DenseRefer3D: A Language and 3D Dataset for Coreference Resolution and Referring Expression Comprehension

by

Akshit Sharma

B.Tech., Chandigarh Engineering College, India, 2016

Thesis Submitted in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

in the
School of Computing Science
Faculty of Applied Sciences

Declaration of Committee

Name:	**Akshit Sharma**
Degree:	**Master of Science**
Thesis title:	**DenseRefer3D: A Language and 3D Dataset for Coreference Resolution and Referring Expression Comprehension**
Committee:	**Chair:** Ke Li Assistant Professor, Computing Science
	Angel X. Chang Supervisor Assistant Professor, Computing Science
	Anoop Sarkar Committee Member Professor, Computing Science
	Angelica Lim Examiner Assistant Professor, Computing Science

Abstract

Coreference resolution is a challenging problem that requires clustering relevant mentions based on referent objects in a text document. Most work on it has relied extensively on text-only datasets, which fail to provide visual cues about the entities represented by the phrases. On this basis, we introduce DenseRefer3D, a language & 3D dataset to create alignment between rich referring expressions and real-world objects and an annotation tool, DenseRefer3D-Annotator, that facilitates the rendering of natural language sentences and 3D scenes. The tool provides functionalities to manage data collection workflow on the MTurk crowdsourcing platform efficiently and enables effective visualization of coreference links and phrases-to-object mappings. We outline several coreference experiments using an end-to-end deep learning approach, analyze the quality of detected mentions and clustering, propose a new task that directly aligns textual phrases with 3D objects, and explore ways to further research in the combined domain of language and vision.

Keywords: Coreference Resolution; Referring Expression Comprehension; Language and 3D Dataset; 3D Annotation Tool; Text-to-3D Scene Alignment

Dedication

This thesis is dedicated to my best friend, my partner, Lucky, for her endless love, support, and encouragement and to my mentor, Sri Sri, for their guidance and blessings. Thank you for all of your help along the way. I am beyond grateful to have you in my life.

Acknowledgements

I would like to extend my sincere gratitude to my supervisor, Dr. Angel Chang, for providing a great opportunity to work with her. I'd also like to thank Dr. Anoop Sarkar, Dr. Angelica Lim, and Dr. Ke Li for taking the time to participate in my thesis defence. Special thanks to all the students at SFU who helped provide initial feedback for our annotation tool and to the thousands of annotators on MTurk for enabling me to collect the dataset. Finally, I am deeply grateful to our amazing families for their constant support and well wishes.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Motivation

Imagine a scenario where one would like to eat their favourite mint chocolate chip cookies placed amongst many other types of cookies in the kitchen cabinet above the stove. For a human, this would be a trivial chore requiring simply opening the cabinet and getting the preferred cookies. However, the same task can be highly challenging for a personal robot. Besides the navigational challenges, it needs to understand the query to identify its different components. For instance, it should determine the main object in the query, mint chocolate cookies, proximity to other mentioned objects, other types of cookies, and the object's location, the kitchen cabinet above the stove. This problem requires understanding the natural language description of the 3D objects and how the words relate to the different attributes of the objects. These tasks fall under the category of 3D referring expressions. Given an indoor 3D RGB-D scene consisting of several objects and a natural language sentence describing these objects, the objective is to map each phrase corresponding to real-world entities with the object(s) from the 3D scene.

We introduce a new dataset for visually grounding referring expressions in 3D RGB-D scenes. Our dataset contains annotations of all the words or phrases representing an object, thereby obtaining dense, many-to-many mapping of phrases-to-objects. These phrase-object relationships are beneficial for natural language and 3D scene understanding tasks as it associates a real-world object to an arbitrary phrase, allowing for learnable grounding of phrases in 3D scenes. We annotate the phrases using their character offsets and link them with objects using the object IDs. This approach provides access to the segmentation mask or the 3D bounding box for an object using its ID. To our knowledge, there are no large-scale datasets for 3D coreference resolution and referring expression comprehension tasks. For our task, we collect annotations using ScanNet [20] and ScanRefer [11] datasets, but with little modification, one may adopt it to make use of other 3D and language datasets.

In order to create our dataset, we developed a comprehensive annotation tool that, besides other functionalities, supports the rendering of language and 3D data together to curate ground-truth labelling of 3D objects for textual phrases. As we detail in Chapter 2, we found that the existing annotation tools are unable to effectively render 3D scenes and natural language sentences together in one view. In addition, the existing tools do not allow for visualization and verification of the annotations. Adapting the existing tools to cater to our desired dataset would have required considerable time and effort. Due to this, we decided to build the annotation tool from the ground up with different interfaces for annotation, visualization, and verification of the collected data. The feature-set built into the tool allows us to enforce critical quality control measures throughout the annotation process without requiring manual human interactions. In addition, the tool encompasses various administration and management controls to monitor the progress of the data collection effectively. We also provide numerous advanced user controls to ensure the efficient flow of the annotation process.

Furthermore, we explore two significant tasks in the language and vision domains: coreference resolution and referring expression comprehension. Coreference resolution involves identifying all the phrases in a sentence that refer to the same real-world entity. The task involves a two-stage process: detecting all the candidate words or groups of words that potentially describe an underlying object and grouping these identified words into clusters based on the objects referred by them. This challenging task includes figuring out the referent object for noun phrases and, especially, its surrogate phrases, such as pronouns. The task complexity rises with the increase in the number of surrogate phrases used to represent the objects. In this work, we provide an exhaustive overview of the coreference resolution task, focusing primarily on an end-to-end approach that detects suitable words or phrases and clusters them together without relying on inputs from any language parsers to aid the detection of these candidate phrases. Most related work on coreference resolution has relied solely on the phrases mentioned in the description, with less attention paid to the underlying real-world entities described by these phrases. This can be attributed to the lack of large-scale multimodal datasets that provide a mapping from phrases in the textual description to real-world 3D objects. This gap was a good motivation to generate our annotations with an emphasis on defining the links between phrases and objects. The availability of such a dataset enables us to investigate the impact of using 3D features, along with features from the text, for resolving the coreferences occurring in a natural language utterance.

The other problem we briefly surveyed is known as referring expression comprehension, which aims to ground the object-describing phrases in a 3D scene by generating a bounding box or a segmentation mask around the described objects. The model is tasked with selecting the 3D object(s) best described by the given referring expression. This task becomes increasingly challenging when a scene includes hundreds of objects available for assigning to the text expressions. The difficulty may also arise when a referring expression contains too little (or too much) information to characterize the object(s) usefully. The existing methods for this task use 2D images as visual inputs, which lack the depth dimension vital in accurately predicting objects. Moreover, most work in this field has operated on image data, which tends to be less accurate in modelling real-world scenarios. Thus, we chose to build the dataset using 3D RGB-D point clouds as our inputs because it contains the precise location of objects which is useful for developing models that can learn from more practical and accurate data points.

The increased availability of large-scale 3D and language datasets has enabled an expanded focus on research in the combined domain of language and vision. To contribute to this essential field, we propose a new task of aligning phrases to objects directly in a 3D RGB-D environment. For a given description, we aim to identify all the phrases that describe some objects in an indoor 3D scene and then ground these phrases in the scene by drawing a 3D bounding box around the referred objects. This task can be seen as a combination of coreference resolution and referring expression comprehension operating on all the referring expressions, not just the co-referring ones. This end-to-end approach of detecting suitable phrases and locating objects for each phrase prevents us from designing separate components for all the associated sub-tasks. It equips us to comprehend the decision-making process of the model reasonably. With this new task, we desire to understand better the importance of different features from language and vision data inputs. We enable access to the code repositories for our annotation tool (*DenseRefer3D-Annotator*) and the baseline methods for our new task. Moreover, we make our new dataset (*DenseRefer3D*) available for the research community upon request.

1.2 Contributions

The main contributions of my thesis are as follows:

- Development of an all-encompassing annotation tool that supports rendering 3D scenes and language descriptions in the same interface, along with capabilities for visualization and verification of the annotations.
- Creation of a large-scale, richly-annotated language and 3D dataset applicable for coreference resolution and referring expression comprehension tasks.
- Thorough analysis of the quality of the datasets by conducting several experiments to evaluate the performance of coreference clustering and mention detection.
- Introduction of a new task that aligns referring expressions from natural language description to the objects in a 3D RGB-D indoor scene.

We describe the above contributions in detail in their respective chapters. We discuss our new 3D annotation tool and DenseRefer3D dataset in greater detail in Chapter 3 and provide the review of coreference resolution and referring expression comprehension tasks in the Section 2.1 and Section 2.2, respectively. Additionally, we provide details on the various experiments we performed and summarize the obtained results in Chapter 5.

This dissertation is an outcome of a collective effort between Akshit Sharma (thesis author) and Angel X. Chang (senior supervisor). I developed our 3D annotation tool (*DenseRefer3D-Annotator*) and curated a new language and 3D dataset (*DenseRefer3D*). In addition, I conducted experiments related to coreference resolution, mention identification, and phrase-to-3D object alignment baseline method proposals under the supervision of my senior supervisor.

Chapter 2

Related Work

2.1 Coreference Resolution

Coreference resolution is a challenging problem in the field of natural language processing that deals with determining the referring expressions in a given text that refer to the same entity. This task lends its usefulness to various application areas of NLP domain, such as Question Answering [62] and Information Extraction [100]. This problem is often associated with anaphora resolution; however, one key difference between the two is that the latter is limited to resolving the references in the backward direction. i.e., it does not consider resolving the referring phrases that have their referent occurring later in the text.

The traditional methods for solving the coreference resolution problem used deterministic systems that relied on enforcing the syntactic and semantic constraints to reason about the agreement between the referring expression and the referent [36]. In comparison, the modern coreference learning approaches are typically modelled into three categories: mention-pair, mention-ranking, and entity-based models. Mention-pair techniques classify every pair of mentions to predict a reference link between them, followed by grouping mentions into clusters based on the links [85, 67, 22]. The downside of this approach is that it does not use global cluster information, as each mention-pair is classified independently. Mention-Ranking models help overcome this shortcoming by computing a pairwise score between each mention and all its candidate antecedents and selecting the highest scoring candidate as the target antecedent [97, 76]. However, the effectiveness of this approach decreases when dealing with singleton mentions and the model's ability (or lack thereof) to avoid merging undesirable clusters.

For the approaches discussed above, the model takes a multi-sentence text as input along with the group of mentions. These mentions are obtained using mention detection pipelines, such as part-of-speech (POS) taggers and dependency parsers. This reliance on output acquired through external sources can lead to cascading errors, consequently affecting the model's performance while making it hard to interpret. Lee et al. [51] addressed this problem by jointly performing the mention detection and clustering tasks, thus eliminating the need to use any external inputs. This approach closely resembles mention-ranking but examines all sequences of tokens, i.e. spans, as the candidate mentions and computes pairwise scores to figure out the best antecedent for all the spans. Each span is encoded using word and character embeddings and processed through a Bidirectional LSTM [39], along with an attention mechanism for finding the head words [6], to obtain vectorized representations. This method outperformed all previous works with more than 3% gains in CoNLL F1 metric score [72] over the previous state-of-the-art model in CoNLL-2012 shared task [71]. The authors improved the performance by proposing a higher-order coreference resolution [52]. In this, a coreference decision between each span pair is informed by the global features of clusters.

The previous works in resolving coreferences have mainly dealt with using language features from the given text. This strategy rules out the tendency to use the visual features of the referent object since these models are supervised by only observing the ground-truth clusters of mentions. In a sense, the models have no concept of the actual entity described by the mentions. It is imperative to explore the design of models that consider the features from both the text and visual modalities to discuss the significance of the supplemental information.

The area of visual coreference resolution has recently started to gain attention. It is the task of identifying the same referent object in an image described by the two or more textual phrases in the description. Kottur et al. [47] proposed a modular approach to this problem, where the model first determines all the referring expressions that describe some objects in the image and grounds the expressions that were not observed before in the processed text. The method attempts to resolve its coreferences for the already discovered expressions by predicting the antecedents based on the previous observations in the text. It relies on MNIST Dialog Dataset [81] for model training, which consists of 2D images as visual input and a series of dialogues in the form of question-answer pairs describing objects in the image as textual input.

Yu et al. [99] builds on End-to-end Coreference Resolution [51] by combining image features extracted from ResNet-152 [38]. The textual features of candidate spans are obtained from visual data to examine the impact of including auxiliary information. In addition, this work proposed VisCoref model [99], in which the image is first processed through an object detector module to acquire object proposals and labels. The labels are encoded using the vectorized span representation described in Lee et al. [51]. The learning objective is to determine the optimal pairing of candidate spans and predicted objects referred to by these spans.

However, this approach focuses on determining antecedents specifically for pronouns in a series of textual dialogues. Similar to the other approach discussed above, it also extracts the visual information from 2D images, which lacks the benefits of the increased level of information available when directly working with 3D data. Our work involves multi-sentence descriptions of objects [11] from 3D RGB-D reconstructions of indoor scenes [20], enabling us to conduct our research on coreference resolution with visual information in a more practical setting.

2.2 Referring Expression Comprehension

Referring expression comprehension (REC) deals with localizing the objects described by the referring expressions in a sentence. It is useful in many combined domains of language and vision, such as Visual Dialogue [102] and Visual Question Answering [95], allowing us to attain better natural language interpretation and visual scene understanding. It is comparable to the object detection [77] task, which aims to locate the object instances in an image by drawing a bounding box around the object. However, object categories are pre-defined in object detection, and the model classifies each object proposal into one of the known categories. On the other hand, in referring expression comprehension, the objects are localized based on the linguistical expressions from the sentence.

The challenges in this task stem from the complexities in the natural language referring expressions. These expressions often encompass varying degrees of helpful information to describe the object, such as its colour, shape, size, and location concerning the surrounding objects. The task entails comprehending available information from referring expressions to find the best-matching real-world object for the given phrase. In addition, the variation in the number of words in an expression can add to the task's difficulties. A shorter phrase with few words may not have enough information to identify an object uniquely. On the contrary, an extremely long expression might require additional reasoning measures to extract valuable details about the described entity. Moreover, the visual media presents its interpretation-related challenges, including noise and the absence of structural rules in the images.

The earliest machine learning-based approaches for the task used graphical methods or relied on parsed textual input. Kong et al. [46] presented the task of aligning all nouns and pronouns in the sentence with the objects in an RGB-D image. This work proposed a probabilistic model based on the Markov Random Field (MRF) that aims to provide reasoning for assigning suitable objects to all the available nouns and pronouns. The model takes an RGB-D image and a multi-sentence text describing the objects and jointly reasons about the text-to-image alignment, detecting objects in the image and determining scene type from the information available in the text description. As part of this work, they generated the Sentences3D dataset [46], which contains the mappings between phrases and visual entities in 2D images. This work demonstrated the effectiveness of utilizing language and image features to improve the task of resolving coreferences. As a result, the model achieved higher scores over Stanford coreference resolver [50]. However, this method works on single-view images, thus failing to apprehend the full scope of objects in a 3D environment. Furthermore, it relies on the output of Stanford's dependency parser [88] to work out the textual entities of interest. Therefore, it is prone to cascading errors from external inputs that are hard to diagnose.

The focus has recently shifted towards deep learning methods, including using a convolutional neural network (CNN) [84] to generate object proposals and a long short-term memory network (LSTM) [34] to produce a contextually-aware sentence representation. This task is often preceded by the referring expression generation (REG), which concerns creating a sentence to describe an object in an image distinctively. Mao et al. [58] introduced a deep learning-based approach for expression generation and comprehension using combined CNN-LSTM architecture. Yu et al. [98] presented a modular system to process a referring expression as an amalgamation of three components: subject, location, and relation with surrounding objects. Each of these textual components is an input to the corresponding visual component. It then focuses on learning attention for the visual and language components to emphasize the prominent phrases in the expression. This model notably outperformed previous state-of-the-art methods. Lu et al. [56] builds upon BERT [23] language model to jointly learn language and vision representations independent of the type of the underlying task. The learned representations can then be transferred for various downstream language and vision tasks. For instance, the authors used the pre-trained representations as a baseline for referring expression grounding and reported significant performance improvements over previous state-of-the-art model results.

Most related work on referring expression comprehension utilizes either 2D or RGB-D images. Chen et al. [11] introduced localization of objects described using a natural language sentence directly in 3D scenes. The model takes the point cloud of a 3D scene as visual input and the vectorized embeddings of all the tokens in the sentence as language input and highlights the 3D object best described by the sentence with a 3D bounding box. For this purpose, they created a new dataset consisting of sentences that uniquely depict a 3D object in an RGB-D scene. The proposed end-to-end architecture involves merging the object proposals and sentence features to learn the correlation between the two modalities. This work further showed the significance of using supplemental visual and language information such as colours, normals, and a language-to-object classifier. Each of these inclusions resulted in improved object localization performance.

For most of the methods discussed above, the fundamental objective is to rank the image proposals with a referring expression. This approach assumes the language input as a referring expression and does not take into account the surrounding context. The context is advantageous when dealing with language-based problems that require the interpretation of sentences with varying constructs. Our work is focused on aligning all the referring expressions in a sentence with corresponding 3D objects in the scene. With this, we aim to investigate the impact of using feature-rich natural language sentences to improve 3D scene understanding.

2.3 Existing Datasets

We provide a quantitative and qualitative analysis of the existing datasets for the coreference resolution and referring expression comprehension tasks.

2.3.1 Coreference Resolution

Many datasets have been curated for coreference resolution over the years. The coreference data by CoNLL-2012 shared task [71] remains the most utilized dataset for working out the coreferences. CoNLL-2012 corpus [71] consists of approximately 1.6 million tokens extracted from around 2400 OntoNotes dataset v5.0 [92] documents. The OntoNotes dataset contains, among several other layers, a coreference layer. This layer includes anaphoric coreferences for noun phrases, pronouns, and headwords in verb phrases. The coreferences are cumulated in the form of mention chains, where each mention in the chain refers to the same real-world entity. It consists of 194,480 mentions referring to 44,221 entities, with 150,259 coreference links between them. The English documents in OntoNotes v5.0 are sourced from different genres, such as media and news broadcasts, magazines, conversational telecasts, and newswire. Each document is rendered with all the OntoNotes annotation layers using the CoNLL tabular format.

WikiCoref [32] and GAP [91] are two additional well-received datasets serving similar purposes but with slight differences. WikiCoref, as the name suggests, contains documents from English Wikipedia pages. WikiCoref is on the smaller side containing 60,000 tokens from just 30 documents, which is likely the reason for the lack of influx of studies using this dataset. Webster et al. [91] presented Gendered Ambiguous Pronouns (GAP) consisting of almost 9000 ambiguous pairs of pronouns and names extracted from Wikipedia. Hence, it is unsuitable for tasks involving inanimate objects, such as indoor scene understanding. PreCo [12] is another popular coreference resolution dataset that was curated to facilitate the research in mention detection and clustering discretely. Additionally, it leads to performance improvements in resolving coreferences by reinforcing the overlap between the training and test data splits, as the low overlap between the two sets was observed as a vital factor in determining the model's performance [61]. This was accomplished by curating an adequately large dataset and confining the domain of the annotations to mainly preschool-level English language vocabulary. It consists of 12.4 million tokens from about 38,000 documents, which makes it around eight times larger dataset than OntoNotes [92]. The reported experiments on OntoNotes and PreCo datasets using an augmented end-to-end coreference resolution [51, 70] model show a significant increase in weighted average F1

Datasets	Modalities	Documents	Tokens	Avg. Tokens/Documents	Coreference Chains
PreCo [12]	text	37,600	12,400,000	329.78	434,000
OntoNotes v5.0 [92]	text	2384	1,600,000	671.14	44,221
GAP [91]	text	4454	338,833	76.07	8908
VisPro [99]	text + 2D images	5000	586,927	117.38	7944
Sentences3D [46]	text + RGB-D images	1449	61,195	42.23	1811
WikiCoref [32]	text	30	59,652	1988.4	1785

Table 2.1: Comparison of popular coreference resolution datasets. OntoNotes v5.0 [92] and PreCo [12] remain the two most prominent datasets due to the presence of a large number of annotated coreference chains.

score of 81.5 when trained using PreCo in comparison to 70.4 on OntoNotes. The annotations of singleton mentions in PreCo enable investigation into understanding the importance of mention detection and clustering tasks individually for coreference resolution.

The coreference resolution efforts have largely revolved around text-only datasets, and the visual constructs described by the entities in the text have mostly been overlooked. Kong et al. [46] curated Sentences3D dataset which contains natural language descriptions of objects from NYU-RGBD v2 dataset [66]. They collected two types of annotations, namely, visual and text-only. Visual annotations provide an alignment from nouns and pronouns to the referred real-world object. In text-based annotations, all nouns and their attributes, such as colours, are annotated with one of the 21 object classes derived from Lin et al. [53]. Moreover, the co-referring words are defined by a link between the head noun and all other words, including pronouns, describing the same real-world object. It relies on the Stanford parser [88] for producing the candidate noun phrases and pronouns. The model trained using this multimodal dataset outperformed the text-based coreference system [50]. However, it is considerably small, consisting of just 1449 multi-sentence descriptions of the objects for the same number of images from the NYUv2 dataset, with an average of three sentences and 40 tokens for each description. The relatively small dataset size makes it less effective for modern deep learning-based methods that usually require much larger datasets to train the model.

Yu et al. [99] introduced VisPro, a large-scale dataset built on top of VisDial v1.0 [21] and primarily focused on resolving coreferences for pronouns occurring in dialogues. The main purpose of this research effort was to leverage the features extracted from images and textual information to reason about the significance of incorporating visual modality for the coreference resolution task. Similar to the previous work, the candidate phrases, including noun phrases and pronouns, are generated using Stanford Parser [88]. The dataset consists of annotations of around 30,000 pronouns acquired from 5000 multi-sentence dialogues, with an average of close to six pronouns available per dialogue. Overall, the dataset contains about 74% of anaphoric pronouns, with the remaining either do not have an appropriate antecedent or are pleonastic in nature. SIMMC 2.0 [48] curated a new multimodal dialogue dataset for enabling research in the visual coreference resolution domain. It comprises 117,000 sentences from 11,000 dialogues between a user and an assistant from 1566 scenes set in the discipline of in-store shopping. Each scene has an average of around 20 objects per dialogue, each describing approximately five objects. The complex nature of the scene environment is reflected in the state-of-the-art model's performance on SIMMC 1.0 [60] and SIMMC 2.0 [48], which significantly drops when using SIMMC 2.0. Hence, further research is required to adapt to the complexities in the newer dataset.

2.3.2 Referring Expression Comprehension (REC)

Historically, the most significant works on the referring expression comprehension problem have leaned on image-based 2D datasets. Kazemzadeh et al. [44] introduced RefCOCO, using ReferItGame [44] as baseline, as a first large-scale referring expression comprehension dataset. It includes annotations of 50,000 objects with about 142,250 natural language referring expressions from nearly 20,000 images. Furthermore, Kazemzadeh et al. [44] presented the RefCOCO+ dataset, which is quantitatively comparable to the former but varies in the description of objects by referring expressions. In RefCOCO+, the expressions focus solely on expressing the objects' appearance attributes and do not include phrases that describe the object's location. These datasets have, on average, around 3.5 tokens with an average of four objects of the same type per image. Mao et al. [58] enhanced RefCOCO by introducing the RefCOCOg dataset, which includes richer natural language expressions with more details for depicting the objects. This is reflected in the dataset statistics, with almost 2.5 times more words per expression than the previous two similar datasets. It involves more than 104,500 natural language referring expressions describing nearly 55,000 objects from around 27,000 images. All three above datasets incorporate 80 most prevalent object categories from MSCOCO dataset [54].

Liu et al. [55] created CLEVR-Ref+, a simulated dataset consisting of referring expressions and referred objects constructed from questions and answers, respectively, in CLEVR dataset [42]. It contains just under a million referring expressions for roughly half a million objects from approximately 100,000 images but employs only three object categories. The main goal behind this effort was to address the biases present in the real-world datasets [101]. It has been used to evaluate recently proposed neural network-based models for the referring expression comprehension task. However, the small number of object categories and lack of elaborate relationships between objects in the images does not help fully represent the complexities present in real-world datasets [40]. Chen et al. [13] focused on understanding the reasoning capabilities of referring expression comprehension models by building the Cops-Ref dataset. In addition to enabling the models to learn to identify objects and formulate simple relationships between them, the dataset allows for the evaluation of the extensive reasoning abilities of the model. It consists of 148,712 referring expressions for

Datasets	Modalities	Average Tokens	Objects	Referring Expressions
CLEVR-Ref+ [55]	text + 2D images	22.4	492,727	998,743
Cops-Ref [13]	text + 2D images	14.4	1,307,885	148,712
RefCOCO [44]	text + 2D images	3.61	50,000	142,209
RefCOCOg [58]	text + 2D images	8.43	54,822	104,560
REVERIE [75]	text + panoramic images	18	4140	21,702
Refer360° [15]	text + panoramic images	43.80	124,880	17,137

Table 2.2: Comparison of notable referring expression comprehension datasets. The datasets with abundant natural language referring expressions are favoured for tackling such problems.

roughly 1,308,000 objects from 75,299 images of COCO dataset [54]. The presence of distractor objects analogous to the target objects described by the referring expressions makes it suitable for the challenging task of localizing referring expressions.

Similarly, Cirik et al. [15] developed the Refer360° dataset to identify entities in the referring expressions. It has more than 17,000 natural language referring expressions describing around 125,000 objects from 2000 images of SUN360 dataset [96], with an average of more than eight expressions per scene. The sentences in the dataset are described based on the dynamically changing limited view of the scene, enabling a realistic human-like interpretation of an environment. Moreover, since the annotators were shown a set of partial views instead of the whole scene, the collected descriptions of the target points are more comprehensive than the existing datasets, with about 44 words per description on average.

As summarized above, referring expression datasets have primarily relied on 2D images due to the wide availability of large-scale 2D-based datasets. Much effort has not been concentrated on performing the task directly in 3D, largely due to the lack of extensive 3D datasets. As previously outlined, Kong et al. [46] proposed one of the earliest efforts to align phrases in the sentence with the objects in the RGB-D scenes. Besides the coreference resolution, Sentences3D dataset [46] facilitated research in 3D scene understanding. The dataset consists of alignments of nouns with real-world objects, with the entity-representing nouns annotated with the object classes based exclusively on the textual sentence. It did not reference the visual data during the assignment of the object labels for nouns. Moreover, the nouns are aligned with the image segmentation masks, which require depth information to reconstruct the 3D objects. An imperfect or noisy depth map of the image can adversely affect the quality of the reconstructed objects.

Chen et al. [11] presented the first work, ScanRefer, on using natural language descriptions to identify objects directly in 3D scenes. As part of this effort, they compiled natural language sentences for 3D objects from ScanNet [20] RGB-D scenes. The ScanRefer dataset consists of 51,583 sentences describing approximately 11,000 objects from 800 3D RGB-D ScanNet scenes. A ScanNet scene contains 14 objects on average, and the ScanRe-

Dataset	Modalities	Scenes	Avg. Objects/Scene	Descriptions	Avg. Tokens	Avg. Descriptions/Object	Avg. Descriptions/Scene
ScanRefer [11]	text + 3D RGB-D scenes	800	13.81	51,583	20.27	4.67	64.48
Nr3D [1]	text + 3D RGB-D scenes	707	14.06	41,503	11.4	7	64.74

Table 2.3: Comparison of ScanRefer and Nr3D datasets. ScanRefer dataset contains around 24% more descriptions than Nr3D and consists of more descriptive representations of objects using around 20 tokens per description compared to 11 for Nr3D.

fer dataset has about five descriptions per object and nearly 65 descriptions for each scene, with an average of 20 words per sentence. The sentences describe more than 250 different types of objects commonly found in an indoor house setting. However, each sentence represents a unique target object and does not directly contain ground-truth annotations of all the phrases depicting the target and other objects with the 3D object regions. Hence, it is a challenging endeavour to identify all the phrases and their referent objects using this dataset.

