Descriptive Attributes for Language-based Object Keypoint Detection

Jerod Weinman[1], Serge Belongie[2], and Stella Frank[2]

[1] Grinnell College, Grinnell, Iowa, USA
jerod@acm.org
[2] Department of Computer Science, University of Copenhagen, Denmark

Abstract. Multimodal vision and language (VL) models have recently shown strong performance in phrase grounding and object detection for both zero-shot and finetuned cases. We adapt a VL model (GLIP) for keypoint detection and evaluate on NABirds keypoints. Our language-based keypoints-as-objects detector GLIP-KP outperforms baseline top-down keypoint detection models based on heatmaps and allows for zero- and few-shot evaluation. When fully trained, enhancing the keypoint names with descriptive attributes gives a significant performance boost, raising AP by as much as 6.0, compared to models without attribute information. Our model exceeds heatmap-based HRNet's AP by 4.4 overall and 8.4 on keypoints with attributes. With limited data, attributes raise zero-/one-/few-shot test AP by 1.0/3.4/1.6, respectively, on keypoints with attributes.

Keywords: Keypoint detection · Vision & language models · Attributes.

1 Introduction

Many computer vision tasks involve finding and identifying things in images, such as objects and their constituent parts. In these settings, the use of multimodal vision and language (VL) models can extend the scope of computer vision models, by replacing a limited set of object classes or part IDs with the ways that people refer to these objects in natural language. These multimodal models support open-world object detection by learning rich representations of the co-occurrences and interactions between words and pixels.

The VL model GLIP [19] was trained for phrase grounding and open-vocabulary object detection. In this paper, we extend GLIP's capabilities to object keypoint detection, *i.e.*, locating the *parts* of particular objects. For example, given an image of a bird and a textual caption containing English part names (*e.g.*, "bill, nape, tail, ..."), our GLIP-KP model detects "objects" corresponding to the parts by grounding the phrases in the query caption to specific regions or points in the image (see Figure 1). The advantage of this approach is that it allows us to use not only the part names but also additional adjectival attributes of the part: the model can detect the 'yellow and black crown' of the goldfinch, or the 'long red bill' of an oyster-catcher.

Appears in *Computer Vision Systems (ICVS 2023)*. Dominik Bauer, Jean-Baptiste Weibel, and Ehsan Yaghoubi (Eds.). Lecture Notes in Computer Science, vol 14253. Springer, Cham. 2023. https://doi.org/10.1007/978-3-031-44137-0_37

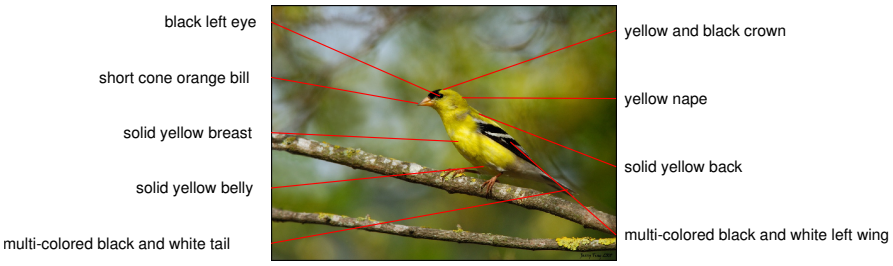

Fig. 1. Example of detecting keypoints with richer descriptions. Attributes are drawn from CUB [35] images and applied to corresponding keypoints of NABird [33] species.

This work explores the following research questions:

RQ1 Architecture: How well does a VL model for object detection perform at keypoint detection?

RQ2 Pretraining: What is the advantage of using the pretrained VL model, along with its image and language encoders?

RQ3 Attributes: How does adding richer descriptive attributes to keypoint names influence detection performance?

We assess these questions in both a small data/few-shot setting, where the multimodal pretraining enables identification of some keypoints, and in a fully finetuned keypoint detection setting. Our system design (Section 3) and the supporting experiments (Section 5) on a bird keypoint detection task (Section 4) demonstrate three primary findings. First, a multimodal vision and language (VL) model designed for object detection offers a compelling, competitive keypoint detector (RQ1). In particular, our model outperforms several top-down pose estimation baselines on the bird keypoint detection task. Second, pretraining the VL model benefits both test performance and train time (RQ2); finetuning pretrained models provides a modest AP improvement, while training VL layers from scratch takes 50% more epochs. Finally, leveraging keypoint descriptions enhances VL model test performance (RQ3), whether in data-limited (zero-/one-/few-shot) or fully finetuned models.

2 Related Work

In this section we situate our work in the context of recent trends in detecting keypoints, leveraging language for object detection and phrase grounding, and using descriptive attributes for transfer to novel prediction tasks.

2.1 Keypoint Detection

Keypoint detection for pose estimation is a core computer vision task commonly performed on human data [21, 1, 18], but it is also applicable to non-human animals [41, 39], including birds [35, 33]. Architectures for keypoint detection typically distinguish between two-stage and single-stage models. Two-stage models

are top-down, in that first the body is found using an object detector, and then the keypoints are identified [36, 32, 20]. Single-stage models process bottom-up, with all visible keypoints (*e.g.*, in a crowd) found at once [28, 10]. Earlier models depend on convolutions both for the image processing backbone and further processing [36, 32]. More recent models factor out the visual backbone, which can either use a convolutional or visual transformer architecture to provide features for the main model [12, 20, 38, 42, 37, 24].

