VQA-BERT: Visual Question Answering Using Deep Bidirectional Transformer

CSC2511 - Final Report

Lan Xiao **Xuling Wang**
Department of Computer Science
University of Toronto
Toronto, ON M5S 2E4
{lxiao, shirlywang}@cs.toronto.edu

Abstract

We address the problem of Visual Question Answering (VQA), which requires joint visual and language understanding to answer a question about a given image. Previous works show that LSTMs and CNNs are powerful to capture the semantic meaning of questions. Recent work by Devlin et al. [2018] introduces a new language representation model, BERT, which obtains new state-of-the-art results on eleven natural language processing tasks. Our model uses BERT to learn question embedding and reason visual information with question-guided attention. Our model presents a modular improvement in language understanding in VQA models, resulting a better performance on VQA dataset from 63.12% to 64.31%.

1 Introduction

The task of Visual Question Answering (VQA) involves presenting an image to the model with a question in the form of natural language and asking the model to provide an answer in the form of short text, shown in Figure 1. With the recent advancement in computer vision (CV) and natural language processing (NLP), VQA has become an active research area.

What is on the coffee table ?
candles

What color is the hydrant ?
black and yellow

What is on the bed ?
books

What is the long stick for ?
whipping

Figure 1: Examples of training questions and their correct answer from the VQA v2 dataset [Antol et al., 2015]

A typical VQA workflow is depicted in Figure 2. Many recent works focus on elevating the role of image understanding in VQA. A vast majority of the models use image features pretrained for object recognition tasks using CNNs (e.g. GoogleNet, ResNet, VGG-Net, etc.). While we agree that computer vision is a critical component, we would like to focus on helping machine understand questions better and consolidate visual information with language understanding. Therefore, we will direct our effort into the word embedding, question modelling and feature integration components in the VQA workflow.

The recent work of Devlin et al. [2018], BERT, pushes the state-of-the-art of many natural language processing tasks by training transformers [Vaswani et al., 2017] on large text corpora using unsupervised learning. In this work, we incorporate BERT into the VQA workflow and achieve signicant perfromance improvement. At time of this writing, this is the first attempt to use BERT in VQA task.

The contributions of this papers are as follows:

- We introduce VQA-BERT, an end-to-end model that uses BERT to extract question embedding and question-guided attention mechanism to consolidate visual information with language understanding for the visual question answering task. Our code is available at https://github.com/BatBate/vqa-bert.
- We perform comprehensive evaluations on the VQA benchmarks, demonstrating that VQA-BERT outperforms RNN-based VQA models on similar configurations.
- We perform a detailed analysis where we visualize the outputs of different attention layers of VQA-BERT. We analyze the success and failure modes of the model, and compare it with RNN-based VQA models.

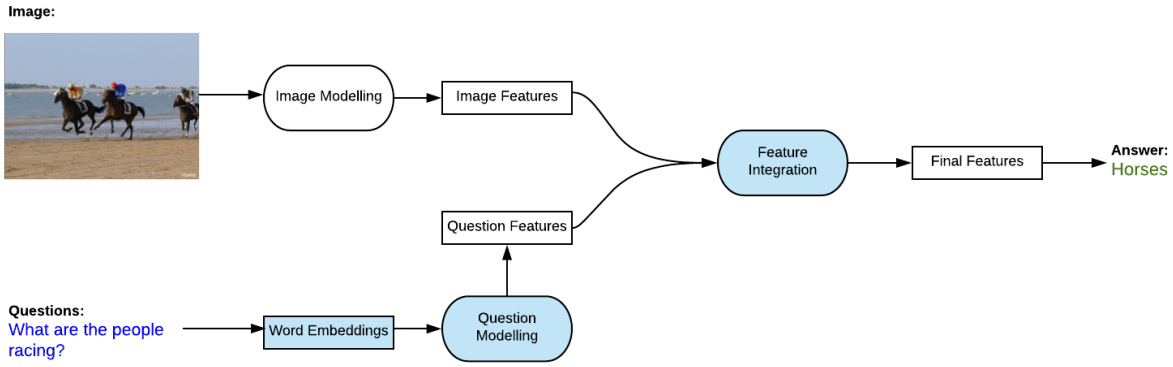


Figure 2: VQA Workflow

2 Background

To achieve our research objectives, we need to understand the current progress in both VQA and language modelling. We review papers in two relevant domains:

- High-performance VQA models with different sentence modelling and feature integration mechanisms
- Advancements in language modelling that capture the meaning of full sentences

2.1 VQA Models

2.1.1 Sentence Modelling

The majority of the VQA models use RNN-type architectures (e.g. LSTM, GRU) to encode either `Word2Vec` or `GloVe` word embeddings into question features [Wu et al., 2017]. Yang et al. [2016] explore both models for question representation in the VQA task. Their RNN based model use one-layer LSTM network to encode the question sequence. The final state of the encoder is taken as the representation vector for the question. For CNN, they use three convolution filters, which look at unigram, bigram and trigram respectively, to extract feature maps. They then apply max-pooling over the feature maps to form the feature representation vector for the question. The two models yield comparable performance while CNN is slightly more superior in terms of accuracy. We believe the two classes of methods are not substantially different at a fundamental level, tuning hyper-parameters like layer size is likely more important than picking the ideal architecture.

2.1.2 Feature Integration Mechanisms

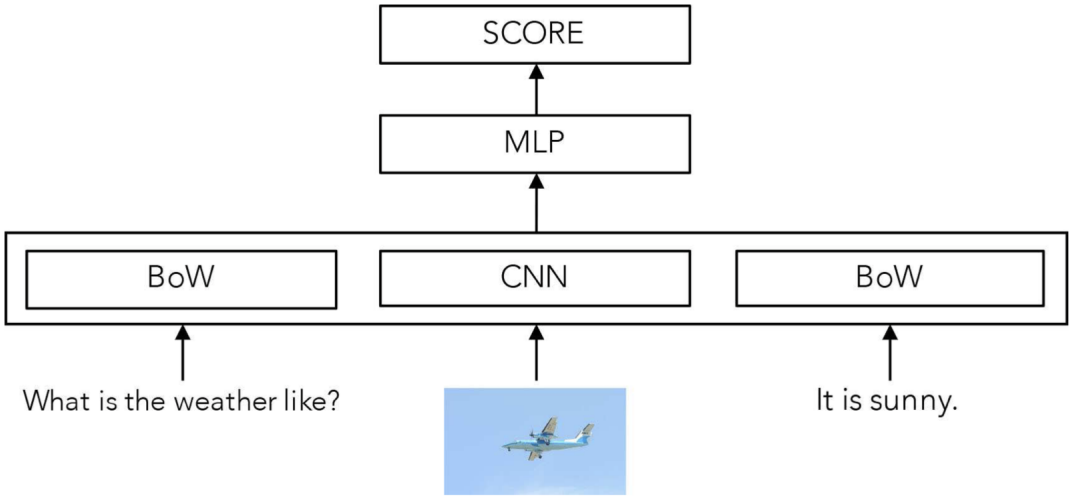


Figure 3: Overview of the Facebook Baseline Model

Simple Concatenation Most VQA models train a multi-class classifier on image and question features to predict an answer. Jabri et al. [2016] present a simple alternative model based on binary classification. Instead of training answers as competing choices, their model takes an image-question-answer triplet as input, and predicts whether or not such an triplet is correct. They further simplify the model by using a simple bag-of-words (BoW) model to represent the questions and answers instead of recurrent neural networks (RNN) with LSTM [Hochreiter and Schmidhuber, 1997] units. In particular, questions and answers are represented by averaging word2vec embedding [Mikolov et al., 2013b] over all words in the question or answer, respectively. The image features are extracted from the penultimate layer of the pre-trained ResNet-101 [He et al., 2016]. As shown in Figure 3, three feature vectors are concatenated and used to train a multi-layer perceptron (MLP) model with binary classification loss function.
The authors perform experiments to understand the strengths and weaknesses of the model. By incorporating answers as input instead of outputs, the VQA task is formulated as a binary classification problem. An alternative is to predict softmax probabilities over a discrete set of 5,000 most common answers. Binary classification model performs 4% bettern than multi-class classification model. This is because in binary model, BoW representation can easily exploit the similarities between different answers while softmax model has to learn them during training. The authors also shows that when training data is scarce, BoW model outperform LSTM model in representing questions and answers. Although data scarcity is not a big concern in

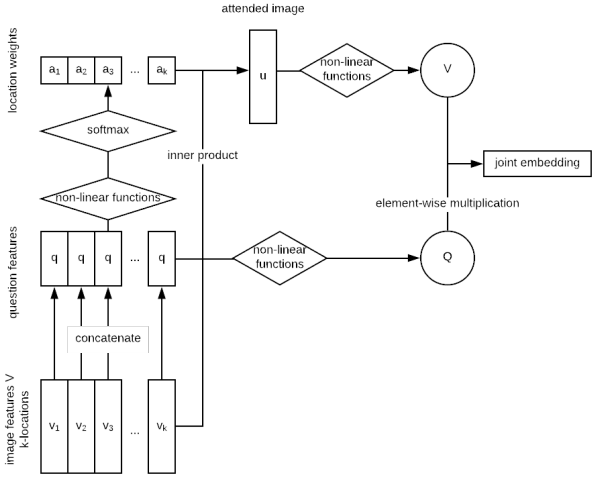
our project, it is worth noting that training a LSTM model requires more data when compared to a BoW model. Despite its simplicity, this model achieves comparable performance with state-of-the-art models. The Multimodal Compact Bilinear Pooling (MCB) Model, which will be analyzed in the subsequent section, outperforms this model whilst being substantially more complex.

