LXMERT: Learning Cross-Modality Encoder Representations from Transformers

Hao Tan **Mohit Bansal**
UNC Chapel Hill
{haotan, mbansal}@cs.unc.edu

Abstract

Vision-and-language reasoning requires an understanding of visual concepts, language semantics, and, most importantly, the alignment and relationships between these two modalities. We thus propose the LXMERT (Learning Cross-Modality Encoder Representations from Transformers) framework to learn these vision-and-language connections. In LXMERT, we build a large-scale Transformer model that consists of three encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. Next, to endow our model with the capability of connecting vision and language semantics, we pre-train the model with large amounts of image-and-sentence pairs, via five diverse representative pre-training tasks: masked language modeling, masked object prediction (feature regression and label classification), cross-modality matching, and image question answering. These tasks help in learning both intra-modality and cross-modality relationships. After fine-tuning from our pre-trained parameters, our model achieves the state-of-the-art results on two visual question answering datasets (i.e., VQA and GQA). We also show the generalizability of our pre-trained cross-modality model by adapting it to a challenging visual-reasoning task, NLVR^2, and improve the previous best result by 22% absolute (54% to 76%). Lastly, we demonstrate detailed ablation studies to prove that both our novel model components and pre-training strategies significantly contribute to our strong results.[1]

1 Introduction

Vision-and-language reasoning requires the understanding of visual contents, language semantics, and cross-modal alignments and relationships. There has been substantial past works in separately developing backbone models with better representations for the single modalities of vision and of language. For visual-content understanding, people have developed several backbone models (Simonyan and Zisserman, 2014; Szegedy et al., 2015; He et al., 2016) and shown their effectiveness on large vision datasets (Deng et al., 2009; Lin et al., 2014; Krishna et al., 2017). Pioneering works (Girshick et al., 2014; Xu et al., 2015) also show the generalizability of these pre-trained (especially on ImageNet) backbone models by fine-tuning them on different tasks. In terms of language understanding, last year, we witnessed strong progress towards building a universal backbone model with large-scale contextualized language model pre-training (Peters et al., 2018; Radford et al., 2018; Devlin et al., 2019), which has improved performances on various tasks (Rajpurkar et al., 2016; Wang et al., 2018) to significant levels. Despite these influential single-modality works, large-scale pretraining and fine-tuning studies for the modality-pair of vision and language are still under-developed.

Therefore, we present one of the first works in building a pre-trained vision-and-language cross-modality framework and show its strong performance on several datasets. We name this framework "LXMERT: Learning Cross-Modality Encoder Representations from Transformers" (pronounced: 'leksmert'). This framework is modeled after recent BERT-style innovations while further adapted to useful cross-modality scenarios. Our new cross-modality model focuses on learning vision-and-language interactions, especially for representations of a single image and its descriptive sentence. It consists of three Transformer (Vaswani et al., 2017) encoders: an object relationship encoder, a language encoder, and a cross-modality encoder. In order to better learn

[1] Published at EMNLP 2019. Code and pre-trained models publicly available at: https://github.com/airsplay/lxmert

Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pages 5100–5111,
Hong Kong, China, November 3–7, 2019.

the cross-modal alignments between vision and language, we next pre-train our model with five diverse representative tasks: (1) masked cross-modality language modeling, (2) masked object prediction via RoI-feature regression, (3) masked object prediction via detected-label classification, (4) cross-modality matching, and (5) image question answering. Different from single-modality pre-training (e.g., masked LM in BERT), this multi-modality pre-training allows our model to infer masked features either from the visible elements in the same modality, or from aligned components in the other modality. In this way, it helps build both intra-modality and cross-modality relationships.

Empirically, we first evaluate LXMERT on two popular visual question-answering datasets, VQA (Antol et al., 2015) and GQA (Hudson and Manning, 2019). Our model outperforms previous works in all question categories (e.g., Binary, Number, Open) and achieves state-of-the-art results in terms of overall accuracy. Further, to show the generalizability of our pre-trained model, we fine-tune LXMERT on a challenging visual reasoning task, Natural Language for Visual Reasoning for Real (NLVR^2) (Suhr et al., 2019), where we do not use the natural images in their dataset for our pre-training, but fine-tune and evaluate on these challenging, real-world images. In this setup, we achieve a large improvement of 22% absolute in accuracy (54% to 76%, i.e., 48% relative error reduction) and 30% absolute in consistency (12% to 42%, i.e., 34% relative error reduction). Lastly, we conduct several analysis and ablation studies to prove the effectiveness of our model components and diverse pre-training tasks by removing them or comparing them with their alternative options. Especially, we use several ways to take the existing BERT model and its variants, and show their ineffectiveness in vision-and-language tasks, which overall proves the need of our new cross-modality pre-training framework.

2 Model Architecture

We build our cross-modality model with self-attention and cross-attention layers following the recent progress in designing natural language processing models (e.g., transformers (Vaswani et al., 2017)). As shown in Fig. 1, our model takes two inputs: an image and its related sentence (e.g., a caption or a question). Each image is represented as a sequence of objects, and each sentence is represented as a sequence of words. Via careful design and combination of these self-attention and cross-attention layers, our model is able to generate language representations, image representations, and cross-modality representations from the inputs. Next, we describe the components of this model in detail.

2.1 Input Embeddings

The input embedding layers in LXMERT convert the inputs (i.e., an image and a sentence) into two sequences of features: word-level sentence embeddings and object-level image embeddings. These embedding features will be further processed by the latter encoding layers.