To predict keypoint locations, many model heads typically use heat-map estimation [36]. More recent works have used direct regression [24] or treat keypoints the same as objects (*i.e.*, as the centers of detected bounding boxes) [25]. Our work extends the latter approach, treating keypoint detection like object detection. Whereas KAPAO [25] trains a keypoint/object detection network from scratch using an architecture based on the YOLO [30] object detector, our work is based on a *pretrained* multimodal VL model, which allows us to easily incorporate few-shot learning and richer keypoint descriptions.

2.2 Open-Vocabulary Object Detection

Open-vocabulary object detection combines language models with object detection to detect objects beyond a limited set of training object classes. Approaches typically leverage pretrained models with vision and language aligned at the image level (*e.g.*, CLIP [29] or ALIGN [13]); a difficulty then lies in how to extract region information, such as bounding boxes, from these image-level models. Early work used VL models to classify regions found by a class-agnostic region-proposal model ViLD [11]. OWL-VIT adds visual-token level classification within a VL encoder to fine-tune these models for object detection with class predictions from the language encoder [26]. Kim *et al.* [15] perform additional pretraining at the region level (using crops), resulting in a VL model with region understanding. Kuo *et al.* [16] train only a detector head on top of a large-scale frozen VL model.

2.3 Phrase-Region Grounding

While open-vocabulary object detection aims to find the correct label for novel objects, text grounding models find the most relevant region in an image for a given phrase. mDETR [14] extends the detection transformer DETR [3] with alignment to captions, going beyond object classes to the free text found in natural language captions. GLIP [19, 43] continues this line of work, based on the functional equivalence between phrase grounding and object detection. GLIP excelled at these tasks by fusing the vision and language encodings more deeply than mDETR, specifically using a dynamic head [6]. FIBER [8] improves further by fusing the vision and language streams in the backbones. In an orthogonal approach, PEVL [40] interleaves language tokens and positional tokens (bounding box coordinates) in the text, turning text-region grounding into a masked language-modelling task. As noted above, this work applies the GLIP framework to keypoint detection, with keypoint labels (and attributes) being the caption of the detected keypoint "object".

2.4 Descriptive Attributes

Declarative attributes have long been used for few-shot transfer learning to novel categories [17]. Zero-shot performance of pretrained models depends on several factors [34, 4]. Vogel *et al.* [34] find that whether attributes aid instance-classification depends on the VL model and the specific way attributes are incorporated. To reduce annotator burden in specialized domains, Mall *et al.* [23] create a method that interactively requests informative attributes for new classification tasks using pretrained models. Forbes *et al.* [9] created birds-to-words, which describes the differences between species in support of a model architecture generating such natural language comparisons.

Although such attributes or descriptions naturally include keypoints among the noun phrases, none of these models localize them. To our knowledge, this work is the first to leverage descriptive attributes for keypoint detection, rather than whole-image classification or object/region grounding.

3 Keypoint Detection System

Our system is based on GLIP [19], which fuses the vision and language representations within the region proposal network. In this section we briefly review that model and then describe how it is adapted to GLIP-KP for keypoint detection.

3.1 GLIP Model

The original GLIP (Guided Language Image Pretraining) model [19] combines a visual backbone with a language backbone, fusing these encodings in an interlinking multilayer head [6]. The model is trained with a mixture of two losses. Classification loss—a binary focal sigmoid loss—rewards agreement between the fused visual regions and language phrases that are matched in the ground truth. Localization loss rewards bounding box proposals that overlap with *and* are centered closer to the ground truth box.

We base our GLIP-KP experiments on the GLIP-L model, which was pretrained on 27M image-text pairs and uses SWIN-L [22] and BERT-base-uncased [7] as the vision and language backbones, respectively.

3.2 Vision and Language Keypoint Detection

We adapt the GLIP model for keypoint detection by treating each keypoint as an object to be detected, which is similar to what KAPAO [25] does for YOLO [30]. Annotated keypoints are typically only pixel coordinates, but object detectors like GLIP are trained to produce bounding boxes. To address the mismatch, we center a small bounding box around each keypoint. Although our keypoint-detecting GLIP-KP is trained to reproduce these small boxes, only the box center is used for evaluating predictions (see Section 5.2). Unlike KAPAO and other keypoint detection models, GLIP-KP uses *language* as the query modality,

rather than symbolic class IDs. Thus "red nape. black left wing." can be used as the query caption to locate these two keypoints (cf. Figure 1).

We arbitrarily chose a fixed 40×40 box size for all keypoints in all experiments. This setting roughly aligns with the sizes found to perform well for KAPAO on human pose estimation [25]. As a feature of the training data (not the model), varying keypoint box size can trivially be incorporated. Although our box size is presently fixed, keypoint bounding boxes could vary with the object size or the precision of a particular keypoint's location. Our evaluation measure (Section 5.2) controls for both object size and keypoint annotator precision.

4 Data

The North American Birds (NABirds) data set [33] is our primary training and testing data, due to its higher data quality and larger size compared to the commonly used Caltech-UCSD Birds 200 (CUB-200) [35]. NABirds consists of nearly 50K bird images from 400 different species, organized in a hierarchy. Our model does not directly use species information for keypoint detection, but attributes are applied at a species level (more below). The images in the NABirds data set are annotated with 11 body part keypoints (see Table 1). We use the official test split (24,633 images, each with a single bird), and take 10% of the official training split for validation (2,410 images in validation, 21,519 in training); the validation set roughly preserves class frequency while ensuring at least one example of each leaf class from the hierarchy.