Bayesian Approach Models Malinowski and Fritz [2014] use semantic segmentation to determine the position of objects in images. The model finds multiple image segmentation to represent different interpretations of the scene. It then uses a Bayesian formulation to marginalize these interpretations and computes the probability of different answers. However, it relies on the assumption of conditional independence, making it difficult to further boost performance. For this reason, we will not employ this framework in our project.

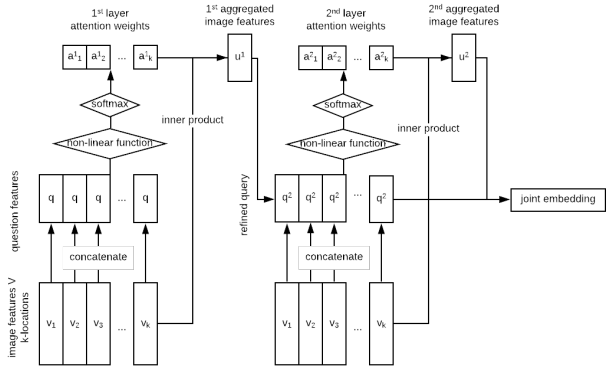
Attention Models To derive a joint embedding of question and image features, many VQA models today use attention mechanisms to focus on image and/or question regions that are relevant to the question-image pair [Wu et al., 2017]. In practice, attention mechanisms vary greatly by architecture. We perform in-depth reviews for four models that use different common attention mechanisms. A visualization of the mechanisms used can be found in Figure 4.

Single Attention

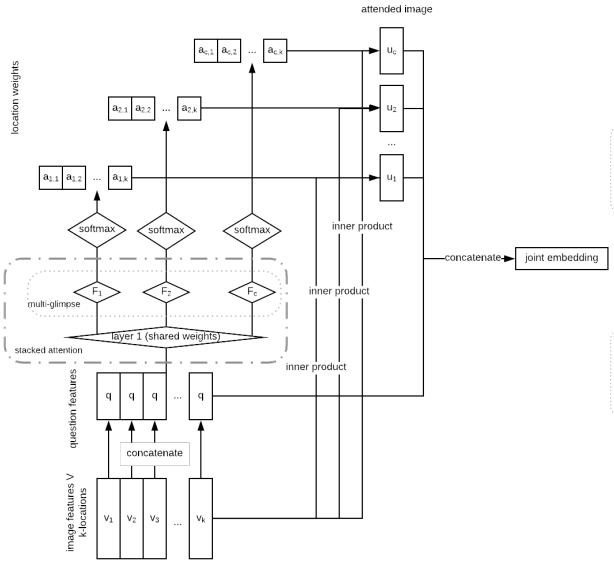
Teney et al. [2018] presents the 2017 VQA Challenge winner model. Their model uses a one-direction, one-glimpse spatial attention mechanism as depicted in Figure 4a to form joint question and image embeddings. Image features at each location $1, 2, ..., K$ are first concatenated with question features. The concatenated features are then passed through a non-linear layer with gated hyperbolic tangent activation. The outputs are normalized using a softmax function to form attention weights across K image locations. By taking the inner product of attention weights and original image features, attended image features are obtained. In this model, attended image features and original question features are each passed through some non-linear functions before element-wise multiplication to obtain the final joint embeddings.
In particular, the authors note that gated tanh activations produce better performance as compared to the more commonly used affine transformation followed by ReLU. They also report that the element-wise multiplication performs better than concatenation.



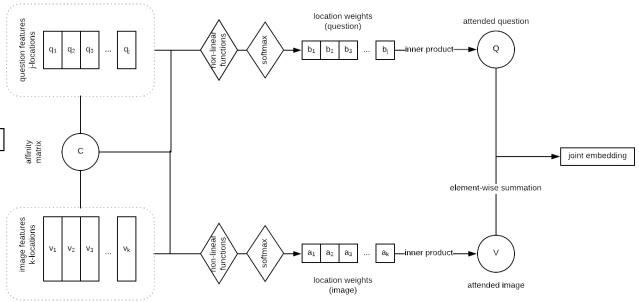
(a) Single Attention



(b) Stacked Attention, Single-Glimpse



(c) Stacked Attention, Multi-Glimpse



(d) Co-Attention

Figure 4: Common Attention Mechanisms

We suspect this simple attention mechanism does not fully account for the model's superior performance. On the other hand, a unique feature of this model that results in performance boost is related to how image features are generated in the first place. Instead of using mainstream features from pretrained CNNs, this model uses image features generated from bottom-up attention, jointly trained with ResNet and annotations from the Visual Genome dataset.

Yu Jiang* et al. [2018] built up upon this model and won the 2018 VQA Challenge. Major modifications include using Adamax optimizer, fine-tuning bottom-up image features, and

augmenting training data.

Stacked Attention, Single-Glimpse

Models with a single attention layer often fail to give precise answers when such answers are related to a set of fine-grained regions in an image. Yang et al. [2016] argue that this is because VQA often requires multiple steps of reasoning. Hence, a multi-layer stacked attention networks (SAN) in which an image is queried multiple times to infer the answer progressively is naturally suited for solving VQA tasks.

The overall architecture of SAN is illustrated in Figure 4b. It consists of three main parts: (1) the image model, which uses standard CNN to extract image features; (2) the question model, which uses either a CNN or a LSTM to extract semantic features; (3) the stacked attention model, which sequentially locates the image regions that are relevant to answering the question. Image features are extracted from the last pooling layer of CNN, which preserves spatial information of the original images. Then, the question feature vector is concatenated to each spatial location of the image feature matrix and pass to the stacked attention model. Softmax function is used to generate the attention distribution $\mathbf{a}^1$ over the regions of the image. Based on $\mathbf{a}^1$, the author calculate the aggregated image feature vector $\mathbf{u}^1$, which is later combined with the question feature to output a refined query $\mathbf{q}^2$. This process is reiterated with an additional attention layer. The second aggregated image feature vector $\mathbf{u}^2$ is combined with the refined query $\mathbf{q}^2$ to form the new query vector. This new query vector is passed to the classification model for answer prediction. Figure 5 shows an example of how the reasoning process works. In the first attention layer, the model find coarse areas in the image that are related to the question `"What are sitting in the basket on a bicycle?"` The second attention layer pinpoints the region that corresponds to the answer `"dog"`.

In their experiments, the two-layer SAN gives the best testing accuracy on the VQA dataset. CNN-based question model gives slightly better performance than LSTM-based question model, as discussed in Section 2.1.1. It is worth noting that the author only compare SAN with basic VQA models with modest performance. Hence, we find the effectiveness of SAN questionable.

Figure 5: Visualization of the learned multiple attention layers [Yang et al., 2016]

Stacked Attention, Multi-Glimpse

Kazemi and Elqursh [2017] build on the idea of SAN but incorporate multiple sets of attention weights (a.k.a. multiple glimpses) in the second attention layer. This change, combined with other model configurations, deliver a strong baseline model whose performance is comparable to 2016 VQA winning models on VQA 2.0. As demonstrated in Figure 4c, parameters in the second attention layers are randomly initialized to produce diverse attention distributions. However, in contrast to Yang et al. [2016], the authors do not notice significant performance boost from using stacked attention.

Besides presenting a multi-glimpse attention mechanism, this paper is also valuable to our project because the authors experiment different setup regarding regularization, word embedding, LSTM question model, and classifier. They notice that applying l2-regularization on ResNet features, having dropout, using soft-attention and a two-layer classifier produce the best model.

Hierarchical Co-Attention Model

All previous attention mechanisms use questions to guide the algorithm to focus on particular regions of the images. However, they do not show what parts of the questions are important. Lu et al. [2016] propose a model that uses image representation to guide the question attention and the question representation to guide the image attention. They call this mechanism 'co-attention'. As illustrated in Figure 4d, an affinity matrix C is calculated from original image

and question features by

$$C = tanh(Q^T W_b V)$$

where W_b contains the weights, Q contains questions features and V contains image features. This affinity matrix connects question attention space and image attention space and is used as an integrated feature to generate two attention weights in parallel. Inner product of attention weights and the corresponding original features gives attended features. Joint embedding is an element-wise summation of attended image and question features. The paper also presents an alternating co-attention mechanism but the parallel mechanism performs better.

