RESEARCH ARTICLE

RSP Science Hub

International Research Journal on Advanced Science Hub
2582-4376
www.rspsciencehub.com
Vol. 08, Issue 09 September

Check for updates

http://dx.doi.org/10.47392/IRJASH.2026.038

Exploring Auto-Annotation Using Image-Caption Pairs and Deep Reinforcement Learning

Gururaja H S [1], M G Aditya [2], Rahul Pramod Krupani [3], Rudramurthy V C [4], Padmanabha J [5]

[1] Associate Professor, Dept. of Computer Science and Business Systems, B.M.S. College of Engineering Bangalore, Karnataka, India

[2,3] Student, Dept. of Information Science and Engineering, B.M.S. College of Engineering, Bangalore, Karnataka, India.

[4]Assistant Professor, Dept. of Computer Science and Business Systems, B.M.S. College of Engineering, Bangalore, Karnataka, India.

[5]Assistant Professor, Dept. of Information Science and Engineering, Bangalore Institute of Technology, Bangalore, Karnataka, India

Emails: *gururajhs.ise@bmsce.ac.in[1], adithyaprakash6@gmail.com[2], rahul.krupani@outlook.com[3], rudramurthyvc.ise@bmsce.ac.in[4], padmanabhaj1@gmail.com[5]*

Article history

Received: 11 August 2026
Accepted: 05 September 2026
Published:30 September 2026

Keywords:
Automatic image annotation, Referring expression grounding, Deep reinforcement learning, Actor–critic, VisualBERT, Detectron2, Bounding-box refinement, Vision–language models, RefCOCO.

Abstract

Automatic image annotation is important in many computer vision applications, such as autonomous vehicles, medical imaging, intelligent monitoring and robotics. Traditional annotation relies on manually labeled datasets, which makes the process long, costly and difficult. Although vision-language systems have improved referring expression grounding, most methods use the same fixed strategy for localization. To address this limitation, this paper introduces a Deep Reinforcement Learning-based Automatic Image Annotation Framework that combines Detectron2, VisualBERT and an actor–critic reinforcement learning mechanism. Detectron2 first detects regions of interest; VisualBERT then merges the visual and textual information, enabling the actor–critic agent to refine the localization using an IOU-based reward. On the RefCOCO dataset, the framework achieved average accuracies of 49.43% on the validation split and 46.70% on the Test A and Test B splits, with maximum accuracies of 95.16%, 82.96% and 99.01%, respectively, showing that it can associate the textual and visual components of an image.

1. Introduction

Computer vision enables machines to interpret images and video and to act on what they see. Modern systems are built on deep learning, in particular convolutional neural networks (CNNs), which learn visual features directly from large collections of labeled examples [1]. This dependence on labels is the main bottleneck. Applications such as autonomous driving, medical imaging, surveillance and robotics each need large, task-specific datasets in which every object is marked with a bounding box and a class, and producing these annotations by hand is slow and expensive. Workarounds such as transfer learning and one-shot learning reduce the amount of labeled

data required, but do not remove the need for it.

Image–caption pairs offer a cheaper source of supervision: a short natural-language description of an object can guide a model to the corresponding region of the image. This paper presents a framework that turns such pairs into bounding-box annotations. Candidate regions are proposed by Detectron2, fused with the caption using VisualBERT, and refined by a deep reinforcement learning (DRL) agent trained with an actor–critic method and an IOU-based reward. The remainder of the paper reviews related work, describes the framework and its implementation, and reports results on RefCOCO.

2. Literature Survey

1. Chen et al. (2025) [2] presented a Task-aware Cross-modal Feature Refinement Transformer (TCRT) that uses large language models to improve visual grounding. The framework adds an LLM adaptation module and a cross-modal feature fusion module to refine visual features with task-aware semantic priors. Experiments on RefCOCO, RefCOCO+, RefCOCOg and ReferItGame showed that it outperforms earlier end-to-end visual grounding approaches. However, it refines features with a transformer rather than optimizing object localization as a sequence of decisions with reinforcement learning.

2. Wang et al. (2025) [3] proposed building a large-scale visual grounding dataset with generative vision-language models, which produce both object-level descriptions and referring expressions for object detection datasets. Training on the resulting data substantially improved referring expression comprehension and segmentation. The method addresses dataset generation only and does not use an optimization method such as reinforcement learning for annotation.

3. Florence-2 (Xiao et al., 2024) [4] is a vision foundation model with a unified, prompt-based interface, trained on a large set of automatically generated visual annotations. It can caption images, detect objects, ground phrases, segment images and answer visual questions from a prompt, and achieves strong zero-shot results across many vision-language tasks. It does not, however, adapt its localization through reinforcement learning.