Achlioptas et al. [1] made their contributions to the problem of locating objects referred to by the natural language phrases with the introduction of a large-scale language and vision dataset, Nr3D. The dataset consists of free-form descriptions of 3D objects from the ScanNet dataset, comprising more than seven descriptions for each object and an average of almost 11 words per description. The sentences in the dataset discriminatively characterize a target object with information about surrounding objects in the scene. Similar to ScanRefer, this dataset does not include the alignment of referring expressions in a description with 3D objects, making it challenging to learn the description components responsible for identifying the target objects.

2.4 Language and 3D Annotation Tools

Annotation tools are an integral part of the success of any data collection effort. A useful annotation tool should provide easy-to-use features to curate the desired datasets and visualize the obtained annotations. Moreover, it should encompass additional functionalities to monitor the data collection workflow. This includes features for guiding the annotators towards the preferred result, the ability to modify annotations, and implementing worker access control based on the quality of their annotations. In essence, it should contain an appropriate assortment of automated and manual capabilities to prevent common mistakes due to human intervention while maintaining the accuracy of the collected data.

Müller and Strube [64] developed the MMAX2 tool to facilitate the creation of coreferring chains of mentions in a given text. It allows the annotator to define and visualize the links between mentions referring to the same real-world entities. The mentions are marked with one of the supported coreference types by clicking on the mention and describing a relation with an adjustable line segment. Although, updating or deleting the defined

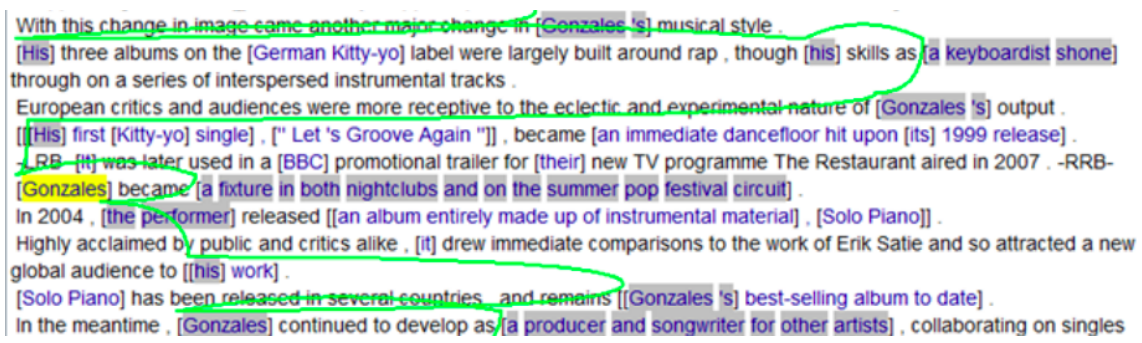

With this change in image came another major change in [Gonzales 's] musical style .
[His] three albums on the [German Kitty-yo] label were largely built around rap , though [his] skills as [a keyboardist shone] through on a series of interspersed instrumental tracks .
European critics and audiences were more receptive to the eclectic and experimental nature of [Gonzales 's] output .
[[His] first [Kitty-yo] single] , [" Let 's Groove Again "]] , became [an immediate dancefloor hit upon [its] 1999 release] .
-LRB- [it] was later used in a [BBC] promotional trailer for [their] new TV programme The Restaurant aired in 2007 . -RRB-
[Gonzales] became [a fixture in both nightclubs and on the summer pop festival circuit] .
In 2004 , [the performer] released [[an album entirely made up of instrumental material] , [Solo Piano]] .
Highly acclaimed by public and critics alike , [it] drew immediate comparisons to the work of Erik Satie and so attracted a new global audience to [[his] work] .
[Solo Piano] has been released in several countries , and remains [[Gonzales 's] best-selling album to date] .
In the meantime , [Gonzales] continued to develop as [a producer and songwriter for other artists] , collaborating on singles

Figure 2.1: Screenshot of visualization interface of MMAX2 [64] annotation tool reproduced from Ghaddar and Langlais [32]. Using pointers to depict the coreference relations makes it hard to trace the cluster mentions.

mention relations is not instinctive. The coreference chains can be visualized by positioning the mouse cursor on any mention in a cluster, and this action highlights all the coreferring mentions. However, with the increased number of annotated coreference clusters, the bracket-based visualization becomes progressively challenging to observe the clustering of mentions. The graphical visualization of clusters, too, fails to accentuate the acquired grouping of mentions comprehensibly. Ghaddar and Langlais [32] utilizes the MMAX2 tool to generate the WikiCoref dataset, which is formatted using standoff XML formatting for distribution purposes.

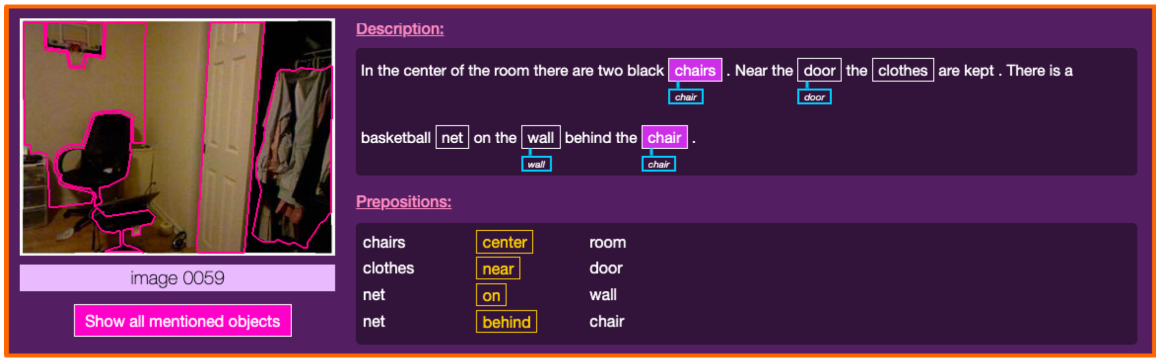
The annotation tools for most studies involved in collecting datasets for the coreference resolution task were never fully released publicly for the benefit of the research community. For instance, Preco dataset [12] was developed using a web-based interface to employ nearly 80 annotators. The annotators were presented with the task instructions and then directed to participate in a qualification test. The public release of the tool did not provide any instructions or illustrations of the interface. The task-oriented GAP dataset [91] provided the same data sample to three different workers tasked with assigning a label from the predefined set of five. Similar to Chen et al. [12], Webster et al. [91] did not supply any specification on the tool created for the data collection.

Kottur et al. [48] implemented a multi-stage approach for the curation of SIMMC 2.0 dataset. The first stage involved showing dialogue and its full context to the task annotators. The task entailed annotating the referring expressions in the text supplied by the user and the virtual assistant. In the second stage, the textual dialogue was supplemented with the image related to the context of the dialogue setting. The tool provided various object classes and their attributes for specifying desired relations. However, due to the lack of public availability of the tool and missing images of annotation and visualization interfaces, it is nearly impossible to determine the effectiveness of their annotation tool based entirely on the description communicated in the paper.

One of the earliest coreference works involving language and vision, Sentences3D [46], collected a multimodal dataset to align the phrases with real-world objects. In order to do that, they created multiple annotation interfaces for gathering text-only ground truth annotations and noun-to-object alignments, along with the capabilities for visualizing the annotated data. The interfaces were developed in MATLAB and were released publicly by the authors as part of their publication. For text-only ground truth annotations, the annotators were required to select an object or scene category. Furthermore, they were tasked with defining the coreference links and labelling the prepositions for all the nouns in the sentence.

The interface consisted of several sections, with the most significant one reserved for annotating sentences. In addition, it provided separate areas for displaying predefined object and scene classes, depicting prepositional relations, and other controls, including interface navigation. The sentences could be navigated with the previous and next buttons or using the intuitive slider to move through them quickly. The tool auto-populated the nouns with available object classes. These classes could be updated by clicking on the button under the noun and making a new selection of an object category, provided the object and scene classes sections were in edit mode (highlighted in yellow). The second button under the noun enabled annotating attributes such as the colour and size of the entity described by the noun. Finally, the coreferences could be defined using the third button under the noun, followed by clicking on the word that co-refers with the noun to create a cluster. However, annotating attributes and coreferences could become tricky without visual indicators, primarily when publishing the task on a crowdsourcing platform. The absence of in-app documentation and sample task demonstrations makes the workflow less efficient, requiring the annotator to refer repeatedly to a separate instructional document.

The other interface for creating links between nouns and visual objects resembles the text-based window. However, it includes a few key differences to support the ground truth alignments of nouns with real-world objects from the images. For instance, most space is reserved for rendering the 2D image with annotated object segments from NYUv2 dataset [66]. The tool also allows adding new object segments beyond what is available via NYUv2 images. It has sections for labelling the objects' colours and sizes, with a pre-populated list of these available attributes. Moreover, it displays the sentence in a different section, where all the nouns are labelled with the visual object IDs. The linked objects could be updated by entering the new IDs or selecting the rest of the co-referring nouns. In our examination of their tool, we discovered that these mechanisms for updating noun-object alignments are prone to errors and often found it challenging to get them to work correctly. As with the text-only annotation tool, the noun-object mapping tool does not directly provide task instructions in the same window. The visualization view highlights all the objects aligned



Description:

In the center of the room there are two black chairs . Near the door the clothes are kept . There is a basketball net on the wall behind the chair .

Prepositions:

chairs	center	room
clothes	near	door
net	on	wall
net	behind	chair

Figure 2.2: Screenshot of web-based Sentences3D [46] dataset visualization reproduced from Sentences3D Dataset Browser [80]. It renders all the linked objects using the same colour. Besides, the colouring scheme used for highlighting words is inconsistent as it does not outline many annotated words in any colour.

with words in the given sentence. However, it does not offer a way to outline the linked objects individually. Besides, the visualization feature is only accessible on their project website[1] and is not publicly available for use with other datasets.

Similar to the coreference resolution, many efforts for generating referring expressions and similar multimodal datasets did not provide adequate information concerning the tools used for annotating and visualizing the data, such as screenshots of various features or the public availability of the interface. For instance, Yu et al. [99], as part of the VisPro dataset collection, deployed their interface on Amazon Mechanical Turk (AMT, MTurk) [41] crowdsourcing platform. The interface consisted of a 2D image and a dialogue surrounding the image context, and the annotators were tasked with finding all the antecedent noun phrases for the highlighted pronoun. The tool populated all the noun phrases with a selectable checkbox which could be assigned as an antecedent of the spotlighted pronoun. In addition, the annotators were required to provide the type of anaphoric relation exhibited by the marked pronoun. The presented image served the purpose of supplementing the context in which the given scenario occurred. Nonetheless, it did not enable the mapping of the noun phrases directly with objects in the displayed image.

Kazemzadeh et al. [44] developed a game-based interface for curating ReferItGame, RefCOCO and its extension datasets and published it on various crowdsourcing platforms, including Amazon Mechanical Turk. In the multiplayer game, the first player was presented with a 2D image with the highlighted target object. The player's objective was to describe the target object as a referring expression. The second player was then entrusted with locating the object in the image using the textual information provided by the first player. The public releases of these works did not contain the annotation tool, which makes it difficult

[1]`https://www.cs.toronto.edu/~fidler/projects/sentences3Ddataset_1.html`

to assess the tool's quality from the provided description and prevents the community from building upon the implemented features instead of starting the development from scratch.

The collection of the REVERIE dataset [75] involved designing a WebGL-based interface for rendering 3D scenes from Matterport3D dataset [10]. It built upon the tool provided by Anderson et al. [3] by incorporating the object labels and bounding boxes that are back-projected from 3D scenes onto 2D images. The tool handled the variations in viewpoints when projecting the 3D bounding boxes. The task objective was to generate referring expressions that the embodied agents easily understand. In order to collect a large number of such expressions, the authors published the task on the AMT crowdsourcing venue. The annotators were first presented with a walk-through of the agent's path, followed by highlighting an object with a bounding box at the end of this sequence. In addition, the tool displayed the target object's label and the category of its environment. The annotators were instructed to furnish the description that helps the intelligent agent quickly locate the highlighted object. The tool facilitated the navigation of the environment surrounding the intended real-world object, allowing annotators to use the contextual information in their descriptions. Furthermore, they deployed a supplementary interface to compare the performance of the embodied agent with the human. In this, the annotators were shown the description of an object. Then, they were directed to maneuver to the target area in the scene and choose the best-matching object amongst several candidates of similar type. In the end, the task was undertaken by more than 1000 annotators, and the collection effort ran for nearly four months.

Liu et al. [55] took advantage of automated annotations using Blender [19] toolset for rendering the objects referred to by the natural language expressions. It converted the referring expressions from the questions of the CLEVR dataset [42] and mapped the answers to the real-world objects to produce the required dataset. Analogous to the automated approach in the previous work, Chen et al. [13] constructed an expression generation mechanism for creating highly elaborate and grammatically accurate referring expressions of the target regions. Cirik et al. [15] organized a setup consisting of multiple tasks and corresponding annotation interfaces to create the Refer360° dataset.

The first task was designed for compiling the natural language sentences that describe any position in the image. This task required locating a randomly chosen target region and providing at least three sentences for finding it. The interface for the second task displayed a 2D image and a set of instructions for helping the annotators locate the described region correctly. The annotators were required to update the region location by moving the selector on top of the intended region. This served as a verification step for evaluating the quality of the descriptions obtained in the first step.

Achlioptas et al. [1] built a two-player online game that was made available on AMT for collecting the Nr3D dataset. The first player was shown a 3D scene rendering from ScanNet dataset [20] with a highlighted object and was tasked to supply the textual description of this object. The second player observed the same 3D scene with all the candidate target objects, including distractors, outlined with a bounding box. They were then responsible for verifying the object description by selecting an object that best fits the defined characteristics. The annotation tool used a lower-resolution 3D scene mesh with texture mapping to enable faster loading times in the web browser while maintaining a high-quality scene rendering.

ScanRefer dataset [11] collection effort involved developing a browser-based tool with support for rendering 3D scenes. Similar to some of the endeavours mentioned above, they carried out the data collection in two phases: gathering descriptions of objects and validating them. During the first phase, the interface spotlighted the target 3D object with the other objects slightly dimmed to make it easy to focus on the intended target in an occupied indoor scene. Additionally, the tool displayed the 2D image of the scene subset to help with the visualization of a partially-visible 3D object due to imperfect scene reconstructions. As part of their description, the annotators were instructed to include the object details, such as its physical features and proximity to surrounding objects. This phase employed workers from English-speaking countries on the AMT platform to collect more than 50,000 natural language descriptions. In the second phase, the tool displayed a 3D scene and the object description accumulated from the previous phase. The objective was to choose the object best described by the accompanying sentence. Furthermore, the verifiers were directed to fix spelling or grammatical errors in the sentences. The verification of the descriptions was carried out in-house by the university students.

We also reviewed other annotation and visualization tools supporting language and vision modalities. Reiter [78] developed an annotation tool that facilitates the labelling of coreference clusters in a given textual description, with support for annotating longer texts and creating extensive chains of co-referring mentions. One of the motivations behind creating the tool was to provide better, less-complicated visualization of mentions in coreference clusters. The tool creates a set of mentions for annotating a coreference chain. All the mentions in a set are underlined with the same unique colour, which makes it easy to visualize and verify different annotated coreference clusters. The interface allows for the selection of phrases using the keyboard in addition to the mouse input. The annotated coreference chains are stored in a standoff format with the help of UIMA [28], which makes it easy to directly export the annotations in various formats, including well-accepted CoNLL-2012. However, the tool does not allow rendering images or scenes, which is essential for understanding the importance of visual features for coreference resolution and referring expression comprehension tasks.

Bornstein et al. [7] focused on designing a tool that works seamlessly for the annotators on the crowdsourcing media. It enabled the implementation of an end-to-end approach for

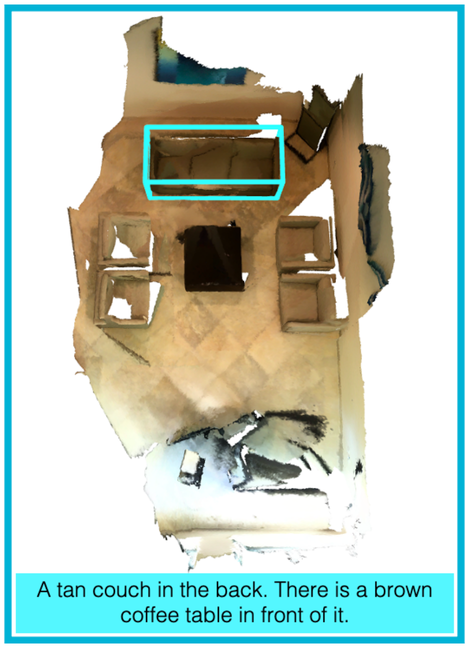
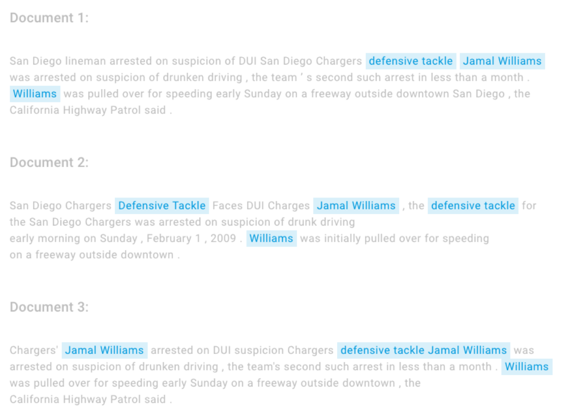

A tan couch in the back. There is a brown coffee table in front of it.

Figure 2.3: A sample annotation from ScanRefer dataset [11]. Each sentence describes the target object and its attributes, such as colour and proximity to surrounding objects.

collecting the required dataset by delivering a comprehensive feature set from task onboarding to data evaluation. The onboarding capability helped impart training to familiarize the annotators with the task. During annotation, the tool outlined a series of mentions, and the annotator was required to assign them to an existing or a new coreference cluster. The highlighted mentions could be updated to include or remove words. In addition, one might define new mentions in the annotation interface. Finally, it provided capabilities for evaluating the quality of the collected clusters of mentions. The appraisal was performed by spotlighting mentions and the clusters they are part of, and the reviewer was instructed to verify the provided clustering. They could also make changes to the listed mentions if necessary. However, it did not include the functionality to visualize all the annotated clusters of mentions in the sentence. It required the reviewer to iterate through each mention to locate the cluster. Besides, it did not highlight any visual links between selected mentions, which led to difficulty in understanding the defined relations among them.

Explosion AI released Prodigy [2], a new annotation tool capable of curating datasets for diverse tasks across the language and vision domains, including annotating coreference relations. In the coreference interface, the potential noun and pronoun phrases are extracted using a pre-trained model, and the remaining tokens in the textual input are disabled for any user intervention. The coreference links between the extracted spans are defined by first clicking on the phrase representing a real-world entity, followed by selecting the co-referring phrase. However, we found the interface challenging to navigate, especially when updating or visualizing the annotated coreference links. Moreover, it does not support rendering the



Document 1:

San Diego lineman arrested on suspicion of DUI San Diego Chargers defensive tackle Jamal Williams was arrested on suspicion of drunken driving , the team ' s second such arrest in less than a month . Williams was pulled over for speeding early Sunday on a freeway outside downtown San Diego , the California Highway Patrol said .

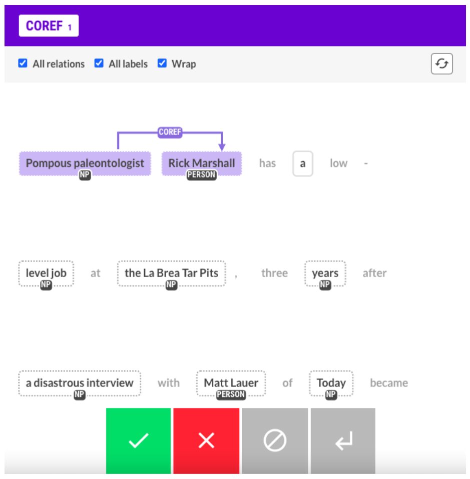
Document 2:

San Diego Chargers Defensive Tackle Faces DUI Charges Jamal Williams , the defensive tackle for the San Diego Chargers was arrested on suspicion of drunk driving early morning on Sunday , February 1 , 2009 . Williams was initially pulled over for speeding on a freeway outside downtown .

Document 3:

Chargers' Jamal Williams arrested on DUI suspicion Chargers defensive tackle Jamal Williams was arrested on suspicion of drunken driving , the team's second such arrest in less than a month . Williams was pulled over for speeding early Sunday on a freeway outside downtown , the California Highway Patrol said .

Figure 2.4: Visualization of annotated mentions in CoRefi [7] annotation tool reproduced from Bornstein et al. [8]. It only allows one to visualize mentions of a single cluster simultaneously.



Pompous paleontologist Rick Marshall has a low - level job at the La Brea Tar Pits , three years after a disastrous interview with Matt Lauer of Today became

Figure 2.5: Prodigy [2] interface for annotating coreference relations reproduced from Explosion AI [26]. It does not provide an easy way to edit the outlined text expressions.

visual data in the same interface to map the candidate co-referring spans directly to the real-world objects. Similarly, Label Studio [87] and INCEpTION [45] tools only enable text-based annotation of coreference clusters.

The lack of open-source or freely available toolkits that promote integrated rendering of text and 3D models, with capacities for an end-to-end data collection workflow, including annotation, visualization, and verification, motivated us to design and develop the desired tool. In Section 3.2, we discuss our 3D annotation tool in greater detail and provide the rationale for designing several included functionalities.

Chapter 3

Data Collection

The goal of this research is to enable improvements to the existing coreference resolution and referring expression comprehension tasks. After thorough analyses of the existing solutions to these problems, we concluded that the current datasets are not well-suited to solve these tasks, especially when dealing with 3D input data and natural language texts. For instance, very few projects have focused on utilizing multiple modalities for coreference resolution. This is discernible since the task entails resolving coreferences in a text containing entities. However, it is often the case that a natural language utterance describes entities in a visual setting. Some entities in the utterance only get their meaning when considering both text and visual modalities.

We set out to take advantage of this construct by curating a new dataset known as DenseRefer3D. Our work is a first-of-its-kind effort to collect large-scale rich annotations of phrases from 50,000+ ScanRefer [11] natural language sentences representing real-world objects in more than 800 3D RGB-D indoor scenes from ScanNet dataset [20]. The following sub-chapters describe our data collection process, including developing a 3D annotation tool, data annotation and verification procedures, and data statistics.

3.1 Proposed Data Structure

We build upon the previous work, ScanRefer [11], that collected natural language descriptions of objects in 3D indoor scenes from ScanNet dataset [20]. In the ScanRefer dataset, each sentence describes one object from a 3D scene, with at most five descriptions of each object. Even though they set out to collect descriptions for each object individually, most descriptions contain references to other nearby objects in the scene to portray the main object collectively. The references to secondary objects include attributes such as their colours, shapes, sizes, positions, and proximities to the main object. We use these detailed descriptions of the primary and secondary objects to collect annotations of all the phrases representing an object(s) or provide a direct or indirect reference to an object(s). We accu-

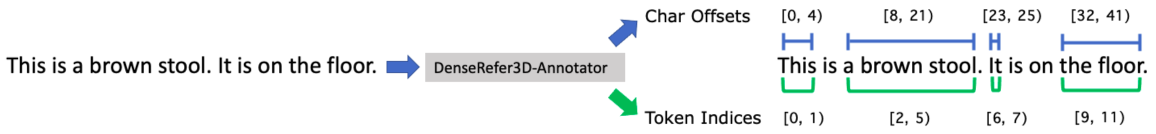


Figure 3.1: Proposed DenseRefer3D annotation format. The phrases are labelled with character as well as token indices.

mulate annotations of every phrase symbolizing an entity with one or more 3D objects.

A phrase in our dataset could be one of the following:

- Noun, such as *bed*
- Noun phrases, such as *a brown desk*
- Noun phrases with modifiers, such as *a black chair in the corner*
- Subjective & Objective Pronouns, such as *it* and *they*
- Demonstrative Pronouns, such as *this*
- Indefinite Pronouns, such as *something*

The above phrases are known as referring expressions. Our annotation tool uses the start and end-character offsets to represent each referring expression. We also provide token indices for the annotated phrases, which are obtained by aligning character indices to token indices executed using spaCy library[1].

Each referring expression is linked with one or more 3D objects, and the link is defined using the labels and unique 3D object IDs. These IDs could be used to obtain the object segment masks and 3D bounding box coordinates for the objects. Our dataset contains many-to-many phrase-to-object mappings, where each phrase refers to one or more objects in a 3D scene.

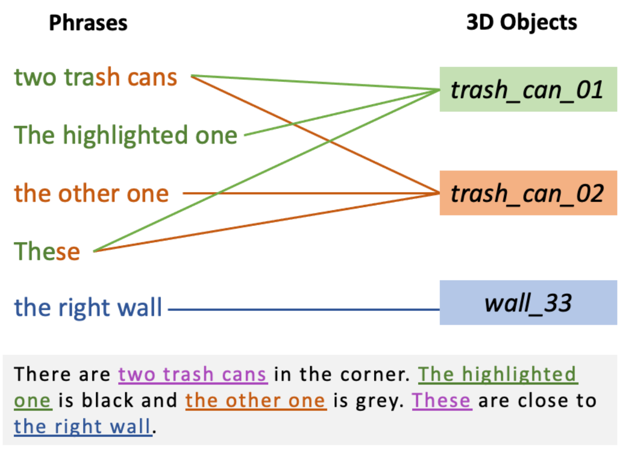
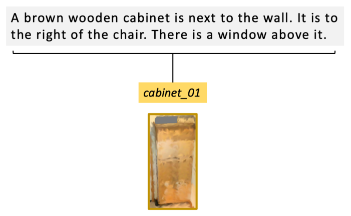
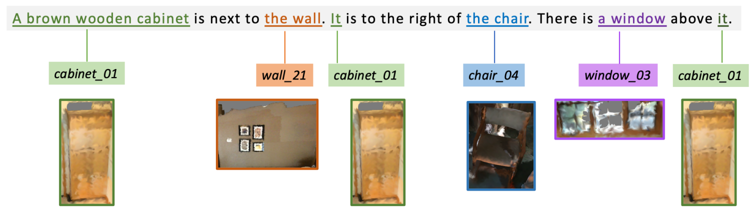

Phrases
3D Objects
two trash cans
The highlighted one
the other one
These
the right wall
trash_can_01
trash_can_02
wall_33
There are two trash cans in the corner. The highlighted one is black and the other one is grey. These are close to the right wall.

Figure 3.2: Many-to-many relations in DenseRefer3D dataset. Phrases characterizing multiple entities are linked with more than one object.

[1]`https://spacy.io/`


A brown wooden cabinet is next to the wall. It is to the right of the chair. There is a window above it.
cabinet_01

(a) A sample annotation from the ScanRefer dataset [11]. In this, each description is annotated with one 3D object.


A brown wooden cabinet is next to the wall. It is to the right of the chair. There is a window above it.
cabinet_01
wall_21
cabinet_01
chair_04
window_03
cabinet_01

(b) A sample annotation from the proposed DenseRefer3D dataset. In this, all the referring expressions are explicitly annotated with 3D objects.

