Abstract
Recent development in vision-language approaches has instigated a paradigm shift in learning visual recognition models from language supervision. These approaches align objects with language queries (e.g. “a photo of a cat”) and improve the models’ adaptability to identify novel objects and domains. Recently, several studies have attempted to query these models with complex language expressions that include specifications of fine-grained semantic details, such as attributes, shapes, textures, and relations. However, simply incorporating language descriptions as queries does not guarantee accurate interpretation by the models. In fact, our experiments show that GLIP, the state-of-the-art vision-language model for object detection, often disregards contextual information in the language descriptions and instead relies heavily on detecting objects solely by their names. To tackle the challenges, we propose a new description-conditioned (DesCo) paradigm of learning object recognition models with rich language descriptions consisting of two major innovations: 1) we employ a large language model as a commonsense knowledge engine to generate rich language descriptions of objects based on object names and the raw image-text caption; 2) we design context-sensitive queries to improve the model’s ability in deciphering intricate nuances embedded within descriptions and enforce the model to focus on context rather than object names alone. On two novel object detection benchmarks, LVIS and OminiLabel, under the zero-shot detection setting, our approach achieves 34.8 APr minival (+9.1) and 29.3 AP (+3.6), respectively, surpassing the prior state-of-the-art models, GLIP and FIBER, by a large margin.

1 Introduction
Training visual recognition models to classify or detect objects with a fixed set of pre-defined categories has been the convention for a long time. However, models trained using this approach often encounter difficulties when adapting to unfamiliar concepts and domains. Recently, there has been a paradigm shift towards training visual recognition models with language supervision, using a contrastive objective on a large amount of image-text data containing a diverse range of visual concepts. These models can then be transferred to downstream tasks via language queries. For example, CLIP [33] can perform image classification by using a template query like “a photo of [class name]”; GLIP [22] can perform object detection by querying the model with “Detect: person, cat, dog ···”.

Early applications of these models typically utilize simple language queries that consist of object names. However, language queries can contain much richer and more comprehensive information, such as object attributes, shapes, textures, and relations. These pieces of information can be especially useful for identifying novel visual concepts that do not appear in the training corpus or specifying
A kind of tool, wooden handle with a round head, used for pounding or hammering.

A kind of tool, long handle, sharp blade, could be used for chopping wood.

Headset, a kind of electronic device, has two earpieces connected to a band...

Microphone, a kind of audio device, long, thin, could be handled, could be connected...

Eclair, a kind of food, long, cylindrical pastry, filled with cream, topped with chocolate.

Tart, a kind of food, round, could be filled with fruits, could be served with cream.

A clown making a balloon animal for a pretty lady.

A clown kicking a soccer ball for a pretty lady.

Headset, a kind of electronic device, has two earpieces connected to a band...

Microphone, a kind of audio device, long, thin, could be handled, could be connected...

Eclair: 0.39  Eclair: 0.58

Ours  GLIP

Eclair, a kind of food, long, cylindrical pastry, filled with cream, topped with chocolate.

Tart: 0.55

Ours  GLIP

Tart, a kind of food, round, could be filled with fruits, could be served with cream.

A clown making a balloon animal for a pretty lady.

A clown kicking a soccer ball for a pretty lady.

Headset: 0.5  Headset: 0.3

Ours  GLIP

Headset, a kind of electronic device, has two earpieces connected to a band...

Microphone: 0.38  Microphone: 0.11

Ours  GLIP

Microphone, a kind of audio device, long, thin, could be handled, could be connected...

Detect with specifications for shape & subpart

Detect with specifications for relation

Detect with specifications for shape & subpart (w/o object name)

Detect small objects

Figure 1: Comparison between our model (DES-CO-GLIP) and the baseline (GLIP [22]). Each image is paired with a positive query for target object and a negative query for confusable object. A successful model should locate the target object and ignore the confusable object in the image based on fine-grained specifications for shapes, subparts, relations, etc. We highlight the descriptions that match and not match to the queried object in blue and red, respectively. Results show that our model can successfully localize the target object and suppresses the negative query even for the difficult cases when the object name is not in the query or the object.

Specific needs. For example, the concept of “mallet” can be described as “a kind of tool, wooden handle with a round head” (Figure 1 bottom-left). This decomposes the task of object recognition into two tasks: 1) recognizing fine-grained details (such as attributes, sub-parts, shapes, etc.) and 2) aligning them to the descriptions. Several studies [22, 37, 28] have explored the idea of guiding language-based recognition models using such descriptive prompts. However, few existing models take complex queries into account during training. As a result, current models often struggle with recognizing intricate object names, attributes, and relations described in natural language sentences (see examples in Figure 1). These observations are consistent with findings from recent research works [47, 39].