4. Kosmos-2 (Peng et al., 2024) [5] is a multimodal large language model whose text output can be linked directly to image regions. It unifies visual grounding, referring expression comprehension, image captioning and visual question answering in one architecture and reasons well across modalities, but it is computationally expensive and does not address reinforcement learning-based annotation.

5. Mi et al. (2024) [6] proposed a zero-shot visual grounding method based on coarse-to-fine representation learning, using multi-head self-attention and hierarchical semantic learning to improve visual and textual representations so that target regions can be located for unseen query–image pairs. It improves grounding accuracy on standard datasets, but remains focused on representation learning rather than adaptive decision-making.

6. Grounding DINO (Liu et al., 2023) [7] is an open-set object detector that combines transformer-based detection with grounded language pre-training. It localizes arbitrary objects from natural-language prompts and performs strongly on RefCOCO, RefCOCO+ and RefCOCOg, but does not use reinforcement learning to refine localization step by step.

7. Li et al. (2023) [8] proposed BLIP-2, which connects frozen image encoders to large language models through a lightweight Querying Transformer (Q-Former). BLIP-2 achieved state-of-the-art results in image captioning, visual question answering and multimodal reasoning with high computational efficiency, showing that frozen foundation models can support many downstream tasks. It is designed for multimodal reasoning rather than reinforcement learning-based localization.

8. Shi et al. (2022) [9] addressed unpaired referring expression grounding with a bidirectional cross-modal matching framework. Instead of paired image–text annotations, the method learns image–language alignment from unpaired data, which greatly reduces annotation cost. It relies on pre-extracted object proposals and does not use reinforcement learning for adaptive localization.

9. Sun et al. (2021) [10] proposed a deep reinforcement learning framework for proposal-free referring expression grounding, in which an agent learns an iterative shrinking policy that localizes the target object step by step. The approach achieved higher localization accuracy than other proposal-free methods on RefCOCOg, but does not use transformer-based multimodal representations.

10. VisualBERT (Li et al., 2019) [11] jointly models

visual and linguistic inputs with a stack of self-attention layers and is pre-trained on image–caption data. It performs competitively on visual question answering, visual commonsense reasoning, natural-language visual reasoning and region-to-phrase grounding, and is a simple yet strong baseline for vision-and-language tasks. It fuses the modalities in a single pass and has no adaptive, reinforcement learning-based localization mechanism.

3. Proposed Methodology

3.1. Overall Architecture

The framework combines computer vision, vision-language learning and deep reinforcement learning to achieve automatic image annotation with precise object localization. It comprises five high-level components: image pre-processing, object detection, multimodal feature fusion, reinforcement learning-based localization and automatic annotation generation (Figures 1 and 2).

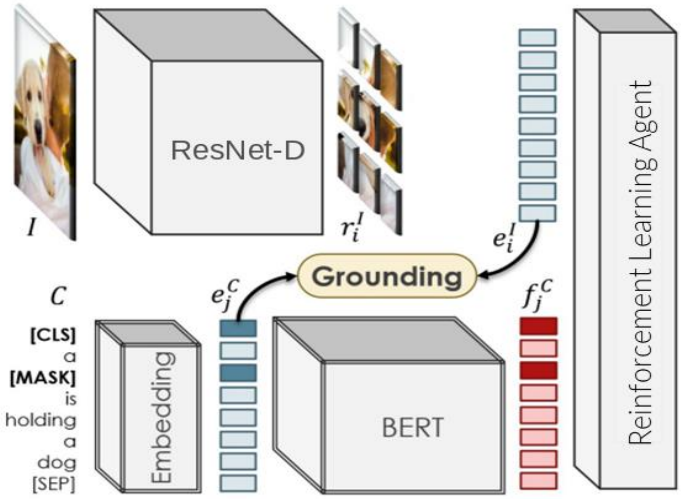


Figure 1 Architecture of the Proposed Vision–Language Grounding Framework (VisualBERT Block Adapted from [11]).