Descriptive Keypoint Attributes The CUB data set includes fine-grained attributes, annotated at the image level. Most of these attributes correspond to keypoints, for example, the bill shape or wing color of a bird. We obtain species-level attributes by retaining attribute values where the majority of images (over half) for that species are annotated with the attribute value. For most species and attributes, there is only one value per attribute, with the exception of some color attributes (*e.g.*, a magpie has a tail color that is both black and white). In order to match CUB species names with NABird species label, we use a manually-obtained many-to-one mapping. This process results in 341 of 555 NABird classes being linked to attributes; roughly 32% of keypoints are assigned enhanced descriptions through these attributes. This rate is fairly consistent across the eleven bird keypoints (mean 35.8%, std. err. 0.86%).

To convert image attribute values to keypoint descriptions for GLIP-KP captions, we apply a set of simple textual transformations. The relevant image attributes (`has_bill_length`, `has_back_color`) are matched to their corresponding keypoints (bill, back). The color attribute values are preserved, and comparative attributes are minimally edited (*e.g.*, the bill length attribute is changed from `shorter`/`longer_than_head` to short/long, respectively). We arrange multiple attributes for a single keypoint according to standard English adjectival order [2]: length, pattern, shape, color, in order to match language model expectations. Table 2 gives examples of the resulting descriptive captions.

Table 1. Part keypoints with their hierarchically ordered attributes and annotator standard deviations ×100, relative to the bird bounding box dimensions.

Keypoint Label	σ_X	σ_Y	Attributes
bill	1.998	1.340	length / shape / color
crown	3.255	2.065	color
nape	2.956	3.479	color
left/right eye	1.068	0.927	color
belly	4.810	4.537	pattern / color
breast	3.132	4.340	pattern / color
back	3.934	3.372	pattern / color
tail	4.335	3.972	pattern / shape / (under, upper) colors
left/right wing	6.017	5.097	pattern / shape / color

Table 2. Examples of descriptive attributes for each keypoint (first instance in **bold**).

short all-purpose grey **bill**. long needle black bill. dagger orange and black bill. spatulate yellow bill.
black and white **crown**. blue crown. brown and buff crown. yellow and black crown. red crown.
black and white **nape**. brown and white nape. brown and black nape. buff nape. blue nape.
black and red **left eye**. black **right eye**. red left eye. yellow right eye.
striped brown and white **belly**. multi-colored yellow belly. solid belly. spotted brown and white belly.
multi-colored yellow and black **breast**. striped yellow and black breast. white breast.
olive **back**. striped brown black and buff back. solid blue back. multi-colored grey and yellow back.
black and white black and white **tail**. solid notched tail. notched brown tail. white black tail.
spotted pointed black and white **left wing**. striped rounded brown black white and buff **right wing**.

5 Experiments

We perform experiments training the model on several variants of the data. In this section we describe the general training and evaluation regimes before presenting the experimental results.

GLIP-KP is trained to predict keypoint box locations given a caption containing the keypoint labels. We train with two initial conditions for model weights and three data variants:

Symbols To isolate and impede the semantic contribution of the model's language stream, each keypoint label is replaced by a single character, forcing the deep fusion layers to rely on the visual characteristics and a symbol with little intrinsic meaning. We omit characters that might suggest a description of visual shape (*i.e.*, x, s, or t). Caption: "f. g. h. k. m. n. p. q. r. w. y."

Labels The basic keypoint labels are used in the text prompt encoded by the language model. Caption: "bill. crown. nape. left eye. right eye. belly. breast. back. tail. left wing. right wing."

Labels+Attributes Available descriptive attributes precede the keypoint label. Caption: "short cone buff bill. grey crown. grey nape. ..." (see Table 2).

In a "Finetune" initial condition, we begin training using the GLIP-L pretrained weights. For "Scratch", the deep fusion layers are randomly initialized

while the language backbone uses the original BERT-base-uncased model pretrained weights and the vision backbone uses the SWIN-L pretrained weights.

To benchmark our overall approach to keypoint detection (*i.e.*, vision and language with attributes) we also use the MMPose library [27] to train and test three methods that predict keypoints via heatmaps: HRNet [32] and two variants of the Simple Baseline model [36] with ResNet-50 [12] and SWIN-L [22] vision backbones.

5.1 Training Details

The validation metric is mAP (Max Dets = 100); see the next section (5.2) for details. The base learning rate begins at 1e–5 and is adjusted by a factor of 0.5 after four epochs with no validation set improvement; we establish a minimum learning rate of 1e–7. Training is stopped altogether after 10 epochs without improvement. Apart from one-shot learning, all models are trained with a batch size of three images (the maximum afforded by our compute setup). One epoch on the full training set takes four hours using one NVIDIA RTX A6000 GPU.

All the keypoints are included together in a single query caption that is to be encoded by the language model. This allows the complete context of the query (including attributes, if available) to influence the encoding. Importantly, we randomly shuffle the keypoints within the captions for each training batch. The model should thus be forced to learn to use the language embedding, rather than simply relying on a positional encoding of the tokens to represent the part.