Word-Level Sentence Embeddings A sentence is first split into words $\{w_1, \ldots, w_n\}$ with length of n by the same WordPiece tokenizer (Wu et al., 2016) in Devlin et al. (2019). Next, as shown in Fig. 1, the word w_i and its index i (w_i's absolute position in the sentence) are projected to vectors by embedding sub-layers, and then added to the index-aware word embeddings:

$$\begin{aligned}\hat{w}_i &= \text{WordEmbed}\left(w_i\right)\\ \hat{u}_i &= \text{IdxEmbed}\left(i\right)\\ h_i &= \text{LayerNorm}\left(\hat{w}_i + \hat{u}_i\right)\end{aligned}$$

Object-Level Image Embeddings Instead of using the feature map output by a convolutional neural network, we follow Anderson et al. (2018) in taking the features of detected objects as the embeddings of images. Specifically, the object detector detects m objects $\{o_1, \ldots, o_m\}$ from the image (denoted by bounding boxes on the image in Fig. 1). Each object o_j is represented by its position feature (i.e., bounding box coordinates) p_j and its 2048-dimensional region-of-interest (RoI) feature f_j. Instead of directly using the RoI feature f_j without considering its position p_j in Anderson et al. (2018), we learn a position-aware embedding v_j by adding outputs of 2 fully-connected layers:

$$\begin{aligned}\hat{f}_j &= \text{LayerNorm}\left(W_{\text{F}} f_j + b_{\text{F}}\right)\\ \hat{p}_j &= \text{LayerNorm}\left(W_{\text{P}} p_j + b_{\text{P}}\right)\\ v_j &= \left(\hat{f}_j + \hat{p}_j\right)/2\end{aligned} \quad (1)$$

In addition to providing spatial information in visual reasoning, the inclusion of positional information is necessary for our masked object prediction pre-training task (described in Sec. 3.1.2).

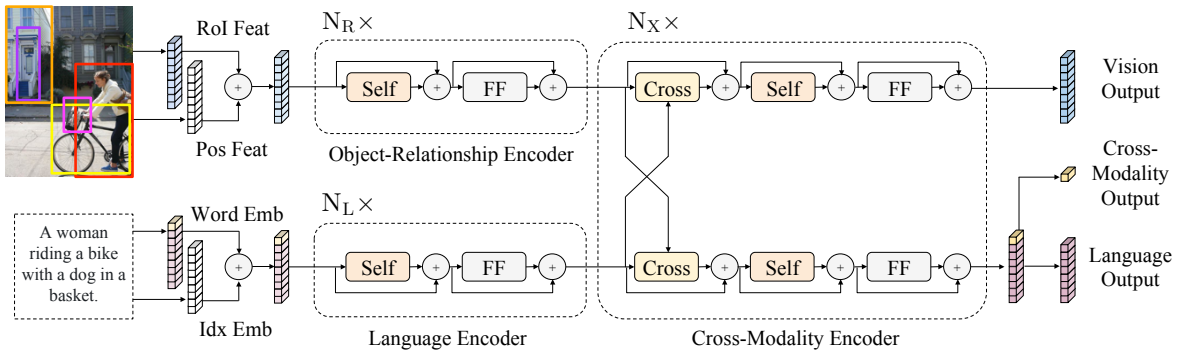

Figure 1: The LXMERT model for learning vision-and-language cross-modality representations. 'Self' and 'Cross' are abbreviations for self-attention sub-layers and cross-attention sub-layers, respectively. 'FF' denotes a feed-forward sub-layer.

Since the image embedding layer and the following attention layers are agnostic to the absolute indices of their inputs, the order of the object is not specified. Lastly, in Equation 1, the layer normalization is applied to the projected features before summation so as to balance the energy of the two different types of features.

2.2 Encoders

We build our encoders, i.e., the language encoder, the object-relationship encoder, and the cross-modality encoder, mostly on the basis of two kinds of attention layers: self-attention layers and cross-attention layers. We first review the definition and notations of attention layers and then discuss how they form our encoders.

Background: Attention Layers Attention layers (Bahdanau et al., 2014; Xu et al., 2015) aim to retrieve information from a set of *context* vectors $\{y_j\}$ related to a *query* vector x. An attention layer first calculates the matching score a_j between the *query* vector x and each *context* vector y_j. Scores are then normalized by softmax:

$$a_j = \text{score}(x, y_j)$$
$$\alpha_j = \exp(a_j) / \sum_k \exp(a_k)$$

The output of an attention layer is the weighted sum of the *context* vectors w.r.t. the softmax-normalized score: $\text{Att}_{\text{X}\to\text{Y}}(x, \{y_j\}) = \sum_j \alpha_j y_j$. An attention layer is called *self-attention* when the *query* vector x is in the set of *context* vectors $\{y_j\}$. Specifically, we use the multi-head attention following Transformer (Vaswani et al., 2017).

Single-Modality Encoders After the embedding layers, we first apply two transformer encoders (Vaswani et al., 2017), i.e., a **language encoder** and an **object-relationship encoder**, and each of them only focuses on a single modality (i.e., language or vision). Different from BERT (Devlin et al., 2019), which applies the transformer encoder only to language inputs, we apply it to vision inputs as well (and to cross-modality inputs as described later below). Each layer (left dashed blocks in Fig. 1) in a single-modality encoder contains a self-attention ('Self') sub-layer and a feed-forward ('FF') sub-layer, where the feed-forward sub-layer is further composed of two fully-connected sub-layers. We take $\mathbf{N_L}$ and $\mathbf{N_R}$ layers in the language encoder and the object-relationship encoder, respectively. We add a residual connection and layer normalization (annotated by the '+' sign in Fig. 1) after each sub-layer as in Vaswani et al. (2017).

