LUNA: Language as Continuing Anchors for Referring Expression Comprehension

Anonymous Author(s)
Affiliation
Address
email

Abstract

Referring expression comprehension aims to localize the description of a natural language expression in an image. Using location priors to remedy inaccuracies in cross-modal alignments is the state of the art for CNN-based methods tackling this problem. Recent Transformer-based models cast aside this idea making the case for steering away from hand-designed components. In this work, we propose LUNA, which uses language as continuing anchors to guide box prediction in a Transformer decoder, and show that language-guided location priors can be effectively exploited in a Transformer-based architecture. Specifically, we first initialize an anchor box from the input expression via a small "proto-decoder", and then use this anchor as location prior in a modified Transformer decoder for predicting the bounding box. Iterating through each decoder layer, the anchor box is first used as a query for pooling multi-modal context, and then updated based on pooled context. This approach allows the decoder to focus selectively on one part of the scene at a time, which reduces noise in multi-modal context and leads to more accurate box predictions. Our method outperforms existing state-of-the-art methods on the challenging datasets of ReferIt Game, RefCOCO/+/g, and Flickr30K Entities.

1 Introduction

Referring expression comprehension (REC) is the task of localizing natural language in images, where an expression in plain text describes a single or a group of objects from an image, and the objective is to put a bounding box around the target. It provides fundamental values to many real-world applications such as robotics [41], image editing [39], and surveillance [21]. With purposes of reducing the search space and mitigating difficulties in cross-modal context modeling, most early methods rely on prior knowledge about the "likely" locations of the target for guiding box prediction.

Specifically, region proposals and anchor boxes are the two most common types of location priors. Two-stage models [14, 45, 50, 52, 46] leverage region proposals, consisting of thousands of massively overlapping boxes extracted using a standalone proposal method [42, 56, 36]. A cross-modal similarity-based ranking method is used to select one proposal as the prediction. Such models cannot recover from proposal failures and generally suffer a low recall rate [48]. On the other hand, one-stage models [48, 47, 30] leverage dense anchor boxes defined over image locations and directly predict a bounding box from integrated multi-modal feature maps. This approach allows box regression to be conditioned on the input expression, thus addressing the aforementioned issue of two-stage models. The definitions of anchors however, are heuristic and greatly influence model performance.

The most recent Transformer-based methods [5, 20] discard location priors and leverage the powerful correlation modeling power of the Transformer architecture [6, 43] for end-to-end prediction. Typically, a Transformer encoder [6] jointly embeds visual and linguistic inputs, and box prediction is made from a context feature vector globally pooled from the encoder outputs based on query-to-

Submitted to 37th Conference on Neural Information Processing Systems (NeurIPS 2023). Do not distribute.

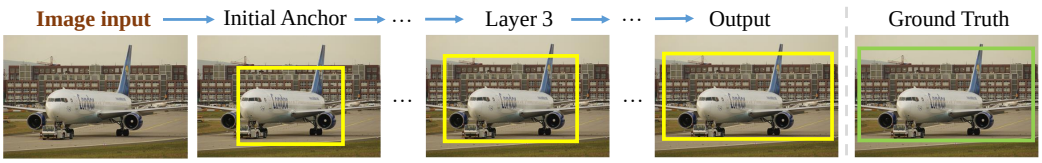


Figure 1: Example anchor boxes learned by our method. The anchor boxes are generated sequentially by a stack of modified Transformer decoder layers. The next anchor box is predicted based on context aggregated by the current anchor box. The last predicted box is used as the final prediction.

feature similarities. The query (which can be a black-box learnable feature vector [5] or a linguistic feature vector summarizing the expression [20]) serves as a representation of the target and context pooling is guided by similarity-based attention.

A potential issue with this approach is that without location priors (pre-designed or learnable), these queries rely on pure "content"-based similarities to decide context locations, which often contain numerous inaccuracies in practice. Li *et al.* [20] demonstrate that even in scenes of relatively simple compositions, the attention of the context feature vector from this approach often peaks at multiple locations, including those outside the target area, which leads to wrong or inaccurate box predictions.

Motivated by the above observations, in this work we propose a Transformer-based decoding method for addressing REC, which we term as LUNA (short for LangUage as contiNuing Anchors). It consists in leveraging the input expression for generating a series of continuously updated anchor boxes (shown in Fig. 1) that guide object localization. LUNA generates the first anchor box by attending to image regions under the guidance of the input expression. This is achieved via a cross-attention-based proto-decoder, which summarizes an object representation based on word-specific visual context and decodes an approximate location of the target. Given the initial object representation and the anchor box, a stack of modified Transformer decoder layers iteratively refine the object representation and update the anchor box. Progressing through each layer, the current anchor box is projected into high-dimensional space and used for pooling multi-modal context; then a new anchor box is predicted from pooled context. We refer to this stack of decoder layers as a continuous anchor-guided decoder. This decoding approach allows the model to focus selectively on one part of the scene at a time, and acquire more accurate context information with focused attention.

To evaluate the efficacy of the proposed method, we conduct extensive experiments on the challenging datasets of ReferIt [18], RefCOCO [51], RefCOCO+ [51], RefCOCOg [32], and Flickr30K Entities [34], for which we improve the state of the art by large margins.

Our contributions are as follows. First, we propose a Transformer-based decoding method which contains a proto-decoder and a continuous anchor-guided decoder for tackling referring expression comprehension. Second, we obtain new state-of-the-art results on five referring expression comprehension benchmarks, demonstrating the effectiveness and generality of the proposed method.

2 Related work

Referring expression comprehension (REC). Since the proposal of this task in several parallel studies [51, 16, 32], the state-of-the-art paradigm for tackling it has transitioned from *two-stage* models to *one-stage* models to the most recent *Transformer-based* models. Two-stage models [14, 45, 50, 52, 46] rank a set of image regions based on their similarities with the referring language expression, where the regions are pre-extracted using a region proposal method. Popular proposal methods include unsupervised ones based on hand-crafted features [42, 56] and pre-trained neural nets [36]. This propose-and-rank approach is slow and suffers from limitations of the proposal method (such as location inaccuracies from unsupervised methods or biases of the proposal network) [22, 48].

To address these issues, one-stage models opt for directly predicting the bounding box from fused multi-modal features, relying on feature fusion mechanisms and dense anchoring [48, 47, 30]. Typically, a cross-modal feature fusion module (such as concatenation-based [48, 47] or attention-based [30]) integrates extracted visual and linguistic features to generate multi-modal features. Then

a set of reference boxes, termed anchor boxes, are densely placed at different locations of the multi-modal feature maps, and boxes are predicted with respect to anchors in a sliding window fashion. By directly conditioning box predictions on the referring expression, one-stage models circumvent the problems faced by their two-stage precursors. However, the heuristics used for defining the anchor boxes (concerning their scales, aspect ratios, and assignments) can significantly influence the accuracy of the model. Another line of work (*e.g.*, [22]) leverages language as correlation filters and does not resort to anchors, but has not reached the accuracy of anchor-based methods.

Most recently, the powerful Transformer [6, 43] architecture is adapted to this problem. The core ingredients of Transformer-based methods [5, 20] include a Transformer encoder for jointly embedding visual and linguistic features, and a cleverly designed "query", which aggregates multi-modal context used for box prediction. In a ViT-like [7] manner, Deng *et al.* [5] leverage a special [REG] feature vector prepended to the input sequence as the query, which pools global context in the encoder's self-attention layers. In a DETR-like [3] manner, Li *et al.* [20] employ a Transformer decoder, and use a linguistic feature vector (which summarizes the input expression) as a query in the decoder. Either the [REG] token approach or the language-as-query approach relies on purely content-based similarities for context aggregation, which tend to have many inaccuracies due to noises in the image and expression. Motivated by previous one-stage models and recent Transformer-based detection models that exploit spatial representations for decoding [54, 24], in this work we leverage learnable anchor boxes as queries for helping guide multi-modal context aggregation. Our model learns more accurate cross-modal alignments compared to models in [5] and [20].

