A Spatial Understanding Test Suite for VLMs

David Krug
University of Washington
krug@uw.edu

Advisor: Shane Steinert-Threlkeld

1 Introduction

In this paper I introduce a spatial understanding test suite for vision-language models (VLMs) in fulfillment of the project option for the CLMS degree from the University of Washington. I begin by performing a literature review in §2, including recent developments related to transformers in the NLP and computer vision fields, existing VLMs, analysis of the spatial understanding of VLMs, and information about prompt sensitivity and prompt engineering. In §3 I describe my projects in detail. In §4 I draw connections between the skills that I used to develop this test suite and the skills that I learned while completing my CLMS degree.

2 Literature Review

2.1 Transformers

Much of the recent advancements in language modeling owes its success to the Transformers (Vaswani et al., 2017). Models which use a Transformer architecture have been shown to have state-of-the-art performance on a variety of natural language processing (NLP) tasks, such as machine translation (Ott et al., 2018), text classification (Chang et al., 2020), and question answering (Lukovnikov et al., 2019).

Transformers achieve their high performance by relying on a self-attention mechanism in order to learn the relationships between elements in a sequence. This allows Transformers to model long distances dependencies between input elements. This also means Transformers can process a sequence in parallel, unlike recurrent networks which must process elements in a sequence one by one. Additionally, Other benefits of Transformer architecture are that it requires minimal inductive biases, it can be applied to multiple domains (as explored in §2.2) and can be scaled up to massive numbers of parameters, such as 1.6 trillion in the case of the Switch Transformer (Fedus et al., 2022).

The self-attention mechanism and massive scalability means that Transformer models are usually trained in two stages, first with unsupervised or self-supervised pre-training on a large training set, and then task specific fine-tuning on a relatively smaller training set. A typical pretraining task is next word prediction, although models like BERT (Devin et al., 2019) attend to the context of a given word in an a-directional fashion, so masked language modeling and next sentence prediction are used instead.

2.2 Vision Transformers

The obvious successful application of Transformers to the field of NLP has encourage their application to the field of computer vision. Vision models which use Transformer architecture have found success at tasks such as image classification (Dosovitskiy et al., 2020), object detection (Carion et al., 2020), low-level image manipulation like resolution enhancement (Yang et al., 2020) and colorization (Kumar et al., 2021), image generation (Ramesh et al., 2021), point-cloud analysis (Guo et al., 2021), and vision-question answering (Yu et al., 2019).

Earlier work in computer vision used CNNs to produce image encodings (LeCun et al., 1989). CNNs can be enhanced with non-local self-attention (Wang et al., 2017) or criss-cross self-attention (Huang et al,. 2019). Unlike these approaches though, Vision Transformers (ViT) (Dosovitskiy et al., 2020) applies self-attention with a Transformer architecture based on Vaswani et al. (2017) with minimal changes. The input embeddings for ViT are flattened image patches paired with positional embeddings.

Various adaptation over the original ViT architecture have been made, such as introducing a teacher-student strategy and distillation tokens (Touvron et al., 2020), using all image patches to contribute to the loss calculation instead of only the classification token (Jiang et al., 2021), subdividing image patches and computing attention at both levels (Han et al., 2021), and

recursively aggregating neighboring tokens into one token (Yuan et al., 2021).

2.3 Vision-language Models

A vision-language model (VLM) is any model which jointly processes information from the vision modality and the language modality. Although this covers a wide range of models spanning many different architectures, most state-of-the-art VLMs use Transformer architecture. The huge leap forward in performance at many NLP and computer vision tasks that Transformers enabled thanks to their pre-training paradigm has propelled Transformers to be the dominant architecture in the field, and indeed all the spatial understanding analyses in this literature review use Transformer architecture. That being said, some VLMs that do not use Transformers instead use RNNs (Donahue et al., 2016), MLPs (Wang et al. 2019), and bag-of-words text model optimized with canonical relation analysis (Gong et al., 2012).

As for Transformer based VLMs, they can be categorized into two categories: multi-stream and single-stream. Multi-stream VLMs feed each modality into a sperate Transformer and then learn cross-modal representations with a third Transformer, while single-stream VLMs use multi-modal inputs to feed a single Transformer.

The first extension of BERT (Devin et al., 2019) into a multi-modal domain was ViLBERT (Lu et al. 2019), a two-stream architecture in which each stream is dedicated either to language or vision inputs. The two-streams consists of a series of Transformer blocks, and cross-modal representation are learned by passing the keys-value pairs from one modality to the attention head of the other. ViLBERT is pre-trained on tasks like predicting if an image and text are related, and then fine tuned on down-stream tasks like visual question answering and caption-based image retrieval.

Some similar models to ViLBERT are LXMERT (Tan and Bansal, 2019), which uses an object-relation encoder and additional pre-training tasks, and PEMT (Lee et al., 2020) which aims to learn cross-modal information in an end-to-end manner, and which reduces memory requirements by sharing parameters across layers and modalities.

CLIP (Radford et al., 2021) is another two-stream VLM , however it has no third transformer to learn cross-modal representations. Instead, it learns cross-modal connections with a training objective that aims to align the otherwise separate language encodings and vision encodings. During training, CLIP takes as input N image-text pairs with the goal of maximizing the cosine similarity between the N correct image-text pairs while minimizing the cosine similarity of the $N^2 - N$ incorrect pairs. The language encoder used is similar to the original ViT, with modifications described in Radford et al. (2019). Two different vision encoders were tested, the first being a Transformer based on ViT and the second being ResNet (He et al., 2015). It was trained on 400 million text-image pairs from the internet. CLIP was able to achieve very impressive one-shot performance at image classification, although has a somewhat high training computation cost.

In contrast to multi-stream VLMs, single-stream VLMs use a single series of Transformers to process multi-modal inputs. VisualBERT (Li et al. 2019) is one such model, and it takes language and vision features as input into the same Transformer which must learn the relations between the two domains through the pre-training tasks of filling in missing language tokens and differentiating between true and false image captions.

VL-BERT (Su et al., 2019) is another single-stream model, and it takes either word or region-of-interest features as input. Furthermore, VL-BERT is trained both on both vision-language datasets as well as text-only datasets.

Unicoder-VL (Li et al., 2019a) achieves state-of-the-art performance by using three task-agnostic pre-training tasks: masked language modeling, masked object identification, and vision-language matching.

Unified VLP (Zhou et al., 2020) uses a single transformer network for both the encoding and decoding instead of independent encoder and decoder networks. It is able to be fine-tuned on very different tasks, from image captioning to VQA.

UNITER (Chen et al., 2020) is yet another single-stream VLM, and it uses four pre-training tasks: masked language modeling, masked region modeling, image text matching, and word-region aligning. It also uses conditional masking on language and vision pre-training, conditioned on the other domain input.

Oscar (Chen et al., 2020) aims to solve the issue that previous VLMs must learn to align image-text semantics by “brute force”. It does this using detected object tags in the training data, along with

the image and text itself. Oscar performs better at vision question answering, NLU, and image captioning than models which do not use object tags during training.

Finally, although image synthesis models are not strictly VLMs as I have defined them (they do not take visual signals as input), it is worth discussing them here. DALL-E (Ramesh et al., 2021) is perhaps the most well know of these. It is able to generate a realistic looking photo from a text prompt, and it works by training a discrete variational autoencoder to compress each image into a smaller grid of image tokens, concatenating byte-pair-encoded text tokens with those image tokens, and training and autoregressive Transformer to model the joint distribution over the text and image tokens.