Cross-Modality Encoder Each cross-modality layer (the right dashed block in Fig. 1) in the cross-modality encoder consists of two self-attention sub-layers, one bi-directional cross-attention sub-layer, and two feed-forward sub-layers. We stack (i.e., using the output of k-th layer as the input of $(k{+}1)$-th layer) $\mathbf{N_X}$ these cross-modality layers in our encoder implementation. Inside the k-th layer, the bi-directional cross-attention sub-layer ('Cross') is first applied, which contains two uni-directional cross-attention sub-layers: one from language to vision and one from vision to language. The query and context vectors are the outputs of the $(k{-}1)$-th layer (i.e., language features $\{h_i^{k-1}\}$ and vision features $\{v_j^{k-1}\}$):

$$\hat{h}_i^k = \text{CrossAtt}_{\text{L}\to\text{R}}\left(h_i^{k-1}, \{v_1^{k-1}, \ldots, v_m^{k-1}\}\right)$$
$$\hat{v}_j^k = \text{CrossAtt}_{\text{R}\to\text{L}}\left(v_j^{k-1}, \{h_1^{k-1}, \ldots, h_n^{k-1}\}\right)$$

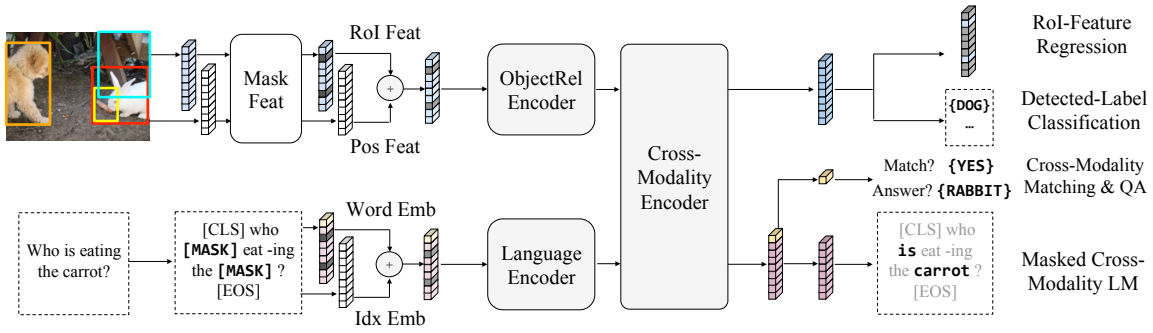

Figure 2: Pre-training in LXMERT. The object RoI features and word tokens are masked. Our five pre-training tasks learn the feature representations based on these masked inputs. Special tokens are in brackets and classification labels are in braces.

The cross-attention sub-layer is used to exchange the information and align the entities between the two modalities in order to learn joint cross-modality representations. For further building internal connections, the self-attention sub-layers ('Self') are then applied to the output of the cross-attention sub-layer:

$$\tilde{h}_i^k = \text{SelfAtt}_{\text{L}\to\text{L}}\left(\hat{h}_i^k, \{\hat{h}_1^k, \ldots, \hat{h}_n^k\}\right)$$
$$\tilde{v}_j^k = \text{SelfAtt}_{\text{R}\to\text{R}}\left(\hat{v}_j^k, \{\hat{v}_1^k, \ldots, \hat{v}_m^k\}\right)$$

Lastly, the k-th layer output $\{h_i^k\}$ and $\{v_j^k\}$ are produced by feed-forward sub-layers ('FF') on top of $\{\hat{h}_i^k\}$ and $\{\hat{v}_j^k\}$. We also add a residual connection and layer normalization after each sub-layer, similar to the single-modality encoders.

2.3 Output Representations

As shown in the right-most part of Fig. 1, our LXMERT cross-modality model has three outputs for language, vision, and cross-modality, respectively. The language and vision outputs are the feature sequences generated by the cross-modality encoder. For the cross-modality output, following the practice in Devlin et al. (2019), we append a special token [CLS] (denoted as the top yellow block in the bottom branch of Fig. 1) before the sentence words, and the corresponding feature vector of this special token in language feature sequences is used as the cross-modality output.

3 Pre-Training Strategies

In order to learn a better initialization which understands connections between vision and language, we pre-train our model with different modality pre-training tasks on a large aggregated dataset.

3.1 Pre-Training Tasks

3.1.1 Language Task: Masked Cross-Modality LM

On the language side, we take the masked cross-modality language model (LM) task. As shown in the bottom branch of Fig. 2, the task setup is almost same to BERT (Devlin et al., 2019): words are randomly masked with a probability of 0.15 and the model is asked to predict these masked words. In addition to BERT where masked words are predicted from the non-masked words in the language modality, LXMERT, with its cross-modality model architecture, could predict masked words from the vision modality as well, so as to resolve ambiguity. For example, as shown in Fig. 2, it is hard to determine the masked word 'carrot' from its language context but the word choice is clear if the visual information is considered. Hence, it helps building connections from the vision modality to the language modality, and we refer to this task as masked *cross-modality* LM to emphasize this difference. We also show that loading BERT parameters into LXMERT will do harm to the pre-training procedure in Sec. 5.1 since BERT can perform relatively well in the language modality without learning these cross-modality connections.