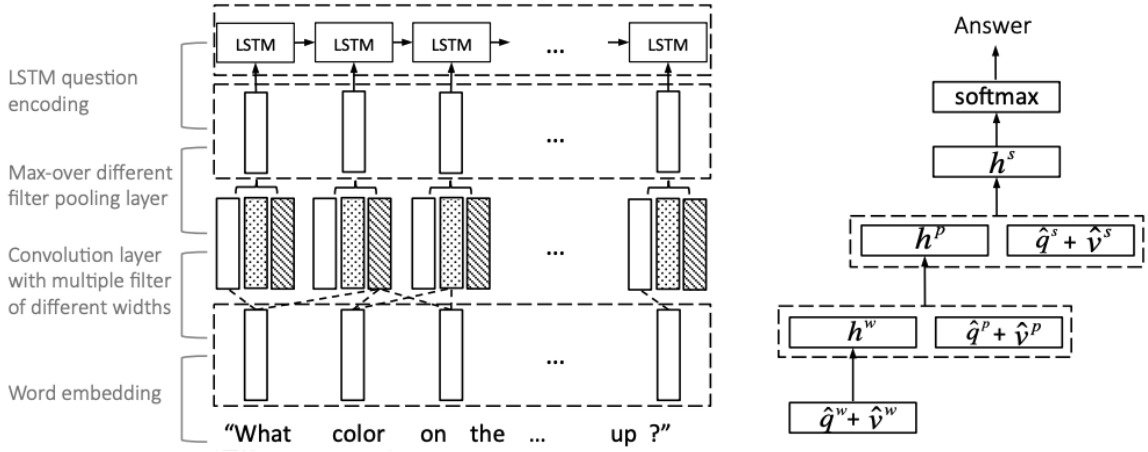
While "what words to listen to" is important, the proposed question attention only achieves marginal improvements as shown in the paper's ablation study. We think a potential source for the failing is that image features may not be the best guide for question attention. Instead, better language modelling may address this issue more effectively.

Another innovation that this paper presents is a hierarchical representation of the question features, as shown in Figure 6a. There are three levels of representations:

- Word-level: word embeddings
- Phrase-level: output from max-pooling across unigram, bigrams, and trigrams at each word location
- Question-level: LSTM hidden vector that encodes phrase-level features

The co-attention mechanism is applied to each level to obtain attended word-level, phrase-level, and question-level co-attended features. As illustrated in Figure 6b, these features are used recursively in a multi-layer perceptron to predict the final answer. However, from ablation study in the paper, word-level and phrase-level attention provide little improvement. We think a hierarchical representation sounds promising, but this paper may not present the best implementation.

Multimodal Compact Bilinear Pooling Unlike an explicit attention mechanism where attention weights are obtained, Fukui et al. [2016] rely on Multimodal Compact Bilinear pooling (MCB) to obtain joint embedding. MCB uses a convolution on Count Stretches of image and question features so that 1) interactions between features are captured more expressively than concatenation, and 2) higher dimensional representation and explicit computation of an outer product are avoided. Figure 7 illustrates the MCB process. Ψ is the Count Sketch projection function.



(a) Hierarchical Representation (b) MLP Classification

Figure 6: Question Hierarchy[Lu et al., 2016]

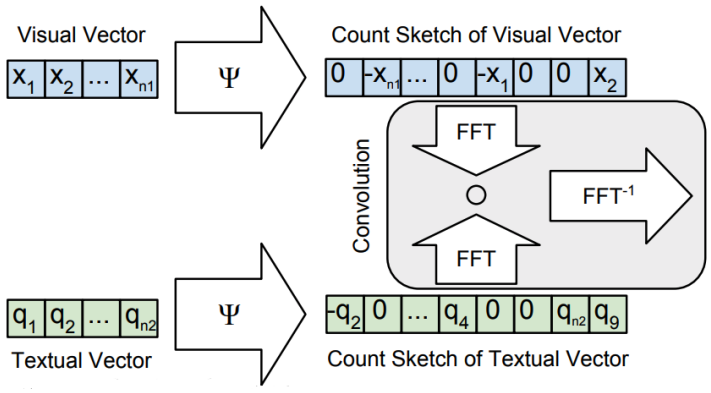


Figure 7: Multimodal Compact Bilinear Pooling [Fukui et al., 2016]

It is worth mentioning that in deriving their final model, the authors add two attention maps to incorporate spatial information and find it beneficial to model performance. They also augment training data with image-question-answer triplets from the Visual Genome dataset and concatenate learned word embedding with GloVe vectors. These configurations win the 2016 VQA Challenge.

2.2 Language Modelling

In this section, we review several notable recent advancement in language modelling.

2.2.1 Word Embedding

Learning a vector space representation of words has been proven successful in capturing fine-grained semantic and syntactic structures. Two main classes of word embedding model include global matrix factorization and local context window methods. The Skip-gram model (also

known as word2vec model) proposed by Mikolov et al. [2013b] is one of the most popular local context window methods that achieves good performance in analogical reasoning task [Mikolov et al., 2013a] and many other downstream NLP tasks such as named entity recognition. An alternative approach proposed by Pennington et al. [2014] combines the advantages of both models, resulting in a new global log-bilinear regression model called GloVe. In this subsection, we analyze and compare the two models. Figure 8 shows the overall performance

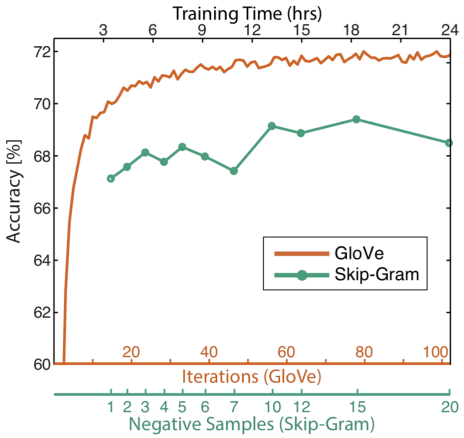


Figure 8: GloVe vs Skip-Gram [Pennington et al., 2014]

on the analogy task as a function of training time. The two x-axes at the bottom indicate the corresponding number of training iterations for GloVe and negative samples for word2vec. The results are reported by training on the 6B token Wikipedia 2014 + Gigaword 5 corpus. GloVe embedding achieves better performance on this corpus. However, while GloVe embedding works better on some corpus, word2vec embedding works better on others. Overall, when we control for all the training hyper-parameters, these two word embedding methods tend to have similar performance in the the analogy task and other downstream tasks [Pennington et al., 2014]. Both models probe the underlying co-occurence statistics of the corpus, but the GloVe model has the advantage of capturing global statistics. One additional advantage of GloVe over word2vec is that the implementation for GloVe is naturally suited for parallelization which takes better advantage of the great computing power of modern GPU and is easier to train with more data.

These word embeddings have already been incorporated in recent high-performing VQA models.

2.2.2 Transformer

Vaswani et al. [2017] propose a self-attention-based encoder-decoder architecture, the Transformer, and outperform other models in benchmark translation tasks. Since no recurrent neural network is involved, this architecture allows for better parallel computation.

Figure 9 illustrates the architecture. The encoder is composed of a stack of N identical layers. Each layer has two sub-layers. The first is a multi-head self-attention mechanism, and the second layer is a position-wise fully connected feed-forward network. To facilitate training, residual connection is used for all layers, followed by layer normalization [Lei Ba et al., 2016]. The decoder is constructed similarly except that 1) a third sub-layer is added to perform multi-head attention over encoder output, and 2) positions in sub-layers do not attend to subsequent positions.

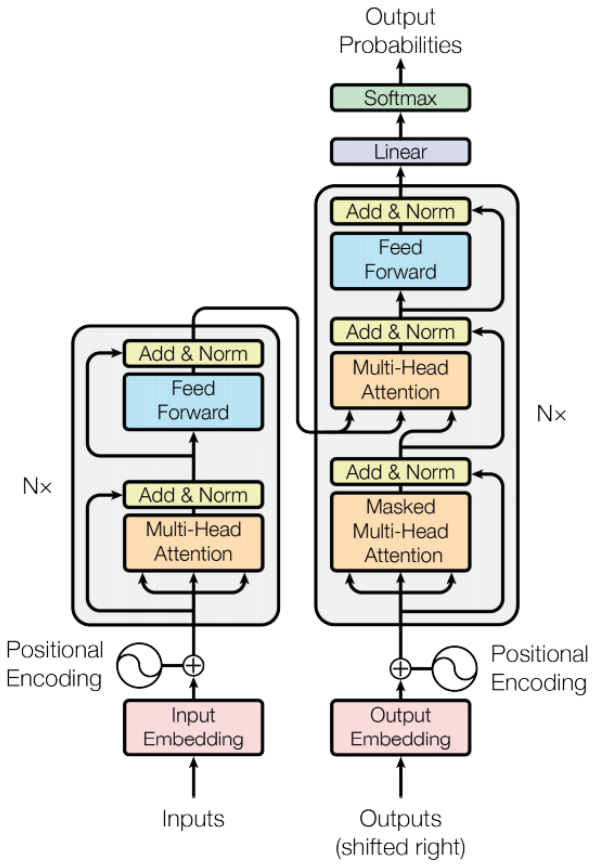


Figure 9: The Transformer Architecture [Vaswani et al., 2017]

The Transformer architecture innovates the use of attention in two ways:

Scaled Dot-Product Attention Given a query Q, key K (of dimension d_k) and value V (of dimension d_v), the attention is calculated as:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_k}}V$$

Multiplicative attetion is used because it is more time- and space-efficient and scaling is used to prevent softmax function from pushing values into regions where there are extremely small gradients.