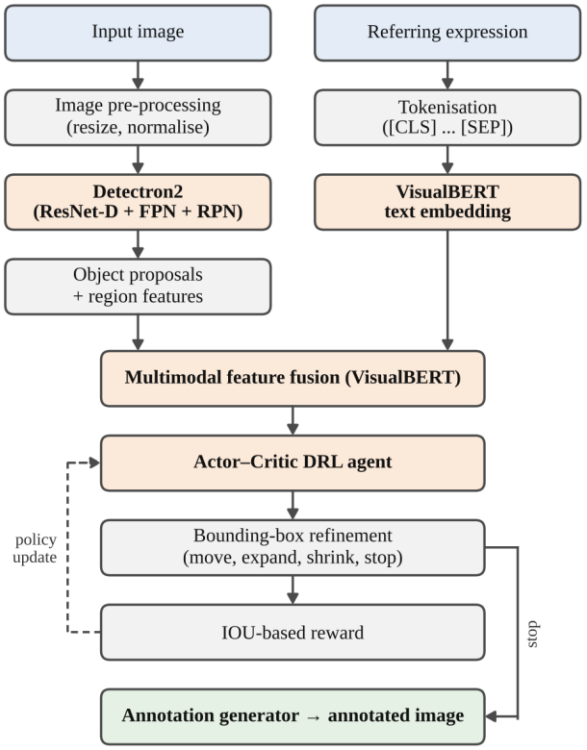


Figure 2 Processing Pipeline of the Proposed Framework, from Input Image and Referring Expression to the Annotated Image

First, a natural-language referring expression and its corresponding image are input to the system. The image is pre-processed (pixel normalization, resizing to a fixed size and quality enhancement) before visual features are extracted. In parallel, the referring expression is tokenized and converted into a contextual embedding by the VisualBERT language encoder. Detectron2 [12] with a ResNet-D backbone [13] and FPN [14] then locates multiple candidate object regions and extracts features for each of them. These region features are fused with the linguistic embedding of the referring expression through the multimodal attention transformer of VisualBERT, so that the visual and linguistic modalities are aligned by semantic similarity and meaningful correlations are established. The fused multimodal features are fed into the deep reinforcement learning (DRL) agent, which uses an actor–critic framework [10], [15] to search for the precise object location. The agent selects actions that adjust the bounding box, either by shifting the region of interest or by changing its size. At each step of interaction with the environment, the action taken by the agent is rewarded with an IOU-based metric. An increase in IOU earns a positive reward, which encourages accurate localization, whereas a decrease earns a negative reward, which discourages incorrect adjustments. Through this interaction, the agent learns an optimal policy, adjusts the bounding box precisely and converges on the target object region. Finally, the refined bounding box and the predicted semantic label of the object form the automatically generated image annotation, providing accurate localization and object context for vision-language applications.

3.2. Dataset Description

The framework is evaluated on RefCOCO [16], which is built on MSCOCO [17]. MSCOCO provides object-level annotations for 80 categories, with about 118k training, 5k validation and 41k test images, and RefCOCO adds natural-language referring expressions for the annotated objects. Evaluation on RefCOCO+ and RefCOCOg is left for future work.

The standard training, validation and test splits are used. The training set is used to learn the joint representations and the reinforcement learning policy, the validation set to select hyperparameters, and the test sets to measure how well grounding and localization generalize.

3.3. Image Pre-processing

Each input image is resized to a fixed resolution while preserving its aspect ratio and normalized with the ImageNet mean and standard deviation [18]. During training, random cropping, scaling and horizontal flipping are applied to improve generalization and reduce overfitting. Flipping doubles the variety of object positions and viewpoints seen by the agent at no labelling cost, but it would invalidate expressions that contain spatial words; whenever an image is flipped, the words “left” and “right” in its referring expression are therefore swapped, so that the expression still describes the same object. In parallel, each referring expression is tokenized with the VisualBERT tokenizer, with the special tokens [CLS] and [SEP] added, to produce contextual word embeddings [11]. The pre-processed image and text are then passed to the feature extraction models.

3.4. Detectron2

Detectron2 [12] is the main object detection framework and proposes candidate object regions. It uses the Faster R-CNN model [19] with a region proposal network (RPN). For each detected object, it returns a bounding box and a feature vector, and it detects multiple objects quickly and accurately.

3.5. ResNet-D Backbone with Feature Pyramid Network

Visual features are learned with a ResNet-D backbone [13] and a feature pyramid network (FPN) [14]. ResNet-D keeps the residual connections of ResNet [1] and improves the down-sampling path, so that high-level features are learned efficiently. The FPN combines low-level spatial detail with high-level semantic features to produce feature maps at several scales, so that objects of different sizes are handled well.

3.6. VisualBERT-Based Multimodal Feature Extraction

VisualBERT [11] links the detected objects to the referring expression. The features of the detected objects are concatenated with the tokenized referring expression, and the resulting sequence is passed through transformer layers that learn the interaction between the two. The output is a multimodal embedding that captures the correspondence between image regions and words.