Multi-modal understanding. One major line of research in the area of multi-modal (vision-language) understanding is the large-scale pre-training of multi-modal representations transferable to a variety of downstream tasks. This problem is typically tackled by leveraging large amounts of paired image-text data (typically sourced from the Internet) and training a Transformer-based model for solving various surrogate tasks, including masked cross-modal modeling [28, 40, 53], contrastive learning [35], and modulated detection [17]. Another line of work [29, 15] takes a multi-task learning perspective and aims to develop a unified model that can address a large number of vision-language tasks at once. In addition, growing efforts have been devoted to a variety of vision-language tasks such as visual question answering [2], image captioning [44], and temporal sentence grounding [1, 10]. In this work, we focus on the task of referring expression comprehension and propose an anchor-based decoding mechanism based on the Transformer architecture.

3 Method

We are interested in the problem of box prediction from language, where the object of interest is represented in two different modalities. Therefore, our model should perform well in two aspects: aligning the representation of the object across the two modalities, and determining its precise location on the image. To achieve our first goal, we exploit the global correlation modeling power of a Transformer encoder [6] for learning fully connected pairwise correspondences between each image patch and each word (please see Sec.3.1). To achieve our second goal, we leverage a series of iteratively updated anchor boxes as location priors for guiding multi-modal context aggregation, which is achieved with a proto-decoder and a continuous anchor-guided decoder described in Sec. 3.2. We name the "proto-decoder" this way to give relevance to its functionality to output an initial representation of the object and an initial estimate of the object location. The overall pipeline of our model is schematically illustrated in Fig. 2.

3.1 Visual-linguistic encoding

We extract visual features, $\mathcal{F}_v \in \mathbb{R}^{HW \times C}$, from the input image by leveraging a vision backbone network (such as ResNet101 [11] or Swin Transformer [26]) followed by a multi-layer perceptron (MLP) for channel reduction. Here, C denotes the channel number, and H and W denote the height and width, respectively. And we extract linguistic features, $\mathcal{F}_l \in \mathbb{R}^{T \times C}$, from the input expression by leveraging a language modeling network (such as LSTM [12] or BERT [6]) followed by an MLP for channel reduction. Here, T denotes the number of words and C denotes the number of channels. The visual features $\mathcal{F}_v$ and linguistic features $\mathcal{F}_l$ are concatenated along the sequence dimension to form the input to our Transformer encoder, $\mathcal{S} = [\mathcal{F}_v; \mathcal{F}_l]$, which is a visual-linguistic sequence of feature vectors. In the rest of the paper, we use "feature vector" and "embedding" interchangeably.

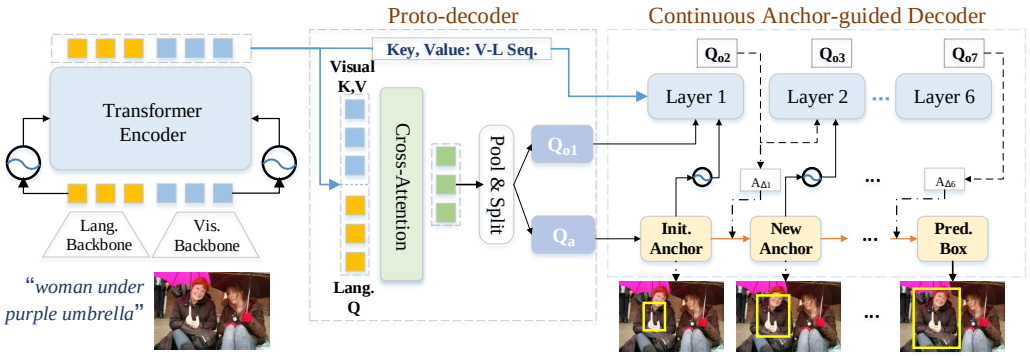


Figure 2: Overall pipeline of the proposed method. Our model consists of three main components: (1) a visual-linguistic Transformer encoder (Sec. 3.1) which jointly encodes the image and the text inputs, (2) a proto-decoder (Sec. 3.2) which produces an initial representation of the object and an initial estimate of its location (the first anchor box), and (3) a continuous anchor-guided decoder (also Sec. 3.2) which sequentially updates the anchor box through a stack of modified Transformer decoder layers. At each layer, both the output embedding from the previous layer and the anchor box are used as queries. The last anchor box corresponds to the final prediction.

The multi-modal Transformer encoder. A multi-modal Transformer encoder takes S as input, and outputs a sequence of multi-modal embeddings which align visual cues and linguistic meanings in a common feature space. We denote the encoder output as $\mathcal{E} \in \mathbb{R}^{(HW+T)\times C}$, where H, W, T, and C follow their previous definitions. Our encoder mostly follows the original implementation of the $\text{BERT}_{\text{BASE}}$ encoder [6], which consists of a stack of 6 identical layers. Each layer mainly includes a multi-head self-attention sub-layer and a feed-forward sub-layer. To encode position information and also differentiate between the two types of inputs, we use positional embeddings, denoted as $\mathcal{P} = [\mathcal{P}_v; \mathcal{P}_l]$, where $\mathcal{P}_v \in \mathbb{R}^{HW\times C}$ are fixed image positional embeddings the same as in [3] and $\mathcal{P}_l \in \mathbb{R}^{T\times C}$ are learnable expression positional embeddings [20]. $\mathcal{P}$ is supplemented to each attention sub-layer.

3.2 Decoding with continuing anchors

As described in Secs. 1 and 2, previous Transformer-based methods [5, 20] tackling this task do not exploit location priors during decoding. As illustrated in Fig. 4, this often leads to noise in the aggregated context feature vector. We use a series of "continuing" anchors (in the sense that each anchor box is updated from the previous one) as queries during decoding for addressing this problem.

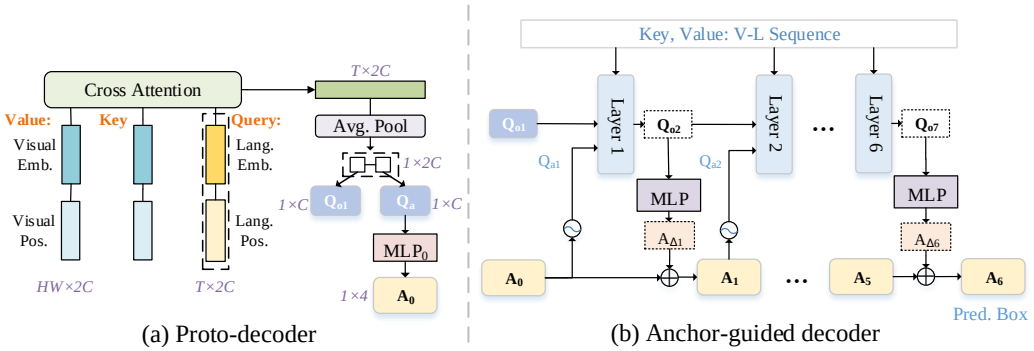
Query generation with a proto-decoder. The proto-decoder (illustrated in Fig. 3 (a)) generates the first anchor box and an object representation (what we refer to as the object query) from multi-modal context. Without introducing extra human-designed strategy, our learnable anchor could naturally perform a role of location prior.