Multi-Headed Attention In stead of performing a single attention, the Tranformer performs multiple attention on different linear transformations of each query, key and value. Each transformation is called a head. Individual attention outputs are concatenated and projected, resulting in the final multi-headed attention output. The process can be described as:

$$\begin{aligned} MultiHead(Q, K, V) &= Concat(head1, ..., head_h)W^O, \\ &where\ head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \\ &and\ W_i^Q, W_i^K, W_i^V are\ projection\ matrices \end{aligned}$$

However, since most VQA models frame answer prediction as a classification rather than text synthesis problem, they are not compatible to a encoder-decoder architecture.

2.2.3 Bidirectional Encoder Representations from Transformers (BERT)

Building upon the sucess of the Transformer, another recent advancement in language representation is BERT, which stands for **B**idirectional **E**ncoder **R**epresentation from **T**ransformers. [Devlin et al., 2018]. BERT's model architecture is a multi-layer bidirectional Transformer encoder, as described in the previous section.

BERT uses the original Transformer implementation, which allows it to pre-train deep bidirectional representations by jointly conditioning on both left and right context in all layers. This means that in BERT each token would have self-attention with every other token in the sentence. Specifically, given an input sequence $(x_1, \ldots, x_n)$, the input representation is a sum of the corresponding token, segment and position embeddings. Specifically, BERT uses WordPiece embeddings [Wu et al., 2016] with a 30,000 token vocabulary. Rare words are split with ##. Unlike the original Transformer implementation, the positional embeddings are learned

with maximum supported sequence lengths of 512 tokens. Each sequence is padded with a special classification embedding (`[CLS]`). The final hidden state corresponding to this token is used as the aggregate sequence representation for downstream tasks. The original BERT implementation supports unambiguous representation for both a single sentence or a pair of sentences (e.g., [Question, Answers]).

To pre-train BERT on a language modelling task, the authors introduce two novel unsupervised prediction tasks:

Masked Language Model (MLM) Some of the input tokens are randomly masked and the model objective is to predict the original tokens based on its context.

Next Sentence Prediction Two sentences, `A` and `B`, are fed into BERT and the model is asked to identify whether `B` follows `A`

In addition, the authors demonstrated how to fine-tune BERT on eleven natural language processing tasks. We found single sentence classification task the most applicable to VQA language modelling. The final hidden state output corresponding to `[CLS]` is used as inputs to the classifier. Figure 10 demonstrates the process.

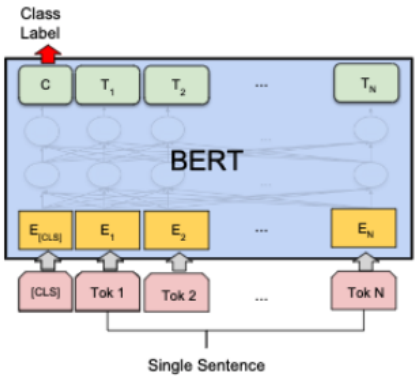


Figure 10: Single Sentence Classification Using BERT [Devlin et al., 2018]

3 Data

3.1 VQA v2.0 dataset

We intend to train and evalutate our model using the VQA v2.0 Balanced Real Images dataset [Goyal et al., 2017]. It is the largest dataset for visual question answering problem, contain-

ing human annotated questions and answers on Microsoft COCO dataset [Lin et al., 2014]. It contains 443,757 training questions, 214,354 validation questions and 447,793 testing questions, and a total of 1,105,904 question-answers pairs. The questions are grouped into three subcategories based on answer types: YES/NO, Number, and Other. Each question has ten human-annotated answers.

3.2 Bottom-up Image Features

We use pretrained image features from Anderson et al. [2018][1]. These features were generated from bottom-up attention, jointly trained with ResNet and annotations from the Visual Genome dataset. Anderson et al. [2018] contains detailed implementation of how the features were generated. We do not intend to modify these image features for the purpose of this project.

3.3 Evaluation

Submissions for the VQA Challenge are evaluated using the following accuracy metric:

$$accuracy = min(\frac{\#humans\ that\ provided\ that\ answer}{3}, 1)$$

That is, an answer is 100% accurate if at least 3 humans provided that exact answer. This accounts for disagreements between human annotators and ground truth. We will use the same metric to evaluate our models.

Since annotations for test set are not available, we will report model accuracy based on validation set. While this is not an ideal practice, prior works rarely observe overfitting and resort to reporting validation performance while test answers are not available [Kazemi and Elqursh, 2017].

3.4 Corpus Analysis

3.4.1 Words Frequencies in Questions

We analyze the most common words appearing in VQA questions from both training and validation set. Figure 11 shows the most common words after lemmatization using Spacy. Besides common English words like 'the', 'a', 'of', question-related words such as 'what', 'how', 'where' and words describing picture scenes like 'picture', 'color' also appear in the list.

[1] https://github.com/peteanderson80/bottom-up-attention

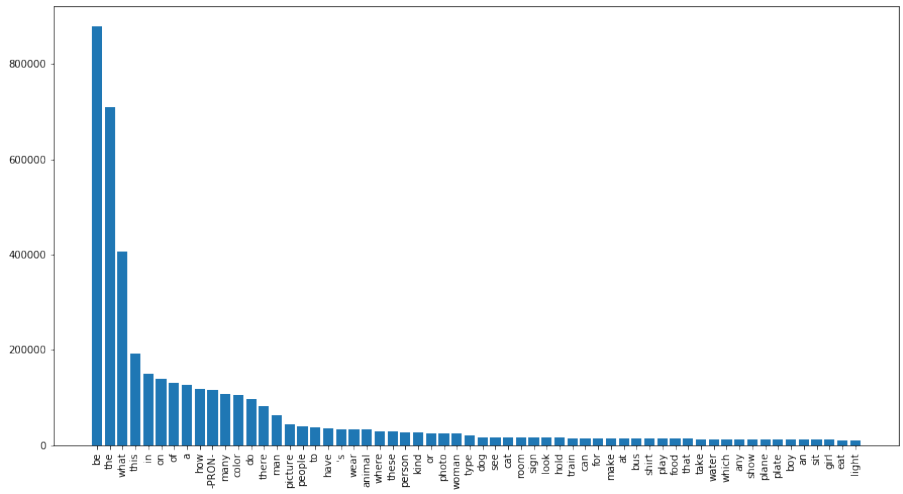


Figure 11: Most Frequent Words in Questions

VQA question corpus is also much smaller and less sparse, as compared to large NLP corpus. There are 11,521 unique words. Looking at both Frequency of Frequencies table and plot of log scaled rank against log scaled frequency (refer to Figure 12), question corpus has comparatively fewer Hapax legomena and more high-frequency words.

Frequency	Number of Words
1	830
2	1710
3	963
4	1192
5	393
6-10	1587
11-50	2456
51-100	682
>100	1708

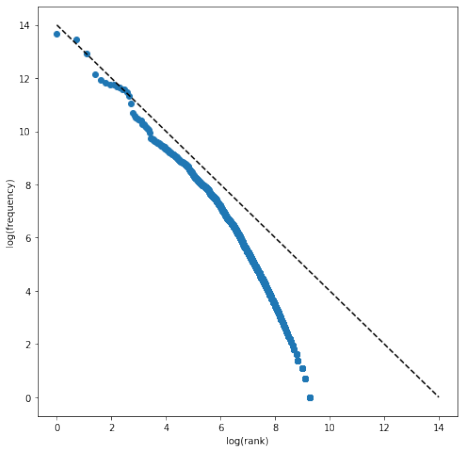
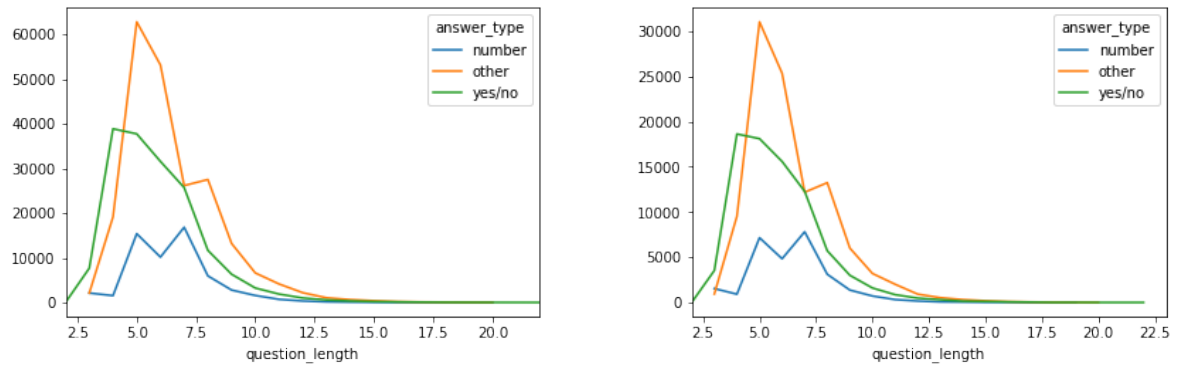


Figure 12: Frequency of Frequencies and Zipf's Law

3.4.2 Question Length

Figure 13 summarizes question lengths by answer types in both training and validation set. The dataset contains mostly short sentences with length of about six words. In addition, questions with binary answers tend to be shorter than the other types of questions.



