An Explainable Toolbox for Evaluating Pre-trained Vision-Language Models

Tiancheng Zhao[1,2], **Tianqi Zhang**[3], **Mingwei Zhu**[3], **Haozhan Shen**[3], **Kyusong Lee**[1,2], **Xiaopeng Lu**[1,2], **Jianwei Yin**[3]
Om Research Lab, Binjiang Institute of Zhejiang University[1]
Linker Technology Research Co. Ltd[2]
College of Computer Science and Technology, Zhejiang University[3]
{tianchez, kyusongl}@zju-bj.com, lu_xiaopeng@hzlh.com
{zhang_tq, zhumw, cnfighting, zjuyjw}@zju.edu.cn

Abstract

We introduce VL-CheckList, a toolbox for evaluating Vision-Language Pretraining (VLP) models, along with a benchmark dataset for fine-grained VLP model analysis. Most existing VLP models evaluate their performance by comparing the fine-tuned downstream task performance. However, only average downstream task accuracy provides little information about the pros and cons of each VLP method. In this paper, we demonstrate how minor input changes in language and vision will affect the prediction outputs. We also provide a guideline for the research community to utilizes and contributes to this toolbox. Lastly, a case study based on VL-CheckList is conducted to analyze one of the representative VLP models. Data and code are available at https://github.com/om-ai-lab/VL-CheckList

1 Introduction

The ability to quickly iterate various methods and obtain insightful feedback is crucial for successful research. For production machine learning (ML) system, comprehensive testing before deployment is crucial for reliable user experience. Therefore, explainable ML evaluation has emerged to complement benchmark evaluation (Bolya et al., 2020; Ribeiro et al., 2020; Du et al., 2022), which strives to provide an interpretable evaluation of a ML systems and analyze the system from a number of disentangled aspects (Bolya et al., 2020).

The advantages of explainable evaluation vs. typical benchmark evaluation include: (1) downstream task performance only provides a black box score and it is difficult to obtain insights for improving a system. (2) typical dataset is not designed to test models' robustness against extreme corner cases, which are however crucial for many real-world tasks, e.g. autonomous driving.

Given the importance of explainable ML evaluation, this paper concerns about Vision-Language Pretraining (VLP) models. VLP models have rapidly improved (Li et al., 2020; Radford et al., 2021; Li et al., 2021; Zhao et al., 2022), thanks to the emergence of multimodal transformers (Vaswani et al., 2017) and the availability of large paired image-text dataset (Sharma et al., 2018; Changpinyo et al., 2021). Many proposed VLP models have aided in achieving the state-of-the-art performance of a variety of downstream multimodal tasks, ranging from visual QA (Lu et al., 2019), multimodal retrieval (Lu et al., 2021) to visual grounding (Kamath et al., 2021) and many others. On the other hand, the current defacto method to evaluate a VLP model is based on the fine-tuned downstream tasks performance, which is insufficient due to the limitations of benchmark evaluation.

To address this challenge, this paper introduces VL-CheckList, an explainable framework that comprehensively evaluates VLP models, facilitates deeper understanding, and inspires new ideas for improvement. The core principle of VL-CheckList are: (1) evaluate a VLP model's fundamental capabilities instead of its performance on applications (2) disentangle capabilities into relatively independent variables that are easier to analyze. Specifically, we choose Image-Text-Matching (ITM) as the main target of evaluation since it is perhaps the most universal pretraining objective that appear in all VLP methods (Li et al., 2019a, 2020; Radford et al., 2021; Li et al., 2021). We then propose a taxonomy that divides the capabilities of VLP systems into three categories: object, attribute and relation. Each aspect is then further divided into more fine-

Proceedings of the The 2022 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pages 30 - 37

grained variables, e.g. attribute is composed of color, material, and size, etc. Then, a linguistic-aware negative sample sampling strategy is proposed to create "hard negative" that challenges a VLP model's discriminative power against small changes in the input space. Lastly, VL-CheckList is implemented as a toolbox that allows the research community to plug into their evaluation pipeline.