2.4 Interpretability and Analysis

The dominance of Transformer architecture in NLP is clear, and with each new variant of a Transformer based VLM comes a slightly higher performance on common VL tasks like ImageNet (Deng et al., 2009). However, performance at real-world tasks is often too complicated to be measured with a single metric (Molnar, 2022). Furthermore, for models which perform task that have dramatic outcomes (eg. diagnosing caner), it is important to understand what cause a certain prediction (eg. a visible cluster of malignant cells). Models are more useful and safer to humans when we can understand why it makes the decision and predictions that it does, what aspects of its input it pays attention to, and how the qualities of its input affect its output.

In an attempt to understand what aspects of an image are attended to by computer vision models, Geirhos et al. (2018) evaluate a CNN's performance at classifying images in ImageNet after various image manipulations, such as removing texture information (silhouette) or applying the texture of one object to the shape of another. They found that CNNs rely on texture much more than shape to identify objects. Furthermore, they found that training on a stylized version of ImageNet results in less bias toward texture, higher accuracy at ImageNet, and higher robustness to image distortions.

Like Geirhos et al., Naseer et al (2021) apply a similar experiment to ViTs. They find that compared to CNNs, ViTs exhibit less reliance on texture, whether trained on the stylized ImageNet or not, and that training ViTs on the stylized ImageNet reduces texture-reliance even more. They also find that by introducing a *space token* and adapting attentive distillation (Touvron et al., 2021), a single ViT can exhibit both texture bias and shape bias with separate tokens. They also examine the effect of disturbing the structure of an image by removing positional encodings from a ViT. They find that positional encodings actually contribute minimally toward giving ViTs structural information about an image, and that ViTs are more robust to having positional encodings removed than CNNs are to having patches of an image shuffled. They conclude that "positional encoding is not absolutely crucial for right classification decisions"

Another method of gaining insight into the inner workings of a Transformer model is through visualization. A visualization of the decision process of a Transformer should explain what regions of an image provide the most information to allow the model to make its prediction. Visualization can help with debugging the models, verifing that they are fair, and enabling downstream tasks. In gradient based visualizations, the gradient of each layer is typically multiplied by the input activations, as in Shrikumar et al. (2017). Attribution propagation-based visualization work by reclusively decomposing a network classification decision into contributions of its input elements, as in Montavon et al. (2017). Chefer et al. (2021) introduces a novel visualization method specifically for Transformers which uses an attribution propagation-based scheme. They compute scores for each attention head in each layer and integrate those scores by incorporating both relevancy and gradient information. Their visualization method is able to handle both positive and negative attributions, and it produces class-specific visualizations.

2.5 Spatial Understanding

Spatial understanding is the ability to understand, reason about, and use visual and spatial relationships between objects. In humans, this skill is innate. Spatial understanding is essential for complex understanding of visual scenes, and so a good bi-modally trained language models must have a solid sense of spatial awareness. Furthermore, spatial awareness is a key task for downstream tasks, such as robotics (Shridhar et al., 2021).

A study into the spatial understand of a visionless model is Ghanimifard and Dobnik (2017). They train an LSTM to predict the two-dimensional probability distribution of various spatial relations (eg. 'left', 'right'), as determined by participant humans. Their model is trained on data that is grounded in spatial relations, which means input word-embeddings are concatenated with encoded location features in the form of 7x7 *spatial templates*. They find that when their model is trained on single-world relations (eg. 'above') and composed relations (eg. 'above or below'), it is able to predict the distribution of unseen single-words that have only appeared in composed relations. Likewise, their model is able to predict unseen composed relations that contain seen single-word relations. Furthermore, they find that introducing distractor word which do not contribute to the spatial semantics of a relations does not significantly distract their model or worsen performance.

Zhang et al. (2020) evaluate large, pre-trained, Transformer-based language models like BERT (Devin et al., 2019) and eLMO (Peters et al., 2018) on their ability to represent scalable attributes like length, mass, and price. They trained linear probes on-top of the pre-trained models to recover a distribution over one of those three attributes. Their conclusion was that pre-trained models are bad at encoding length, mass, and price information. Bhagavatula et al. (2020) also evaluates large pre-trained models for their spatial understanding. They introduce a new dataset, ART, which evaluates the abductive commonsense reasoning of language models. They find a significant gap between BERT performance and human performance, and that BERT especially underperforms on test instances involving spatial or numerical descriptions.

The studies above describe models which are trained only on text information (or text information and spatial templates). In contrast, Collell et al. (2018) apply spatial understanding to a model which also gets input signals from the vision domain, but only indirectly. For their task, a model must predict the coordinates and size of the bounding box of an object relative to another object, given the names of the two objects, a word denoting their spatial relation, and the position of the other bounding box. They find that their models generalize well and can predict the position of unseen objects. This indicates that the word embeddings provided as input have information about spatial properties embedded in them, or that objects with similar word embeddings have similar spatial distributions. They conclude that in general, "models acquire solid common sense spatial knowledge".

As for analyzing studies about models which take visual signals as input directly, Liu et al. (2019) introduces CLEVR-Ref, itself a modified version of CLEVR (Johnson et al., 2017). CLEVR-Ref is a diagnostic dataset aimed at evaluating a VLM's ability to identify (by bounding box or segmentation) the objects in an image which are uniquely referred to by some referring expression. Referring expressions may pick out objects by referring to their size, shape, color, etc. Referring expressions may also pick out objects by referring to their relation to other objects, such as "second sphere from the left" or "sphere to the left of the cube" They found that models they tested had higher performance when tested on referring expression that mentions the color, shape, or visibility. This suggests that these features are more salient or easier to learn for VLMs than the size, ordinality, or material of an object.

Cirik et al. (2018) point out that state-of-the-arts VLMs involve complex neural parameterizations which make it difficult to interpret what is actually being learned, and they criticize existing referring expression datasets on the basis that they can be solved by exploiting biases in the datasets. They show that shuffling the order or words in a referring expression, discarding preposition, or even discarding the entire referring expression (using only the image is input) only results in a minor drop in VLM performance at locating the referred object. This suggests that existing referring expression dataset can be solved by exploiting statistical biases.

VSR (Liu et al., 2022a) is a dataset specifically aimed at evaluating the spatial skills of VLMs. Each instance in their dataset consists of one image, a caption, and a binary label. The binary label reflects if the caption truly or falsely describes the image, and the captions describe the spatial relation between two objects. They find a significant gap between human and VLM performance, suggesting that VLMs have a hard time at learning spatial relations. Furthermore, they find the number of training instance for a given relations does not correspond strongly with performance at that relation. The also find that

VLMs have a hard time generalizing spatial relations to unseen concepts, and that spatial relations involving orientation are particularly challenging.

Liu et al. (2022b) introduce another dataset for evaluating the spatial skills of VLMs. Other such datasets, like VSR, focus mostly on explicit relations, like 'above' and 'to the left of'. Liu et al.'s dataset additionally tests implicit relations, like the relation between a person and a bike while the person is riding the bike (ie. 'on'). Their dataset consists of images paired with both explicit captions (eg. 'person on a bike') and implicit captions (eg. 'person riding a bike'). They also evaluate image synthesis models by providing an explicit or implicit caption and evaluating the resulting image. They find that models which take visual signals as input have more accurate and consistent spatial commonsense than text only models. They also find that image synthesis models have even more accurate and consistent spatial commonsense than non-synthesis VLMs.

