KRISP: Supplemental Material

Kenneth Marino[1,2] Xinlei Chen[2] Devi Parikh[2,3] Abhinav Gupta[1,2] Marcus Rohrbach[2]

[1]Carnegie Mellon University [2]Facebook AI Research [3]Georgia Tech

1. Methodology Additional Details

1.1. Knowledge Graph Construction

Here we provide additional details on our knowledge graph construction.

DBPedia First we extracted DBPedia [1]. DBPedia is actually a set of datasets collected from Wikipedia articles and tables. For our knowledge graph we used the October 2016 crawl of Wikipedia.[1] For our DBPedia edges we used the following files: Article Categories, Category Labels, DBPedia Ontology, Instance Types, Instance Types Sdtyped Dbo, Mappingbased Objects, and Person Data.

Next we wrote string parsers and regular expressions to translate these triplets into lowercase multi-word english expressions. This involved extracting the category words from the hyperlink: e.g., "`<http://dbpedia.org/resource/Tadeusz_Borowski>`" would be extracted as "tadeusz borowski". We will provide the parsing code and the processed final dbpedia files when we release the code for this paper.

Before final filtering, this knowledge source contains $24,685,703$ edges.

VisualGenome As we say in the main text we collect a knowledge graph on VisualGenome [9] by taking the most common edges in the scene graphs. We first create a split of VisualGenome. So that this graph is maximally useful down the road, we take a split that only contains the intersection of COCO [12] train, VisualGenome train, and LVIS [5] train so that the graph can safely be used for any of these datasets on COCO. This also means that this split does not contain and of OK-VQA [15] test set images.

For the remaining images, we take any edge which appear at least 50 times in that set and add to our list.

Before final filtering, this knowledge source contains $3,326$ edges.

hasPart KB / ConceptNet These two knowledge sources were already in a fairly processed state, so no additional processing was necessary before our task-specific filtering. hasPart KB [2] was directly downloaded from source website.

ConceptNet [13] was from the training data used for [11] which has already been processed.[2]

hasPart KB contained $49,848$ edges and ConceptNet contained $102,400$.

Combining and Filtering To combine and filter these four knowledge bases into one graph, the first step was to simply combine all of the knowledge triplets from the four knowledge sources. Then, we removed all stop word concepts (e.g. is, the, a) from the knowledge graph to avoid non-meaningful edges.

Next, as we discuss in the main text we collect all of the symbolic entities from the dataset (question, answers and visual concepts) and then include edges that only include these concepts. We also limit the number of 25 edge types that are the most common and useful for our end task, shown in Fig.2 of the main text.

The final graph is $36,199$ edges, 7643 nodes and 25 edge types.

We will release our processed knowledge graphs with the source code.

1.2. Image Symbols

To get our image symbols, as we say in the main paper, we run four classifiers and detectors on our dataset. The classifiers/detectors we use are the following.

1. A ResNet-152 [6] trained on ImageNet [17]. Implementation from default PyTorch [16] nn library.
2. A ResNet-18 trained on Places365 [20] using that publication's released code.
3. A Faster R-CNN trained on Visual Genome [9] using the baseline from [7].
4. An EQL loss [19] -trained Mask R-CNN model on LVIS (v1.0) [5] using the code from [19].

[1] https://wiki.dbpedia.org/downloads-2016-10

[2] https://ttic.uchicago.edu/~kgimpel/resources.html

Dataset	# Symbols
ImageNet	1000
VisualGenome	1600
LVIS	1203
Places	365

Table 1. Multi-modal BERT Hyperparameters

In Table 1 we show the number of symbols in each of these datasets.

1.3. Answer vocab

For our answer vocab, we take any answer that appears in the training set at least 10 times in the answer annotations. For OKVQA v1.0, our vocabulary size is 2253 and on v1.1 it is 2250.

1.4. Graph Network to Multi-modal BERT Baseline

Here we more fully describe one of our baselines where we feed the graph network into Multi-modal BERT without making a separate prediction.