G. Deep Reinforcement Learning Framework

A DRL framework is designed to refine the localization of the object. The environment is

formed by the detected candidate objects, and the state consists of the multimodal features, the text embedding and the current bounding-box coordinates. Based on the current state, the agent takes actions (moving, enlarging, shrinking or terminating) to update the bounding box. The environment then computes the IOU reward to evaluate the quality of the action. The agent interacts with the environment for several steps until the final bounding box is predicted.

3.7. Policy Optimization by Actor–Critic

A DRL agent must learn a policy, and that policy must be optimized for good performance. An actor–critic policy optimization method is used, since it has been shown to improve stability and reduce the variance of the policy gradient. The actor chooses actions that move the bounding box towards the target region, and the critic estimates the value of the current state; the temporal-difference error computed by the critic is used to update both the actor and the critic.

3.8. Automatic Annotation Generation

The refined bounding box at the end of the DRL process is taken as the localization of the target object, and the semantic label obtained from VisualBERT is used to generate the annotation automatically. The final output is the location of the object together with its semantic context.

4. Methodology

The system has a linear pipeline that consists of two main parts:

4.1. Visual pipeline

The visual pipeline prepares images with bounding boxes on which the reinforcement learning model acts. The RefCOCO set is generated from MSCOCO with the Cython-based masking API, and its test split provides the unlabeled but bounded data. Feature extraction, the next major step, is implemented with Detectron2 in PyTorch, and each image is resized to a fixed input size beforehand. The vanishing-gradient problem is addressed with skip connections, in which the output of a layer X is passed directly to layer X+2, bypassing the layer in between.

4.2. Linguistic pipeline

The linguistic pipeline processes natural-language input to guide decision-making in tasks that require visual-linguistic reasoning. It starts by tokenizing and embedding textual data using models such as BERT [20], which transform instructions or queries into structured data. Semantic parsing is then applied to extract key entities and relationships, aligning the linguistic information with the visual features processed earlier. Attention mechanisms map relevant parts of the text to specific regions of the image, enabling the system to create contextually aware embeddings. These embeddings, processed through bidirectional encoders or transformers, help the model understand the sequence and meaning of the text. Once integrated with the visual data, these linguistic features assist reinforcement learning models in tasks such as object detection, question answering, and other interactive operations, improving accuracy and decision-making.

5. Framework Components

5.1. CNN Feature Extractor

Visual features are extracted with a convolutional neural network (Figure 3). Each stage applies convolution and ReLU layers followed by down-sampling, so the feature maps lose spatial resolution but gain semantic depth; the outputs of the last four stages (C2–C5) form the input to the feature pyramid network.

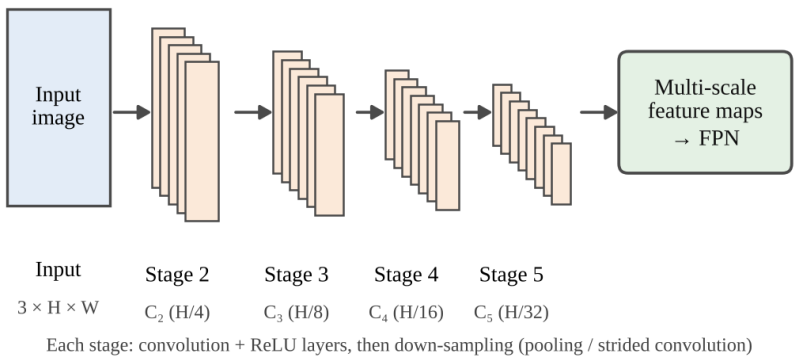


Figure 3 CNN-Based Feature Extraction: Successive Convolution Stages Produce Multi-Scale Feature Maps (C2–C5) That Are Passed to The FPN

5.2. VisualBERT

VisualBERT [11] is a vision-and-language model built from a stack of transformer layers in which self-attention aligns words in the text with regions in the image. Image regions and their bounding boxes are first extracted with a pre-trained object detector; the region features and the word embeddings are projected into a shared space and processed together. As in BERT [20], the text is enclosed by [CLS] and [SEP] tokens, and segment embeddings distinguish the textual and visual parts. The model supports tasks such as visual question answering, visual reasoning and region-to-phrase grounding, which makes it well suited to linking a caption with an object.