2 Related Work

Benchmark evaluation is a common method to compare different ML models in previous research (Rajpurkar et al., 2016; Bowman et al., 2015; Wang et al., 2018). However, researchers have reported several limitations of the existing VLP benchmark. 1) the objects of interest have a biased distribution of size and location, i.e., tend to be large objects that appeared in the center region. 2) benchmark evaluation returns only a plain score instead of fine-grained details on the taxonomy. Therefore, it is difficult to understand the strengths and weaknesses of a model without a comprehensive analysis. Recent studies show even the state-of-the-art systems that achieved better scores than humans, may still be insufficient in real-world applications (Ribeiro et al., 2020). Thus, researchers have attempted to evaluate ML models with more fine-grained details and avoid bias on the test set.

One of the successful tools for the qualitative analysis of natural language processing (NLP) is CheckList (Ribeiro et al., 2020) which evaluates general linguistic capabilities and revealed weaknesses in several state-of-the-art NLP models and commercial applications. In computer vision, the Vision CheckList was proposed to help system designers to understand model capabilities (Du et al., 2022). They offered a set of transformation operations to generate diverse test samples of different test types, such as rotation, replacing image patches, blur, and shuffle. However, target objects in the transformed images are unchanged, still center and large.

The idea of the CheckList has also been applied other fields, e.g. evaluating Reinforcement Learning (RL) agents (Lam et al., 2022), Dynabench (Kiela et al., 2021) was proposed to generate dynamic benchmark datasets. It overcomes the problem that the existing benchmark fails to cover fundamental linguistic challenges. TIDE (Bolya et al., 2020) is a tool to analyze the errors of object detection. It defines critical error types and shows a comprehensive analysis.

3 VL-CheckList

An intuitive approach to evaluate multi-modal systems is to check if a model correctly predicts alignment between different modalities. We choose image-text matching (ITM) to check the alignment between vision and language for the following reasons. Specifically, ITM is defined as the function that outputs the probability of an image i is matched to a sentence t.

The ITM task is used as an effective and universal pretraining objective in almost all VLP models (Li et al., 2020). The ITM task is also model agnostic and applies to all multi-modal fusion architectures. Thus, we exploit the ITM to fairly compare the VLP models without finetuning them on downstream tasks.

The basic principle of the VL-CheckList is to probe the model's robustness on the negative examples. A robust VLP model should be able to return a higher ITM score for the positive image-text pair than the negative example on the ITM head. We perturb the one-side modality to manipulate them and compare the score with original samples. LV-CheckList offers both language-side and vision-side variations.

3.1 Language Variation

To provide a fine-grained analysis of the robustness of the text-side, we build evaluation taxonomies that are selected based on common mistakes or frequent usage. Based on the common issues in VLP models, the proposed framework places the three input properties (object, attribute, and relation) as the top layer of the evaluation taxonomy.

Object: A strong VLP model is supposed to recognize whether or not the objects mentioned in a text exist in the image. Therefore, if we replace objects in the correct text with some other random noun phrases, a VLP model should give it a lower ITM score than the original sentence. Furthermore, a strong VLP model should be able to recognize the ex-

istence of objects, regardless of its location and sizes. Thus, we further evaluate the robustness Object ITM by testing location variance (e.g., center, middle, and margin) and size variance (e.g., small, large, medium), specifically:

$$\text{loc(x, y)}= \begin{cases} center & \text{if } \frac{y}{x} \leq \frac{1}{3} \\ mid & \text{if } \frac{1}{3} < \frac{y}{x} \leq \frac{2}{3} \\ margin & \text{otherwise} \end{cases}$$

where, x is the half-length of the diagonal of the full image $x = \frac{\sqrt{w^2+h^2}}{2}$. and y is the distance between its central point and the central point of the full image.