Following earlier findings that unfreezing visual backbones during finetuning can improve performance [31], the SWIN-L model visual backbone is frozen at layer 2 (subsequent layers are trainable), and the BERT-base-uncased language backbone is not frozen (all layers are updated).

To randomly augment the training data, images may undergo a horizontal flip (with the necessary alteration to the sagittal wing and eye keypoint labels) or may be resized to between 640 and 1024 pixels (preserving aspect ratio).

For the baseline comparison models, we retain the default parameters: input *object* size (cropped bird) is 256×256 and heatmap size is 64×64. All models train for 210 epochs and the best checkpoint is selected using the validation set.

5.2 Evaluation

Because our system is a hybrid of object and keypoint detection, we hybridize object and keypoint detection evaluation measures for our evaluation. Compared to traditional keypoint detection, there are two main differences in our setup: 1) not all keypoints must be predicted on the image, because the model can decide not to ground a keypoint caption label; 2) multiple detections can be predicted for each label, not just the single best detection. There is also the superficial difference between predicting bounding boxes versus pixel coordinates; we simply take the center of each box as the predicted keypoint location.

Traditional keypoint evaluations are limited because they only measure performance on keypoints annotated as visible. Thus, no false positives result from

predicting keypoints that are absent from an object; false positives only occur if a keypoint is matched to the wrong parent object. With only a single instance of the object in each NABirds image (and many other pose estimation data sets), these types of false positives are not possible. This omission makes precision metrics relatively meaningless in the traditional keypoint evaluation scheme.

By definition, recall likewise ignores false positives. When considering only the best-scoring detection for each keypoint, recall is therefore equivalent to accuracy (percentage of correct keypoints, or PCK), another traditional metric.

Ultimately, the traditional evaluation schemes can be misleading when a system is to be applied to images where the keypoint visibility is unknown and false positives are to be minimized (as in cases intended for human training).

Thus, to evaluate keypoint detections, we combine elements of the COCO evaluation frameworks for both keypoints and object detection. Specifically, we replace the COCO object detection intersection-over-union (IoU) calculation for matching bounding boxes with a variant of the COCO keypoint detection challenge's object-keypoint-similarity (OKS) calculation. The COCO OKS utilizes the square root of the object area as a "scale" factor so it can account for size-normalized annotator variation in keypoint locations. Rather than collapse height and width, we retain these scales separately, noting that the annotator variation along the x and y axes is anisotropic (see Table 1).

To classify a detection as correct or incorrect, we must first measure the relative detection difference $\mathbf{d} = \mathbf{p} - \mathbf{t}$ between the prediction $\mathbf{p}$ and the ground truth point $\mathbf{t}$. The "error" $\mathbf{e}$ in this difference is normalized by a factor incorporating both the object size (w, h) and the annotator variation (σ_X, σ_Y),

$$\mathbf{e} = \left[\frac{d_x}{k_x w}, \frac{d_y}{k_y h} \right], \tag{1}$$

where $k = 2\sigma$ is the per-dimension scaling factor. Passing this error through an unnormalized Gaussian,

$$s = \exp\left(-\frac{1}{2} \|\mathbf{e}\|^2 \right), \tag{2}$$

produces an interpretable object keypoint similarity (OKS) with $s \in (0, 1]$, like the IoU. This setup "means that 68%, 95%, and 99.7% of human annotated keypoints should have a keypoint similarity of .88, .61, or .32 or higher, respectively" [5], according to the COCO keypoint evaluation description. In summary, we use the traditional COCO evaluation scheme for objects, replacing the IoU criterion with our OKS (2).

We report a subset of the COCO evaluation metrics for all object (bird) sizes:

mAP Mean average precision with OKS=0.50:0.05:0.95
AP$^{0.50}$ Average precision at OKS=0.50 (a "loose" metric)
AP$^{0.75}$ Average precision at OKS=0.75 (a "strict" metric)
AR Average recall with OKS=0.50:0.05:0.95

Except when given for individual keypoint labels (*i.e.*, Figure 3), these metrics are reported as averages over all the keypoint labels.

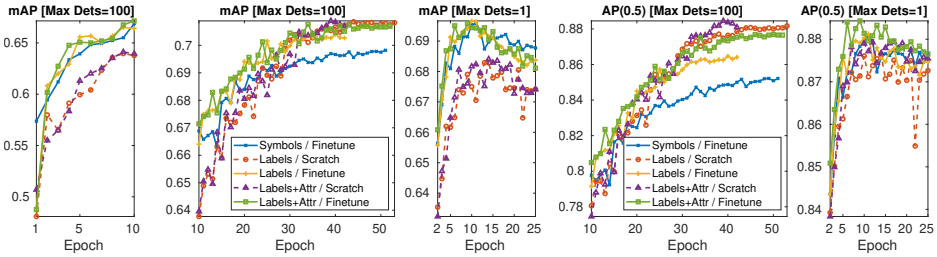

Fig. 2. Training metrics versus train time on validation data. See discussion below.

While we report AR for some comparability with keypoint detectors designed to optimize PCK, we remain more interested in the AP metrics as sensitive to both recall (the true positive rate), and precision (true positives among *predicted* positives). To capture the overall fidelity of the model, we treat the keypoints as if they were independent objects and evaluate accordingly.

