Signal, Image and Video Processing
https://doi.org/10.1007/s11760-024-03449-x

ORIGINAL PAPER

A rich RGBD images captioning for scene understanding

Khadidja Delloul[1] · **Slimane Larabi**[1]

Received: 12 May 2024 / Revised: 25 June 2024 / Accepted: 18 July 2024

Abstract

In recent years, image captioning and segmentation have gained prominence in computer vision, finding applications in various fields, from autonomous driving to content analysis. While several solutions have been devised to enhance user experiences in navigating their environments, there remains a need for applications that provide detailed textual descriptions of scenes. Most existing models primarily focus on specific tasks, limiting their versatility in different scenarios. In this paper, we propose an innovative approach aimed at enhancing the comprehension of surroundings through image captioning. Our research distinctively offers textual descriptions for each segment within an image, including spatial orientations (e.g., left, right, front). This level of granularity ensures that anyone using our system can capture and comprehend every piece of information present in the image. We further extend the applicability of our solution by training and applying our methodology to theatre dataset. Our results demonstrate improved efficiency compared to state-of-the-art methods, coupled with the provision of more detailed descriptions for each segment within the input image.

Keywords Image captioning · Egocentric scene description · Dense captioning · Segmentation · RGB-D images

1 Introduction

With remarkable advancements in deep learning technologies, numerous applications have emerged to assist individuals facing various challenges in their lives. These applications range from tools designed to aid in navigation and obstacle detection [1, 2] to those identifying currency bills, and objects, and providing reading assistance or online support [3, 4].

While these applications offer valuable assistance in daily life, they primarily focus on addressing physical challenges and navigating environments. However, there is still a need for solutions that provide detailed information about the nature and appearance of obstacles. Additionally, there is a notable gap in addressing entertainment-related needs, particularly in enabling access to and understanding of theater scenes for individuals with specific challenges.

This work draws inspiration from *MindsEye Radio* [5], a platform that offers audio translations of visual events, videos, and news throughout the day. It serves as a valuable resource for individuals with various visual disabilities, providing timely access to information. Their oral descriptions are detailed, from describing shapes and colors to actions and movements.

The proposed solution aims to generate descriptive captions for RGB-D images from the newly developed TS-RGBD dataset, focusing on Theatre Scenes. The solution involves three main steps: Panoptic Segmentation, Segment Captioning, and Determining Regions' Direction.

Initially, the RGB image undergoes panoptic segmentation, which identifies and delineates different regions within the image. Each pixel is assigned a unique identifier and a class code, comprehensively representing all regions, including the background and non-salient objects. From these segments, bounding boxes are extracted for further processing. Following segmentation, the solution employs a modified dense captioning architecture to generate textual descriptions for each identified segment. This model includes a feature extractor, a multi-scale region of interest alignment, an intermediate fully connected layer, and an LSTM language network. The dense captioning architecture is adapted to focus strictly on panoptic segments, ensuring detailed scene understanding without redundancies or omissions.

✉ Khadidja Delloul
kdelloul@usthb.dz

Slimane Larabi
slarabi@usthb.dz

[1] RIIMA Laboratory, Computer Science Faculty, USTHB, BP 32 El Alia, 16111 Algiers, Algeria

The final component involves using depth information from the RGB-D images to determine the spatial relationship of each segment with the user. By calculating the centroid of each segment and transforming it into real-world coordinates, the model determines the direction (left, right, front) based on computed angles from the point cloud. This integration of visual and spatial information results in comprehensive captions that enhance the understanding of Theatre Scenes.

To evaluate the effectiveness of our solution, we apply it to RGB-D images from our newly developed dataset, TS-RGBD [6]. This dataset includes RGB images and depth maps collected using the Microsoft Kinect sensor, introducing a novel application for image captioning in the context of theater scenes.

The key novelty of our research lies in developing an intelligent image captioning system that generates detailed, egocentric descriptions for each image segment using RGB-D data, facilitated by our newly introduced TS-RGBD dataset, specifically designed for theater scenes.

This paper is organized into the following sections. In the next section, we outline the context in which the research was conducted. In section 3 we present existing literature on the research problem and its limits. In Sect. 4, we present the solution proposed in this study along with an exploration of various models and the developed algorithm during our research. Section 5 is devoted to experiments and results. We provide a detailed account of the research implementation, encompassing the training of models and a thorough analysis of the obtained results. In the final section, we present a summary of key findings, limitations, and suggestions for future research.

2 Context

Visual information is vital for understanding the world around us, yet individuals who are blind or visually impaired face significant challenges in accessing and comprehending visual content. Image captioning technology offers a promising solution by generating verbal descriptions of images. However, existing approaches have limitations, such as scalability and variability in quality.

In this study, we aim to address these challenges by developing an improved image captioning system tailored for blind and visually impaired users. Our research focuses on enhancing accessibility, usability, and scalability to empower individuals with visual impairments to understand visual content better. Through our work, we seek to contribute to the advancement of inclusive technology and equal access to information for all individuals.

Firstly, we explore segment captioning, a novel technique that generates descriptions for specific regions within an image, providing more detailed and contextually relevant information, without drowning the users in dense captions.

Secondly, we investigate egocentric captioning, which focuses on describing the user's perspective within a scene, offering a more personalized and immersive experience for blind and visually impaired individuals.

Lastly, we introduce a novel RGBD dataset of theatre scenes, specifically curated to facilitate the development and evaluation of image captioning systems for individuals with visual impairments. This dataset provides rich visual and depth information, enabling researchers to train and test algorithms in realistic and diverse environments.

Through these contributions, we strive to advance the field of image captioning for accessibility and promote equal access to visual information for all individuals.

3 Related works

Image captioning serves as a bridge between computer vision and natural language processing, wherein an intelligent system takes an image as input and generates a corresponding text description that accurately portrays the content depicted in the image. Image captioning models can output one sentence for a given image, a paragraph, or multiple captions of each region of interest.

Single-sentence captioning models can be classified according to their architectures; image analyzing models, attention mechanisms and transformers, CNN-LSTM networks, or even GANs [7]. These solutions, whether they rely on transformers and attention mechanisms [8–12], encoder-decoder architectures [13], or scene graphs as presented in [14], in which learning is supervised, or relying on beam search analysis or gated recurrent units (GRU) units, in which learning is unsupervised [15, 16], or even for weakly supervised learning [17, 18], generate one single sentence for each input image. Such models are trained on RGB image datasets [19, 20].

One sentence is insufficient to describe an image's semantic and contextual content. This motivated researchers to improve single-sentence image captioning to output a describing paragraph for an input image. Solutions for paragraph captioning are based on single-sentence captioning to generate a set of sentences that will be combined to form a coherent paragraph [7]. Such models are built upon recurrent network or encoder-decoder architectures [21–24], and encoder with attention [25]. One suggested method [26] generates paragraphs describing the positional relationships between objects. However, these relationships are vague, emphasizing actions between objects or regions of interest (such as; holding, being held, walking on, etc.). The sentences lack specific details about the described region of interest in the image, giving only a general description of shape or color, and fail to specify the positions of these objects relative to the user.

Dense captioning is the task of textually describing multiple detected regions of interest within an input image by generally starting with the task of objects or regions of interest detection [27]. One of the most relevant works is the DenseCap model proposed in [28]. It uses a Region Proposal Network (RPN) to detect k regions of interest within an image, then each region will be given a caption by the Long Short-Term Memory (LSTM) language network. The DenseCap model was trained on RGB images of the Visual Genome dataset [29], a set of MS-COCO [30], and Flickr indoor or outdoor images.

There are gaps and limitations within the cited literature. We will next shed light on the primary limitations of the existing research, providing critical insights that pave the way for the novel contributions and advancements put forth in our study.

Regarding Image Captions, models that generate one sentence to describe an input image tend to focus on mobile objects, and salient regions, and completely ignore any background information and static objects, which makes the sentence less informative about what is present in the scene. Background description is important as well when it comes to enriching the extracted information from the image.

While paragraph captions provide more information about an input image than sentence captions, paragraph generation models often focus on objects on movement, and rely on background to only name the environment.

While dense captioning provides detailed captions for each region of interest in an input image, it introduces challenges when it comes to the practical use of the framework. Specifically, when generating captions for multiple regions k in an image, the presence of overlapping bounding boxes can lead to redundancy and duplication of information.

Providing k as an input would select regions among generated ones randomly and will not eliminate unwanted regions.

Regarding Datasets, our solution focuses on applying image captioning techniques to RGB-D images. Although these images are not available in mainstream computer vision datasets that typically include generic images or scans, we aim to adapt and tailor the image captioning approach to cater specifically to the unique context of segment captioning. In addition, available datasets focus on annotating bounding boxes and segments with labels rather than phrases, making it impossible to apply dense captioning on panoptic segments.

Furthermore, it’s worth noting that the majority of available datasets primarily consist of RGB images, and a significant portion of image captioning models are exclusively trained on RGB images. This limitation poses a challenge for our specific requirements, as we aim to work with RGB-D data and point cloud information, which offers a richer and more comprehensive understanding of the visual environment.

Our solution will be applied to our newly collected and annotated dataset “TS-RGBD”: a dataset of RGB-D images with sentences for each panoptic segment in Theatre Scenes [6].

To summarize, notable limitations persist in the current state of image captioning research. Existing image captioning models, whether generating single sentences, paragraphs, or multiple region-specific captions, face several limitations. Single-sentence models often neglect background and static objects, focusing only on salient and mobile objects, thus providing incomplete scene descriptions. Paragraph captioning improves information content but still emphasizes moving objects and offers only general background descriptions. Dense captioning, which aims to describe multiple regions, suffers from redundancy and information duplication due to overlapping bounding boxes. Furthermore, current models and datasets predominantly use RGB images, lacking depth information and often providing only labels for segments instead of detailed phrases, making them unsuitable for our aim of using RGB-D data to capture a richer visual context.

