X-LXMERT: Paint, Caption and Answer Questions with Multi-Modal Transformers

Jaemin Cho[1,2*] **Jiasen Lu**[1] **Dustin Schwenk**[1] **Hannaneh Hajishirzi**[1,3] **Aniruddha Kembhavi**[1,3]
Allen Institute for AI[1] UNC Chapel Hill[2] University of Washington[3]
jmincho@cs.unc.edu {jiasenl,dustins,hannah,anik}@allenai.org

Code and Demo: https://prior.allenai.org/projects/x-lxmert

Abstract

Mirroring the success of masked language models, vision-and-language counterparts like VILBERT, LXMERT and UNITER have achieved state of the art performance on a variety of multimodal discriminative tasks like visual question answering and visual grounding. Recent work has also successfully adapted such models towards the generative task of image captioning. This begs the question: *Can these models go the other way and generate images from pieces of text?* Our analysis of a popular representative from this model family – LXMERT – finds that it is unable to generate rich and semantically meaningful imagery with its current training setup. We introduce X-LXMERT, an extension to LXMERT with training refinements including: discretizing visual representations, using uniform masking with a large range of masking ratios and aligning the right pre-training datasets to the right objectives which enables it to paint. X-LXMERT's image generation capabilities rival state of the art generative models while its question answering and captioning abilities remains comparable to LXMERT. Finally, we demonstrate the generality of these training refinements by adding image generation capabilities into UNITER to produce X-UNITER.

1 Introduction

The past year has seen a spate of BERT-style (Devlin et al., 2019) transformer-based architectures (Lu et al., 2019; Chen et al., 2019; Li et al., 2019) proposed for vision-and-language tasks. These models are typically pre-trained on large image captioning corpora, extending ideas from masked language modeling to mask both the image and text modalities and produce state of the art results on a variety of vision and language tasks including visual question answering, visual grounding and image retrieval. These impressive results as well as recent probing mechanisms (Ilharco et al., 2020) suggest that these models are able to capture a variety of semantics in images including objects, attributes and their relationships and ground these in natural language.

While these models have been extensively evaluated over several discriminative tasks, relatively little attention has been paid to their generative capabilities. Bidirectional transformer models like BERT which exploit context preceding and following the current token are not explicitly designed for generation. Recent work for language-only transformers (Wang and Cho, 2019; Dong et al., 2019; Liao et al., 2020) adapt these models towards this capability using sampling procedures. Such techniques have also been adapted successfully for image captioning - inputting an image and sampling the textual side of the model to generate a relevant caption (Zhou et al., 2020). This begs the question: Can we go the other way and sample images from input pieces of text? i.e. *Do vision-and-language BERT models know how to paint?*

In this work, we probe the ability of a powerful and popular representative from this family of models - LXMERT (Tan and Bansal, 2019), to produce high fidelity and semantically meaningful images conditioned on captions. Interestingly, our analysis leads us to the conclusion that LXMERT in its current form does not possess the ability to paint - it produces images that have little resemblance to natural images. This is a somewhat surprising finding given LXMERT's masked training objectives for both modalities and its impressive performance on tasks that seemingly require a similar skill set.

We find that this is largely due to the regression training objective used by this family of models to predict masked features on the visual side. This is

*This work was done as part of the Pre-Doctoral Young Investigator residency program at the Allen Institute for AI.

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 8785–8805,
November 16–20, 2020.

in contrast with the textual side, where they predict masked tokens within a large discrete vocabulary using a classification objective. Regressing features in high dimensional spaces is challenging to optimize and introduces noise at inference. This gets compounded when using iterative sampling procedures to predict the entire set of visual features. A downstream image generator consuming these predictions isn't able to recover from this noise even when fine-tuned on LXMERT's predictions.

We introduce X-LXMERT that builds upon LXMERT and enables it to effectively perform discriminative as well as generative tasks. Our key refinements include: (a) simplifying the visual inputs to use grid features instead of object detection bounding boxes, (b) discretizing visual representations, (c) using uniform masking with a large range of masking ratios to enable the model to predict the entire set of visual clusters at inference time and (d) aligning the right pre-training datasets to the right objectives. When coupled with our proposed image generator, X-LXMERT is able to generate rich imagery that is semantically consistent with the input captions. Importantly, X-LXMERT's image generation capabilities rival state-of-the-art image generation models (designed only for generation), while its question answering capabilities show little degradation compared to LXMERT.

These refinements are not LXMERT-specific. They are designed to be easily applicable to a wide variety of multimodal BERT models. We find that UNITER, a single stream model for vision-and-language tasks, produces very poor images when coupled with a generator, but with our extensions, the resulting X-UNITER produces images of a similar quality to X-LXMERT.

In summary, we present X-LXMERT, a unified multimodal transformer model that can answer questions, and also generate captions and images. Our extensions to enable these capabilities are not tied to LXMERT's underlying architecture. We expect that the entire family of multimodal BERT models can be enhanced with image generative capabilities using our introduced strategy.

2 Related works

Visual-Language transformer models Recent multi-modal pre-training models show significant improvements on a wide range of downstream tasks, including discriminiative (eg., visual question answering) and generation task (eg. image captioning (Zhou et al., 2020)). Some methods use a single transformer architecture to jointly encode text and image (Li et al., 2019; Su et al., 2019; Alberti et al., 2019; Rahman et al., 2020; Li et al., 2020; Chen et al., 2019; Qi et al., 2020; Huang et al., 2020), while others use two-stream architectures (Lu et al., 2019, 2020; Tan and Bansal, 2019). These models typically consume object detection features. We probe this family of models at the task of image generation and present extensions that enable them to reliably generate images.

Sequence generation with undirectional transformer When generating sequences with conventional transformer language models, it is natural to sample tokens from left to right. However, since undirectional transformers (eg. BERT) are not trained with a specific generation order, a line of works has investigated different strategies for sequence generation with undirected models. Wang and Cho (2019) use Gibbs sampling from an all-mask sequence, and Dong et al. (2019); Bao et al. (2020) use causal attention during training for left-to-right generation. Liao et al. (2020); Mansimov et al. (2019); Ghazvininejad et al. (2019) sample masks from a uniform distribution during training for arbitrary order or parallel generation. We adapt these techniques for grid-based image generation.

Text-to-image synthesis Synthesizing images from text descriptions continues to be challenging. Since the pioneering work of Reed et al. (2016), many methods have adopted GANs (Goodfellow et al., 2014) to generate high-fidelity images. Nguyen et al. (2017) generate images that maximizes activation of a pretrained captioning model. Recent works (Zhang et al., 2017, 2018; Xu et al., 2018; Li et al., 2019) use multi-stage generation, where low-resolution images are initially sampled, then gradually upsampled and improved in later stages. These models are specialized toward image generation, whereas our model can not just generate images, but also answer questions and generate captions. Also, our design is modular in nature. While we use a compact image generator with X-LXMERT, one can also replace it with either of the aforementioned model architectures. There is another line of works predicting object layouts from text and generating image based on the layouts (Hong et al., 2018; Tan et al., 2019). These models use bounding box annotations to train layout predictors, while X-LXMERT implicitly learns the layouts only from text and image alignments.

Grid visual representation Compared to bounding box representations which requires expensive object detection annotations, grid representations of images can be naturally obtained from CNNs. Jiang et al. (2020); Huang et al. (2020) have recently shown that these can be almost as powerful as bounding box representations for VQA. Grid representation have been widely used in vision tasks, including self-supervised learning (Oord et al., 2018; Henaff et al., 2019; Trinh et al., 2019; Gidaris et al., 2020; Noroozi and Favaro, 2016) and image generation (van den Oord et al., 2017; Lin et al., 2019). We leverage grid visual representations to enable LXMERT to generate images.

3 Background: Revisiting LXMERT

Over the past year, a large number of transformer based architectures for multimodal data have produced impressive results across a variety of discriminative tasks. Some of these models have been shown to perform very well at the generative task of Image Captioning, but little attention has been paid to the reverse generative task: generating images given text. In this work, we first probe one popular representative from this family - LXMERT (Tan and Bansal, 2019) - in its ability to paint; and propose extensions that enable it to paint.

LXMERT is a cross modality transformer with inputs: image I and text T. This is represented as the sequence $\{v_1, \ldots, v_{\mathcal{T}}, \texttt{CLS}, w_1, \ldots, w_T, \texttt{EOS}\}$ where $\{v_i\}_{i=1}^{\mathcal{T}}$ are image region features, $\{w_j\}_{j=1}^{T}$ are word tokens and CLS and EOS are special tokens. LXMERT outputs embeddings for each input $\{h_{v_i}\}_{i=1}^{\mathcal{T}}$, $\{h_{w_j}\}_{j=1}^{T}$ and $h_{\texttt{CLS}}$, $h_{\texttt{EOS}}$. $h_{\texttt{CLS}}$ is used as the cross-modality output. Internally, LXMERT consists of two types of encoders: single-modality encoders for each modality and a cross-modality encoder using bi-directional cross attention to exchange information and align entities across the modalities.

LXMERT is pretrained on several vision-and-language datasets with five objectives: Masked language modeling (MLM), Masked visual feature regression (MVFR) - reconstructing randomly masked words and regions given the remaining inputs, Masked object classification (MOC) - object classification on masked image regions, Image-text matching (ITM) - image-caption alignment prediction and Question answering (QA) - answering a question given image input. After pretraining, LXMERT is finetuned for various downstream tasks. Unless noted, we use the default settings and hyperparameters of LXMERT in our experiments.

4 Probing LXMERT's Ability to Paint

In order to probe LXMERT's ability to paint, we first modify its input image representation to a *grid based* feature set (Sec. 4.1) and then pass these to an image generator (Sec. 4.2).