Figure 3.3: Comparison of ScanRefer [11] and DenseRefer3D annotations. DenseRefer3D dataset consists of dense mappings between phrases and 3D objects.

3.2 Annotation Tool

This section provides a detailed walk-through of the annotation tool we developed for creating a new dataset with the desired annotations. We begin by examining our approach to the feature development process, including a discussion on constituting the central feature set for our specific task. In doing so, we fundamentally present the rationale for the following two questions:

1. What are the essential features the tool must include to collect the proposed annotations? and,
2. What are some additional capabilities that may help us improve the quality of annotations during and after the collection process?

We then deliver a thorough tour of various tasks that our tool is built to carry out efficiently, including annotation and verification of the data. We inspect all the *necessary* and *advanced* features by focusing on the tasks for which these functionalities were designed to assist. In addition, we cover several administration controls available in the tool to help manage the flow of the dataset collection process effectively. We end this section by providing the technical details of developing our tool.

3.2.1 Development Philosophy

To collect the desired annotations, we use descriptions and 3D scenes from ScanRefer [11] and ScanNet [20] datasets, respectively. Thus, our annotation tool's first and foremost requirement was to allow the rendering of 3D scenes and descriptions together in the same interface. The tool's design revolved around finding the right balance of the rendered interface area occupied by the textual and visual components to enable effective interactions between the two modalities. Additionally, we focused on allowing effortless navigation of objects in rendered 3D scenes using a computer mouse.

In order to obtain phrases-to-objects mappings, our development efforts involved enabling the selection of phrases of any length in the given description and linking the selected phrases with 3D objects. During the planning stage, we brainstormed several ideas for making our task easier to understand and faster to perform without compromising the annotation quality. As any external input can be prone to errors, we emphasized making the task more robust by automating numerous functionalities throughout the annotation process. Another central area of focus was the ability to outline the selected phrases and visualize the linked objects to help annotators quickly uncover any issues before submitting their work.

Furthermore, we desired to enhance the overall user experience to maintain a high retention rate for good annotators. This endeavour included planning on providing various advanced features in our tool to better aid experienced annotators with the task. We followed an iterative design and development strategy based on users' feedback to make continuous and timely improvements to our tool.

Lastly, since the data collection processes for curating large-scale datasets can be highly time-consuming, our design methodology was also tailored towards benefiting from crowdsourcing the data collection on a need basis. This approach consisted of devising instructions and interface elements that are simple to understand, quick to adapt, and easier to recall. Our tool embodies these decisions in the design of the two primary components of our collection effort: annotation and verification of the data.

3.2.2 Feature Overview

This chapter provides a comprehensive overview of the feature set in our annotation tool. We classify these features into two categories: *essential* and *advanced* features. The essential category consists of features necessary to collect quality annotations, whereas the advanced features further improve the user experience and make the overall data collection process more efficient.

3.2.2.1 Essential Features

Phrase Selection

The annotation tool supplies an intuitive way of selecting phrases, akin to the generally accepted method of highlighting text using a mouse. It follows the click-drag-drop approach, which involves placing the cursor at the start of the phrase, holding the left button on the mouse, dragging the cursor till the end of the phrase and releasing the pressed mouse button. After selecting, the interface highlights the phrase with a unique colour and attaches an object box directly under it. If the desired phrase consists of just one token, the tool enables a quick way of selecting the word by simply double-clicking on it using the left mouse button. Furthermore, the tool allows removing the selected phrase using the ⊖

button on the foremost object box under each phrase or using the *Reset* button to simultaneously clear all the selected phrases and linked objects under them. Moreover, the tool incorporates error-handling measures to prevent the cross-selection of phrases.

Link Objects

After selecting the desired phrase, the next step in the annotation process is to link one or more 3D objects with the selected phrase. This is performed by clicking on the object box under the selected phrase to spotlight it and double-clicking on the intended object from 3D scene. This operation populates the object box with the label and ID of the target object. The DenseRefer3D annotation tool displays the object name when hovering over the objects with the mouse, serving as an auxiliary layer for validating the objects described by the phrase.

In the case of phrases representing more than one object, the tool includes functionality to link multiple objects with a phrase, which can be accomplished using ⊕ on the main object box. The subsequent step of linking additional objects is similar to the one described above. The interface allows adding up to eight objects for each selected phrase and contains capabilities for removing the linked objects using ⊖ on the primary object box.

After linking the objects, the tool processes the annotations and updates the colours of the phrases based on the linked objects, i.e. it assigns the same colour to all the phrases that refer to the same object. Additionally, the tool makes it easy to update the linked objects by selecting the appropriate object box and double-clicking on the new intended object to confirm the action. Analogous to phrase selection, the annotation tool enforces measures to prevent multiple references to an object for a selected phrase.

Visualization

Visualization is a salient aspect of our annotation tool, acting as an observable confirmation technique for the users before submitting their work. The objects linked with the selected phrases can be visualized by hovering the mouse cursor over each object box. This action highlights the object in the same colour as the phrase and surrounds it with a gravity-aligned bounding box. Furthermore, all the linked objects could be highlighted simultaneously using *Show annotated objects* button. This activity, too, renders the object in the phrase colour to deliver visible cues for evaluating phrases-to-objects mappings in the annotation.

Comments

It is crucial to provide a way for the users to report any issues faced during the annotation process. These issues may include problems with scene rendering, spelling errors in the sentence, and incorrect labelling of 3D objects. We supply *Comments* button for documenting the problems encountered by the users. It is accessible for all the scene-description pairs and, when used, records metadata such as the description and the scene associated with the raised issue for diagnostic purposes. Additionally, after completing the task, we offer another channel for submitting the general feedback to help us improve the annotation tool and the overall data collection process.

Others

In addition to the features described above, we desired to enforce best practices to steer the users toward the intended goal. For instance, the users can navigate to the next assigned scene and description only if the current one has been annotated appropriately per the task instructions. The *Submit & Next* button delivers helpful prompts based on the types of inaccuracies observed in the annotated data. The prompts include sufficient information about the blunders and directions for addressing them. For example, if the selected phrase is not linked with any 3D object, the tool alerts the users about the problem and presents the necessary steps for linking the objects with the selected phrase.

3.2.2.2 Advanced Features

Locating Objects

The annotators can use the mouse to locate the objects described by the phrases in the description, and hovering over an object reveals the label associated with it. However, it might sometimes become challenging to identify the target object for various reasons. For instance, the target object might be small and occluded by a considerably larger object, or it may not be decidedly visible due to incomplete scene rendering. To overcome these challenges, we include an *All objects in scene* menu that lists all the available objects in the scene. The annotators can click on any object in the scrollable list to highlight it in the 3D scene with a bounding box rendered around it. This menu also contains a search bar to quickly filter objects based on the input query.

After locating the target object, the primary way to link it with the selected phrase is by double-clicking on it. Additionally, the tool enables another mode for linking the object by double-clicking on the object label listed in the *All objects in scene* menu. This operation, too, respects the colours assigned to the selected phrases by spotlighting the objects using the same colours. This menu serves as a supplementary approach for identifying the objects and furnishing desired annotations.

Keyboard & Mouse Shortcuts

It is vital to design and develop techniques that help make the overall annotation task more efficient. To realize this objective, we created several keyboard and mouse shortcuts in our tool to enhance productivity when executing the required operations. Some of the operations performable using keyboard shortcuts include adding more object boxes to link multiple objects with the selected phrase using *A* key, removing objects using *R* key, and highlighting all the linked objects at once using *C* key. The quick functions based on mouse controls include rotating, moving and zooming on the 3D scene using the mouse's *left-click*, *right-click*, and *scroll wheel*, respectively.

Correction Measures

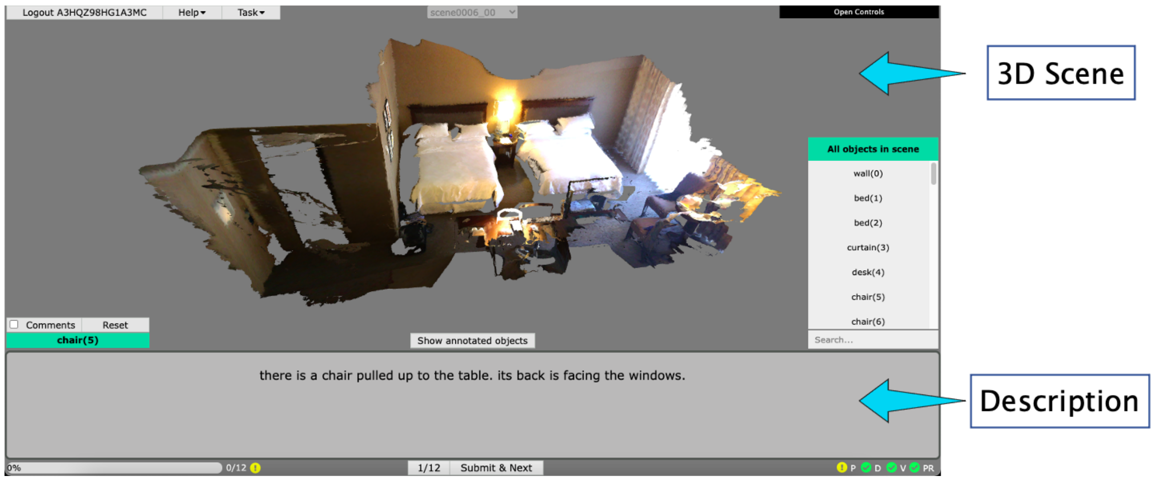
A good annotation tool consists of features that assist in fixing mistakes occurring during the annotation process. Following these design principles, we include options for rectifying the errors using the *Go Back* and *Start Over* buttons. The first button helps navigate back one step at a time to the previous scene-description pair to correct the inaccuracies. The second button is convenient for restarting the task as it brings the annotator back to the first assignment. These measures help the users improve their annotations and increase the prospects of their work acceptance during evaluation.

Quick Access to Documentation/Guidelines

The task instructions can be easily accessed anytime throughout the data collection process using the *Instructions* button. It opens the instructions in a pop-up window, thus preventing any disruption to the current annotation interface and allowing the users to refer to the documentation while working on the task simultaneously. Similarly, the tool offers many annotation examples and general feedback for activities to steer clear of using the *General Guidelines* button. Any significant updates to the task instructions and the tool's user interface are delivered using prompts at the beginning of the task.

Others

Our annotation tool incorporates functionality to control the rendered scene, such as adjusting the material's transparency and calibrating the speed of camera zooming. These actions can be accomplished using *Opacity* and *Zoom Speed* settings, respectively. Furthermore, the tool makes notable use of indicators during the annotation process. We utilize indicators for pronouns, determiners, verbs, and prepositions. These indicators signal the inclusion or exclusion of the respective word types. For example, a yellow exclamation indicator for pronouns denotes that the user did not annotate all the pronouns in the given description. On the other hand, a green tick indicator for verbs signifies that none of the selected phrases contain any verbs, which is part of the requirements. This helps the annotators avoid common mistakes by prompting them to include or exclude the appropriate word types in the selected phrases. Additionally, the tool has a progress bar to reveal the status of completed and remaining assignments.



there is a chair pulled up to the table. its back is facing the windows.

Figure 3.4: Annotation interface for Amazon Mechanical Turk users. The 3D scene and natural language description are rendered in the same interface, with additional useful features available to help with the annotation process.

3.2.3 Annotation Interface

Our annotation interface is designed with specific interactable components to facilitate coherent data collection. We developed the interface by keeping our primary end-user in mind. These design preferences are reflected throughout the interface, from scene rendering to selecting phrases and linking objects with them.

As the user logs in to the annotation interface, they are presented with a 3D model of an indoor scene consisting of many everyday objects and natural language sentences describing one of the objects. A significant portion of the main user interface is dedicated to rendering the 3D model. This provides adequate space for the annotators to easily move the scene around or take a closer look at the objects by zooming in on them. The tool enables the manipulation of the rendered scene in various ways. For instance, users can zoom in and out using the scroll wheel on the computer mouse. Furthermore, they can pan the 3D model along either side of horizontal and vertical axes using the right mouse click and rotate it to their preferred orientation with the left mouse click. The rendering of 3D models in our tool is implemented using three.js[2] library, further details on it are provided in Section 3.2.6.

The users are instructed to locate the objects depicted in the given sentences. As described in the Locating Objects section, the interface allows multiple methods for finding the objects in the scene. The most instinctive way is by simply moving the mouse cursor over the 3D objects. This action highlights all the object pixels under the cursor and draws an oriented bounding box (OBB) surrounding it. The highlighted object's label is revealed as a tooltip if

[2]`https://threejs.org/`

the user maintains the cursor on the object for a short moment. The other mode of locating the objects is using the *All objects in scene* menu, which can pinpoint the object's location when the user clicks on an object label from the list. This method is beneficial when dealing with cluttered 3D scenes, which might make it hard to find some objects.

Once all the described objects are discovered, the user is required to select all the phrases representing any physical objects. The process of selecting the phrases is similar to the universal way of making a text selection with the mouse. The Phrase Selection section provides more details on this. Each selected phrase is allotted a unique colour at the start and an empty rectangular field, an object box, under it. This box serves as a container for the 3D object to be linked with the selected phrase. The annotators can insert additional sub-boxes using the ⊕ button on the box.

The next step in the annotation process is to link object(s) with each selected phrase. We explain this operation in the Link Objects section, which must be performed for all the selected phrases. The annotators can verify the phrase-object association when the object box is auto-populated with the target object's label and ID.

The colouring of the selected phrases plays an essential role in our tool. The interface initially assigns a unique colour to each selected phrase. While linking objects, the tool determines if the new object is already associated with another phrase and updates the colour of all phrases linked with the same object. This enriches the user experience by signalling that the identical-coloured phrases represent the same object in the scene.

Once all the object boxes are populated, the users can finalize their selections for all the annotated objects by visualizing the phrases-objects linkages using the methods described in the Visualization section.

Moreover, the tool lets users quickly fix issues with their selected phrases and linked objects. For instance, if the user makes a mistake in their phrase selection, they can directly remove the phrase by clicking on ⊖ button under the phrase or clear all phrases at once using the *Reset* button. To edit the linked objects, they can click on the object box to be updated and execute the above-described action of linking the new object. We also encourage the users to provide feedback about any errors in the description or the scene using the *Comments* button.

Most of the steps described above can be carried out using the keyboard shortcuts available in the interface. For example, the user could display all the objects simultaneously, same as clicking on *Show annotated objects* button, by pressing the *C* key. After ensuring their annotations are reasonable, they can submit their work using the *Submit & Next* button and begin annotating the subsequent description and 3D scene. At the end of the task, users are presented with a feedback form to provide valuable suggestions or report any issues faced during the process.

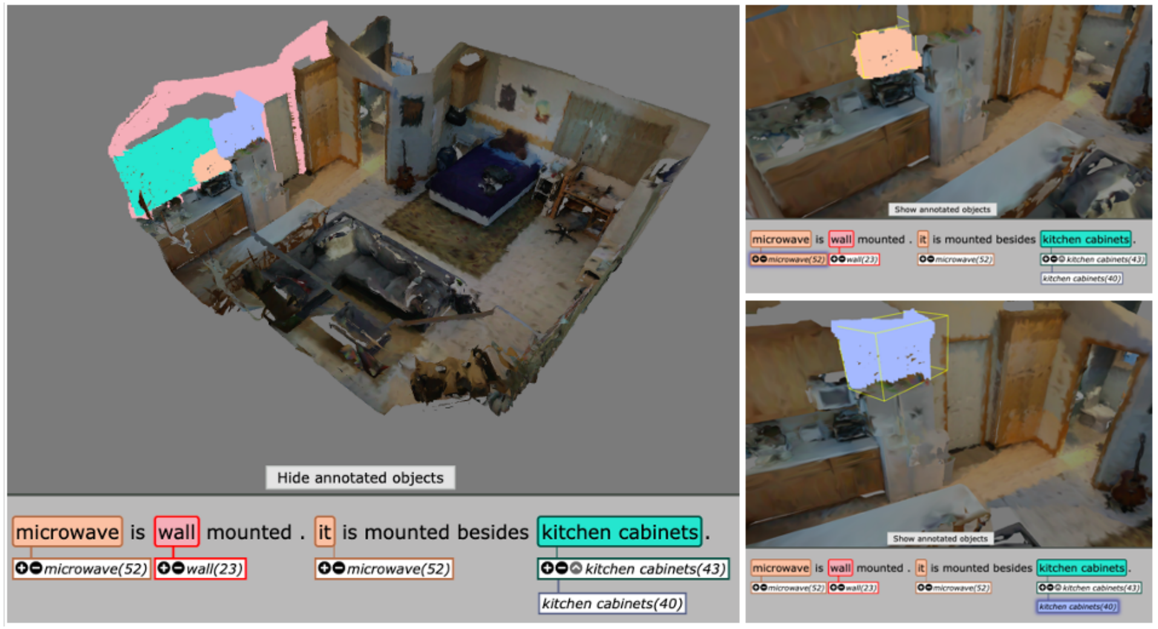


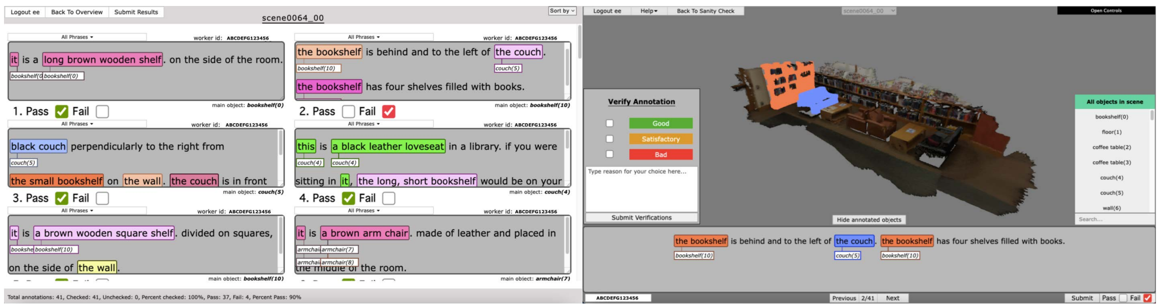
Figure 3.5: Illustration of an annotated description in our dataset. The objects linked with selected phrases can be visualized one at a time or simultaneously in the tool. The linked objects are highlighted with the same colour assigned to the phrase.

3.2.4 Verification Interface

The annotation interface boosts a user-friendly, worker-focused process of collecting quality data. These annotations are stored in the database and can be exported to the desired format. However, we needed the ability to verify the annotated data in an easy-to-use interface without exiting the tool to access the database manually. With that objective in mind, we designed a verification interface within the tool that allows us to visualize and verify the submitted annotations effortlessly. As discussed in greater detail in Section 3.4.5, our verification strategy consists of two modes: *Sanity-check* and *Second-stage thorough evaluation*.

The sanity check phase is intended for detecting prominent errors in the selected phrases. In this mode, the interface only displays the textual part of the annotated dataset to quickly inspect typical mistakes, such as missing annotations of some referring expressions in the description or inaccurate phrase selection containing undesirable words or phrases. This fast manner of validating annotated phrases allows us to approve the submitted assignments efficiently and, if necessary, provide feedback to the users so that they can avoid making similar errors in their future attempts.

The next mode of verification is a thorough evaluation. Here, besides outlining the annotated phrases, it provides support for visualizing the phrase-object relationships. The visualization is helpful to ensure that the objects linked by the users corroborate the given



scene0064_00

worker id: ABCDEFG123456

it is a long brown wooden shelf. on the side of the room.
bookshelf(0) bookshelf(0)

1. Pass ✓ Fail ☐ main object: *bookshelf(0)*

worker id: ABCDEFG123456

the bookshelf is behind and to the left of the couch.
bookshelf(10) couch(5)
the bookshelf has four shelves filled with books.

2. Pass ☐ Fail ✓ main object: *bookshelf(10)*

worker id: ABCDEFG123456

black couch perpendicularly to the right from
couch(5)
the small bookshelf on the wall. the couch is in front

3. Pass ✓ Fail ☐ main object: *couch(5)*

worker id: ABCDEFG123456

this is a black leather loveseat in a library. if you were
couch(4) couch(4)
sitting in it, the long, short bookshelf would be on your

4. Pass ✓ Fail ☐ main object: *couch(4)*

worker id: ABCDEFG123456

it is a brown wooden square shelf. divided on squares,
bookshe bookshelf(10)
on the side of the wall.

main object: *bookshelf(10)*

worker id: ABCDEFG123456

it is a brown arm chair. made of leather and placed in
armchai armchair(7)
armchai armchair(8)
the middle of the room.

main object: *armchair(7)*

Total annotations: 41, Checked: 41, Unchecked: 0, Percent checked: 100%, Pass: 37, Fail: 4, Percent Pass: 90%

scene0064_00

bookshelf(0)
floor(1)
coffee table(2)
coffee table(3)
couch(4)
couch(5)
wall(6)

the bookshelf is behind and to the left of the couch. the bookshelf has four shelves filled with books.
bookshelf(10) couch(5) bookshelf(10)

ABCDEFG123456 2/41 Pass ☐ Fail ✓

Figure 3.6: Two-stage verification interfaces in the DenseRefer3D annotation tool. The first stage facilitates a quick review of the annotated phrases, whereas the second stage provides a comprehensive environment for thoroughly examining the specified phrases-to-objects mappings.

description. In other words, it enables us to quickly highlight the linked objects to verify their colours, shapes, sizes, position, and location concerning other objects in the scene. The verification interface consists of three straightforward options for validating an annotation: *Good*, *Satisfactory*, and *Bad*. The interface necessitates the users to include an explanation for their choice if they mark the annotation as *Satisfactory* or *Bad*. This enforcement gives us adequate information necessary to fix or discard the annotation.

The interface follows similar design principles implemented in the annotation view. For instance, it respects the colouring of the phrases by highlighting them in the same colour if they are linked with an identical object. The keyboard shortcuts are adapted to the requirements of this verification mode, allowing users to perform many actions directly via the keyboard input. They have access to the same scene manipulation controls discussed in the previous sections. However, we prevent them from changing either the selected phrases or the linked objects to avoid unintended modifications.

Similar to our ideology behind the annotation interface, we designed the verification view for end-users on crowdsourcing platforms, which is reflected in the simplicity of accessible options. This approach enabled us to move the second-stage evaluation to crowdsourcing venues and publish it as a set of assignments if needed. In doing so, we added instructions for verifying the submitted data and an assessment test to ensure they understood the task.

This two-stage verification strategy is crucial in making our task better streamlined while providing an intuitive way of visualizing and verifying the obtained annotations. Moreover, our design principles allow us to, if necessary, adopt an end-to-end approach for crowdsourcing the entire collection process, from data annotation to verification.

3.2.5 Administration Interface

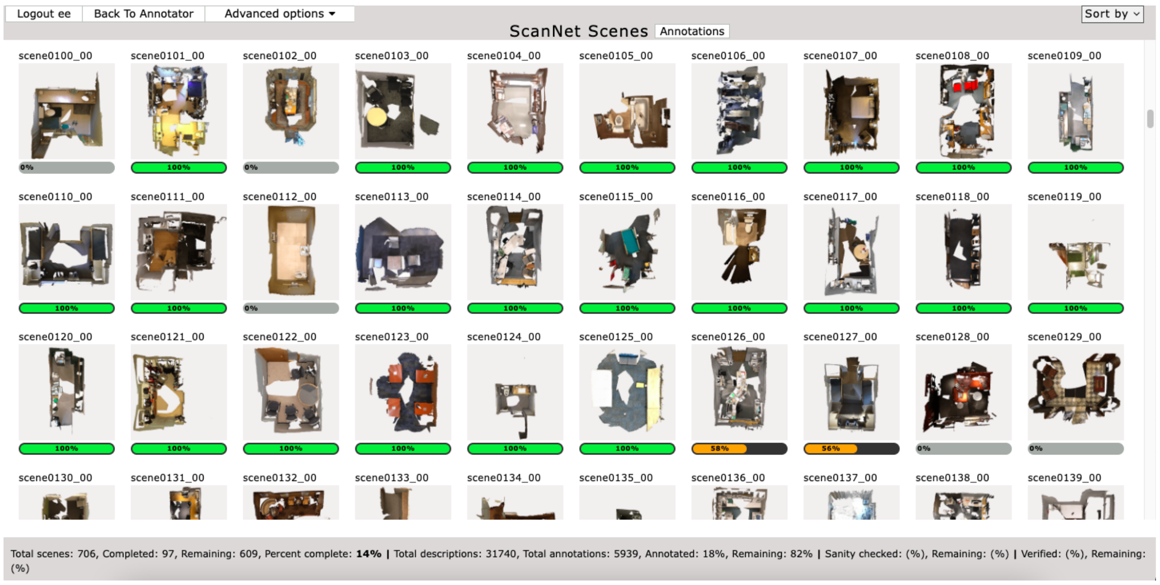
A good annotation tool, and broadly any good piece of software dealing with the collection and verification of data, should include the functionality to administer these processes effectively. To adhere to these well-accepted software principles, we developed an interface to supervise the entire task. Our administration interface provides an overview of the progress in data annotation and verification processes. This interface lists all 3D scenes from ScanNet dataset [20] in a compact view with a progress bar under each scene to monitor the number of descriptions annotated for a scene. In addition, we provide extra statistics, such as keeping tabs on the descriptions with multiple annotations, as one may wish to acquire more than one annotation per description for cross-validation purposes. These statistics are also available for both *Sanity-check* and *Second-stage* modes of verification.

This view provides quick access to the *Sanity-check* verification mode by clicking the scene thumbnail. As discussed earlier, the *Sanity-check* mode only displays the annotated phrases without rendering the 3D scene. However, it shows the object names linked with each annotated phrase for verifying if the object labels are comparable with the phrases that represent them. For instance, the phrase *A trash can* may be linked with an object labelled *dust bin*. In other words, the object described by the phrase and the object label may not always agree with one another verbatim, but displaying this link can give us an idea about the annotated association. In the above example, *A trash can* and *dust bin* usually represent the same physical object. Furthermore, the interface contains a scrollable drop-down list comprising all the selected phrases to easily separate the annotated ones from the rest for long descriptions. For reference, the interface displays the target object label from ScanRefer dataset [11] under each annotation to verify it with the linked objects. After evaluating the phrases and linked object labels, the reviewer marks the annotation as either *Pass* or *Fail*. The tool stores the results of this assessment in the database.

The administrators have access to the *Second-stage* verification mode directly from the sanity-check view by clicking on the scene's title. This view is slightly different from the verification interface discussed in the previous section and was designed primarily for task administrators. It sequentially displays all the annotations in the current scene with controls for moving between different annotations. In this, we allow the user to modify the linked objects, which we discuss further in Section 3.4. Essentially, the tool provides administrators with easy access to both verification modes.

Feedback from our end-users is vital for improving the quality of our processes. The administration interface supports capabilities for displaying the feedback collected during the annotation and verification tasks. Additionally, it is often useful to have the ability to prohibit some users from working on our task. The tool consists of a user-management interface that conveniently enforces user access rights using blocking and unblocking mechanisms.