Based on this analysis, our work makes the following contributions:

- We propose a novel approach to image captioning known as segment captioning.
- Our generated captions are egocentric, providing directional information for each region regarding the user. Captions combine textual descriptions with egocentric positions.
- Our solution is applied to RGB-D images from our TS-RGBD dataset, pioneering a new captioning field: Theatre Scenes.

4 Model

Our goal is to design an architecture that is capable of generating captions for each segment present in the image, and then enrich those captions with the directions.

Our solution (see Fig. 1) consists of three main steps: *(i) Panoptic Segmentation, (ii) Segment Captioning, (iii) Determining Regions Direction.*

The RGB image serves as the input to the panoptic segmentation module. This module analyses the image and generates segmentation results, from which we would get bounding boxes that delineate different regions within the image. The segment captioning module takes the generated bounding boxes as input, as well as the image, and generates descriptive captions for each region. These captions provide detailed descriptions of the identified segments. Additionally, our solution incorporates the computation of directions

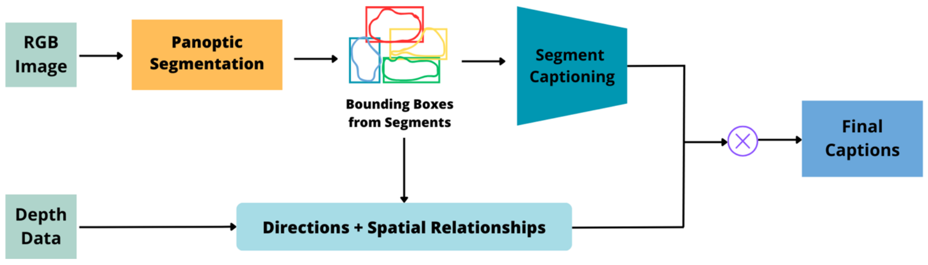


Fig. 1 General solution architecture

that uses the depth map and the boxes as input. We process the depth information from each bounding box to determine the spatial relationship with the user. The output captions from the segment captioning module are combined with the results from the directions computation. This combination results in comprehensive captions that encompass both the detailed segment descriptions and the contextual information derived from the depth map.

4.1 Model architecture

4.1.1 Segmentation

Our goal is first to generate a descriptive caption for each region present in the image. Instance segmentation provides segments identifying and delineating individual objects within an image with precise boundaries, so it eliminates background and is only interested in salient objects. Semantic Segmentation labels each pixel in an image with its corresponding class, and so multiple objects or regions belong to the same segment.

Both instance segmentation and semantic segmentation are not suitable solutions for our problem. Therefore, we have opted for panoptic segmentation, where each pixel is assigned a specific identifier unique to each instance, and a code corresponding to its class. With panoptic segmentation, we have as output every region on the image, be it salient, background, or non-salient objects. Regions are not divided into a set of objects but represented as a whole by segments. Figure 2 illustrates the distinctions among instance, semantic, and panoptic segmentation applied to a single image. In the case of instance segmentation, only a limited number of objects were successfully detected. Moving to semantic segmentation, both individuals are filled in pink, both plants are marked in green, and both pots are highlighted in purple, resulting in the grouping of identical objects into unified regions with the same identifiers.

However, the desired outcome is achieved with panoptic segmentation. Here, each region is distinctly highlighted with a unique color, ensuring independence from similar objects and assigning a unique identifier to each segment. Then for each region, a bounding box was extracted as shown in Fig. 3.

4.1.2 Segment captioning

To generate textual descriptions for each segment, we proposed a modified DenseCap architecture. The Densecap model consists of these following parts [28] illustrated in Fig. 4:

- *The Feature Extractor:* a pre-trained model for features extraction (VGG-16, Resnet-50, FPN with Resnet-50...etc), which extracts features map from an input image.
- *RPN:* a region proposal network that would detect multiple regions of interest per image, it returns bounding boxes and scores for each, hence the word "Dense".
- *RoI Align:* "Region of Interest Alignment" to get features of each bounding box detected by the RPN from the features map of the input image.
- *Intermediate Layer:* that transforms each region features extracted by the RoI Align into a vector of a dimension D.
- *Language Network (LSTM):* that takes feature vectors as input and generates a sentence for each.

Modified DenseCap Architecture

We aim to strictly describe panoptic segments, prioritizing scene understanding over decomposing each object or regrouping regions, which aligns with common sense. To achieve that, we had to modify DenseCap architecture to limit the descriptions to panoptic segments. The RPN and RoI Align serve as the localization layer for the DenseCap model. We adapted this layer by splitting it into distinct components, where the RPN was eliminated and the RoI alignment was replaced by a multi-scale alignment. To get descriptions for regions of interest, we feed the model with the image along with the ground truth bounding boxes of panoptic segments, as illustrated in Fig. 5. By removing the RPN layer, the model is relieved from the task of detecting multiple regions of interest (k regions), thereby helping to prevent redundancies or omissions.

Our model architecture is now composed of a Feature Extractor, RoI Multi-Scale Align, Intermediary Layer (Recognition Network), and the LSTM captionner. Each component will be detailed in the following.

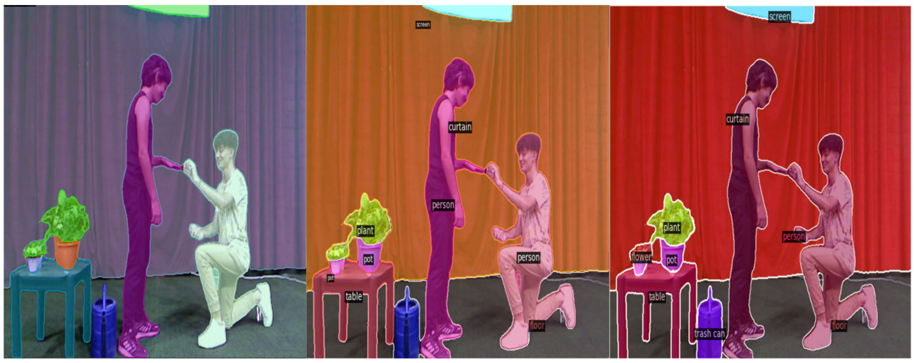


Fig. 2 From left to right: instance, semantic and panoptic segmentation using Detectron2 [31]

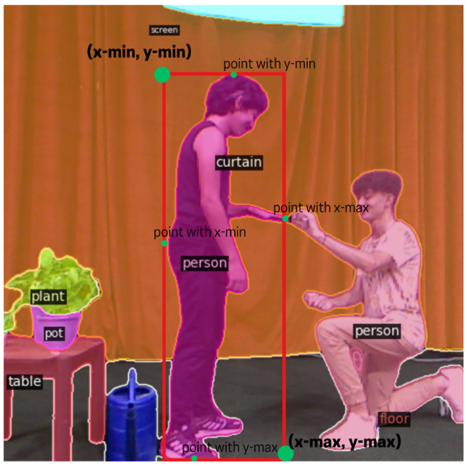


Fig. 3 A bounding box from a panoptic segment

a. Feature Extractor

A convolutional neural network that is designed to capture hierarchical patterns and important features from the input images. These layers utilize filters to convolve over the input image, detecting edges, textures, and other relevant information. In our proposed solution, we experimented with two feature extractors: first with *VGG-16*, followed by *ResNet-50*.

VGG-16 [32] is a CNN architecture for classification, with 13 layers of 3×3 convolutions interspersed with 5 layers of 2×2 max pooling. The final fully connected layer is removed, to get only feature maps and discard classification results that are not needed in this context. So for an input image of shape $3 \times h \times w$, we get an output of $c \times h' \times w'$ where $c = 512$ is the number of feature maps, $w' = \frac{w}{16}$ and $h' = \frac{h}{16}$, Fig. 6 shows an illustration.

Resnet-50 [33] on the other hand is a CNN more suitable for object detection tasks. It employs residual learning, using shortcut connections to skip layers during training, which helps mitigate the vanishing gradient problem. The network consists of convolutional layers, batch normalization, and identity blocks. An identity block comprises two 3×3 convolutional layers with batch normalization and rectified linear unit $ReLU$ activation functions, the fully connected layer is removed to keep only the feature maps, Fig. 7.

Feature Extraction Head 1

Initially, we proposed a model that would take as input an image with bounding boxes of ground truth segments. Pixels that belong to the box but that are outside of the segment would be turned to black to cancel them when training the VGG-16 feature extractor as illustrated in Fig. 8. Pixels that are canceled do not affect the context when extracting feature maps, so passing such data through a features extractor is equivalent to extracting features of a segment.

To determine if a pixel belongs to the polygonal envelope defined by the panoptic segment, we used a linear problem-solving approach. Specifically, we examined whether a given point could be expressed as a convex combination of points derived from the panoptic segmentation. In our methodology, all pixels sharing the same identifier in the panoptic segmentation were considered to form a set of points, which we subsequently used to test the convex hull with points from the segment bounding box. Each pixel from the bounding box, if it belongs in the polygonal envelope, it would be kept, if not, it would be put to 0. Pixels put to 0 are canceled in the CNN features extractor. The feature extractor used in this head is the VGG-16 model. Each box would be passed through VGG-16, to extract its map of features, as illustrated in Fig. 9.

When applying VGG-16 to small images of variable sizes, we observed that the resulting feature maps varied in size. We needed to skip layers as these maps became exceedingly small after pooling. Moreover, the presence of multiple feature maps imposed a significant burden on memory, resulting in slower execution times. This circumstance prompted us to explore passing the entire image through the feature extractor at once, rather than dividing it into smaller frames or panoptic segments. This led to the introduction of the second head of feature extraction.