To get the size of an object, we use the object area information (i.e., the bounding box of height multiplies the width).

$$\text{size(x)}= \begin{cases} small & \text{if } area \leq S \\ medium & \text{if } S < area \leq M \\ large & \text{otherwise} \end{cases}$$

where, $area = w*h$, S denotes small size and M is the medium size. We set $S = 1024, M = 9216$ in this paper.

Attribute: Determining specific attributes for any object is very challenging. The attribute generally contains color, material, size, state, and action.

- Size: replace the size expression like small, big, and medium with another (e.g., There is a big apple vs. there is a small apple)
- Material: replace a material word in the sentence (e.g., a metal box vs. a wood box)
- State: replace the state expression, such as cleanliness and newness (e.g., a man with dry hair vs. a man with wet hair).
- Action: replace the action-related word in the text (e.g., a standing person vs. a sitting person).
- Color: replace the color word in the text (e.g., A red apple is on the table vs. A green apple is on the table)

Relation: Relation cares about the interaction between two objects. It covers replacing the predicate in a triple (e.g., subject, predicate, object), where the subject and object are both objects in the image. A strong VLP ITM head should assign a higher score to text matching the pair-wise object interaction. Further, we divide prediction into spatial and action. If a predicate is one of the spatial prepositions (e.g., in, on, at, etc), it is sub-categorized as 'spatial'; otherwise, it is labeled 'action.'

- Spatial: If a model can predict spatial relation between two objects (e.g, <cat, on, table> vs. <cat, under, table>).
- Action: If a model can predict other relation than a spatial preposition, usually action verbs like run, jump, kick, eat, break, cry, or smile (e.g., <cat, catch, fish> vs. <cat, eat, fish>)

3.2 Vision Variation

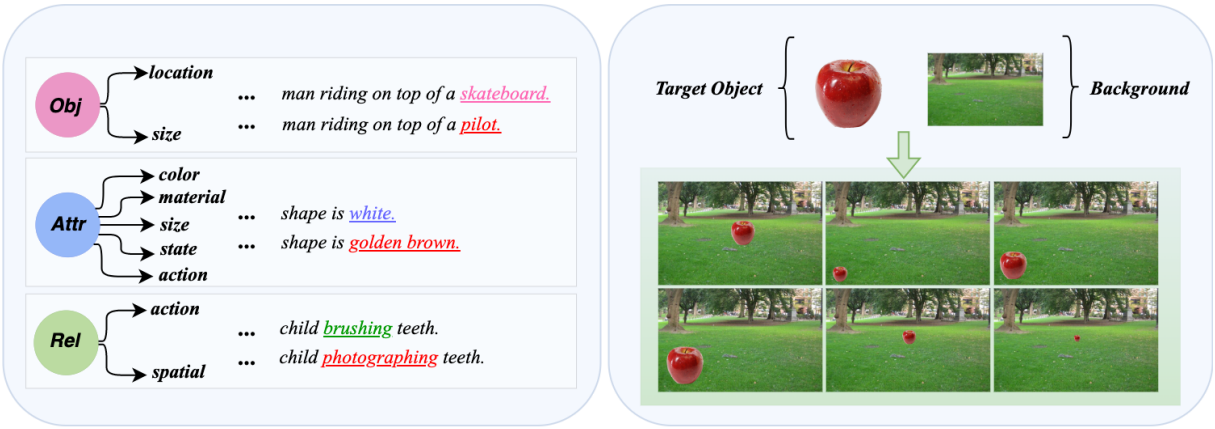
A strong VLP model should be able to return consistent scores when an image is transformed with augmentation techniques such as rotation, shift, flip, random brightness, etc. However, previous augmentations are applied on the entire image-level. We provide the object-level data augmentation by combining cropped objects and image background. The generated images are utilized to investigate the robustness of the model outputs in various locations and sizes of the target object. Strong VLP models should be able to return consistent scores regardless of the location and size of target objects unless the language description is related to location and size (e.g., an apple is the left side of the tree, an apple is small). We allow to input cropped objects and background images and randomly place the target objects from margin to center with various sizes to probe the robustness. The goal of the LV-CheckList on the vision-variations is to show how simple input changes such as object location and size will affect the prediction outputs in the VL-CheckList Demo.