5.3 Results and Discussion

In this section we present the results of training our GLIP-KP model on the three data variants (Symbols, Labels, Labels+Attributes) with the two initial conditions (Finetune and Scratch).

To gauge the transferability of VL pretraining, we evaluate the "Finetune" models under limited training conditions. From the results in Table 3, we see that in the zero-, one-, and few-shot cases, using labels greatly outperforms the non-semantic "Symbols" variant. In most instances, adding attributes provides a performance boost, particularly among the keypoints that have them. However, "Symbols" surprisingly outperforms all others by a fairly wide margin at one epoch of training on all the data. We hypothesize that at this early stage of training, the symbolic model can focus more directly on the keypoint detection task, while the models using language might be hampered by ambiguous semantics in the keypoint names (*e.g.*, 'crown', 'left wing').

This performance boost evaporates by the second epoch, as can be seen in Figure 2, which illustrates some of the evaluation metrics on the validation data as training progresses. Aside from epoch 1 (Fig. 2(a)), the "Symbols" model lags by a wide margin, particularly in the 'broad' case (Max Dets=100—Fig. 2(b,d)), when multiple keypoint detections are considered for each label; this demonstrates the ultimate benefit of involving language in keypoint detection. Models trained from scratch lag behind initially in the 'broad' case, but they do eventually catch up to the finetuned models in later epochs (Fig. 2(b)) and actually excel on the 'loose' metric (Fig. 2(d)). By contrast, finetuned models outperform in the 'narrow' case (Max Dets=1—Fig. 2(c,e)), which evaluates a single prediction per keypoint. Notably, the narrow metrics peak early before overfitting sets

Table 3. Held-out test data results ("Finetune" initial condition) for zero-, one-, and few-shot training as well as one full epoch of all the training data. One-shot (1 example of each keypoint) and few-shot (10 examples of each keypoint) are both trained for 500 iterations over the limited data, reporting averages and standard errors of ten runs.

All Data

		Broad: Max Dets = 100			Narrow: Max Dets = 1			
	Data	mAP	AP(0.5)	AP(0.75)	mAP	AP(0.5)	AP(0.75)	AR
Zero	Symbols	0.20	0.52	0.13	0.13	0.34	0.08	1.95
	Labels	9.46	16.30	9.03	7.36	12.36	7.11	22.70
	Labels+Attr.	**9.74**	**16.63**	**9.33**	**7.58**	**12.83**	**7.29**	**23.64**
One	Symbols	21.40 ±0.38	31.73 ±0.51	21.92 ±0.40	17.79 ±0.36	26.85 ±0.51	18.06 ±0.37	35.65 ±0.45
	Labels	28.97 ±0.18	**43.57** ±0.23	29.68 ±0.20	25.04 ±0.18	39.33 ±0.25	25.20 ±0.20	47.32 ±0.24
	Labels+Attr.	28.74 ±0.22	43.08 ±0.14	29.50 ±0.27	**25.62** ±0.23	**40.17** ±0.15	**25.82** ±0.29	**49.38** ±0.20
Few	Symbols	26.23 ±0.33	36.81 ±0.37	27.35 ±0.36	23.95 ±0.34	36.10 ±0.43	24.42 ±0.36	44.07 ±0.39
	Labels	35.46 ±0.09	48.34 ±0.17	37.19 ±0.10	**33.84** ±0.08	**49.86** ±0.11	**34.81** ±0.10	**57.88** ±0.05
	Labels+Attr.	35.43 ±0.13	**49.83** ±0.15	36.96 ±0.15	33.07 ±0.12	49.51 ±0.13	33.93 ±0.14	57.17 ±0.10
Epoch	Symbols	**56.70**	**68.45**	**60.47**	**62.74**	**81.57**	**66.93**	**74.50**
	Labels	48.10	58.36	51.33	51.41	67.76	54.80	73.17
	Labels+Attr.	48.14	58.42	51.35	51.81	68.10	55.23	73.40

Only Data with Attributes

Zero	Symbols	0.10	0.25	0.05	0.07	0.17	0.03	1.93
	Labels	4.95	8.44	4.85	4.14	6.88	4.13	22.89
	Labels+Attr.	**6.09**	**9.58**	**6.11**	**5.09**	**8.08**	**5.10**	**25.56**
One	Symbols	14.54 ±0.30	22.50 ±0.44	14.77 ±0.32	12.99 ±0.29	20.32 ±0.44	13.11 ±0.30	37.75 ±0.47
	Labels	19.05 ±0.17	30.80 ±0.24	19.02 ±0.18	17.24 ±0.17	28.70 ±0.25	16.97 ±0.18	49.23 ±0.26
	Labels+Attr.	**21.76** ±0.20	**33.61** ±0.28	**22.15** ±0.21	**20.59** ±0.20	**32.41** ±0.29	**20.81** ±0.21	**52.43** ±0.11
Few	Symbols	18.05 ±0.27	27.36 ±0.34	18.44 ±0.30	17.02 ±0.28	27.24 ±0.39	17.01 ±0.30	46.21 ±0.39
	Labels	23.89 ±0.13	35.90 ±0.23	24.43 ±0.13	23.24 ±0.11	37.10 ±0.20	23.24 ±0.12	**60.09** ±0.07
	Labels+Attr.	**26.22** ±0.22	**40.31** ±0.30	**26.70** ±0.24	**24.88** ±0.22	**39.85** ±0.30	**24.95** ±0.24	58.65 ±0.11
Epoch	Symbols	**48.56**	**64.76**	**51.94**	**53.74**	**75.77**	**57.00**	**75.84**
	Labels	37.37	50.28	39.76	40.33	57.85	42.48	74.78
	Labels+Attr.	39.60	52.23	42.30	42.70	60.07	45.21	75.36

in. However, additional training helps in the broad cases, where the focal loss likely shifts to improving all predictions.