4.1 Grid Image Features

Most popular multimodal BERT models use image features extracted from the output of a Faster R-CNN (Ren et al., 2015) object detector. The detected objects typically have various locations and sizes. Passing these features into an image generator poses some challenges: (1) LXMERT is not trained to predict locations of given objects (2) it is not trivial to predict both object classes and their locations simultaneously (3) object detections do not cover backgrounds.

We modify LXMERT to use a uniform $N \times N$ grid and use RoI Pooling to extract the grid features. Note that we use the same detection backbone pretrained on the Visual Genome dataset to maintain parity with the original LXMERT. Our experiments in Sec 6 show that moving to a grid based input causes very little degradation to downstream QA tasks, a finding consistent with Jiang et al. (2020).

Sampling grid features: Given text input, we sample predicted visual features $\{h_{v_i}\}_{i=1}^{\mathcal{T}}$ where $\mathcal{T} = N \times N$ is the number of image regions, using Gibbs sampling in a manner similar to language generation using BERT by Wang and Cho (2019).

4.2 Image Generation

We use a compact image generator inspired by recent state of the art image synthesis methods leveraging Generative Adversarial Networks (GAN) (Goodfellow et al., 2014). Its takes as inputs an $N \times N$ grid of visual features from the pretrained Faster-RCNN network and generates an image. As shown in Fig 1, the input grid features are projected through convolutional layers and then passed to an image generator, which consists of multiple residual blocks (Miyato et al., 2018). Each generator residual block has SPADE layer (Park et al., 2019) which guides generator to outptut high fidelity images given semantic grid layouts. In our experiments, we use an image generator which takes 8×8 grid features and outputs an 256×256 image.

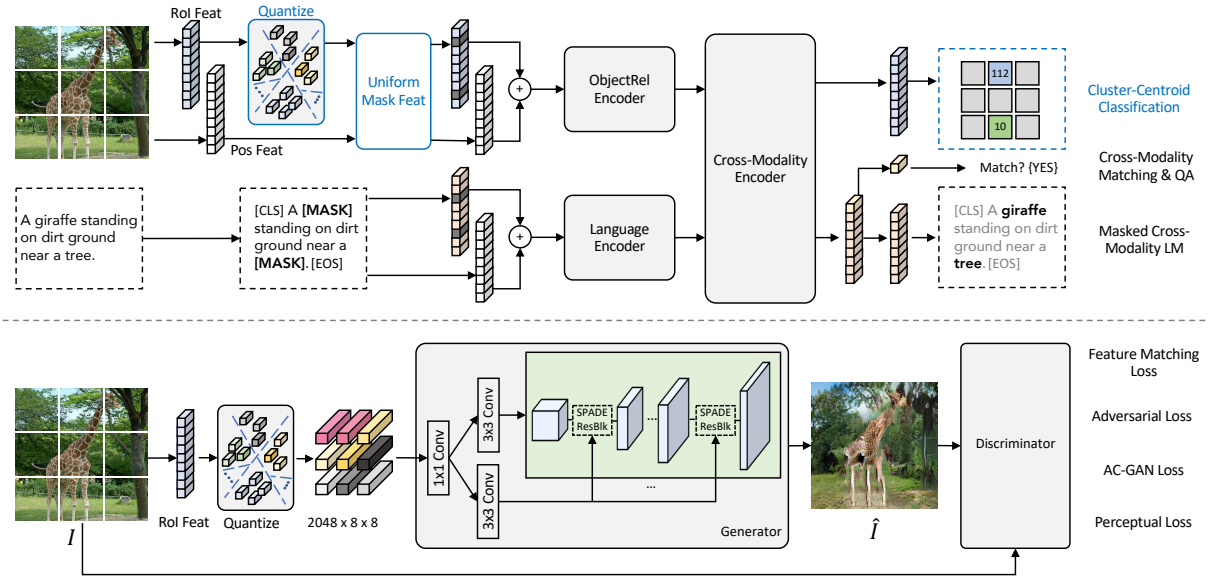

Figure 1: **Top**: Overview of the proposed X-LXMERT model. Blocks in blue are the modifications we make to LXMERT model to enable it to paint. **Bottom**: Overview of the image generation architecture. The input to the model is a natural image that is compressed to a quantized latent map of size 8×8 by RoI Pooling. We use a generator consisting of multiple residual blocks with SPADE layer which encodes 8×8 grid features.

Training the image generator: The generator is pre-trained using 8×8 ground truth Faster-RCNN features, akin to *teacher forcing*, without any inputs from LXMERT. We train the generator with the same loss as Park et al. (2019), but replacing the segmentation map with a grid feature map.

Fig. 2 (b) shows that our generation architecture can successfully reconstruct images using ground truth pre-trained grid features. Note that the generator still displays some reconstruction errors compared with modern auto-encoders such as VQ-VAEv2 (Razavi et al., 2019) primarily due to (1) freezing the encoder backbone in order to match LXMERT's training settings (2) restricting grid features to have a low (and manageable) dimension.

4.3 Can LXMERT Paint?

Our experiments in Section 6 reveal that LXMERT is unable to produce visual features that can be converted to a meaningful image by a generator. Figure 2 shows an example. Recall that the LXMERT loss function includes a regression loss - MVFR - that corresponds to regressing target visual features given the textual and visual context. Unfortunately, at inference, this loss on the validation set remains high, causing the predicted visual features to be fairly noisy. In addition, the Gibbs sampling procedure causes this error to propagate over the entire set of features. The resulting predictions aren't suitable to be used for downstream image generation.

5 X-LXMERT

In this section, we present X-LXMERT[1] that extends LXMERT, enabling it to paint, while still maintaining a high performance on discriminative tasks. X-LXMERT has three key refinements that enable it to paint (Sec. 5.1): discretizing visual representations, using uniform masking with a large range of masking ratios, and aligning the right pre-training datasets to the right objectives. We then leverage Gibbs sampling to generate visual features given textual input (Sec. 5.2).

5.1 From LXMERT to X-LXMERT

Discrete visual representations: We observe that the visual features regressed by LXMERT are not suitable for image generation. Instead, akin to VideoBERT (Sun et al., 2019), we first create a visual vocabulary using K-mean clustering, approximate the target visual features via a nearest neighbor search, and modify LXMERT to predict the cluster ID for each masked visual token. A new Cluster-Centroid Classification objective (CCC) is used to replace the previous regression objective with a high cardinality classification objective. Our experiments show that discretizing visual representations results helps in predicting better visual features, stems the propagation of feature noise over sampling iterations and generates rich imagery.

[1] X-LXMERT is an LXMERT with a "display server"

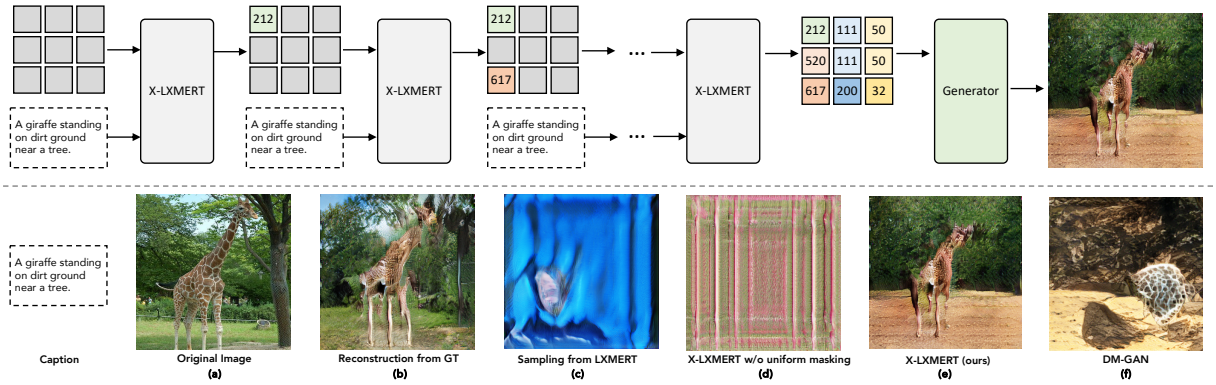

Figure 2: **Top**: Image generation from X-LXMERT. Given the text input and all masked visual feature, we first sample grid features by using Gibbs sampling with multiple iterations. Then the sampled grid features are fed into the generator to generate the image. **Bottom**: Sampled images, from left to right (a) Original image (b) Reconstruction from GT features (c) Sampling from LXMERT + Grid (d) Sampling from X-LXMERT without uniform masking pretraining (e) Our proposed X-LXMERT (f) Generated image from DM-GAN (Zhu et al., 2019).

Uniform instead of Bernoulli masking: Following BERT, LXMERT uses Bernouli sampling (with $p = 0.15$) to determine positions of the masked tokens on the visual and textual features. In order to generate an image from captions, all tokens on the vision side must be masked and predicted. A low probability Bernoulli sampling procedure does not prepare the model well for the generation task, and increasing the probability to very high values leads to poor pre-training. To resolve this, we use Uniform masking on the vision modality. X-LXMERT's uniform masking first samples the masking ratio from a uniform prior distribution ([0,1]), and then samples the desired number of positions randomly. This subjects the model to a variety of masking ratios, and our experiments reveal that this greatly benefits image generation.

Updating pre-training data: LXMERT uses a variety of data to pre-train the model: QA data from multiple sources, caption data from COCO and captions from Visual Genome (VG). Since X-LXMERT uses the CCC loss function, predicting visual features given questions like: "What is shown in the image?" is very ambiguous and results in models that cannot predict visual clusters. Similarly, many captions from VG (e.g., "A bag" or "Glasses on the hair") tend to describe small regions of the image and not the whole image, which makes them unsuited to train the CCC objective. X-LXMERT drops QA data and the captions from VG for CCC objective for visual cluster prediction.