Feature Extraction Head 2

Using VGG-16 on smaller boxes extracted from the entire image results in using VGG-16 on variable sizes images. The model encountered challenges reaching the last layer consis-

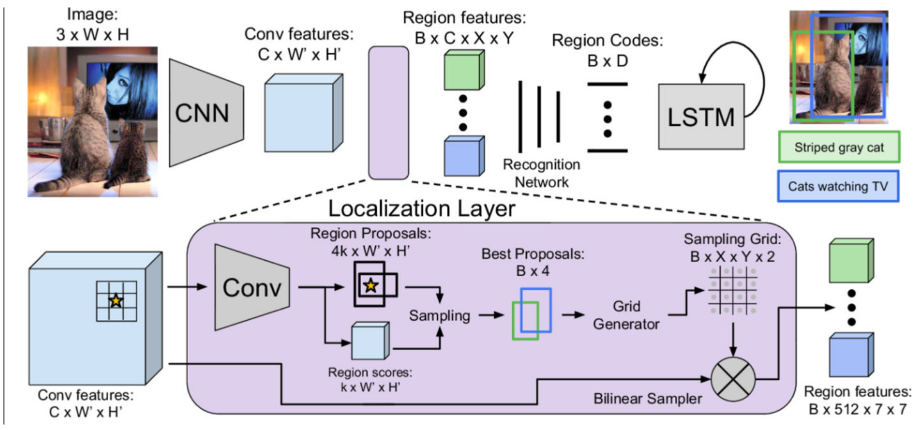


Fig. 4 DenseCap model architecture [28]

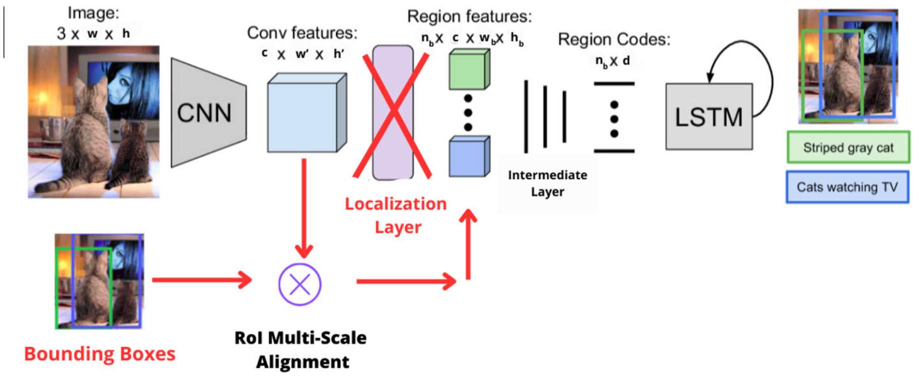


Fig. 5 The modified DenseCap model architecture. The localization layer was eliminated to feed the model with the bounding boxes extracted from panoptic segments beforehand

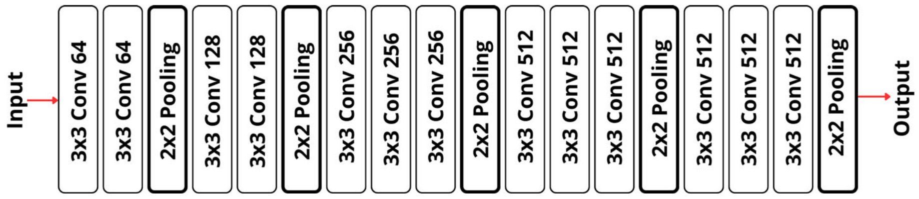


Fig. 6 The VGG-16 architecture stripped of the fully connected layer

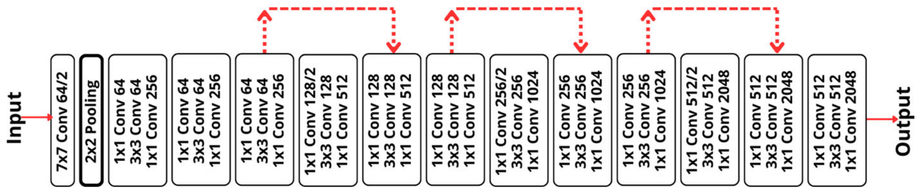

Input
7x7 Conv 64/2
2x2 Pooling
1x1 Conv 64 3x3 Conv 64 1x1 Conv 256
1x1 Conv 64 3x3 Conv 64 1x1 Conv 256
1x1 Conv 64 3x3 Conv 64 1x1 Conv 256
1x1 Conv 128/2 3x3 Conv 128 1x1 Conv 512
1x1 Conv 128 3x3 Conv 128 1x1 Conv 512
1x1 Conv 128 3x3 Conv 128 1x1 Conv 512
1x1 Conv 256/2 3x3 Conv 256 1x1 Conv 1024
1x1 Conv 256 3x3 Conv 256 1x1 Conv 1024
1x1 Conv 256 3x3 Conv 256 1x1 Conv 1024
1x1 Conv 512/2 3x3 Conv 512 1x1 Conv 2048
1x1 Conv 512 3x3 Conv 512 1x1 Conv 2048
1x1 Conv 512 3x3 Conv 512 1x1 Conv 2048
Output

Fig. 7 The Resnet-50 architecture, without the fully connected layer

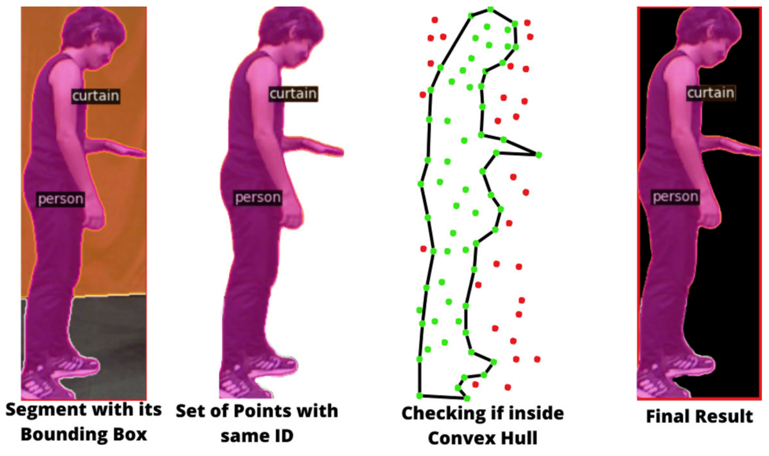

curtain
person
Segment with its Bounding Box
Set of Points with same ID
Checking if inside Convex Hull
Final Result

Fig. 8 Elimination of pixels that do not belong to the segment

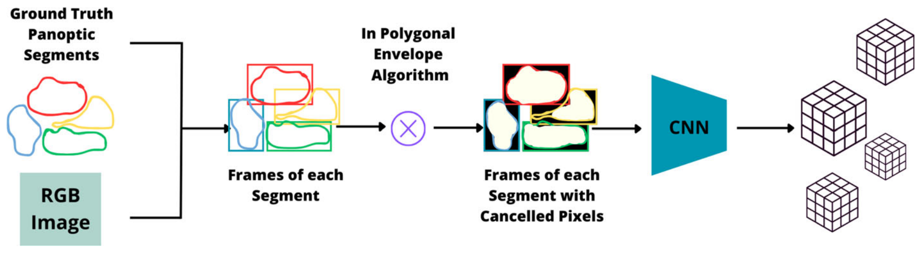

Ground Truth Panoptic Segments
RGB Image
Frames of each Segment
In Polygonal Envelope Algorithm
Frames of each Segment with Cancelled Pixels
CNN

Fig. 9 Illustration for the detailed architecture of the first feature extractor head

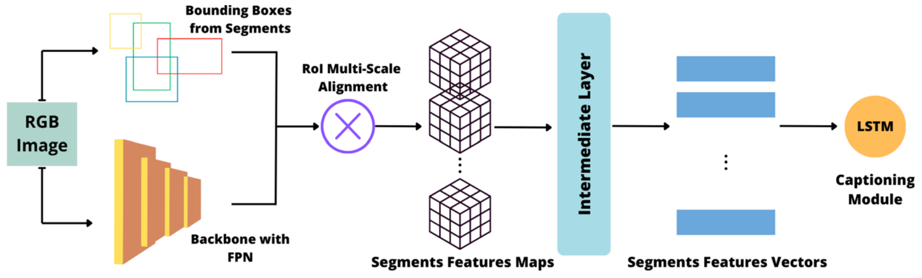


Fig. 10 Illustration for the architecture of the second head: a modified version of DenseCap

tently due to pooling operations reducing the size excessively. The process was also considerably slow due to computing the convex hull for each segment. Consequently, we opted to implement the feature extractor on the entire image, addressing both the size variation issue and improving computational efficiency. Figure 10 presents the architecture of the proposed solution with the second head.

VGG-16 is more suitable for straightforward image classification so features were extracted using a pertained Feature Pyramid Network (FPN) with Resnet-50 backbone to get multi-scale feature maps for each image.

FPN with a ResNet-50 backbone is a neural network architecture designed for object detection tasks. It leverages the hierarchical feature maps generated by the ResNet-50 backbone at different stages to create a feature pyramid. This pyramid allows the model to capture multi-scale representations of the input image. FPN contributes to improve accuracy of models by integrating high-level semantic information with detailed spatial features. It generates 4 maps with 256 features of the whole image. The output would have the size 4×256. Because we still need feature maps for each bounding box, a region of interest alignment was needed for the extraction of maps for each box from the output of the FPN, and to do so, we used RoI Multi-Scale Alignment.

b. RoI Multi-Scale Alignment

To retrieve the bounding boxes from the feature maps of images, we used the multi-scale region of interest alignment that infers the scale of the pooling via the heuristics specified in [34] using the following equation:

$$k = \lfloor k_0 + log_2(\sqrt{w_b h_b}/224) \rfloor$$

Where k is the level of the feature pyramid, k_0 is the target level into which the segment bounding box should be mapped, which is equal to 4 in our case (FPN with the output of 4 feature maps). w_b and h_b are the width and height of the box, and 244 is the canonical ImageNet pre-training size.