Moreover, the administrators have access to the *Edit Annotations* interface for fixing errors in the collected annotations without requiring direct modifications to the database documents. The organizational features discussed above are crucial in determining the suc-



Total scenes: 706, Completed: 97, Remaining: 609, Percent complete: **14%** | Total descriptions: 31740, Total annotations: 5939, Annotated: 18%, Remaining: 82% | Sanity checked: (%), Remaining: (%) | Verified: (%), Remaining: (%)

Figure 3.7: Administration interface of DenseRefer3D annotation tool. It displays an overview of the progress in annotation and verification tasks. Each scene is represented with a small thumbnail and a task-specific (i.e., annotation and two verification modes) progress bar under it.

cess of different processes. The features are only accessible to the permitted users with *admin* role assigned to them, thus preventing unauthorized entrance to administrative controls.

3.2.6 Implementation Details

In this section, we cover the implementation details of our tool by discussing various technologies involved in the development process.

Our annotation tool is a web-based application accessible via an internet browser. Where necessary, we use the great work published by the amazing open-source community. Our build platform of choice was Node.js [31], an open source environment for creating event-driven web servers using V8 JavaScript engine [33]. We employ Express.js [27] framework to enable several client-server interactions. This includes setting up middleware to handle HTTP requests and manage routing based on user performance on the qualification test, fetching 3D scenes and descriptions for annotation tasks, and securely storing the completed work in the database. We utilize MongoDB [59] database for all our annotation data storage and retrieval requirements. It supplies drivers for the Node.js environment, allowing us to perform several tasks involving database access seamlessly.

In addition to the primary target users on the Amazon MTurk platform, our tool supports multiple content access levels based on assigned user roles for additional functionalities. These user roles include *MTurk annotators* and *verifiers*, *in-house registered users*, *administrators*, and *root users*. We utilize Passport [37] middleware package to efficiently define the authentication strategies for different user roles. Furthermore, we use logging throughout the tool to keep track of various user events and to identify and resolve issues. We employ Winston [93] logger for configuring different logging levels based on the event's severity.

All the interface views across the tool are rendered using EJS [25] template engine to enable fast rendering and more accessible debugging capabilities. Since our dataset consists of detailed RGB-D 3D scenes, we make use of three.js [63], a JavaScript library that facilitates the rendering of 3D graphics directly in a web browser using WebGL [35]. The library provides an efficient way of adding and manipulating 3D objects, adjusting the lighting to focus on a specific section or the whole scene, and enabling camera controls to update the field of view effortlessly.

For object picking, we use the raycasting method provided by Raycaster in three.js, which projects a ray from the mouse's position to the scene for determining the object to be picked. This is essential for our task since the users are required to locate the objects described in the provided sentences for annotating the relevant phrases. The tool highlights the 3D object under the mouse cursor and displays a gravity-aligned oriented bounding box around it. Since 3D scenes from the ScanNet dataset [20] are available in polygon file format (PLY), we use three.js PLYLoader for parsing the 3D meshes. However, we apply our modifications to read the ScanNet object IDs for the available objects.

The tool makes it possible to perform several actions for controlling the 3D scene, such as panning, rotating, or zooming, with the help of OrbitControls extension. Moreover, the tool consists of additional scene interactions implemented using Dat.GUI [86] library. This menu lets users fiddle with controls such as adjusting the opacity of the material and zoom speed.

We designed an intuitive mechanism for selecting the phrases and linking objects with them. This method, developed purely in JavaScript, assigns a unique colour to the selected phrases and records the text selections and their start and end character indices in the sentence. Additionally, the tool allows linking up to eight objects for each selected phrase, with controls available for quickly removing or updating these objects if needed. When an object is linked with a selected phrase, it is assigned the same colour as the phrase. In addition, if two or more phrases are annotated with the same object, the colour of these phrases is updated to reflect their reference to the same object. This colour-based visualization helps the users easily distinguish between phrases-to-objects mapping pairs.

Lastly, we utilize the tippy [65] library to create helpful tooltips for displaying useful information regarding several interactable elements. Moreover, the tool delivers timely notifications, such as valuable prompts to direct users toward the desired result, offer specific in-task instructions, and provide details on supported keyboard & mouse shortcuts in the form of pop-up alerts implemented using the SweetAlert [24] library.

3.3 Data Collection Process

This chapter describes all of the stages of our data collection process. We begin by providing details about the configuration of our entire infrastructure, including the organization of our task. We then present the steps involved in managing annotations submitted by the workers on the crowdsourcing platform. Finally, we conclude this chapter by sharing quantitative statistics about the data collection task.

3.3.1 Infrastructure Setup

We developed the DenseRefer3D annotation tool to create an optimized and readily available environment. We employ various open-source and proprietary services to achieve the desired efficient setup. The infrastructure setup consists of the following:

Annotation Server Setup

We set up our DenseRefer3D annotation server[3] using Node [31]; more specifically, we utilize Express [27] framework to define the middleware functions for handling various routing requests from the client. Some of these operations include assessing qualification test performance, assigning scene-description pairs for annotation, storing annotations in the database, redirecting users based on their roles, and providing access to other administrative tasks. As mentioned earlier, we rely on MongoDB database [59] to store annotations and supplementary data. It enables effective application-database interactions for various data storage and retrieval actions through the provided official Node driver. Furthermore, we use Nginx [68] as our front-end server for proxying the requests to the Node server and for load balancing to process concurrent user traffic while delivering high performance. Moreover, it allows us to redirect the HTTP requests to HTTPS to enforce an encrypted connection. This is essential since the Amazon Mechanical Turk platform mandates the external website to use an SSL certificate with HTMLQuestion data structure. We deploy our front-end Nginx Server, DenseRefer3D Node server, and MongoDB database on an Amazon Elastic Compute Cloud (EC2) [17] instance. This setup enables us to deliver a highly available, scalable, and reliable application to serve many users at the same time effectively.

Data

As mentioned before, we curate our desired dataset using the descriptions and 3D scenes from ScanRefer [11] and ScanNet [20] datasets, respectively. We store the descriptions on a local instance of MongoDB running on an EC2 virtual machine. This provides a lag-free retrieval of descriptions during the annotation process. The 3D dataset is stored on a secure internal server and loaded directly using the available URLs for ScanNet scenes. In

[3]`https://github.com/3dlg-hcvc/dense-scanrefer-annotator.git`

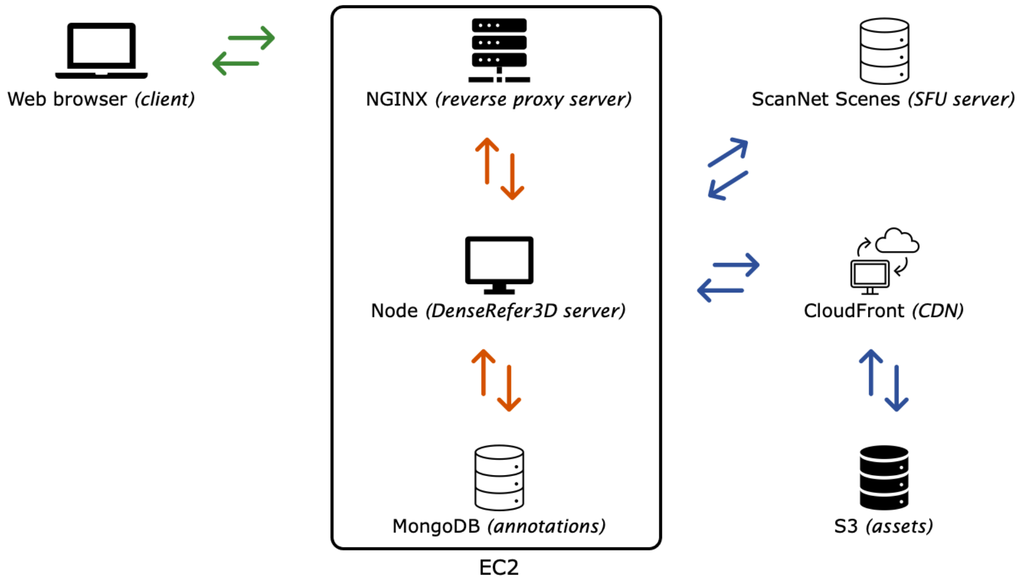

Web browser (client)
NGINX (reverse proxy server)
Node (DenseRefer3D server)
MongoDB (annotations)
EC2
ScanNet Scenes (SFU server)
CloudFront (CDN)
S3 (assets)

Figure 3.8: Overview of the infrastructure setup in our DenseRefer3D annotation tool. We deployed a low-cost, highly optimized framework to deliver reliable performance and enable concurrent access to our task.

addition, we use Amazon Simple Storage Service (S3) [82] to create storage buckets for saving sizable additional assets required during the data collection, such as the resources necessary to render the qualification test and scene thumbnails for populating the metrics in the overview page. The S3 service comes with the benefits of high data availability, faster retrieval performance, and increased data security.

Requests

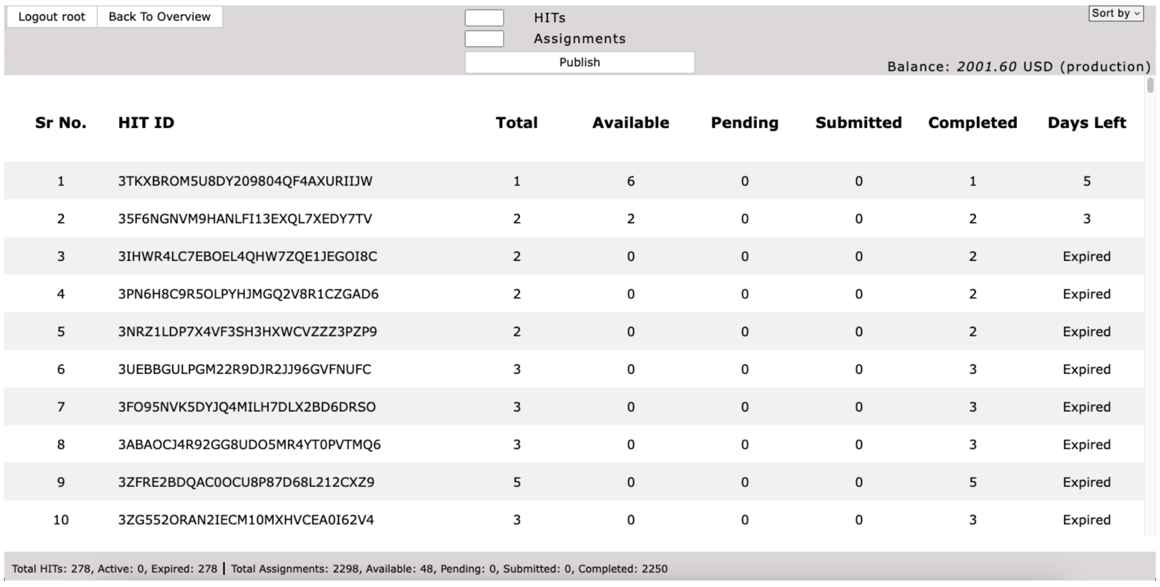
Our comprehensive tool provides access to various operations through the request calls, which are handled efficiently by our DenseRefer3D server. The number of such requests grows with the additional functionalities required to deliver the desired tool. This can lead to higher costs due to the limited number of requests available with the free tier of Amazon Web Services [83] account. To address this, we employ Amazon CloudFront [18], a content delivery network (CDN) service also included in the AWS free tier, allowing two million requests per month. Besides the availability of many free requests, CloudFront delivers faster data transfers and low latency, enabling us to restrict any direct access to the data stored in the S3 bucket. Furthermore, it caches the data at locations close to the user, thus serving the requests with fewer delays and enabling load management on the origin server. This highly optimized setup helps us significantly reduce overall expenses while facilitating efficient communication between our servers and concurrent clients.

3.3.2 Task Setup

We use Amazon Mechanical Turk (MTurk) [41] as our favoured crowdsourcing platform to recruit the annotators for curating our dataset. We chose MTurk due to the large availability of workers on the platform and access to different task operations made possible using AWS SDK [79] and MTurk API [4] available for popular web development languages. The following sections provide more details on workflows involved in designing and publishing tasks.

Creating Tasks
A task on the MTurk platform is called a Human Intelligence Task (HIT), with each HIT consisting of one or more assignments for annotators to accept. A worker is only permitted to accept at most one assignment per HIT. We utilize AWS Python SDK (Boto3) [30] to manage several operations related to creating tasks. These include creating HITs with the intended number of assignments, reviewing submitted assignments, messaging workers, and making bonus payments. The recommended method of creating many HITs is first to create a HIT type. A HIT type lets us specify the standard properties pertinent to each HIT. The properties include assignment duration, the reward amount to be paid on successful completion of an assignment, description and title of the task, and common search keywords for discovering our task on the Amazon Mechanical Turk platform. Additionally, it allows us to define specific qualification requirements such as the worker's approval rate, the number of approved assignments, and the preferred region of the worker. We conducted various experiments to find the optimal values for these parameters by publishing small trial batches of tasks. The assignment approval rate and count of total approved assignments provide access to high-quality workers. At the same time, the locale values allow us to restrict our target audience to countries where English is the predominant language. This control is essential since the ScanRefer [11] dataset contains descriptions written in natural English.

Publishing Tasks
Similar to creating new HITs, we rely on MTurk APIs [4] to publish the tasks on the platform. Since our task requires creating numerous HITs for annotating around 50,000 descriptions from the ScanRefer [11] dataset, we wrote scripts to automate creating new HITs in large quantities at once. It supports parameters such as the required number of HITs, maximum assignments for each HIT, and the contents of our HIT. We use the HTMLQuestion data structure to describe the HIT contents. It enables us to deploy our annotation server for hosting the task without implementing methods for processing the HIT-related information for the assignments. This is useful due to the complexities involved in our task. Our server automatically generates many unique scene-description pairs for the annotators. The automation helps maintain the coverage of available descriptions and scenes by prevent-



Balance: *2001.60* USD (production)

Sr No.	HIT ID	Total	Available	Pending	Submitted	Completed	Days Left
1	3TKXBROM5U8DY209804QF4AXURIIJW	1	6	0	0	1	5
2	35F6NGNVM9HANLFI13EXQL7XEDY7TV	2	2	0	0	2	3
3	3IHWR4LC7EBOEL4QHW7ZQE1JEGOI8C	2	0	0	0	2	Expired
4	3PN6H8C9R5OLPYHJMGQ2V8R1CZGAD6	2	0	0	0	2	Expired
5	3NRZ1LDP7X4VF3SH3HXWCVZZZ3PZP9	2	0	0	0	2	Expired
6	3UEBBGULPGM22R9DJR2JJ96GVFNUFC	3	0	0	0	3	Expired
7	3FO95NVK5DYJQ4MILH7DLX2BD6DRSO	3	0	0	0	3	Expired
8	3ABAOCJ4R92GG8UDO5MR4YT0PVTMQ6	3	0	0	0	3	Expired
9	3ZFRE2BDQAC0OCU8P87D68L212CXZ9	5	0	0	0	5	Expired
10	3ZG552ORAN2IECM10MXHVCEA0I62V4	3	0	0	0	3	Expired

Total HITs: 278, Active: 0, Expired: 278 | Total Assignments: 2298, Available: 48, Pending: 0, Submitted: 0, Completed: 2250

Figure 3.9: Administration interface for creating new HITs. This allows us to quickly publish many assignments at once on MTurk platform. Additionally, it displays statuses of all the existing assignments, which can be filtered using the several sorting options.

ing the annotators from working on previously annotated data. Furthermore, it manages incomplete assigned scene-description pairs by adding them back to the available pool for future assignments. As a security measure, our tool generates a unique return code at the end of each assignment that the annotators are required to enter while submitting their work for review, which is verified to forbid the misuse of the task.

3.3.3 Assignment Evaluation

In addition to publishing tasks using MTurk API [4] as described in the previous section, we designed an interface within our tool that allows us to quickly create multiple HITs with the required number of assignments allotted to each HIT. This integration enables us to seamlessly perform most operations in the collection process from within the tool. This interface presents the task administrators with the statuses of the published HITs and the statistics to track the progress of assignments. This includes observing the number of available assignments for the workers, the number of submitted assignments requiring further review before approval or rejection, and the expiration date of HITs. These values help monitor the progress of the data collection and enable us to promptly add more assignments based on the requirements. The tool supports various sorting functionalities, such as displaying HITs by the number of submitted assignments or filtering assignments based on the days left, to enable the administrators to sift through the published HITs and focus on the most prominent information. Furthermore, it shows the current available balance in the requester's MTurk account. To create new HITs, the administrators must enter the number of HITs and assignments count for each HIT. Finally, they can confirm their selections and click on the *Publish* button to issue the HITs on the MTurk platform. All these actions are performed internally using the AWS JavaScript SDK [29] with support for logging and error handling enabled for each operation.

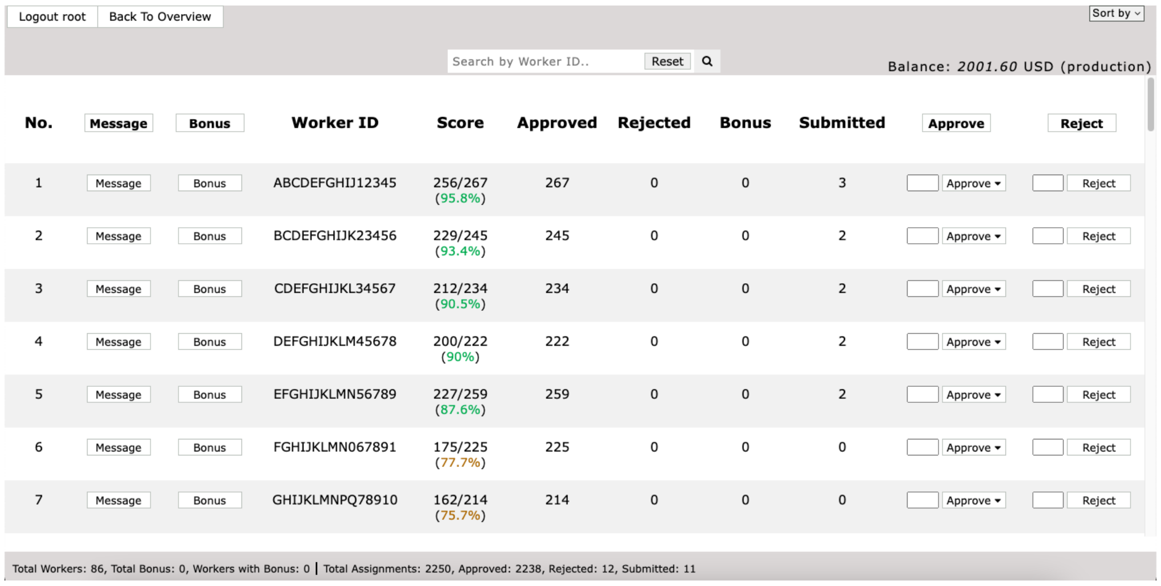
Evaluating submissions provided by the annotators is a crucial step in determining the quality of the collected data. It allows us to understand the complexities involved in the task and the appropriate measures we can perform to minimize them. For instance, during the review process, we can identify some of the blunders made by the workers and notify them accordingly so that they can avoid these mistakes in future submissions.

We follow a two-stage review strategy for evaluating the quality of the annotations. We detail the two-stage process in Section 3.4.5. Essentially, the first stage allows us to expedite the annotation assessment by displaying only the textual parts of the annotations. i.e., it does not render the linked objects but instead lists the object names along with the selected phrases. In most cases, the first stage serves as an adequate evaluation channel, enabling us to focus on the quality of the words or phrases included in the annotations. However, if the need arises, the tool provides an easy way to switch to the second stage of the verification process. In this stage, besides displaying selected phrases, the objects linked with them are also rendered in a 3D viewer for thorough analysis.

After reviewing the annotations, the administrators can switch to the *Review Assignments* interface. This interface displays the information directly associated with the workers and their assignment submissions. For example, it lists worker IDs and the number of rejected, approved and submitted assignments for each worker. Moreover, it shows the results of the review process discussed above by highlighting the scores obtained by each worker on their submissions.

The interface supports three actions for an assignment submission: reject, approve, and override. To approve (or reject) the assignments for a worker, the administrator must enter the number of assignments to be approved (or rejected) next to the *Approve* button in the same row as the worker ID and confirm this action by clicking on *Approve*. Furthermore, the previously rejected assignments can be quickly approved using the *Override Rejection* button.

In addition to executing these actions on the assignments, the interface contains additional functionalities for efficiently conducting the data collection process. After reviewing the assignments, we notify the workers of their achieved scores and provide general feedback based on the review. The tool comprises a *Message* button for each worker to inform them of feedback or send timely updates. The *Message* button next to the worker ID helps notify the intended worker. However, we might sometimes wish to notify more than one worker with the same message. Using the individual *Message* button next to each worker ID can


Logout root | Back To Overview | Sort by

Search by Worker ID.. | Reset

Balance: *2001.60* USD (production)

No.	Message	Bonus	Worker ID	Score	Approved	Rejected	Bonus	Submitted	Approve	Reject
1	Message	Bonus	ABCDEFGHIJ12345	256/267 (95.8%)	267	0	0	3	Approve	Reject
2	Message	Bonus	BCDEFGHIJK23456	229/245 (93.4%)	245	0	0	2	Approve	Reject
3	Message	Bonus	CDEFGHIJKL34567	212/234 (90.5%)	234	0	0	2	Approve	Reject
4	Message	Bonus	DEFGHIJKLM45678	200/222 (90%)	222	0	0	2	Approve	Reject
5	Message	Bonus	EFGHIJKLMN56789	227/259 (87.6%)	259	0	0	2	Approve	Reject
6	Message	Bonus	FGHIJKLMN067891	175/225 (77.7%)	225	0	0	0	Approve	Reject
7	Message	Bonus	GHIJKLMNPQ78910	162/214 (75.7%)	214	0	0	0	Approve	Reject

Total Workers: 86, Total Bonus: 0, Workers with Bonus: 0 | Total Assignments: 2250, Approved: 2238, Rejected: 12, Submitted: 11

Figure 3.10: Administration interface for reviewing completed HITs. Besides displaying annotation scores for each worker, it includes capabilities for approving, rejecting, or overriding assignments. Furthermore, it provides options for notifying workers and sending them bonus payments.

be tedious. So to help with that, the tool incorporates the feature to send notifications to multiple workers simultaneously using the *Message* button located in the header row.

As described in Section 3.4.2, workers must take a qualification test the first time they use the tool. We reward them with a one-time bonus based on the test results and additional criteria. The tool enables us to make payments effortlessly using the *Bonus* button in each worker-information row. Using this feature, the administrators can send the bonus and a message describing the reason for the payment to the intended worker. For security purposes, the bonus amount to send can be changed only by the *root user*. Similar to notifying multiple workers simultaneously, the tool also supplies the functionality for making concurrent bonus payments to multiple workers. This can be accomplished using the *Bonus* button in the header row.

Since our data collection undertaking is an exhaustive effort, this interface displays a sizeable amount of information regarding the submissions provided by the workers. Thus, it becomes essential to filter through the vast information to obtain the required results. The tool facilitates data sifting using the provided sorting functionalities similar to the sort features in the *Create Assignments* interface discussed above. Furthermore, this interface includes search capabilities to allow the administrator to spotlight the information based on the provided worker ID. The features discussed in this section play a crucial role in streamlining the assignment evaluation process.

3.3.4 Task Statistics

This section briefly summarizes the task analytics accumulated during the data collection process. A comprehensive quantitative breakdown of the collected annotations is outlined in Section 3.5.

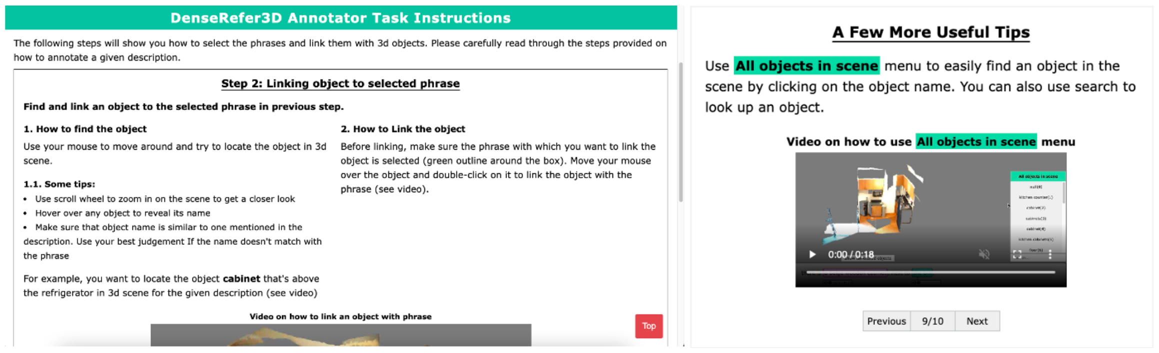
We conducted our data collection effort in multiple phases. We initially started with the testing phase, where we invited participants from the university to try out the annotation tool and provide feedback. This vital step allowed us to refine various facets of the process, including simplifying the task instructions presented at the beginning and paving the way for several valuable features we implemented in the tool. In this phase, we tested many standard MTurk operations, such as approving assignments, notifying workers, and sending bonus payments.

After testing our tool and collecting a small sample of annotations, we set the launch phase in motion. However, we decided to be cautious by publishing only a small subset of HITs at once. The idea behind this approach was to maintain the quality of annotations on par with our expectations. After completing the data collection process, we ended up with the annotations of 207,177 pairs using the nearly 50,000 descriptions and more than 700 3D scenes from ScanRefer [11] and ScanNet [20] datasets, respectively. Moreover, the *Sanity Check* verification stage resulted in between 85-90% average approval rate of annotations for the entire dataset.

Besides the annotation statistics, we collected additional information that helped improve our understanding of the whole process. Overall, 1,280 workers from the MTurk platform participated in creating the DenseRefer3D dataset. On average, the workers attempting our task for the first time took about 20 minutes to complete an assignment. The annotation time decreased significantly for the workers who chose to work on more of our HITs. The returning workers spent less than 14 minutes on each assignment. Additionally, we collected the time workers spent reading the instructions and the time it took to finish the qualification test. On average, a worker spent about 9 minutes going through the instructions and completing the qualification test. We published more than 500 HITs, with around 4,700 assignments. In the end, the total expenditure amount was upwards of 10,000 USD, with most of the payments going towards the reward for the approved assignments and sending one-time bonus payments to hundreds of workers.

3.4 Quality Control

This section highlights some of the quality control measures we enforced throughout the data collection process.



DenseRefer3D Annotator Task Instructions

The following steps will show you how to select the phrases and link them with 3d objects. Please carefully read through the steps provided on how to annotate a given description.

Step 2: Linking object to selected phrase

Find and link an object to the selected phrase in previous step.

1. How to find the object

Use your mouse to move around and try to locate the object in 3d scene.

1.1. Some tips:

- Use scroll wheel to zoom in on the scene to get a closer look
- Hover over any object to reveal its name
- Make sure that object name is similar to one mentioned in the description. Use your best judgement If the name doesn't match with the phrase

For example, you want to locate the object **cabinet** that's above the refrigerator in 3d scene for the given description (see video)

2. How to Link the object

Before linking, make sure the phrase with which you want to link the object is selected (green outline around the box). Move your mouse over the object and double-click on it to link the object with the phrase (see video).

Video on how to link an object with phrase

A Few More Useful Tips

Use **All objects in scene** menu to easily find an object in the scene by clicking on the object name. You can also use search to look up an object.