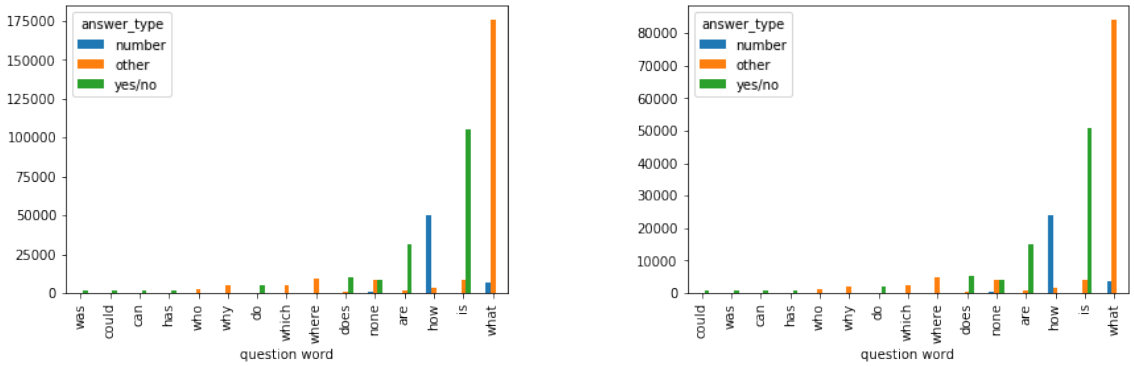
(a) Question lengths in training set

(b) Question lengths in validation set

Figure 13: Question Lengths

3.4.3 Question Key Words

A similar analysis is conducted with respect to question words. We define a question word as the first word of a question. If the question does not start with a proper question word, it is assigned 'none'. Figure 14 shows that the most common question words for Yes/No, Number and Other questions are 'is', 'how (many)', and 'what' respectively. It also demonstrates that many questions are descriptive (e.g. using the word 'what') rather than inferential (e.g. using the word 'why').



(a) Question words in training set

(b) Question words in validation set

Figure 14: Question Words

3.4.4 Answer Vocabulary

Lastly, we analyze VQA answer vocabulary for each answer type. Note that the answer type is provided as part of the dataset and there are rare mis-classifications. As shown in Figure 15, over 90% of the answers contain a single word.

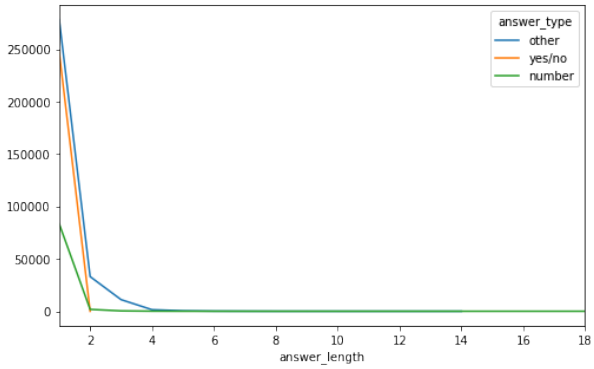


Figure 15: Answer Lengths

One of the key differences between VQA v2.0 dataset and VQA v1.0 dataset is the use of balanced yes/no questions. Our analysis confirms this (refer to a frequency plot in Figure 16). Some questions are mis-classified as yes/no questions but they occur rarely.

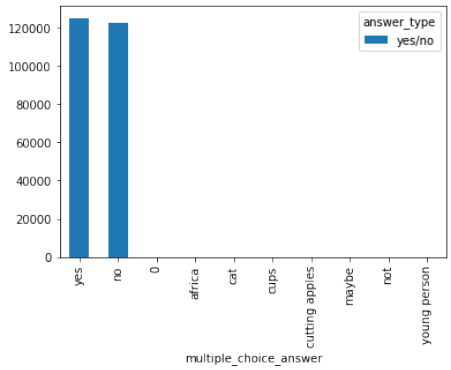


Figure 16: Yes/no Answers

Common answers and their frequencies for Number questions are shown in Figure 17. The top 35 common answers account for over 90% of all answers. In particular, most numerical questions involve counting fewer than ten objects.

More answer words appear in Other questions. The top 60 common answers account for less than 50% of all answers in this category and their frequencies are plotted in Figure 18. Other than color-related answers, we do not observe concentrations of specific groups of answers.

4 Methods

We present `VQA-BERT` and its implementation details in this section. The complete model is summarized in Fig. 4a. As a high-level overview, the model implements a joint embedding of

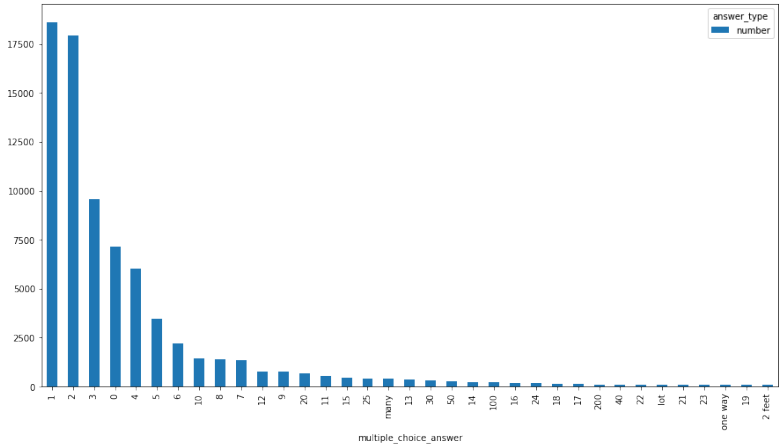


Figure 17: Common Answers in Number Questions

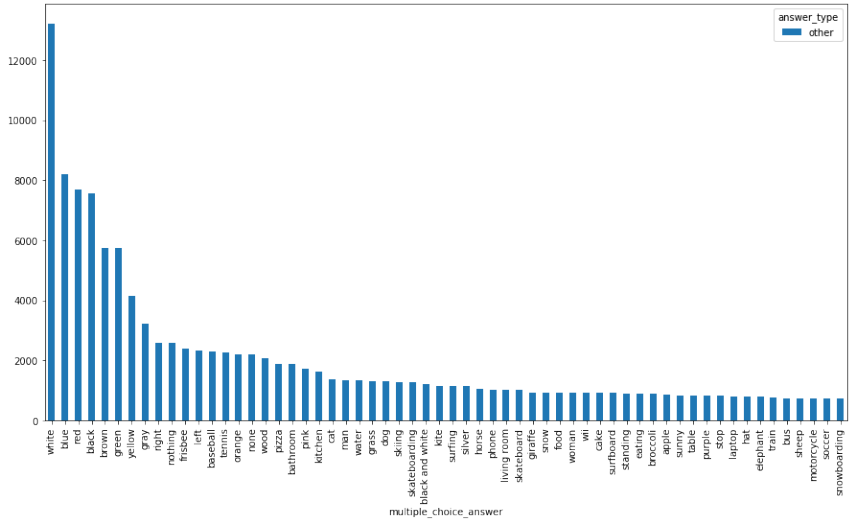


Figure 18: Common Answers in Other Questions

the input questions and images, followed by a multi-class classifier over a fixed set of candidate answers. The model uses the bottom-up image features which has been carefully described in Section 3.2. `VQA-BERT` is build on `Pythia` [Jiang et al., 2018, Yu Jiang* et al., 2018]. In the rest of this section, we will focus on the details of question embedding and question-guided attention over images.

4.1 Question Embedding

For the VQA dataset, the input question are generally short sentences with maximum sentence length of 23 words. Since 99.75% of questions in the VQA v2.0 dataset are shorter than 14 words, for computational efficiency, we trimmed the questions to have a maximum sentence length of 14 words. The extra words are discarded. Questions shorter than 14 words are end-

padded with zero vectors, which are frozen during training. After trimming, the questions are tokenized using WordPiece embeddings. Numbers and number-based words (e.g. 1,000 or 2:15pm) are also considered as words.