Each segment bounding box of size $w_b \times h_b$ is projected onto the 4 grids of convolutional features, and the equation is used to infer the scale and get features of size w_b' $\times h_b$' for each segment as illustrated in Fig. 11. This process ensures that spatial information is preserved and contributes to the module's effectiveness in tasks like object detection and segmentation.

c. Intermediate Layer

The intermediate layer is a fully connected layer that transforms $n_b \times c \times w'_b \times w'_b$ features into a vector of d features, where n_b is the number of boxes, c is the number of extracted feature maps, and d is the length of the output vector. The features from each region are flattened into a vector and passed through two fully connected layers, each using rectified linear units and regularized using Dropout. It helps to represent each region as a compact vector of $d = 4096$ features, instead of a multidimensional matrix.

d. LSTM Network for Captioning

The DenseCap LSTM model [28] for captioning was used, with one recurrent layer that takes as input the $n_b \times d$ vectors and outputs n_b sentences, where n_b is the number of boxes per image and d is the size of the features vector output by the intermediate layer.

The model takes a sequence of feature vectors from the intermediary layer, ground truth sentences, and a list of potential tokens as input. In the preprocessing step, ground truth sentences are encoded with a special START token ($< bos >$) at the beginning and an END token ($< eos >$) at the end. Each word from the vocabulary is converted into a corresponding token code. This encoding mechanism ensures that the model generates sentences with proper sentence boundaries, contributing to the coherent structure of the output. These tokens are passed through an embedding layer

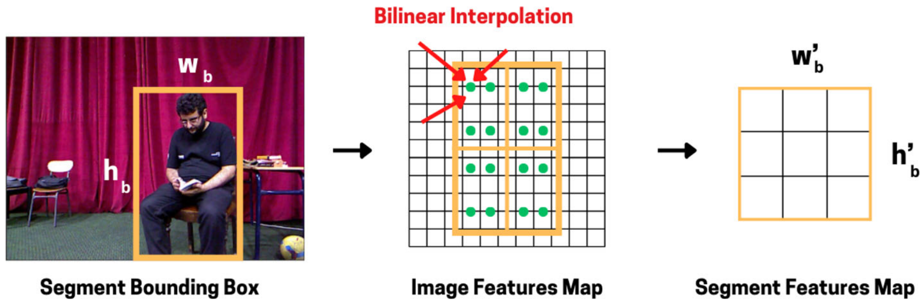


Fig. 11 Illustration for the region of interest multi-scale alignment and how it returns feature maps for each bounding box from the FPN resulting feature map of the image. The arrows indicate the movement or transformation of points from the RoI space to the feature map space, accounting for any scaling, translation, or rotation necessary to align the RoI with the feature map. Bilinear interpolation is used to interpolate the feature values at these transformed points on the feature map, ensuring accurate feature extraction from the RoI

of size equal to the size of our dataset vocabulary including the START and END tokens.

For each sentence of size N, the LSTM network is fed with a sequence of tokens $t_1, t_2, ..., t_{N+2}$ of size $N + 2$ where t_1 becomes the START token and t_{N+2} is the END token. The LSTM computes hidden states h_t using the following recurrence formula where o_t is the output gate, c_t is the cell state, and $\odot$ is the Hadamard product.

$$h_t = o_t \odot tanh(c_t)$$

Outputs are passed through a linear layer with the ReLU activation function to generate the sequence of tokens for each generated sentence.

4.1.3 Loss function, training, and optimization

During training our ground truth consists of segment boxes and textual descriptions.

Since the model does not predict boxes but gets coordinates as inputs, the only new data to be predicted are sentences for each box. To evaluate the model, we used the cross-entropy loss function at every time step of the language model.

For the cross-entropy loss computation, we need the predicted sentences and the target sentences as inputs. This criterion computes the cross entropy loss between input tokens and targets. It computes the loss using this formula:

$$loss(input, target) = -log\left(\frac{e^{input[target]}}{\sum_j e^{input[j]}}\right)$$

We train the full model end-to-end in a single step of optimization. We initialize the CNN with weights pre-trained on Visual Genome and our dataset, we used Adam Optimizer, torch Grad Scaler, and a learning rate of 0.001. Training is done on data with a batch size of 4 with 100 epochs.

4.2 Determining region positional relationship with user

Our solution includes determining the direction in which the user can face each described region in the given images. Incorporating depth information is crucial because relying solely on the 2D image, where pixels of (x_i, y_i) coordinates depict the projections of real-world points onto the image plane, leads to inaccuracies in directional analysis. Our proposed solution uses depth information from images captured using the Microsoft Kinect v1 sensor for RGB images and depth maps. Its focal length value f is a built-in value.

Before delving into the steps we followed to retrieve orientations, it is important to establish the mathematical framework. We provide a brief description of the coordinate system. The world coordinates frame $(OXYZ)$ is defined as follows:

- The origin O (located on the image plane center) coincides with the impact point of the optical axis with the image plane.
- The X axis represents the horizontal dimension.
- The Y axis represents the vertical dimension.
- The Z axis represents the depth dimension and coincides with the optical axis.

The image coordinates frame $(O'uv)$ is defined as follows:

- The point O' is the top left pixel.
- The horizontal line represents the u axis.
- The vertical line represents the v axis.

In the following, each point $p(x, y, z)$ from the real world is projected as $p_i(u_i, v_i)$ onto the image plane.

To get directions, we compute the x-coordinate of the centroid P of a region of interest and infer the angle θ (see Fig. 12).

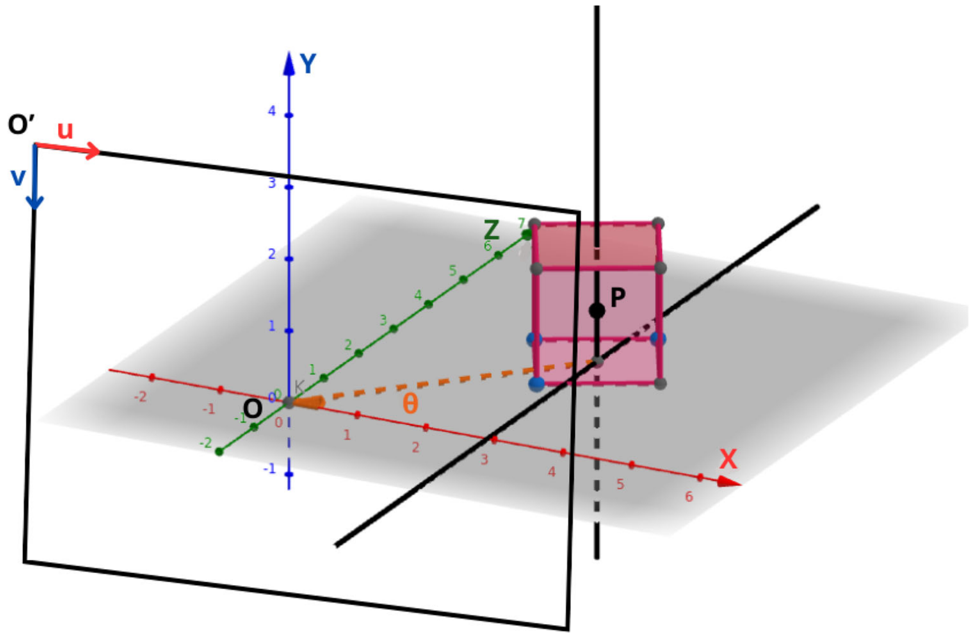


Fig. 12 $(OXYZ)$is the world coordinates frame. $P(x, y, z)$ is the centroid of the bounding box of a region of interest

Our proposed solution follows the steps below:

1. Calculate the centroid $P(x, y, z)$ of the pixels of each region, by taking the mean values of the coordinates (u_i, v_i) and depth values from the bounding boxes pixels. The number of pixels per box is $n = w_b \times h_b$, where w_b and h_b are the width and height of the bounding box, respectively;
2. Transform the values of the centroids to point-cloud coordinates using the built-in focal length parameter f, which determines the field of view and the camera's magnification. Since the origin of the real-world coordinates is at the center of the image, and the pixel coordinates have their origin at the top left corner, a translation operator c_x is required as well (see Fig. 13). The c_x value is obtained after calibrating the camera to get the camera's intrinsic parameters. The transformation of the x-coordinate to point-cloud coordinate follows the equation:

$$x_w = (x - c_x) \times \frac{z}{f} \tag{1}$$

3. Use the spherical coordinates system to compute the angle that the orthogonal projection of the region's centroid into the (OXZ) plane forms with the (X) axis:

$$\theta = arctan(\frac{z}{x_w}) \tag{2}$$

   the given result is the angle $\theta \in [-\frac{\pi}{2}, \frac{\pi}{2}]$, as shown in Fig. 14;
4. Use θ to determine to which field each region belongs, as illustrated in Fig. 15.

Finally, we would have three sets of segments with captions for each field: Right, Left, and Front with 30° of upfront focus.

5 Experiments and results

5.1 Working environment and used datasets

On a machine with AMD Ryzen 5700X 8 Core 3,4GHz CPU, 16 Go RAM, 2060 RTX Nvidia 8Go GPU, and Windows 11 as the operating system, training was performed on GPU with version 3.9.7 of PyTorch Libraries.

The PyTorch version of DenseCap provided in GitHub [35] served as a foundational framework for both our solutions.

We used our newly developed dataset of RGB-D Theatre scenes, the TS-RGBD dataset, that we collected using Microsoft Kinect v1 [36], with 120 images of size (640, 480), a total of 1183 sentences, 109 different words.

We annotated our data manually using the "LabelMe" open-access framework [37]. We had to manually draw polygonal envelopes for each region, and to replace labels with phrases. For each image, it generated JSON files that we preprocessed to build datasets with a dictionary for tokens, and bounding boxes for each polygonal envelope. Figure 16 shows a summary.

5.2 Data preprocessing

We split data into two sets, 50% for training and 50% for validation, with random data distribution.