5.3. ResNet-D

ResNet-D is a variant of the ResNet architecture designed to enhance performance in deep neural networks, particularly in tasks such as object detection and image classification. The key modification in ResNet-D is its improved down-sampling strategy, where an average pooling layer is introduced before convolution to retain more spatial information, especially for small objects. This modification helps reduce information loss while maintaining computational efficiency. Additionally, ResNet-D [13] retains the residual (skip) connections of ResNet, improving gradient flow and enabling deeper networks to train effectively. These enhancements make ResNet-D particularly well suited to tasks requiring fine-grained feature extraction and robustness in training deep models.

5.4. Detectron2

Detectron2 [12] is an open-source library from Facebook AI Research (FAIR) for object detection and segmentation, written in PyTorch with CUDA support. It is a modular rewrite of the earlier Caffe2-based Detectron, so new models can be added without changing the rest of a pipeline, and it provides pre-trained models for tasks such as instance segmentation and person keypoint detection. It includes established architectures such as Faster R-CNN and Mask R-CNN as well as newer ones such as Cascade R-CNN and Panoptic FPN. The code is released under the Apache 2.0 license.

5.5. Feature Pyramid Network (FPN)

The FPN [14] builds a multi-scale feature pyramid from a single backbone (Fig. 4). The bottom-up pathway is the ordinary forward pass of the backbone, and the output of the last layer of each stage forms one pyramid level. The top-down pathway starts from the coarsest level and up-samples each map by a factor of two (nearest-neighbor), producing maps that are spatially coarse but semantically strong. Lateral connections merge each up-sampled map with the bottom-up map of the same size: the bottom-up map passes through a 1×1 convolution to match the channel dimension, and the two are added element-wise.

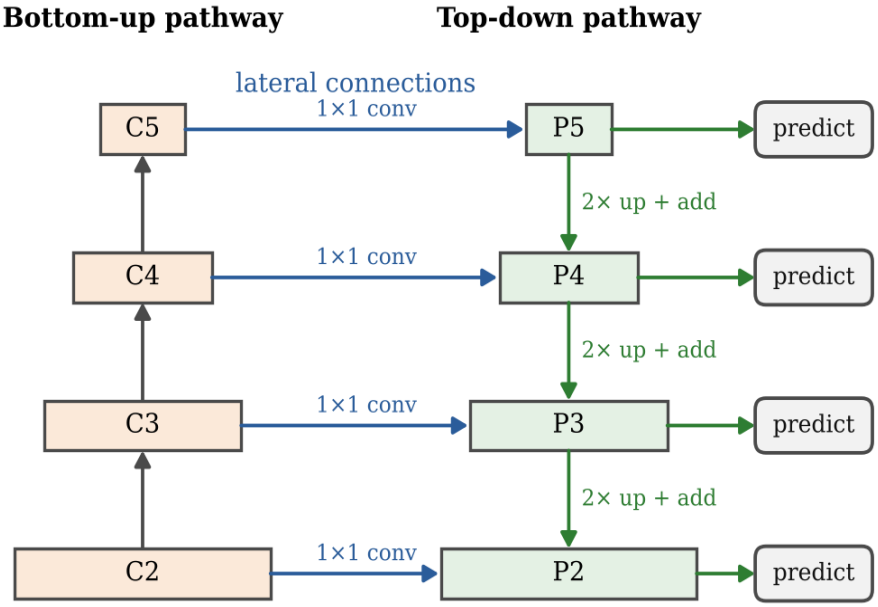


Figure 4 Bottom-Up and Top-Down Pathways of the Feature Pyramid Network, Joined by Lateral Connections

5.6. Reinforcement Learning Model

Reinforcement learning (RL) studies how an agent should act in an environment so as to maximize the reward it collects over time [15]; together with supervised and unsupervised learning it forms one of the three main branches of machine learning. At each time step t the agent is in state s_t, chooses an action $a_t \in A(t)$ from the set of available actions, moves to state s_{t+1} and receives a reward r_t. The agent aims to maximize the cumulative discounted reward (return):

$$R = \sum_{t=t_0}^{\infty} \gamma^{\,t-t_0}\, r_t$$

In this framework, the state is $S = \{F_v, F_l, F_{bbox}\}$, where
F_v: visual features,
F_l: linguistic features,
F_{bbox}: current bounding-box coordinates.
The action set is A = {Left, Right, Up, Down, Expand, Shrink, Stop}, and the reward is based on the change in IOU:

$$R = \begin{cases} +1, & IOU_{t+1} > IOU_t \\ 0, & IOU_{t+1} = IOU_t \\ -1, & IOU_{t+1} < IOU_t \end{cases}$$