We use BERT [Devlin et al., 2018] pre-trained on the concatenation of BooksCorpus (800M words) [Zhu et al., 2015] and English Wikipedia (2,500M words). For computational efficiency, we choose `bert-base-uncased` as the backbone model. Uncased means that the text has been lowercased before processing as our task is not sensitive to case information. This model consists L = 12 layers of transformer blocks, the hidden size for the self-attention sublayer is H = 768, and the number of self-attention head is A = 12. The feed-forward network takes input tensor of size H and output tensor of size 4H, in this case 3072. This intermediate layer is compressed and used as input to the next transformer layer. The total number of parameters in `bert-base-uncased` is 110M. Fine-tuning BERT for the VQA task follows the same procedure for fine-tuning sequence-level classification proposed in the original paper. To formulate a fixed-dimensional pooled representation of the questions, we use the output (i.e., the last hidden state) of the Transformer for the first token in the input, which by construction corresponds to a special `[CLS]` token. we apply L2 normalization to the output token and pass it through a linear layer to get a 2048 dimensional question embedding

Parameters of BERT are fine-tuned jointly with the rest of the model to maximize the log probability of the correct answer. A dropout layer with 0.1 dropout probability is applied after all layers. Note that the last hidden state of `[CLS]` alone is not a good sentence representation because the pre-trained model is not fine-tuned on any downstream tasks. Therefore, we need to fine-tune the entire model in a end-to-end fashion.

4.2 Question-guided Image Attention

We uses a one-glimpse, unidirectional attention mechanism to guide the model which part of the image to focus on according to question embeddings. Let $\mathbf{v}$ represent the extracted image feature with K spatial locations, and let $\mathbf{q}$ represent the question embeddings. Each image vector $\mathbf{v}_i, i = 1, ..., K$ is passed through a convolution layer with 5000 size 1 kernels followed by weight normalization layer and ReLU activation to get $\tilde{\mathbf{v}}_i$. The question embedding $\mathbf{q}$ is passed through a linear layer followed by weight normalization and ReLU activation to get a

5000-dimensional vector $\tilde{\mathbf{q}}$.

$$\tilde{\mathbf{q}} = \max(0, \mathbf{W}\mathbf{q} + \mathbf{b}) \tag{1}$$

$$\tilde{\mathbf{v}} = \max(0, \mathbf{h} * \mathbf{v}) \tag{2}$$

For each location $i = 1 \ldots K$ in the image, the feature vector $\tilde{\mathbf{v}}_i$ is element-wise multiplied with the question embedding $\tilde{\mathbf{q}}$ to obtain a scalar attention score a_i. The attention weights are normalized over all locations with a softmax function to obtain the final attention weight α_i.

$$\mathbf{a} = \mathbf{w_a}(\tilde{\mathbf{q}} \circ \tilde{\mathbf{v}}) \tag{3}$$

$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{a}) \tag{4}$$

The image features from all locations are then weighted by the normalized values and summed over K spatial dimension to obtain a 1×5000 vector $\hat{\mathbf{v}}$ representing the attended image features.

$$\hat{\mathbf{v}} = \boldsymbol{\alpha} \cdot \tilde{\mathbf{v}} \tag{5}$$

4.2.1 Multimodal fusion

The transformed question embedding $\tilde{\mathbf{q}}$ and the attended image embedding $\hat{\mathbf{v}}$ are combined using element-wise multiplication to form the joint embedding $\mathbf{h}$, which is later fed to the output classifier. Formally,

$$\mathbf{h} = \tilde{\mathbf{q}} \circ \hat{\mathbf{v}} \tag{6}$$

4.3 Output classifier

We frame the question answering problem as a multi-class classification problem, where the classes come from a predetermined set of candidate answers. We consider only answers in the training set that appear more than 8 times, resulting N = 3129 candidate answers. In this case, 7% training questions have no correct answer within the selected candidate answers. We do not discard these questions because they provide a useful training signal.

The joint embedding $\mathbf{h}$ is passed through a fully connected layer with weight normalization and sigmoid activation to predict a score s for each of the N candidate answers:

$$\hat{s} = \sigma(\mathbf{w_o}\mathbf{h}) \tag{7}$$

The sigmoid normalizes the final scores to fall between (0, 1), which are followed by a loss similar to binary cross-entropy, but with soft target scores. Formally, the objective function is

$$L = -\sum_{i}^{M}\sum_{j}^{N} s_{ij}\log(\hat{s}_{ij}) - (1 - s_{ij})\log(1 - \hat{s}_{ij}) \tag{8}$$

where the indices i iterates over M training questions and j iterates over N candidate answers. For VQA v2.0 dataset, each training question is associated with 10 human annotated answers, each labeled with soft accuracy between [0, 1]. s_{ij} are the confidence scores of these ground truth answers. The above formulation is proved to be much more effective than a softmax classifier as used in most other VQA models [Teney et al., 2018]. This is because the sigmoid outputs allow optimization for multiple correct answers. Also, using soft scores as targets provides richer training signal as they capture the potential uncertainties in ground truth annotations.

4.4 Training the Model

We optimize this model with Adamax optimizer, a variant of Adam with infinite norm [Kingma and Ba, 2014]. The learning rate is set to 0.002 with a batch size of 128. Therefore, we begin with a learning rate of 0.002, linearly increase it at each iteration until it reaches 0.01 at iteration 1000. Then, we gradually decrease the learning rate by a factor of 0.1 at every 5K iterations, until reaching 25K, and stop training at 36K. For fine-tuning BERT, we notice that the model does not converge if we jointly optimize BERT's parameters with the same schedule. Therefore, we optimize BERT's parameters separately using Adam with learning rate of 1e-5, $\beta_1 = 0.9, \beta_2 = 0.999$, L2 weight decay of 0.01, learning rate warm-up over the first 10,000 steps, and linear decay of the learning rate. We use a dropout probability of 0.1 on all layers.

5 Results

Table 1 summarizes our results on the validation set. We report both overall validation score and validation scores by answer types. Standard deviation of the scores is calculated by taking 1,000 bootstrapped samples of size 10,000 from the validation set. Incorporating BERT improves model performance for all answer types and the improvement in yes/no questions is the largest.

Model	Overall	Yes/No	Number	Other
Baseline	63.12±0.44	80.51 ± 0.36	42.09 ± 0.47	55.40 ± 0.46
VQA-BERT	64.31±0.44	82.16 ± 0.35	43.38 ± 0.46	56.30 ± 0.46

Table 1: Summary of Results

5.1 Detailed Comparisons

To understand which questions and/or answers that our models are good at, we preform a detailed break-down of the validation scores.

5.1.1 Question Words and Question Lengths

Figure 19 and 20 summarize model performance by question words. The blue and orange lines indicate model validation score and the bars indicate number of questions by answer types (i.e. yes/no, number, other). Consistent to general results reported in the previous section, both `VQA-BERT` and baseline model perform better when there are more yes/no questions and worse when there are more number questions. In addition, models perform worse when questions are longer and require inference (i.e. using question words like 'why'). These harder questions also appear less frequently in both training and validation set.

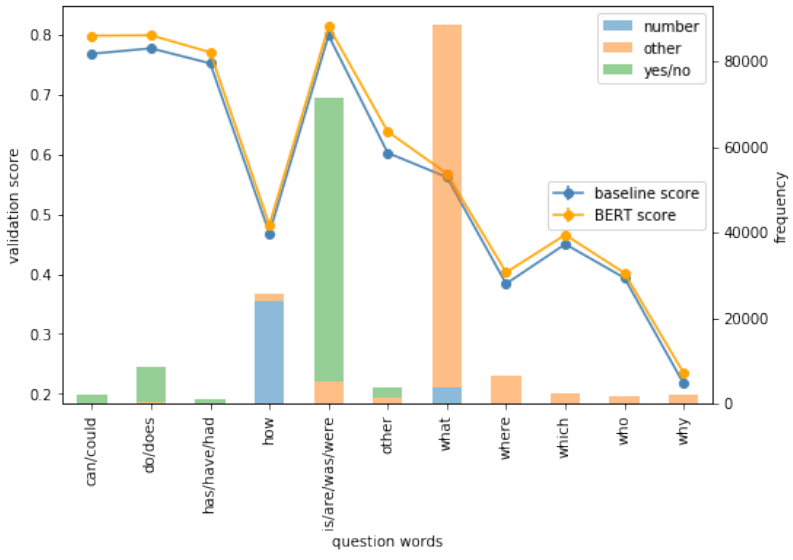


Figure 19: Validation Score by Question Words

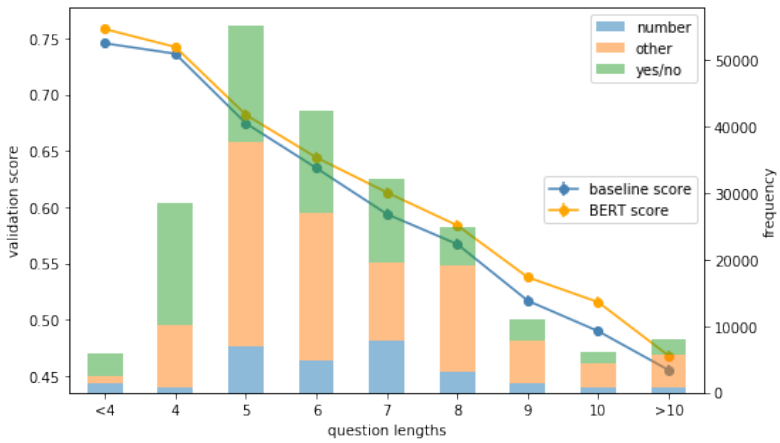


Figure 20: Validation Score by Question Lengths

While VQA-BERT improves validation scores across nearly all question words and question lengths, the most noticeable improvements come from infrequent question words such as 'can/could', 'do/does', 'where' and 'which' and infrequent question lengths. This suggests that BERT is more capable of capturing sentence meaning without relying too much on specific question styles.