Video on how to use All objects in scene menu

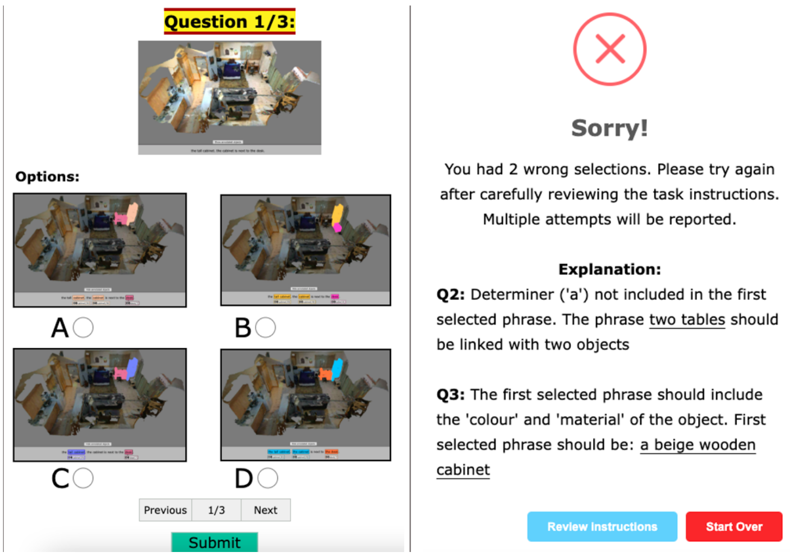
Figure 3.11: Task instructions for MTurk workers in our DenseRefer3D annotation tool. The instructions were divided into steps describing the required actions, such as selecting phrases or linking objects, with illustrated examples and video demonstrations. Furthermore, we included helpful suggestions to assist annotators in performing the task.

3.4.1 Task Instructions

Our task consisted of two types of instructions for the MTurk worker website and the DenseRefer3D annotation tool. The former was intended for new workers to make it easier to understand the task while keeping the instructions succinct. The objective was to help them get started with the task, provide information on accessing the annotation tool, and brief details on the offered reward amount. Once they began using our tool for the first time, it presented detailed step-by-step instructions for our task. The first step summarized the process of selecting a phrase. This included details on what constitutes a good phrase selection, with helpful tips for incorporating the object attributes while performing phrase selection. Furthermore, this explained how they could perform the phrase selection using the computer mouse. We even provided a helpful video to demonstrate this operation.

The next step exemplified linking 3D objects with the selected phrases. It started with a few recommendations on locating the intended object(s) referred to by the selected phrases, which in our tool can be done either through direct interaction with the 3D scene or using the *All objects in scene* menu to determine the target object(s). We followed this up by showing them steps to perform the phrases-to-objects association, i.e., linking their selected phrases with one or more 3D objects. Similar to the previous step, we included a demo video showcasing these actions.

The remaining steps served as additional guidelines by providing them with several examples, highlighting things to do and not to do, and instructions on offering us feedback throughout the process. Furthermore, it included demonstrating various mouse controls and keyboard shortcuts supported in our tool to make the task more efficient for the workers. For instance, one of the steps showed ways of linking more than one object with a selected phrase using the controls provided on the object box. At the same time, another step illustrated some error correction measures that the annotators can perform, such as removing



Sorry!

You had 2 wrong selections. Please try again after carefully reviewing the task instructions. Multiple attempts will be reported.

Explanation:

Q2: Determiner ('a') not included in the first selected phrase. The phrase two tables should be linked with two objects

Q3: The first selected phrase should include the 'colour' and 'material' of the object. First selected phrase should be: a beige wooden cabinet

Figure 3.12: Interface for conducting a one-time qualification test in our DenseRefer3D tool. Each question is presented with four choices illustrating existing annotations, requiring the workers to select the best annotation. The tool ensures that access to the annotation task is granted only if the workers achieve a minimum score on the test.

incorrect objects from the phrase-objects association and updating an existing linked object by replacing it with a different one. With an illustrative example, we also covered methods for visualizing annotations to verify their work before submitting it.

Most of the steps we detailed as part of the task instructions were accompanied by a demo video revealing the execution of the involved actions. Thus, the step-by-step instructions with several textual and visual examples helped make the task easier to understand and allowed us to obtain the desired annotations with minimal error rates.

3.4.2 Qualification Test

After reviewing the task instructions, the workers were directed to participate in a qualification test. The qualification test was a mandatory one-time quiz that helped us assess workers' understanding of our task based on reading the instructions in the previous step. In this, the tool generated three random questions laid out in the multiple-choice format. A question consisted of an image of a 3D scene and an unannotated sentence describing the objects from the scene. It was presented with four choices in the form of images, highlighting the existing annotations of phrases linked with 3D objects. They were required to select the most appropriately annotated image as their answer choice. After making their selections, the tool evaluated their responses and displayed the results. Furthermore, it explained the incorrect answers, if any, and provided links to additional instructions. They were only permitted to commence working on our task if at least two out of the three answers were correct. If they scored less than that, the tool prompted them to retake the qualification test. On every re-attempt, the tool randomized the test questions and multiple choices to prevent them from memorizing the correct answers. In the end, we sent out a bonus as a reward for completing the quiz, provided they met specific criteria. For example, one of the conditions to be eligible for a bonus was correctly answering all three questions on their first attempt.

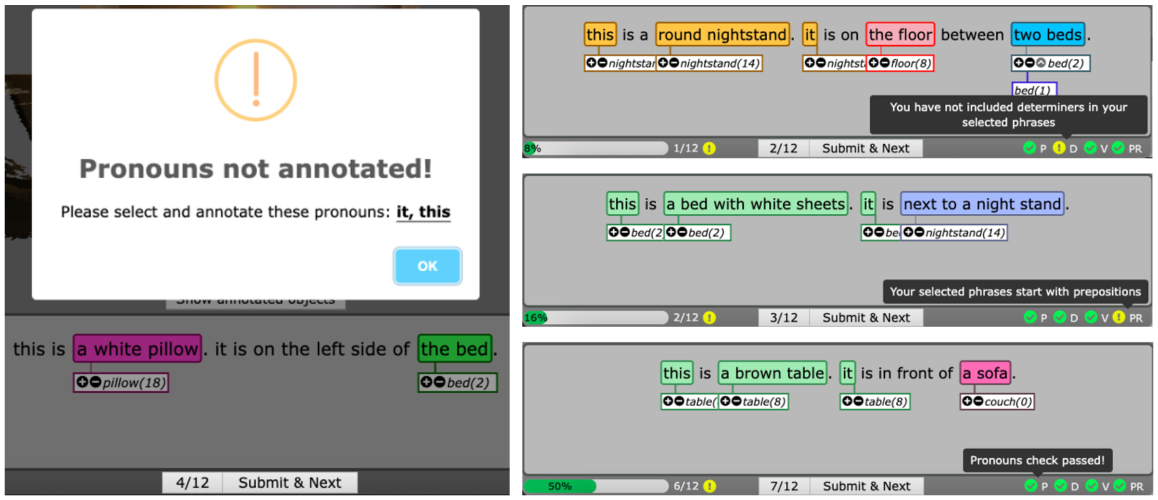
We kept track of each worker's time spent reading the task instructions and taking the qualification test. This helped us determine the appropriate bonus amount based on the computed average time. The questions on the test covered diverse types of annotations, including some edge case scenarios. For instance, one test question included the scenario of linking the plural phrases with multiple objects. Therefore, in most cases, the workers could choose the correctly annotated options only if they had thoroughly reviewed the instructions. This approach ensured that quality annotators, with the fundamental understanding of the requirements in the instructions, worked on our task and was an important stepping stone to acquiring high-calibre annotations.

3.4.3 Prompts

It is imperative for a data collection tool to have features to detect any mistakes made by the workers during the annotation process and alert them with practical steps to fix the issues. For instance, the workers could forget to select and annotate some pronouns in the given sentence, which was one of the essential requirements in the process. Our annotation tool consists of several utilities to verify the current annotations before the workers submit them and notify them, if necessary, to perform further actions.

As mentioned above, one of these diagnostic utilities is responsible for detecting missing annotations of all the pronouns present in the sentences. At the beginning of each description-scene pair, this utility scans each sentence for the presence of pronouns. If the pronouns exist, our tool includes a pronoun indicator used for visually signalling to workers that some of the pronoun terms have not been annotated yet. We used the ❗ icons to suggest potential issues to the workers, and the warnings' details could be revealed by hovering over the pronoun indicator. In addition to the indicator icons, we communicated similar cautionary messages via pop-up prompts displayed when they attempted to submit their work. The prompt message included information about the pronouns missing from the annotations.

Similarly, the tool contains utilities and indicators for determiners, verbs, and prepositions. The determiner utility is responsible for checking if any selected phrases have a preceding determiner and verifying if it is included as part of the phrase selection. Similar to pronoun prompts and indicators, it alerted workers with relevant details if the determiners were missing from the phrases. Furthermore, the verb utility reviews all the selected phrases by checking if the workers included the verbs in any selections. This prevented them from se-


Pronouns not annotated!

Please select and annotate these pronouns: it, this

this is a white pillow. it is on the left side of the bed.

this is a round nightstand. it is on the floor between two beds.

You have not included determiners in your selected phrases

this is a bed with white sheets. it is next to a night stand.

Your selected phrases start with prepositions

this is a brown table. it is in front of a sofa.

Pronouns check passed!

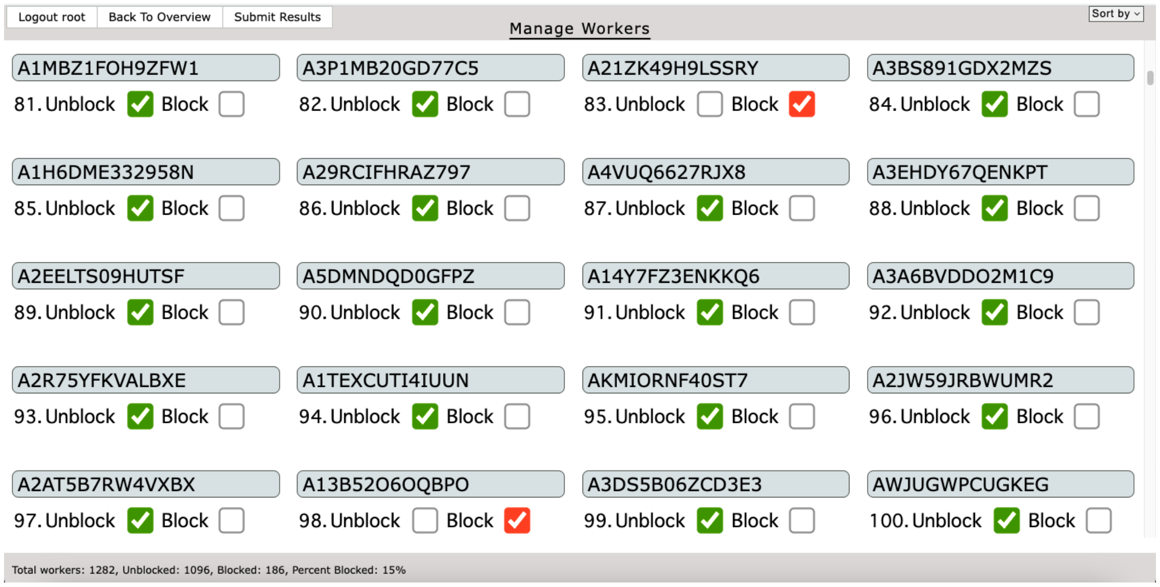
Figure 3.13: Screenshots of various prompts and indicators displayed by the utilities throughout the annotation process in our DenseRefer3D tool. For instance, the tool alerts the workers to annotate pronouns if they fail to do so. These serve as automated pre-submission validation and help workers avoid common mistakes.

lecting the entire sentences instead of the desired referring expression phrases. Finally, the tool includes the preposition utility that searches for prepositional words at the beginning of the selected phrases. Like verb utility, this ensured the workers adhered to the task instructions and followed the best practices for selecting useful phrases.

All the above-described utilities and indicator icons came into action as soon as the annotator selected a phrase and linked it with 3D objects. The timely delivery of notifications to the workers via utility prompts allowed them to correct their mistakes before submitting their work. At the same time, the indicator icons served as a pre-submission verification for workers. For instance, all ✓ icons signified that the annotations do not contain any issues concerning the above-listed categories, and the worker could proceed with the submission. These steps helped effectively execute quality control measures that improved annotation quality.

3.4.4 User Management

The success of a data collection effort requires the ability to monitor the quality of the collected annotations continuously. A vital part of the monitoring process is to provide user-access capabilities that allow the administrators to enable or disable authorization to use the tool effortlessly. Our tool includes a user management interface for administrators to impose soft blocks on workers instantly. This was useful to prevent workers with below-par annotation quality from working on our task. Besides, it allowed us to impose temporary timeouts to prevent worker burnout which may lead to a decline in the data quality.



A1MBZ1FOH9ZFW1	A3P1MB20GD77C5	A21ZK49H9LSSRY	A3BS891GDX2MZS
81. Unblock ✓ Block ☐	82. Unblock ✓ Block ☐	83. Unblock ☐ Block ✓	84. Unblock ✓ Block ☐
A1H6DME332958N	A29RCIFHRAZ797	A4VUQ6627RJX8	A3EHDY67QENKPT
85. Unblock ✓ Block ☐	86. Unblock ✓ Block ☐	87. Unblock ✓ Block ☐	88. Unblock ✓ Block ☐
A2EELTS09HUTSF	A5DMNDQD0GFPZ	A14Y7FZ3ENKKQ6	A3A6BVDDO2M1C9
89. Unblock ✓ Block ☐	90. Unblock ✓ Block ☐	91. Unblock ✓ Block ☐	92. Unblock ✓ Block ☐
A2R75YFKVALBXE	A1TEXCUTI4IUUN	AKMIORNF40ST7	A2JW59JRBWUMR2
93. Unblock ✓ Block ☐	94. Unblock ✓ Block ☐	95. Unblock ✓ Block ☐	96. Unblock ✓ Block ☐
A2AT5B7RW4VXBX	A13B52O6OQBPO	A3DS5B06ZCD3E3	AWJUGWPCUGKEG
97. Unblock ✓ Block ☐	98. Unblock ☐ Block ✓	99. Unblock ✓ Block ☐	100. Unblock ✓ Block ☐

Total workers: 1282, Unblocked: 1096, Blocked: 186, Percent Blocked: 15%

Figure 3.14: User management interface in DenseRefer3D tool. It lists all the users in a simple layout with their worker IDs and easy controls for enforcing blocking mechanisms for respective workers. This enables us to quickly implement soft blocks without impacting workers' reputations on the MTurk platform.

The MTurk platform has a built-in blocking mechanism. However, we do not endorse this feature since it can impact workers' ability to continue working on other future tasks and may even lead to their accounts receiving suspensions. Our user management interface comprises an accessible layout where all the existing task workers are displayed along with the block statuses. The interface contains straightforward *block/unblock* switches for every worker listed in the interface. The tool immediately withdraws the worker authorization when the administrator confirms the change in block status by selecting the block option for the worker. Thus, it prevents them from gaining access to the task until the administrator unblocks them through the interface.

Similar to several other administrator-level interfaces, it also includes the sorting functionality to quickly filter the workers based on the available sort criteria. This interface functioned as an additional layer of supervising the data collection process and assisted in maintaining the quality of the annotations.

3.4.5 Two-stage Review

This section explains our two-stage review strategy for verifying the collected annotations. This breakdown into two phases helps us quickly review annotations and perform an exhaustive inspection when needed.

Sanity Check

The first stage in our annotation evaluation process is named *Sanity-check*. In this mode, only the textual portions of the annotations are displayed without rendering the 3D scene. The interface shows all the annotations for a given scene in a single layout. The annotated phrases are colour-coded and follow the same colour-matching feature as available in the annotation interface. In other words, all the phrases aligned with the same 3D object are depicted using the same colours. Similarly, the interface shows the linked objects' labels (and object IDs) right under the annotated phrases. Moreover, it displays additional helpful information during the review process, including the worker ID associated with each annotation and the primary object's label and ID from the original ScanRefer [11] dataset. Furthermore, to help assess annotations for longer descriptions, the interface contains a drop-down *All Phrases* button for each annotation, which lists all selected phrases when hovered over.

In order to evaluate an annotation, the tool supplies a set of *Pass* and *Fail* options for marking the annotation as good or bad, respectively. The annotations can be sorted based on criteria such as *checked/unchecked* or *pass/fail* to filter out the annotations based on the requirement. The progress of the *Sanity-check* review process can be tracked using the values provided in the footer section of the interface, such as the percentage of annotations yet to be reviewed and the average pass rate amongst the reviewed annotations in the scene. The *Sanity-check* mode can be accessed from the overview page by clicking on the scene image. This layout of annotated descriptions makes the review process efficient by allowing the administrators to rapidly scan through the text-only annotations to identify incorrectly selected phrases.

Second-Stage

The next step in the assessment process is called *Second-stage* verification. The interface in this mode is analogous to the annotation interface in that it renders both the description and 3D scene in the same view. Furthermore, it includes the *All objects in scene* menu and *Main object* button to help quickly locate the referred objects in connection with the central object. It employs a similar strategy discussed in the Sanity Check section for visualizing the selected phrases. It assigns a distinctive colour to each phrase and outlines two or more phrases with the same colour if they are associated with the same object. Besides highlighting selected phrases, it enables visualization of 3D objects linked with them. The reviewer may spotlight an object individually by hovering over the object box or prefer to visualize all the annotated objects together using the *Show annotated objects* button.

For verifying the annotations in *Second-stage*, the interface contains simple choices that can be specified for an annotation depending on its quality. Each annotation can be appraised with one of the three available options: *Good*, *Satisfactory*, or *Bad*. If the reviewer chooses either *Satisfactory* or *Bad*, the interface warrants a reason for the selected option.

As mentioned before, this constraint helps us understand the issues in the annotation to make necessary modifications if needed. Additionally, the interface reveals the results of the *Sanity-check* verification stage, which can also be updated directly from *Second-stage* interface. Consequently, the reviewer can filter out the annotations marked as *Pass* in the first stage and focus only on exhaustively reviewing the problematic ones.

The interface allows for seamless navigation between the annotations by either changing one annotation at a time using the *Next* and *Previous* buttons or rapidly moving through the annotations by holding down the navigation buttons to leap to the preferred annotation. There are multiple ways to access the *Second-stage* verification interface. One method of doing this is from the *Sanity-check* interface by clicking on the scene ID at the top or directly from the overview page by first enabling the *verification* mode using the *Verification Stats* button. In *verification* mode, the overview interface displays the progress of *Second-stage* verification for all the scenes, and clicking on a scene thumbnail directs the reviewer to the Second-stage interface. This evaluation mode is useful for comprehensively assessing the quality of the selected phrases and the assigned 3D objects in the collected annotations.

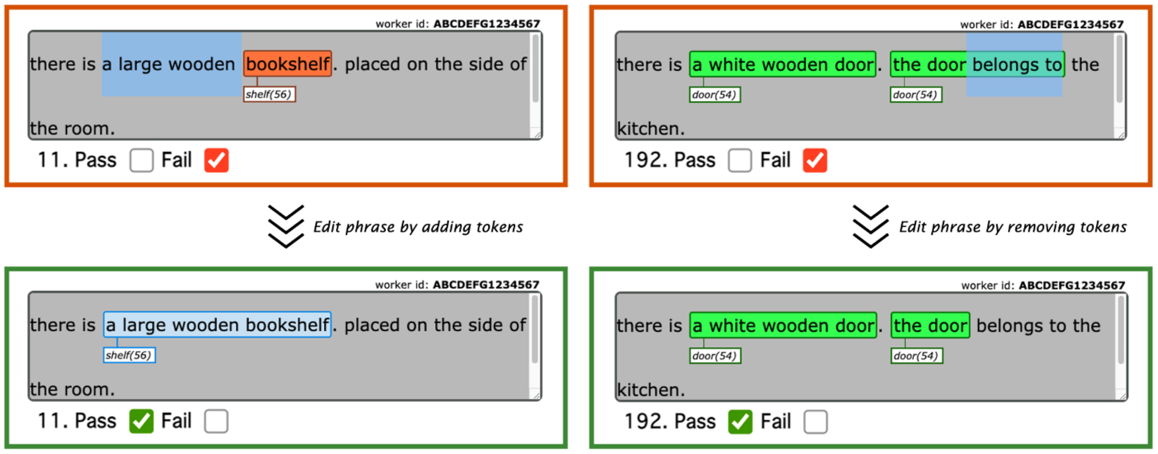
3.4.6 Annotation Correction

The two-step verification strategy helps seamlessly evaluate the collected annotations using the *Sanity-check* stage while also performing a rigorous assessment of the phrases-to-objects mappings in the second stage. However, the evaluation is beneficial if it is complemented by the functionalities to rectify the issues found during the review phases. To that end, our tool consists of features that allow administrators to fix the errors in the annotations, eliminating the need to modify the database documents manually.

Essentially, we anticipate two kinds of significant problems in the annotations. The first kind deals with phrase selection issues. The annotated phrases might not contain all the valuable information in the sentence. For instance, a selected phrase could be missing determiners, countable/uncountable terms, shape, colour, size, or the location of the object defined by the phrase. Our tool provides ways to amend the selected phrases in the existing annotations. For example, suppose the annotator selected the phrase *desk* even though the sentence included a more useful expression *a large desk in the corner*. In that case, one can easily drag the phrase box on either side using a mouse to add (or remove) the required words in the existing phrase.

The other problem we might encounter with the annotations concerns 3D objects linked with the selected phrases. The annotator might end up choosing an incorrect object for a phrase. Our tool is equipped with an elementary method for updating the objects associated with the phrases. To do this, one merely needs to select the intended object box, locate a new object, and double-click on it to confirm the corrected phrase-to-object mapping.

After submitting the modifications to the annotated phrases or linked 3D objects, the amended annotations are stored as new documents in our database to prevent overwriting


there is a large wooden bookshelf. placed on the side of the room.

there is a white wooden door. the door belongs to the kitchen.

there is a large wooden bookshelf. placed on the side of the room.

there is a white wooden door. the door belongs to the kitchen.

worker id: ABCDEFG1234567
shelf(56)
11. Pass Fail
worker id: ABCDEFG1234567
door(54)
door(54)
192. Pass Fail
Edit phrase by adding tokens
Edit phrase by removing tokens
worker id: ABCDEFG1234567
shelf(56)
11. Pass Fail
worker id: ABCDEFG1234567
door(54)
door(54)
192. Pass Fail

Figure 3.15: Interface for updating annotations in the DenseRefer3D tool. The interface provides an intuitive method of modifying selected phrases using the drag-and-drop approach to add or remove tokens from the phrases.

original annotations. These intuitive correction approaches allowed us to effortlessly update the existing annotations without requiring manual changes to the data stored in the database.

3.4.7 Others

HIT Qualifications

The most useful HIT qualifications to obtain optimal quality annotations include *Number of HITs Approved*, *HIT Approval Rate*, and *Locale* parameters. We performed numerous experiments to determine the best values for these parameters in a HIT. We tested different input combinations and compared the results' quality and the total time to collect the annotations for a small sample. We discovered that setting large parameter values resulted in the highest-quality annotations. However, doing so decreased the pool of available workers meeting the criteria and took much longer to complete the task. Thus, finding the optimal parameter values was vital to reduce the turnaround time for our task while maintaining the favourable quality of collected data. We also utilized the locale parameter to restrict our task to English-speaking regions. We made this preference to minimize the learning curve as the ScanRefer [11] dataset consists of sentences written using the natural English language.

Highlighting Objects

It was essential to assist workers in locating the intended objects to gather accurate phrase-to-object mappings. Our tool highlights the object under the mouse cursor and surrounds it with a bounding box to direct the primary focus on it. This enabled seamless navigation between objects and helped identify the target objects described in the phrases. Additionally, the tool displays the label of the highlighted object next to it, enabling workers to verify that the object referred to by the phrase resembles the highlighted 3D object. Moreover, a single mouse click on the highlighted object hides (or reveals, if not already displayed) the object's label.

Updating Phrases and Linked Objects

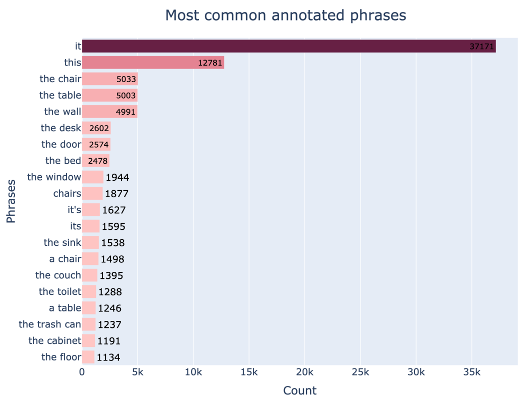
The workers must be able to make quick fixes to the selected phrases or the objects linked with them. Our tool provides the functionality to remove an inaccurate phrase individually without using the *Reset* feature that clears all the annotations at once. Each selected phrase has its own ⊕ and ⊖ buttons to add or remove object boxes under it, and the latter one could be used to delete the selected phrase and the objects associated with it. Furthermore, the tool enables updating the linked objects under the phrase by selecting the problematic object box and double-clicking on the replacement object in the 3D scene. The tool prevents workers from linking an already-associated object with the phrase when performing this operation. These quick ways of updating the annotations allowed the workers to continue supplying quality work with minor interruptions. Finally, the tool contains two extra features for improving the annotations: *Go Back* and *Start Over*. The *Go Back* button brings the workers to the previous assigned scene-description pair, one at a time, so that they can address any issues with the previous annotations. In comparison, the *Start Over* button restarts the task from the first assigned scene-description pair, which was valuable for redoing the entire assignment due to too many errors.

Multiple Annotations

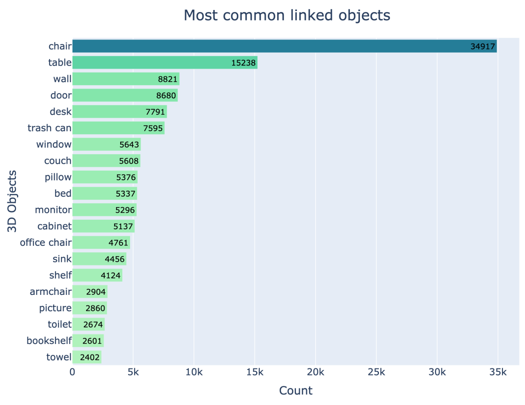
We collected multiple annotations for some assignments to cross-reference them for identifying high-quality workers. We assigned the same scene-description pairs to multiple workers and verified the quality of the provided annotations. To determine the quality of their work, we employed university student volunteers to annotate the same scene-description pairs and thoroughly verified the accuracy of the annotations. We then compared the submissions by the MTurk workers with the verified student annotations and identified the workers that provided the most similar annotations of textual phrases and linked objects to the verified annotations. Based on these verifications, we filtered the annotations by only keeping one annotation per scene-description pair and discarding the rest. Thus, in addition to the mandatory qualification test and HIT parameters experiments, this step provided us with high-quality annotators with a sound understanding of our task.

3.5 Data Statistics

In this section, we summarize the results of our data collection effort and provide an exhaustive comparison with similar existing datasets.



(a) Most commonly annotated phrases in our DenseRefer3D dataset.



(b) Most commonly linked objects in our DenseRefer3D dataset.

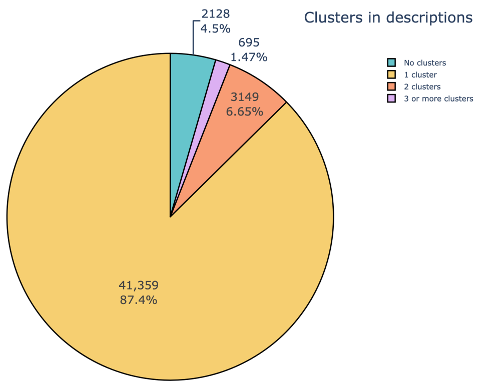
Figure 3.16: Plots of phrases and 3D objects with the most number of annotations. The pronoun terms, such as *it* and *this*, are the most frequently occurring phrases aligned with objects. The objects such as *chair* and *table* are the most linked objects in our dataset.