To start with, word-specific multi-modal context is computed via a cross-attention layer. The query input to this layer is a concatenation of $\mathcal{E}_l \in \mathbb{R}^{T\times C}$ and $\mathcal{P}_l$ (the expression positional embeddings described in Sec. 3.1) along the channel dimension. Here, $\mathcal{E}_l$ is a segment of the encoder output $\mathcal{E}$ from positions corresponding to word embeddings. The key and the value inputs to this layer is a concatenation of $\mathcal{E}_v \in \mathbb{R}^{HW\times C}$ and $\mathcal{P}_v$ (the image positional embeddings described in Sec. 3.1) along the channel dimension. Similarly, $\mathcal{E}_v$ is a segment of $\mathcal{E}$ from positions corresponding to image region embeddings. The output from the cross-attention layer encodes word-specific multi-modal context and is denoted as $O \in \mathbb{R}^{T\times 2C}$. Next, weighted average pooling is applied on O across the sequence dimension:

$$W = \text{softmax}(\gamma(O)), \tag{1}$$

$$Q = W^T O, \tag{2}$$

where γ is implemented as an MLP, $W \in \mathbb{R}^{T\times 1}$ are learned averaging weights, and $Q \in \mathbb{R}^{1\times 2C}$ is the pooling output. In a final step, we split Q along the channel dimension to obtain two feature



(a) Proto-decoder (b) Anchor-guided decoder

Figure 3: Details of the proto-decoder and the continuous anchor-guided decoder. (a) Proto-decoder generates the first learnable anchor query and object query. (b) Anchor-guided decoder refine the anchor query after multi-modal context aggregation of each layer.

vectors: $Q_{o1} \in \mathbb{R}^{1\times C}$ and $Q_a \in \mathbb{R}^{1\times C}$. Q_{o1} is the object query we want. And Q_a is projected by an MLP with 4 output channels and a terminal sigmoid activation. The output is our first anchor box, denoted as $A_0 \in \mathbb{R}^{1\times 4}$. The 4 values along the channel dimension have a physical meaning and can be represented as a 4-tuple: (x_0, y_0, w_0, h_0), where x_0 and y_0 are the coordinates of the center point, and w_0 and h_0 are the height and width of the anchor, respectively. In rest of this section, A_0 (and the subsequent anchor boxes and their offsets) can be represented in either the matrix format or the tuple format, whichever is appropriate.

Continuous anchor-guided decoding. To provide better guidance for referring localization, inspired by the DAB-DETR [24] detection model, we propose a continuous anchor-guided decoder, which uses the anchor query A_0 as an initial bounding box and iteratively updates it based on multi-modal context aggregation. It is illustrated in Fig. 3 (b). We employ a stack of Transformer decoder layers from the original BERT [6] with one modification: The self-attention sub-layer is removed and each decoder layer only contains a cross-attention sub-layer and a feed-forward sub-layer. This adaptation is made as the input query to each layer consists of a single embedding instead of a sequence of embeddings, as detailed in the following.

Assuming there are n decoder layers (hence n cross-attention sub-layers), we use $i \in [1, 2, ...n]$ to index each layer. Given an anchor box A_{i-1}, similarly to [24], we project it to high-dimensional space to obtain an anchor query as follows,

$$Q_{ai} = \theta([\mathrm{PE}(x_{i-1}); \mathrm{PE}(y_{i-1}); \mathrm{PE}(w_{i-1}); \mathrm{PE}(h_{i-1})]), \tag{3}$$

where '[;]' indicates concatenation along the channel dimension, PE is a sinusoidal positional encoding function as we describe next, and θ is a fully connected layer with C output channels. To put it into words, to convert the anchor box into a high-dimensional embedding for use as a query, we first apply a positional encoding function which operates element-wise on each of the 4 parameters of the box and embeds each parameter into a high-dimensional embedding; then we concatenate the four embeddings along the channel dimension and projects it. Note that θ is shared across all layers. We adopt the same positional encoding function as defined in [6]:

$$\mathrm{PE}(z)_{2j} = \sin\left(\frac{z}{\mathcal{T}^{2j/D}}\right), \quad \mathrm{PE}(z)_{2j+1} = \cos\left(\frac{z}{\mathcal{T}^{2j/D}}\right), \tag{4}$$

where D defines the number of output channels, j is the index along the channel dimension, z is the input position variable, and $\mathcal{T}$ represents temperature. We discuss the hyper-parameters D and $\mathcal{T}$ in Sec. 4.2. The query input to the i-th cross-attention sub-layer is the following:

$$Q_i = [Q_{oi}; Q_{ai}], \tag{5}$$

where '[;]' denotes concatenation along the channel dimension, $Q_{oi} \in \mathbb{R}^{1\times C}$ is the object query, and $Q_{ai} \in \mathbb{R}^{1\times C}$ is the anchor query. With the multi-head attention mechanism in the cross-attention sub-layer, the concatenation of these two types of queries decouples context aggregation [31, 24], *i.e.*, the output embedding, $Q_{o(i+1)} \in \mathbb{R}^{1\times C}$, has the top/bottom half of channels generated based on the attention of the object query/anchor query, correspondingly. $Q_{o(i+1)}$ is the object query in the next

decoder layer. To generate the next anchor box, similarly to [24], we predict a set of offsets:

$$A_{\Delta i} = \sigma(\text{MLP}(Q_{o(i+1)})), \tag{6}$$

where σ denotes the sigmoid function and $A_{\Delta i}$ is the set of box offsets. Note that the MLP is shared across all layers. We obtain a new anchor box by updating the previous one with the predicted offsets:

$$A_i = A_{i-1} + A_{\Delta i}, \tag{7}$$

We iterate through all n layers of the decoder according to Eqs. (3)-(7). The last anchor box, A_n, is used as the final prediction.

The loss function. We use the sum of the Generalized IoU [37] loss and the L1 loss for training our model, which is defined as $\mathcal{L} = \mathcal{L}_{\text{giou}}(B, \tilde{B}) + \|B - \tilde{B}\|_1$, where B is the ground truth, and $\tilde{B}$ is our prediction. We do not apply balancing weights on the two losses.

4 Experiments

4.1 Datasets and metrics

RefCOCO [51], RefCOCO+ [51], and RefCOCOg [32]. The RefCOCO dataset contains 19,994 images, 50,000 objects, and 142,209 referring expressions. The RefCOCO+ dataset contains 141,564 expressions for 49,856 objects in 19,992 images. The RefCOCOg dataset contains 85,474 expressions for 54,822 objects in 26,711 images. Referring expression annotations in RefCOCO and RefCOCO+ are collected in an interactive two-player game. In this game, each player is motivated to provide a minimally sufficient description for the other to identify the target. As a result, expressions in these two datasets tend to be succinct, averaging fewer than 4 words per sentence. A special property of RefCOCO+ is that location words (such as "top", "bottom", "left", *etc.*) are banned in its annotations, which makes it the harder of the two. On the contrary, expressions in RefCOCOg are annotated on Amazon Mechanical Turk in a non-competitive way, and thus tend to be longer (8.43 words per sentence on average) and more expressive. This makes RefCOCOg a particularly challenging dataset compared to the other two.

ReferIt Game [18]. The ReferIt Game (or sometimes, simply ReferIt) dataset contains 20,000 images collected from the SAIAPR-12 dataset [8] with automatically generated referring expressions. Following standard practice [16, 48, 5, 47], we split this dataset into 9,000 images for training, 1,000 images for validation, and 10,000 images for testing, with 54K, 5.8K, and 60K referring expressions for these splits, respectively.