In this paper, we develop a vision-language model capable of leveraging description-rich language queries to perform object detection. This work aligns with the recent surge of interest in instruction/prompt-aware vision-language models (see a discussion in Section 2). Our goal is to equip VLMs with the ability to comprehend complex language input describing visual concepts, similar to the capability of large language models. We specifically study instruction/prompts in the form of descriptive queries. We focus on the task of object detection as it requires fine-grained recognition and is more challenging than image-level tasks. However, our method can be generalized to other vision tasks such as classification and segmentation [53].

We identify two major challenges that prevent existing models from efficiently utilizing rich language descriptions: (1) Fine-grained descriptions are rare in image-caption data that used in training current VLMs. This resembles the reporting bias phenomenon [31]: when writing captions for images, humans tend to directly mention the entities rather than give a detailed description. For example, for the bottom-left image in Figure 1 one may directly write “A toy bear holding a mallet” rather

---

2We count the region-label data used by models like GLIP as image-caption data because the labels are converted into captions through templates.
than mentioning the shape and sub-parts of “mallet”. (2) Even when provided with data rich in descriptions, models often lack the incentive to leverage these descriptions effectively. During training, the primary objective is to align positive phrases with relevant regions while suppressing negative phrases. However, if positive/negative phrases can be distinguished without descriptions, the training mechanism fails to incentivize the model to use the provided description. For example, a positive phrase like “A toy bear holding a mallet, which has a wooden handle with a round head,” and a negative phrase like “A toy bear holding an ax, which has a long handle and a sharp blade,” can be differentiated based solely on the words mallet and ax. This issue resembles the issue discovered by [47], where vision-language models ignore word order and treat a query as a “bag-of-words” due to insufficient incentives from the contrastive objective. In addition, we also find that current models suffer severe hallucination when given natural language queries (in contrast to “template-like” queries) due to shortcuts introduced in training query formulation. This can be seen in the bottom-right picture of Figure 1, where GLIP hallucinates and predicts multiple wrong boxes for “microphone” while “microphone” does not appear in the image.

Based on the observations, we present a Description-Conditioned (DesCo) paradigm of learning object recognition models from language descriptions based on two synergistic ideas:

1. **Generating descriptions with large language models.** Instead of learning from raw image-caption pairs, we use large language models as a world knowledge base and generate detailed descriptions based on the original caption. We prompt GPT-3 [2] with “What features should object detection models focus on for (an entity in the caption)?”. This serves as a scalable approach to transfer the image-caption data into image-description data.

2. **Context-sensitive query construction.** As discussed, even if we provide descriptions during pre-training, models can still ignore the language context. Our solution is to create a “context-sensitive query”, which is a set of positive and negative phrases that can only be distinguished by reading the descriptions (Figure 2). We explore two strategies: 1) constructing “Winograd-like” [12, 39] queries by using large language models to generate confusable object descriptions and captions and 2) generalizing the original grounding task to allow full-negative queries, reducing hallucination.

We apply our approach to fine-tune two state-of-the-art language-conditioned object detection models GLIP [22] and FIBER [7]. We use the same raw training data as the baselines but convert the data into description-rich queries. We evaluate our methods under two scenarios. (1) Zero-shot generalization to novel categories (LVIS [10]), where we use GPT-3 to generate descriptions given class names. DesCo-GLIP (Tiny) improves upon GLIP (Tiny) by 10.0 AP\text{r}, even outperforming the larger GLIP (Large); DesCo-FIBER improves upon FIBER by 9.1 AP\text{r}. (2) Zero-shot generalization to natural descriptions given by humans (OmniLabel [34]). DesCo-GLIP and DesCo-FIBER improve upon the baselines by 4.5 AP and 3.6 AP, setting a new state-of-the-art performance level. Code will be released at [https://github.com/liunian-harold-li/DesCo](https://github.com/liunian-harold-li/DesCo).

2 Related work

**Language-based visual recognition models.** Visual recognition models are typically trained to make predictions based on a fixed set of classes [19, 5, 23, 35, 30, 51]. The trained models are hard to generalize to open-domain settings where the models need to deal with concepts that are novel or involve complex compositions. To alleviate the limitation, recent studies develop visual recognition models that take in language queries, i.e., language-based recognition. This line of research can be traced back to early work of generalizing image classification [38] and object detection [1] models with word embeddings. Recently, CLIP [33] reformulates image classification as image-text matching and pre-trains models on large-scale image-caption pairs to learn transferrable representations. They demonstrate strong zero-shot performance on various classification tasks. Recent work has applied the technique to fine-grained recognition tasks, such as object detection [16, 9, 22, 50, 49, 3, 29, 7, 25], and segmentation [20, 8, 15, 42]. These works either use pure image-text data as supervision [42], or reformulate labeled data into image-text data [20], or pseudo labels image-text data with fine-grained labels [22]. Orthogonal to architecture design or scaling-up, which is the focus of many prior studies, this paper points out that the vanilla way of using image-text data is insufficient and studies how to train these models to take more flexible and informative language queries.
Prompting vision-language models. As vision recognition models become language-aware, there is a growing interest in studying whether these models can take complex language prompts, such as task instructions (e.g., GPV [11], SEEM [54], VisionLLM [41]), descriptions [21], or even dialogues (e.g., LLaV [24]). We study specifically descriptive prompts, which are especially useful for generalizing to novel categories and customized detection needs; a model that can understand descriptive prompts can also serve as the backbone for supporting aforementioned other types of prompts. Similar to our work, K-LITE [37] proposes to retrieve knowledge for a visual concept using external knowledge bases, then use the enriched concepts for image classification or object detection; similar techniques have also been proposed by [28, 43]. DetCLIP [44] builds a large-scale concept dictionary, based on which they provide definitions from WordNet. Different from these studies, our methods show that simply presenting the descriptions at training or inference time is not enough; we propose a simple technique to force models to focus on the provided descriptions (Section 3.2.2).