5.2 Sampling Strategies for X-LXMERT

Given text input, predicting the entire set of visual features in one step does not produce good results. Instead, we employ Gibbs sampling to iteratively sample features at different spatial locations. In contrast to text generation, where left-to-right is considered a natural order, there is no natural order for generating images. The grid sampling process starts with N^2 grids filled with the `MASK` special token. The model then iteratively updates locations either one-by-one or multiple in parallel. There are several sampling strategies for sampling locations on the square grid, primarily falling into two buckets: auto-regressive and parallel.

Autoregressive sampling In each iteration, a grid position is sampled, masked and predicted. Then the corresponding `MASK` token is replaced with the predicted one, and the process is repeated until all locations are updated.

- TL→BR: Positions are sequentially chosen from top-left to bottom-right, similar to PixelRNN (van den Oord et al., 2016).
- Random (Liao et al., 2020): Positions are selected in random order. After N^2 steps, locations may be updated more than once.

Non-autoregressive sampling In each iteration, multiple positions are sampled, masked with `MASK`, predicted and then replaced.

- Mask-predict-K (Ghazvininejad et al., 2019): This requires K sampling steps. In the first iteration, all N^2 locations are updated. Then, we linearly decay the number of tokens updated per iteration. For example, for a 2×2 grid whereby $N^2 = 4$, if $K = 4$ then (4, 3, 2, 1) positions are updated in each iteration. Within each iteration, positions with the lowest confidence are updated.

Our experiments show that Mask-Predict-4 consistently produces good results across a variety of

generation metrics and we propose using it for X-LXMERT. Our uniform masking aligns well with the linear decay of Mask-Predict and makes the model robust to a varied number of masked locations.

5.3 Training Details

Generator Following (Park et al., 2019), our generator and discriminator are jointly trained with 4 losses: (1) hinge adversarial loss (Lim and Ye, 2017; Tran et al., 2017), (2) AC-GAN loss (Odena et al., 2017), (3) discriminator feature matching loss (Wang et al., 2018) and (4) perceptual loss (Johnson et al., 2016). The coefficients for different loss are $(1, 1, 10, 10)$ respectively. The perceptual loss is calculated with ResNet-50 (He et al., 2016) pre-trained on ImageNet (Deng et al., 2009). We use Adam optimizer (Kingma and Ba, 2015) with $(\beta^1, \beta^2) = (0, 0.999)$ and two-time update rule (Heusel et al., 2017) with learning rate of 0.0004 and 0.0001 for generator and discriminator respectively. We train the generator with batch size 96 for 60 epochs. Note that the generator parameters are fixed after training and not finetuned. Please refer Sec. D for more details.

Pre-training Following LXMERT (Tan and Bansal, 2019), we use AdamW optimizer (Loshchilov and Hutter, 2019) with $(\beta^1, \beta^2) = (0.9, 0.999)$ and learning rate `1e-5` with 5% linear warmup schedule. We train X-LXMERT on with batch size 920 for 20 epochs. Instead of using all pretraining tasks for each step, we first uniformly sample a modality to mask from `[image, text, no-mask]` and run corresponding tasks. Please refer to Sec. C.5 for more details.

Finetuning For each downstream task, a task head consisting of two fully connected layers is trained along with pre-trained X-LXMERT. We used the same parameter setting with LXMERT. Please refer to Sec. C.6 for more details.

6 Experimental Setup

In this section we present experimental setups to evaluate image generation, visual question answering and visual reasoning.

6.1 Evaluating Image Generation

We train and evaluate models using the MS COCO captioning dataset (Lin et al., 2014). We compare X-LXMERT with LXMERT and state-of-the-art text-to-image generation methods: StackGAN (Zhang et al., 2018), PPGN (Nguyen et al., 2017), AttnGAN (Xu et al., 2018), ControlGAN (Li et al., 2019), and DM-GAN (Zhu et al., 2019). Image generation is a particularly difficult task to evaluate, due to the variability in acceptable outputs for a given caption, as well as the subjective nature of perceiving image quality. We present a suite of automated and manual metrics to compare models.

Automated Metrics: Evaluate image quality We use Inception score (IS) (Salimans et al., 2016) to measure image diversity and Fréchet Inception Distance (FID) (Heusel et al., 2017) to measure authenticity; using Inception v3 (Szegedy et al., 2016) as a surrogate net.

Automated Metrics: Evaluate semantics We use two variants of R-precision (Xu et al., 2018), R-prec-easy and R-prec-hard to evaluate if the image is well conditioned on the input text. Given a generated image, a positive caption and negatives, R-precision measures the retrieval rate for the positive caption using a surrogate multi-modal network. We use an independent surrogate - ViLBERT-MT (Lu et al., 2020) for this purpose. R-prec-easy is the variant of R-precision with easy negatives (sampled randomly amongst the caption set). R-prec-hard is the variant with hard negatives (swapping a word in a caption with another word within the same category, e.g., red ⇒ green). We choose words from one of 4 categories: nouns (80 COCO objects), 64 verbs, 10 colors and 10 numbers.

The above automatic metrics, while cheap and reproducible, are noisy because they depend on imperfect surrogate models. The ultimate measure of quality and semantics for image generation continues to be crowd-sourced human studies.

Human Study: Pairwise preferences We conduct a human preference evaluations between X-LXMERT and the best performing model in the automated metrics—DM-GAN. We measure (1) *Semantic preference* by showing two image and asking annotators to select the one that best matches the source caption. (2) *Fidelity preference* by showing the two images alone and asking which appears more realistic. Both evaluations also allow a third option (Tie) to be selected. For each evaluation, 5000 image pairs were used, and 357 unique crowdworkers participated in total (median annotations per worker—17).

Human Study: Our new metric – HUMMUS The above pairwise test is very useful and widely used to evaluate generative models, but measur-

Methods	Text-to-Image Generation							Visual Question Answering			Visual Reasoning	
	IS↑	FID↓	R-prec -easy↑	R-prec -hard↑	HUMMUS	Human pairwise pref		VQA		GQA	NLVR^2	
						Semantics	Fidelity	test-dev	test-std	test-std	dev	test-P
Original Image	36.6	-	89.6	47.6	0.73	-	-	-	-	-	-	-
StackGAN	8.5	-	-	-	-	-	-	-	-	-	-	-
PPGN	9.6	-	-	-	-	-	-	-	-	-	-	-
AttnGAN	25.9	35.5	-	-	-	-	-	-	-	-	-	-
ControlGAN	24.1	-	-	-	-	-	-	-	-	-	-	-
DM-GAN	**30.5**	**32.6**	**51.8**	**27.5**	**0.49**	37.0	35.9	-	-	-	-	-
X-LXMERT	22.7	37.4	40.8	25.1	**0.49**	**52.0**	**50.0**	68.6	68.7	58.4	72.4	72.4
LXMERT*+Grid	1.6	316.7	0.5	6.6	0.27			71.1	71.2	60.1	74.6	74.0
LXMERT	-	-	-	-	-	-	-	**72.4**	**72.5**	**60.3**	**74.9**	74.5
LXMERT*	-	-	-	-	-	-	-	70.9	71.1	59.9	**74.9**	**75.0**

Table 1: Comparing X-LXMERT, LXMERT and baselines on image generation, visual question answering and visual reasoning tasks. The pairwise metric compares LXMERT and DM-GAN; numbers do not sum to 100 due to the TIE option provided to annotators. Note that X-LXMERT and LXMERT*+Grid are the only models that are able to produce results for all tasks. *: Our re-implementation of LXMERT.

ing new models becomes challenging, since they must compare to all old models. To expand human evaluation, we present a novel metric to test semantic consistency between the caption and image inspired by masked token modeling, named - HUmans Measuring seMantics Using maSking (HUMMUS). To compute HUMMUS, human annotators are shown an image and its caption with a single word masked out. They are asked to complete the partial caption based on information in the image, and a match is counted only when a majority of annotators supply the correct word. The total score is reported as a ratio of these successful matches. The task was run on 2800 image-caption pairs (2289 unique images), with 5 annotators per pair. A total of 280 unique crowdworkers completed the task, with a median of 13 images annotated per worker. A high HUMMUS score reveals that the generated images contain the corresponding semantics, well enough to be recognized. The masked word is chosen from one of 3 categories: 80 COCO nouns, verbs and colors.

6.2 Evaluating Visual Question Answering

We train and evaluate models for visual question answering using the VQA2.0 (Goyal et al., 2019) and GQA (Hudson and Manning, 2019) datasets, which provide an image and a question and require the model to generate an answer.

6.3 Evaluating Visual Reasoning

We train and evaluate models for visual reasoning using the NLVR^2 (Suhr et al., 2019) dataset and report numbers on the dev and test-P splits. The NLVR^2 dataset requires models to look at two images and determine if an accompanying caption is True or False. This is a particularly challenging dataset for present day vision and language models.

7 Experimental Results

We now present a comparison of X-LXMERTwith several baselines on the generative and discriminative tasks, along with ablation studies and qualitative results. We also show the generality of our techniques via extending UNITER to create X-UNITER.

7.1 Quantitative Results

Table 1 provides detailed metrics for X-LXMERT and baselines. It also provides generation metrics for the original image in the dataset for the corresponding input text. Note that X-LXMERT and LXMERT+Grid are the only models that are able to produce results for all tasks.

Image Generation As seen, X-LXMERT significantly outperforms LXMERT across all generation metrics. X-LXMERT even outperforms two specialized generation models, comparable to AttnGAN and ControlGAN. Our model is lower compared to DM-GAN in terms of automated metric (IS and FID), however, it is competitive with DM-GAN at semantic metric (R-prec-hard)[3].

Note that X-LXMERT's image generator is much smaller than the one used by DM-GAN (1.7M vs 22.3M parameters). While the transformer employed in X-LXMERT is large, it is a unified tex-

[2]We use coco pre-trained model from https://github.com/MinfengZhu/DM-GAN

[3]Note: R-prec and HUMMUS are reported only for DM-GAN (the strongest of the 5 baselines), since this was the only model with code and pretrained weights. IS and FID numbers are from their respective publications or from Zhu et al. (2019). The detailed R-prec-hard numbers across categories are presented in the appendix.