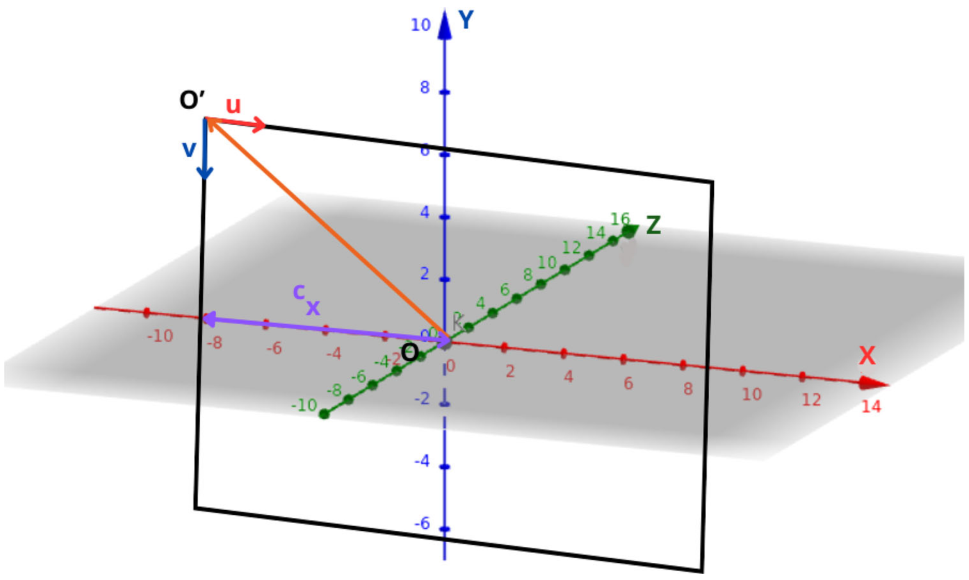


Fig. 13 Translation of the coordinates frame from O' to O origin

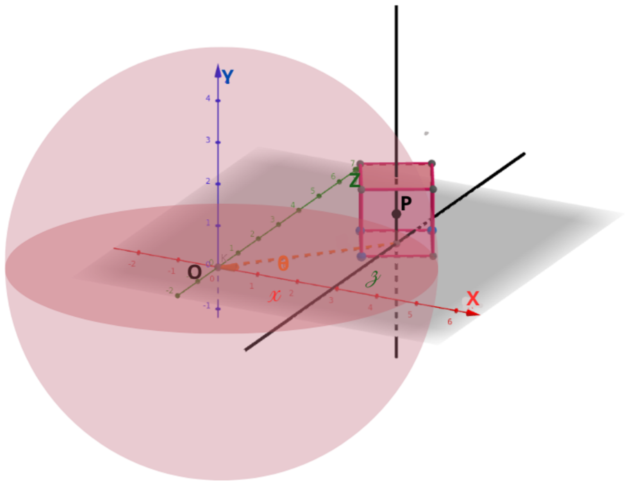


Fig. 14 To compute the correct angle θ we built the point cloud, then we used the spherical coordinates to get the tangent of the angle formed by an object's centroid and the (X) axis

The Microsoft Kinect v1 was used to capture depth values, resulting in depth maps that exhibit non-smooth characteristics. These maps may contain multiple 0 values and 4000 values (millimeters) that are considered noise, which can potentially affect the accuracy of mean, maximum, and minimum values derived from them.

To eliminate this problem, the mean filter was applied to depth maps to smooth such values. As for sentence coding, we used a token $< unk >$ to represent words that appear less than 2 times. All sentences with more than 10 words were eliminated. Each word is coded by an integer, and each sentence starts with token $< bos >$ and ends with token $< eos >$.

5.3 Panoptic segmentation

As for panoptic segmentation, numerous recent models are available within the open-access community.

We were unable to use the most recent real-time panoptic segmentation model proposed in [38] due to compatibility issues with our working environment and the lack of support for our platform. We used OneFormer [39], a recent model that gave the best qualitative results for our dataset; Fig. 17. It relies mainly on transformer architecture without resorting to RPN models.

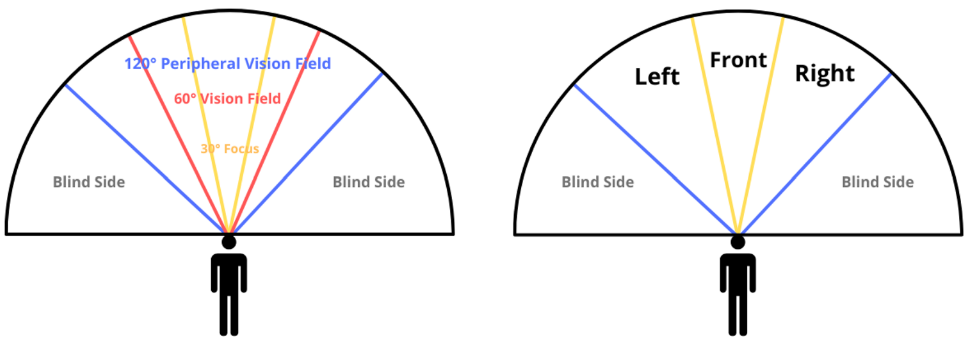
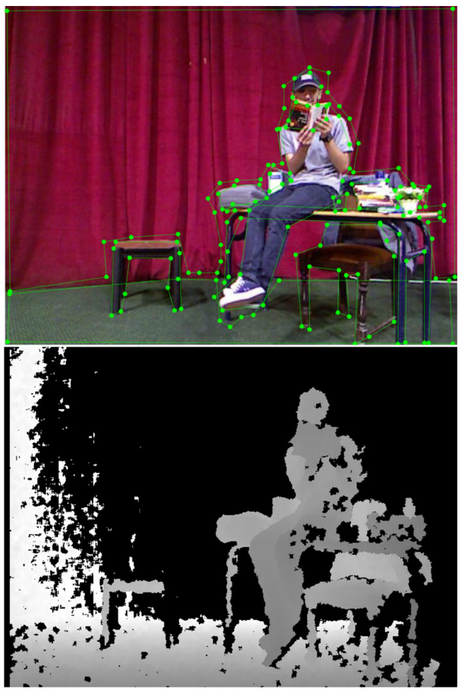


Fig. 15 Fields of vision of humans, 30° focus upfront, 45° for left and 45° for the right



ground: a green rug
background: a red theatre curtain
Left: a brown nesting table
Right: a wooden chair, a stack of books, a white pot of plants
Front: a man wearing a dark blue hat and blue sneakers, a blue files box, a purple water cup, a wooden table with blue feet, a pocket book, a gray bag

Fig. 16 Example of collected and annotated data. Polygonal envelopes were defined by manually drawing points that contour a region of interest (green points) in the above image. The image below is the corresponding depth frame

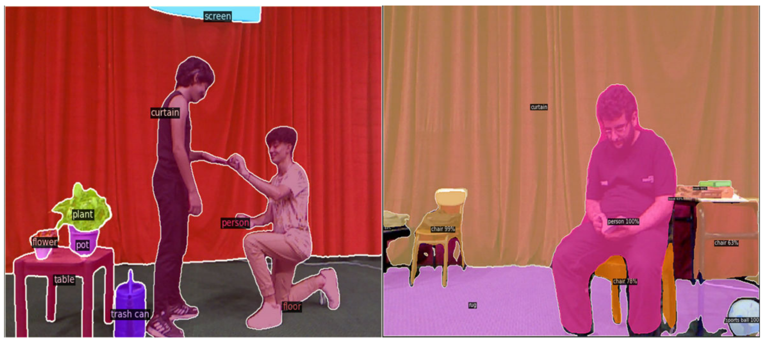


Fig. 17 Examples of panoptic segmentation on theatre images using the OneFromer Model

When testing the whole framework, we don't use ground truth boxes, so for a segmented image, we would get the bounding boxes of each segment by creating for each segment a binary image where segment pixels are equal to one. Since segments are inferred by a model, there could be some other pixels outside of a segment with the same id, so we contoured all white regions and took the largest one among them. For the largest contoured area, we extract the bounding box.

5.4 Region captioning results

We conducted experiments on our proposed architectures with different parameters and hyperparameters.

In the following section, we will showcase and analyze the various results we have achieved in our study.

Table 1 Results on different metrics, for our models, the higher the better. Head-1 is the feature extractor with VGG-16. Head-2 is the FPN with Resnet-50 backbone. Head-2 gave better results

Metrics	BLEU@1	BLEU@2	BLEU@3	BLEU@4	ROUGE	CIDEr
Training from Scratch						
Head-1	0.03	0.01	0.012	0.001	0.09	0.015
Head-2	0.17	0.11	0.09	0.07	0.22	0.23
Transfer learning						
Head-2	0.22	0.14	0.11	0.08	0.56	0.25
Fine tuning						
Head-2	**0.39**	**0.34**	**0.31**	**0.29**	**3.25**	**0.47**

The bold values indicate the better results

5.4.1 Parameters and evaluation metrics

The training was conducted on different instances of our dataset. First, we considered sentences no longer than 15 words, then sentences with a maximum length of 10 words. The latter gave better results. For each sentence length, we considered tokens that appeared at least 2 times and those with at least one appearance. The first gave better results. Evaluation metrics for the language are BLEU, ROUGE, and CIDEr where for each, the higher the value the better.

5.5 Training

Training on our dataset led to an important overfitting due to the lack of data. The LSTM model could generate accurate captions but would also add unnecessary tokens to each phrase due to overfitting.

Although the loss curves of both models (head 1 and head 2) were decreased, the evaluation curves stabilised after several epochs. The Table 1 shows results obtained from both heads on our images after 100 epochs.

The quantitative results showed that the feature extractor plays an important role in the accuracy of captions. When using a feature extractor on the region only, it slows the process and decreases accuracy. In addition, the FPN feature extractor gives better results than the VGG-16. Due to the evident problem of overfitting, we decided to explore transfer learning as a potential solution.