3 Approach

In this section, we first briefly introduce language-based object detection models, then illustrate the details of our proposed approach.

3.1 Background

We give an introduction to language-based object detection models [16][22][7], which take a language query and an image as inputs, and predict bounding boxes and their alignment to phrases in the language query. In the following, we use GLIP as an example.

Baseline: Grounded Language-Image Pre-training (GLIP). At the center of these approaches is “reformulating any task-specific fixed-vocab classification problem as a task-agnostic open-vocabulary vision-language matching problem” [48]. The best example is CLIP which reformulates image classification as image-text matching. Similarly, GLIP unifies training data into a grounding format: \((I, Q, B, T)\). \(I\) is the image; \(Q\) is the text query; \(B \in R^{N \times 4}\) is the bounding boxes; \(T \in \{0, 1\}^{N \times K}\) indicates the ground-truth alignment label between the \(N\) bounding boxes and \(K\) tokens in the query. The key is how to formulate the query with data from different sources:

- **Detection data.** For object detection data, the query is the concatenation as a list of object classes, such as “Detect: person, bicycle, car, ···, toothbrush”. Note that negative object classes are included in the query; this makes such query-based detection models equivalent to classical detection models when all classes in the dataset can be included in the prompt.

- **Grounding data.** Typically, \(Q\) is an image caption, containing entities that can be aligned to annotated object regions [22]. For example, “A toy bear holding a mallet” is the caption; “toy bear” and “mallet” are the “groundable” entities. For densely annotated grounding data (multiple captions for one image) [32], we can concatenate multiple captions into a longer query. Image-caption data (without annotated boxes) can be transferred into grounding data via pseudo labeling with a grounding model [22].

Given \(I\) and \(Q\), we compute the alignment scores \(S_{\text{ground}}\) between image regions and words in the query:

\[
O, L = \text{Enc}(I, Q), S_{\text{ground}} = OL^\top, \mathcal{L} = \text{loss}(S_{\text{ground}}, T) + \mathcal{L}_{\text{loc}}
\]

where \(L \in R^{K \times d}\) is the contextual token features and \(O \in R^{N' \times d}\) are the regions features. \(\text{ENC}\) is a vision and language encoder that takes both image and text as inputs and fuses their representations. The training loss contains the region-word matching loss and a localization loss as in conventional object detection models.

Inference with language query. At inference time, the model can be used to locate entities/class names appearing in the query. One could simply provide a list of candidate object names (as in the detection data training format). [22] also show the promise of using descriptions for generalization to novel concepts; however, we show that while GLIP can be influenced by the description, it does not always take the details in the description into account.
3.2 Learning with language descriptions

To train object recognition models that fully utilize language descriptions, we propose to generate descriptions with large language models and construct context-sensitive queries during training. The following subsections provide further details.

3.2.1 Description generation with large language models

Fine-grained descriptions could be scarce in image-caption data due to reporting bias. While this problem can be alleviated by scaling up the pre-training corpus, we show that large language models can be used to effectively generate the descriptions and thus enrich our training corpus.

We leverage a large language model to transform a query $Q$ into a description-rich query $L_{LM}(Q)$. In this work, we only focus on generating descriptions for entities mentioned in the original query. We construct a vocabulary consisting of 10K entities appearing frequently in the pre-training corpus. For each entity, we prompt a large language model: what features should object detection models focus on for {entity}? We find that large language models give high-quality responses (see examples in Figure 1 and Figure 4). More details on the prompts and API cost can be found in the appendix.

3.2.2 Context-sensitive query construction

Can we simply add the description-rich data to the pre-training corpus? An intuitive idea is to append the description to the original entity to form a description-rich training query (e.g., “Detect: mallet, bear···” → “Detect: mallet, a kind of tool, wooden handle···”). However, we find that models naively trained with these description-rich queries still do not exhibit “context-sensitivity”, i.e., they make predictions solely based on the entity names while ignoring other contexts (see Section 4.1 for quantitative analysis). As a result, we observe no evident benefit in incorporating descriptions during inference (Table 3). In the following, we elaborate on why the model learns to ignore the descriptions and propose two solutions.