5.1.2 Answer Vocabulary

We analyze model performance with respect to the correct answers' lengths and we show the results in Figure 21. Longer answers are mostly from non-binary questions, appear less often in our dataset, and have lower prediction score. VQA-BERT does not appear to solve this challenge.

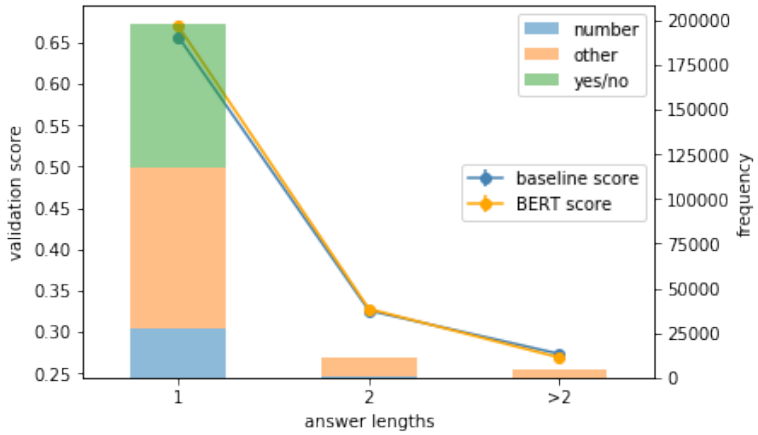
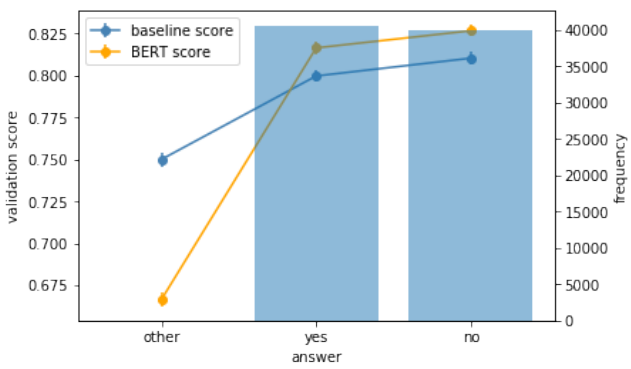


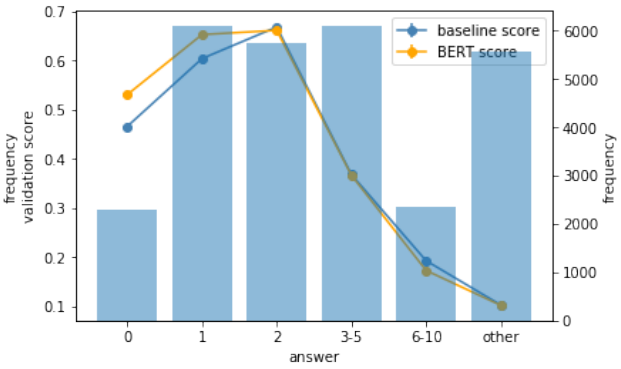
Figure 21: Validation Score by Answer Lengths

We also evaluate our model performance on most frequent answer words by answer types. As shown in Figure 22b, our model needs improvement at counting large number of objects, and

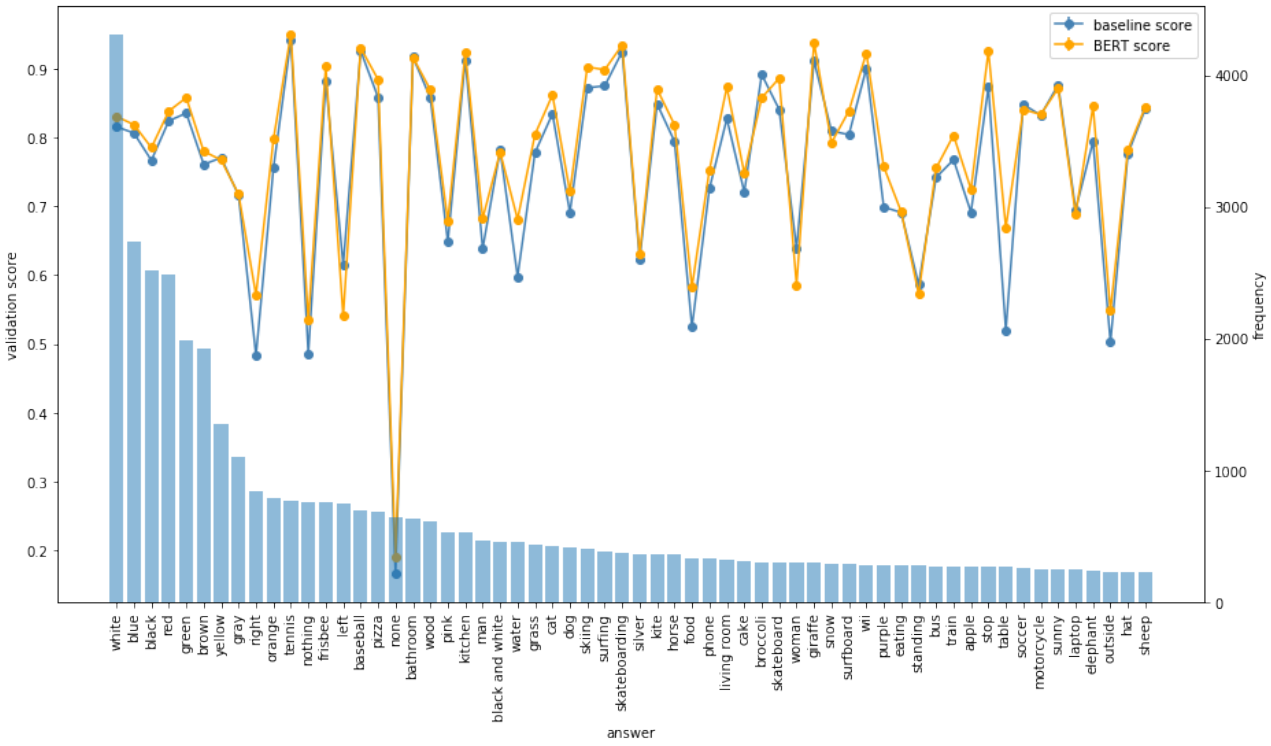
answering Number questions that do not involve counting objects such as reading time from a clock. Figure 22c shows a relatively consistent performance across frequent answers for Other questions, except for the answer key 'none'. A 'none' answer is assigned if majority of the human annotators do not provide answers for this question. Since the number of ground-truth answers for such questions are reduced, the model score will be negative affected.



(a) Yes/no answers



(b) Number answers



(c) Other answers

Figure 22: Baseline Validation Score by Frequent Answers

Unlike in the previous section where questions are categorized based on question styles, `VQA-BERT` does not appear to significantly improve performance for specific groups of answers. This is also reasonable as answer words are not captured in BERT, and a better understanding

of questions should in theory help the final classifier to pick any reasonable answers. Therefore, the impact of using BERT on specific answer words would be less direct.

5.2 Ablation Study

Language modeling is used for two purposes in our VQA model architecture: 1) to generate question-guided image attention, and 2) to be combined with attended image features for the final classification task. To understand why BERT improves model performance, we conduct an ablation study where BERT was only used in one of the two areas. Table 2 summarizes the results.

Model	Overall	Yes/No	Number	Other
Baseline	63.12±0.44	80.51 ± 0.36	42.09 ± 0.47	55.40 ± 0.46
VQA-BERT (Attention)	58.24±0.48	72.84 ± 0.41	38.76 ± 0.44	52.28 ± 0.46
VQA-BERT (Classification)	61.88±0.46	80.18 ± 0.38	42.08 ± 0.45	53.16 ± 0.46
VQA-BERT	64.31±0.44	82.16 ± 0.35	43.38 ± 0.46	56.30 ± 0.46

Table 2: Ablation Studies

Results for using BERT in either generating attended image features or classification alone do not outperform baseline model. This is likely due to higher model complexity from using both GRU and BERT. We notice that model results have not fully converged when reaching the default maximum number of iterations. Due to computational resource and time constraint, we do not proceed with more iterations.

5.3 Case Studies

This subsection shows examples of failure and success modes of `VQA-BERT`. We use images from the validation set. We explore how `VQA-BERT` improves performance with respect to longer questions and questions with infrequent words, such as 'can/could', 'do/does', 'where' and 'which' questions. We also present some failure cases at counting large number of objects and inference questions.