4 Detailed User Guideline

This section describes a guideline for researchers to use and contribute to the VL-CheckList project.

First, users can install from GitHub[1] or from `pip install vl-checklist`. We further provide a HuggingFace demo for people to try out different VLP models[2]. Then the following is a step-by-step guideline to use VL-CheckList.

[1] `github.com/om-ai-lab/VL-CheckList`
[2] `huggingface.co/spaces/omlab/VL_checklist_demo`



(a) Language Variation *(b) Vision Variation*

Figure 1: Language Variation: negative samples are based on object, attribute and relation. Vision Variation: a user inputs target objects and backgrounds and evaluates the various synthesized images

1) Define Corpus: a user defines a corpus in the yaml config file. We provide four initial pre-processed corpora using the semi-structured dataset such as VG (Krishna et al., 2017), SWiG (Pratt et al., 2020), VAW (Pham et al., 2021) and HAKE (Li et al., 2019b). We build a benchmark dataset for each capability test in the proposed framework. We provide the pre-processed datasets in the *corpus* folder of our Github page. An example of the corpus config yaml file is as follows:

```
ANNO_PATH: "Attr/action.json"
IMG_ROOT: "vg/"
TYPE: "TUPLE_JSON"
```

ANNO_PATH is the specific Json file path that includes positive and negative captions and the specific image path.

The data type is TUPLE_JSON. We converted the corpus into list of image path and captions(positive and negative), in the format of a list of `[[{image_path:str, "POS":pos_captions:list, "NEG": neg_captions:list}] . . . ]`

2) Define evaluation configuration: Users can specify the evaluation settings in another yaml to define evaluation in detail as the following example:

```
MAX_NUM: 2000
MODEL_NAME: "CLIP"
BATCH_SIZE: 4
TASK: "itc"
DATA:
  TYPES: ["Object/Location/mid"]
  TEST_DATA: ["vg_obj"]
OUTPUT:
  DIR: "output/clip"
```

The "MAX_NUM" is the maximum number of data points and the "MODEL_NAME" needs to be specified. Appropriate "BATCH_SIZE" should be input based on the GPU resources. The "TASK" can be either "ITC" or "ITM". The "ITC" score compares models' scores on both positive and negative captions. It counts as a true positive when the score on the original is higher than the negatively transformed one. The "ITM" is predicting each image and a caption. It is called the true positive when a softmax score on a positive example on the image is higher than the threshold of 0.5. The Data tag consists of TYPES and TEST_DATA. The TYPES is the storage paths of the "ymal_files". In the top-level directory, we can divide it into three categories: Object, Relation, and Attribute. For Swig, Vg, etc., there are multiple data subsets, so the data subset type should be filled in the TEST_DATA. We can specify the evaluation data, output directory, and format as the example above. After defining a config file, users can simply start the evaluation as follows:.

```
from engine import Model
from vl_checklist import Evaluate
if __name__ == '__main__':
    model = Model('model.ckpt')
    eval = Evaluate("sample.yaml",
                    model=model)
    eval.start()
```

3) Define Model: Users can import VL-CheckList to their projects (e.g., `import vl_checklist`) and need to implement one model class that includes the essential functions, "predict". The predict function should return probabilities on each pair of images and texts. We included several representative models for quick comparisons, such as ViLT (Kim et al., 2021), ALBEF (Li et al., 2021), OSCAR (Li et al., 2020), etc as example projects.

5 Experimental Settings

In this section, we profile one of the most representative VLP models, CLIP (Radford et al., 2021) by testing its ability to understand an object, attribute, and relationship between a text prompt and a given image for language variations.