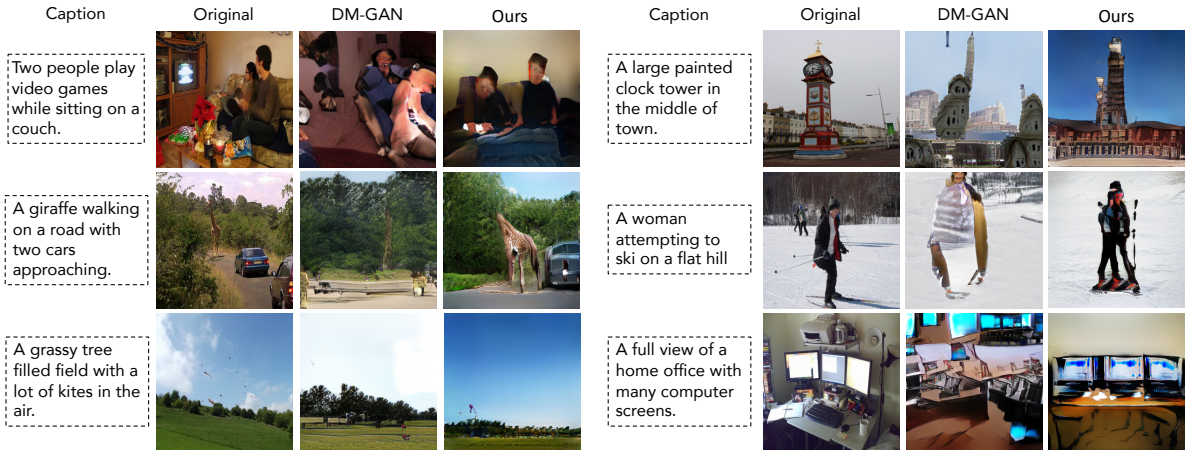

Figure 3: Images generated by DM-GAN^2(Zhu et al., 2019) and images generated by our proposed X-LXMERT.

tual and visual encoder used for multiple tasks and is not finetuned for image generation. We expect X-LXMERT's image quality to improve further when coupled with a larger image generator such as the one by DM-GAN.

Table 1 also presents HUMMUS scores. Here we see that the semantics generated by X-LXMERT is on par with DM-GAN and still significantly better than LXMERT. All models are still a distance away from the original image. HUMMUS matches on the lemmatized forms of masked words to allow for lexical variation, but it misses synonyms and other valid descriptors. This causes the score for the original image to drop to its reported value. See the appendix for R-prec-hard and HUMMUS broken down into categories.

Finally we present human pairwise preference scores between X-LXMERT and DM-GAN (its closest competitor). Here we see that human annotators clearly prefer X-LXMERT to DM-GAN for semantics as well as fidelity.

In summary, X-LXMERT's generation capabilities rival state of the art specialized generation models. In fact, our human studies demonstrate that X-LXMERT produces better results than even DM-GAN, its closest competitor. Our analysis also shows the limitations of current automatic evaluation metric for text-to-image synthesis.

Visual Question Answering Table 1 compares models on the VQA2.0 and GQA datasets. Converting LXMERT to use grid inputs causes a slight or no drop, consistent with findings by Jiang et al. (2020), but hugely simplifies the pipeline. X-LXMERT shows 1.5 - 2.5% drop on these datasets but note that its numbers are still very competitive.

Visual Reasoning Table 1 compares models on NLVR^2 dataset. Consistent with VQA, grid inputs cause a slight drop. X-LXMERT shows a roughly 2% drop but retains most of the massive jumps obtained by LXMERT on NLVR^2 compared to the previous generation of models.

Our implementation of X-LXMERT uses a small 8×8 grid. Increasing the grid size will likely shrink gaps in VQA2.0, GQA and NLVR^2 datasets as per the recent findings by Jiang et al. (2020).

7.2 From X-LXMERT to X-UNITER

The proposed refinements (Sec. 5.1) to enable image generation capabilities are not LXMERT-specific. We apply these changes to UNITER (Chen et al., 2019), a single stream multi-modal transformer architecture. Instead of following (Chen et al., 2019) Table 2 shows that UNITER + Grid produces very poor images, but X-UNITER obtains image generation scores comparable to X-LXMERT– showing the generality of our extensions.

	IS↑	FID↓
UNITER + Grid	2.4	253.5
X-UNITER	**20.1**	**51.4**
LXMERT + Grid	1.6	316.7
X-LXMERT	**22.7**	**37.4**

Table 2: Adding image generation capabilities to LXMERT and UNITER.

7.3 Qualitative Results

Fig 3 shows qualitative examples by X-LXMERT compared to DM-GAN (Zhu et al., 2019). While the images lack fine details, they do a reasonable job at preserving high level semantics, as revealed by the metrics. For complex scene, our model is able to preserve better semantics (e.g. 'two people', 'clock tower' and 'home office') compared to DM-GAN. We do not show images produced by LXMERT since they tend to be incomprehensible.

Figure 4: Intermediate images generated by X-LXMERT at during 140 steps of random position sampling. Images are gradually improved as sampling steps proceed.

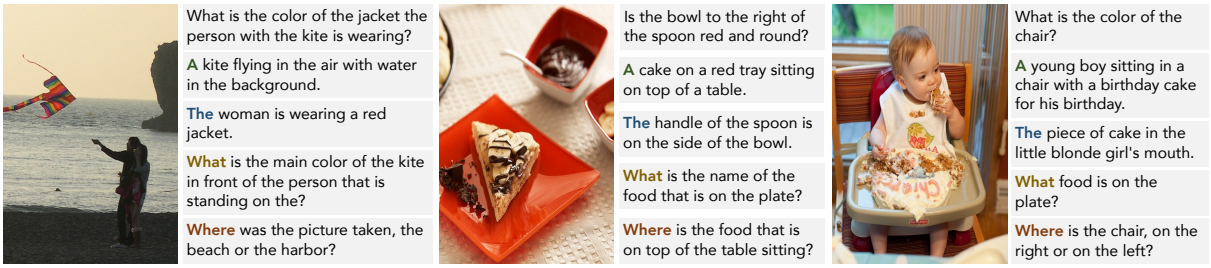

Figure 5: Captions generated by X-LXMERT using Gibbs sampling. We control the samples by providing different prefix word into the model. Those prefix words are common starting word such as '*A*', '*The*', '*What*', '*Where*'.

Ablations	IS↑	FID↓
LXMERT + Grid	1.6	316.7
X-LXMERT	**22.7**	**37.4**
w/o discrete visual representations	1.5	304.4
w/o uniform masking	2.1	227.9
w/o updating pre-training data	21.6	46.1

Table 3: An ablation study for the three refinements.

	IS↑/FID↓	R-prec↑ easy/hard	HUMMUS ↑ Noun / Verb / Color / Avg.
Mask-Pred-4	**22.7**/37.4	**40.8/25.1**	**0.55** / **0.42** / 0.50 / **0.49**
TL→BR	19.8/48.5	26.9/18.9	0.45 / **0.42** / 0.41 / 0.43
Random	22.6/**35.9**	39.5/24.7	0.52 / **0.42** / 0.51 / 0.48
Mask-Pred-1	19.5/51.4	36.8/21.4	0.48 / 0.40 / **0.54** / 0.47

Table 4: Image quality across sampling strategies.

To better understand the image generation process, We show intermediate images generate by X-LXMERTin Fig 4. We use random autoregressive sampling with 140 steps. Interestingly, the model first coarsely generates salient objects (ex. giraffe, monitors) in the caption followed by details and background.

Our model is able to generate captions given image. For each image, we sample text from X-LXMERT using Gibbs sampling as shown in Fig 5. We control the samples by providing different prefix word into the model. Those prefix words are common starting word such as 'A', 'The', 'What', 'Where'. X-LXMERT can produce long meaningful captions as well as questions (like the ones in VQA datasets).

7.4 Ablation Studies

We examine the effects of our proposed refinements and our sampling strategies to the image generation quality. Table 3 shows that two of the proposed refinements to LXMERT (moving to discrete visual representations and using uniform masking) are critical to produce high quality images. The third refinement – updating pre-training data for the CCC objective – is less critical, but useful nonetheless.

Table 4 shows that X-LXMERT is fairly robust to sampling strategy, particularly for image semantics, with the exception of TL→BR which tends to produce worse results. This is interesting in that TL→BR is typically the default strategy used by practitioners (van den Oord et al., 2016, 2017).

8 Conclusion

We develop a probing mechanism and find that LXMERT, a powerful vision-and-language transformer model, is not able to generate meaningful images conditioned on text. We present X-LXMERT, a unified model for image generation, captioning, QA and visual reasoning, and show that our extensions can easily be applied to other vision-and-language transformer models.

References

Chris Alberti, Jeffrey Ling, Michael Collins, and David Reitter. 2019. Fusion of Detected Objects in Text for Visual Question Answering. In *EMNLP*.

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

Hangbo Bao, Li Dong, Furu Wei, Wenhui Wang, Nan Yang, Xiaodong Liu, Yu Wang, Songhao Piao, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2020. UniLMv2: Pseudo-Masked Language Models for Unified Language Model Pre-Training.

Yen-chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2019. UNITER: Learning UNiversal Image-TExt Representations.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. In *NAACL*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified Language Model Pre-training for Natural Language Understanding and Generation. In *NeurIPS*.

Marjan Ghazvininejad, Omer Levy, Yinhan Liu, and Luke Zettlemoyer. 2019. Mask-Predict: Parallel Decoding of Conditional Masked Language Models. In *EMNLP*.

Spyros Gidaris, Andrei Bursuc, Nikos Komodakis, Patrick Pérez, and Matthieu Cord. 2020. Learning Representations by Predicting Bags of Visual Words. In *CVPR*.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative Adversarial Networks. In *NIPS*.