3.1.2 Vision Task: Masked Object Prediction

As shown in the top branch of Fig. 2, we pre-train the vision side by randomly masking objects (i.e., masking RoI features with zeros) with a probability of 0.15 and asking the model to predict proprieties of these masked objects. Similar to the language task (i.e., masked cross-modality LM), the model can infer the masked objects either from visible objects or from the language modality. Inferring the objects from the vision

Image Split	Images	Sentences (or Questions)					
		COCO-Cap	VG-Cap	VQA	GQA	VG-QA	All
MS COCO - VG	72K	361K	-	387K	-	-	0.75M
MS COCO ∩ VG	51K	256K	2.54M	271K	515K	724K	4.30M
VG - MS COCO	57K	-	2.85M	-	556K	718K	4.13M
All	180K	617K	5.39M	658K	1.07M	1.44M	9.18M

Table 1: Amount of data for pre-training. Each image has multiple sentences/questions. 'Cap' is caption. 'VG' is Visual Genome. Since MS COCO and VG share 51K images, we list it separately to ensure disjoint image splits.

side helps learn the object relationships, and inferring from the language side helps learn the cross-modality alignments. Therefore, we perform two sub-tasks: **RoI-Feature Regression** regresses the object RoI feature f_j with L2 loss, and **Detected-Label Classification** learns the labels of masked objects with cross-entropy loss. In the 'Detected-Label Classification' sub-task, although most of our pre-training images have object-level annotations, the ground truth labels of the annotated objects are inconsistent in different datasets (e.g., different number of label classes). For these reasons, we take detected labels output by Faster R-CNN (Ren et al., 2015). Although detected labels are noisy, experimental results show that these labels contribute to pre-training in Sec. 5.3.

3.1.3 Cross-Modality Tasks

As shown in the middle-rightmost part of Fig. 2, to learn a strong cross-modality representation, we pre-train the LXMERT model with 2 tasks that explicitly need both language and vision modalities.

Cross-Modality Matching For each sentence, with a probability of 0.5, we replace it with a mismatched[2] sentence. Then, we train a classifier to predict whether an image and a sentence match each other. This task is similar to 'Next Sentence Prediction' in BERT (Devlin et al., 2019).

Image Question Answering (QA) In order to enlarge the pre-training dataset (see details in Sec. 3.2), around $1/3$ sentences in the pre-training data are questions about the images. We ask the model to predict the answer to these image-related questions when the image and the question are matched (i.e., not randomly replaced in the cross-modality matching task). We show that pre-training with this image QA leads to a better cross-modality representation in Sec. 5.2.

[2] We take a sentence from another image as the mismatched sentence. Although the sentence and the image still have chance to match each other, this probability is very low.

3.2 Pre-Training Data

As shown in Table. 1, we aggregate pre-training data from five vision-and-language datasets whose images come from MS COCO (Lin et al., 2014) or Visual Genome (Krishna et al., 2017). Besides the two original captioning datasets, we also aggregate three large image question answering (image QA) datasets: VQA v2.0 (Antol et al., 2015), GQA balanced version (Hudson and Manning, 2019), and VG-QA (Zhu et al., 2016). We only collect **train and dev** splits in each dataset to avoid seeing any test data in pre-training. We conduct minimal pre-processing on the five datasets to create aligned image-and-sentence pairs. For each image question answering dataset, we take questions as sentences from the image-and-sentence data pairs and take answers as labels in the image QA pre-training task (described in Sec. 3.1.3). This provides us with a large aligned vision-and-language dataset of 9.18M image-and-sentence pairs on 180K distinct images. In terms of tokens, the pre-training data contain around 100M words and 6.5M image objects.

3.3 Pre-Training Procedure

We pre-train our LXMERT model on the large aggregated dataset (discussed in Sec. 3.2) via the pre-training tasks (Sec. 3.1). The details about the data splits are in the Appendix. The input sentences are split by the WordPiece tokenizer (Wu et al., 2016) provided in BERT (Devlin et al., 2019). The objects are detected by Faster R-CNN (Ren et al., 2015) which is pre-trained on Visual Genome (provided by Anderson et al. (2018)). We do not fine-tune the Faster R-CNN detector and freeze it as a feature extractor. Different from detecting variable numbers of objects in Anderson et al. (2018), we consistently keep 36 objects for each

Method	VQA				GQA			$NLVR^2$	
	Binary	Number	Other	**Accu**	Binary	Open	**Accu**	Cons	**Accu**
Human	-	-	-	-	91.2	87.4	89.3	-	96.3
Image Only	-	-	-	-	36.1	1.74	17.8	7.40	51.9
Language Only	66.8	31.8	27.6	44.3	61.9	22.7	41.1	4.20	51.1
State-of-the-Art	85.8	53.7	60.7	70.4	76.0	40.4	57.1	12.0	53.5
LXMERT	**88.2**	**54.2**	**63.1**	**72.5**	**77.8**	**45.0**	**60.3**	**42.1**	**76.2**

Table 2: Test-set results. VQA/GQA results are reported on the 'test-standard' splits and $NLVR^2$ results are reported on the unreleased test set ('Test-U'). The highest method results are in bold. Our LXMERT framework outperforms previous (comparable) state-of-the-art methods on all three datasets w.r.t. all metrics.

image to maximize the pre-training compute utilization by avoiding padding. For the model architecture, we set the numbers of layers $\mathbf{N_L}$, $\mathbf{N_X}$, and $\mathbf{N_R}$ to 9, 5, and 5 respectively.[3] More layers are used in the language encoder to balance the visual features extracted from 101-layer Faster R-CNN. The hidden size 768 is the same as $\text{BERT}_{\text{BASE}}$. We pre-train all parameters in encoders and embedding layers from scratch (i.e., model parameters are randomly initialized or set to zero). We also show results of loading pre-trained BERT parameters in Sec. 5.1. LXMERT is pre-trained with multiple pre-training tasks and hence multiple losses are involved. We add these losses with equal weights. For the image QA pre-training tasks, we create a joint answer table with 9500 answer candidates which roughly cover 90% questions in all three image QA datasets.