The agent is trained with an actor–critic method [15]. The actor is a policy network $\pi_\theta(a|s)$ that gives a probability for each action in A for the current state, and the critic is a value network $V_w(s)$ that estimates the expected return from that state. At every step the reward is given by the change in IOU between the current box and the ground-truth box, as defined above: +1 when the IOU increases, 0 when it is unchanged and −1 when it decreases. During training, actions are sampled from π_θ, which provides exploration; at inference the most probable action is taken. An episode ends when the agent chooses Stop or reaches a maximum number of steps. After each action the critic computes the temporal-difference (TD) error δ_t, which measures whether the outcome was better or worse than expected, and both networks are updated with it: where γ is the discount factor and αa and αc are the learning rates of the actor and the critic. A positive δt makes the chosen action more likely in that state and a negative δt makes it less likely. Because both the policy and the value function are neural networks, the approach is a form of deep reinforcement learning.

$$\delta_t = r_t + \gamma\, V_w(s_{t+1}) - V_w(s_t)$$

$$\theta \leftarrow \theta + \alpha_a\, \delta_t\, \nabla_\theta \log \pi_\theta(a_t \mid s_t)$$

$$w \leftarrow w + \alpha_c\, \delta_t\, \nabla_w V_w(s_t)$$

6. Implementation

6.1. Dataset Creation

The RefCOCO dataset is generated from MSCOCO with the corresponding APIs. Cython is used for efficient dataset conversion because its performance is close to that of C.
RefCOCO [16] consists of 142,209 referring expressions for 50,000 objects in 19,994 images. It is generated from the MSCOCO dataset with the COCO API (https://github.com/cocodataset/cocoapi) and the refer API (https://github.com/lichengunc/refer).

Table 1 Summary of The REFCOCO Dataset Used for Training and Evaluation

Item	Details
Base dataset	MSCOCO 2017
Derived dataset	RefCOCO
Total images	19,994
Annotated objects	50,000
Referring expressions	142,209
Annotation type	Bounding boxes with referring expressions
Annotation format	JSON
Dataset conversion tool	Cython-based COCO API
Dataset purpose	Training and evaluation of automatic image annotation

MSCOCO is a large-scale object detection, segmentation and captioning dataset sponsored by Microsoft and Meta. The dataset was converted with Cython, a superset of Python that adds C-like static

typing, so that the conversion code runs at speeds close to those of C.

6.2. Feature Extraction

Visual features are extracted with ResNet-D and a Feature Pyramid Network (FPN). Detectron2 segments the image into objects, which are then fed into VisualBERT to establish the text–image correlation.

6.3. Linguistic Feature Extraction

The BERT tokenizer is used to extract linguistic features from the referring expressions, and the visual and linguistic features are fused with VisualBERT to establish the text–image correlation.

6.4. Actor-Critic Model

An actor model predicts actions from the integrated visual and linguistic features, and a critic model evaluates the value of the predicted actions, providing feedback for reinforcement learning.

6.5. Model Training

The actor and critic models are optimized with policy-gradient methods to maximize the rewards obtained while interacting with the environment. Rewards are computed from the change in Intersection over Union (IOU) between the predicted and ground-truth bounding boxes, as defined in the reinforcement learning model. The Adam optimizer [21] with configurable learning rates and weight decay is used for efficient optimization.

Table 2 Training Parameters of the Proposed Model

Parameter	Value
Optimizer	Adam [21]
Learning strategy	Actor–critic reinforcement learning
Reward function	IOU-based reward
Discount factor (γ)	0.99
Learning rate	1×10^{-4}
Weight decay	1×10^{-5}
Loss function	Actor loss + critic loss
Stopping criterion	Maximum epochs / convergence

Table 3 State Representation and Action Space of the Proposed Actor–Critic Model

Component	Description
State, $S(t)$	Visual features + linguistic features + bounding-box coordinates
Actions, $A(t)$	Move left, move right, move up, move down, expand, shrink, stop
Reward, $R(t)$	Intersection over union (IOU)-based reward
Policy	Actor–critic policy optimization
Objective	Maximize cumulative reward and localization accuracy

Algorithm

Input: RefCOCO dataset, Actor-Critic architecture
Output: Trained Actor and Critic models

- Load the RefCOCO dataset and preprocess the data.
- Extract visual features using ResNet-D and FPN architecture.
- Segment images into objects using Detectron2.
- Tokenize referring expressions to extract linguistic features.
- Fuse visual and linguistic features using VisualBERT.
- Train the Actor model to predict actions based on the integrated features.
- Train the Critic model to evaluate the value of predicted actions.
- Optimize the Actor and Critic models using policy gradient methods.
- Compute rewards based on the IOU between the predicted and ground-truth bounding boxes.
- Update model parameters using Adam optimizer with appropriate learning rates and weight decay.
- Repeat steps 6-10 until convergence or maximum iterations are reached.
- Output the trained Actor and Critic models.