As previously mentioned, to curate our DenseRefer3D dataset[4], we utilized natural language descriptions from ScanRefer dataset [11] depicting the objects in 3D scenes from ScanNet dataset [20]. The objective of our data collection process was to obtain the mappings of referring expressions in the sentences to the 3D objects described by these expressions in the scenes. We collected the desired annotations for 46,173 ScanRefer [11] sentences and 706 3D RGB-D ScanNet [20] scenes.

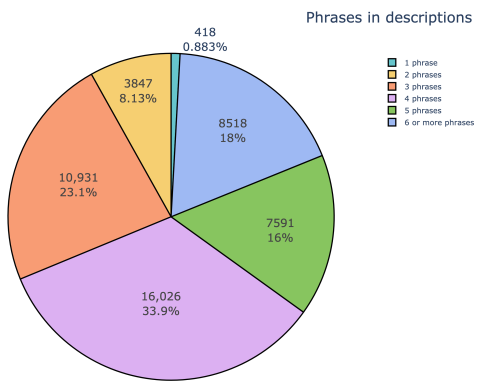
Overall, we obtained 124,270 coreference clusters with annotations of 207,177 referring expressions representing different 3D objects. Moreover, our dataset contains 125,604 pairwise coreference links between the mentions. In comparison, CoNLL-2012 corpus [71] consists of 44,221 coreference clusters containing 194,480 mentions referring to entities. In terms of the total number of available tokens, ScanRefer [11] has more than 1 million tokens in all the descriptions, whereas CoNLL-2012 [71] and PreCo [12] datasets contain about 1.6 and 12.4 million tokens, respectively. The DenseRefer3D dataset is significantly larger in the number of annotated coreference links than the WikiCoref [32] and GAP [91] datasets, with the latter consisting of just around 9000 annotated mention-pronoun pairs.

The existing datasets mentioned above are text-only, implying they do not contain the annotations of the entities referred to by the text phrases. The closest effort to ours, Sentences3D dataset [46], consists of the alignments between the tokens and the objects from NYU-RGBD v2 dataset [66]. However, the Sentences3D dataset is tiny, comprising just 61,195 total tokens from 1449 multi-sentence descriptions and an equal number of NYUv2

[4]`https://github.com/3dlg-hcvc/dense-scanrefer-baselines/blob/main/data/DenseRefer3D.zip`



(a) A pie chart representing the distribution of annotated clusters amongst ScanRefer [11] descriptions in our dataset.



(b) A pie chart representing the distribution of annotated phrases amongst ScanRefer [11] descriptions in our dataset.

Figure 3.17: Representation of annotated phrases and coreference clusters in the DenseRefer3D dataset. Our dataset contains at least one annotated cluster in 95% of ScanRefer [11] descriptions. Furthermore, around 90% of the descriptions contain three or more annotated phrases that describe 3D objects.

images, with an average of 40 tokens per description. Furthermore, the text-based annotations primarily consist of noun terms mapped to the objects. They do not necessarily contain additional useful terms such as determiners, countable and uncountable types, and relational words. In contrast, our dataset incorporates rich details about phrases that, besides including the headwords, also enclose tokens for describing the attributes of the main object and its relationship with surrounding objects.

Another similar dataset, VisPro [99], is relatively larger, with about 30,000 tokens annotated with objects. However, their effort mainly revolved around curating annotations for the pronoun terms. As a comparison, in addition to annotations of other types of phrases, our dataset consists of more than 55,000 pronoun tokens linked with 3D objects. Finally, SIMMC 2.0 dataset [48] incorporates roughly 117,000 sentences from nearly 1566 in-store shopping scenes. Moreover, it contains about 20 objects per scene, similar to the ScanNet dataset consisting of several objects per scene. However, the biggest advantage of our DenseRefer3D dataset over the SIMMC 2.0 [48] and other datasets mentioned above is the presence of richer, more descriptive annotations of phrases explicitly associated with real-world 3D objects. Besides, our dataset includes the annotations of the anaphoric and cataphoric terms with 3D objects.

Additionally, the coreference chains in our dataset have, on average, nearly two mentions per cluster and around 40% of the annotated clusters contain two or more mentions. Our dataset also includes annotations of phrases that do not belong to any cluster, with 74,003

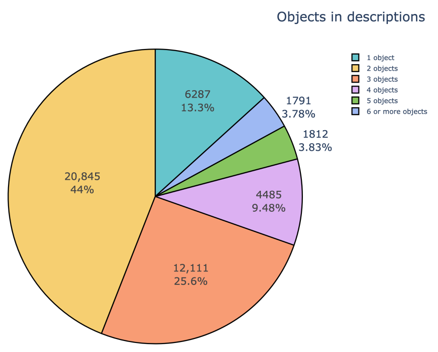


Figure 3.18: A pie chart displaying the number of linked objects corresponding to the phrases in ScanRefer [11] descriptions. Similar to the previous chart, our dataset contains rich alignments between phrases and linked objects in the descriptions, with more than 86% of them comprising at least two distinct objects linked with phrases.

Number of referring expressions	207,177
Number of coreference clusters	124,270
Number of pair-wise coreference links	125,604
Number of singletons	74,003
Number of pronouns	55,518
Number of plural expressions	22,120
Average tokens in an expression	3.88
Average clusters per scene	176.77
Average singletons per scene	105.26
Average pronouns per scene	78.97

Table 3.1: DenseRefer3D dataset statistics. Our dataset consists of around 125,000 coreference clusters containing 207,177 mentions. Besides, it includes annotations of 74,003 singleton expressions and more than 22,000 plural phrases referring to multiple 3D objects. The referring expressions are made up of around four tokens to represent the objects.

clusters classified as singletons. The average length of the mentions in our dataset is approximately 3.9 tokens, with each description containing more than four annotated mentions. Each mention in the description is comprised of about 2.3 tokens. Moreover, in our dataset, around 95.5% of all the ScanRefer descriptions have at least one coreference cluster referring to distinct objects, and there are, on average, 2.62 clusters per description. Our dataset also incorporates annotations of phrases made up of plural terms portraying multiple objects. Additionally, our dataset has more than 55,000 pronoun-to-object alignments, easily surpassing any similar existing datasets. The pronoun terms (*it* and *this* being the most annotated ones) are also the top two most common types of phrases, followed by phrases such as *the chair* and *the table*. Corresponding to these phrases, the top three most common linked objects are *chair*, *table*, and *wall*. On the other hand, the least common phrases usually describe one-off obscure object(s), such as *the upper, wall mounted cabinets* and *small black round trash can in a room*, and the least common linked objects are *stapler*, *oven mitt*, and *fire alarm*. This is related to the fact that the ScanRefer dataset has about five descriptions per object in the scene. However, most descriptions usually define a subset of the objects belonging to certain semantic ScanNet classes, with phrases that describe the objects outside these classes occurring less frequently.

Finally, our dataset consists of around 177 annotated coreference clusters and almost 179 links between the mentions in the clusters per ScanNet scene. Further, annotations of approximately 79 pronouns and 105 singletons are available for a 3D scene. Thus, it provides many dense alignments between textual referring expressions and real-world 3D objects.

Chapter 4

3D Coreference Resolution

4.1 Coreference Resolution

In the natural language processing domain, the underlying objective is to comprehend the contents of a given sentence, including figuring out the involved context. One of the essential sub-tasks in the discipline deals with determining the meaning of co-referring phrases, known as coreference resolution. It is the task of identifying all the words or referring expressions that depict the same entity in a given description. The entity represented by the referring expressions is known as the referent. Two or more referring expressions are said to co-refer if they describe the same referent object. Thus, the connotation of one or more co-referring phrases can be interpreted using another phrase in the same group. These groups of co-referring mentions are known as clusters or coreference chains. Furthermore, a sentence may comprise referring expressions depicting a referent object unaccompanied by a partnering co-referring phrase. These types of phrases are known as singleton mentions. Moreover, the referent does not need to be a real-world entity; the mentions may very well refer to an imaginary object. If executed correctly, it should be possible to interchange these referring expressions without affecting the meaning of the description. Thus, this task is instrumental in dealing with ambiguity arising from diverse phrases, such as pronouns. The co-referring mentions are classified into anaphoric and cataphoric terms.

Anaphor and Cataphor

An anaphor is one of the co-referring mentions that occurs after the mention referred by it. For example, in the description, *The chair is next to a desk. It is black*, the phrases *The chair* and *It* refer to the same entity (i.e., a *chair*). In this case, *It* is the anaphoric term since it appears after the entity-describing mention *the chair*, called the antecedent. The meaning of the anaphoric mentions is inferred using the antecedent. On the other hand, a cataphor is a mention that emerges before the mention referred by it. For instance, in the description *It is a large bed next to the desk*, *It* is known as cataphor whereas *a large bed* is called postcedent. It helps explain the meaning of the cataphoric mention.

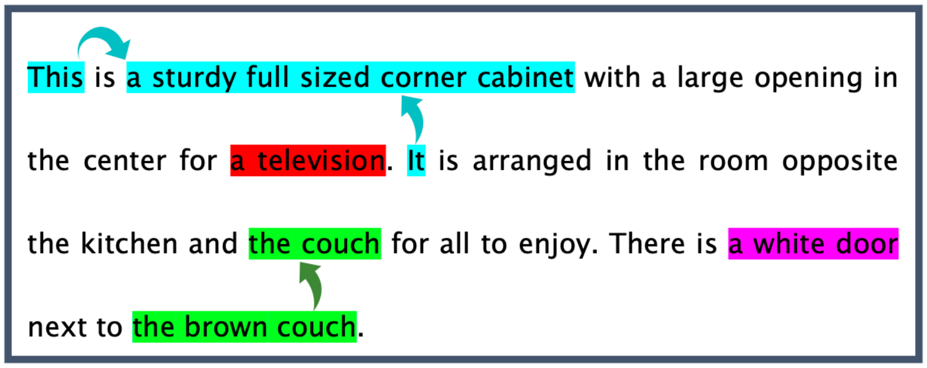

This is a sturdy full sized corner cabinet with a large opening in the center for a television. It is arranged in the room opposite the kitchen and the couch for all to enjoy. There is a white door next to the brown couch.

Figure 4.1: An example description showcasing coreference resolution task. The task requires identifying all the spans of tokens characterizing entities and grouping them into clusters based on the referent objects. In this example, *This*, *It*, and *a sturdy full sized corner cabinet* belong to the same cluster since all these phrases refer to the same object *cabinet*.

4.2 Approaches for Resolving Coreferences

The coreference resolution problem comprises two sub-tasks: mention detection and mention clustering. Mention detection determines the spans of texts (known as mentions or referring expressions) representing the referent objects. This is crucial since not all the spans in a given sentence qualify as the candidate mentions. Consider the example description *A brown desk is next to a black chair with wheels. It is the only desk in the entire room.* All the tokens in this description may not be considered potential mentions as they might not describe any entity. Here, the span *next to the* does not directly depict an object, whereas *A brown desk* is a reasonable referring expression candidate for describing the object *desk*.

It is also essential to differentiate between two or more good candidate mentions to select the best one. For instance, the above example description contains multiple mention candidates referring to the object *chair*, such as *chair*, *the black chair*, and *the black chair with wheels*. Thus, it is important to figure out the most suitable span for referring expression candidates. In our example, the phrase *the black chair with wheels* is conceivably the best candidate for mention representing the referent *chair* since it includes additional information about the described object, such as its colour and the parts associated with it.

Mention Clustering involves grouping the candidate mentions based on the referred objects. All the mentions in a group (i.e., a cluster) portray the same real-world entities. The example description, *A black chair is in the corner of the room. It is next to a gray desk with a lamp on it*, consists of *A black chair*, *It*, *a gray desk*, *a lamp*, and *it* as candidate mentions. The next step following identifying mention candidates entails forming chains of mentions based on the characterized entity. The above example contains two clusters of co-referring mentions: [*A black chair, it*] and [*a gray desk, it*]. The pronoun term *it* is present in both clusters, but both instances of *it* are not related to each other as they are members of two different clusters. In the first cluster, the *it* refers to the object *chair*, whereas the second instance represents the object *desk*. Thus, mention clustering and mention identification help resolve pronoun terms by providing meaning using the other co-referring mentions. In addition, the singleton mentions that do not have a co-referring mention pair are not grouped into clusters.

Most modern approaches in addressing the problem of resolving coreferences perform the mention detection and mention clustering tasks. These techniques are categorized mainly into three architectures: mention-pair, mention-rank, and entity-based architectures.

Mention-Pair

The mention-pair modelling is possibly the simplest of all the approaches. It is concerned with learning a classifier to determine the coreference relation between a pair of mentions. In essence, the model finds an antecedent for a given mention by calculating the coreference probabilities between the mention and its candidate antecedents. A lengthy description with many tokens can result in a massive pool of candidate antecedents for a mention. This, supported by the fact that the coreference relations do not exist for many pairs of mention and their potential antecedents, may cause a high data imbalance skewed towards a large number of non-co-referring pairs. Several techniques have been proposed to address this problem, with the most accepted one described in Soon et al. [85]. The antecedents closest to the mention are considered positive samples, while the remaining pairs of mention and potential antecedents are marked as negative samples.

The next step in this approach is clustering, which includes organizing the mentions into groups depending on the pairwise probabilities computed in the previous step. This simple approach, however, deals with two primary issues: lack of direct comparisons between potential antecedents to identify the most suitable one and prioritization of local pairwise decisions instead of considering entities along with the mentions.

Mention-Rank

The mention-rank approach aims to address one of the shortcomings of the mention-pair models. All the potential antecedents are ranked against each other by assigning a score to each comparison. The antecedent with the highest score is selected as the best antecedent for the mention. Thus, for a mention, all the proceeding spans of text are considered candidate antecedents. Besides, a mention may be assigned an ϵ, a dummy value for when an antecedent does not exist for a given mention. The model then computes the probability of an antecedent for a given mention for all candidate antecedents, including the ϵ, and the antecedent with the highest probability value is chosen as the target antecedent. After that, we can resolve the final clustering of mentions using transitive closure on all the identified mention-antecedent pairs. Since there may exist multiple ground-truth antecedents for a given mention, training such models can be challenging to figure out the best gold antecedent to employ for training the model.

Entity-Based

The previous two methods involve the model forming determinations about the mentions. In contrast, the entity-based approach operates by assigning a mention to a chain of mentions instead of allocating it to another mention. Thus, in the simplest form, a mention-ranking approach can be converted into an entity-based one by learning a classification model over mention chains in place of discrete mentions. Neural network-based models presented in Wiseman et al. [94], Clark and Manning [16], can learn the representations for chains of mentions on their own by encoding the state corresponding to the mention chains with the help of a recurrent neural network (RNN) applied on the sequence of chains as described in Wiseman et al. [94]. This approach, however, has not resulted in substantial performance improvements despite the availability of auxiliary information on chains of mentions.

4.3 End-to-End Coreference Resolution

Lee et al. [51] presented an end-to-end approach to address the problem of resolving coreferences. This is based on the mention-ranking architecture discussed in Mention Ranking section. In this, the model does not require information from external preprocessing sources such as output from syntactic parsers or hand-crafted rules to detect mentions and only uses the ground-truth clusters of mentions during the training process. It processes all the spans of text in the input document as potential mentions, and for each span, all the spans occurring before it are considered candidate antecedents. The model then assigns an antecedent from the candidate antecedent set to each span (i.e., potential mention). In addition, rather than assigning an antecedent, it may link a special antecedent, ϵ, which could indicate either that the text span in question may not be a suitable mention or the span does not have a coreference relation with any of the previous spans.

So, for an input document consisting of T tokens, the total number of spans of text considered as candidate mentions by the model would be $\frac{T(T-1)}{2}$. For a span i, the set of candidate antecedents is represented as $Y(i)$, where $Y(i) = \{span\ 1, span\ 2, \ldots, span\ i-2, span\ i-1, \epsilon\}$. The model is then trained to learn a probability distribution of these candidate antecedents for each span, denoted as follows as per Lee et al. [51]:

$$P(y_i) = \prod_{i=1}^{N} \frac{\exp\left(s(i,\, y_i)\right)}{\sum_{y' \in Y(i)} \exp\left(s(i, y')\right)}$$

The pairwise score $s(i, j)$ is composed of three factors: 1. $s_m(i)$ (unary score for i), implying if the span i is a suitable mention, 2. $s_m(j)$ (unary score for j), indicating if the span j is a suitable mention, and 3. suggesting if span j is a suitable antecedent for span i. In case when span i is assigned a dummy antecedent ϵ, for the reasons described above, the coreference score between span i and span j is marked as 0. This enables the model to assign the span with the highest score as the target antecedent when ϵ is not assigned to j while refraining from making any antecedent prediction for span i when ϵ is assigned to j. Thus, $s(i,j)$ is represented as follows as per Lee et al. [51]:

$$s(i,j) = \begin{cases} s_m(i) + s_m(j) + s_a(i,j) & j \neq \epsilon \\ 0 & j = \epsilon \end{cases}$$

The number of all spans expands with the increase in the length of the document. Thus, it is vital to introduce methods to control the number of spans of text processed by the model. Lee et al. [51] addresses this by only keeping the spans consisting of up to L tokens. To further minimize the set of candidate spans to consider, it employs vigorous pruning techniques, such as removing the spans with low unary mention scores s_m and limiting the number of candidate antecedents up to K for each span.

Representing Spans

In order to calculate the unary mention score for span i and pairwise coreference scores between span i and span j, the spans i, j need to be encoded in a manner that accurately captures the information available in them. This work utilizes the internal organization within a span, the contextual information encompassing the candidate mention span, and the headword in each span.

The tokens in the input document are vectorized by concatenating the pre-trained 300-dimensional GloVe embeddings [69] and 50-dimensional embeddings based on Turian et al. [89]. Furthermore, it obtains the 8-dimensional character embeddings learned using a 1D-CNN to represent the characters in the tokens. Thus, we produce the vectorized representation of a token i as h_i. The vectorized span representation is generated using a bi-directional LSTM [39] to capture the information within the span and the context adjoining it. Additionally, as discussed above, the headword is learned using an attention-based mechanism described in Bahdanau et al. [6]. The final encoded span representation for span i is acquired by combining the output of bi-directional LSTM and the attention layer and is depicted in the following equation from Jurafsky and Martin [43]:

$$g_i = [h_{START(i)}, h_{END(i)}, h_{ATT(i)}]$$

Computing Unary and Pairwise Mention Scores

The unary and pairwise mention scores are computed using feedforward neural networks as follows according to Lee et al. [51]:

$$s_{m(i)} = w_m \cdot FFNN_m(g_i)$$
$$s_a(i, j) = w_a \cdot FFNN_a([g_i, g_j, g_i \circ g_j])$$

where, $g_i \circ g_j$ performs an element-wise operation to compute the similarity between spans i and j.
Once the joint probability distribution of candidate antecedents is learned for each mention span, the final clusters of co-referring mentions can be assembled using the transitive closure on assigned antecedents.

Model Training
The labelled coreference datasets consist of ground-truth clusters of mentions for each document. However, each mention is not annotated with a distinct ground-truth antecedent, as all the co-referring mentions are associated with the same cluster. Thus, there only exists an immanent antecedent for each mention. Therefore, it uses the marginal log-likelihood function to maximize the probability of a span co-referring with a mention span for any true antecedents. As explained above, for a span i, $Y(i)$ denotes the set of all the potential antecedents for i, i.e., all the spans that appear before the span i. If we consider $GOLD(i)$ for depicting the set of ground-truth clusters of mentions of which i too is a member, then all the ground-truth mentions occurring before the span i, $\widehat{Y}$, can be obtained by $Y(i) \cap GOLD(i)$. Hence, it is reasonable to optimize the following:

$$\log \prod_{i=1}^{N} \sum_{\hat{y} \in \widehat{Y}} P(\hat{y})$$

where, $GOLD(i) = \epsilon$ if span i is not associated with any ground-truth coreference cluster.

4.4 Evaluation Metrics

In this section, we provide a brief overview of the most prominent metrics used to evaluate the coreference resolution task.

MUC
Vilain et al. [90] proposed MUC metric for assessing the performance of coreference resolution models. It is a link-based metric, where a link is the existence of a coreference relation between two mentions. It performs the evaluation depending on the links in both predicted and ground-truth coreference clusters. It calculates the precision and recall values using the

identified links as follows:

$$Precision = \frac{\textit{Total links common to both predicted and ground-truth clusters}}{\textit{Total links in predicted clusters}}$$

$$Recall = \frac{\textit{Total links common to both predicted and ground-truth clusters}}{\textit{Total links in ground-truth clusters}}$$

As indicated above, MUC does not consider singletons because such mentions do not form any coreference links with other mentions. Moreover, it favours the models that predict larger coreference clusters with fewer mentions.

B^3

Bagga and Baldwin [5] presented a mention-based metric, B^3, for the coreference resolution task. Rather than using the links in the clusters, it calculates the precision and recall for each mention in the ground-truth coreference clusters. In essence, it determines all the mentions correctly predicted by the model, obtained using the intersection of the predicted and ground-truth clusters. The precision and recall for all mentions are computed as follows:

$$Precision = \sum_{i=1}^{N} w_i \frac{\textit{Predicted clusters containing mention } i \cap \textit{GT clusters containing mention } i}{\textit{Total number of mentions in the predicted clusters containing mention } i}$$

$$Recall = \sum_{i=1}^{N} w_i \frac{\textit{Predicted clusters containing mention } i \cap \textit{GT clusters containing mention } i}{\textit{Total number of mentions in the ground-truth clusters containing mention } i}$$

where, w_i is the weight for each entity.

However, in B^3, an entity may be utilized more than once to evaluate precision and recall values.

CEAF$_{\phi_4}$

CEAF$_{\phi_4}$, proposed by Luo [57], is an entity-based coreference evaluation metric. Contrary to the previous two metrics that rely on coreference links or mentions in the clusters, it directly compares the entities with each other, i.e., the predicted cluster of mentions with the gold clusters, to find their similarity. The function used to calculate the similarity between the entities is as follows:

$$\phi_4(P_i, G_j) = \frac{2 \times |P_i \cap G_j|}{|P_i| + |G_j|}$$

Then, the precision and recall can be computed using:

$$Precision = \max_{m} \frac{\sum_{i \in N_p} \phi_4(P_i, m(P_i))}{\textit{Total number of predicted clusters}}$$

$$Recall = \max_{m} \frac{\sum_{i \in N_p} \phi_4(P_i, m(P_i))}{\textit{Total number of ground-truth clusters}}$$

where m is a function which, when given an input predicted cluster P_i, returns a mapped ground-truth cluster, but it enforces that no two ground-truth clusters are mapped to the same predicted cluster.

We can then obtain the $F1$ score using the precision and recall values for each of the above-discussed evaluation metrics. The final $F1$, CoNLL score is computed by averaging the $F1$ scores of MUC, B^3, and CEAF$_{\phi_4}$ metrics.

$$CoNLL\ F1 = \frac{MUC + B^3 + CEAF_{\phi_4}}{3}$$

4.5 Coreference Resolution using 3D

As detailed in Chapter 2, most work on coreference resolution and referring expression comprehension tasks have exploited only textual information or a combination of text and 2D visual inputs. Since these tasks have real-world applications involving day-to-day 3D visual settings, exploring the employment of extra information from 3D modality is imperative. In this section, we provide an overview of our DenseRefer3D architecture, in which we combine the sentences with 3D scenes to learn the mapping from coreference clusters to referring objects. Our model takes a natural language sentence and 3D RGB-D scene as inputs and generates alignments between chains of coreferences and 3D objects.

4.5.1 DenseRefer3D Model Architecture

The architecture comprises three main components: a 3D module, a language module, and a mapping module.

4.5.1.1 3D Module

This module takes a point cloud representing a 3D RGB-D scene as input and processes it through a 3D object detection model, VoteNet [73]. VoteNet is an end-to-end approach for detecting objects directly in 3D using the Hough voting technique. The 3D object detection task entails approximating oriented 3D bounding boxes and objects' semantic classes. It deploys a PointNet++ [74] backbone to perform learning of point set features.

It then generates votes from the point features. These are structurally similar to points but provide improved object localization. The votes are aggregated into clusters and fused

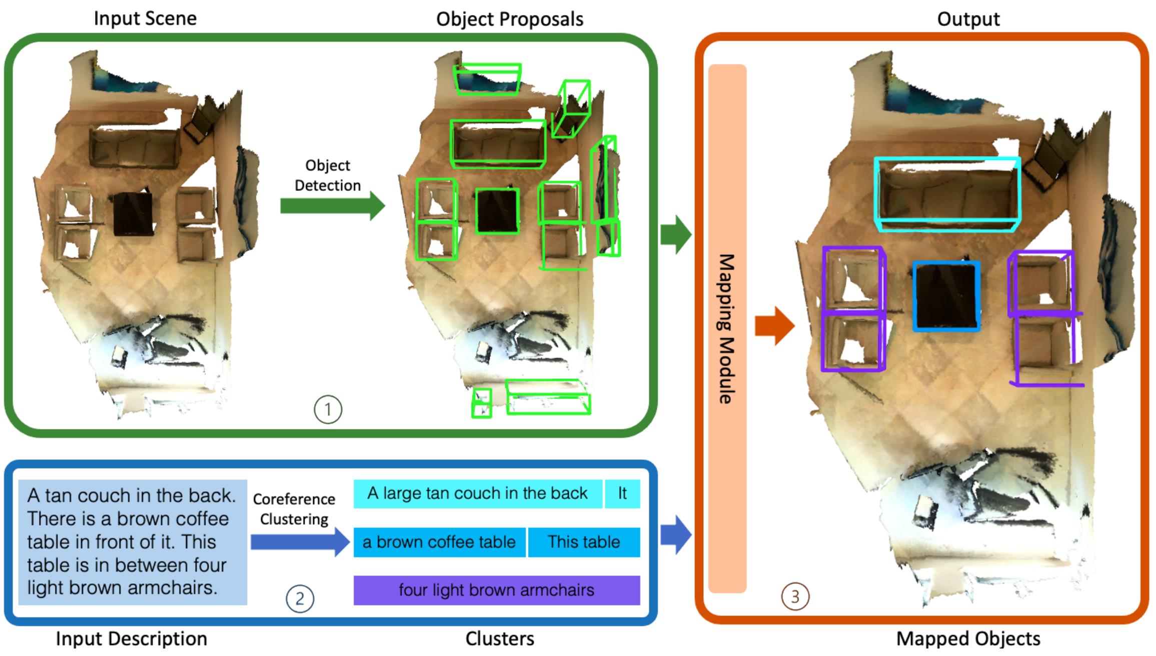

Input Scene
Object Proposals
Object Detection
1
Output
Mapping Module
3
A tan couch in the back. There is a brown coffee table in front of it. This table is in between four light brown armchairs.
Coreference Clustering
2
A large tan couch in the back
It
a brown coffee table
This table
four light brown armchairs
Input Description
Clusters
Mapped Objects

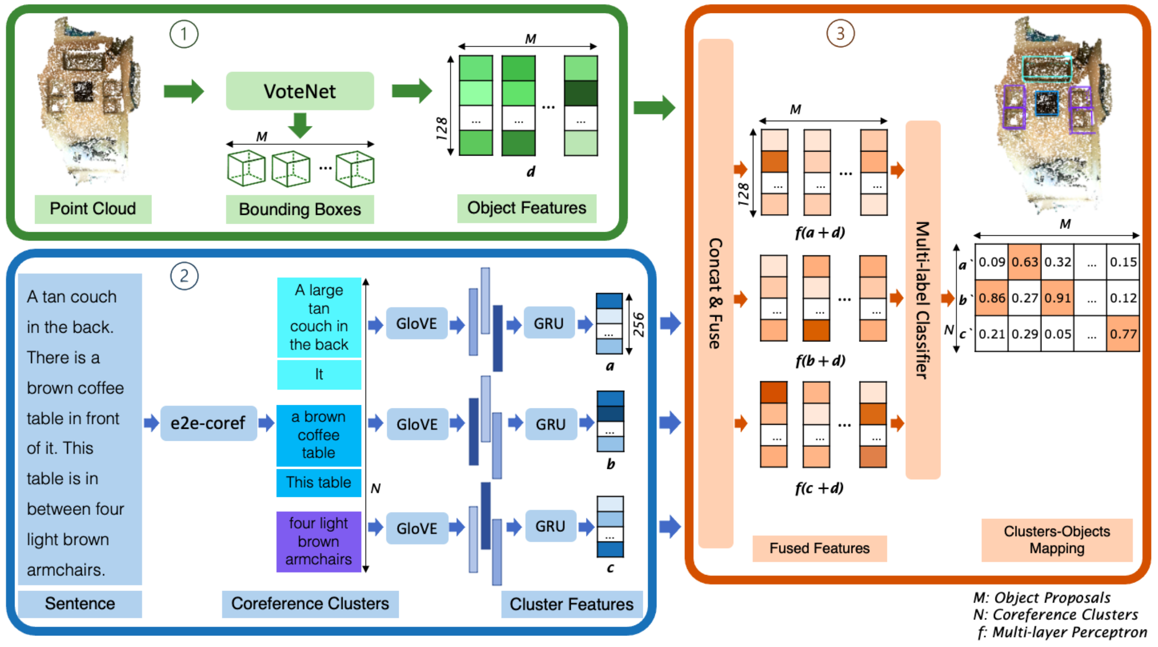
Figure 4.2: DenseRefer3D task overview. Our model processes multi-modal inputs consisting of natural language sentences and RGB-D 3D scenes and aligns the coreference clusters with 3D objects referred to by the mentions in them.

to produce the object proposals. We exploit these capabilities of VoteNet [73] to extract aggregated point features as the output of our 3D module.