Flickr30K Entities [34]. The Flickr30K Entities dataset is a popular benchmark for the phrase grounding task, in which the model is required to localize multiple entities from an image caption. In our experiments, we follow a variant of the above setting which is commonly adopted by many previous REC methods [38, 52, 48, 47, 50, 5], *i.e.*, each phrase in a given image caption is treated as an independent referring expression without considering the context of the whole sentence. This dataset contains 31,783 images and 427K entities, with 29,783 images for training, 1,000 images for validation, and 1,000 images for testing.

Evaluation metrics. We adopt the standard metric of Acc@0.5, which measures the percentage of test samples for which the intersection-over-union between the prediction and the ground truth is above 0.5. This metric is also often referred to as top-1 accuracy or simply accuracy in the literature.

4.2 Implementation details

We use Swin-B [26] as our visual backbone with classification weights pre-trained on ImageNet-22K. We adopt the uncased $\text{BERT}_{\text{BASE}}$ [6] model from the HuggingFace library [23] as our language model, and initialize it with official weights pre-trained on BooksCorpus [55] and English Wikipedia. All other parts of our model are randomly initialized. We adopt AdamW [27] as the optimizer, and set the learning rate to 0.0001 for the visual-linguistic Transformer encoder and 0.00005 for the visual and linguistic backbones. Following [5] [20], images are resized to 640×640 and augmented with random intensity saturation and affine transformations. On RefCOCO, RefCOCO+, RefCOCOg, and ReferIt Game, the model is trained for 90 epochs with the learning rate dropped by a factor of 10 after 60 epochs. On Flickr30K Entities, the model is trained for 60 epochs with the learning rate

Table 1: Comparison with state-of-the-art methods on four mainstream REC datasets in terms of Acc@0.5. Embedding initialization is not detailed except for vision-language pre-training methods, which do not employ a separate language model.

Method	Image Backbone	RefCOCO			RefCOCO+			RefCOCOg		ReferIt
		val	test A	test B	val	test A	test B	val	test	test
Task-specific:										
RvG-Tree [13]	ResNet-101	75.06	78.61	69.85	63.51	67.45	56.66	66.95	66.51	-
MAttNet [50]	ResNet-101	76.65	81.14	69.99	65.33	71.62	56.02	66.58	67.27	29.04
DGA [46]	ResNet-101	-	78.42	65.53	-	69.07	51.99	-	63.28	-
NMTTree [23]	ResNet-101	76.41	81.21	70.09	66.46	72.02	57.52	65.87	66.44	-
CM-Att-Erase [25]	ResNet-101	78.35	83.14	71.32	68.09	73.65	58.03	67.99	68.67	-
DDPN [52]	ResNet-101	76.8	80.1	72.4	64.8	70.5	54.1	-	-	63.00
FAOA [48]	Darknet-53	72.54	74.35	68.50	56.81	60.23	49.60	61.33	60.36	59.30
ReSC-Large [47]	Darknet-53	77.63	80.45	72.30	63.59	68.36	56.81	67.30	67.20	64.60
MCN [30]	Darknet-53	80.08	82.29	74.98	67.16	72.86	57.31	66.46	66.01	-
RCCF [22]	DLA-34	-	81.06	71.85	-	70.35	56.32	-	65.73	63.79
TransVG [5]	ResNet-101	81.02	82.72	78.35	64.82	70.70	56.94	68.67	67.73	70.73
RefTR [20]	ResNet-101	82.23	85.59	76.57	71.58	75.96	62.16	69.41	69.40	71.42
LUNA (Ours)	ResNet-101	83.53	85.79	80.08	72.60	77.90	64.51	72.40	71.81	72.67
LUNA (Ours)	Swin-B	**86.12**	**88.43**	**82.63**	**75.96**	**80.62**	**68.36**	**76.51**	**76.55**	**73.69**

dropped by a factor of 10 after 40 epochs. When adopting pre-training, we follow [20] and train our model on Visual Genome [19] for 6 epochs, and fine-tune it on the evaluation dataset for 50 epochs. The maximum expression length is 20 on all datasets. The visual-linguistic Transformer encoder and our decoder both have 6 layers. C, $\mathcal{T}$, and D in Sec. 3 are empirically set to 256, 20, and 128, respectively. Models adopting a Swin-B backbone are trained with mini-batches of size 8, and those adopting a ResNet-101 [11] backbone are trained with batch size 40. We run several experiments with different random seeds, and the difference between different runs generally lies in $\pm 0.5\%$.

4.3 Comparison with others

In Table 1, we compare LUNA to state-of-the-art methods on four standard benchmarks for the REC task. We group the methods into task-specific ones, which do not employ additional training data, and pre-trained ones, which generally have a different training focus from ours, *i.e.*, learning transferable representation via large-scale pre-training. Among the task-specific methods, our method attains the highest Acc@0.5 across all subsets of all datasets. Specifically, on the val/test A/test B subsets of RefCOCO [51] and RefCOCO+ [51], LUNA outperforms the second best methods by absolute margins of 3.89%/2.84%/6.06% and 4.38%/4.66%/6.20%, respectively. On the most challenging RefCOCOg [32] dataset, LUNA is able to achieve the largest improvements by far: On the validation set and test set, LUNA obtains 7.10% and 7.15% absolute improvements over the second-best RefTR [20] model.

Observing the pattern across the three datasets, we can see that the more difficult the dataset is (please see Sec. 4.1 for a discussion), the larger the improvement LUNA is able to bring with respect to the state of the art. This corroborates our case for exploiting learnable location priors to facilitate box decoding. As discussed in Sec. 4.1, banning the use of location words makes RefCOCO+ a significantly harder dataset than RefCOCO is, as cross-modal alignment can only rely on appearance information. Similarly, the longer and more flowery descriptions in the RefCOCOg dataset makes purely content-based similarities less reliable as the basis of grounding. The relatively large improvements of LUNA on these two datasets provide evidence that when content-based mapping is not easy or accurate, our proposed

Table 2: Results on Flickr30K Entities.

Method	Visual Backbone	Flickr30K test
CITE [33]	VGG-16	61.89
Similarity Net [45]	VGG-16	60.89
DDPN [52]	ResNet-101	73.30
FAOA [48]	Darknet-53	68.69
ReSC-Large [47]	Darknet-53	69.28
TransVG [5]	ResNet-101	79.10
RefTR [20]	ResNet-101	78.66
LUNA (ours)	Swin-B	**81.14**

learnable location priors can make up for it. Moreover, on the ReferIt [18] dataset, LUNA also surpasses all previous methods.

While pre-training is not the focus of this work, additional pre-training of LUNA on the Visual Genome [19] region description splits can further boost performance, and our method obtains comparable or better results with respect to other methods that employ pre-training. More result and comparison please see Table 7 in Supplementary Material.

In Table 2, we evaluate LUNA against the state-of-the-art methods on the Flickr30K Entities [34] dataset. Please refer to Sec. 4.1 for a discussion on the specific task setting. Compared to TransVG [5] and RefTR [20], two other state-of-the-art models based on Transformers, our method obtains 2.04% and 2.48% (absolute) improvements in terms of Acc@0.5, respectively.

4.4 Ablation study

In this section, we conduct extensive ablation experiments to study the effects of the vision backbone, the main model components, and alternative design choices in the proto-decoder and the continuous anchor-guided decoder. Experiments in this section are conducted on the validation and test sets of the RefCOCOg dataset. ResNet-101 is adopted as the vision backbone and BERT is adopted as the language backbone, unless otherwise specified.