First, the graph network forward prediction to G is the same as in Sec. 3.2 of the main paper except without the $z^{implicit}$ input as this would make a circular connection between the graph network and MMBERT. So we take the input symbols and word2vec and we use the graph convolution layers $H^{(l+1)} = f(H^{(l)}, KG)$ where KG is the knowledge graph. As before we end up with $H^{(L)} = G$ which is a $\mathbb{R}^{n \times d_h}$ matrix which corresponds to having a hidden state of size f_h for each node (and therefore concept) in our graph.

Next, we summarize all of these separate hidden states $z_i^{symbolic}$ for each node i in the graph. We do this by adding a dummy node and dummy edge type to the input graph where each node in the graph is connected to the dummy node by this dummy edge type. The idea is that we create a special edge type that will try to "summarize" the information from all graph hidden states and pass it to this dummy node. We then perform one final RGCN conv layer: $H^{(Summary)} = f(G, KG)$, and extract the hidden state for the dummy node $z_{dummy}^{symbolic}$ or $z_{summary}^{symbolic}$.

With this summary embedding $z_{summary}^{symbolic}$, we then add this summary vector as an additional input to the MMBERT model. We compute a linear embedding layer for this input to processes the graph summary vector and make it the same input size as the other transformer inputs. We then append this to the inputs of the MMBERT.

We tried other methods to get a single vector representation for the graph network, including a self-attention mechanism, and the self-attention mechanism for only these subset of hidden states (only question and image nodes, only answer nodes etc.). All of these performed worse than this particular way of summarizing the graph network output into one vector.

2. Network / Training Hyperparameters

Here we record the network and training parameters. In Table 2 we show the network parameters for the MMBERT baseline and subpart. In Table 3 we show the network parameters for the Graph Network. And in Table 4 we show the training meta-parameters used to train all models.

Parameter	Value
Hidden Size	768
Visual Embedding Dim	2048
Num Hidden	12
Num Attention Heads	12
Hidden Dropout Prob	0.1
Transfer function	ReLU
BERT model name	bert-base-uncased

Table 2. Multi-modal BERT Hyperparameters

Parameter	Value
Node Hidden Size	128
Num Conv Layers	2
Graph Conv Type	RGCN
Transfer function	ReLU
Multi-modal BERT input compress dim	128

Table 3. Graph Network Hyperparameters

Parameter	Value
Optimizer	AdamW [8]
Scheduler	Warmup Cosine
Batch Size	56
Learning Rate	$5e-5$
Eps	$1e-8$
Weight Decay	0
Warmup Steps	2000
Training Steps	88000

Table 4. Training Hyperparameters

3. Variance Values for Tables

Here we show the sample standard deviations for the runs in our tables in Table 5 and Table 6.

	Method	accuracy	std
1.	KRISP (ours)	**32.31**	0.24
	Ablation of Symbolic Knowledge		
2.	MMBERT	29.26	0.76
3.	KRISP w/ random graph	30.15	0.17
	Ablation of Implicit Knowledge		
4.	KRISP w/o BERT pretrain	26.28	0.20
5.	MMBERT w/o BERT pretrain	21.82	0.34
	Ablation of Network Architecture		
6.	KRISP no late fusion	31.10	0.12
7.	KRISP no MMBERT input	31.10	1.41
8.	KRISP no MMBERT input or late fusion	25.00	1.83
9.	KRISP no backprop into MMBERT	27.98	1.23
10.	KRISP with GCN	30.58	0.52
11.	KRISP feed graph into MMBERT	30.99	0.16
	Ablation of Graph Inputs		
12.	KRISP no Q to graph	31.74	0.31
13.	KRISP no I to graph	31.59	0.34
14.	KRISP no symbol input	30.26	1.30
15.	KRISP no w2v	31.95	0.12

Table 5. KRISP ablation on OK-VQA v1.1, with sample standard deviations. Mirrors Table 2 in the main text.

	Method	accuracy	std
1.	KRISP $\max(y^{implicit}, y^{symbolic})$ (ours)	**32.31**	0.24
2.	KRISP $y^{implicit}$	31.47	0.05
3.	KRISP $y^{symbolic}$	29.36	0.50
4.	KRISP no backprop $y^{implicit}$	28.19	1.17
5.	KRISP oracle$(y^{implicit} \mid y^{symbolic})$	36.71	0.29

Table 6. KRISP Subpart Analysis on OK-VQA v1.1, with sample standard deviations. Mirrors Table 3 in the main text.