7. Results

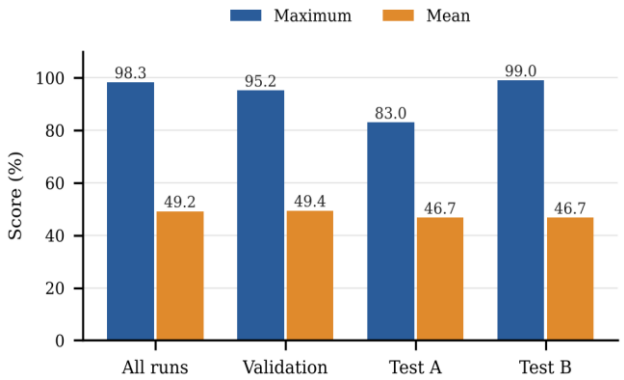


Figure 5 Maximum and Mean Scores of the Proposed Framework on All Runs and on the Validation, Test A and Test B Splits.



Figure 6 Qualitative Localization Result of the Proposed Automatic Image Annotation Framework

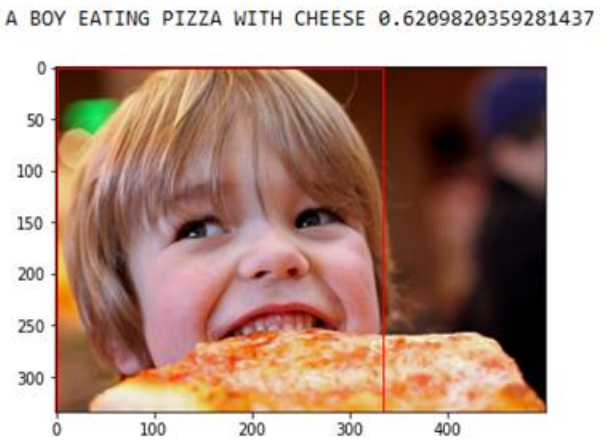


Figure 7 Qualitative Annotation Result for The Referring Expression "A Boy Eating Pizza with Cheese"

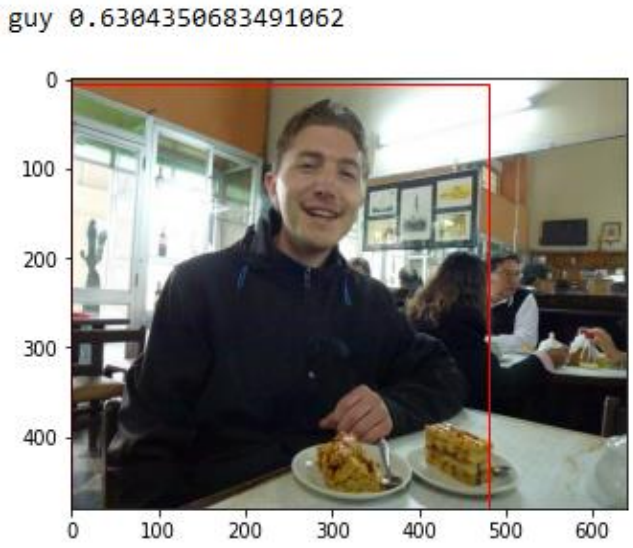


Figure 8 Qualitative Annotation Result for The Referring Expression "Guy"

7.1. Model Performance

The maximum accuracy achieved across all experiments is 98.27%, with an average accuracy of 49.17%. Li et al. [11] report ablation studies of VisualBERT in which removing task-agnostic pre-training or early fusion of vision and language causes the largest drops in performance, which motivates the use of VisualBERT as the fusion module in the proposed framework. Qualitative annotation results are shown in Figs. 6–8.

7.2. Testing Results

- On the validation set, the maximum accuracy achieved is 95.16%, with an average accuracy of 49.43%.
- On Test Set A, the maximum accuracy achieved is 82.96%, with an average accuracy of 46.70%.
- On Test Set B, the maximum accuracy achieved is 99.01%, with an average accuracy of 46.70%.
- These results summarize the performance of the model on the validation and test splits.