PaintSkills (Cho et al., 2022) is a diagnostic data set for evaluating image synthesis models. They test four visual reasoning skills. They find that Transformer image synthesis models are better at recognizing and counting objects than they are at recognizing colors and understanding spatial relations. Conwell and Ullman (2022) also evaluate image synthesis models for their visual reasoning skills, specifically focused on spatial understanding. They give DALL-E 2 (Ramesh et al., 2022) various text prompts involving spatial relations (eg. 'teacup in a box'), and had human participants decide if the generated images matched the text prompts. Their conclusion was that DALL-E 2 lacks commonsense relational understanding. Part of DALL-E 2's challenge in generating accurate images was that it was much more biased towards producing frequently observed relations (eg. 'spoon in a teacup' is easier to generate than 'spoon on a teacup').

Briand et al. (2022) uses an adapted version of the SpatialVOC2K dataset (Belz et al., 2018) to evaluate the spatial understanding of VLMs. The dataset in their experiment consists of minimal semantic pairs of captions, one of which correctly describes an image and one of which incorrectly describes an images (eg. "a cat in a box" versus "a cat to the left of a box"). The goal of the VLM is to identify the correct caption. Additionally, they put each caption through 18 different sentence patterns to generate re-wordings of the original captions (eg. "a cat in a box" and "a picture of a cat in a box"). The goal of this was to identify which linguistic features most affect the model's performance at encoding spatial relations. They evaluated CLIP's performance and this task and found that CLIP does not produce robust encodings of the spatial relations between objects in a scene. Echoing results about DALL-E 2 found by Cho et al. (2022), Briand et al. found that CLIP has a bias to select certain relations over others (eg. preferring 'on' rather than 'under', regardless of input image). This suggests that CLIP may be selecting the more frequently observed relation than attending to the facts of the image. Finally, their results showed that CLIP mostly ignored negation, treating a sentence with a negative polarity item the same as the positive version.

2.6 Prompt Sensitivity

As language model grow larger and larger, training a new model for each new task is not practical. Furthermore, a model which is good at many tasks is desirable, since general knowledge from one task may be applicable to other tasks. Prompt engineering is effectively the process of designing inputs which elicit good performance from a pre-trained language model which has not been specifically trained at the given task. Good prompt engineering allows for good zero-shot performance at the given task.

Petroni et al. (2019) was an early work in this area that showed that large pre-trained language models can act as a relational knowledge base when given correctly worded text inputs. For example, rather than using complex NLP pipelines to extract information and store it in a knowledge base, such as the capitals of countries, one could simply ask a pre-trained LM to fill in the masked token of a text prompt like "[MASK] is the capital of France".

Jiang et al. (2020) showed that large pre-trained language models are highly sensitive to their input text, and that simply re-wording a prompt could cause a model to succeed or fail at retrieving a particular piece of information. They also propose two methods of systematically generating diverse prompts which improve knowledge retrieval by 8%. This indicates that pre-trained language models are somewhat knowledgeable, but they are sensitive to how that knowledge is queried.

Liu and Chilton (2022) apply prompt engineering to vision models. They perform a series of experiments in which the develop a set of guidelines for good prompt design specifically for image synthesis models. They found that changing the subject and style keywords had a greater impact of generated images than function words and connecting words. They also found that using style cues had a large impact on generate images, but that it was best if the subject matter matched the given style cues in abstraction.

Zhou et al. (2021) note that simply adding the article 'a' in front of the class label of a prompt improves VLM performances at object identification by about 5%. They write that "a slight change in wording could have a huge impact on performance [of VLMs]".

Oppenlaender (2022) established a taxonomy of types of prompt modifiers used by the text-based generative art community to encourage image-generating models to create a desired image. They found that participants in the generative art community change the characteristics of generated images by using subject terms (eg. "an old car in a meadow"), style modifiers (eg. "surrealist painting"), quality boosters (eg. "highly detailed"), repetition (eg. "space whale. a whale in space"), and magic terms (eg. "control the soul", which produces unpredictable but visually interesting results).

3 SUTS for CVLMs

3.1 Overview

Inspired by the above literature review, and especially by Briand et al. (2022), I introduce a new spatial understanding test suite for VLMs[1]. The goal of the test suite is to target specific linguistic and visual features and analyze what impact they have on the ability of VLMs to identify spatial relations accurately and robustly in synthetically generated images and captions. The initial findings of Briand et al. (2022) have helped guide and refine the captions used in this test suite. This test suite also includes three new test types not considered in Briand et al. (2022).

The test suite is broken into two categories, with each category consisting of two sub-categories. The first category evaluates a VLMs ability to identify the position of an object relative to the boarders of the 2D image itself. The second category evaluates a VLMs ability to identify the position of an object in 3D space. Each test instance consists of a single synthetically generated image and two captions which vary only in their spatial relations, one which accurately describes the image and one which falsely describes it.

A concept that is important in these tests is the taxonomy of orientability. For the purpose of defining which spatial relations are possible, I classify all objects into one of four categories: fully orientable, vertically orientable, horizontally orientable, and non-orientable. A fully orientable object, such as a cat, has a left side, right side, top, bottom, front and back. In other words, for a fully orientable object like a cat, one could make statements about "the cat's left" or "the direction that the cat is facing". In contrast, vertically orientable objects, such as a water bottle, only have a top and a bottom. It would be illogical to make statements about "the water bottle's left". Likewise, horizontally orientable objects, such as a torpedo, only have a front and back. Finally, non-orientable objects, such as a baseball, can rotate freely in space and no one would make mention of "the front of the baseball" unless it had some orientable mark like a signature. This taxonomy of objects addresses Liu et al.'s (2022) criticism that existing vision-language datasets fail to account for things like "whether the object has a *front*".

In the following sections I will describe my process of generating the test suite. §3.2 covers image generation. §3.3 covers the definitions of the spatial relations used. §3.4 covers caption generation. §3.5 demonstrates one VLMs performance at this test suite.

3.2 Image Generation

To synthetically generate images, I used the game engine software Unity (Unity Technologies, 2005). Although Unity is usually used for developing video games, it has also been used for developing synthetic training sets, such as training a computer vision system to recognize dust levels in work environments (Xiong and Tang, 2021). Unity is a useful tool for synthetic data generation because it is easy to use and allows for fine control over the visual qualities of the generated images. The overall process for generating images was to manually place objects in the foreground and

[1] Found at: github.com/DavidK0/SUTS-for-VLMs/

background to create different scenes (a cityscape, a countryside, indoors, etc.), randomly scatter other objects in that scene, detect the spatial relations of those objects, and then save a picture of that scene along with the relational information.

The first step in generating images was to select the 3D models to be used. For this I used Free3D (Turbosquid, Inc., 2013), a website which offers 3D models of many objects at various price points and under various licenses. For this dataset I selected 21 free models available with a personal use license. All the objects and their relative frequencies in the dataset are listed in appendix A.

To place the objects in a scene, I defined spawning areas and pre-place those in a scene. A spawning area is a rectangular prism which defines the valid locations that an object can appear. Spawning areas are manually placed within a scene and remain stationary for every generated image. A spawning area can be limited to be two-dimensional so that objects will only appear on the bottom of the spawn area. This is to allow objects to be spawned 'on' a surface and prevent hovering over the surface. Examples of spawning areas can be seen in appendix B.