After considering the advantages of the second head, we opted for a pre-trained Feature Pyramid Network (FPN) with a pre-trained ResNet-50 backbone. We selectively blocked back-propagation to ensure the desired updates, allowing only the intermediary layer and the LSTM to be updated. Results are shown in Table 1. Figure 18 shows the obtained loss curve and Fig. 19 the accuracy using the BLEU metric.

Although the accuracy of the results improved noticeably, the overall shape of the learning curve remained unchanged. Both training approaches yielded stabilised results, indicating that the LSTM is stuck in a repetitive loop and failing to converge. The oscillating shape of the evaluation curve

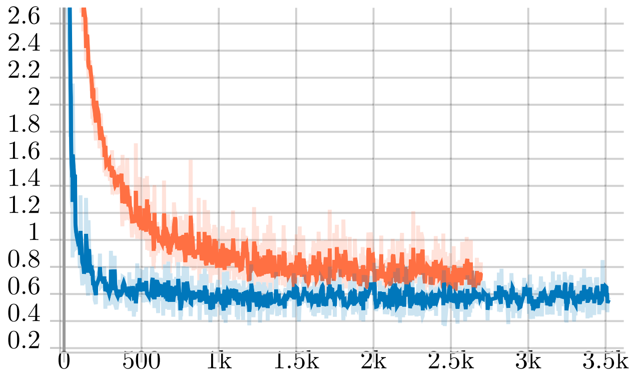


Fig. 18 Loss curves from training our model: orange is the loss curve of training from scratch, blue is the loss curve of the transfer learning

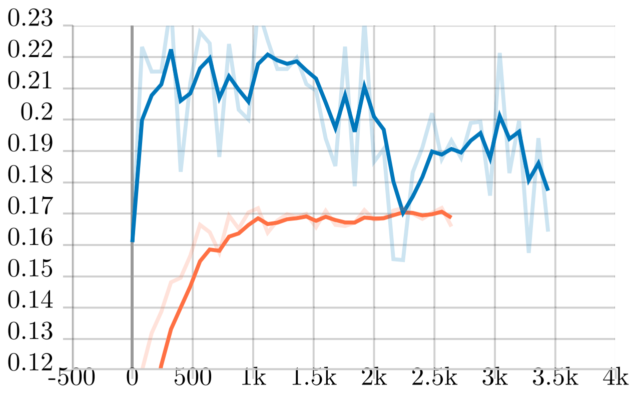


Fig. 19 BLEU metric curves form training our model: orange is the BLEU results curve of training from scratch, and light blue is for transfer learning

confirms that. These results led to our training phase's final step, Fine Tuning.

Before developing our solution, we had already trained DenseCap on our data. So we retrieved the weights obtained from the training, and we fed our model with FPN, Resnet-50 Backbone, Intermediary Layer, and LSTM weights as starting points. After only 10 epochs, the LSTM converged successfully and the accuracy reached the results shown in Table 1.

Figure 20 shows how 10 epochs were enough to get a smaller loss value with a better shape of the loss curve. Fig-

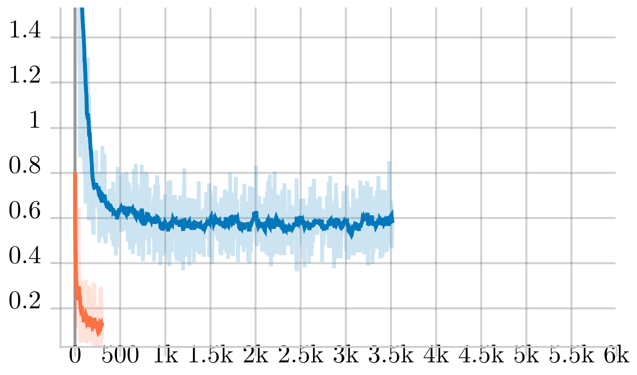


Fig. 20 Loss curves from training our model: blue is the loss curve of the transfer learning, orange is the loss curve of the fine-tuning after only 10 epochs

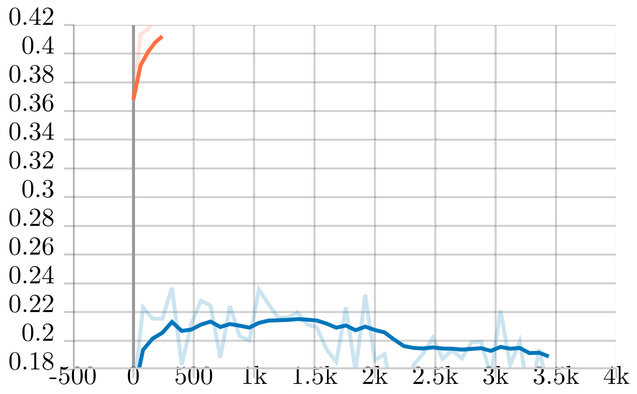


Fig. 21 BLEU metric curves from training our model: blue is the loss curve of the transfer learning, orange is the loss curve of the fine-tuning after only 10 epochs

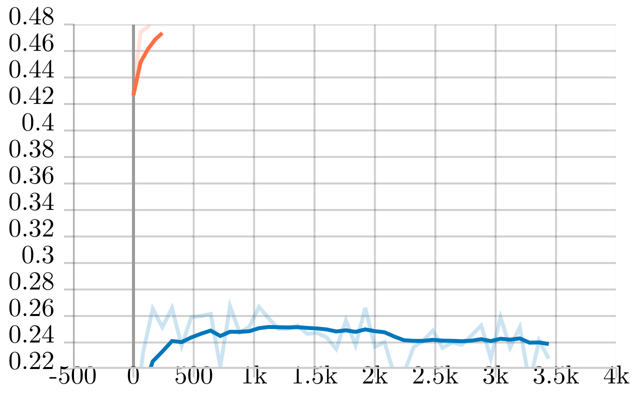


Fig. 22 ROUGE metric curves form training our model: blue is the loss curve of the transfer learning, orange is the loss curve of the fine-tuning after only 10 epochs

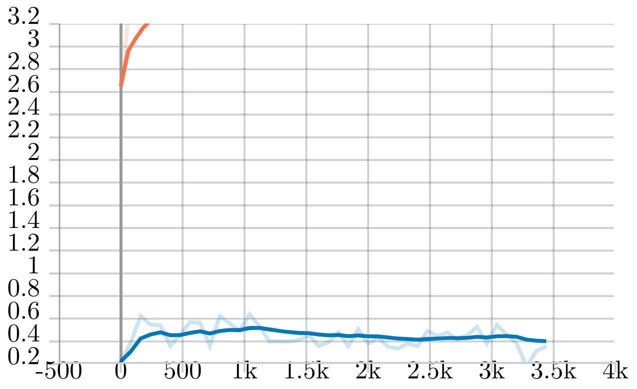


Fig. 23 CIDEr metric curves form training our model: blue is the loss curve of the transfer learning, orange is the loss curve of the fine-tuning after only 10 epochs

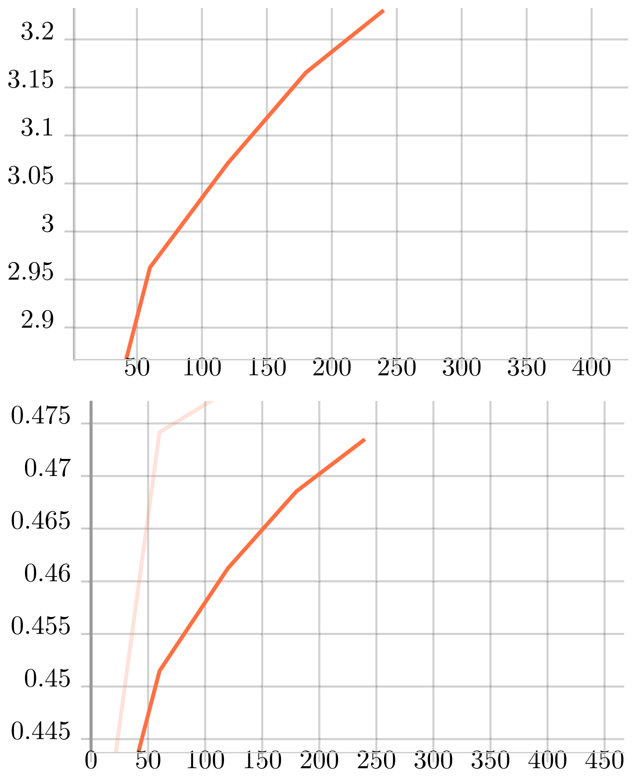


Fig. 24 Curves of ROUGE and CIDEr results after only 10 epochs for fine-tuning

ures 21, 22 and 23 show how the accuracy using the BLEU, ROUGE, and CIDEr metrics when fine-tuning gave largely better results.

Figure 24 shows how the accuracy is increasing while keeping a smooth and steady shape without oscillations.

5.6 Comparative study

Table 2 shows the average test run time for the DenseCap model and our Segment Captioning model on our environment.

The reduction in processing time can be attributed to the removal of the Localization Layer, which incorporates an RPN (Region Proposal Network). Instead of spending time to find boxes with high scores, our model takes the coordinates of bounding boxes as input. Additionally, our Features

Table 2 Execution time comparison between DenseCap and our model. Our segment captioning model shows better execution time

Model	RPN (ms)	CNN (ms)	LSTM (ms)	Total per image (ms)
DenseCap	36	55	4	95
Ours	–	48	2	**50**

The bold values indicate the better results

Table 3 Execution time comparison between DenseCap and our model when eliminating unnecessary regions

Model	Inference (ms)	Box align (ms)	Total per Image (ms)
DenseCap	95	135	225
Ours	**50**	–	**50**

The bold values indicate the better results

Table 4 Results of our final framework in terms of execution time and accuracy

Captioning		Directions	
Time	BLEU	Time	Acc
50 ms	0.332	1e-5 ms	99%

Extractor has become marginally faster as a result of reducing the number of regions that require computing the RoI Align.