Method	accuracy
KRISP	**32.31**
KRISP Instance graph	31.98

Table 7. KRISP versus instance graph KRISP.

4. Asymptotic analysis and instance graph ablation

Instance knowledge graph The main paper used a fixed size knowledge graph for the experiments, but this was a choice of convenience rather than an inherent limitation of the method.

To show this, we re-ran KRISP where we filter the knowledge graph dynamically to only include the sub-graph relevant to the input instance. We get a similar final performance. See Table 7.

Graph asymptotic size We also note that, for the purpose of dataset scaling, that while the instance graph method results in a constant graph size w.r.t. dataset size, even for fixed graphs, the graph grows less quickly than linear time. Because many concepts are common, as we increase the number of questions in the dataset, we add fewer and fewer new edges to the overall graph. We tested this with VQAv2, which is $\sim$80x bigger, the number of edges in the graph only increased from $36,199$ to $61,327$ ($\sim$2x more).

Runtime/memory On a 4 V100 GPU machine, KRISP took $\sim$18.5 hours to train, vs $\sim$13 hours for the MMBERT baseline. For batch size 1, KRISP takes up 3.4 GB of memory vs 3.1GB for MMBERT.

5. Pretraining and State-of-the-Art

In this section we provide the details for our state-of-the-art method, as well as study the benefit of visio-linguistic pretraining which has shown to be beneficial for many vision-and-language tasks (see e.g. [14, 10]) including OK-VQA [3] and compare the results to prior work.

Pretraining	MMBERT	KRISP
BERT only	29.29	32.31
Masked COCO Captions	33.19	35.04
Masked VQA Questions	34.32	35.74
VQAv2	37.10	37.79
VQAv2 (incl. graph)	–	**38.90**

Table 8. Vision/Language Pretraining results on OK-VQA v1.1. We compare MMBERT and KRISP on our three pretraining tasks, Masked COCO, Masked VQA and VQA. For the MMBERT model we only pretrain the transformer except in the last experiment where we pretrain the entire model (incl. graph).

Pretraining First, we look at three kinds of pretraining for our model and how it affects the performance.

The first two are Masked COCO and Masked VQA, introduced in [18]. The objective is that given the image regions as $v = \{v_1, ..., v_N\}$, the input texts as $l = \{l_1, ..., l_M\}$ we train a model to reconstruct either l and/or v from corrupted versions $\hat{v}$ and $\hat{l}$ where some words l_m or image regions v_n are masked. In the Masked COCO task, the captains are used as l and for the Masked VQA task, the questions are used as l. The third task is simply training on the question answering objective of VQAv2 [4].

In Table 8 we show the results of KRISP as well as the baseline MMBERT pretrained on these tasks. Note that the transformers are still pretrained on BERT—we do this pretraining starting from BERT. For all but the last line in the table, we only pretrain the transformer model on these tasks. For the final number, we pretrain our entire KRISP model including the graph network on the VQA task.

As we can see, all forms of pretraining improve our models. The most effective method of pretraining is to train on VQA. This is intuitive since OK-VQA and VQA are quite similar tasks. We also see that our KRISP model consistently outperforms MMBERT, which is our model without symbolic knowledge.

Interestingly, we find that it is not only beneficial to pretrain the transformer but also the symbolic graph network (note that for MMBERT the entire model is pretrained already in the second to last line as it does not have a graph component). Our fully pretrained KRISP achieves 38.90% accuracy, compared to fully pretrained MMBERT of 37.10%.

6. Additional Ablations

We show the results of two final sets of ablations here.

First in Table 9 we ablate which sources knowledge graphs we use. We show at the top our normal result where we have all 4 knowledge graph sources. Below that we have the accuracies for just the DBPedia graph, just the VisualGenome graph, just the hasPart KB graph and just the ConceptNet graph. As you might expect, all of these ablations get lower numbers than the combined graph. The two best graphs from this analysis seem to be DBPedia and ConceptNet.