For every generated image, between 3 and 9 objects are placed in a random spawning area with random coordinates. Each object is given a random rotation about the vertical axis, which turns the object to the left or right. Additionally, each fully orientable or vertically orientable object has a 10% chance to be rotated 90 degrees about the horizontal axis and a 10% to be rotated 180 degrees about the horizontal axis. This either flips the object sideway or upside down.

The initial placement of objects within a spawning area does not guarantee that objects will not overlap. To prevent objects from overlapping, I used Unity's built-in physics engine to detect collision. To simplify collision calculations, the shape of each of the 3D models was approximated with rectangular prisms and spheres. See appendix C for an example of this shape approximation. If a collision is detected while and object is being placed, that object is given a new random position until a valid spawning location is found.

The position and angle of the virtual camera is also randomized within a scene. Like the spawning areas for objects, I define 'camera spots' which are placed in a scene to restrict where the camera can be placed. A camera spot consists of a rectangular prism which defines the location that the camera can be place, as well as a cone that defines the range of angles the camera can have. The camera's position and angle are randomized for every generated image. When placing objects in spawning areas, objects are restricted to only appear within the camera's field of view. Objects also will not be placed if it would be occluded by another object. See appendix D for an example of a camera spot.

To add variety to the generated images, five different scenes are used. A scene consists of a background and optionally a few pre-placed objects in the foreground. The scenes in this test suite are: a cityscape, a bridge, a hilly countryside, indoors, and floating (in outer space or in the clouds). The cityscape and hilly countryside have a picnic table pre-placed. Very large models (the car, the bus, the bench, and the picnic table) are limited to only spawning on the bridge or in outer space. Floating is the only scene which includes a three-dimensional spawning area. For an example image of each of the five scenes, see appendix G.

For added variety, the skybox is randomly chosen for each image. A skybox is Unity's way of controlling what the sky looks like (sunny, cloudy, etc.). For each image, one of 14 skyboxes was randomly chosen. For a list of all the skyboxes used, see appendix E.

After the objects and camera have been placed in a scene and the sky has been randomized, a picture from the perspective of the virtual camera is saved. The image resolution is fixed at 720 pixels by 480 pixels. After the image is saved, the next step is to detect spatial relations between the objects, save those, and repeat the process. Detecting spatial relations is described in the next section. 500 images were generated in this way for each of the 5 scenes, for a total of 2,500 images.

3.3 Relation Detection

When one looks at an image and states something like "There is a toaster to the left of the cat", there are two possible interpretations of that statement. Either, the toaster is further left *on the image* than the cat, or the toaster is to *the cat's* left. In the case that the cat is facing towards the camera, the two interpretations point in different directions. To understand how VLMs handle these two interpretations, my test suite has two categories of tests: frame position test and word position tests. Definitions of these spatial relations and the

manner of detecting them are outlined in the following sections.

3.3.1 Frame Position

The two test types that fall under the frame position category are absolute frame position and relative frame position. Absolute frame position involves only one argument, for example the sentence "The cat is on the right". Relative frame position involves two arguments, for example "The toaster is to the left of the cat".

Absolute frame position is detected by dividing an image into several separate regions, as seen in Figure 1. There are 9 continuous regions (colored red, blue, green, and cyan in Figure 1) which correspond to English phrases like "left", "top", or "bottom right", plus one discontinuous region (colored white in Figure 1) which is not associated with any English phrase. The determine the absolute frame position of an object, I check if a majority of the eight corners of the three-dimensional bounding box of that object lie within the same region. If that is the case, then that object is also said to lie in that region. If that is not the case, then the object is not associated with any region.

For each object which is associated with some region, I generate a set of pairs of captions where one caption in the pairs truly describes the absolute frame position of the object and one caption falsely describes the position. Each of the pairs communicates the same true relationships but may have different false relationships. All the pairs of sentences have varying linguistic features, as described in §3.4.

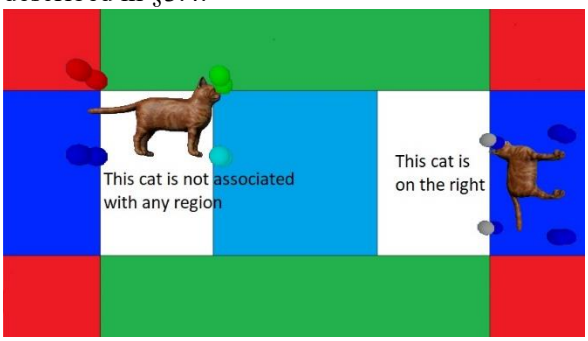

Figure 1: An example of determining the absolute frame position of two cats. The right-most cat has a majority of its corners located in the "right" box. The corners of the left-most cat do not form a simple majority, so it lies in no particular region.

The second type of frame position test is relative frame position. To detect the relative frame position of an object relative to some second object, I divide an image into 9 continuous regions based on the two-dimensional bounding box of the second object, as seen in figure 2. As with absolute frame position, relative frame position is determined by checking if a majority of the eight corners of the three-dimensional bounding box all lie within a region. For each object for which this is true, I generate a set of pairs of true and false sentences which vary by their linguistic properties.

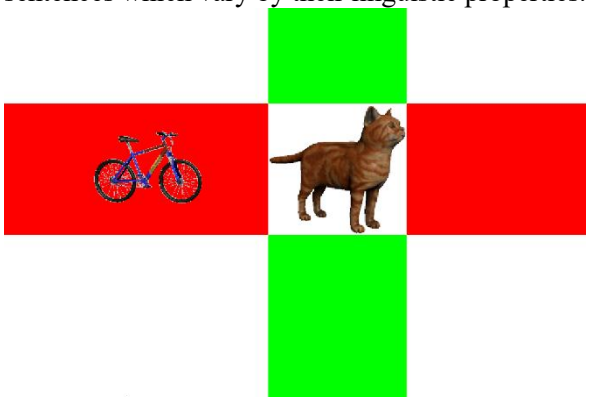

Figure 2: An example of determining the relative frame position of an object. The bike is to the left of the cat.

3.3.2 World Position

The two test types that fall under the world position category are relative world position and orientation. Like with relative frame position, relative world position involves two arguments, for example "The cat is facing the toaster". Orientation involves only one argument, such as "The cat is upside down". Including orientation in this test suite is partially inspired by Liu et al.'s (2022) observations that previous vision-language datasets lack tests related to orientation.

To detect the relative world position of an object relative to a second object, I first consider the taxonomy of orientability with respect to that second object. For example, the sentences "The cat is to the water bottle's left" or "The baseball is facing the cat" are not sensible. Therefore, I only check to the left and right of fully orientable objects, and I do not check the front and back for vertically orientable objects. The check a particular direction, compare an objects bounding box corners again 6 cone-shaped areas, as seen in figure 3.

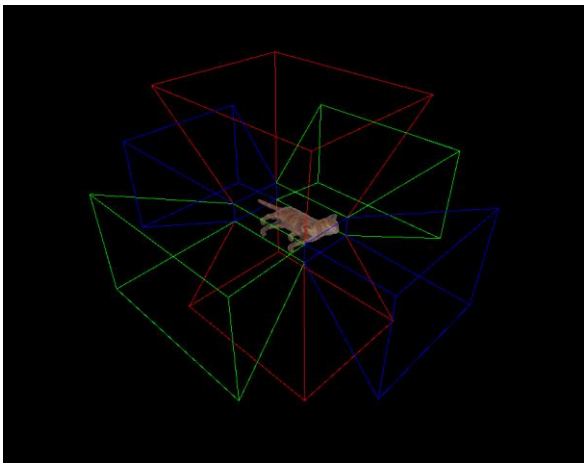

Figure 3: An example of a sideways fully orientable object (a cat) and its corresponding front/back (blue), left/right (red), and top/bottom (green) spatial cones. The cones actually extend forever in their respective directions.