Model learn statistical shortcuts. We first illustrate that without careful design, the model could learn two statistical shortcuts that make them insensitive to descriptive queries.

(1) Entity shortcut. The model is trained to align the entities in the query to image regions (this includes predicting “no-alignment” for entities not appearing in the image). Intuitively, if the
alignment can be predicted without relying on the context information in the query, then the model is not incentivized to focus on the context information. Figure 2 illustrates this issue with an example. The left side shows the training data of GLIP, where the top query (“Detect: Mallet. Bear. Cat...”) comes from detection data and the bottom query (“A toy bear holding a mallet.”) comes from grounding data. The problem with such queries is that they can be grounded by only focusing on the entity names and ignoring the context. We denote the gold alignment label of regions as $T$, the entities in the query as $E$, and the non-entity part (context) of the query as $C$. The mutual information between $C$ and $T$ given $E$ and the image $I$ is effectively zero: $I(T; C|E, I) = 0$. That is, the non-entity parts of the queries do not affect the label of the region. Adding descriptions to $C$ does not help as the mutual information still stays zero. Training models on such data will not encourage the model to focus on the descriptions as they provide no additional information. This is similar to the “memorization overfit” issue observed in [45]: the model can simply choose to “memorize” the alignment between the entities and regions.

(2) Format shortcut (hallucination). Popularized by GLIP [22], a line of work adopts a unified view of phrase grounding and object detection: detection can be seen as language-context-free grounding while grounding can be seen as language-context-dependent detection. However, this unification is still imperfect: phrase grounding (or referring expression [46]) traditionally only concerns locating entities in a caption that always exist in the image; thus the model learns to always treat the natural-language queries (in contrast to the template-like queries) as positive and tries to ground every entity mentioned in the sentence. This will result in failure examples as illustrated in the bottom-right picture of Figure 1. Such “hallucination” can be commonly seen on models trained on language grounding data [16]; these models are almost incapable of distinguishing positive and negative “natural-language-like” queries.

Constructing context-sensitive queries. This motivates our solution of creating queries that are hard to solve without context (Figure 2 and Figure 3). We explore two strategies in this study.
We construct training queries similar to the Winograd format. For example, when training on detection data, instead of “Detect: mallet, a kind of tool, ···”, we remove the entity name “mallet” from the query and sample another description of a “confusable” entity that is also a kind of tool. Pairing the descriptions of the two “confusable” entities creates a strong supervision signal (the middle example in Figure 2): the alignment label (0 or 1) of the word “tool” now depends on its context. The confusable entities are obtained by prompting the large language models as well. Similarly, for training on grounding data, we prompt language models to generate confusable captions that differ from the original captions only by a few words (the example on the right in Figure 2). Note that the label of the word “mallet” is now affected by the context: the first “mallet” is assigned 0 as the caption (“A polar bear holding a mallet”) is negative. Mixing in such hard negative captions encourages the model to focus on the context surrounding the entities, such as relations to other entities. To make the confusable caption generation process scalable for image-caption data, we first perform in-context inference and ask GPT-3 to generate around 50K negative captions based on positive captions; then we distill this knowledge to the open-sourced LLaMA-7B model that is instruction-finetuned using low-rank adaptation and perform inference on large-scale image-caption data.

To resolve the hallucination issue, we generalize the original grounding task. Instead of always feeding the model a query that matches the image, we allow the query to be negative. Thus, the model cannot blindly ground all entities mentioned in the query; implicitly, it needs to perform image-text matching as well as phrase grounding. This was partly done in GLIP (see Appendix C.1 in the original GLIP paper), but the query still contains at least one positive entity. In this work, we pack several captions/queries to form a query. The positive caption can be dropped from the query, and the query would contain all negative captions in this case (Figure 3). This is crucial for reducing model hallucination.

### 4 Experiment

In this section, we first investigate whether current models (GLIP) can utilize language descriptions out-of-the-box; then we show that our method allows the model to utilize language descriptions and improves performance on LVIS and OmniLabel significantly.

#### 4.1 Can language-conditioned detection models utilize language descriptions?

As a proof of concept, we first show the GLIP struggles to utilize language descriptions out of the box and analyze the failure patterns.

**GLIP does not effectively utilize language descriptions.** We make an attempt at using descriptions to transfer GLIP to LVIS, which contains over 1,200 classes. The process is similar to that of DesCo-GLIP. For each category, we prompt a large language model (GPT-3) to give details descriptions (as in Section 3). We append the description to the original class name to form a new query. An example of the queries can be seen shown in Figure 1 (bottom row). Directly appending the description to the object name at inference time only degrades the performance: GLIP-T achieves 20.8 AP on rare categories while appending the descriptions makes the performance drop to 12.2 AP. This is likely due to model hallucination on natural-language-like queries.