We take Adam (Kingma and Ba, 2014) as the optimizer with a linear-decayed learning-rate schedule (Devlin et al., 2019) and a peak learning rate at $1e-4$. We train the model for 20 epochs (i.e., roughly 670K[4] optimization steps) with a batch size of 256. We only pre-train with image QA task (see Sec. 3.1.3) for the last 10 epochs, because this task converges faster and empirically needs a smaller learning rate. The whole pre-training process takes 10 days on 4 Titan Xp.

Fine-tuning Fine-tuning is fast and robust. We only perform necessary modification to our model with respect to different tasks (details in Sec. 4.2). We use a learning rate of $1e-5$ or $5e-5$, a batch size of 32, and fine-tune the model from our pre-trained parameters for 4 epochs.

[3]If we count a single modality layer as one half cross-modality layer, the equivalent number of cross-modality layers is $(9+5)/2+5=12$, which is same as the number of layers in $\text{BERT}_{\text{BASE}}$.

[4]For comparison, ResNet on ImageNet classification takes 600K steps and BERT takes 1000K steps.

4 Experimental Setup and Results

In this section, we first introduce the datasets that are used to evaluate our LXMERT framework and empirically compare our single-model results with previous best results.

4.1 Evaluated Datasets

We use three datasets for evaluating our LXMERT framework: VQA v2.0 dataset (Goyal et al., 2017), GQA (Hudson and Manning, 2019), and $NLVR^2$. See details in Appendix.

4.2 Implementation Details

On VQA and GQA, we fine-tune our model from the pre-trained snapshot without data augmentation (analysis in Sec. 5.2). When training GQA, we only take raw questions and raw images as inputs and do not use other supervisions (e.g., functional programs and scene graphs). Since each datum in $NLVR^2$ has two natural images img_0, img_1 and one language statement s, we use LXMERT to encode the two image-statement pairs (img_0, s) and (img_1, s), then train a classifier based on the concatenation of the two cross-modality outputs. More details in Appendix.

4.3 Empirical Comparison Results

We compare our single-model results with previous best published results on VQA/GQA test-standard sets and $NLVR^2$ public test set. Besides previous state-of-the-art (SotA) methods, we also show the human performance and image-only/language-only results when available.

VQA The SotA result is BAN+Counter in Kim et al. (2018), which achieves the best accuracy among other recent works: MFH (Yu et al.,

2018), Pythia (Jiang et al., 2018), DFAF (Gao et al., 2019a), and Cycle-Consistency (Shah et al., 2019).[5] LXMERT improves the SotA overall *accuracy* ('Accu' in Table 2) by 2.1% and has 2.4% improvement on the 'Binary'/'Other' question sub-categories. Although LXMERT does not explicitly take a counting module as in BAN+Counter, our result on the counting-related questions ('Number') is still equal or better.[6]

GQA The GQA (Hudson and Manning, 2019) SotA result is taken from BAN (Kim et al., 2018) on the public leaderbaord. Our 3.2% *accuracy* gain over the SotA GQA method is higher than VQA, possibly because GQA requires more visual reasoning. Thus our framework, with novel encoders and cross-modality pre-training, is suitable and achieves a 4.6% improvement on open-domain questions ('Open' in Table 2).[7]

NLVR^2 NLVR^2 (Suhr et al., 2019) is a challenging visual reasoning dataset where some existing approaches (Hu et al., 2017; Perez et al., 2018) fail, and the SotA method is 'MaxEnt' in Suhr et al. (2019). The failure of existing methods (and our model w/o pre-training in Sec. 5.1) indicates that the connection between vision and language may not be end-to-end learned in a complex vision-and-language task without large-scale pre-training. However, with our novel pre-training strategies in building the cross-modality connections, we significantly improve the *accuracy* ('Accu' of 76.2% on unreleased test set 'Test-U', in Table 2) by 22%. Another evaluation metric *consistency* measures the proportion of unique sentences for which all related image pairs[8] are correctly predicted. Our LXMERT model improves *consistency* ('Cons') to 42.1% (i.e., by 3.5 times).[9]

[5] These are state-of-the-art methods at the time of our EMNLP May 21, 2019 submission deadline. Since then, there have been some recently updated papers such as MCAN (Yu et al., 2019b), MUAN (Yu et al., 2019a), and MLI (Gao et al., 2019b). MCAN (VQA challenge version) uses stronger mixture of detection features and achieves 72.8% on VQA 2.0 test-standard. MUAN achieves 71.1% (compared to our 72.5%).

[6] Our result on VQA v2.0 'test-dev' is 72.4%.

[7] Our result on GQA 'test-dev' is 60.0%.

[8] Each statement in NLVR^2 is related to multiple image pairs in order to balance the dataset answer distribution.