Yash Goyal, Tejas Khot, Aishwarya Agrawal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. 2019. Making the V in VQA Matter: Elevating the Role of Image Understanding in Visual Question Answering. *International Journal of Computer Vision*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep Residual Learning for Image Recognition. In *CVPR*.

Olivier J. Henaff, Ali Razavi, Carl Doersch, Ali S. M. Eslami, and Aaron van den Oord. 2019. Data-Efficient Image Recognition with Contrastive Predictive Coding.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian Error Linear Units (GELUs).

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. 2017. GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium. In *NIPS*.

Seunghoon Hong, Dingdong Yang, Jongwook Choi, and Honglak Lee. 2018. Inferring Semantic Layout for Hierarchical Text-to-Image Synthesis. In *CVPR*.

Xun Huang and Serge Belongie. 2017. Arbitrary Style Transfer in Real-time with Adaptive Instance Normalization. In *ICCV*.

Zhicheng Huang, Zhaoyang Zeng, Bei Liu, Dongmei Fu, and Jianlong Fu. 2020. Pixel-BERT: Aligning Image Pixels with Text by Deep Multi-Modal Transformers.

Drew A. Hudson and Christopher D. Manning. 2019. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *CVPR*.

Gabriel Ilharco, Rowan Zellers, Ali Farhadi, and Hannaneh Hajishirzi. 2020. Probing Text Models for Common Ground with Visual Representations.

Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. 2020. In Defense of Grid Features for Visual Question Answering. In *CVPR*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*.

Justin Johnson, Alexandre Alahi, and Li Fei-Fei. 2016. Perceptual Losses for Real-Time Style Transfer and Super-Resolution. In *ECCV*.

Tero Karras, Samuli Laine, and Timo Aila. 2019. A Style-Based Generator Architecture for Generative Adversarial Networks. In *CVPR*.

Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and Improving the Image Quality of StyleGAN. In *CVPR*.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A Method for Stochastic Optimization. In *ICLR*.

Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li Jia-Li, David Ayman Shamma, Michael Bernstein, and Li Fei-Fei. 2016. Visual Genome: Connecting Language and Vision Using Crowdsourced Dense Image Annotations. *International Journal of Computer Vision*.

Bowen Li, Xiaojuan Qi, Thomas Lukasiewicz, and Philip H S Torr. 2019. Controllable Text-to-Image Generation. In *NeurIPS*.

Gen Li, Nan Duan, Yuejian Fang, Daxin Jiang, and Ming Zhou. 2020. Unicoder-VL: A Universal Encoder for Vision and Language by Cross-modal Pre-training. In *AAAI*.

Yi Liao, Xin Jiang, and Qun Liu. 2020. Probabilistically Masked Language Model Capable of Autoregressive Generation in Arbitrary Word Order. In *ACL*.

Jae Hyun Lim and Jong Chul Ye. 2017. Geometric GAN. In *NIPS*.

Chieh Hubert Lin, Chia-Che Chang, Yu-Sheng Chen, Da-Cheng Juan, Wei Wei, and Hwann-Tzong Chen. 2019. COCO-GAN : Generation by Parts via Conditional Coordinating. In *ICCV*.

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 *ECCV*.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *ICLR*.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. ViLBERT: Pretraining Task-Agnostic Visiolinguistic Representations for Vision-and-Language Tasks. In *NeurIPS*.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-Task Vision and Language Representation Learning. In *CVPR*.

Elman Mansimov, Alex Wang, and Kyunghyun Cho. 2019. A Generalized Framework of Sequence Generation with Application to Undirected Sequence Models.

Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. 2018. Spectral Normalization for Generative Adversarial Networks. In *ICLR*.

Takeru Miyato and Masanori Koyama. 2018. cGANs with Projection Discriminator. In *ICLR*.

Anh Nguyen, Jeff Clune, Yoshua Bengio, Alexey Dosovitskiy, and Jason Yosinski. 2017. Plug & Play Generative Networks: Conditional Iterative Generation of Images in Latent Space. In *CVPR*.

Mehdi Noroozi and Paolo Favaro. 2016. Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles. In *ECCV*.

Augustus Odena, Christopher Olah, and Jonathon Shlens. 2017. Conditional Image Synthesis With Auxiliary Classifier GANs. In *ICML*.

Aaron van den Oord, Nal Kalchbrenner, and Koray Kavukcuoglu. 2016. Pixel Recurrent Neural Networks. In *ICML*.

Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural Discrete Representation Learning. In *NIPS*.

Aaron Van Den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation Learning with Contrastive Predictive Coding.

Taesung Park, Ming-yu Liu, Ting-chun Wang, and Junyan Zhu. 2019. Semantic Image Synthesis with Spatially-Adaptive Normalization. In *CVPR*.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chana, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in PyTorch. In *NIPS Workshop*.

Di Qi, Lin Su, Jia Song, Edward Cui, Taroon Bharti, and Arun Sachet. 2020. ImageBERT: Cross-modal Pre-training with Large-scale Weak-supervised Image-Text Data.

Wasifur Rahman, Md Kamrul Hasan, Amir Zadeh, Louis-Philippe Morency, and Mohammed Ehsan Hoque. 2020. M-BERT: Injecting Multimodal Information in the BERT Structure. In *ACL*.

Ali Razavi, Aaron van den Oord, and Oriol Vinyals. 2019. Generating Diverse High-Fidelity Images with VQ-VAE-2. In *NeurIPS*.

Scott Reed, Zeynep Akata, Xinchen Yan, Lajanugen Logeswaran, Bernt Schiele, and Honglak Lee. 2016. Generative adversarial text to image synthesis. In *ICML*.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. 2015. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In *NIPS*.

Tim Salimans, Ian Goodfellow, Wojciech Zaremba, Vicki Cheung, Alec Radford, and Xi Chen. 2016. Improved Techniques for Training GANs. In *NIPS*.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. 2019. VL-BERT: Pre-training of Generic Visual-Linguistic Representations. In *ICLR*.

Alane Suhr, Stephanie Zhou, Ally Zhang, Iris Zhang, Huajun Bai, and Yoav Artzi. 2019. A Corpus for Reasoning About Natural Language Grounded in Photographs. In *ACL*.

Chen Sun, Austin Myers, Carl Vondrick, Kevin Murphy, and Cordelia Schmid. 2019. VideoBERT: A Joint Model for Video and Language Representation Learning. In *ICCV*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2016. Rethinking the Inception Architecture for Computer Vision. In *CVPR*.

Fuwen Tan, Song Feng, and Vicente Ordonez. 2019. Text2Scene: Generating Compositional Scenes From Textual Descriptions. In *CVPR*.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning Cross-Modality Encoder Representations from Transformers. In *EMNLP*.

Dustin Tran, Rajesh Ranganath, and David M. Blei. 2017. Hierarchical implicit models and likelihood-free variational inference. In *NIPS*.

Trieu H. Trinh, Minh-Thang Luong, and Quoc V. Le. 2019. Selfie: Self-supervised Pretraining for Image Embedding.

Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. 2016. Instance Normalization: The Missing Ingredient for Fast Stylization.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *NIPS*.

Alex Wang and Kyunghyun Cho. 2019. BERT has a Mouth, and It Must Speak: BERT as a Markov Random Field Language Model. In *NAACL Workshop*.

Ting-chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. 2018. High-Resolution Image Synthesis and Semantic Manipulation with Conditional GANs. In *CVPR*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. 2019. HuggingFace's Transformers: State-of-the-art Natural Language Processing.

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V. Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Łukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation.

Tao Xu, Pengchuan Zhang, Qiuyuan Huang, Han Zhang, Zhe Gan, Xiaolei Huang, and Xiaodong He. 2018. AttnGAN: Fine-Grained Text to Image Generation with Attentional Generative Adversarial Networks. In *CVPR*.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, and Senior Member. 2018. StackGAN++ : Realistic Image Synthesis with Stacked Generative Adversarial Networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–16.

Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. 2017. StackGAN : Text to Photo-realistic Image Synthesis with Stacked Generative Adversarial Networks. In *ICCV*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason J. Corso, and Jianfeng Gao. 2020. Unified Vision-Language Pre-Training for Image Captioning and VQA. In *AAAI*.

Minfeng Zhu, Pingbo Pan, Wei Chen, and Yi Yang. 2019. DM-GAN: Dynamic memory generative adversarial networks for text-to-image synthesis. In *CVPR*.

Yuke Zhu, Oliver Groth, Michael Bernstein, and Li Fei-Fei. 2016. Visual7W: Grounded Question Answering in Images. In *CVPR*.

A Qualitative samples

More qualitative samples In Fig 6, we show more qualitative examples of images generated by DM-GAN, reconstruction from ground truth clusters, LXMERT, our proposed X-LXMERT with different sampling strategies. Fig 7 shows images generated by X-LXMERT with the same subject placed in a variety of contexts.

B Source code

Please refer to the project page for more details about this research, at https://prior.allenai.org/projects/x-lxmert. This includes an animation of the iterative image generation process, a demo of X-LXMERT accessible at https://vision-explorer.allenai.org/text_to_image_generation and code available at https://github.com/allenai/x-lxmert.

C LXMERT / X-LXMERT details

For a fair comparison, we re-implement LXMERT and LXMERT with grid features. Our models have 226.5M trainable parameters, slightly smaller than 228M of original LXMERT implementation due to weight sharing of MVFR head and MOC head. We use PyTorch (Paszke et al., 2017) and Huggingface Transformers (Wolf et al., 2019) libraries for implementation.

C.1 LXMERT Architecture

LXMERT architecture consists of text embedder, object embedder, transformer backbone, and task-specific heads.