The vision backbone. We unify the vision backbone in our method and the previous state-of-the-art RefTR [20] model to separate contributions of the vision backbone. As shown in Table 3, when adopting the same ResNet-101 [11] backbone, our method outperforms RefTR by absolute margins of 2.99% and 2.41% on the validation and test sets of RefCOCOg, respectively. When adopting the Swin-B backbone, our method also establishes 1.60% and 1.44% advantages on the validation and test sets, respectively.

Table 3: Ablation study of the vision backbone network on RefCOCOg.

Backbone	val		test	
	RefTR [20]	Ours	RefTR [20]	Ours
R101	69.41	**72.40**	69.40	**71.81**
Swin-B	74.91	**76.51**	75.11	**76.55**

Table 4: Component ablations on RefCOCOg.

Proto-decoder	CA-guided decoder	val	test
✓	✓	72.40	71.81
✓		69.58	69.82
	✓	70.66	70.56

Model components. In Table 4, we investigate the contributions of the proto-decoder and the continuous anchor-guided decoder (CA-guided decoder). We first remove the CA-guided decoder from the full model, and use the initial anchor produced from the proto-decoder as the prediction. This model variant leads to 2.82% and 1.99% drop in Acc@0.5 on the validation and test sets of RefCOCOg, respectively. These results validate the effectiveness of our proposed CA-guided decoder. We then remove the proto-decoder from the full model, and randomly initialized learnable embeddings of object query and anchor box which send to the CA-guided decoder. This model variant leads to 1.74% and 1.25% drop in Acc@0.5 on the validation and test sets of RefCOCOg, respectively. This demonstrates the effectiveness of the proposed proto-decoder.

Object representation in the proto-decoder. As described in Sec. 3.2, after obtaining a sequence of word-specific visual context embeddings, we obtain an object representation by averaging the sequence with learned weights for predicting the initial object query and anchor box. In Table 5, we study three other alternatives for summarizing object information. In the '[CLS] Token' alternative, we use the embedding output from the special '[CLS]' token input as representation of the object. In the 'Max Pooling' alternative, we directly pool the maximum activation along each token dimension across the sequence. As for 'Average pooling' alternative, we apply average pooling on the sequence length dimension to generate the object representation. As shown in Table 5, the three methods produce comparable results with weighted average pooling having a slight advantage, and we adopt it as the default choice.

The number of layers in CA-guided decoder. In Table 6, we study the effects of the number of decoding layers in our proposed CA-guided decoder. We test model variants which adopt 2, 4, 6, and 8 layers. Accuracy improves as more layers are added until 6 layers have been used.

Alternative studies of proto-decoder. We carefully designed the structure of proto-decoder to generate more relyable initial object query and anchor query, and consider several replaceable cases

Table 5: Alternative object representations in the proto-decoder.

Method	val	test
CLS Token	72.36	71.09
Max Pooling	71.73	71.69
Average Pooling	72.03	71.66
Weighted Avg.	**72.40**	**71.81**

Table 6: Effects of the number of layers in CA-guided decoder.

# of Layers	val	test
0	69.58	69.82
2	70.44	70.43
4	71.32	70.64
6	**72.40**	**71.81**
8	70.42	70.38

(including self-attention, visual query *etc.*). Due to the page limit, more details please see Table 8 in Supplementary Material.

4.5 Visualization

In Fig. 4, we visualize the attention patterns and predictions of LUNA and RefTR [20] on some challenging examples. Notice that for both models, the prediction is made where query attention is focused. This is expected as the box is predicted from a context feature vector pooled according to query attention. In a cluttered scene depicted on the left, our model correctly localizes the muffins among many similar distractors such as sandwiches in the middle or on top. In the scene on the right, the wing of the airplane is depicted from a rare angle and not quite noticeable from its background. Our model also correctly localizes the target in this case. In comparison, the query attention of RefTR peaks at wrong locations and leads to wrong box predictions.

Figure 4: Visualized query attention from the last decoder layer and the final predictions of our method and RefTR [20]. Cyan boxes are the ground truth and red boxes are predictions. Examples are from the validation set of RefCOCOg [32].

5 Conclusion

In this paper, we have presented a Transformer-based decoding method for referring expression comprehension, which exploits a series of language-guided anchor boxes as helpful spatial cues for guiding context pooling in a Transformer decoder. Extensive experiments on five benchmarks demonstrate its advantage with respect to the state-of-the-art methods.

Limitations and future work. Our model generates one anchor query to estimate the location of referred-to object. We plan to explore the options of generating multiple anchors in parallel and using a negative anchor mining strategy to select a best position prior. It is also desirable to extend our anchor-guided solution to other visual grounding tasks such as phrase grounding which requires localizing multiple objects in the image.

References

[1] Lisa Anne Hendricks, Oliver Wang, Eli Shechtman, Josef Sivic, Trevor Darrell, and Bryan Russell. Localizing moments in video with natural language. In *ICCV*, 2017. 3

[2] Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *ICCV*, 2015. 3

[3] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, 2020. 3, 4, 14

[4] Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *ECCV*, 2020. 17

[5] Jiajun Deng, Zhengyuan Yang, Tianlang Chen, Wengang Zhou, and Houqiang Li. Transvg: End-to-end visual grounding with transformers. In *ICCV*, 2021. 1, 2, 3, 4, 6, 7, 8, 14

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1, 3, 4, 5, 6

[7] 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. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 3

[8] Hugo Jair Escalante, Carlos A Hernández, Jesus A Gonzalez, Aurelio López-López, Manuel Montes, Eduardo F Morales, L Enrique Sucar, Luis Villasenor, and Michael Grubinger. The segmented and annotated iapr tc-12 benchmark. In *CVIU*, 2010. 6

[9] Zhe Gan, Yen-Chun Chen, Linjie Li, Chen Zhu, Yu Cheng, and Jingjing Liu. Large-scale adversarial training for vision-and-language representation learning. In *NeurIPS*, 2020. 17

[10] Jiyang Gao, Chen Sun, Zhenheng Yang, and Ram Nevatia. Tall: Temporal activity localization via language query. In *ICCV*, 2017. 3

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 3, 7, 8

[12] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. In *Neural Computation*, 1997. 3

[13] Richang Hong, Daqing Liu, Xiaoyu Mo, Xiangnan He, and Hanwang Zhang. Learning to compose and reason with language tree structures for visual grounding. In *TPAMI*, 2019. 7

[14] Ronghang Hu, Marcus Rohrbach, Jacob Andreas, Trevor Darrell, and Kate Saenko. Modeling relationships in referential expressions with compositional modular networks. In *CVPR*, 2017. 1, 2

[15] Ronghang Hu and Amanpreet Singh. Unit: Multimodal multitask learning with a unified transformer. In *ICCV*, 2021. 3

[16] Ronghang Hu, Huazhe Xu, Marcus Rohrbach, Jiashi Feng, Kate Saenko, and Trevor Darrell. Natural language object retrieval. In *CVPR*, 2016. 2, 6

[17] Aishwarya Kamath, Mannat Singh, Yann LeCun, Gabriel Synnaeve, Ishan Misra, and Nicolas Carion. Mdetr-modulated detection for end-to-end multi-modal understanding. In *ICCV*, 2021. 3, 17

[18] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to Objects in Photographs of Natural Scenes. In *EMNLP*, 2014. 2, 6, 8

[19] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International journal of computer vision*, 123(1):32–73, 2017. 7, 8

[20] Muchen Li and Leonid Sigal. Referring transformer: A one-step approach to multi-task visual grounding. In *NeurIPS*, 2021. 1, 2, 3, 4, 6, 7, 8, 9, 14, 16, 17

[21] Shuang Li, Tong Xiao, Hongsheng Li, Wei Yang, and Xiaogang Wang. Identity-aware textual-visual matching with latent co-attention. In *ICCV*, 2017. 1