[9] These are the unreleased test set ('Test-U') results. On the public test set ('Test-P'), LXMERT achieves 74.5% Accu and 39.7% Cons.

Method	VQA	GQA	NLVR^2
LSTM + BUTD	63.1	50.0	52.6
BERT + BUTD	62.8	52.1	51.9
BERT + 1 CrossAtt	64.6	55.5	52.4
BERT + 2 CrossAtt	65.8	56.1	50.9
BERT + 3 CrossAtt	66.4	56.6	50.9
BERT + 4 CrossAtt	66.4	56.0	50.9
BERT + 5 CrossAtt	66.5	56.3	50.9
Train + BERT	65.5	56.2	50.9
Train + scratch	65.1	50.0	50.9
Pre-train + BERT	68.8	58.3	70.1
Pre-train + scratch	**69.9**	**60.0**	**74.9**

Table 3: Dev-set accuracy of using BERT.

5 Analysis

In this section, we analyze our LXMERT framework by comparing it with some alternative choices or by excluding certain model components/pre-training strategies.

5.1 BERT versus LXMERT

BERT (Devlin et al., 2019) is a pre-trained language encoder which improves several language tasks. As shown in Table 3, we discuss several ways to incorporate a $\text{BERT}_{\text{BASE}}$ pre-trained model for vision-language tasks and empirically compare it with our LXMERT approach. Although our full model achieves accuracy of 74.9% on NLVR^2, all results without LXMERT pre-training is around 22% absolute lower.

BERT+BUTD Bottom-Up and Top-Down (BUTD) attention (Anderson et al., 2018) method encodes questions with GRU (Chung et al., 2015), then attends to object RoI features $\{f_j\}$ to predict the answer. We apply BERT to BUTD by replacing its GRU language encoder with BERT. As shown in the first block of Table. 3, results of BERT encoder is comparable to LSTM encoder.

BERT+CrossAtt Since BUTD only takes the raw RoI features $\{f_j\}$ without considering the object positions $\{p_j\}$ and object relationships, we enhance BERT+BUTD with our novel position-aware object embedding (in Sec. 2.1) and cross-modality layers (in Sec. 2.2). As shown in the second block of Table 3, the result of 1 cross-modality layer is better than BUTD, while stacking more cross-modality layers further improves it. However, without our cross-modality pre-

Method	VQA	GQA	NLVR^2
1. P20 + DA	68.0	58.1	-
2. P20 + FT	68.9	58.2	72.4
3. P10+QA10 + DA	69.1	59.2	-
4. P10+QA10 + FT	**69.9**	**60.0**	**74.9**

Table 4: Dev-set accuracy showing the importance of the image-QA pre-training task. P10 means pre-training without the image-QA loss for 10 epochs while QA10 means pre-training with the image-QA loss. DA and FT mean fine-tuning with and without Data Augmentation, resp.

Method	VQA	GQA	NLVR^2
1. No Vision Tasks	66.3	57.1	50.9
2. Feat	69.2	59.5	72.9
3. Label	69.5	59.3	73.5
4. Feat + Label	**69.9**	**60.0**	**74.9**

Table 5: Dev-set accuracy of different vision pre-training tasks. 'Feat' is RoI-feature regression; 'Label' is detected-label classification.

training (BERT is language-only pre-trained), results become stationary after adding 3 cross-attention layers and have a 3.4% gap to our full LXMERT framework (the last bold row in Table 3).

BERT+LXMERT We also try loading BERT parameters[10] into LXMERT, and use it in model training (i.e., without LXMERT pre-training) or in pre-training. We show results in the last block of Table. 3. Compared to the 'from scratch' (i.e., model parameters are randomly initialized) approach, BERT improves the fine-tuning results but it shows weaker results than our full model. Empirically, pre-training LXMERT initialized with BERT parameters has lower (i.e., better) pre-training loss for the first 3 pre-training epochs but was then caught up by our 'from scratch' approach. A possible reason is that BERT is already pre-trained with single-modality masked language model, and thus could do well based only on the language modality without considering the connection to the vision modality (as discussed in Sec. 3.1.1).

5.2 Effect of the Image QA Pre-training Task

We show the importance of image QA pre-training task (introduced in Sec. 3.1.3) by excluding it or comparing it with its alternative: data augmentation.

Pre-training w/ or w/o Image QA To fairly compare with our original pre-training procedure (10 epochs w/o QA + 10 epochs w/ QA, details in Sec. 3.3) , we pre-train LXMERT model without image QA task for 20 epochs. As shown in Table 4 rows 2 and 4, pre-training with QA loss improves the result on all three datasets. The 2.1% improvement on NLVR^2 shows the stronger representations learned with image-QA pre-training, since all data (images and statements) in NLVR^2 are not used in pre-training.

Pre-training versus Data Augmentation Data augmentation (DA) is a technique which is used in several VQA implementations (Anderson et al., 2018; Kim et al., 2018; Jiang et al., 2018). It increases the amount of training data by adding questions from other image QA datasets. Our LXMERT framework instead uses multiple QA datasets in pre-training and is fine-tuned only on one specific dataset. Since the overall amounts of data used in pre-training and DA are similar, we thus can fairly compare these two strategies, and results show that our QA pre-training approach outperforms DA. We first exclude the QA task in our pre-training and show the results of DA fine-tuning. As shown in Table. 4 row 1, DA fine-tuning decreases the results compared to non-DA fine-tuning in row 2. Next, we use DA after QA-pre-training (row 3) and DA also drops the results.