Metric: We return the model output scores between the text description and the generated negative samples. If the model score on the original text description is higher than the score on the generated negative samples, we regard it as positive output. We obtain the accuracy with the following equation.

$$acc = \frac{\sum_{i=0}^{i<n} f(x_i^p, x_i^n)}{N} \quad (1)$$

where, $f(x_i^p, x_i^n) = 1$ if $p(x_i^p|I_i) > p(x_i^n|I_i)$, otherwise 0. x_i^p denotes a positive sample of i^{th} data. x_i^n means a positive sample of i^{th} data. The N is the total number of pairs of positive and negative samples. I_i is i^{th} image data.

Data: The proposed VL-CheckList focuses on a directional expectation test, in which the label is expected to change in a certain way. For example, when there is a black bear in the photo and the text description is "A black bear is holding a stick". We can transform several variations (e.g., <a black bear → a red bear>, <a stick → an apple>, <holding → throwing>, etc). The negative sampling strategy is the essential step for unbiased evaluations. To generate hard negative examples, we use the structured text description datasets such as Visual Genome (VG) (Krishna et al., 2017), SWiG (Pratt et al., 2020), and Human Activity Knowledge Engine (HAKE) (Li et al., 2019b). The VG provides attributes, relationships, and region graphs which can make a hard negative sample by replacing one attribute in the relation in the image. The SWiG dataset provides structured semantic summaries of images with roles such as agent and tool. We generate hard negative samples by replacing one of the roles in the text description to mismatch with the image. HAKE dataset provides the relationship between instance activity and body part states (e.g., "head" inspect rear view, "right hand" hold wheel, "hip" sit on chair seat).

For the VG dataset, we first assign each attribute, object, and relation to the closet type by cosine similarity from sentence transformers. For objects and relationships, we randomly sample a corresponding instance with a cosine similarity threshold of 0.5. For attribute, we randomly sample a corresponding instance from the same attribute class with a cosine similarity threshold of 0.5. We further conduct manual correction on the generated to data to fix inappropriate data.

For vision variations, we only conduct qualitative analysis by visualizing the output scores via the GUI demo. (Figure 2).

6 Results and Analysis

In general, the ability of CLIP to understand object changes is promising when the object is center and large (see the prefix-O at Figure 3). We hypothesize that the CLIP model pays more attention to the central region and focus on salient objects, similar to the perspective of human observation. On the other hand, CLIP's ability on recognizing attribute and relation-related variants is surprisingly low, especially for Relation-spatial variations (Figure 3).

Then, We investigate whether performance can be improved by cropping the regions of interest (ROI) first and then encoding the cropped ROIs via CLIP. We extract text descriptions on each bounding box on the VG dataset to form a new image-text pair **(Image$_{local}$,text)**, and construct new datasets for VG: Local$_{subj}$, Local$_{obj}$. Results on Local$_{subj}$ and Local$_{obj}$ show that Region CLIP outperforms the original CLIP (whole image encoding) by 3.9% and 5.7% respectively (Table 1). This confirms our hypothesis that the original CLIP was trained to match the entire image to a text description, without capturing the fine-grained alignment between image regions