To study the performance of our model and DenseCap model when used as captioners for our whole framework, we had to remove boxes proposed by DenseCap and keep only the regions that align with segments. In order to remove unwanted boxes from the results of DenseCap and retain only those that align with the regions provided by panoptic segmentation, we calculate the intersection over union scores for each region and select the best matching boxes based on these scores.

The Table 3 shows the execution time for the general region captioning.

In conclusion, our model is better when it comes to Region/Segment Captioning as it decreases the execution time T.

$$\boxed{\begin{aligned} T(Seg) + T(Ours) << T(Seg) + T(DenseCap) \\ + T(BoxAlign) \end{aligned}}$$

5.7 Final captions

By running the previously detailed directions computation, we assign to each region a direction. Objects on the ground and in the background are kept apart. The object with the maximum c_y value (y coordinate of its centroid) is the object on the ground. The object with the maximum depth value z_{mean} (the mean value of its overall depth) is the object in the background. For our annotated data with 1183 regions, the Table 4 summarises the achieved results.

Qualitative results are shown in the Figs. 25, 26 and 27. The Generated captions are satisfactory regarding the accuracy achieved on different metrics. Even in images with less lighting showing the effectiveness of the model. Egocentric descriptions are correct because they are based on the computation of a real-world angle using depth values that were captured using the Kinect in millimeters. Even if some directions would seem false when looking only at the RGB image, they are correct compared to ground truth. This is due to perspectives that change after capturing a 3D world scene into a 2D image, hence the need of the depth maps.

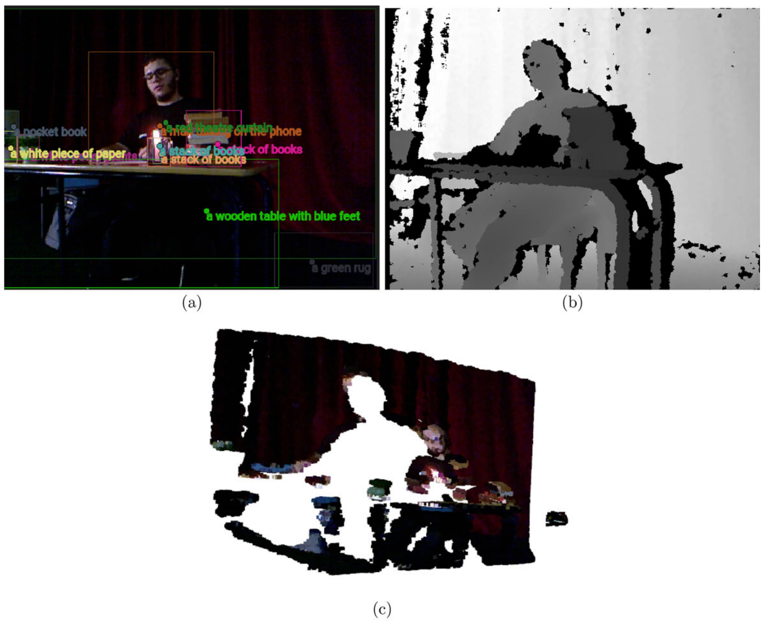
5.7.1 Discussion

From qualitative, quantitative results and training curves:

- The framework answered the problem of egocentric scene description;
- Due to the small size of our dataset, the overfitting is still present and it is noticed when changing slightly the coordinates of bounding boxes, hence including more or fewer pixels for a region to be described, the LSTM captioner performance drops;
- From results given by Head 1 and Head 2, we deduce that information from pixels surrounding a bounding box is important for more contextual meaning since Head 1 eliminates pixels outside of the polygonal envelope gave worse results;
- The directions computation performance is satisfying compared to ground truth;
- The execution time of the whole framework is satisfactory.

Although our model has shown promising results on the small validation set, indicating its effectiveness, it is still presents weaknesses as we present below:

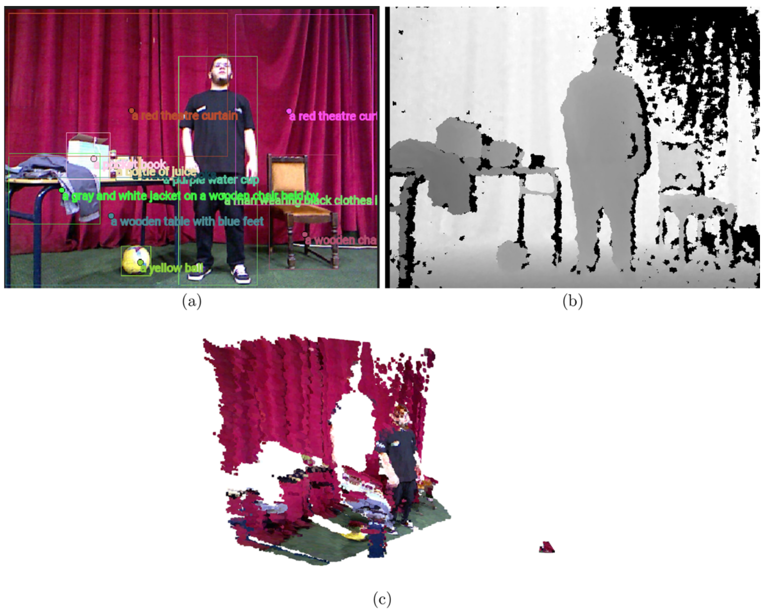
1. Due to the overfitting of the captioning model, this solution cannot be applied to new images;
2. Due to the overfitting of the captioning model, if an input bounding box of a region is less accurate than the ground



On the ground there is: a green rug
On the background there is: a red theatre curtain
On the Left there is: a white pot of plants, a white piece of paper, a pocket book
On the Right there is: nothing
Up Front there is: a wooden table with blue feet, a stack of books, of a stack of books, a man talking on the phone, a stack of books, a stack of books

(d)

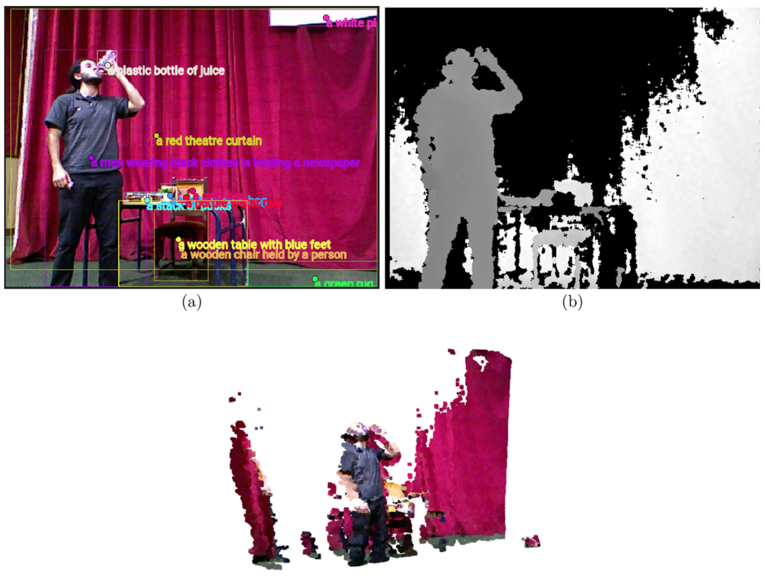
Fig. 25 First example from our theatre scenes RGBD dataset. Image (**a**) represents the results from the captioning model. Image (**b**) is the corresponding. Image (**c**) is the point-cloud. Image (**d**) shows the final egocentric captions



On the ground there is: a yellow ball
On the background there is: a red theatre curtain
On the Left there is: a pocket book, a gray and white jacket on a wooden chair held by
On the Right there is: a wooden chair held by a person, a red theatre curtain
Up Front there is: a man wearing black clothes is holding a newspaper, a wooden table with blue feet, a purple water cup, a stack of books, a bottle of juice, a wooden table with blue feet

(d)

Fig. 26 Second example from our theatre scenes RGBD dataset. Image (**a**) represents the results from the captioning model. Image (**b**) is the corresponding. Image (**c**) is the point-cloud. Image (**d**) shows the final egocentric captions


a white pi
a plastic bottle of juice
a red theatre curtain
a man wearing black clothes is holding a newspaper
a stack of books
a wooden table with blue feet
a wooden chair held by a person
a green rug

(a) (b)

(c)

On the ground there is: a green rug
On the background there is: a green rug
On the Left there is: a man wearing black clothes is holding a newspaper
On the Right there is: a white piece of paper
Up Front there is: a plastic bottle of juice, a stack of books, a white plastic bag, a purple water cup, a red theatre curtain, a white plastic bag, a purple water cup, a red theatre curtain, a gray bag, a wooden chair held by a person, a wooden table with blue feet

(d)

Fig. 27 Third example from our theatre scenes RGBD dataset. Image (**a**) represents the results from the captioning model. Image (**b**) is the corresponding. Image (**c**) is the point-cloud. Image (**d**) shows the final egocentric captions

truth bounding box, or shifted, then the accuracy of the model drops;

3. Our solution depends on panoptic segmentation, if the segmentation model gives bad results it would mislead the captioning model;
4. Our solution depends on panoptic segmentation, if the used model is slow it would slow up the execution;
5. The directions calculation depends on the camera parameters when computing the point cloud;

However, this research encourages us to continue with data annotation and consider collecting additional data with more diverse and sophisticated materials, that involve real actors and disguises, which can further enhance the performance of our model.

The accuracy of generated phrases and their correct semantics motivates us to refine our annotations for future work, incorporating more theatre-specific language such as using terms like "prince" instead of "man" and including female characters as "princesses" for example, along with other theatrical elements and settings.

We also plan on retraining a fast segmentation model to keep up with the real-time execution and enriching captions with textual descriptions of actions by applying the methods proposed in [40] since they rely on depth as well.

Furthermore, we aim to extend our work to video processing, as our framework has demonstrated good execution time, achieving 24 frames per second.

Finally, the solution will be presented to actual users with visual disabilities for a better evaluation.