5.3 Effect of Vision Pre-training tasks

We analyze the effect of different vision pre-training tasks in Table 5. Without any vision tasks in pre-training (i.e., only using the language and cross-modality pre-training tasks), the results (row 1 of Table 5) are similar to BERT+3 CrossAtt in Table 3. The two visual pre-training tasks (i.e., RoI-feature regression and detected-label classification) could get reasonable results (row 2 and row 3) on their own, and jointly pre-training with these two tasks achieves the highest results (row 4).

6 Related Work

Model Architecture: Our model is closely related to three ideas: bi-directional attention, Transformer, and BUTD. Lu et al. (2016) applies bi-

[10] Since our language encoder is same as BERT$_{\text{BASE}}$, except the number of layers (i.e., LXMERT has 9 layers and BERT has 12 layers), we load the top 9 BERT-layer parameters into the LXMERT language encoder.

directional attention to the vision-and-language tasks while its concurrent work BiDAF (Seo et al., 2017) adds modeling layers in solving reading comprehension. Transformer (Vaswani et al., 2017) is first used in machine translation, we utilize it as our single-modality encoders and design our cross-modality encoder based on it. BUTD (Anderson et al., 2018) embeds images with the object RoI features, we extend it with object positional embeddings and object relationship encoders.

Pre-training: After ELMo (Peters et al., 2018), GPT (Radford et al., 2018), and BERT (Devlin et al., 2019) show improvements in language understanding tasks with large-scale pre-trained language model, progress has been made towards the cross-modality pre-training. XLM (Lample and Conneau, 2019) learns the joint cross-lingual representations by leveraging the monolingual data and parallel data. VideoBert (Sun et al., 2019) takes masked LM on the concatenation of language words and visual tokens, where the visual tokens are converted from video frames by vector quantization. However, these methods are still based on a single transformer encoder and BERT-stype token-based pre-training, thus we develop a new model architecture and novel pre-training tasks to satisfy the need of cross-modality tasks.

Recent works since our EMNLP submission: This version of our paper (and all current results) was submitted to EMNLP[11] and was used to participate in the VQA and GQA challenges in May 2019. Since our EMNLP submission, a few other useful preprints have recently been released (in August) on similar cross-modality pre-training directions: ViLBERT (Lu et al., 2019) and VisualBERT (Li et al., 2019). Our LXMERT methods differs from them in multiple ways: we use a more detailed, multi-component design for the cross-modality model (i.e., with an object-relationship encoder and cross-modality layers) and we employ additional, useful pre-training tasks (i.e., RoI-feature regression and image question answering). These differences result in the current best performance (on overlapping reported tasks): a margin of 1.5% accuracy on VQA 2.0 and a margin of 9% accuracy on NLVR^2 (and 15% in consistency). LXMERT is also the only method which ranks in the top-3 on both the VQA and GQA challenges among more than 90 teams. We provide a detailed analysis to show how these additional pre-training tasks contribute to the fine-tuning performance in Sec. 5.2 and Sec. 5.3.

[11] EMNLP deadline was on May 21, 2019, and the standard ACL/EMNLP arxiv ban rule was in place till the notification date of August 12, 2019.

7 Conclusion

We presented a cross-modality framework, LXMERT, for learning the connections between vision and language. We build the model based on Transfermer encoders and our novel cross-modality encoder. This model is then pre-trained with diverse pre-training tasks on a large-scale dataset of image-and-sentence pairs. Empirically, we show state-of-the-art results on two image QA datasets (i.e., VQA and GQA) and show the model generalizability with a 22% improvement on the challenging visual reasoning dataset of NLVR^2. We also show the effectiveness of several model components and training methods via detailed analysis and ablation studies.

Acknowledgments

We thank the reviewers for their helpful comments. This work was supported by ARO-YIP Award #W911NF-18-1-0336, and awards from Google, Facebook, Salesforce, and Adobe. The views, opinions, and/or findings contained in this article are those of the authors and should not be interpreted as representing the official views or policies, either expressed or implied, of the funding agency. We also thank Alane Suhr for evaluation on NLVR^2.

References

Peter Anderson, Xiaodong He, Chris Buehler, Damien Teney, Mark Johnson, Stephen Gould, and Lei Zhang. 2018. Bottom-up and top-down attention for image captioning and visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6077–6086.

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

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2015. Gated feedback recur-

rent neural networks. In *International Conference on Machine Learning*, pages 2067–2075.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255. IEEE.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*.

Peng Gao, Zhengkai Jiang, Haoxuan You, Pan Lu, Steven C. H. Hoi, Xiaogang Wang, and Hongsheng Li. 2019a. Dynamic fusion with intra- and inter-modality attention flow for visual question answering. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Peng Gao, Haoxuan You, Zhanpeng Zhang, Xiaogang Wang, and Hongsheng Li. 2019b. Multi-modality latent interaction network for visual question answering. *arXiv preprint arXiv:1908.04289*.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. 2014. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2017. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6904–6913.

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

Dan Hendrycks and Kevin Gimpel. 2016. Bridging nonlinearities and stochastic regularizers with gaussian error linear units. *https://openreview.net/forum?id=Bk0MRI5lg*.

Ronghang Hu, Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Kate Saenko. 2017. Learning to reason: End-to-end module networks for visual question answering. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 804–813.