[22] Yue Liao, Si Liu, Guanbin Li, Fei Wang, Yanjie Chen, Chen Qian, and Bo Li. A real-time cross-modality correlation filtering method for referring expression comprehension. In *CVPR*, 2020. 2, 3, 7

[23] Daqing Liu, Hanwang Zhang, Feng Wu, and Zheng-Jun Zha. Learning to assemble neural module tree networks for visual grounding. In *ICCV*, 2019. 6, 7

[24] Shilong Liu, Feng Li, Hao Zhang, Xiao Yang, Xianbiao Qi, Hang Su, Jun Zhu, and Lei Zhang. DAB-DETR: Dynamic anchor boxes are better queries for DETR. In *ICLR*, 2022. 3, 5, 6

[25] Xihui Liu, Zihao Wang, Jing Shao, Xiaogang Wang, and Hongsheng Li. Improving referring expression grounding with cross-modal attention-guided erasing. In *CVPR*, 2019. 7

[26] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 3, 6

[27] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2019. 6

[28] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019. 3, 17

[29] Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 12-in-1: Multi-task vision and language representation learning. In *CVPR*, 2020. 3

[30] Gen Luo, Yiyi Zhou, Xiaoshuai Sun, Liujuan Cao, Chenglin Wu, Cheng Deng, and Rongrong Ji. Multi-task collaborative network for joint referring expression comprehension and segmentation. In *CVPR*, 2020. 1, 2, 7

[31] Depu Meng, Xiaokang Chen, Zejia Fan, Gang Zeng, Houqiang Li, Yuhui Yuan, Lei Sun, and Jingdong Wang. Conditional detr for fast training convergence. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, 2021. 5

[32] Varun K. Nagaraja, Vlad I. Morariu, and Larry S. Davis. Modeling context between objects for referring expression understanding. In *ECCV*, 2016. 2, 6, 7, 9, 16

[33] Bryan A Plummer, Paige Kordas, M Hadi Kiapour, Shuai Zheng, Robinson Piramuthu, and Svetlana Lazebnik. Conditional image-text embedding networks. In *ECCV*, 2018. 7

[34] Bryan A Plummer, Liwei Wang, Chris M Cervantes, Juan C Caicedo, Julia Hockenmaier, and Svetlana Lazebnik. Flickr30k entities: Collecting region-to-phrase correspondences for richer image-to-sentence models. In *ICCV*, 2015. 2, 6, 8

[35] 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. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 3

[36] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015. 1, 2

[37] Hamid Rezatofighi, Nathan Tsoi, JunYoung Gwak, Amir Sadeghian, Ian Reid, and Silvio Savarese. Generalized intersection over union: A metric and a loss for bounding box regression. In *CVPR*, 2019. 6

[38] Anna Rohrbach, Marcus Rohrbach, Ronghang Hu, Trevor Darrell, and Bernt Schiele. Grounding of textual phrases in images by reconstruction. *ArXiv*, abs/1511.03745, 2016. 6

[39] Jing Shi, Ning Xu, Yihang Xu, Trung Bui, Franck Dernoncourt, and Chenliang Xu. Learning by planning: Language-guided global image editing. In *CVPR*, 2021. 1

[40] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, 2019. 3

[41] Stefanie Tellex, Nakul Gopalan, Hadas Kress-Gazit, and Cynthia Matuszek. Robots that use language. In *Annual Review of Control, Robotics, and Autonomous Systems*, 2020. 1

[42] Jasper RR Uijlings, Koen EA Van De Sande, Theo Gevers, and Arnold WM Smeulders. Selective search for object recognition. In *IJCV*, 2013. 1, 2

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 3

[44] Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. In *CVPR*, 2015. 3

[45] Liwei Wang, Yin Li, Jing Huang, and Svetlana Lazebnik. Learning two-branch neural networks for image-text matching tasks. In *TPAMI*, 2018. 1, 2, 7

[46] Sibei Yang, Guanbin Li, and Yizhou Yu. Dynamic graph attention for referring expression comprehension. In *ICCV*, 2019. 1, 2, 7

[47] Zhengyuan Yang, Tianlang Chen, Liwei Wang, and Jiebo Luo. Improving one-stage visual grounding by recursive sub-query construction. In *ECCV*, 2020. 1, 2, 6, 7

[48] Zhengyuan Yang, Boqing Gong, Liwei Wang, Wenbing Huang, Dong Yu, and Jiebo Luo. A fast and accurate one-stage approach to visual grounding. In *ICCV*, 2019. 1, 2, 6, 7

[49] Fei Yu, Jiji Tang, Weichong Yin, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. Ernie-vil: Knowledge enhanced vision-language representations through scene graph. In *AAAI*, 2021. 17

[50] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. Mattnet: attention network for referring expression comprehension. In *CVPR*, 2018. 1, 2, 6, 7

[51] Licheng Yu, Patrick Poirson, Shan Yang, Alexander C Berg, and Tamara L Berg. Modeling context in referring expressions. In *ECCV*, 2016. 2, 6, 7, 15

[52] Zhou Yu, Jun Yu, Chenchao Xiang, Zhou Zhao, Qi Tian, and Dacheng Tao. Rethinking diversified and discriminative proposal generation for visual grounding. *IJCAI*, 2018. 1, 2, 6, 7

[53] Luowei Zhou, Hamid Palangi, Lei Zhang, Houdong Hu, Jason Corso, and Jianfeng Gao. Unified vision-language pre-training for image captioning and vqa. In *AAAI*, 2020. 3

[54] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ICLR*, 2021. 3

[55] Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, 2015. 6

[56] C Lawrence Zitnick and Piotr Dollár. Edge boxes: Locating object proposals from edges. In *ECCV*, 2014. 1, 2

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [N/A] No theoretical results are included.
   (b) Did you include complete proofs of all theoretical results? [N/A] No theoretical results are included.
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [Yes]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes] Please see section 3.5 for details.
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [Yes] Please see section 3.5 for details.
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [Yes]
   (b) Did you mention the license of the assets? [No]
   (c) Did you include any new assets either in the supplemental material or as a URL? [No]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [No]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [No]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A] We did not use crowdsourcing or conducted research with human subjects.
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A] We did not use crowdsourcing or conducted research with human subjects.
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A] We did not use crowdsourcing or conducted research with human subjects.

A Learnable linguistic position embedding

In this section, we describe the learnable linguisic position embedding in detail (introduced in Sec. 3.1). To inject positional information of each tokens in multi-modal sequence, position encoding is an important procedure of transformer encoder. Existing works [5, 20] project linguistic token indices to high dimensional space by linear layers. We design a new position embedding method which unifies the position encoding of multi-modal sequence into 2D-sinusoidal embedding space. Specifically, for visual tokens, we follow [3] using sinusoidal positional encoding PE_{sin} to generate position embedding $\mathcal{P}_v$, while for each linguistic token, we leverage Multilayer Perceptrons to project the BERT embedding $\mathcal{F}_l$ and indices of tokens $X = \{0, 1, 2, ..., T\}$ into 2D coordinate:

$$X_1, X_2 = \sigma(\text{MLP}(\mathcal{F}_l) + \mathbf{W}X), \tag{8}$$

where MLP learns 2D coordinates from BERT embedding, and $\mathbf{W} \in \mathbb{R}^{2\times 1}$ projects 1D indices into 2D coordinates, σ denotes sigmoid activate function, X_1 and X_2 is the 2D position of linguistic tokens. Then we encode the 2D indices X_1, X_2 by the same PE_{sin}, that is:

$$\mathcal{P}_l = [PE_{sin}(X_1); PE_{sin}(X_2)], \tag{9}$$

where '[;]' denotes concatenation operation, $\mathcal{P}_l$ denotes the final language position embedding.