Table 4 shows the test results when training proceeds to completion on all the data. In the narrow case, the attributes provide a modest performance boost in the data overall, but it is more pronounced among the data with attributes. Attributes raise the mAP by about 6 in the broad case and 2.7 in the narrow case. Considering that AR is fairly comparable across most models, it seems the descriptive attributes largely help eliminate false positives and/or improve the prediction confidences.

The HRNet and SimpleBaseline results lag far behind our GLIP-KP model as well, regardless of visual encoder. We hypothesize this may have to do with the use of a heatmap head for prediction, which offers limited location precision. (KAPAO outperforms the SimpleBaseline by a similar degree on human pose estimation.) Since our GLIP-KP uses the SWIN-L visual backbone, we can reasonably compare it to the SimpleBaseline equipped with SWIN-L. Using the "Symbols" data eliminates the model's semantic associations, yet we still find

Table 4. Results for fully-trained models on held-out test data.

		All Data								
		Broad: Max Dets = 100				Narrow: Max Dets = 1				
Data	VL Weights	Epch	mAP	AP(0.5)	AP(0.75)	Epch	mAP	AP(0.5)	AP(0.75)	AR
Symbols	Finetune	51	69.02	84.90	73.71	11	68.79	87.46	73.54	79.70
Labels	Scratch	44	**70.07**	87.63	**74.94**	14	67.64	87.16	72.38	79.21
Labels	Finetune	40	69.57	86.12	74.33	10	68.88	87.51	73.65	**79.92**
Labels+Attr.	Scratch	39	69.95	**87.77**	74.74	14	67.85	87.42	72.64	79.06
Labels+Attr.	Finetune	34	69.75	86.92	74.43	9	**68.92**	**87.69**	**73.80**	79.67
ResNet50 + Heatmap	SimpleBaseline [36]					40	63.58	84.78	68.37	75.67
SWIN-L + Heatmap	SimpleBaseline [36]					180	62.95	84.51	67.52	75.26
HRNet [32]	SimpleBaseline [36]					130	64.53	84.84	69.63	77.71
		Only Data with Attributes								
Symbols	Finetune	51	57.82	78.69	61.74	11	59.31	81.90	63.38	80.63
Labels	Scratch	44	57.94	80.70	61.65	14	57.40	80.80	61.03	80.03
Labels	Finetune	40	57.87	79.44	61.62	10	59.03	81.66	63.03	**80.72**
Labels+Attr.	Scratch	39	**63.99**	**86.12**	**68.48**	14	60.10	83.30	64.19	79.87
Labels+Attr.	Finetune	34	63.87	85.65	68.30	9	**61.76**	**84.21**	**66.14**	80.70
ResNet50 + Heatmap	SimpleBaseline [36]					40	53.59	77.60	57.09	76.85
SWIN-L + Heatmap	SimpleBaseline [36]					180	52.30	77.07	55.37	76.42
HRNet [32]	SimpleBaseline [36]					130	53.35	76.04	57.18	78.78

GLIP-KP exceeds the mAP of the SimpleBaseline by 7.0 in this case. This boost confirms the value of formulating keypoint detection as object detection

Tying Table 4's "Epoch" columns to what Figure 2 shows, finetuning the VL weights requires fewer epochs than training them from scratch, and training with attributes is generally faster than without (for the broad evaluation). The narrow case requires the least time, with validation performance peaking early.

Figure 3 highlights the relative difficulty of the various keypoints and the relative benefits of adding attributes as the amount of training data varies. Higher-performing eyes and bill are well-localized (at the center and tip, respectively) compared to other labels, bearing out lower deviations in Table 1. With a fully-trained model ("All" data), attributes benefit every keypoint label. The crown label benefits significantly from attributes in the one- and few-shot learning cases, perhaps because attributes help resolve semantic ambiguity.

6 Conclusion

Vision and language models pretrained for object detection or phrase grounding perform well in a wide-variety of scenarios. This work demonstrates that such models can be adapted for keypoint detection as well. We treat the keypoint detection task as a special case of object detection, albeit with small, fixed-size bounding boxes as the prediction targets. Using GLIP-L as the base model, we demonstrate compelling keypoint detection performance, outperforming comparison models even when the language capabilities are hobbled. Moreover, its