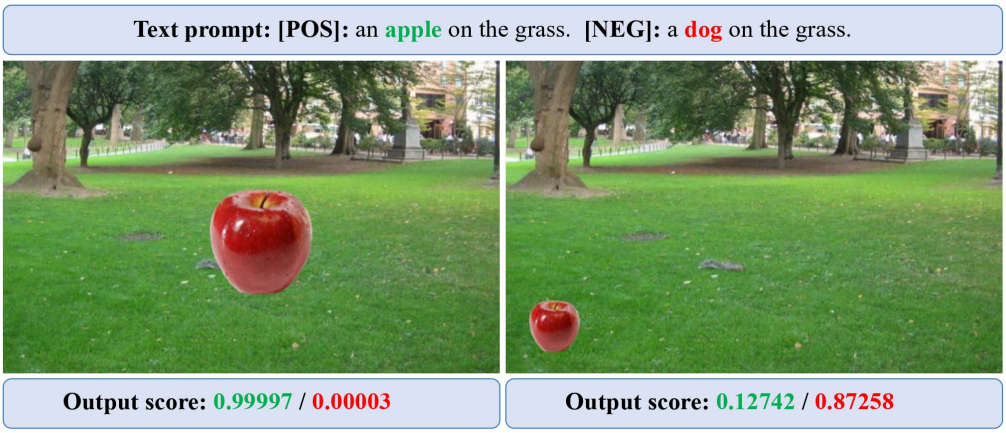


Figure 2: A comparison of CLIP's performances of the image with a big object in the center and image with the same small object in the corner

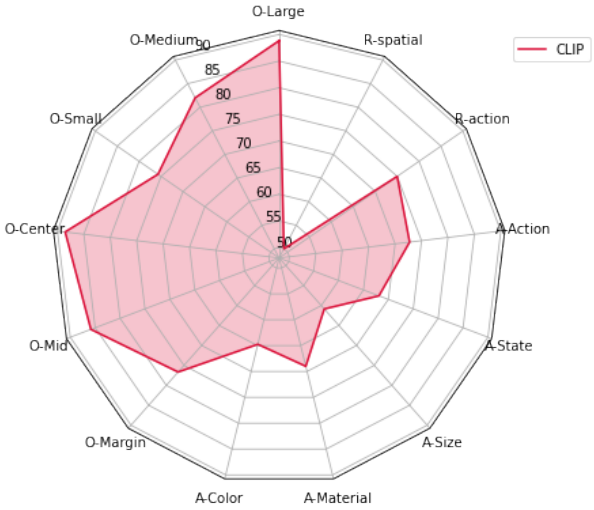


Figure 3: A radar chart for text variance on the CLIP model. (The prefix O, A, and R is Object, Attribute, and Relation respectively)

and text spans. Thus the understanding of minor objects in the image for CLIP is still challenging and explore more fine-grained region-to-text multimodal alignment is a promising direction (Zhong et al., 2022).

For vision variations, we synthesize images by changing an object's size and location. In Figure 2, the image on the left is a big apple in the center, while the image on the right is a small apple in the corner. The text prompt we input is "an apple on the grass" and "a dog on the grass". The accuracy of the left image with a big and center apple is nearly 1.00, while the right image with a small and corner apple only obtains 0.127 of accuracy. The location and size of the object in the image can significantly affect the judgment of the model.

Thus Experimental results indicate that the current benchmark evaluation reveals a gap of performance for real applications. CLIP mostly focus on objects that appeared in the center of the image and the size of the objects should be large. This limits its performance if the target objects are minor in the marginal regions for real-world applications.

Model \ VG_{data_type}	Subj	Obj
CLIP_Global	80.7	86
CLIP_Local	**84.6**	**91.7**

Table 1: Subj and Obj are two attribute subsets extracted from VG dataset. A new dataset is constructed using the bounding box tag of VG to merge and extract the region image pointed by *subj* and *obj* fields. The text remains the same as previous content **(Image$_{local}$,text)**. It only does the expansion experiment for CLIP.

7 Conclusion

This paper introduces VL-CheckList to analyze VLP models from language and vision variations. For language variance, we evaluated from three aspects: object, attribute and relation. For vision variance, we generated synthesized images using cropped target objects and background. We found limitations of the CLIP model: 1) limited understanding for small objects in the corner 2) incompetence for recognizing relations and attributes. In the future, we plan to include more fine-grained taxonomies

and synthesizing strategies into VL-CheckList and also improve existing VLP methods under the guidance of VL-CheckList report.

8 Acknowledgement

This study is supported by National Natural Science Foundation of China under Grant (No. 61825205).

References

Daniel Bolya, Sean Foley, James Hays, and Judy Hoffman. 2020. Tide: A general toolbox for identifying object detection errors. In *European Conference on Computer Vision*, pages 558–573. Springer.