This simple implementation of unified 2D sinusoidal position embedding benefits our model from various aspect. For example, as described in Sec. 3.2, we leverage a proto-decoder to exploit the position prior and generate the first object query and anchor query. In an other word, this module decomposes a language query into two different type of queries, where the inner mechanism is: the cross attention layer decoupled semantic and positional attention calculation. The semantic similarity from query to key is conducted by dot products between visual and linguistic encoded features $\mathcal{F}_v$ and $\mathcal{F}_l$, while the positional similarity is the dot products between 2D sinusoidal position embedding $\mathcal{P}_v$ and $\mathcal{P}_l$. Benefited from our learnable language position embedding, the positional part can be naturally seen as calculate the correlation between language feature and visual patch positions, resulting a position prior over the image of the language query.

B Position prior from language

In this section, we visualize the position prior learned from linguistic information. It is sensible that a good REC model should be capable of estimating the location of object directly from language when the given language provides a strong position prior. As shown in Fig. 5, our model can predict the box precisely while RefTR [20] fails to handle these preposition words (*e.g.* under, above *etc.*) appeared in language.

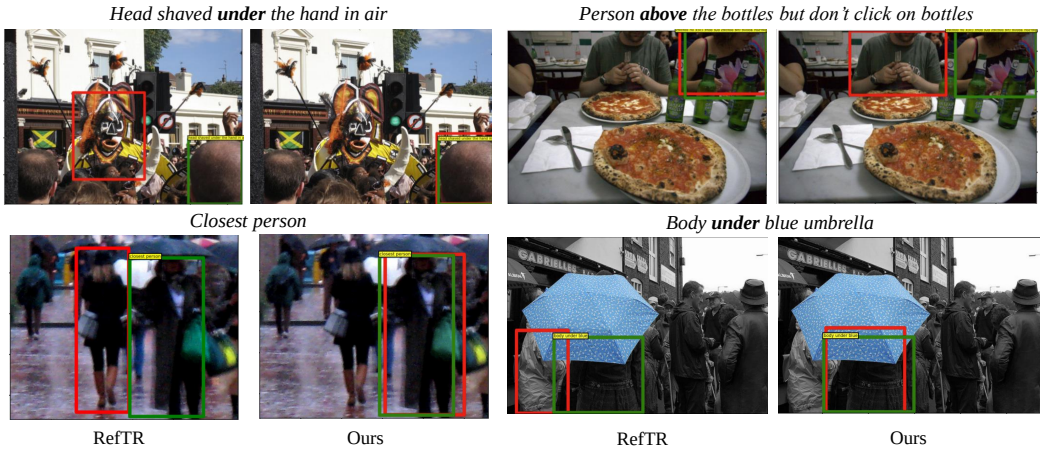


Figure 5: The comparison between our model and RefTR [20] when language contains directional words. Examples are from the validation set of RefCOCO+.

To have a deep understanding of exploiting position prior directly from language, we eliminate the impact of visual information during evaluation: we manually generate several image-sentence pairs,

where each image is random Gaussian noise and each sentence contains strong position information, such as “left object”, “right object”. The results are shown in Fig. 6.

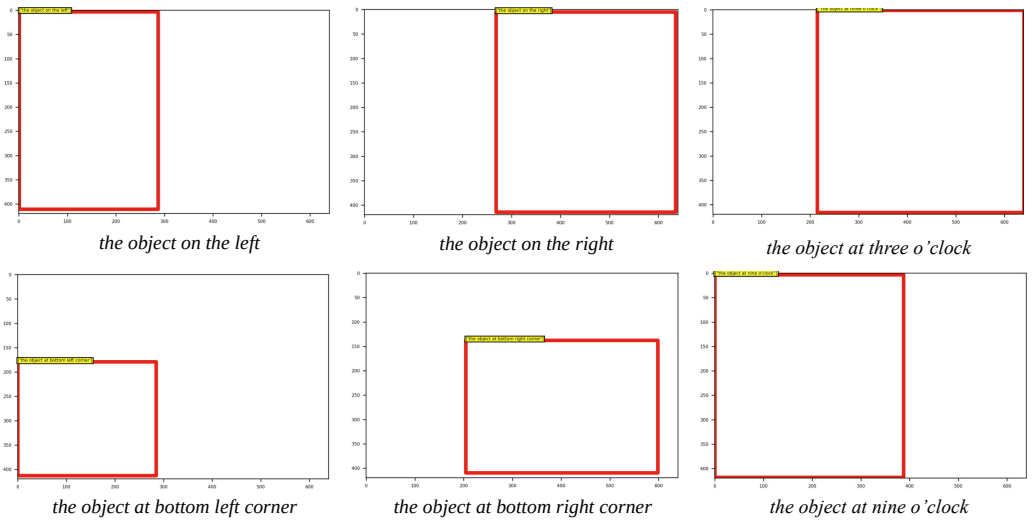


Figure 6: Visualization of our method when only language information provided and the image input are Gaussian noises. Examples of first and second columns are generated by model trained on RefCOCO dataset, and the last column examples are generated by model trained on RefCOCO+.

The first and second columns in Fig. 6 shows the prediction of our model trained on RefCOCO [51] dataset. We observe that when the visual information is unavailable, the model can still learn the position prior from language directly, locating the “object” under the guidance of preposition words appeared in sentence. The third column shows prediction of our model trained on RefCOCO+ [51]. Notice that in this dataset, some location word like “left” or “right” are taboo words. Instead, using “on nine o’clock” or “on three o’clock” to represent the direction of the object. As shown in Fig. 6, our model can identify these directional words and estimate the position of the object.

C Visualization of continuous anchor

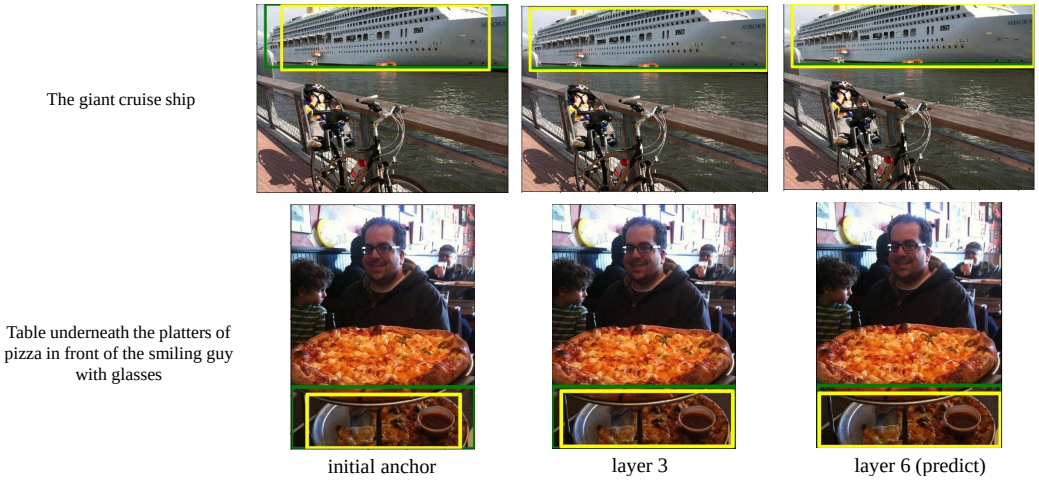


Figure 7: Visualization of continuous anchors. Left column shows the initial anchor query, middle column shows the prediction of layer 3, and last column shows the final prediction. Yellow boxes are anchors and green box are ground truth.

