lu, J.; Yang, J.; Batra, D.; and Parikh, D. 2016. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*, 289–297.
- Papineni, K.; Roukos, S.; Ward, T.; and Zhu, W.-J. 2002. Bleu: a Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318. Philadelphia, Pennsylvania, USA: Association for Computational Linguistics. doi:10.3115/1073083.1073135. URL <https://www.aclweb.org/anthology/P02-1040>.
- Pennington, J.; Socher, R.; and Manning, C. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543. Doha, Qatar: Association for Computational Linguistics. doi:10.3115/v1/D14-1162. URL <https://www.aclweb.org/anthology/D14-1162>.
- Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; and Sutskever, I. 2019. Language models are unsupervised multi-task learners. *OpenAI Blog* 1(8).
- Shibata, Y.; Kida, T.; Fukamachi, S.; Takeda, M.; Shinohara, A.; and Shinohara, T. 1999. Byte Pair Encoding: A Text Compression Scheme That Accelerates Pattern Matching. Technical Report DOI-TR-161, Department of Informatics, Kyushu University.
- Simonyan, K.; and Zisserman, A. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.
- Tan, H.; and Bansal, M. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 5099–5110. Hong Kong, China: Association for Computational Linguistics. doi:10.18653/v1/D19-1514. URL <https://www.aclweb.org/anthology/D19-1514>.
- Xu, K.; Ba, J.; Kiros, R.; Cho, K.; Courville, A.; Salakhudinov, R.; Zemel, R.; and Bengio, Y. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International Conference on Machine Learning*, 2048–2057.