Conclusion and Future Work

This paper presented a Deep Reinforcement Learning-based Automatic Image Annotation Framework that employs Detectron2, ResNet-D with FPN, VisualBERT and an actor–critic RL architecture to locate objects precisely and provide semantic image annotations. By combining strong visual feature extraction with multimodal language understanding, the framework iteratively optimizes object localization from natural-language

descriptions. The reinforcement learning agent learns an optimal policy with IOU as the reward, which yields more accurate and contextualized annotations. Experiments were conducted on the RefCOCO dataset, which is derived from MSCOCO. The framework aligns visual objects with referring expressions and generates semantic annotations. VisualBERT strengthens the image–text correspondence, and actor–critic optimization keeps training stable and efficient. Compared with traditional supervised methods, the framework reduces the dependence on manual annotation and scales to other vision-language tasks. The framework has strong potential in applications such as image retrieval, autonomous systems, medical image analysis, intelligent surveillance, robotics, assistive technology, and visual question answering. Future research includes improving multimodal feature alignment using up-to-date vision-language foundation models such as BLIP-2, CLIP, Florence-2, and SAM for more accurate text-image correspondences and localization, investigating more advanced RL policies such as PPO and SAC to optimize the policy and improve training stability, and applying the framework to large-scale datasets such as RefCOCO+, RefCOCOg, Visual Genome and Flickr30K Entities to better illustrate its generality and efficiency on various real-world vision-language applications.

References

[1] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2016, pp. 770–778.

[2] W. Chen, Z. Xu, R. Xu, S. Wu, and H. S. Wong, “Task-aware cross-modal feature refinement transformer with large language models for visual grounding,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2025.

[3] S. Wang, D. Kim, A. Taalimi, C. Sun, and W. Kuo, “Learning visual grounding from generative vision and language model,” in Proc. IEEE/CVF Winter Conf. Appl. Comput. Vis. (WACV), 2025, pp. 8046–8056.

[4] B. Xiao, H. Wu, W. Xu, X. Dai, H. Hu, Y. Lu, M. Zeng, C. Liu, and L. Yuan, “Florence-2: Advancing a unified representation for a variety of vision tasks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2024.

[5] Z. Peng, W. Wang, L. Dong, Y. Hao, S. Huang, S. Ma, and F. Wei, “Kosmos-2: Grounding multimodal large language models to the world,” in Proc. Int. Conf. Learn. Represent. (ICLR), 2024.

[6] J. Mi, S. Jin, Z. Chen, D. Liu, X. Wei, and J. Zhang, “Zero-shot visual grounding via coarse-to-fine representation learning,” Neurocomputing, vol. 610, Art. no. 128621, 2024.

[7] S. Liu et al., “Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection,” arXiv:2303.05499, 2023.

[8] J. Li, D. Li, S. Savarese, and S. Hoi, “BLIP-2: Bootstrapping language-image pre-training with frozen image encoders and large language models,” in Proc. Int. Conf. Mach. Learn. (ICML), 2023.

[9] H. Shi, M. Hayat, and J. Cai, “Unpaired referring expression grounding via bidirectional cross-modal matching,” arXiv:2201.06686, 2022.

[10] M. Sun, J. Xiao, and E. G. Lim, “Iterative shrinking for referring expression grounding using deep reinforcement learning,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2021.

[11] L. H. Li, M. Yatskar, D. Yin, C.-J. Hsieh, and K.-W. Chang, “VisualBERT: A simple and performant baseline for vision and language,” arXiv:1908.03557, 2019.

[12] Y. Wu, A. Kirillov, F. Massa, W.-Y. Lo, and R. Girshick, “Detectron2,” 2019. [Online]. Available:https://github.com/facebookresearch/detectron2

[13] T. He, Z. Zhang, H. Zhang, Z. Zhang, J. Xie, and M. Li, “Bag of tricks for image classification with convolutional neural networks,” in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2019.

[14] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, “Feature pyramid networks for object detection,” in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2017.

[15] R. S. Sutton and A. G. Barto, Reinforcement Learning: An Introduction, 2nd ed. Cambridge, MA, USA: MIT Press, 2018.

[16] L. Yu, P. Poirson, S. Yang, A. C. Berg, and T. L. Berg, "Modeling context in referring expressions," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2016.

[17] T.-Y. Lin et al., "Microsoft COCO: Common objects in context," in Proc. Eur. Conf. Comput. Vis. (ECCV), 2014.

[18] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), 2009.

[19] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," in Adv. Neural Inf. Process. Syst. (NeurIPS), 2015.

[20] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in Proc. NAACL-HLT, 2019.

[21] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in Proc. Int. Conf. Learn. Represent. (ICLR), 2015.