Each image on average has 4.5 objects represented by 15.5 spatial relations. All of the raw spatial relation information for an image is saved in the .json file format along with that image. The next step is to generate various sentences from that raw spatial relation data.

3.4 Caption Generation

After all the images have been taken and all the corresponding spatial relations saved in a .json file by Unity, the next step was to process the raw spatial relations into natural language. For this I used a template based sentence patterns, along the lines of "{noun1} is {relation} {noun2}" or "{adj} {noun1} is to the {relation} of {noun2}". In total, nine different types of sentence patterns were used, though not all nine sentence patterns apply to all objects and all relations. The goal of having a variety of sentence for each relation is to better understand which linguistic features of a sentence cause changes in a VLM's performance. An example of each of the sentence patterns for each of the nine test types can be found in Appendix F.

In order to generate sentences that include adjectives, verbs, and adverbs, I manually assigned accurate adjectives, verbs, and adverbs for most of the 3D models. I did not include adverbs for any inanimate object. Animate objects were give verbs such as "walk", "sit", or "stand", while inanimate objects were only give "sit" or "stand". See Appendix A to see each of the adjectives, verbs, and adverbs for each of the models.

The first type of sentence pattern is basic sentences. In these, the noun phrases are restricted to a single article followed by a noun, and the verb is only the copula 'is'. Additionally, the relative frame position and relative world position test types have two alternate sentence patterns, both of which are structured as "The toaster is to the left of the cat". Because this sentence could be interpreted in two ways, it will be interesting to see which one VLMs prefer.

The second pattern type is like the previous sentence pattern type, except that additional non-relevant linguistic information is added to make the sentences slightly longer. The goal of including this pattern is to evaluate if VLMs get distracted by non-relevant information.

The third type includes adjectives. In these, the noun phrases are an article plus an adjective plus a noun. In sentences with two arguments, adjectives are added to both arguments.

The fourth type includes verbs. These sentences are similar to the basic sentences except the copula is replaced with an active verb. The verb associated with each 3D model was chosen based on that model's pose (eg. a walking cat).

The fifth type includes verbs as well as adverbs. These sentences are not applied to every relation since inanimate objects are not given adverbs.

The sixth and seventh type restructure the sentences into questions and commands. Although these can not truly be called captions since they do not describe the image, it would still be interesting to see if VLMs are more likely to select the text input for which the correct answer to its question is "yes", or commands which have been 'completed', in a sense. It is worth noting that these sentence types slightly deviate from the format of this test suite, since it can not really be said that one sentence is "correct" while the other is "incorrect". Even for a human to complete these sentence patterns, it is not clear how one could say that one command or question is more accurate than another.

The eighth sentence pattern type includes prompt engineering. Inspired by the prompt modifiers used by the text-based generative art community that Oppenlaender (2022) identified, I include style modifiers and repeated prompts as sentence patterns. Style prompts are two styles, "A 3D render of…" and "A picture of…". Repeated prompts repeat a sentence with slightly different wordings.

The ninth and final type are ungrammatical sentences. Like the question and command

sentence patterns, these deviate from the stand format of this test suite because an ungrammatical sentence can not really be said to correctly describe an image. Nevertheless, it makes for an interesting evaluation case. Each ungrammatical sentence still contains within it the names of the objects being described and the core relation (ie. preposition) between them, although the structure of the sentence is lost. This means that if a VLM is able to identify the "correct" caption despite the caption being ungrammatical, that VLM is not actually attending to the linguistic structure of a sentence and is merely picking out individual words to make its judgements.

When generating a false caption for an image, the same sentence pattern as the true caption is used and only the spatial relation changes. The false spatial relation is randomly chosen from the set of relations which could not be used to describe any object in the image. Across the 2,500 images, there are 38291 spatial relations and 499982 true/false sentences pairs.

3.5 CLIP Evaluation

To demonstrate how this test suite can be used, and to set a baseline of performance for it, I evaluate CLIP's (Devin et al., 2019) performance. I used the ViT-B/32 version of CLIP. Due to hardware limitations, I only evaluated CLIP on a random subset of 500 image.

Overall, CLIP's performance is very bad at this test suite, achieving and accuracy of 43.0%. The results for each of the four test types can be seen in figure 4. If CLIP were guessing randomly, it would achieve chance performance, which is 50%. For three of the test categories however, CLIP has worse than random performance. The absolute frame position results are the most surprising. CLIP predicts the wrong frame position about 70% of the time, which suggest that is somewhat attending to the spatial relations in a sentence despite not correctly aligning that with the ground truth of the image. Qualitative analysis reveals that CLIP is unable to identify the correct frame position even in cases where it would likely be trivially easy for a human, such as figure 5.

Test Type	Amount	Accuracy
Absolute Frame	15,399	29.2%
Relative Frame	47,200	44.5%
World Position	24,770	42.8%
Orientation	5,241	51.3%

Figure 4: CLIP's accuracy.at each of the four test types, along with the number of sentences evaluated.

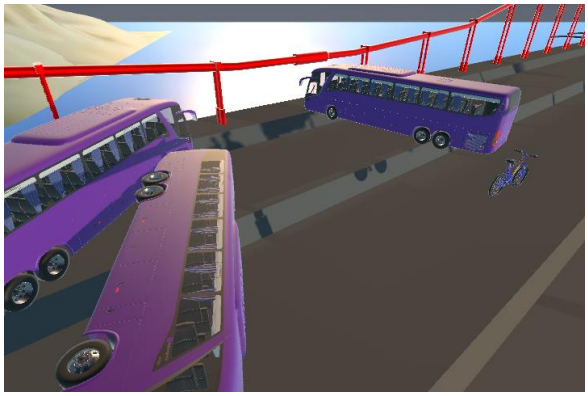

Figure 5: For this image, CLIP prefers the caption "The bike is on the left of this picture" over "The bicycle is on the right of this picture".

Test Type		**Amount**	**Accuracy**
Basic		16,725	45.2%
Filler		7,826	40.0%
Adjective		20,832	42.7%
Verb		12,771	39.8%
Adverb		4,017	37.4%
Question		7,807	44.8%
Command		7,887	43.6%
Prompt Engineering	Picture	10,344	46.2%
	Render	10,295	48.9%
	Repeat	7,546	38.2%
Ungrammatical		7,851	40.2%

Figure 6: CLIP's accuracy at each of the sentence patterns and the number of sentences evaluated.

To understand which linguistic features affect CLIP's performance, we can check its accuracy for each of the nine sentence patterns across all test types. CLIP's accuracy at each sentence pattern is given in figure 6. CLIP has worse than random chance on all categories, again showing that it is not guessing entirely randomly, and yet it also not predicting correctly. CLIP's performance is slightly improved over average with the captions that use the basic sentence pattern, and when the style of the image is specified. Adding adjectives, verbs, or adverbs to a caption makes CLIP guess less randomly, but also less accurately.