Text embedder A text input is tokenized by WordPiece Tokenizer (Wu et al., 2016) and special tokens CLS and EOS are concatenated: $\{\texttt{CLS}, w_1, \ldots, w_T, \texttt{EOS}\}$. We use the same vocabulary used in BERT[4] and LXMERT with size 30522. Text is truncated with maximum token length of 20, including two special tokens. 768-dimensional embedding is learned for each token and position. Final text embedding is obtained by sum of token embedding and positional embedding.

Object embedder An input image is resized within minimum length 800 and maximum length 1333 while preserving aspect ratio. We use Faster R-CNN trained on Visual Genome to extract 36 bounding boxes from each image[5]. We take fc6 feature, which is between RoI-Pool layer and final object classification head and has 2048 dimension. This is encoded into 768 dimensional vector followed by layer norm (Ba et al., 2016). Four bounding box coordinates (x_0, x_1, y_0, y_1) are $[0, 1]$-normalized by width and height. Then they are also encoded into 768 dimensional vectors with fully connected layer followed by layer norm. Final object embedding is obtained by element-wise average of object and positional feature.

Transformer backbone Transformer backbone of LXMERT consists of object relation encoder, language encoder and cross modality encoder, which are composed of 9 self-attention layer (Vaswani et al., 2017), 5 self-attention layer, and 5 cross-attention layer respectively. The self-attention layers are same as the ones used in BERT and the dimension of the layers is 768.

Task-specific heads LXMERT is pretrained with five objectives[6] (MLM, MVFR, MOC, ITM, QA) as explained in Sec. 3. For MLM, MVFR, ITM, QA task, a task head consisting of two fully connected layers with GeLU activation (Hendrycks and Gimpel, 2016) and layer norm is trained. For MOC task, a fully connected layer is applied on ouput of MVFR head, similar to original object detection pipeline[7]. For MLM, MVFR, MOC tasks, task heads are applied on cross-modal encoder outputs corresponding to masked tokens. For ITM, QA tasks, tasks heads are applied on CLS token.

C.2 X-LXMERT Architecture

X-LXMERT shares most components with LXMERT, except for minor modifications below.

Object embedder → Grid embedder We extract 8×8 grid features of fc6 layer of Faster R-CNN, by giving positional information of 8×8 grids into RoI-Pool layer. Then we quantize these features with nearest neighborhood search from 10,000 cluster centroids. Remaining are same with object embedder of LXMERT.

[4] bert-base-uncased

[5] We use PyTorch version (https://gitlab.com/vedanuj/vqa-maskrcnn-benchmark), instead of Caffe version (https://github.com/peteanderson80/bottom-up-attention) used in original implementation.

[6] We do not use 400 object attributes predicted from Faster R-CNN, which were used by original implementation.

[7] Original implementation trains separate head for MOC task.

Figure 6: More qualitative examples of images generated by X-LXMERT.

Figure 7: Images generated by X-LXMERT demonstrating its ability to place objects within varied contexts.

MOC, MVFR tasks → CCC task We replace MOC, MVFR tasks with CCC task (see Sec. 5.1) for X-LXMERT. For CCC head, we simply modify the output dimension of fully connected layer used in MOC task to the number of clusters (1600 → 10000).

C.3 Datasets

For pretraining, we use same datasets used in LXMERT. We use vision-and-language datasets whose images come from MS COCO (Lin et al., 2014) or Visual Genome (Krishna et al., 2016). Besides the two original captioning datasets, we also aggregate three large image question answering (image QA) datasets: VQA v2.0 (Goyal et al., 2019), GQA balanced version (Hudson and Manning, 2019), and VG-QA (Zhu et al., 2016). Table 5 shows statistics of the datasets. Note that X-LXMERT only uses COCO captions for CCC task.

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 5: Dataset statistics used in pretraining. 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. This table is from LXMERT (Tan and Bansal, 2019).

C.4 Visual vocabulary clustering

To create visual vocabularies, we run K-means clustering on Faster R-CNN grid features of COCO `train2014` images. `train2014` has 82783 images, resulting 8 x 8 x 82783 = 5.3M grid features. We use FAISS (Johnson et al., 2017) library for clustering. We sample 2.6M features in training data and run 20 iteration, which takes 2 hours.

C.5 Training

We train LXMERT and X-LXMERT for 20 epochs with mixed precision using Apex[8] (opt-level `O1`). We use AdamW optimizer (Loshchilov and Hutter, 2019) with $(\beta^1, \beta^2) = (0.9, 0.999)$ and learning rate `1e-5` with 5% linear warmup schedule. We use gradient clipping with maximum norm 1.

Instead of using all pretraining tasks for each step, we first uniformly sample a modality to mask from `[image, text, no-mask]` and run corresponding tasks similar to (Chen et al., 2019; Lu et al., 2020). When `image` is selected, we use MVFR, MOC for LXMERT and CCC for X-LXMERT. When `text` is selected, we use MLM. When `no-mask` is selected, we replace given text with a random sentence from training data with 0.5 probability. If the text is replaced, we use ITM. If not, we use ITM and QA.

Training LXMERT takes 60 hours with batch size 1280, and training X-LXMERT takes 40 hours with batch size 920. We use 4 Titan RTX GPUs ($4 \times$ 24GB) for training both models.

C.6 Finetuning

During finetuning on VQA/GQA/NLVR^2, a task head consisting of two fully connected layers with GeLU activation and layer norm is trained along with pre-trained LXMERT and X-LXMERT. For VQA/GQA, the parameters are initialized from pretrained QA head. We use AdamW optimizer with learning rate `5e-4`. We train LXMERT and X-LXMERT for 10 epochs for each task. For VQA/GQA/NLVR^2, finetuning takes 3/5/1 hours respectively on 4 Titan RTX GPUs ($4 \times$ 24GB).

[8] https://github.com/NVIDIA/apex

D Generator details

Our image generation system adopts GAN (Goodfellow et al., 2014) framework and has two networks trained: generator and discriminator.

D.1 Generator Architecture

Our generator consists of multiple residual blocks following SNGAN (Miyato and Koyama, 2018). The generator takes (quantized) 8×8 grid features of Faster R-CNN as input and outputs 256×256 RGB images. We use a generator with 5 residual blocks, where each block bilinearly-upsamples feature map by 2. We use 32 channels of 3x3 kernel for every convolution layer in residual blocks. Note that many existing generator architectures (Miyato and Koyama, 2018; Wang et al., 2018; Karras et al., 2019, 2020) have residual blocks starting from higher dimensions (eg. 512, 1024) in low-resolution then gradually decrease the dimension as feature maps are spatial upsampled. However, we found that using fixed-sized small dimension for all residual blocks makes training more stable. Each residual block has spatially adaptive instance norm (SPADE) (Park et al., 2019; Huang and Belongie, 2017) that guides the residual block using spatial information of 8×8 grid features. After each spatially adaptive instance norm, we multiply spatial gaussian noise on feature maps to make model less focus on local texture following StyleGAN (Karras et al., 2019). We use spectral normalization (Miyato et al., 2018) after each convolution layer in generator. Following StyleGAN-v2 (Karras et al., 2020), we use skip connection for each

residual block to generate final output. Our generator has 1.7M trainable parameters. The detailed architecture of our generator is illustrated at Fig. 8.

D.2 Discriminator Architecture

Discriminator also consists of multiple residual blocks. We use a discriminator with 5 residual blocks, where each residual block downsamples feature map by 2. We use 64 channels of 3x3 kernel for every convolution layer in residual blocks. We use spectral normalization after each convolution layer in discriminator. In contrasts to generator, discriminator (1) uses instance norm (Ulyanov et al., 2016) instead of adaptive instance norm, (2) does not gaussian noise multiplication and (3) does not use skip connection. Output of the 5 residual blocks are 8×8 feature map. Our discriminator have two heads taking these feature maps: (1) adversarial head spatially averaging 8×8 feature map and predicting whether input image is from original image domain or not and (2) classification head predicting cluster ids of 8×8 spatial layouts from input image. Our discriminator has 0.5M trainable parameters. The detailed architecture of our discriminator is illustrated at Fig. 9.

D.3 Dataset

We train our model on COCO `train2014` split, which consits of 82783 images.

D.4 Training

Our generator and discrminator are trained with 4 losses: (1) hinge adversarial loss (Lim and Ye, 2017; Tran et al., 2017), (2) AC-GAN loss (Odena et al., 2017), (3) discriminator feature match loss (Wang et al., 2018) and (4) perceptual loss (Johnson et al., 2016) following (Park et al., 2019). Following pix2pixHD (Wang et al., 2018), coefficients for the losses are (1, 1, 10, 10) respectively. Adversarial loss guides generator to output images close to original images. The rest of the losses guide generator to output images close to specific target images using spatial layout inputs. We use ResNet-50 (He et al., 2016) for perceptual loss. Detail of losses are explained in Sec. D.5.

We use Adam optimizer (Kingma and Ba, 2015) with $(\beta^1, \beta^2) = (0, 0.999)$ and two-time update rule (Heusel et al., 2017) with learning rate of 0.0004 and 0.0001 for generator and discriminator respectively. We train the image generator for 60 epochs with batch size 96. Training takes 15 hours on 8 NVIDIA Titan V GPUs ($8 \times$ 12GB).

D.5 Losses

In below equations, $\hat{X}$ and X refer to generated image and target image respectively.

Adversarial loss

$$L^G_{adv} = -D^{adv}(\hat{X}) \tag{1}$$

$$\begin{aligned} L^D_{adv} &= max(1 - D^{adv}(\hat{X}), 0) \\ &\quad + max(1 - D^{adv}(X), 0) \end{aligned} \tag{2}$$

where D^{Adv} is discriminator adversarial head.

AC-GAN loss

$$\begin{aligned} L_{ACGAN} = &-\frac{1}{N^2}\sum_{h,w} \log P(D^{cls}_{h,w}(\hat{X})) \\ &-\frac{1}{N^2}\sum_{h,w} \log P(D^{cls}_{h,w}(X)) \end{aligned} \tag{3}$$

where D^{cls} is discriminator classification head.