Our 3D data is represented similar to Chen et al. [11]. Thus, for an input point cloud consisting of features such as the vertices' position and colour, we randomly sample $40,000$ vertices to prepare the 3D point cloud vector $P_c \times 6$. Each point is represented using three positional coordinates and three values of RGB colour components. The resulting point cloud is fed into VoteNet [73] to generate $M \times 128$, where M is the number of object proposals, each with a vector size of 128.

4.5.1.2 Language Module

The module is responsible for processing input sentences to extract textual features. Our mapping module merges text features with visual features obtained from the 3D module. For text feature extraction, the input sentence is first processed through the end-to-end coreference resolution model [51]. It takes a tokenized sentence as input, extracts all the mentions representing real-world objects and assigns them to groups based on the objects referred to by them. The groups of mentions are known as coreference clusters. Afterward, we represent each phrase in each cluster using pre-trained 300-dimensional Global Vectors for Word Representation (GloVe) [69] to generate the cluster embeddings by averaging the embedding vectors for all the tokens in a phrase. This ensures that the phrases with similar meanings are encoded with similar representations.



a	0.09	0.63	0.32	...	0.15
b	0.86	0.27	0.91	...	0.12
c	0.21	0.29	0.05	...	0.77

Figure 4.3: DenseRefer3D model architecture. The 3D module is responsible for extracting point features using VoteNet [73] while our language module deals with deriving cluster-level features from the End-to-end coreference resolution model [51]. Finally, our mapping module learns the alignment between the coreference clusters and 3D objects.

We obtain the cluster embeddings as $N_c \times N_t \times 300$, where N_c represents the coreference clusters generated using the end-to-end coreference model [51] and N_t depicts the tokenized sequence of co-referring phrases in each cluster. Each token in the co-referring phrase is encoded using $300D$ GloVe vectors. We pad these vectors to construct the same-sized inputs to make it easy for the model to process the input sequences of varying lengths.

The model then processes the cluster embeddings, which are fundamentally sequences of word embeddings for each phrase in a cluster, through a Gated Recurrent Unit (GRU) [14] to extract cluster features for each coreference cluster. GRU helps integrate cluster-level information from the sequence of phrases in a cluster. The output of the language module is the textual features extracted from all the coreference chains, i.e., $N_c \times 256$, where 256 is the final hidden state dimension of a GRU cell.

4.5.1.3 Mapping Module

The outputs of the language module, cluster features, and 3D module, object features, are passed as inputs to the mapping module. It is responsible for learning the mapping of the N coreference clusters with the M object proposals. As a cluster in the coreference chain may refer to one or more objects, we model our mapping module as a multi-label classification problem. The features of each coreference cluster are combined with object features to formulate fused language and 3D feature vectors. In other words, we fuse each 256-dimensional coreference cluster embedding vector with $M \times 128$ object proposals to output a $N \times M \times 128$ multi-dimensional vectors.

Furthermore, the unified features are processed through a mapping model, which learns to align the coreference clusters with one or more object proposals. It generates an $N \times M$ vector, which signifies alignment scores of all M object proposals for every cluster in N. To achieve this, we deploy a multi-layered neural network that consists of two hidden layers with a dimension size of 256 and an M-dimensional output layer. We use the sigmoid activation function to confine the cluster-to-object mapping scores to the probability range $[0, 1]$.

4.5.2 Loss Function

Since our network is a direct extension of ScanRefer [11], we employ a similar loss function for detecting object proposals and bounding boxes. For detecting proposals, the model computes the loss for generated vote clusters and for classifying proposals into a small subset of ScanNet [20] classes. Moreover, it calculates the regression losses for the size and center of the bounding box and classification loss for the size of the predicted bounding box.

We denote the overall loss for our 3D module as L_{3D}. The loss function in our language module is composed of an end-to-end negative log-likelihood across all the antecedent spans as described in Lee et al. [51]. In doing so, it can reason about the factors involved in qualifying a span as a good candidate mention while learning to assign a potential antecedent for the mention. We use L_{lang} to signify the loss for our language module.

Our mapping model is primarily a series of binary classifiers that processes an input coreference cluster and predicts the alignments between clusters and the object proposals. In this, our module learns M-binary object classifiers, where M is the maximum number of object proposals for a given 3D scene obtained from our 3D module. The loss for a binary classifier is computed using binary cross-entropy loss, which can be extended for a multi-label classifier problem.

The final layer of our mapping module returns an $N \times M$-dimensional array, where N represents the maximum number of coreference clusters in a given sentence. Each element in N clusters is represented by an M-dimensional vector, which consists of probabilities of M objects belonging to it. We refine the $N \times M$ outputs and the ground-truth object labels by obtaining the non-padded values before calculating the losses. The valid outputs and non-padded ground-truth object labels for all the cluster elements are processed through the binary cross-entropy loss function, included in the Pytorch library, with reduction set to *none*. We then add the losses for all the valid elements in N clusters.

Furthermore, we compute weights for each class to address the class imbalance in our dataset. We account for the computed class weights when back-propagating the loss so that more weightage is focused on the samples of positive classes and less on the samples of negative classes. The weights for each class are calculated as the ratio of total negative samples to total positive samples of the class. Before back-propagating, we divide the summed losses by the number of non-padded elements to obtain the batch loss. Pytorch's binary cross-entropy loss function documentation further describes this strategy for calculating the class weights.

Therefore, we use the weighted binary cross entropy function to compute the loss across the proposals and represent the loss for our mapping module as L_{map}. The final loss of our DenseRefer3D model is a weighted combination of losses from 3D, language, and mapping modules, and is denoted as $L_{DenseRefer3D} = \alpha L_{3D} + \beta L_{lang} + \gamma L_{map}$, where constants α, β, and γ enable us to perform the scaling of losses.

4.5.3 Training and Inference

Training

We provide details on how our 3D, language, and mapping modules are involved in the training process. We use a similar training strategy to Chen et al. [11] for our 3D module, in that the object proposals generated by VoteNet [73] are merged with the encoded coreference clusters produced by our language module. Our language module processes the input sentence via an end-to-end coreference resolution model [51] with pre-trained weights to generate the clusters where the mentions referring to the same objects belong to the same group. However, since the end-to-end model [51] does not generate singleton mentions, we utilize the ground-truth knowledge to encode mentions that are not part of any clustering. Finally, as described above, the clusters from the language module are fused with object proposals from the 3D module. Our mapping module employs the fused features for learning the cluster-to-object alignments and the objects' bounding boxes for all the coreference clusters in a given sentence.

Inference

During the inference phase, we use the test split to keep the coreference clusters, gold labelling of objects in the clusters, and ground-truth bounding boxes hidden to prevent the model from learning from our test split. This helps maintain fairness for testing and benchmarking our task. Besides, we assess the quality of coreference chains produced by the model against the ground-truth clustering using the evaluation described in Lee et al. [51]. Additionally, we remove the overlapping object proposals using the strategy executed in Chen et al. [11] before merging them with the cluster features. Using the combined features, we evaluate the quality of the predicted bounding boxes and the mapping of coreference clusters with the objects using the hidden ground-truth information. We compute the final $F1$-score as the average of the $F1$-score for coreference clustering and $F1$-scores at 0.25 and 0.5 IOU thresholds for cluster-to-objects mapping.

Chapter 5

Experiments and Results

5.1 Coreference Resolution

This section details the experiments performed involving the task of coreference resolution. Most experiments for resolving coreferences were executed using the end-to-end approach [51] as the baseline method.

5.1.1 Experiment Setup

Pretrained e2e-coref
We employed a pre-trained coreference resolution model [52] to test the performance on different coreference datasets. This set of experiments allowed us to evaluate the model's accuracy, deliver insight into the method's shortcomings, and assess the quality of the existing large-scale coreference datasets. We provide details on executed experimentation using various datasets in the following sections.

(a) CoNLL-2012 dataset
Most research on resolving coreferences in the text has been conducted using CoNLL-2012 dataset [71]. This is due to many English language documents in the dataset, with well-labelled ground-truth clusters of mentions. It contains about 45,000 clusters with more than 150,000 links between approximately 195,000 mentions. It is one of the largest coreference datasets, with 1.6 million English words spanning several domains. Since the end-to-end coreference model [52] was trained on the CoNLL-2012 dataset [71], the model's results on the CoNLL-2012 test set served as a benchmark for comparing with similar datasets, including our DenseRefer3D dataset. The performance was measured using the metrics described in Section 4.4.

(b) Sentences3D dataset
Sentences3D dataset [46] consists of ground-truth alignments of phrases to the real-world objects along with the chains of gold clusters, with each cluster comprising nouns and pronouns that refer to the same entity. This dataset is similar to ours,

DenseRefer3D, containing the mapping of text phrases to the objects in the images. However, it includes just 1449 natural language descriptions with about 3.4 candidate nouns and, on average, 0.5 pronouns per description. Therefore, our dataset is significantly larger than this and is curated using rich 3D scenes from ScanNet dataset [20]. The experiments involving the Sentences3D dataset [46] provided us with a baseline for collating with our dataset and an opportunity to further research improvement areas in our dataset and resulting methods.

(c) Sentences3D dataset (removed non-gold mention clusters)
We made some modifications to the Sentences3D dataset and the model output to investigate the shortcomings of the original dataset and study the effects of the applied refinements on the model's performance. In this experiment, we updated the output of the pre-trained end-to-end model by removing the predicted clusters that did not contain any gold mentions. We performed this step to prevent the scoring scripts from penalizing the precision for the mentions that did not belong to one of the 21 object classes employed in the Sentences3D dataset.

(d) Sentences3D dataset (extracted headwords from predictions)
We updated the model's output by extracting the headwords from the predicted mentions to analyze further the impact of different dataset components on the model. By doing this, we reasoned about the quality of the gold mentions in the coreference chains and obtained an insight into the mention detection part of the model.

(e) Sentences3D dataset (added noun and pronoun phrases to gold mentions)
In addition to the previous experiment, we modified the mentions in the gold coreference clusters by forming noun or pronoun phrases from the available ground-truth words. We examined the influence of including the contextual information surrounding the word on the precision of the model. Since the gold mentions in the Sentences3D dataset consist of singular tokens, performing these experiments motivated us to curate our dataset based on the significance of incorporating context.

(f) DenseRefer3D dataset
We exploited the results of all the previous evaluations on several datasets and their modifications to include the influential features in our desired dataset. We then executed the pre-trained end-to-end model on our DenseRefer3D dataset. This served the purpose of performing the smoke testing to validate the quality of the collected annotations compared to other datasets and to understand the shortcomings of the employed approach.

Fine-tuned e2e-coref

Besides the experiments involving the pre-trained model, we performed fine-tuning of the

Model	Dataset	Metrics									
		MUC			B^3			CEAFϕ_4			
		Precision ↑	Recall ↑	$F1$ ↑	Precision ↑	Recall ↑	$F1$ ↑	Precision ↑	Recall ↑	$F1$ ↑	CoNLL $F1$ ↑
e2e-coref [52]	CoNLL (test) [71]	81.4	79.5	80.4	72.2	69.5	70.8	68.2	67.1	67.6	72.93
Sentences3D [46]	Sentences3D (val) - v2 [46]	83.69	51.08	63.44	88.42	70.02	78.15	-	-	-	-
e2e-coref (pre-trained) [52]	Sentences3D (val) [46]	2.65	1.93	2.23	9.55	6.77	7.92	17.32	13.8	15.36	8.50
e2e-coref (pre-trained) [52]	Sentences3D (val) - v2 [46]	2.9	1.93	2.32	10.46	6.75	8.2	19.13	13.73	15.99	8.83
e2e-coref (pre-trained) [52]	Sentences3D (val) - v3 [46]	76.75	51.12	61.37	80.75	52.13	63.36	79.16	56.8	66.14	63.62
e2e-coref (pre-trained) [52]	Sentences3D (val) - v4 [46]	78.69	52.41	62.92	82.37	53.24	64.67	80.28	57.61	67.08	64.89
e2e-coref (pre-trained) [52]	DenseRefer3D (val)	80.81	47.47	59.81	84	48.05	61.13	79.7	66.01	72.21	64.38
e2e-coref (fine-tuned) [52]	Sentences3D (val) - v2 [46]	61.25	40.8	48.98	67.5	43.26	52.73	74.86	43.26	52.73	51.48

Table 5.1: Coreference resolution experiment results using the end-to-end coreference resolution model on different datasets. Besides running experiments on the original Sentences3D dataset, we make various modifications to the dataset to analyze the rationale for low scores across all the metrics on the original version. Here, v2, v3, and v4 refer to the transformations applied by removing non-gold mention clusters, extracting headwords from prediction, and adding noun & pronoun phrases to gold mentions, respectively.

end-to-end coreference model. We executed this by re-training the mention detection engine to predict chains of coreferences consisting of only headwords. This provided a more reasonable ground for assessing the performance on Sentences3D due to the nature of the gold mentions.

5.1.2 Results

We conducted several coreference resolution experiments using various state-of-the-art models to evaluate the performances using some prominent datasets in the field. Our initial experiments employed the pre-trained end-to-end coreference resolution model [51]. Using this, we report the results obtained on the test split of the CoNLL-2012 dataset [71]. This resulted in an average $F1$ score of 72.93.

Since our dataset and task are closely related to the work performed by Kong et al. [46], we focus extensively on evaluations using the Sentences3D dataset [46] to assess the performance of an end-to-end approach and analyze the complexities involved in the dataset for this task. As a result of this experiment, we observe significantly lower values of the $F1$ measure across the three metrics, with the CoNLL $F1$ score of just 8.50. These low scores are due to the differences between the gold and the predicted mentions in the clusters. Most gold mentions in the Sentences3D dataset only contain headwords and do not include additional characteristics to represent the objects. To further diagnose the cause(s) for the low scores, we make minor independent modifications to the Sentences3D dataset.

As part of the first modification, we remove the predicted clusters if their mentions are not present in the ground-truth clusters. In doing so, we observe a slight increase in the precision values (consequently increasing $F1$ score) across the evaluation metrics, as shown in Table 5.1.

For the second modification, we extract the headwords from all the mentions in the predicted clusters since the ground-truth clusters in the Sentences3D dataset only contain annotations of headwords as mentions. This results in a significant increase in precision and recall values for all three evaluation metrics. By extension, we see a significant jump in the average $F1$ score, with a value of 63.62.

Our final modification involves updating the ground-truth clusters to transform the mentions into noun and pronoun phrases. Since this modification is analogous to the previous one, we observed comparable results for the final average $F1$ score. These modifications to the Sentences3D dataset confirm our assessment of the disparities between ground truth and predicted mentions.

We also ran the pre-trained end-to-end coreference model [51] on our dataset, the DenseRefer3D dataset. This allows us to compare our dataset with the Sentences3D dataset to observe similarities and differences when using a state-of-the-art model. On the DenseRefer3D dataset, the pre-trained model produced the predicted clusters that achieved higher precision and recall values than the experiments on Sentences3D without any applied modifications, thus giving us a higher CoNLL $F1$ score. The scores generated using the DenseRefer3D dataset are lower than Sentences3D - v4 experiments and the ones performed on the CoNLL-2012 dataset. This is due to the availability of postcedents (sentences where the pronoun or pro-form mention precedes the entity mention) in the DenseRefer3D dataset.

Furthermore, our dataset consists of different types of mentions, such as plural expressions that refer to two or more objects. Besides, many annotated mentions in our dataset are very descriptive and include a detailed representation of the main object's attributes and proximity to surrounding objects. Thus, all these factors contribute to making our dataset more challenging.

In addition to using the pre-trained model to conduct our experiments, we performed fine-tuning of the end-to-end coreference resolution model [51] so that the mention detection engine can identify mentions that consist of headwords only. This is similar to the second modification described above but does not require us to manually change the mentions in the predicted clusters. The differences in scores, compared to previous experiments on the Sentences3D, are due to the small size of the dataset, even after applying text augmentation and the impact on clustering caused by transforming mentions into simpler forms.

5.1.3 Singletons Evaluation

The singletons refer to real-world objects but do not have a co-referring phrase representing the same object. Most approaches for resolving coreferences consider pairs of co-referent mentions and disregard the singletons by removing them from gold and predicted clusters [9]. In addition, many notable datasets, such as CoNLL-2012 [71], do not include annotations of singleton mentions. The prominent evaluation metrics for coreference resolution are not well-suited for dealing with singletons, i.e., clusters with just one expression. For instance, *MUC* [90], a link-based metric, only considers coreference chains and links between the phrases in the chains. It ignores the mentions without a coreferent mention during the evaluation. The presence of singletons in the ground truths and outputs generated by the coreference resolvers impact the scores of the evaluation metrics. Some metrics even receive an artificial increase in the scores merely due to the existence of singletons [49]. The detection of singleton mention boundaries are rewarded (or punished) by B^3 [5] and mention-based *CEAF* [57], which leads to bias in preferring models identifying all the mentions over chains of coreferences. As an example, B^3 calculates the precision and recall for coreference resolution by computing the precision and recall values for each detected mention. As a result, the presence of a singleton mention in the gold standard would cause an increase in the overall model scores.

Thus, further research and experimentation are needed to provide an evaluation strategy inclusive of singleton mentions since they are essential in creating a more naturalistic environment. Due to these reasons, we do not include singletons while evaluating the models to keep the analysis equitable compared with existing works and to focus on the performance of identifying links between mentions in coreference clusters.

5.2 Mention Identification

In addition to investigating the coreference clustering performance, we conducted several experiments to analyze the quality of the detected mentions. We outline the details of the mention identification experiments in this section.

5.2.1 Experiment Setup

The coreference experiments detailed above helped enhance the comprehension of the model's interpretability. Furthermore, it provided a good understanding of the quality of several datasets. However, it was essential to dig further in our quest to examine the model's clustering capabilities by inspecting the datasets' effectiveness. We performed several experiments on identifying mentions by the model using different datasets.

Pretrained e2e-coref

Similar to the coreference experiments, we used the pre-trained end-to-end model to measure the performance of its capabilities for detecting mentions. These experiments were performed for original and modified versions of Sentences3D datasets and on our DenseRefer3D dataset. It segmented the overall analysis of coreference clusterings by enabling us to focus solely on evaluating the quality of detected mentions.

Fine-tuned e2e-coref
Sentences3D dataset consists of annotated mentions, either a noun or pronoun type, referring to the same real-world entity. We altered the mention detection engine to enable the model to only predict the relevant phrase's headword. This was valuable for probing the effects of the absence of contextual information on the downstream task involving clustering of mentions.

5.2.2 Results

Besides performing experiments to analyze the coreference clustering on different datasets using an end-to-end model [51], we closely examined the capabilities of the mention detection module responsible for classifying a text span as a suitable mention. Similar to the coreference resolution experiments, we began by employing the provided pre-trained end-to-end model [51] on the original Sentences3D dataset [46]. In this case, we noticed low values of precision and recall of 19.82 and 15.05, respectively. As a result, the $F1$ score for mention identification stood in the mid-10s. The low coreference resolution scores are directly related to the low precision and recall values for the Sentences3D dataset.

We followed the same theme of applying modifications to the Sentences3D dataset as described in the previous section to examine the impact of these slight adjustments to the data on the performance of detecting mentions by the model. Our first modification of removing predicted chains containing non-gold mentions led to a modest increase in the precision, recall, and $F1$ values, as shown in Table 5.2. On the other hand, after applying the second modification of only keeping the headwords in the mentions and removing the rest, we witnessed a notable boost in precision and a substantial increase in the recall values. This led to an increase in the $F1$ score from 17.7 to 73.16.

With our final modification of converting the ground-truth mentions to noun and pronoun phrases (that includes the ground-truth mentions), we saw even more considerable improvements in the final scores as presented in the Table 5.2.

Furthermore, we produced predictions of coreference clusters using the pre-trained end-to-end model on the validation split of the DenseRefer3D dataset (data splits as defined in Chen et al. [11]). Using the output, we analyzed the quality of the predicted mentions by the model and reported precision, recall and $F1$ scores of 89.15, 60.81, and 72.3, respectively. In this case, we observed high precision scores likely due to fewer pronoun variations. Our dataset consists of a large number of pronouns of a similar type, such as *this* and *it*. Whereas the slightly lower recall value is due to several mentions specific to our dataset describing inanimate objects in an indoor environment.

For the experiment involving the fine-tuned end-to-end coreference model for re-training the model to only predict headwords in the mentions, we observed scores marginally lower

Model	Dataset	Metrics Precision ↑	 Recall ↑	 $F1$ ↑
e2e-coref (pre-trained) [52]	Sentences3D (val) [46]	19.82	15.05	17.1
e2e-coref (pre-trained) [52]	Sentences3D (val) - v2 [46]	21.7	14.94	17.7
e2e-coref (pre-trained) [52]	Sentences3D (val) - v3 [46]	89.69	61.77	73.16
e2e-coref (pre-trained) [52]	Sentences3D (val) - v4 [46]	90.79	62.53	74.06
e2e-coref (pre-trained) [52]	DenseRefer3D (val)	89.15	60.81	72.3
e2e-coref (fine-tuned) [52]	Sentences3D (val) - v2 [46]	80.93	55.43	65.8

Table 5.2: Mention identification experiment results using the end-to-end coreference model [52] on different datasets. The low coreference resolution $F1$ scores on the Sentences3D dataset [46] correlate with the lower precision and recall values for mention detection.

than the ones obtained for the experiment with the second modification, with a 10.06% decrease in $F1$ score. As reasoned above in the coreference experiment results, the limitation in the number of available gold mentions in the Sentences3D dataset negatively impacts the evaluation scores.

5.3 Coreference Resolution using 3D

This section outlines the baseline methods we designed for our task. In addition, we summarize the evaluation metrics to assess the performance of the baseline models.

5.3.1 Experiments

Baseline-1

In the first baseline method, we freeze the object detection module in our network. This is done to make use of ground-truth bounding boxes during the process of matching the clusters with the objects. This setup helps us create an upper baseline method concerning the quality of the objects detected from the scene. It signifies the room for improvement for our 3D module if it backpropagates the loss to learn the weights and predict the object boxes, hence serving as a standard comparison with the prospective methods.

As for the language module, we deploy a similar strategy to the one described above for the 3D module. In this case, we directly operate on ground truth coreference chains for extracting cluster features. The justification for doing this is similar to the 3D module upper baseline details mentioned above. It allows us to generate an upper baseline for our language module to discern the potential improvement scope for clustering mentions into groups based on the objects represented by them.

We then combine the features from the 3D and language module baseline to learn the mapping of the coreference clusters to the 3D objects. Thus, our final loss in this setup only consists of the loss from our match module, while the other components of loss from the

3D and language module do not contribute to the final representation. For each cluster, the match module predicts the probability of each object proposal as a referent and localizes the objects using the ground-truth bounding boxes for the classified objects.

Baseline-2

In our second baseline method, we utilize a pre-trained VoteNet [73] model as part of our 3D module. We load the learned weights published by Qi et al. [73] and generate the predicted vote features and bounding boxes for the object proposals. We freeze the object detection pipeline, similar to the first baseline, to avoid updating the loaded pre-trained model weights. It helps set up a practical foundation for the progression of methods.

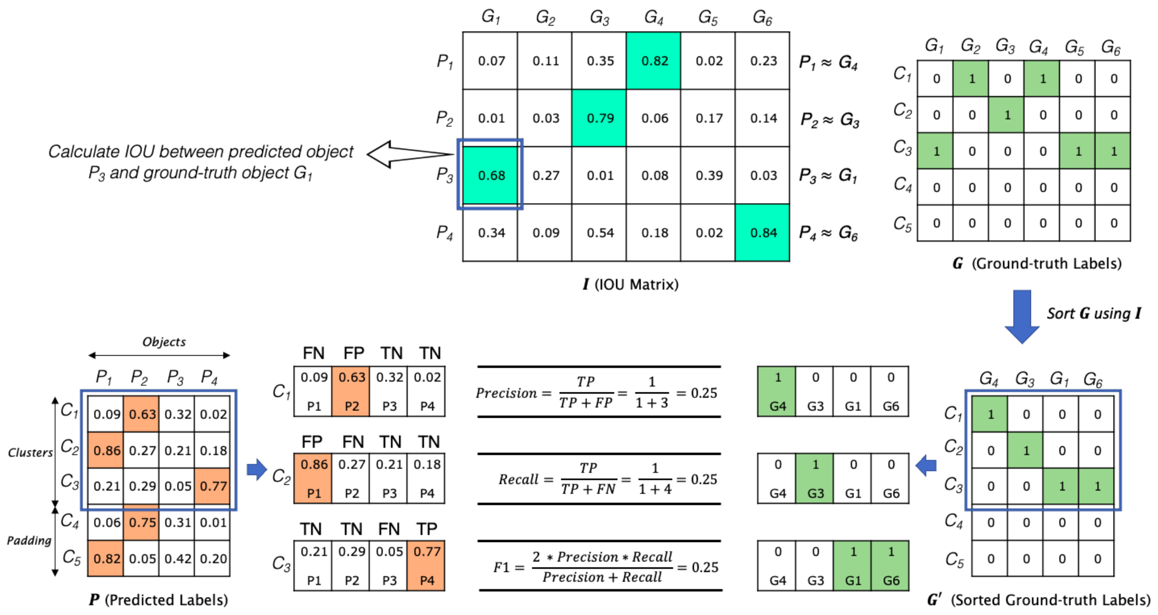
Similarly, we employ the pre-trained e2e-coref [51] model to generate the predictions of coreference clusters. Our language module then extracts the features from the obtained chains of coreferences. The language module's loss component does not affect the final loss of our DenseRefer3D network.