Briand et al. (2022) found that despite CLIP's poor spatial understanding, it had relatively good object recognition skills. They originally sourced the images in their study from Belz et al. (2018). For a comparison between datasets, and to confirm that synthetic images are not too out of domain for CLIP, I checked CLIP's ability to simply identify the objects in the generated images. To do this, I used another sentence template that generated sentence like "A cat" and "This picture has a cat in it". CLIP was better at this task, achieving an accuracy of 60.8%[2]. CLIP was able to correctly identify certain objects, like the bus, bike, and car, with 90% accuracy, while the jar was misidentified 75% of the time.

4 Connection to CLMS

In this section I will discuss how the CLMS coursework related to and prepared me for this project. Prior to enrolling in the CLMS program at the UW, I had minimal experience with computational linguistics or natural language processing. I had a moderate amount of linguistics skills thanks to my bachelor's in linguistics from the UW, and some programming skills as well. During my time completing the CLMS degree, I learned a plethora of skills that are essential for succeeding in the field of computational linguistics and for succeeding at this project. I have also refined the skills that I already had.

The skill that was applied most broadly across my course work, and which I used extensively in this project, was programming. I improved at this skill both actively and passively. I studied coding actively by taking CSE 373 Data Structures and Algorithms. In that class I learned techniques for storing and manipulating data. I applied this knowledge by deciding to use hash maps to store the thousands of images and hundreds of thousands of sentences in this data set.

I practiced my general Python skills during the course work by writing homework scripts using python in LING 570, 571, 572, 573, and 574, which I applied to this project by efficiently writing the sentence generation code in Python. I also specifically practiced my PyTourch skills in LING 574 by writing homework scripts that use Prof. Steinert-Threlkeld's edugrad. This made me comfortable enough at working with tensors that I was able to figure out how to batch the inputs to CLIP, which sped up processing a lot.

In LING 570 I learned about shallow processing techniques for natural language processing. A valuable technique from this class that I used in this project is the regular expression. Combine with my improving Python skills, I used Python module `re` to allow me to more easily create the sentence patterns. Creating the sentence patterns also required a basic understanding of English morphology (eg. 'a' versus 'an', various conjugations), which is a skill I brushed up on while taking LING 581 Morphology.

The class that most directly prepared me for this project is LING 575 Analyzing Language Neural Models with Prof. Steinert-Threlkeld. In that class, I learned about various analysis methods to attempt to understand what large language models actually learn while training. We covered topics like visualizations, linear probing, and adversarial data. In that class, I worked with two other classmates to perform an analysis of the spatial understanding of CLIP by using adversarial data constructed specifically to examine what features trip up the model. In this project I extended that work by creating a test suite to help analyze the spatial understanding of vision-language models with more test types and refined linguistics tests.

In that class as well as in LING 574 Deep Learning for NLP, I learned about Transformer architecture. Although my test suite is not limited to only analyzing Transformer based VLMs, it was originally inspired by my earlier attempt at specifically evaluating CLIP, a Transformer based VLM. Transformers dominate in the world of NLP and VLMs, but their complexity makes it difficult to see what is happening under the hood. In LING 574 I learned the history leading up to Transformers, what separates them from other architecture, and in LING 575 I learned how to pick them apart and analyze them.

In LING 575 NLP in the Humanitarian Sector, I studied how NLP connects with society and large, and the social impacts that technology can have when released to the public. We learned about factors like handling people's private data and equal access to opportunities. This also connects with CSE 373, which I took with Kevin Lin who stresses understanding the possible positive ang

[2] An earlier version of my synthetic image generator had less camera placement variation, so objects were more consistently in the center of the frame. This gave CLIP an object detection performance of 80%.

negative outcomes that new technology can have on society. There, we learned to pay attention to what actions a technology will make harder or more easy, as well has what biases a technology may have. During my project I considered these factors and decided that creating this type of test suite does not present a large threat to society due to the fact that it does not handle personal information, it is not biased toward or against any groups, and it can not be used for abuse.

In 573 NLP Systems and Applications I gained skill at completing a project end to end. Going from nothing to a finished product in a roughly ten week span could be daunting, but in thar class I practiced starting from an initial challenge, gathering information and doing research, creating a plan, and then executing that plan to deliver the project. A key idea from that class was starting a working foundation and cyclically revising to get to the ideal point. We also learned about tools to help us work on such projects, like Hugging Face and version control. All of these are skills that I directly applied while completing this project.

Finally, an important skill I learned during my CLMS course work was how to interact with academia in general, and the field of NLP specifically. Part of this includes doing research and quickly identify what information might be valuable by reading title and abstracts as well as reading papers efficiently (I prefer to read the intro first, then conclusion, then body). This skill also includes writing papers. No doubt, my writing skills were at least decent prior to CLMS, but one can always improve. Furthermore, I specifically learned how to write for an NLP-inclined audience, and about ACL formatting.

References

Anja Belz, Adrian Muscat, Pierre Anguill, Mouhamadou Sow, Gaétan Vincent, and Yassine Zinessabah. 2018. SpatialVOC2K: A multilingual dataset of images with annotations and features for spatial relations between objects. In *Proceedings of the 11th International Conference on Natural Language Generation*, pages 140–145, Tilburg University, The Netherlands. Association for Computational Linguistics.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Wen tau Yih, and Yejin Choi. 2020. Abductive commonsense reasoning. In *International Conference on Learning Representations*.

Andrew Briand, Chris Haberland, and David Krug. 2022. Bi-modally trained language models have mediocre spatial awareness. Unpublished manuscript, University of Washington.

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. 2020. End-to-end object detection with transformers.

Wei-Cheng Chang, Hsiang-Fu Yu, Kai Zhong, Yiming Yang, and Inderjit S. Dhillon. 2020. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery Data Mining*, KDD '20, page 3163–3171, New York, NY, USA. Association for Computing Machinery.

Hila Chefer, Shir Gur, and Lior Wolf. 2021. Transformer interpretability beyond attention visualization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 782–791.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. 2020. Uniter: Universal image-text representation learning. In *Computer Vision – ECCV 2020*, pages 104–120, Cham. Springer International Publishing.

Jaemin Cho, Abhay Zala, and Mohit Bansal. 2022. Dalleval: Probing the reasoning skills and social biases of text-to-image generative transformers.

Volkan Cirik, Louis-Philippe Morency, and Taylor BergKirkpatrick. 2018. Visual referring expression recognition: What do systems actually learn? In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 781–787, New Orleans, Louisiana. Association for Computational Linguistics.

Guillem Collell, Luc Van Gool, and Marie-Francine Moens. 2018. Acquiring common sense spatial knowledge through implicit spatial templates. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Colin Conwell and Tomer Ullman. 2022. Testing relational understanding in text-guided image generation.

Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. 2009. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Jeff Donahue, Lisa Anne Hendricks, Marcus Rohrbach, Subhashini Venugopalan, Sergio Guadarrama, Kate Saenko, and Trevor Darrell. 2016. Long-Term recurrent convolutional networks for visual recognition and description. *IEEE Trans Pattern Anal Mach Intell*, 39(4):677–691.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. 2020. An image is worth 16x16 words: Transformers for image recognition at scale.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Robert Geirhos, Patricia Rubisch, Claudio Michaelis, Matthias Bethge, Felix A. Wichmann, and Wieland Brendel. 2018. Imagenet-trained cnns are biased towards texture; increasing shape bias improves accuracy and robustness.

Mehdi Ghanimifard and Simon Dobnik. 2017. Learning to compose spatial relations with grounded neural language models. In *IWCS 2017 - 12th International Conference on Computational Semantics - Long papers*.