Discriminator feature match loss

$$L^G_{FM} = \sum_k \frac{1}{H^k W^k C^k} \sum_{h,w,c} \ell_{huber} |D^k(\hat{X}) - D^k(X)| \tag{4}$$

where

$$\ell_{huber}(x) = \begin{cases} 0.5 * x^2, & if\ |x| \le 1 \\ |x| - 0.5, & otherwise \end{cases}$$

and D^k is discriminator's k-th resblock.

Perceptual loss

$$L^G_{FM-E} = \sum_k \frac{1}{H^k W^k C^k} \sum_{h,w,c} \ell_{huber} |E^k(\hat{X}) - E^k(X)| \tag{5}$$

where E^k is ResNet-50 (He et al., 2016)'s k-th resblock (`conv2_x`, `conv3_x`, `conv4_x`, `conv5_x`).

Total loss

$$\begin{aligned} L^G = &\lambda_{adv} * L^G_{adv} \\ &+ \lambda_{ACGAN} * L_{ACGAN} \\ &+ \lambda_{FM} * L^G_{FM} \\ &+ \lambda_{FM-E} * L^G_{FM-E} \end{aligned} \tag{6}$$

$$\begin{aligned} L^D = &\lambda_{adv} * L^D_{adv} \\ &+ \lambda_{ACGAN} * L_{ACGAN} \end{aligned} \tag{7}$$

where $(\lambda_{GAN}, \lambda_{ACGAN}, \lambda_{FM}, \lambda_{FM-E}) = (1, 1, 10, 10)$.

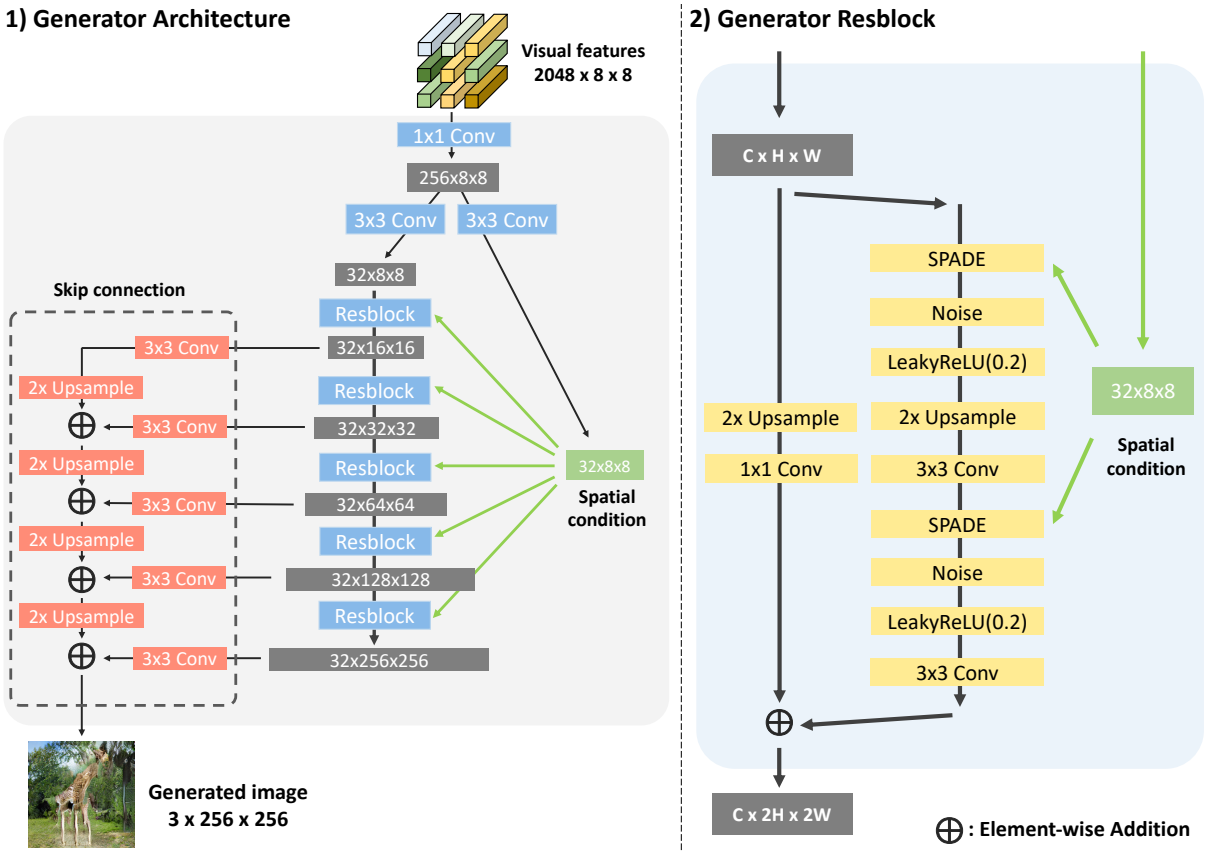

Figure 8: Generator architecture that takes 8x8 grid visual features and generates 256x256 images.

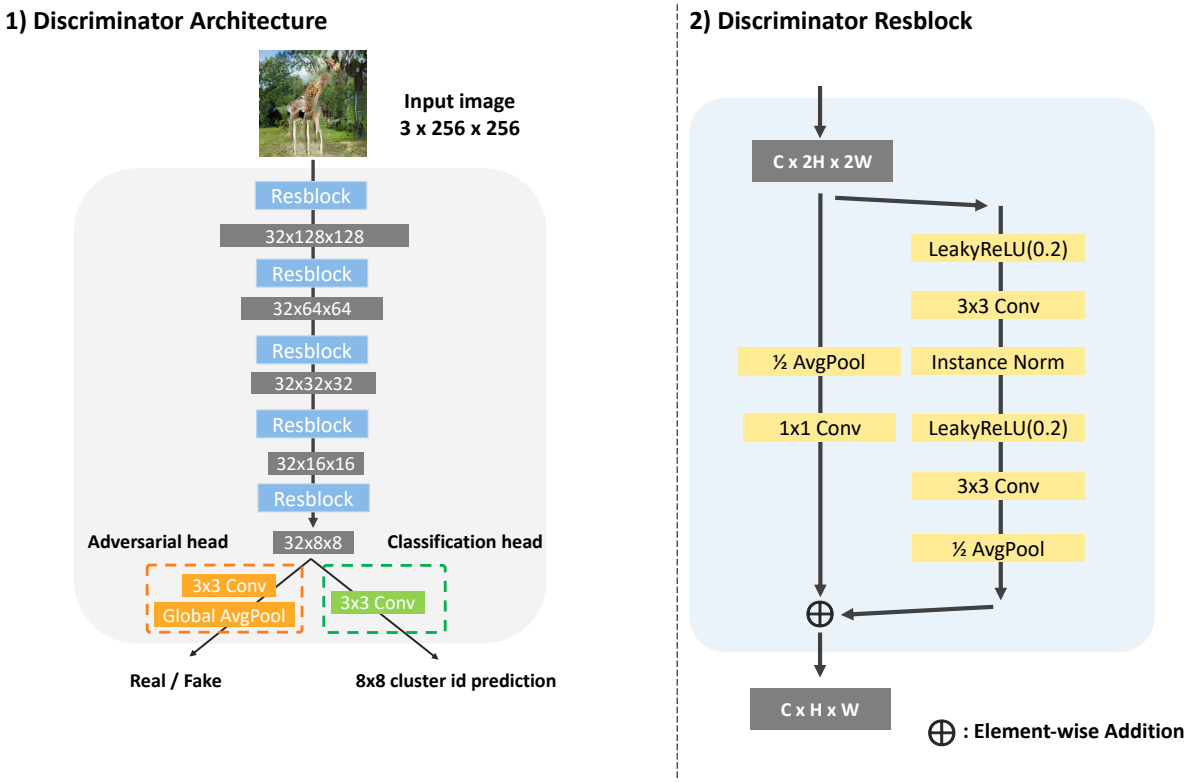

Figure 9: Discriminator architecture that takes 256x256 images.

E Evaluation details

E.1 Image metrics

To calculate image metrics, we follow Xu et al. (2018) and randomly sample 30000 images from MS COCO `val2014` split and sample a caption for each image. Then we generate images from those 30000 captions for each method. We use subset of these 30000 captions for automatic image evaluation.

Inception Score (IS) Following Zhu et al. (2019), we use all 30000 generated images. We use OpenAI implementation[9] to calculate IS.

Fréchet Inception Distance (FID) Following Zhu et al. (2019), we use all 30000 generated images. We use PyTorch port of official implementation[10] to calculate FID.

R-precision-easy We use all 30000 generated images. For R-precision-easy, we sample 99 negative captions for each caption, where all negative captions correspond to different `val2014` images.

R-precision-hard For each R-precision-hard category (noun/verb/color/number), we use 1000 randomly sampled caption that contains a category word. Then we generate 9 negative captions by swapping the detected category word with another word with same category. We use POS-tagging with spaCy[11] to find category words from a caption. We present per-category score of R-precision-hard at table 6.

E.2 Human evaluation

We use Amazon Mechanical Turk[12] for human evaluation.

HUMMUS score For each HUMMUS category (noun/verb/color), we use 100 randomly sampled images. Then we mask out words in the same fashion as in R-precision-hard metric. A total of 280 unique crowdworkers completed the task, with a median of 13 images annotated per worker. We present per-category score of HUMMUS score at table 7. Fig 10 shows screenshot of HUMMUS score (noun category) evaluation task.

Pairwise preference For *Semantic preference* task, we ask annotators (1) 'Which image best matches the caption?' with caption. For *Fidelity preference* task, we ask annotators 'Which image looks more realistic?' without providing the caption. A total of 357 unique crowdworkers completed the task, with a median of 17 annotations performed per worker.