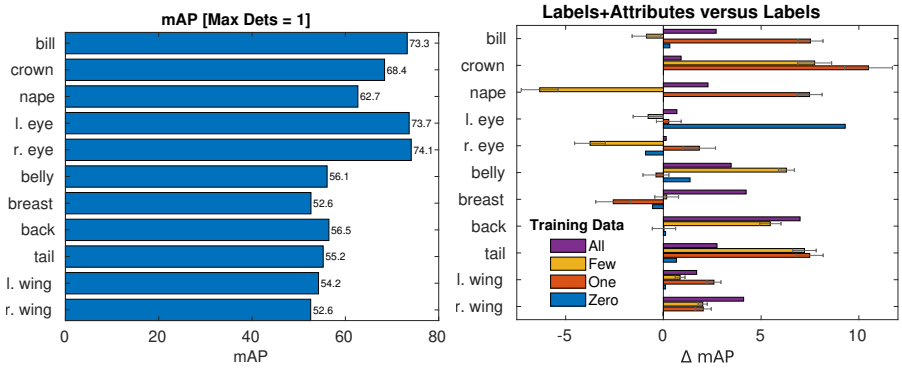

Fig. 3. Per-keypoint test performance on the Finetuned Labels+Attributes model. Absolute mAP on the fully trained model (left) and comparative performance of the Labels+Attributes versus Labels (right) for various training conditions.

language flexibility ultimately allows us to further improve results by incorporating English-language descriptive attributes of the target keypoints.

We also propose an evaluation methodology better-suited to keypoints-as-objects, replacing IoU with a variant of object keypoint similarity (OKS) in the standard COCO object detection framework. We hope the familiarity of the COCO framework and the benefits of considering false positives in keypoint detection will lead to wider adoption of such metrics.

In the future, we hope to investigate not only joint object detection (*i.e.*, the bird bounding box) in the model, but also fine-grained species classification. In addition to the descriptive keypoints, the language-based model may be able to share representations (and better distinguish) among species with similar names and appearances, such as the red-shouldered hawk and the red-tailed hawk.

Acknowledgments We thank Grant Van Horn for the mapping between NABirds and CUB, Jonathan M. Wells for helpful conversation, Vésteinn Snæbjarnarson for experimental assistance, and the reviewers for important feedback. This work was supported in part by the Pioneer Centre for AI, DNRF grant number P1.

References

1. Andriluka, M., Pishchulin, L., Gehler, P., Schiele, B.: 2D human pose estimation: New benchmark and state of the art analysis. In: Proc. CVPR (2014)
2. Cambridge University Press: Adjectives: order (Cambridge Grammar) (2022), https://dictionary.cambridge.org/us/grammar/british-grammar/adjectives-order
3. Carion, N., Massa, F., Synnaeve, G., Usunier, N., Kirillov, A., Zagoruyko, S.: End-to-end object detection with transformers. In: Proc. ECCV. pp. 213–229 (2020)
4. Cascante-Bonilla, P., Karlinsky, L., Smith, J.S., Qi, Y., Ordonez, V.: On the transferability of visual features in generalized zero-shot learning (2022). https://doi.org/10.48550/arXiv.2211.12494

5. COCO - Common Objects in Context: Keypoint evaluation (2017), https://cocodataset.org/#keypoints-eval
6. Dai, X., Chen, Y., Xiao, B., Chen, D., Liu, M., Yuan, L., Zhang, L.: Dynamic head: Unifying object detection heads with attentions. In: Proc. CVPR. pp. 7373–7382 (2021)
7. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT (1). pp. 4171–4186 (2019)
8. Dou, Z.Y., Kamath, A., Gan, Z., Zhang, P., Wang, J., Li, L., Liu, Z., Liu, C., LeCun, Y., Peng, N., Gao, J., Wang, L.: Coarse-to-fine vision-language pre-training with fusion in the backbone. In: Advances in NeurIPS (2022)
9. Forbes, M., Kaeser-Chen, C., Sharma, P., Belongie, S.: Neural naturalist: Generating fine-grained image comparisons. In: Proc. EMNLP-IJCNLP. pp. 708–717 (2019)
10. Geng, Z., Sun, K., Xiao, B., Zhang, Z., Wang, J.: Bottom-up human pose estimation via disentangled keypoint regression (2021). https://doi.org/10.48550/arXiv.2104.02300
11. Gu, X., Lin, T.Y., Kuo, W., Cui, Y.: Open-vocabulary object detection via vision and language knowledge distillation (2022). https://doi.org/10.48550/arXiv.2104.13921
12. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: Proc. CVPR (2016)
13. Jia, C., Yang, Y., Xia, Y., Chen, Y.T., Parekh, Z., Pham, H., Le, Q., Sung, Y.H., Li, Z., Duerig, T.: Scaling up visual and vision-language representation learning with noisy text supervision. In: Proc. ICML. pp. 4904–4916 (2021)
14. Kamath, A., Singh, M., LeCun, Y., Misra, I., Synnaeve, G., Carion, N.: MDETR - modulated detection for end-to-end multi-modal understanding. Proc. ICCV pp. 1760–1770 (2021)
15. Kim, D., Angelova, A., Kuo, W.: Region-aware pretraining for open-vocabulary object detection with vision transformers. In: Proc. CVPR. pp. 11144–11154 (2023)
16. Kuo, W., Cui, Y., Gu, X., Piergiovanni, A.J., Angelova, A.: Open-vocabulary object detection upon frozen vision and language models. In: Proc. CoLT (2023)
17. Larochelle, H., Erhan, D., Bengio, Y.: Zero-data learning of new tasks. In: AAAI (2008)
18. Li, J., Wang, C., Zhu, H., Mao, Y., Fang, H.S., Lu, C.: Crowdpose: Efficient crowded scenes pose estimation and a new benchmark. In: Proc. CVPR (2019)
19. Li, L.H., Zhang, P., Zhang, H., Yang, J., Li, C., Zhong, Y., Wang, L., Yuan, L., Zhang, L., Hwang, J.N., et al.: Grounded language-image pre-training. In: Proc. CVPR. pp. 10965–10975 (2022)
20. Li, Y., Zhang, S., Wang, Z., Yang, S., Yang, W., Xia, S.T., Zhou, E.: Tokenpose: Learning keypoint tokens for human pose estimation. In: Proc. ICCV (2021)
21. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft COCO: Common objects in context. In: Proc. ECCV. pp. 740–755 (2014)
22. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.: Swin transformer: Hierarchical vision transformer using shifted windows. In: Proc. ICCV. pp. 10012–10022 (2021)
23. Mall, U., Hariharan, B., Bala, K.: Field-guide-inspired zero-shot learning. In: Proc. ICCV. pp. 9546–9555 (2021)
24. Mao, W., Ge, Y., Shen, C., Tian, Z., Wang, X., Wang, Z., Hengel, A.v.d.: Poseur: Direct human pose regression with transformers. In: Proc. ECCV (2022)