Samuel Bowman, Gabor Angeli, Christopher Potts, and Christopher D Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642.

Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. 2021. Conceptual 12m: Pushing web-scale image-text pre-training to recognize long-tail visual concepts. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3558–3568.

Xin Du, Benedicte Legastelois, Bhargavi Ganesh, Ajitha Rajan, Hana Chockler, Vaishak Belle, Stuart Anderson, and Subramanian Ramamoorthy. 2022. Vision checklist: Towards testable error analysis of image models to help system designers interrogate model capabilities. *arXiv preprint arXiv:2201.11674*.

Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. 2021. Mdetr-modulated detection for end-to-end multi-modal understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1780–1790.

Douwe Kiela, Max Bartolo, Yixin Nie, Divyansh Kaushik, Atticus Geiger, Zhengxuan Wu, Bertie Vidgen, Grusha Prasad, Amanpreet Singh, Pratik Ringshia, et al. 2021. Dynabench: Rethinking benchmarking in nlp. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4110–4124.

Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt: Vision-and-language transformer without convolution or region supervision. In *International Conference on Machine Learning*, pages 5583–5594. PMLR.

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.

Kin-Ho Lam, Delyar Tabatabai, Jed Irvine, Donald Bertucci, Anita Ruangrotsakun, Minsuk Kahng, Alan Fern, Jeongyeon Kim, Yubin Choi, Juho Kim, et al. 2022. Beyond value: Checklist for testing inferences in planning-based rl. *ACM Transactions on Interactive Intelligent Systems*, 12(1).

Junnan Li, Ramprasaath Selvaraju, Akhilesh Gotmare, Shafiq Joty, Caiming Xiong, and Steven Chu Hong Hoi. 2021. Align before fuse: Vision and language representation learning with momentum distillation. *Advances in Neural Information Processing Systems*, 34.

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

Xiujun Li, Xi Yin, Chunyuan Li, Pengchuan Zhang, Xiaowei Hu, Lei Zhang, Lijuan Wang, Houdong Hu, Li Dong, Furu Wei, et al. 2020. Oscar: Object-semantics aligned pre-training for vision-language tasks. In *European Conference on Computer Vision*, pages 121–137. Springer.

Yong-Lu Li, Liang Xu, Xijie Huang, Xinpeng Liu, Ze Ma, Mingyang Chen, Shiyi Wang, Haoshu Fang, and Cewu Lu. 2019b. Hake: Human activity knowledge engine. *ArXiv*, abs/1904.06539.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32.

Xiaopeng Lu, Tiancheng Zhao, and Kyusong Lee. 2021. Visualsparta: An embarrassingly simple approach to large-scale text-to-image search with weighted bag-of-words. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5020–5029.

Khoi Pham, Kushal Kafle, Zhe Lin, Zhi Ding, Scott D. Cohen, Quan Tran, and Abhinav Shrivastava. 2021. Learning to predict visual attributes in the wild. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 13013–13023.

Sarah Pratt, Mark Yatskar, Luca Weihs, Ali Farhadi, and Aniruddha Kembhavi. 2020. Grounded situation recognition. In *European*

Conference on Computer Vision, pages 314–332. Springer.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. 2021. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR.

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.

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin, and Sameer Singh. 2020. Beyond accuracy: Behavioral testing of nlp models with checklist. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4902–4912.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. 2018. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

Tiancheng Zhao, Peng Liu, Xiaopeng Lu, and Kyusong Lee. 2022. Omdet: Language-aware object detection with large-scale vision-language multi-dataset pre-training. *arXiv preprint arXiv:2209.05946*.

Yiwu Zhong, Jianwei Yang, Pengchuan Zhang, Chunyuan Li, Noel Codella, Liunian Harold Li, Luowei Zhou, Xiyang Dai, Lu Yuan, Yin Li, et al. 2022. Regionclip: Region-based language-image pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16793–16803.