6 Conclusion

In this article, we have provided a comprehensive review of the recent advancements in AI technologies for visual scene understanding. Through an extensive analysis of the literature, we have examined the state-of-the-art techniques and methodologies employed in various domains, including image captioning, image segmentation, and scene understanding.

Furthermore, we have discussed the challenges and limitations that still exist in visual scene understanding. Despite significant advancements, issues such as handling occlusions, robustness to diverse environmental conditions, and generalization to unseen scenarios remain areas of active research.

In our study, we addressed the challenge of developing a framework that answers the problem of enhancing scene understanding by providing textual descriptions of regions of interest within an image. Our objective was to generate captions that not only describe the visual attributes of these regions but also incorporate their relative positions with respect to the user.

We applied our solution to our new TS-RGBD dataset, a dataset of RGB-D images of theatre scenes, which is considered a novel field of application for image captioning.

Our solution proved to be more effective compared to other image captioning models, in terms of the number of captions per image and the execution time.

The egocentric descriptions were successfully processed by fast and simple algorithms, independent of machine learning methods. Those algorithms used the depth information from the depth maps to improve the accuracy of the descriptions.

For future work, we will improve the quality of image ground truth captions by making them more theatre-specific, and augment our TS-RGBD dataset by collecting and annotating additional data.

Our solution will be presented to blind and visually impaired users for a better evaluation of the generated captions.

Author Contributions K.D. contributed to analyzing, conceptualizing, and developing the presented work and manuscript writing. S.L. contributed to analyzing, and conceptualizing,of the presented work, and manuscript writing as the thesis supervisor.

Data availibility No datasets were generated or analysed during the current study.

Declarations

Conflict of interest The authors report there are no conflict of interest to declare.

References

1. Zatout, C., Larabi, S.: A novel output device for visually impaired and blind people's aid systems. In: 1st International Conference on Communications, Control Systems and Signal Processing (CCSSP), El Oued, Algeria, pp. 119–124 (2020). https://doi.org/10.1109/CCSSP49278.2020.9151820
2. Zatout, C., Larabi, S.: Semantic scene synthesis: application to assistive systems. Vis. Comput. **38**, 2691–2705 (2022). https://doi.org/10.1007/s00371-021-02147-w
3. Be My Eyes. https://www.bemyeyes.com/
4. Microsoft Seeing AI. https://www.microsoft.com/en-us/ai/seeing-ai
5. MindsEye Radio, Translating Vision Into Audio. https://mindseyeradio.org/. (Accessed on 9 July) (2023)
6. Benhamida, L., Delloul, K., Larabi, S.: TS-RGBD Dataset: A Novel Dataset for Theatre Scenes Description for People with Visual Impairments. https://doi.org/10.48550/arXiv.2308.01035. Preprint (2023)
7. Delloul, K., Larabi, S.: Egocentric scene description for the blind and visually impaired. In: 5th International Symposium on Informatics and Its Applications (ISIA), M'sila, Algeria, pp. 1–6 (2022). https://doi.org/10.1109/ISIA55826.2022.9993531
8. Xian, T., Li, Z., Zhang, C., Ma, H.: Dual global enhanced transformer for image captioning. Neural Netw. **148**, 129–141 (2022). https://doi.org/10.1016/j.neunet.2022.01.011
9. Jiang, W., Li, Q., Zhan, K., Fang, Y., Shen, F.: Hybrid attention network for image captioning. Displays **73**, 102238 (2022). https://doi.org/10.1016/j.displa.2022.102238
10. Wang, J., Yang, Z., Hu, X., Li, L., Lin, K., Gan, Z., Liu, Z., Liu, C., Wang, L.: Git: A generative image-to-text transformer for vision and language (2022)
11. Xiaobao, Y., Yang, Y., Wu, J., et al.: Ca-captioner: a novel concentrated attention for image captioning. Expert Syst. Appl. **250**, 123847 (2024). https://doi.org/10.1016/j.eswa.2024.123847
12. Chen, L., Li, K.: Dual-adaptive interactive transformer with textual and visual context for image captioning. Expert Syst. Appl. **243**, 122955 (2024)
13. Saeidimesineh, R., Adibi, P., Karshenas, H., Darvishy, A.: Parallel encoder–decoder framework for image captioning. Knowl. Based Syst. **282**, 111056 (2023). https://doi.org/10.1016/j.knosys.2023.111056
14. Jia, J., Ding, X., Pang, S., Gao, X., Xin, X., Hu, R., Nie, J.: Image captioning based on scene graphs: a survey. Expert Syst. Appl. **231**, 120698 (2023). https://doi.org/10.1016/j.eswa.2023.120698
15. Yang, R., Cui, X., Qin, Q., Deng, Z., Lan, R., Luo, X.: Fast rf-uic: a fast unsupervised image captioning model. Displays **79**, 102490 (2023)
16. Shambharkar, P.G., Kumari, P., Yadav, P., Kumar, R.: Generating caption for image using beam search and analyzation with unsupervised image captioning algorithm. In: 5th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, pp. 857–864 (2021). https://doi.org/10.1109/ICICCS51141.2021.9432245
17. Cai, C., Wang, S., Yap, K., Wang, Y.: Top-down framework for weakly-supervised grounded image captioning. Knowl. Based Syst. **287**, 111433 (2024)
18. Du, S., Zhu, H., Lin, G., et al.: Weakly supervised grounded image captioning with semantic matching. Appl. Intell. **54**, 4300–4318 (2024)
19. Boroujerdi, A.S., Khanian, M., Breuss, M.: Deep interactive region segmentation and captioning (2017)
20. Patankar, R., Sethi, H., Sadhukha, A., Banjade, N., Mathur, A.: Image captioning with audio reinforcement using rnn and cnn. In: International Conference on Sustainable Computing and Smart Systems (ICSCSS), Coimbatore, India, pp. 591–596 (2023). https://doi.org/10.1109/ICSCSS57650.2023.10169692
21. Ruifan, L., Haoyu, L., Yihui, S., Fangxiang, F., Xiaojie, W.: Dual-cnn: a convolutional language decoder for paragraph image captioning. Neurocomputing **396**, 92–101 (2020). https://doi.org/10.1016/j.neucom.2020.02.041
22. Chunpu, X., Min, Y., Xiang, A., Ying, S., Ruifeng, X., Jinwen, T.: Retrieval-enhanced adversarial training with dynamic memory-augmented attention for image paragraph captioning. Knowl. Based Syst. **214**, 106730 (2020). https://doi.org/10.1016/j.knosys.2020.106730
23. Zha, Z.J., Liu, D., Zhang, H., Zhang, Y., Wu, F.: Context-aware visual policy network for fine-grained image captioning. IEEE Trans. Pattern Anal. Mach. Intell. **44**, 710–722 (2022). https://doi.org/10.1109/tpami.2019.2909864
24. Kanani, C.S., Saha, S., Bhattacharyya, P.: Improving diversity and reducing redundancy in paragraph captions. In: International Joint

Conference on Neural Networks (IJCNN), Glasgow, UK, pp. 1–8 (2020). https://doi.org/10.1109/IJCNN48605.2020.9206644
25. Tang, T., Chen, J., Huang, Y., et al.: Image paragraph captioning with topic clustering and topic shift prediction. Knowl. Based Syst. **286**, 111401 (2024). https://doi.org/10.1016/j.knosys.2024.111401
26. Che, W., Fan, X., Xiong, R., Zhao, D.: Visual relationship embedding network for image paragraph generation. IEEE Trans. Multimed. **22**(9), 2307–2320 (2020). https://doi.org/10.1109/TMM.2019.2954750
27. Long, Y., et al.: Capdet: unifying dense captioning and open-world detection pretraining. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), Vancouver, BC, Canada, pp. 15233–15243 (2023). https://doi.org/10.1109/CVPR52729.2023.01462
28. Johnson, J., Karpathy, A., Fei-Fei, L.: Densecap: Fully convolutional localization networks for dense captioning (2015)
29. Krishna, R., Zhu, Y., Groth, O., et al.: Visual genome: connecting language and vision using crowdsourced dense image annotations. Int. J. Comput. Vis. **123**, 32–73 (2017). https://doi.org/10.1007/s11263-016-0981-7
30. Lin, T., Maire, M., Belongie, S., Bourdev, L., Girshick, R., et al.: Microsoft coco: common objects in context (2014)
31. Wu, Y., Kirillov, A., Massa, F., Lo, W.Y., Girshick, R.: Detectron2. https://github.com/facebookresearch/detectron2 (2019)
32. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition (2014)
33. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition (2015)
34. Lin, T., Dollár, P., Girshick, R., He, K., Hariharan, B., Belongie, S.: Feature Pyramid Networks for Object Detection (2017). https://doi.org/10.48550/arXiv.1612.03144
35. DenseCap in Pytorch. https://github.com/soloist97/densecap-pytorch. Accessed on 3 August (2023)
36. Dataset: RGB-D Theatre Scenes Dataset. https://github.com/khadidja-delloul/RGB-D-Theatre-Scenes-Dataset. Accessed on 3 August (2023)
37. LabelMe. Image Polygonal Annotation with Python. https://github.com/wkentaro/labelme. accessed on 3 August (2023)
38. Hu, J., Huang, L., Ren, T., Zhang, S., Ji, R., Cao, L.: You only segment once: towards real-time panoptic segmentation (2023)
39. Jain, J., Li, J., Chiu, M., Hassani, A., Orlov, N., Shi, H.: Oneformer: one transformer to rule universal image segmentation. In: CVPR (2023). https://doi.org/10.48550/arXiv.2211.06220
40. Benhamida, L., Larabi, S.: Human action recognition and coding based on skeleton data for visually impaired and blind people aid system. In: First International Conference on Computer Communications and Intelligent Systems (I3CIS), Jijel, Algeria, pp. 49–54 (2022). https://doi.org/10.1109/I3CIS56626.2022.10075662