Our match module combines the features extracted from pre-trained VoteNet [73] and pre-trained e2e-coref [51] and learns to classify the objects for each extracted coreference cluster in the given sentences. It then assigns the highest scoring objects to the clusters and localizes them by selecting the predicted bounding boxes for the mapped objects.

5.3.2 DenseRefer3D Evaluation Metrics

As described in Section 4.5.1, our DenseRefer3D model consists of 3D, language, and mapping modules. Thus, to evaluate the complete model, we must incorporate the performance of respective modules to generate the overall assessment results. The first component of our evaluation is responsible for evaluating the quality of the coreference clusters. We use the same technique used in Lee et al. [51] to calculate the $F1_{lang}$ as the average of the $F1$-scores from three prominent coreference resolution metrics: MUC [90], B^3 [5], and $CEAF_{\phi_4}$ [57].

The final two evaluation components consider the quality of assessing object proposals and matching the coreference clusters with the oriented bounding boxes. We compute an $M \times N$ IOU matrix for each scene. In this matrix, each value represents an intersection-over-union (IOU) overlap between the predicted and the ground-truth object bounding boxes. This helps determine the alignments between predictions and target objects, which we require since the ordering of the predicted object might differ from that of ground-truth ones. In other words, if we consider the set of ground-truth objects as $G = \{g_1, g_2, ..., g_M\}$ and predicted objects as $P = \{p_1, p_2, ..., p_M\}$, then it cannot be insinuated that target object g_1 corresponds to predicted object p_1 without verifying the overlap between their bounding boxes. After that, we employ the IOU matrix to calculate the evaluation metrics for our 3D



Calculate IOU between predicted object P_3 and ground-truth object G_1
I (IOU Matrix)
$P_1 \approx G_4$
$P_2 \approx G_3$
$P_3 \approx G_1$
$P_4 \approx G_6$
G (Ground-truth Labels)
Sort G using I
G' (Sorted Ground-truth Labels)
P (Predicted Labels)
Objects
Clusters
Padding
$Precision = \frac{TP}{TP+FP} = \frac{1}{1+3} = 0.25$
$Recall = \frac{TP}{TP+FN} = \frac{1}{1+4} = 0.25$
$F1 = \frac{2*Precision*Recall}{Precision+Recall} = 0.25$

Figure 5.1: DenseRefer3D evaluation strategy. We compute an IOU matrix to determine the overlap between predicted and ground-truth object bounding boxes. We sort the gold cluster-to-objects label mappings using this matrix to accurately compute precision and recall metrics.

module, designated as $F1_{3D}$.

As we mentioned in Section 4.5.2, for a cluster in N coreference clusters, our model outputs a $1 \times M$ vector consisting of probabilities for each object in M aligned with the cluster. We filter the prediction results by keeping probabilities greater than the threshold value (we use 0.5 as the threshold in our experiments) and setting them to one. Moreover, we set the cluster predictions to zero for padded gold clusters. We then utilize the IOU matrix to sort the ground-truth cluster-to-objects labels so that the target labels correspond with their prediction counterparts.

The following steps involve calculating true/false positives, true/false negatives, precision, recall, and $F1$. We mark the prediction as true positive if the label in both gold and the predicted cluster is one. Additionally, we consider prediction a false positive if the predicted cluster label is one, but the target label is zero. In contrast, the predicted cluster object with the label assigned as zero and the ground-truth label as one is marked as a false negative. Finally, we mark the object prediction as a true negative if the label for both the prediction and ground-truth is one. However, we remove the padded values from the true negatives using the masks to reflect the actual count of true negatives accurately. We perform the above steps for IOU overlap threshold 0.25 & 0.50, respectively.

These metrics are totalled across all the batches, and at the end of an epoch, we calculate precision, recall, accuracy, and $F1$-score using these metrics. We denote the $F1$-score for our third component as $F1_{map}$. The final $F1$-score for our Denserefer3D model is obtained by averaging $F1_{lang}$, $F1_{3D}$ and $F1_{map}$. i.e.,

$$F1_{DenseRefer3D} = \frac{(F1_{lang} + F1_{3D} + F1_{map})}{3}$$

5.3.3 Discussion

We proposed a new set of experiments involving our DenseRefer3D dataset besides conducting coreference resolution and mention identification analyses. We undertook the task of aligning the natural language referring expressions contained in the sentences with the 3D objects referred by them. Accordingly, we organized two baseline experiments involving our DenseRefer3D model: a frozen object detection module with ground-truth coreference clusters and a pre-trained VoteNet [73] object detector with pre-trained end-to-end coreference model [51].

Since this is a first-of-its-kind task which includes modules such as object detection, coreference clustering, and clusters-to-objects alignments, we came up with the above experiments as a stepping stone for further investigation in this effort. We compute the precision, recall, and $F1$ values of the individual modules in the architecture, as well as the cumulative scores defined in Section 5.3.2.

For our first baseline method, we do not incorporate the results of the object detection module in our final scores because we freeze this module to simplify the experiment and focus more on the phrases-to-objects mapping. As part of our second baseline method, we use the results reported by Qi et al. [73] for the pre-trained VoteNet. We do not have empirical results to report for this new task. However, we provide the model architecture and evaluation strategy for baseline methods to foster further research and experimentation for resolving text coreferences using 3D features.

Chapter 6

Conclusion and Future Work

This effort aimed to advance the research in coreference resolution and referring expression comprehension. More specifically, we desired to create a first-of-its-kind large-scale dataset using both language and vision modalities to tackle these problems using supplementary information available through the added modality. Most literature concerning coreference resolution tasks have leaned exclusively on language-based datasets. However, in this problem, the mentions directly refer to real-world entities in most cases. Therefore, it becomes imperative to explore solutions using visual information and textual data. Further, utilizing real-world 3D objects to curate the desired datasets is beneficial due to the benefits of employing a 3D environment.

To that end, we began by reviewing the recent efforts in the tasks mentioned above to understand the different approaches involved. We investigated several notable coreference resolution architectures extensively used in this problem, focusing on prevalent work by Lee et al. [51]. This was the first work to resolve coreferences in an end-to-end architecture. In other words, it did not require information from external preprocessing channels, which had been the norm in most earlier efforts. We reviewed the two primary operations involved: mention detection and clustering, to comprehend the model's capability to select reasonable spans of texts as mentions and organize them into groups according to the objects represented by these mentions. This acted as a practical starting point for our research work by accentuating the shortcomings of similar procedures.

To further discover the improvement areas for this task, we inspected several works that contributed datasets particularly to address this problem. We found that most datasets were created solely using textual data. Most similar work to ours, Sentences3D [46] utilized both textual sentences and images. However, the dataset size is too small to provide any functional advantage using modern deep learning-based approaches, with only 61,195 annotated mentions belonging to just 1811 coreference clusters. In addition, it utilized RGB-D images to annotate the alignments between text tokens and visual object segments. Moreover, the textual annotations mainly consisted of singular tokens, thus lacking valuable details about the primary object and its surroundings. Hence, we preferred to use natural language sentences that richly describe the objects and their characteristics concerning secondary objects present in a 3D setting.

We also surveyed the annotation tools used to create the existing datasets in coreference resolution and referring expression comprehension tasks. Most of these task-specific tools did not support the rendering of language and 3D in the same layout. This was essential as we employed natural language descriptions and 3D RGB-D scenes from ScanRefer [11] and ScanNet [20] datasets, respectively, to collect the desired annotations. One of the significant drawbacks of the existing annotation tools was the complicated ways of visualizing the chains of coreferences and the alignments between phrases and real-world objects. Moreover, most of these tools were not developed to collect large-scale datasets and did not provide a cohesive environment easily deployable on a crowdsourcing platform.

Through the knowledge acquired by conducting these assessments, we created a comprehensive new 3D tool, DenseRefer3D Annotator, consisting of a suite of interfaces for annotating, visualizing, verifying, and managing the entire data collection process in an end-to-end manner. Our tool enables the users to quickly select textual phrases using the familiar drag-and-drop approach and link 3D objects directly with the selected phrases with a simple mouse action. It outlines all the phrases linked with matching objects using the same colours, thus making it easy to visualize the coreference clusters. Moreover, it provides practical and intuitive methods for highlighting the alignments between phrases and 3D objects, enabling comprehensive visualization of the provided annotations. Since we set out to collect annotations involving nearly 50,000 descriptions and more than 700 3D scenes, we wanted to employ the resources provided by the MTurk crowdsourcing platform to recruit workers for creating a large-scale dataset. To that end, we developed several interfaces to create and review HITs on MTurk directly from our DenseRefer3D Annotator, enabling us to manage various data collection procedures effectively. We collected the annotations of 207,177 phrases and 124,270 chains of coreferences, with around two mentions per coreference cluster. The referring expressions in our dataset consisted of, on average, 3.87 tokens, thereby providing detailed representations of objects. Our dataset also consists of annotations of 74,003 singletons (i.e., the mentions that are not co-referential).

We performed several experiments using variations of the end-to-end coreference resolution model [52] on different datasets. For instance, we conducted many experiments using a pre-trained coreference model [52] to analyze the quality of coreference clusters in the Sentences3D dataset [46]. We further investigated the cause of low scores obtained using this dataset by modifying the annotations and fine-tuning the model. In doing so, we observed significant improvements across the prominent coreference evaluation metrics and the CoNLL $F1$ score. We also executed similar experiments on our dataset, DenseRefer3D, for sanity-checking our collected annotations, understanding the complexities involved in our coreference chains, and for comparative purposes with other similar datasets. Additionally, we conducted numerous experiments to analyze the model's mention detection capabilities on different datasets, including our DenseRefer3D dataset. With these experiments, we observed that the DenseRefer3D dataset provides a challenging setting for resolving coreferences. This is due to many annotated coreference clusters, pronouns, singletons, and varying mention types, such as plural phrases.

Furthermore, we propose a new task of aligning all the entity-describing mentions with the objects referred to by them directly in 3D scenes. This entails figuring out the text spans that form reasonable mentions, clustering the ones referring to the same objects in groups, and localizing the referent objects with a 3D bounding box surrounding the target objects. In essence, our task combines the two discussed problems, coreference resolution and referring expression comprehension, and performs both tasks end-to-end. Our model comprises three components: a language module for extracting coreference cluster-level features, a 3D module for obtaining object features and oriented bounding boxes, and a mapping module to learn the alignment between coreference clusters of mentions (including singletons) and 3D objects in an indoor environment.

As part of this, we present two baseline methods involving the modules in our architecture. The first method serves as an upper baseline for the 3D module as it operates on the ground-truth object bounding boxes and the language module as it extracts cluster features using gold coreference chains. This setup reflects the best-case scenario for object detection and coreference clustering. For our second baseline, as part of our 3D module, we use pre-trained VoteNet [73] to extract object proposals and generate bounding box predictions. Following a similar theme, we utilize a pre-trained end-to-end model [52] to produce predicted coreference chains, which our language module employs to extract cluster features. Our mapping module fuses the 3D and language features to learn the alignment of coreference clusters to 3D objects. In this, we treat the problem as a series of binary classifiers, each learning to classify an object belonging to a coreference cluster.

These methods can be easily extended to cover more comprehensive scenarios. For instance, we could start with modification to the language module in the first baseline by enabling it to learn to detect mentions and perform clustering. Similarly, we could keep the language module in the second baseline intact but train the 3D module to generate object proposals and bounding box predictions. Finally, the natural progression would be to train all the modules end-to-end without loading any pre-trained models or using other external input sources.

Additionally, many similar annotation tools are either closed-source or unavailable for free use. We created our DenseRefer3D Annotator to make it accessible for similar existing and future projects. As part of this work, we provide open access to the code repository[3] and tool setup instructions. With a few modifications, our tool can be adapted for visualizing coreference relations and phrases-to-objects mappings in existing 3D and language datasets and for future similar efficient data collection efforts on crowdsourcing platforms with a minimal learning curve.

Lastly, our DenseRefer3D dataset[4] is suitable for future tasks dealing with navigating a real-world 3D indoor scene using rich sentences containing detailed referring expressions. We began by introducing the concept of a navigational agent that comprehends the user's natural language queries, identifies the intended target object, and ultimately fetches the correct item. With this work, we aimed to bridge the gap in the combined language and vision domain by providing a large-scale language and 3D dataset and to facilitate further research in improving 3D scene understanding using natural language text representations of objects.

Bibliography

[1] Panos Achlioptas, Ahmed Abdelreheem, Fei Xia, Mohamed Elhoseiny, and Leonidas Guibas. ReferIt3D: Neural listeners for fine-grained 3D object identification in real-world scenes. In *European Conference on Computer Vision*, pages 422–440. Springer, 2020.

[2] Explosion AI. An annotation tool for AI, Machine Learning & NLP, 2017. `https://prodi.gy/`.

[3] Peter Anderson, Qi Wu, Damien Teney, Jake Bruce, Mark Johnson, Niko Sünderhauf, Ian Reid, Stephen Gould, and Anton Van Den Hengel. Vision-and-Language Navigation: Interpreting visually-grounded navigation instructions in real environments. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3674–3683, 2018.

[4] Amazon Mechanical Turk API. Amazon.com, Inc., 2021. `https://docs.aws.amazon.com/mturk/index.html`.

[5] Amit Bagga and Breck Baldwin. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, volume 1, pages 563–566. Citeseer, 1998.

[6] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

[7] Aaron M. Bornstein, Arie Cattan, and Ido Dagan. CoRefi: A crowd sourcing suite for coreference annotation. In *EMNLP*, 2020.

[8] Aaron M. Bornstein, Arie Cattan, and Ido Dagan. CoRefi (Demo): A crowd sourcing suite for coreference annotation, 2020. `https://aribornstein.github.io/corefidemo/`.

[9] Arie Cattan, Alon Eirew, Gabriel Stanovsky, Mandar Joshi, and Ido Dagan. Streamlining cross-document coreference resolution: Evaluation and modeling. *arXiv preprint arXiv:2009.11032*, 2020.

[10] Angel Chang, Angela Dai, Thomas Funkhouser, Maciej Halber, Matthias Niessner, Manolis Savva, Shuran Song, Andy Zeng, and Yinda Zhang. Matterport3D: Learning from RGB-D data in indoor environments. *International Conference on 3D Vision (3DV)*, 2017.

[11] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. ScanRefer: 3D object localization in RGB-D scans using natural language. *16th European Conference on Computer Vision (ECCV)*, 2020.

[12] Hong Chen, Zhenhua Fan, Hao Lu, Alan Loddon Yuille, and Shu Rong. PreCo: A large-scale dataset in preschool vocabulary for coreference resolution. In *EMNLP*, 2018.

[13] Zhenfang Chen, Peng Wang, Lin Ma, Kwan-Yee K Wong, and Qi Wu. Cops-Ref: A new dataset and task on compositional referring expression comprehension. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10086–10095, 2020.

[14] Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *arXiv preprint arXiv:1412.3555*, 2014.

[15] Volkan Cirik, Taylor Berg-Kirkpatrick, and Louis-Philippe Morency. Refer360°: A referring expression recognition dataset in 360: A referring expression recognition dataset in 360° images. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7189–7202, 2020.

[16] Kevin Clark and Christopher D. Manning. Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 643–653, Berlin, Germany, August 2016. Association for Computational Linguistics.

[17] Amazon Elastic Compute Cloud. Amazon.com, Inc., 2021. `https://aws.amazon.com/ec2/`.

[18] Amazon CloudFront. Amazon.com, Inc., 2021. `https://aws.amazon.com/cloudfront/`.

[19] Blender Online Community. *Blender - a 3D modelling and rendering package*. Blender Foundation, Stichting Blender Foundation, Amsterdam, 2018. URL `http://www.blender.org`.

[20] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. ScanNet: Richly-annotated 3D reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[21] Abhishek Das, Satwik Kottur, Khushi Gupta, Avi Singh, Deshraj Yadav, José MF Moura, Devi Parikh, and Dhruv Batra. Visual Dialog. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 326–335, 2017.

[22] Pascal Denis and Jason Baldridge. A ranking approach to pronoun resolution. In *IJCAI*, volume 158821593, 2007.

[23] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

[24] Tristan Edwards. SweetAlert, 2021. URL `https://sweetalert.js.org/`.

[25] Matthew Eernisse. EJS, 2021. `https://ejs.co/`.

[26] Explosion AI. An annotation tool for AI, Machine Learning & NLP, 2017. `https://demo.prodi.gy/?=null&view_id=rel_coref/`.

[27] Express. IBM Inc., 2021. `https://expressjs.com/`.

[28] David Ferrucci and Adam Lally. UIMA: An architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10(3-4):327–348, 2004.

[29] AWS SDK for JavaScript. Amazon.com, Inc., 2021. `https://docs.aws.amazon.com/AWSJavaScriptSDK/v3/latest/clients/client-mturk/index.html`.

[30] Boto3 The AWS SDK for Python. Amazon.com, Inc., 2021. `https://github.com/boto/boto3`.

[31] OpenJS Foundation. Node.js, 2021. `https://nodejs.org/en/`.

[32] Abbas Ghaddar and Philippe Langlais. WikiCoref: An english coreference-annotated corpus of wikipedia articles. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 136–142, 2016.

[33] Google. Google Inc., 2021. `https://v8.dev/`.

[34] Klaus Greff, Rupesh K Srivastava, Jan Koutník, Bas R Steunebrink, and Jürgen Schmidhuber. LSTM: A search space odyssey. *IEEE transactions on neural networks and learning systems*, 28(10):2222–2232, 2016.

[35] Khronos Group. WebGL, 2021. `https://www.khronos.org/webgl/`.

[36] Aria Haghighi and Dan Klein. Simple coreference resolution with rich syntactic and semantic features. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 1152–1161, 2009.

[37] Jared Hanson. Passport, 2021. `https://www.passportjs.org/`.

[38] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[39] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[40] Drew A Hudson and Christopher D Manning. GQA: A new dataset for compositional question answering over real-world images. *arXiv preprint arXiv:1902.09506*, 3(8), 2019.

[41] Amazon Inc. Amazon Mechanical Turk, 2021. `https://www.mturk.com`.

[42] Justin Johnson, Bharath Hariharan, Laurens Van Der Maaten, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2901–2910, 2017.

[43] Daniel Jurafsky and James Martin. *Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.* 2021.

[44] Sahar Kazemzadeh, Vicente Ordonez, Mark Matten, and Tamara Berg. ReferItGame: Referring to objects in photographs of natural scenes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 787–798. Association for Computational Linguistics, October 2014.

[45] Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. The INCEpTION Platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9, Santa Fe, New Mexico, 2018. URL `https://www.aclweb.org/anthology/C18-2002`.

[46] Chen Kong, Dahua Lin, Mohit Bansal, Raquel Urtasun, and Sanja Fidler. What are you talking about? text-to-image coreference. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3558–3565, 2014.

[47] Satwik Kottur, José MF Moura, Devi Parikh, Dhruv Batra, and Marcus Rohrbach. Visual coreference resolution in visual dialog using neural module networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–169, 2018.

[48] Satwik Kottur, Seungwhan Moon, Alborz Geramifard, and Babak Damavandi. SIMMC 2.0: A task-oriented dialog dataset for immersive multimodal conversations. In *EMNLP*, 2021.

[49] Sandra Kübler and Desislava Zhekova. Singletons and coreference resolution evaluation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing 2011*, pages 261–267, 2011.

[50] Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational linguistics*, 39(4):885–916, 2013.

[51] Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end Neural Coreference Resolution. *arXiv preprint arXiv:1707.07045*, 2017.

[52] Kenton Lee, Luheng He, and Luke Zettlemoyer. Higher-order Coreference Resolution with Coarse-to-fine Inference. *arXiv preprint arXiv:1804.05392*, 2018.

[53] Dahua Lin, Sanja Fidler, and Raquel Urtasun. Holistic scene understanding for 3D object detection with rgbd cameras. In *Proceedings of the IEEE international conference on computer vision*, pages 1417–1424, 2013.

[54] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[55] Runtao Liu, Chenxi Liu, Yutong Bai, and Alan L Yuille. CLEVR-Ref+: Diagnosing visual reasoning with referring expressions. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4185–4194, 2019.

[56] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. ViLBERT: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

[57] Xiaoqiang Luo. On coreference resolution performance metrics. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 25–32, 2005.

[58] Junhua Mao, Jonathan Huang, Alexander Toshev, Oana Camburu, Alan L Yuille, and Kevin Murphy. Generation and comprehension of unambiguous object descriptions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 11–20, 2016.

[59] MongoDB. MongoDB Inc., 2021. `https://www.mongodb.com/`.

[60] Seungwhan Moon, Satwik Kottur, Paul A Crook, Ankita De, Shivani Poddar, Theodore Levin, David Whitney, Daniel Difranco, Ahmad Beirami, Eunjoon Cho, et al. Situated and interactive multimodal conversations. *arXiv preprint arXiv:2006.01460*, 2020.

[61] Nafise Sadat Moosavi and Michael Strube. Lexical features in coreference resolution: To be used with caution. *arXiv preprint arXiv:1704.06779*, 2017.

[62] Thomas S Morton. Using coreference for question answering. In *Coreference and Its Applications*, 1999.

[63] mrdoob. three.js, 2021. `https://threejs.org/`.

[64] Christoph Müller and Michael Strube. Multi-level annotation of linguistic data with MMAX2. In Sabine Braun, Kurt Kohn, and Joybrato Mukherjee, editors, *Corpus Technology and Language Pedagogy: New Resources, New Tools, New Methods*, pages 197–214. Peter Lang, Frankfurt a.M., Germany, 2006.

[65] James N. Tippy.js, 2021. `https://atomiks.github.io/tippyjs/`.

[66] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012.

[67] Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 104–111, 2002.

[68] NGINX. Nginx, Inc., 2021. https://www.nginx.com/.

[69] Jeffrey Pennington, Richard Socher, and Christopher D Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[70] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL*, 2018.

[71] Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. CoNLL-2012 Shared Task: Modeling multilingual unrestricted coreference in Ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40, 2012.

[72] Sameer Pradhan, Xiaoqiang Luo, Marta Recasens, Eduard Hovy, Vincent Ng, and Michael Strube. Scoring coreference partitions of predicted mentions: A reference implementation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 30–35, Baltimore, Maryland, June 2014. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/P14-2006.

[73] Charles R Qi, Or Litany, Kaiming He, and Leonidas J Guibas. Deep hough voting for 3D object detection in point clouds. In *proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 9277–9286, 2019.

[74] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. *Advances in neural information processing systems*, 30, 2017.

[75] Yuankai Qi, Qi Wu, Peter Anderson, Xin Wang, William Yang Wang, Chunhua Shen, and Anton van den Hengel. REVERIE: Remote embodied visual referring expression in real indoor environments. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9982–9991, 2020.

[76] Altaf Rahman and Vincent Ng. Supervised models for coreference resolution. In *Proceedings of the 2009 conference on empirical methods in natural language processing*, pages 968–977, 2009.

[77] Joseph Redmon and Ali Farhadi. YOLO9000: Better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017.

[78] Nils Reiter. CorefAnnotator - A New Annotation Tool for Entity References. In *Abstracts of EADH: Data in the Digital Humanities*, December 2018. doi: 10.18419/opus-10144.

[79] AWS SDK. Amazon.com, Inc., 2021. https://aws.amazon.com/developer/tools/.

[80] Sentences3D Dataset Browser. What are you talking about? text-to-image coreference, 2014. https://www.cs.toronto.edu/~fidler/projects/sentences3Ddataset_2.html/.

[81] Paul Hongsuck Seo, Andreas Lehrmann, Bohyung Han, and Leonid Sigal. Visual reference resolution using attention memory for visual dialog. *Advances in neural information processing systems*, 30, 2017.

[82] Amazon Simple Storage Service. Amazon.com, Inc., 2021. `https://aws.amazon.com/s3/`.

[83] Amazon Web Services. Amazon.com, Inc., 2021. `https://aws.amazon.com/`.

[84] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[85] Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4): 521–544, 2001.

[86] Google Data Arts Team. dat.GUI, 2021. `https://github.com/dataarts/dat.gui`.

[87] Maxim Tkachenko, Mikhail Malyuk, Nikita Shevchenko, Andrey Holmanyuk, and Nikolai Liubimov. Label Studio: Data labeling software, 2020-2021. URL `https://github.com/heartexlabs/label-studio`. Open source software available from https://github.com/heartexlabs/label-studio.

[88] Kristina Toutanova, Dan Klein, Christopher D Manning, and Yoram Singer. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 252–259, 2003.

[89] Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word Representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th annual meeting of the association for computational linguistics*, pages 384–394, 2010.

[90] Marc Vilain, John D Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *Sixth Message Understanding Conference (MUC-6): Proceedings of a Conference Held in Columbia, Maryland, November 6-8, 1995*, 1995.

[91] Kellie Webster, Marta Recasens, Vera Axelrod, and Jason Baldridge. Mind the GAP: A balanced corpus of gendered ambiguous pronouns. *Transactions of the Association for Computational Linguistics*, 6:605–617, 2018.

[92] Ralph Weischedel, Martha Palmer, Mitchell Marcus, Eduard Hovy, Sameer Pradhan, Lance Ramshaw, Nianwen Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini, Mohammed El-Bachouti, Robert Belvin, and Ann Houston. OntoNotes release 5.0 LDC2013T19. *Linguistic Data Consortium, Philadelphia, PA*, 23, 2013.

[93] WinstonJS. Winston, 2021. `https://github.com/winstonjs/winston`.

[94] Sam Wiseman, Alexander M. Rush, and Stuart M. Shieber. Learning global features for coreference resolution. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 994–1004, San Diego, California, June 2016. Association for Computational Linguistics.

[95] Qi Wu, Damien Teney, Peng Wang, Chunhua Shen, Anthony Dick, and Anton van den Hengel. Visual Question Answering: A survey of methods and datasets. *Computer Vision and Image Understanding*, 163:21–40, 2017.

[96] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2695–2702. IEEE, 2012.

[97] Xiaofeng Yang, Guodong Zhou, Jian Su, and Chew Lim Tan. Coreference resolution using competition learning approach. In *Proceedings of the 41st annual meeting of the association for computational linguistics*, pages 176–183, 2003.

[98] Licheng Yu, Zhe Lin, Xiaohui Shen, Jimei Yang, Xin Lu, Mohit Bansal, and Tamara L Berg. MAttNet: Modular attention network for referring expression comprehension. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1307–1315, 2018.

[99] Xintong Yu, Hongming Zhang, Yangqiu Song, Yan Song, and Changshui Zhang. What You See is What You Get: Visual pronoun coreference resolution in dialogues. *arXiv preprint arXiv:1909.00421*, 2019.

[100] Dmitry Zelenko, Chinatsu Aone, and Jason Tibbetts. Coreference resolution for information extraction. In *Proceedings of the Conference on Reference Resolution and Its Applications*, pages 24–31, 2004.

[101] Peng Zhang, Yash Goyal, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Yin and Yang: Balancing and answering binary visual questions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5014–5022, 2016.

[102] Zilong Zheng, Wenguan Wang, Siyuan Qi, and Song-Chun Zhu. Reasoning visual dialogs with structural and partial observations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6669–6678, 2019.