Method	accuracy	std
KRISP (ours)	**32.31**	0.24
KRISP DBPedia graph	31.69	1.19
KRISP VG graph	30.62	0.20
KRISP hasPart KB graph	30.68	0.59
KRISP ConceptNet graph	31.82	0.37

Table 9. Knowledge Graph Ablation

Method	accuracy	std
KRISP (ours)	**32.31**	0.24
KRISP ImageNet Symbols Only	31.68	0.23
KRISP Places Symbols Only	31.47	0.27
KRISP LVIS Symbols Only	31.48	0.39
KRISP VG Symbols Only	31.95	0.52

Table 10. Image Symbol Ablation

Next in Table 10 we ablate which image classifiers (and thus which symbols) we use as input to our graph network. At the top we show the full results with all 4 sets of symbols. Then we individually show the results if we only use the ImageNet symbols, if we only use the Places symbols, the LVIS symbols and the VisualGenome symbols. Again, we see that using any one of these image classifiers rather than all 4 performs worse than our final method, although the difference between them is not huge small. Based on this experiment, VisualGenome detections were the most significant inputs to the graph network.

7. More Qualitative Examples

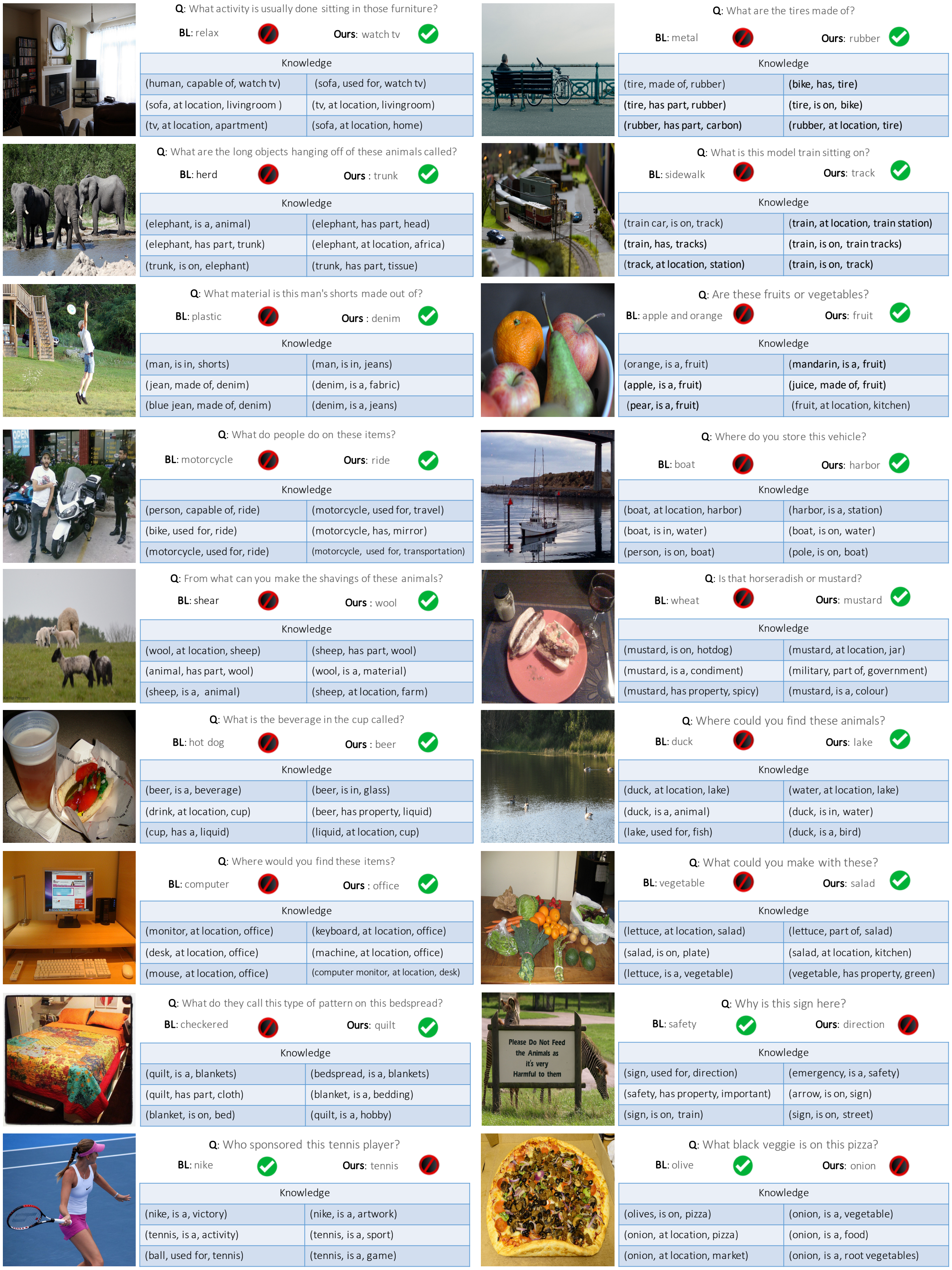
Finally we show additional qualitative examples in Fig. 1.