**GLIP is insensitive to context changes.** Examining the model predictions, we find that the model does not utilize language descriptions; it ignores the descriptions and tends to only focus on entity names, as we hypothesized. To quantitatively verify the phenomenon, we introduce a context-sensitivity test, inspired by the WinoGround benchmark. For each image, we provide the model with a positive query $Q^+$ describing an object that appears in the image and a negative query $Q^-$ describing a confusable object. The original object names are removed from the query. An example of the test is shown in Figure 1 (bottom left), where the model is challenged to distinguish “mallet” and “ax”. $Q^+$ and $Q^-$ describe objects from the same general category (e.g., both are “a kind of tool”) while differing in other aspects, similar to the Winograd test.

### Table 1: GLIP is insensitive to context changes compared to DesCo-GLIP.

<table>
<thead>
<tr>
<th>Model</th>
<th>$\Delta$Box</th>
<th>$\Delta$Conf</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLIP [22]</td>
<td>0.291</td>
<td>0.05</td>
<td>4.7</td>
</tr>
<tr>
<td>DesCo-GLIP</td>
<td>0.381</td>
<td>0.11</td>
<td>12.4</td>
</tr>
</tbody>
</table>

3https://github.com/tloen/alpaca-lora
Table 2: Zero-shot transfer to LVIS and OmniLabel. Numbers that are grayed out are supervised models. DESCO-GLIP and GLIP-T are directly comparable; DESCO-FIBER and FIBER-B are directly comparable; the rest are listed for reference and not directly comparable.

Intuitively, if a model can effectively utilize the descriptions, it should exhibit two properties: 1) it should give higher alignment scores to entities in $Q^+$ compared to $Q^-$; 2) even if the model cannot “guess” the hidden entity, at least, the model predictions should change drastically when given two different descriptions. We thus introduce two metrics. 1) AP, which measures how accurate the model’s predictions are. 2) $\Delta$Box and $\Delta$Conf, which are the differences between the model’s predictions for $Q^+$ and $Q^-$. $\Delta$Box measures the changes in box coordinates while $\Delta$Conf measures the changes in alignment scores of boxes. Details of the metrics are in the appendix.

We find that the baseline model not only cannot identify the correct description (low AP); but it effectively ignores the language context (low $\Delta$Box and $\Delta$Conf) (Table 1). On average, the confidence of the predicted boxes changes only 0.05 between $Q^+$ and $Q^-$. One could see the examples in Figure 1. GLIP models make almost identical predictions for two different queries. Such insensitivity to language context makes it infeasible and unreliable to use descriptions to control model predictions.

4.2 Setup

In this section, we introduce the experimental setup of DESCO-GLIP and DESCO-FIBER.

Models. We perform experiments on GLIP [22] and FIBER [7]. Their visual backbone is Swin Transformer [27] and the text backbones are BERT [6] for GLIP and RoBERTa [26] for FIBER. Both models use Dynamic Head [4] as the detection architecture. Built upon the two models, we introduce two model variants: DESCO-GLIP and DESCO-FIBER.

Datasets. Following GLIP [22], we train the models on 1) O365 (Objects365 [35]), consisting of 0.66M images and 365 categories; 2) GoldG that is curated by MDETR [16] and contains 0.8M human-annotated images sourced from Flickr30k [32], Visual Genome [18], and GQA [14]; 3) CC3M [36]: the web-scraped Conceptual Captions dataset with the same pseudo-boxes used by GLIP. We down-sample CC3M to around 1.4M images to save training costs, based on whether high-confidence boxes exist in the image. As illustrated in Section 3, we convert the text caption of each instance into a detailed language description to construct description-rich data.

To evaluate how well the models generalize to novel concepts, we perform a zero-shot evaluation on the LVIS [10] and OmniLabel [34] datasets. LVIS is a popular dataset that has over 1,200 object categories with a challenging long tail of rare objects; OmniLabel is recently proposed and focuses on object detection with diverse and complex object descriptions in a naturally open-vocabulary setting. For evaluation on LVIS, for each category, we append the GPT-3 generated description to the category name; we group several descriptions into one query to save inference time. More details on the evaluation are in the appendix. For OmniLabel evaluation, we follow the original evaluation protocol without modifications. We also verify that the models still possess the ability to perform the
Table 3: Ablation study. Directly appending the description does not improve performance on rare categories (Row 1 v.s. Row 2, LVIS APr). Constructing context-sensitive queries is crucial.