Yunchao Gong, Qifa Ke, Michael Isard, and Svetlana Lazebnik. 2012. A multi-view embedding space for modeling internet images, tags, and their semantics.

Meng-Hao Guo, Jun-Xiong Cai, Zheng-Ning Liu, Tai-Jiang Mu, Ralph R. Martin, and Shi-Min Hu. 2021. Pct: Point cloud transformer. Computational Visual Media, 7(2):187–199.

Kai Han, An Xiao, Enhua Wu, Jianyuan Guo, Chunjing XU, and Yunhe Wang. 2021. Transformer in transformer. In *Advances in Neural Information Processing Systems*, volume 34, pages 15908–15919. Curran Associates, Inc.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition.

Zilong Huang, Xinggang Wang, Lichao Huang, Chang Huang, Yunchao Wei, and Wenyu Liu. 2019. Ccnet: Criss-cross attention for semantic segmentation. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 603–612.

Zhengbao Jiang, Frank F. Xu, Jun Araki, and Graham Neubig. 2020. How can we know what language models know? *Transactions of the Association for Computational Linguistics*, 8:423–438.

Zi-Hang Jiang, Qibin Hou, Li Yuan, Daquan Zhou, Yujun Shi, Xiaojie Jin, Anran Wang, and Jiashi Feng. 2021. All tokens matter: Token labeling for training better vision transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 18590–18602. Curran Associates, Inc.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017. Clevr: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Manoj Kumar, Dirk Weissenborn, and Nal Kalchbrenner. 2021. Colorization transformer.

Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel. 1989. Backpropagation Applied to Handwritten Zip Code Recognition. *Neural Computation*, 1(4):541–551.

Sangho Lee, Youngjae Yu, Gunhee Kim, Thomas Breuel, Jan Kautz, and Yale Song. 2020. Parameter efficient multimodal transformers for video representation learning.

Gen Li, Nan Duan, Yuejian Fang, Ming Gong, Daxin Jiang, and Ming Zhou. 2019a. Unicoder-vl: A universal encoder for vision and language by cross-modal pre-training.

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

Fangyu Liu, Guy Emerson, and Nigel Collier. 2022a. Visual spatial reasoning.

Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L. Yuille. 2019. Clevr-ref+: Diagnosing visual reasoning with referring expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Vivian Liu and Lydia B Chilton. 2022. Design guidelines for prompt engineering text-to-image generative models. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, CHI '22, New York, NY, USA. Association for Computing Machinery.

Xiao Liu, Da Yin, Yansong Feng, and Dongyan Zhao. 2022b. Things not written in text: Exploring spatial commonsense from visual signals. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2365–2376, Dublin, Ireland. Association for Computational Linguistics.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. 2019. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Denis Lukovnikov, Asja Fischer, and Jens Lehmann. 2019. Pretrained transformers for simple question answering over knowledge graphs. In *The Semantic Web – ISWC 2019*, pages 470–486, Cham. Springer International Publishing.

Christoph Molnar. 2022. Interpretable Machine Learning, 2 edition.

Grégoire Montavon, Sebastian Lapuschkin, Alexander Binder, Wojciech Samek, and Klaus-Robert Müller. 2017. Explaining nonlinear classification decisions with deep taylor decomposition. *Pattern Recognition*, 65:211–222.

Muhammad Muzammal Naseer, Kanchana Ranasinghe, Salman H Khan, Munawar Hayat, Fahad Shahbaz Khan, and Ming-Hsuan Yang. 2021. Intriguing properties of vision transformers. In *Advances in Neural Information Processing Systems*, volume 34, pages 23296–23308. Curran Associates, Inc.

Jonas Oppenlaender. 2022. A taxonomy of prompt modifiers for text-to-image generation.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 1–9, Brussels, Belgium. Association for Computational Linguistics.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander Miller. 2019. Language models as knowledge bases? In *Proceedings of the 2019, Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2463–2473, Hong Kong, China. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. 2021. Learning transferable visual models from natural language supervision.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical textconditional image generation with clip latents.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8821–8831. PMLR.

Mohit Shridhar, Lucas Manuelli, and Dieter Fox. 2022. Cliport: What and where pathways for robotic manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pages 894–906. PMLR.

Avanti Shrikumar, Peyton Greenside, and Anshul Kundaje. 2017. Learning important features through propagating activation differences. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 3145–3153. JMLR.org.

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.

Hao Tan and Mohit Bansal. 2019. LXMERT: Learning cross-modality encoder representations from transformers. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5100–5111, Hong Kong, China. Association for Computational Linguistics.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. 2021. Training data-efficient image transformers amp; distillation through attention. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings*

of Machine Learning Research, pages 10347–10357. PMLR.

Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Hervé Jégou. 2020. Training data-efficient image transformers amp; distillation through attention.

TurboSquid, Inc., 2013. Free3d. https://free3d. com/. Accessed: 2022-06-15.

Unity Technologies. 2005. Unity. Version 2020.3.13f1.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz

Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. 2019. Learning two-branch neural networks for image-text matching tasks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 41:394– 407.

Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. 2017. Non-local neural networks.

Ruoxin Xiong and Pingbo Tang. 2021. Machine learning using synthetic images for detecting dust emissions on construction sites. *Smart and Sustainable Built Environment*, 10(3):487–503.

Fuzhi Yang, Huan Yang, Jianlong Fu, Hongtao Lu, and Baining Guo. 2020. Learning texture transformer network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zhou Yu, Jun Yu, Yuhao Cui, Dacheng Tao, and Qi Tian. 2019. Deep modular co-attention networks for visual question answering. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 6274–6283.

Li Yuan, Yunpeng Chen, Tao Wang, Weihao Yu, Yujun Shi, Zihang Jiang, Francis EH Tay, Jiashi Feng, and

Shuicheng Yan. 2021. Tokens-to-token vit: Training vision transformers from scratch on imagenet.

Xikun Zhang, Deepak Ramachandran, Ian Tenney, Yanai Elazar, and Dan Roth. 2020. Do language embeddings capture scales? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 292–299, Online. Association for Computational Linguistics.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for visionlanguage models. *International Journal of Computer Vision*.

Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. 2020. Unified visionlanguage pre-training for image captioning and vqa. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(07):13041–13049.

Appendix

A 3D Models

Here is a list of each of the 21 3D models in the dataset, their relative frequency, and the adjectives, verbs, and adverbs used in generating sentences about them. The frequency column tracks what percent of images contain at least one of that object (eg. a third of all images contain a baseball, though it may not be used in a spatial relation).

Object	Adj(s)	Verb(s)	Adv(s)	Orientability	Frequency
baseball	white, round	sit		none	34%
basket	wicker, rattan, picnic	sit		vertical	41%
bench	park, wood	sit		vertical	17%
bike	sports	sit		full	17%
bus	purple, passenger	drive	quickly	full	16%
cactus	green, small	sit, grow		vertical	19%
car	blue	drive	fast	full	15%
cat	orange	stand, stare	calmly, patiently	full	41%
fan	electric	blow	gently	full	14%
horse	brown	walk	quickly	full	16%
jar	green	sit		vertical	34%
knife	sharp	sit		none[3]	34%
lamp	yellow, monochrome	sit		vertical	20%
laptop	affordable	sit		full	19%
person	army	stand	still	full	19%
plate	white	sit		vertical	33%
swan	white	stand	calmly	full	20%
table	picnic	sit		vertical	15%
table	wooden	sit		vertical	36%
toaster	red	sit		full	32%
torpedo	metal, long	sit		horizontal	8%

[3] It could be argued that the tip of a knife it its front or its top, but for the sake of avoiding inaccurate sentences, I simply do not generate orientation-sensitive sentences when it comes to the knife.