Drew A Hudson and Christopher D Manning. 2019. Gqa: a new dataset for compositional question answering over real-world images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

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

Jin-Hwa Kim, Jaehyun Jun, and Byoung-Tak Zhang. 2018. Bilinear attention networks. In *Advances in Neural Information Processing Systems*, pages 1564–1574.

Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. 2017. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123(1):32–73.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. 2019. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*.

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

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *arXiv preprint arXiv:1908.02265*.

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

Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. 2018. Film: Visual reasoning with a general conditioning layer. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of NAACL-HLT*, pages 2227–2237.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *International Conference on Learning Representations*.

Meet Shah, Xinlei Chen, Marcus Rohrbach, and Devi Parikh. 2019. Cycle-consistency for robust visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*.

Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Alane Suhr, Stephanie Zhou, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. A corpus for reasoning about natural language grounded in photographs. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. Videobert: A joint model for video and language representation learning. *arXiv preprint arXiv:1904.01766*.

Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. 2015. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9.

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

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. *EMNLP 2018*, page 353.

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

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*, pages 2048–2057.

Zhou Yu, Yuhao Cui, Jun Yu, Dacheng Tao, and Qi Tian. 2019a. Multimodal unified attention networks for vision-and-language interactions. *arXiv preprint arXiv:1908.04107*.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019b. Deep modular co-attention networks for visual question answering. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6281–6290.

Zhou Yu, Jun Yu, Chenchao Xiang, Jianping Fan, and Dacheng Tao. 2018. Beyond bilinear: Generalized multimodal factorized high-order pooling for visual question answering. *IEEE Transactions on Neural Networks and Learning Systems*, 29(12):5947–5959.

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

Appendix

A Evaluated Datasets Description

We use three datasets for evaluating our LXMERT framework.

VQA The goal of visual question answering (VQA) (Antol et al., 2015) is to answer a natural language question related to an image. We take VQA v2.0 dataset (Goyal et al., 2017) which reduces the answer bias compared to VQA v1.0. The dataset contains an average of 5.4 questions per image and the total amount of questions is 1.1M.

GQA The task of GQA (Hudson and Manning, 2019) is same as VQA (i.e., answer single-image related questions), but GQA requires more reasoning skills (e.g., spatial understanding and multi-step inference). 22M questions in the dataset are generated from ground truth image scene graph to explicitly control the question quality.

NLVR^2 Since the previous two datasets are used in pre-training for increasing the amount of pre-training data to a certain scale, we evaluate our LXMERT framework on another challenging visual reasoning dataset NLVR^2 where all the sentences and images are not covered in pre-training. Each datum in NLVR^2 contains two related natural images and one natural language statement. The task is to predict whether the statement correctly describes these two images or not. NLVR^2 has 86K, 7K, 7K data in training, development, and test sets, respectively.

B Details of NLVR^2 Fine-tuning

Each datum in NLVR^2 consists of a two-image pair (img_0, img_1), one statement s, and a ground truth label y^* indicating whether the statement correctly describe the two images. The task is to predict the label y given the images and the statement.

To use our LXMERT model on NLVR^2, we concatenate the cross-modality representations of the two images and then build the classifier with GeLU activation(Hendrycks and Gimpel, 2016). Suppose that $\text{LXMERT}(img, sent)$ is the single-vector cross-modality representation, the predicted probability is:

$$\begin{aligned} x_0 &= \text{LXMERT}(img_0, s) \\ x_1 &= \text{LXMERT}(img_1, s) \\ z^0 &= W_0[x_0; x_1] + b_0 \\ z^1 &= \text{LayerNorm}\left(\text{GeLU}(z^0)\right) \\ prob &= \sigma(W_1 z^1 + b_1) \end{aligned}$$

where σ is sigmoid function. The model is optimized by maximizing the log-likelihood, which is equivalent to minimize the binary cross entropy loss:

$$\mathcal{L} = \text{-}y^* \log prob - (1 - y^*) \log(1 - prob)$$

C Training, Validation, and Testing Splits

We carefully split each dataset to ensure that all testing images are not involved in any pre-training or fine-tuning steps. Our data splits for each dataset and reproducible code are available at https://github.com/airsplay/lxmert.

LXMERT Pre-Traininig Since MS COCO has a relative large validation set, we sample a set of 5k images from the MS COCO validation set as the mini-validation (minival) set. The rest of the images in training and validation sets (i.e., COCO training images, COCO validation images besides minival, and all the other images in Visual Genome) are used in pre-training. Although the captions and questions of the MS COCO test sets are available, we exclude all of them to make sure that testing images are not seen in pre-training.

Fine-tuning For training and validating VQA v2.0, we take the same split convention as in our LXMERT pre-training. The data related to images in LXMERT mini-validation set is used to validate model performance and the rest of the data in train+val are used in fine-tuning. We test our model on the VQA v2.0 'test-dev' and 'test-standard' splits. For GQA fine-tuning, we follow the suggestions in official GQA guidelines[12] to take *testdev* as our validation set and fine-tune our model on the joint train + validation sets. We test our GQA model on GQA 'test-standard' split. The images in NLVR^2 are not from either MS COCO or Visual Genome, we thus keep using the original split: fine-tune on train split, validate the model choice on val split, and test on the public ('Test-P') and unreleased ('Test-U') test splits.

D Training Details of 'BERT versus LXMERT'

When training with BERT only, we train each experiments for 20 epochs with a batch size 64/128 since it was not pre-trained on these cross-modality datasets. The learning rate is set to $1e-4$ instead of $5e-5$.

[12] https://cs.stanford.edu/people/dorarad/gqa/evaluate.html