<table>
<thead>
<tr>
<th>Row</th>
<th>Model</th>
<th>LVIS MiniVal</th>
<th>OmniLabel COCO</th>
<th>Context Sensitivity</th>
<th>AP</th>
<th>APc</th>
<th>APd</th>
<th>APd-P</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>GLIP-T</td>
<td>20.8 21.4 31.0 26.0</td>
<td>18.7 45.7 11.7 31.2</td>
<td>0.291 0.05 4.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>+ Description w/ Entity Name</td>
<td>20.5 23.9 35.5 29.2</td>
<td>23.6 47.4 14.7 36.0</td>
<td>0.293 0.06 5.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>+ Description w/o Entity Name</td>
<td>25.6 25.9 35.9 30.7</td>
<td>24.0 46.8 16.0 37.0</td>
<td>0.382 0.10 10.7</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>+ Description w/o Name + Hard</td>
<td>26.5 27.1 35.8 31.3</td>
<td>24.7 48.2 16.6 36.2</td>
<td>0.381 0.10 10.5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Implementation details. We initialize DESCO-GLIP from the GLIP-T checkpoint and DESCO-FIBER from the FIBER-B checkpoint. We fine-tune the models on both the original data and the new description-rich data. For DESCO-GLIP, we fine-tune with a batch size of 16 and a learning rate of $5 \times 10^{-5}$ for 300K steps; for DESCO-FIBER, we fine-tune with a batch size of 8 and a learning rate of $1 \times 10^{-5}$ for 200K steps. Experiments can be replicated with 8 GPUs each with 32GB memories.

4.3 Zero-Shot Transfer to LVIS and OmniLabel

LVIS. Our method exhibits significant enhancements over the baselines on the LVIS MiniVal dataset (Table 2). Specifically, we achieve notable improvements over GLIP and FIBER, surpassing them by 8.6 and 5.7 AP points, respectively. These enhancements are particularly prominent in the case of rare object categories, as demonstrated by the performance differences in APr, with an increase of 10.0 for GLIP and 9.1 for FIBER. Results on the Val 1.0 set are in the appendix.

These results provide strong evidence for the effectiveness of integrating comprehensive language descriptions into our approach. They also confirm that our methods can benefit from these descriptions, particularly when dealing with novel visual concepts. Notably, our best-performing model achieves an impressive APr of 34.8 and AP of 39.5, surpassing supervised models by a substantial margin.

OmniLabel. Our method also exhibits significant improvements over baselines on the OmniLabel dataset (Table 2). OmniLabel assesses model performance using both plain categories (APc) and free-form descriptions (APd). Because our models are trained with description data, they naturally excel in supporting this type of query, leading to substantial increases in APd compared to the baselines. Specifically, DESCO-GLIP achieves a notable improvement of +4.6, while DESCO-FIBER achieves an even more impressive improvement of +5.0. Furthermore, our model’s effectiveness extends beyond free-form descriptions to encompass plain categories as well, as illustrated in the table. This highlights the robustness of our method across different evaluation settings and its ability to achieve improvements in various types of queries. Our method wins the 1st place in the Omnilabel challenge 2023 on all three tracks (see Appendix for details).

4.4 Ablation Study

In this part, we perform several ablations on our proposed methods to investigate the importance of each component. The ablation models are initialized from GLIP-T and trained for 100K steps.

Directly appending descriptions. We examine the impact of directly adding language descriptions to text queries, without incorporating context-sensitive query construction. The results are presented in Row 2 of Table 3. The performance on rare categories (APr) sees no improvement but decreases. To further evaluate the sensitivity of the model to contextual changes, we conduct the same context sensitivity analysis as the one described in Section 4.1. The context sensitivity of the model almost remains unchanged (Row 1-2): $\Delta$Box changes only 0.002 and $\Delta$Conf changes only 0.01. The results indicate that the model remains as insensitive to context changes as the baseline model. This suggests that the model struggles to accurately interpret and effectively utilize the provided language descriptions when context-sensitive query construction is removed.
Scarecrow, a kind of object, tall, with a straw in its mouth, could have a hat, could be made of straw.

From GPT-Curie:
Scarecrow, a kind of decoration, made of straw, has a hat and clothes, could have a face.

From GPT-Davinci:
Rollerblade, a kind of sports equipment, blades that rotate on the ground, attached to a boot, used for skating.

Dropping the entity name. As in Section 3.2.2, we hypothesize that removing the center entity name can force the models to concentrate on the contextual information. Remarkably, the results presented in Table 3 (Row 2-3) demonstrate that this simple and intuitive approach proves to be highly effective. It significantly enhances the model’s contextual sensitivity while concurrently improving object detection performance.

Hard negative captions. We also investigate the effectiveness of using language models to generate hard negative captions. As shown in Row 4 of Table 3, including negative phrases can improve the model detection performance across datasets while preserving its robust contextual comprehension. These results indicate that this technique effectively enhances the model’s ability to grasp the subtleties embedded in the given language descriptions.

Language description quality. We explore the effect of the language model size on detection performance. We evaluate the pre-trained DESCo-GLIP on LVIS with descriptions generated from the GPT families. As presented in Table 4, higher-quality language models significantly improve object detection performance. This finding highlights the importance of employing strong language models, as they possess the ability to embed valuable visual information through extensive pre-training. We showcase two examples in Figure 4.