25. McNally, W., Vats, K., Wong, A., McPhee, J.: Rethinking keypoint representations: Modeling keypoints and poses as objects for multi-person human pose estimation. In: Proc. ECCV. vol. 13666, pp. 37–54 (2022)
26. Minderer, M., Gritsenko, A., Stone, A., Neumann, M., Weissenborn, D., Dosovitskiy, A., Mahendran, A., Arnab, A., Dehghani, M., Shen, Z., et al.: Simple open-vocabulary object detection. In: Proc. ECCV. pp. 728–755 (2022)
27. MMPose Contributors: OpenMMLab pose estimation toolbox and benchmark (Aug 2020), https://github.com/open-mmlab/mmpose
28. Newell, A., Huang, Z., Deng, J.: Associative embedding: End-to-end learning for joint detection and grouping. In: Advances in NeurIPS. vol. 30 (2017)
29. Radford, A., Kim, J.W., Hallacy, C., Ramesh, A., Goh, G., Agarwal, S., Sastry, G., Askell, A., Mishkin, P., Clark, J., Krueger, G., Sutskever, I.: Learning transferable visual models from natural language supervision. In: Proc. ICML. vol. 139, pp. 8748–8763 (2021)
30. Redmon, J., Divvala, S., Girshick, R., Farhadi, A.: You only look once: Unified, real-time object detection. In: Proc. CVPR (June 2016)
31. Shen, S., Li, L.H., Tan, H., Bansal, M., Rohrbach, A., Chang, K.W., Yao, Z., Keutzer, K.: How much can CLIP benefit vision-and-language tasks? In: Proc. ICLR (2022)
32. Sun, K., Xiao, B., Liu, D., Wang, J.: Deep high-resolution representation learning for human pose estimation. In: Proc. CVPR (2019)
33. Van Horn, G., Branson, S., Farrell, R., Haber, S., Barry, J., Ipeirotis, P., Perona, P., Belongie, S.: Building a bird recognition app and large scale dataset with citizen scientists: The fine print in fine-grained dataset collection. In: Proc. CVPR. pp. 595–604 (2015)
34. Vogel, F., Shvetsova, N., Karlinsky, L., Kuehne, H.: VL-Taboo: An analysis of attribute-based zero-shot capabilities of vision-language models (2022). https://doi.org/10.48550/arXiv.2209.06103
35. Welinder, P., Branson, S., Mita, T., Wah, C., Schroff, F., Belongie, S., Perona, P.: Caltech-UCSD birds 200. Tech. rep., California Institute of Technology (2010)
36. Xiao, B., Wu, H., Wei, Y.: Simple baselines for human pose estimation and tracking. In: Proc. ECCV (2018)
37. Xu, Y., Zhang, J., Zhang, Q., Tao, D.: ViTPose: Simple vision transformer baselines for human pose estimation. In: Advances in NeurIPS (2022)
38. Yang, S., Quan, Z., Nie, M., Yang, W.: Transpose: Keypoint localization via transformer. In: Proc. ICCV. pp. 11802–11812 (2021)
39. Yang, Y., Yang, J., Xu, Y., Zhang, J., Lan, L., Tao, D.: APT-36k: A large-scale benchmark for animal pose estimation and tracking. In: Advances in NeurIPS Datasets and Benchmarks Track (2022)
40. Yao, Y., Chen, Q., Zhang, A., Ji, W., Liu, Z., Chua, T.S., Sun, M.: PEVL: Position-enhanced pre-training and prompt tuning for vision-language models. In: Proc. EMNLP. pp. 11104–11117 (Dec 2022)
41. Yu, H., Xu, Y., Zhang, J., Zhao, W., Guan, Z., Tao, D.: AP-10K: A benchmark for animal pose estimation in the wild. In: Conf. NeurIPS Datasets and Benchmarks Track (Round 2) (2021)
42. Yuan, Y., Fu, R., Huang, L., Lin, W., Zhang, C., CHEN, X., Wang, J.: HRFormer: High-resolution vision transformer for dense prediction. In: Advances in NeurIPS (2021)
43. Zhang, H., Zhang, P., Hu, X., Chen, Y.C., Li, L., Dai, X., Wang, L., Yuan, L., Hwang, J.N., Gao, J.: GLIPv2: Unifying localization and vision-language understanding. In: Advances in NeurIPS. vol. 35, pp. 36067–36080 (2022)