Fig 11 shows screenshot of Semantic preference evaluation task, and Fig 12 shows screenshot of Fidelity preference evaluation task.

[9] https://github.com/openai/improved-gan/tree/master/inception_score

[10] https://github.com/mseitzer/pytorch-fid/tree/802da3963113b5b5f8154e0e27580ee4c97460ab

[11] https://spacy.io/

[12] https://www.mturk.com/

	R-precision-hard↑	R-precision-hard categories ↑			
		Noun	Verb	Color	Number
Original Image	47.6	80.4	25.3	53.4	31.4
DM-GAN (Zhu et al., 2019)	27.5	48.9	9.5	35.8	15.7
LXMERT	6.6	5.6	1.7	10.2	8.7
X-LXMERT	25.1	41.4	9.8	30.7	18.5
X-LXMERT sampling variations:					
Autoregressive					
TL→BR	18.9	31.6	7.3	21.0	15.5
Random	24.7	41.2	10.1	28.8	18.7
Random-200	23.3	41.2	10.1	26.5	16.5
Easy-First	22.0	35.6	8.1	25.3	18.9
Parallel					
Mask-Predict-1	21.4	35.2	7.7	29.8	12.7
Mask-Predict-4 = **X-LXMERT**	25.1	41.4	9.8	30.7	18.5
Mask-Predict-10	22.6	37.3	10.0	26.1	16.9

Table 6: R-precision-hard per-category scores

	HUMMUS↑	HUMMUS Categories↑		
		Noun	Verb	Color
Original Image	0.73	0.79	0.52	0.89
DM-GAN	**0.49**	0.42	**0.45**	**0.60**
LXMERT	0.27	0.16	0.43	0.21
X-LXMERT	**0.49**	**0.55**	0.42	0.50
X-LXMERT sampling variations:				
TL→BR	0.43	0.45	0.42	0.41
Random	0.48	0.52	0.42	0.51
Mask-Predict-1	0.47	0.48	0.40	0.54

Table 7: Evaluating semantics with HUMMUS.

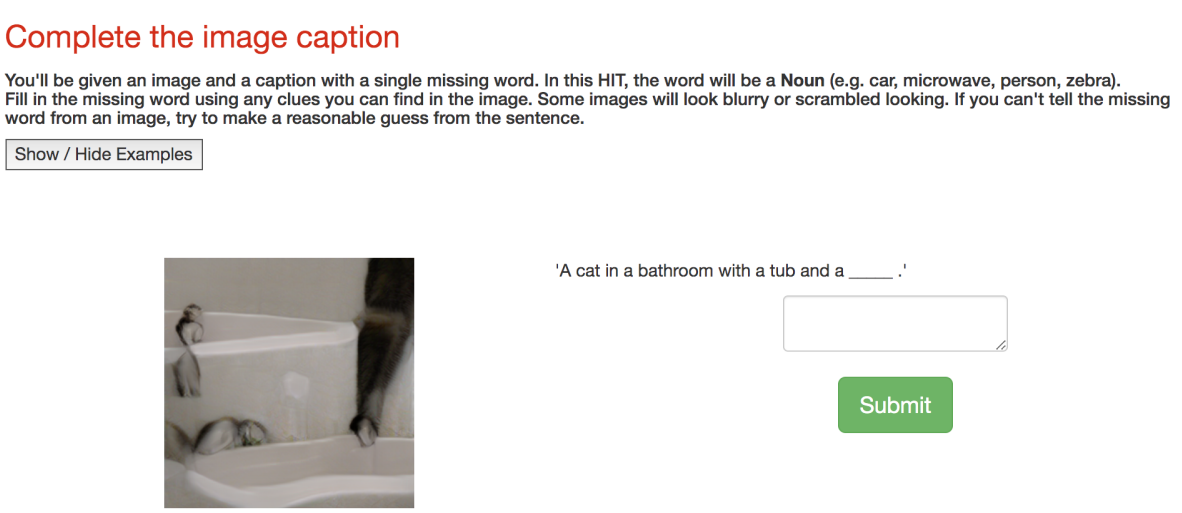

Figure 10: Screenshot of HUMMUS score evaluation system

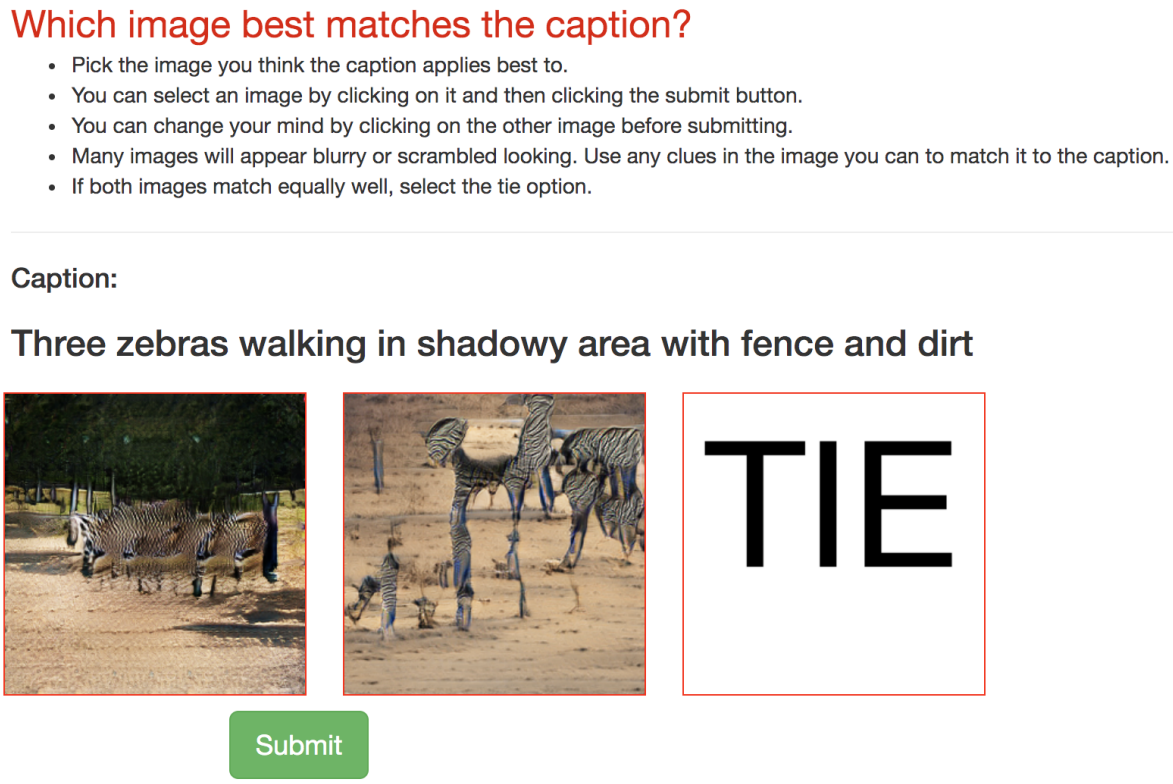

Figure 11: Screenshot of Semantic preference evaluation system

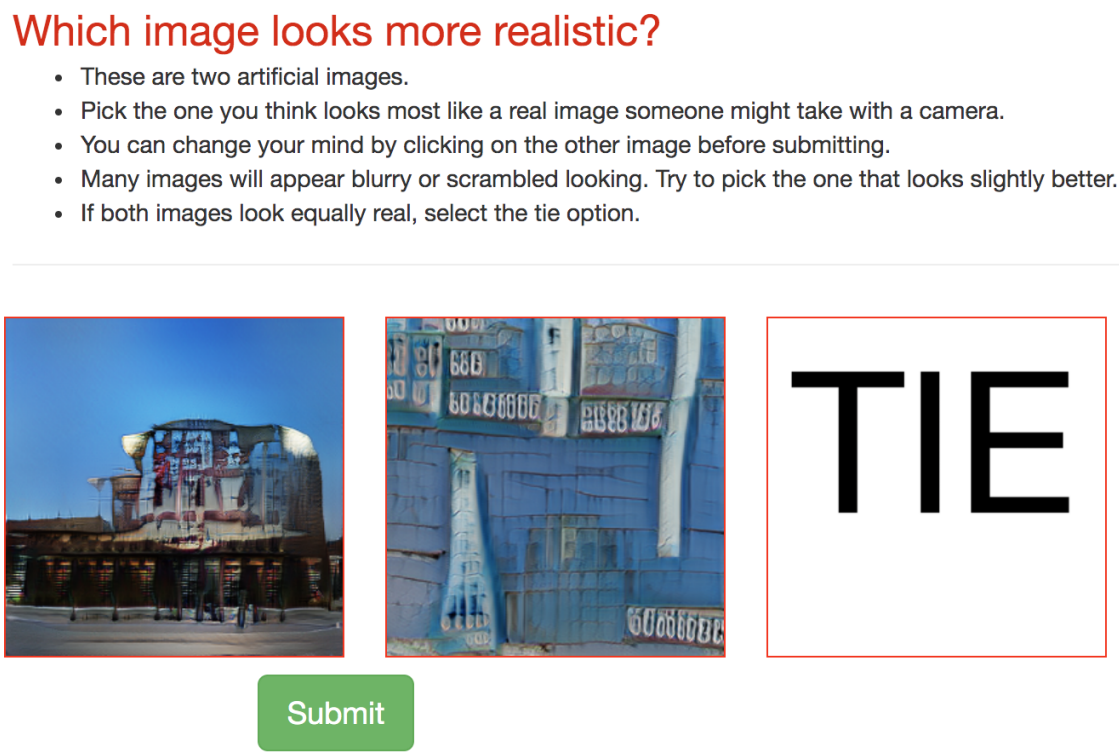

Figure 12: Screenshot of Fidelity preference evaluation system