<table>
<thead>
<tr>
<th>GPT</th>
<th>APr</th>
<th>APc</th>
<th>APf</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ada</td>
<td>19.9</td>
<td>23.2</td>
<td>33.7</td>
<td>28.0</td>
</tr>
<tr>
<td>babbage</td>
<td>24.2</td>
<td>26.7</td>
<td>36.5</td>
<td>31.3</td>
</tr>
<tr>
<td>curie</td>
<td>24.7</td>
<td>28.4</td>
<td>38.2</td>
<td>32.8</td>
</tr>
<tr>
<td>davinci</td>
<td>30.8</td>
<td>30.5</td>
<td>39.0</td>
<td>34.6</td>
</tr>
</tbody>
</table>

Table 4: Detection performance improves when language model size grows.

5 Conclusion and limitations

In this study, we introduced a new paradigm of learning object detection models from language supervision. We show that large language models can be used to generate rich descriptions and the necessity to construct context-sensitive queries. We hope that our method sheds light on empowering vision models with the ability to accept flexible language queries.

While we greatly improve the models’ ability to understand flexible language queries, our method has several limitations that can be addressed in future work. 1) We use a large language model to automatically generate the descriptions, which inevitably introduces noise as not all generated descriptions are accurate or beneficial for representation learning. Future work could consider automatically selecting useful descriptions sampled from the language model, similar to [43]. 2) The format of the descriptions we explored is still limited (e.g., “{entity}, a kind of {type}, {list of simple features}”); it might be useful to consider more diverse descriptions by prompting the language model with more diverse prompts. 3) Similar to large language models, querying our model also requires a certain amount of prompt engineering. Future work could explore how to make the model more robust to different kinds of queries.

4https://platform.openai.com/docs/models
References


A Approach

A.1 Description generation with large language models

We prompt davinci-003 with the following prompt:

<table>
<thead>
<tr>
<th>Question: What features should object detection models focus on for a given input? Answer:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input: zucchini, Output: &quot;type&quot;: &quot;vegetable&quot;, &quot;description&quot;: &quot;cylindrical, green, smooth; could have brown and rough stems; could be sliced into round pieces; could have green leaves&quot;, &quot;similar objects&quot;: [&quot;cucumber&quot;, &quot;eggplant&quot;, &quot;green bean&quot;]</td>
</tr>
<tr>
<td>Input: zebra, Output: &quot;type&quot;: &quot;animal&quot;, &quot;description&quot;: &quot;black and white stripes; has a long mane&quot;, &quot;similar objects&quot;: [&quot;horse&quot;, &quot;giraffe&quot;, &quot;elephant&quot;]</td>
</tr>
<tr>
<td>Input: apple, Output: &quot;type&quot;: &quot;fruit&quot;, &quot;description&quot;: &quot;red, round, has a stem&quot;, &quot;similar objects&quot;: [&quot;orange&quot;, &quot;banana&quot;, &quot;pear&quot;]</td>
</tr>
<tr>
<td>Input: wok, Output: &quot;type&quot;: &quot;cooking tool&quot;, &quot;description&quot;: &quot;round, deep, has a handle&quot;, &quot;similar objects&quot;: [&quot;pan&quot;, &quot;pot&quot;, &quot;frying pan&quot;]</td>
</tr>
<tr>
<td>Input: ambulance, Output: &quot;type&quot;: &quot;vehicle&quot;, &quot;description&quot;: &quot;red; has a glaring siren; could with a stretcher&quot;, &quot;similar objects&quot;: [&quot;police car&quot;, &quot;taxi&quot;, &quot;garbage truck&quot;]</td>
</tr>
<tr>
<td>Input: lantern, Output: &quot;type&quot;: &quot;lighting tool&quot;, &quot;description&quot;: &quot;round; could be made of papers&quot;, &quot;similar objects&quot;: [&quot;lamp&quot;, &quot;flashlight&quot;, &quot;candle&quot;]</td>
</tr>
<tr>
<td>Input: {entity}</td>
</tr>
</tbody>
</table>

Table 5: Text prompt used to sample descriptions from large language models.

We construct a vocabulary of $10^k$ entities by extracting noun phrases from the pre-training text corpus (GoldG and CC3M) using NLTK. We query the language model to generate descriptions for $10^k$ entities; this can be finished within 1 day via OpenAI API.

A.2 Context-sensitive query construction

As shown in Table 5, when prompting the language model, we also ask the model to name a few “similar objects”. Thus, when constructing a query, we include both positive descriptions and several negative descriptions for such “similar objects”. GLIP has a max query length of 256 tokens. On average, we can pack 8 descriptions into one query. We randomly drop descriptions if the length exceeds the length limit.