We visualize the continuous anchor generated by our model in Figure 7. The first column shows the initial anchors which are predicted by proto-decoder (decoupled positional information from language query), while the second and third columns demonstrate the refined anchors by layer 3 and layer 6 of our anchor-guided decoder (the final predicted bounding box). Notice that the sample of first row is generated by the model trained with auxiliary loss, hence the difference between layer 3 and layer 6 predictions is slight.

D More visualization of attention map

In Figure 8, we visualize more attention patterns and predictions of our model and RefTR [20]. Same as aforementioned in Sec. 4.5, thanks to the position prior, the attention of our model better focus on the region language referred.

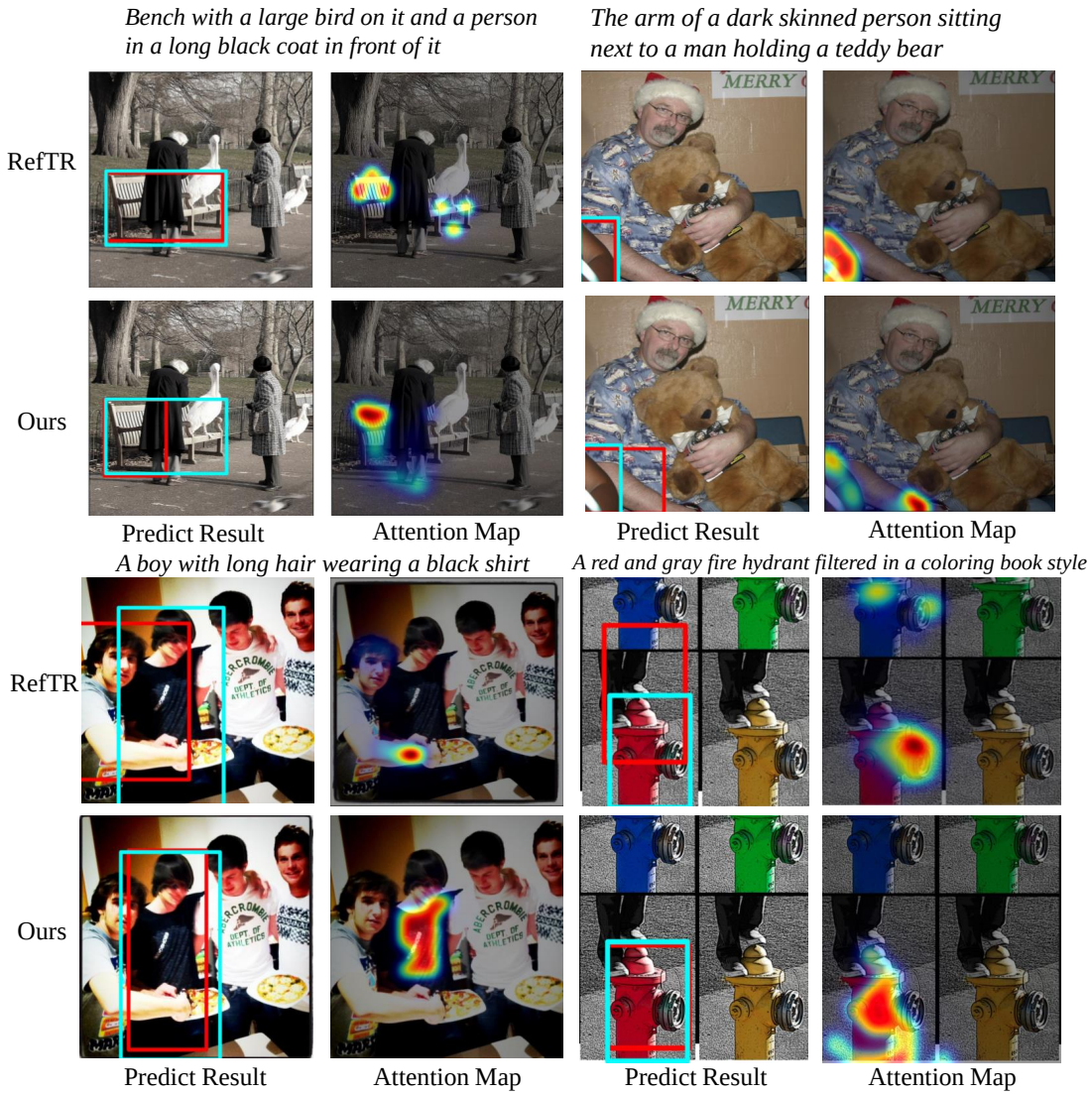


Figure 8: More visualization of attention map from the last decoder layer and the final predictions of our method and RefTR [20]. Cyan boxes are the ground truth and red boxes are predictions. Examples are from the validation set of RefCOCOg [32].

E Pretraining Result

Notably, LUNA outperforms RefTR [20] and MDETR [17] (which also employ bounding box-level supervision during pre-training similarly to LUNA) on all subsets of RefCOCO and RefCOCO+. Our method comes to the second only on the RefCOCOg subsets after MDETR, which uses twice as much pre-training data [17].

Table 7: Comparison with state-of-the-art pretrained-based methods on four mainstream REC datasets in terms of Acc@0.5.

Method	Image Backbone	RefCOCO			RefCOCO+			RefCOCOg		ReferIt
		val	test A	test B	val	test A	test B	val	test	test
Pre-trained:										
ViLBERT [28]	ResNet-101	-	-	-	72.34	78.52	62.61	-	-	-
ERNIE-ViL-large [49]	ResNet-101	-	-	-	75.89	82.37	66.91	-	-	-
UNITER-large [4]	ResNet-101	81.41	87.04	74.17	75.90	81.45	66.70	74.86	75.77	-
VILLA-large [9]	ResNet-101	82.39	87.48	74.84	76.17	81.54	66.84	76.18	76.71	-
RefTR-pre-trained [20]	ResNet-101	85.65	88.73	81.16	77.55	82.26	68.99	79.25	80.01	76.18
MDETR [17]	EfficientNet-B3	87.51	90.40	82.67	81.13	85.52	72.96	**83.35**	**83.31**	-
LUNA-pre-trained (Ours)	Swin-B	**88.85**	**91.31**	**84.75**	**81.26**	**86.05**	**74.85**	82.78	82.00	**79.52**

F Alternative studies of proto-decoder

In this section, we studied multiple alternative options of proto-decoder, the variant structures and experiment results are shown in Table 8.

The several replacable cases are: (1) A one-layer Transformer encoder ("Self-attention" in Table 8), where the output is a vision-language sequence which we average pool for obtaining Q.

(2) a variant of the proto-decoder ("reversed proto-decoder" in Table 8) in which the key, value are word embeddings and language position embeddings and the query is image region embeddings and visual position embeddings, where the output is a sequence of image region embeddings which we average pool for obtaining Q.

(3) a pair of parallel cross attention layers to calculate content embeddings and positional embeddings correlations respectively between image and sentence, instead of one cross attention layer.

(4) The cross attention layer in proto-decoder only calculate similarity of content embeddings then pooled without splitting. The first anchor query is randomly initialized.

(5) Calculate the similarity between language and visual positions via language content embeddings and visual position embeddings (instead of learnable language position embedding and visual position embeddings in Figure 3)

These results should better illustrate the effectiveness of the proposed proto-decoder.

Methods	val	test
Self-attention	70.42	70.36
reversed proto-decoder	72.18	71.56
Parallel Cross Attention	70.82	71.47
Content only Cross Attention	70.50	70.88
2 $\times$ Lang. Content Query	71.11	71.23
Ours Proto-decoder	**72.40**	**71.81**

Table 8: Alternatives to the proto-decoder.