B Spawning Area

Figure 7 shows four spawning areas that have been manually placed around a picnic table. The yellow wireframes do not appear in the final image that is generated. The spawning areas are placed in a way that encourages diverse spatial relations (ie. above and under the table and on either side).

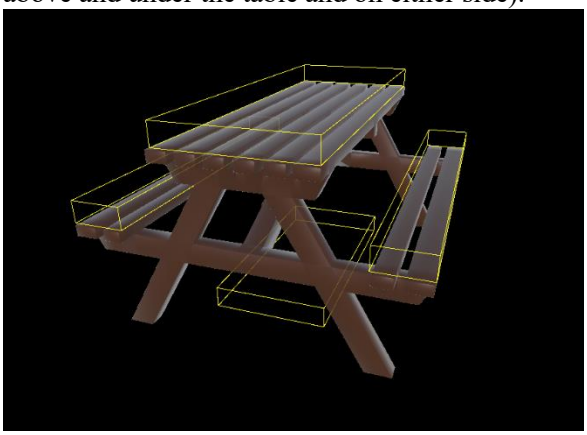

Figure 7: Spawning areas are the four yellow wireframe boxes. Although not visible, these spawning areas are configured to be two-dimensional.

C Collision Detection

Detecting when boxes and spheres overlap is much easier than detecting when arbitrary meshes overlap. Therefore, I approximated the shape of all objects with boxes and sphere. This had almost no impact on the final dataset since shape of the objects was still mostly preserved.

Figure 8: The white wireframes boxes approximate the shape of a bench.

D Camera Spot

The virtual camera is given a random position and a random angle within a camera spot. Figure 8 shows a camera spot. The wireframe rectangular prism shows the coordinates that the camera can be placed in. The wireframe cone shows the range of angles that the camera can be pointed in.

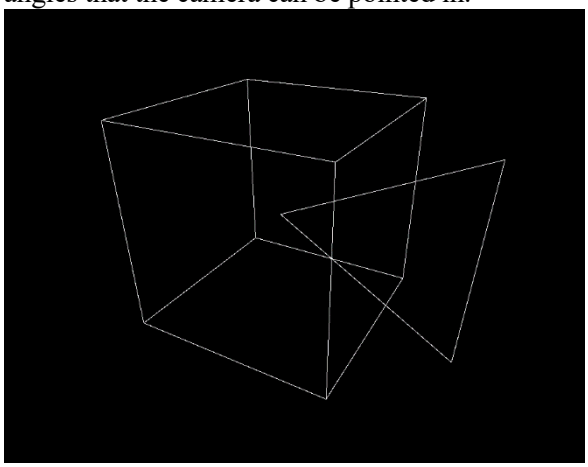

Figure 9: A camera spot pointing to the right

E Skyboxes

The relative frequency of the skyboxes in each of the scenes are the same. Some of the skyboxes were made to depict day, some are nighttime, etc. In total there were 14 different skyboxes broken into the following categories.

Skybox Type	**Amount**
Mostly sunny, day	6
Mostly cloudy	6
Outer space	2
Total	14

Figure 10: The quantity of each of the types of skyboxes.

F Sentence Examples

Here is one example sentence from each test type, which could be true given the correct image.

Basic	The cat is at the bottom.
Filler	Here is a cat which is at the bottom.
Adjective	The orange cat is at the bottom.
Verb	The cat stands at the bottom.
Adverb	The cat stands calmly at the bottom.
Question	Is the cat at the bottom?
Command	Put the cat at the bottom!
Prompt Engineering	The cat is at the bottom of this picture.
	The cat is at the bottom of this 3D render.
	The cat is at the bottom, at the bottom is the cat.
Ungrammatical	Than cat about is an on the bottom

Figure 11: Absolute frame position

Basic	The cat is further left then the toaster.
	The cat is to the left of the toaster.
Filler	Here is a cat which is above the toaster.
Adjective	The orange cat is above the toaster.
Verb	The cat stands above the toaster.
Adverb	The cat stands calmly above the toaster.
Question	Is the cat above the toaster?
Command	Put the cat above the toaster!
Prompt Engineering	In this picture, the cat is above the toaster.
	A 3D render of a cat above a toaster.
	The cat is above a toaster, the toaster above a plate.
Ungrammatical	Than cat about is above so toaster.

Figure 12: Relative frame position

Basic	The cat is to the toaster's left.
	The cat is left of the toaster.
Filler	Here is the cat which is left of the toaster.
Adjective	The orange cat is to the toaster's left.
Verb	The cat stands to the toaster's left.
Adverb	The cat stands calmly to the toaster's left.
Question	Is the cat to the toaster's left?
Command	Put the cat to the toaster's left!
Prompt Engineering	In this picture, the cat is to the toaster's left.
	A 3D render of a cat to a toaster's left.
	The cat is to the toaster's left, the cat to the toaster's left.
Ungrammatical	Cat is to toaster's the left.

Figure 13: Relative world placement

Basic	The cat is sideways.
	A sideways cat.
Filler	Here is a plate which is sideways.
Adjective	The orange cat is sideways.
Verb	The cat stands sideways.
Adverb	The cat stands calmly sideways.
Question	Is the cat sideways?
Command	Put the cat sideways!
Prompt Engineering	A picture of a sideways cat.
	A 3D render of a sideways cat.
	The cat is sideways, a sideways cat.
Ungrammatical	Than an about cat is sideways.

Figure 14: Orientation

G Scenes

Here is one example image from each of the five scenes.

Figure 15: Cityscape

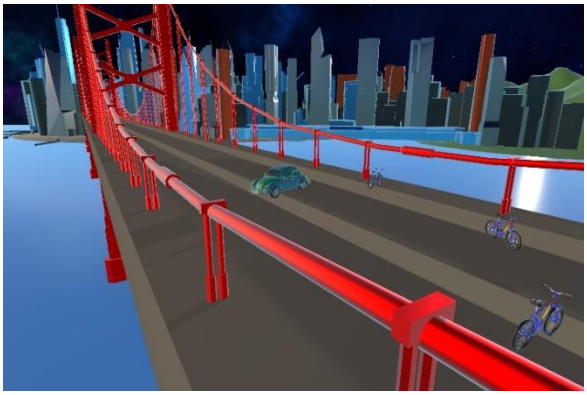

Figure 16: Bridge

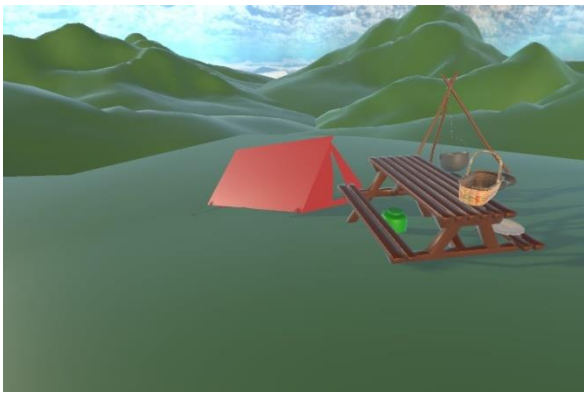

Figure 17: Hilly countryside

Figure 18: Indoors

Figure 19: Floating