A.3 Query construction of the original GLIP

Algorithm 1: Generating Queries for Detection Data

Input: $T, E, V$

1: $Q^- \leftarrow \text{RandSample}(V \setminus E)$
2: $Q \leftarrow \text{ShuffleConcate}(E \cup Q^-)$
3: $Q, T \leftarrow \text{LabelAssign}(Q, T, E)$

Algorithm 2: Generating Queries for Grounding Data

Input: $T, D, C$

1: $Q^- \leftarrow \text{RandSample}(D \setminus \{C\})$
2: $Q \leftarrow \text{ShuffleConcate}(\{C\} \cup Q^-)$
3: $Q, T \leftarrow \text{LabelAssign}(Q, T, C)$

Figure 5: Algorithms for generating queries from detection data and grounding data for GLIP. Algorithm 1: Compare to DesCo, note that no positive entities are dropped from the query; thus $B$ is not involved in the query construction. Algorithm 2: $D$ is the corpus of image captions. Compare to DesCo, note that the positive caption $C$ is always included in the query; this creates the hallucination issue.

B Experiments

B.1 Context-sensitivity test

To create the test, we go over the LVIS validation set and for each image that has a rare object (defined by the LVIS taxonomy), we query the model with a $Q^+$ (the description of the rare object,
without the object name) and a $Q^-$ (the description of a confusable object, without the object name). The confusable object comes from prompting the LLM as shown in Table 5. Then we calculate the difference between the predictions made by the model for $Q^+$ and $Q^-$. For $Q^+$ and $Q^-$, the model will give two sets of predictions (two lists of boxes). We first match the two lists of boxes by their IoU overlap. Then $\Delta\text{Box}$ is the percentage of boxes that have high IoU overlap (>0.5) between the two sets of predictions; $\Delta\text{Score}$ is the confidence score changes for those matched boxes. With higher $\Delta\text{Box}$ and $\Delta\text{Score}$, the model predictions change more drastically. The evaluation data and code will be released upon acceptance.

### B.2 Evaluation on COCO and Flickr30K

<table>
<thead>
<tr>
<th>Model</th>
<th>COCO mAP</th>
<th>Flickr30K Val</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLIP-T*</td>
<td>44.6</td>
<td>85.6 95.8 97.3</td>
</tr>
<tr>
<td>DESCO-GLIP</td>
<td>45.8</td>
<td>85.3 95.8 97.3</td>
</tr>
<tr>
<td>FIBER-B</td>
<td>49.3</td>
<td>87.1 96.1 97.4</td>
</tr>
<tr>
<td>DESCO-FIBER</td>
<td>48.9</td>
<td>86.9 96.4 98.0</td>
</tr>
</tbody>
</table>

Table 6: DESCO-GLIP and DESCO-FIBER maintain similar performance to GLIP and FIBER on common object detection and phrase grounding. DESCO-GLIP and DESCO-FIBER are fine-tuned with a smaller batch size than GLIP and FIBER; thus some minor performance drops are expected.

In Table 6, we report the performance on common object detection (COCO, zero-shot) and Flickr30K. The performance of DESCO-GLIP and DESCO-FIBER is similar to GLIP and FIBER. GLIP-T* is the checkpoint that achieves the best performance on LVIS released in https://github.com/microsoft/GLIP; it has slightly worse performance than GLIP-T on COCO. We used GLIP-T* to initialize DESCO-GLIP, thus we compare with GLIP-T* in this case.

### B.3 Evaluation on LVIS

<table>
<thead>
<tr>
<th>Model</th>
<th>LVIS Val APr</th>
<th>APr</th>
<th>APC</th>
<th>APf</th>
<th>AP</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLIP</td>
<td>10.1</td>
<td>12.5</td>
<td>25.5</td>
<td>17.2</td>
<td></td>
</tr>
<tr>
<td>DESCO-GLIP</td>
<td>19.6</td>
<td>22.0</td>
<td>33.6</td>
<td>26.2</td>
<td></td>
</tr>
<tr>
<td>FIBER</td>
<td>18.0</td>
<td>21.6</td>
<td>35.0</td>
<td>26.3</td>
<td></td>
</tr>
<tr>
<td>DESCO-FIBER</td>
<td>23.0</td>
<td>26.3</td>
<td>38.5</td>
<td>30.5</td>
<td></td>
</tr>
</tbody>
</table>

Table 7: DESCO-GLIP and DESCO-FIBER achieve strong performance on LVIS val 1.0.

We also evaluate models on the full validation set of LVIS using the fixed AP protocol. It can be seen that our approach achieves large performance gains compared to the baselines.

### B.4 OmniLabel Challenge 2023 winning entry

We submit DESCO-FIBER to the OmniLabel Challenge 2023 and won the 1st place. Different from the model reported in the main text, the submitted version is first initialized from DESCO-FIBER, then undergoes a second-stage pre-training on Objects365, GoldG, CC3M, and RefCOCO [46] (including RefCOCO, RefCOCO+, RefCOCOg), with a batch size of 8 for another 200K steps.