References

[1] Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. Dbpedia: A nucleus for a web of open data. In *The semantic web*, pages 722–735. Springer, 2007. 1

[2] Sumithra Bhakthavatsalam, Kyle Richardson, Niket Tandon, and Peter Clark. Do dogs have whiskers? a new knowledge base of haspart relations. *arXiv preprint arXiv:2006.07510*, 2020. 1

[3] François Gardères, Maryam Ziaeefard, Baptiste Abeloos, and Freddy Lecue. Conceptbert: Concept-aware representation for visual question answering. In *EMNLP*, pages 489–498, 2020. 3



Q: What activity is usually done sitting in those furniture?

BL: relax ✗ **Ours**: watch tv ✓

Knowledge	
(human, capable of, watch tv)	(sofa, used for, watch tv)
(sofa, at location, livingroom)	(tv, at location, livingroom)
(tv, at location, apartment)	(sofa, at location, home)

Q: What are the tires made of?

BL: metal ✗ **Ours**: rubber ✓

Knowledge	
(tire, made of, rubber)	(bike, has, tire)
(tire, has part, rubber)	(tire, is on, bike)
(rubber, has part, carbon)	(rubber, at location, tire)

Q: What are the long objects hanging off of these animals called?

BL: herd ✗ **Ours** : trunk ✓

Knowledge	
(elephant, is a, animal)	(elephant, has part, head)
(elephant, has part, trunk)	(elephant, at location, africa)
(trunk, is on, elephant)	(trunk, has part, tissue)

Q: What is this model train sitting on?

BL: sidewalk ✗ **Ours**: track ✓

Knowledge	
(train car, is on, track)	(train, at location, train station)
(train, has, tracks)	(train, is on, train tracks)
(track, at location, station)	(train, is on, track)

Q: What material is this man's shorts made out of?

BL: plastic ✗ **Ours** : denim ✓

Knowledge	
(man, is in, shorts)	(man, is in, jeans)
(jean, made of, denim)	(denim, is a, fabric)
(blue jean, made of, denim)	(denim, is a, jeans)

Q: Are these fruits or vegetables?

BL: apple and orange ✗ **Ours**: fruit ✓

Knowledge	
(orange, is a, fruit)	(mandarin, is a, fruit)
(apple, is a, fruit)	(juice, made of, fruit)
(pear, is a, fruit)	(fruit, at location, kitchen)

Q: What do people do on these items?

BL: motorcycle ✗ **Ours**: ride ✓

Knowledge	
(person, capable of, ride)	(motorcycle, used for, travel)
(bike, used for, ride)	(motorcycle, has, mirror)
(motorcycle, used for, ride)	(motorcycle, used for, transportation)

Q: Where do you store this vehicle?

BL: boat ✗ **Ours**: harbor ✓

Knowledge	
(boat, at location, harbor)	(harbor, is a, station)
(boat, is in, water)	(boat, is on, water)
(person, is on, boat)	(pole, is on, boat)

Q: From what can you make the shavings of these animals?

BL: shear ✗ **Ours** : wool ✓

Knowledge	
(wool, at location, sheep)	(sheep, has part, wool)
(animal, has part, wool)	(wool, is a, material)
(sheep, is a, animal)	(sheep, at location, farm)

Q: Is that horseradish or mustard?

BL: wheat ✗ **Ours**: mustard ✓

Knowledge	
(mustard, is on, hotdog)	(mustard, at location, jar)
(mustard, is a, condiment)	(military, part of, government)
(mustard, has property, spicy)	(mustard, is a, colour)

Q: What is the beverage in the cup called?