(a) Which giraffe is taller, the one standing on the left or the one standing on the right? (17 words)
Baseline: right, VQA-BERT: left

(b) Does this photo show a place where a pedestrian should be able to safely cross the street? (17 words)
Baseline: No, VQA-BERT: Yes

(c) What colors do the chair, the rug and cat's fur have in common? (15 words)
Baseline: black and white, Bert: brown

(d) Is there a food in this picture that would taste great smothered in strawberries? (14 words)
Baseline: yes, VQA-BERT: no

Figure 23: Success cases answering long questions where VQA-BERT predicts correct answers while baseline model predict wrong answers

(a) Where is the girl standing on a skateboard?
Baseline: floor, VQA-BERT: living room

(b) Which dessert tower is taller?
Baseline: cake, VQA-BERT: left

(c) Could they lose their equipment over the fence?
Baseline: no, VQA-Bert: yes

(d) Does the bear have anything to play with?
Baseline: yes, VQA-BERT: no

Figure 24: Success cases answering infrequent questions where VQA-BERT predicts correct answers while baseline model predict wrong answers

(a) How many windows on the side of the bus' second level?
VQA-BERT: 8, True: 6

(b) How many people are in the scene?
VQA-BERT: 8, True: 18

(c) Why would it be hard for the sheep to engage in their regular eating habits?
VQA-Bert: yes, True: no grass

(d) Why did the girl with curly hair chose to have a picnic in the shade?
VQA-BERT: party, True: to avoid sun

Figure 25: Failure modes for our model at counting large number of objects and inference questions

6 Why Are Some Questions Hard to Answer

In this section, we provide both quantitative and qualitative analysis on why certain questions are answered incorrectly by `VQA-BERT`.

6.1 Infrequent/Novel Answers

For each question in the validation set, we compute an *answer frequency score* as the natural log of average appearances of each human-annotated answer in the training ground-truth answers. For example, if human-annotated answers are 5 'white', and 5 'black', and 'white' appears 5,000 times and 'black' appears 3,000 times in training answers, the answer frequency score for this question is $log(5000 \times \frac{5}{10} + 3000 \times \frac{5}{10}) = log(4000) = 8.29$. Figure 26 presents the distribution of answer frequency score for each validation score category, after removing outliers. We define outliers as having answer frequency scores that are equal to or less than 0,

or equal to or greater than 10. Questions with perfect scores, on average, have answers that appear more frequent in the training set.

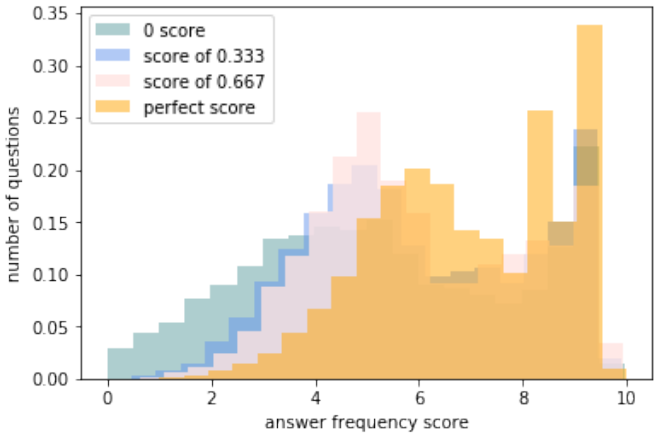


Figure 26: Distribution of Answer Frequency Score: higher means indicate having more frequent answers

In addition, we notice that a significant portion of the questions with 0 score have novel answers that never appear in the training questions.

The average answer frequency score is around 8.3 for single-word answers, 3.5 for two-word answers and 3.2 for longer answers. Inability to predict infrequent or novel answers well attribute to poor performance on questions with multi-word answers.

6.2 Lack of World Knowledge

Sometimes the correct answer depends on world knowledge, or common sense, and cannot be inferred from visual information in the picture alone. For example, as shown in Figure 25c, to correctly answer the question, the model needs the world knowledge that sheep eats grass. Another example is shown in Figure 25d, where the model needs to know the association between shade and sun. Without such knowledge, the model is unable to give predictions that make sense.

6.3 Vague Answers

In number questions, some human annotators use vague answers such as ‘many’ and ‘lot’. Since annotators have different interpretation of these words, the model encounters difficulties learning when to predict ‘many’, ‘lot’, etc. In addition, the evaluation metric may underestimate model performance in such cases. For example, if the model predicts ‘20’ and the human

answer set contains 2 '20' and 8 'many', the prediction score is only 0.667 even though the model answer is the same as the ground truth. In other words, although the evaluation metric takes into consideration multiple ground-truths, it does not consider cases where a single ground-truth can be expressed in multiple ways.

We conduct a simple experiment to test if enforcing a standard definition for 'many' improves model performance. For each number question starting with 'How (many)', we force ground-truth answer to be 'many' if the current ground truth is larger than 5. We also change the model answer to be 'many' if the current answer is larger than 5. As a result, model performance on number questions improves from 42.09 ± 0.47 to $47.35 \pm +0.48$. This confirms our hypothesis that vague answers negatively affects model performance.

7 Future Work

Experimental results demonstrate that `VQA-BERT` outperforms previous RNN-based approaches on all three question types (i.e. yes/no, number, other). Our results show the effectiveness of BERT in capturing language information in given questions. The following three areas can be explored in future work:

1. We formulate the VQA task as a classification problem, where answers that frequently reoccur during training are used at test time as plausible answers. Specifically, we consider only answers in training set that appear more than 8 times. One major drawback of this problem formulation is that `VQA-BERT` cannot predict novel answers, as demonstrated in Section 6.1. This leads to poor model performance in generating long answers. In the future, the plausibility of using a decoder architecture in the context of VQA, where a seq2seq model is used and generate one character at a time to formulate the final answer, can be explored.

2. Due to limited resources and time, we are only able to test `VQA-BERT` on the VQA dataset. Therefore, we cannot conclude how well `VQA-BERT` generalizes across different visual question answering datasets. Specifically, it is inconclusive if the high performance is due to capturing biases in the VQA dataset, or learning underlying visual and language structure. Future development include testing the model on other datasets

such as DAQUAR-ALL [Malinowski and Fritz, 2014] or Visual7W Telling dataset [Zhu et al., 2016].

3. `VQA-BERT` uses a simple unidirectional attention mechanism to construct question-guided image features. Many advanced attention mechanism has been proposed in recent years such as Yang et al. [2016] and Lu et al. [2016]. The impact of incorporating more advanced attention mechanisms can also be explored in the future.

8 Conclusion

In this paper we present `VQA-BERT`, a VQA model using BERT language representation, that provides modular improvements in question understanding and results in better overall performance. BERT allows our model to better understand the question and consolidate visual information with language understanding. Experiments show that `VQA-BERT` improves validation score across most question types, and is better at answering longer questions. Though `VQA-BERT` is evaluated on visual question answering task, it has the potential of being applied to other tasks involving language and vision integration.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. Bottom-up and top-down attention for image captioning and visual question answering. In *CVPR*, 2018.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Akira Fukui, Dong Huk Park, Daylen Yang, Anna Rohrbach, Trevor Darrell, and Marcus Rohrbach. Multimodal compact bilinear pooling for visual question answering and visual grounding. *arXiv preprint arXiv:1606.01847*, 2016.

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

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9 (8):1735–1780, 1997.

Allan Jabri, Armand Joulin, and Laurens van der Maaten. Revisiting visual question answering baselines. In *European conference on computer vision*, pages 727–739. Springer, 2016.

Yu Jiang, Vivek Natarajan, Xinlei Chen, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia. `https://github.com/facebookresearch/pythia`, 2018.

Vahid Kazemi and Ali Elqursh. Show, ask, attend, and answer: A strong baseline for visual question answering. *arXiv preprint arXiv:1704.03162*, 2017.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. 07 2016.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances In Neural Information Processing Systems*, pages 289–297, 2016.

Mateusz Malinowski and Mario Fritz. A multi-world approach to question answering about real-world scenes based on uncertain input. In *Advances in neural information processing systems*, pages 1682–1690, 2014.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013a.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013b.

Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

Damien Teney, Peter Anderson, Xiaodong He, and Anton van den Hengel. Tips and tricks for visual question answering: Learnings from the 2017 challenge. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4223–4232, 2018.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008, 2017.

Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual question answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40, 2017.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Zichao Yang, Xiaodong He, Jianfeng Gao, Li Deng, and Alex Smola. Stacked attention networks for image question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 21–29, 2016.

Yu Jiang*, Vivek Natarajan*, Xinlei Chen*, Marcus Rohrbach, Dhruv Batra, and Devi Parikh. Pythia v0.1: the winning entry to the vqa challenge 2018. *arXiv preprint arXiv:1807.09956*, 2018.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. Visual7w: Grounded question answering in images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4995–5004, 2016.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *Proceedings of the IEEE international conference on computer vision*, pages 19–27, 2015.