BL: hot dog ✗ **Ours** : beer ✓

Knowledge	
(beer, is a, beverage)	(beer, is in, glass)
(drink, at location, cup)	(beer, has property, liquid)
(cup, has a, liquid)	(liquid, at location, cup)

Q: Where could you find these animals?

BL: duck ✗ **Ours**: lake ✓

Knowledge	
(duck, at location, lake)	(water, at location, lake)
(duck, is a, animal)	(duck, is in, water)
(lake, used for, fish)	(duck, is a, bird)

Q: Where would you find these items?

BL: computer ✗ **Ours** : office ✓

Knowledge	
(monitor, at location, office)	(keyboard, at location, office)
(desk, at location, office)	(machine, at location, office)
(mouse, at location, office)	(computer monitor, at location, desk)

Q: What could you make with these?

BL: vegetable ✗ **Ours**: salad ✓

Knowledge	
(lettuce, at location, salad)	(lettuce, part of, salad)
(salad, is on, plate)	(salad, at location, kitchen)
(lettuce, is a, vegetable)	(vegetable, has property, green)

Q: What do they call this type of pattern on this bedspread?

BL: checkered ✗ **Ours**: quilt ✓

Knowledge	
(quilt, is a, blankets)	(bedspread, is a, blankets)
(quilt, has part, cloth)	(blanket, is a, bedding)
(blanket, is on, bed)	(quilt, is a, hobby)

Q: Why is this sign here?

BL: safety ✓ **Ours**: direction ✗

Knowledge	
(sign, used for, direction)	(emergency, is a, safety)
(safety, has property, important)	(arrow, is on, sign)
(sign, is on, train)	(sign, is on, street)

Q: Who sponsored this tennis player?

BL: nike ✓ **Ours**: tennis ✗

Knowledge	
(nike, is a, victory)	(nike, is a, artwork)
(tennis, is a, activity)	(tennis, is a, sport)
(ball, used for, tennis)	(tennis, is a, game)

Q: What black veggie is on this pizza?

BL: olive ✓ **Ours**: onion ✗

Knowledge	
(olives, is on, pizza)	(onion, is a, vegetable)
(onion, at location, pizza)	(onion, is a, food)
(onion, at location, market)	(onion, is a, root vegetables)

Figure 1. More qualitative examples from KRISP. Showing predictions by our model and the implicit knowledge baseline Multi-modal BERT. We show the question, image, and answers given by both models. We also show knowledge in the graph related to the question, answers or image that seemed most relevant.

[4] Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *CVPR*, 2017. 4

[5] Agrim Gupta, Piotr Dollar, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *CVPR*, 2019. 1

[6] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 1

[7] Huaizu Jiang, Ishan Misra, Marcus Rohrbach, Erik Learned-Miller, and Xinlei Chen. In defense of grid features for visual question answering. In *CVPR*, 2020. 1

[8] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv*, 2014. 2

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

[10] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019. 3

[11] Xiang Li, Aynaz Taheri, Lifu Tu, and Kevin Gimpel. Commonsense knowledge base completion. In *ACL*, pages 1445–1455, 2016. 1

[12] T. Lin, M. Maire, S. J. Belongie, R. B. Girshick, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: common objects in context. *ECCV*, 2014. 1

[13] Hugo Liu and Push Singh. Conceptnet—a practical commonsense reasoning tool-kit. *BT technology journal*, 22(4):211–226, 2004. 1

[14] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 3

[15] Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *CVPR*, 2019. 1

[16] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, pages 8026–8037, 2019. 1

[17] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *IJCV*, 115(3):211–252, 2015. 1

[18] Amanpreet Singh, Vedanuj Goswami, and Devi Parikh. Are we pretraining it right? digging deeper into visio-linguistic pretraining. *arXiv preprint arXiv:2004.08744*, 2020. 4

[19] Jingru Tan, Changbao Wang, Buyu Li, Quanquan Li, Wanli Ouyang, Changqing Yin, and Junjie Yan. Equalization loss for long-tailed object recognition. In *CVPR*, pages 11662–11671, 2020. 1

[20] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *TPAMI*, 2017. 1