Scan and Tell: 3D Dense Captioning using Sparse Convolutions

Baris Sen
baris.sen@tum.de
Technical University of Munich

Ayhan Can Erdur
can.erdur@tum.de
Technical University of Munich

Abstract

Computer Vision and NLP are the two emerging fields of AI. In order to boost the capabilities of computers and improve the human-machine interaction, it is essential to incorporate these fields in a fundamental human task: seeing the surrounding objects and describing them. For this task, the very first step that a computer should make is to successfully understand the semantics of a given scene. This task has been well-explored in images but remains as a challenging problem for 3D inputs. In this work, we utilize the state-of-the-art 3D instance segmentation model as the detection backbone in the dense captioning pipeline to understand the scene better. The new detection mechanism clearly outperforms the baseline detection backbone resulting in better Intersection over Union (IoU) scores and captioning scores such as BLEU, METEOR, CIDEr, and ROUGE.

1. Introduction

There has been lots of research combining visual and natural language processing together in several tasks. Dense captioning is one of them, where multiple objects in a scene are localized and described. However, the progress so far in this task is limited to 2D images, and captioning in 3D scenarios remains to be explored. The very first 3D dense captioning model is introduced by Chen et al. [4], which is referred to as *Scan2Cap*. In their work, they use *VoteNet* [12] for the 3D object detection part of the network.

PointGroup [6] is an instance segmentation network that utilizes sparse convolutions [8] in their model. Sparse convolutions approximate regular convolutions through sparse convolutional kernel matrices [8]. Consequently, they manage to zero out 90% of the parameters of the convolution, which significantly reduces the computational complexity of the convolutional layers [8]. By utilizing sparse convolutions and their successful point clustering method, PointGroup achieves state-of-the-art 3D instance segmentation results in the ScanNet [5] benchmark at the time of publishing (April 2020).

In our project, we stand by the idea that, once it has a better visual understanding of a given scene, the dense captioning network may make more accurate descriptions. To prove this, we utilize PointGroup as the detection backbone for the dense captioning task, and qualitatively compare the outcomes with the current baseline that uses VoteNet.

2. Related Work

Our work combines the high-level field of 3D computer vision and captioning. The joint field of computer vision and natural language is well-explored in various tasks with 2D images, e.g., image captioning, dense captioning, visual grounding, text-to-image generation. However, the named tasks are open to research with 3D data. The reason behind that can be argued to be that 3D computer vision itself is a challenging and improving area.

As two of the few examples of natural language with 3D data, Chen et al. [3] and Achlioptas et al. [1] independently introduce novel datasets and models to localize objects using language queries on a scene level. More recently, Chen et al. [4] expand their work to detect multiple objects and generate descriptions to them, i.e., dense captioning in 3D. They utilize VoteNet [12] as the 3D object detection backbone of their proposed architecture. A newer model, PointGroup [6], shows impressive results in the complex task of 3D instance segmentation. In our work, we integrate PointGroup to the dense captioning network.

3. General Approach

The main focus of this work is to improve the general captioning pipeline by improving the object detection performance of the model, thus improving the model's understanding of the scene. In order to validate this hypothesis, we define two research questions:

- **RQ1:** Does the PointGroup architecture outperform the VoteNet architecture on the object detection task?
- **RQ2:** Does the network architecture that performs better in detecting objects lead to better captioning?

For answering the research questions, we perform a quantitative experiment for each of them. First, we use VoteNet

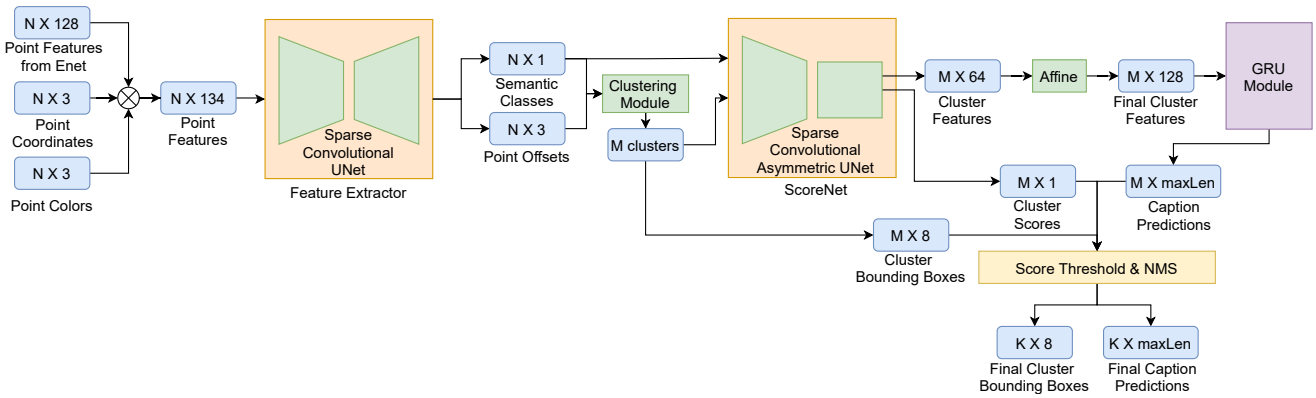

Figure 1. Architecture of the captioning network using the PointGroup backbone. First, ENet point features, point coordinates and point colors are concatenated into 134-dimensional feature vectors. These pointwise feature vectors are fed into a sparse convolutional UNet that outputs the semantic predictions for each point as well as their offset towards cluster center. These outputs are used in the clustering module to create cluster proposals. The clusters and semantic predictions and point clusters are fed into ScoreNet. ScoreNet returns the score and the feature vector for each cluster. GRU modules generated captions for each cluster using the feature vectors. Lastly, the cluster scores, cluster captions, and the cluster bounding boxes -which are calculated naively from point coordinates in each cluster- are filtered using NMS and score thresholds and returned as output.

and PointGroup as object detection models and compare their performance using the mAP (mean average precision) scores thresholded by different levels of IoU. In the next step, we integrate both of the models into the same captioning pipeline as detection backbones and measure the captioning performance of the networks using common captioning metrics.

Initially, our goal has been to compare the detection backbones using the state-of-the-art 3D dense captioning model, namely Scan2Cap. However, due to the complexity of the Scan2Cap model and the time constraints of the project, we compare the backbones using a simpler captioning module. Nevertheless, this comparison also provides us with the sufficient evidence to draw the conclusion about which backbone outperforms the other. Integrating the better backbone to the state-of-the-art 3D dense captioning model remains a future step after the project.

4. Method

We follow a simple end-to-end architecture for the dense captioning task. We utilize the inspected 3D backbone, either VoteNet or PointGroup, to cluster the input point cloud into object proposals and return feature vectors for each proposal. Then, the second part of our network, the captioning module, generates proposal descriptions using these features. Figure 1 illustrates the data flow through the network.

4.1. Data

The input data for the 3D backbones differs in the two stages of our research, as the tasks are different. In the first part, where we focus only on the object detection, we build the point cloud $\mathcal{P}$ from the ScanNet [5] scans containing geometric coordinates and the RGB colors. This results in each point being six dimensional vectors. In dense captioning, additional point features are helpful for a deeper understanding of detected objects. We follow Chen et al. [4] to enhance the point cloud by back-projecting multi-view image features, which are extracted using a pre-trained ENet [10]. Hence, our final representation of the point cloud for dense captioning becomes $\mathcal{P} = (p_i, r_i, f_i) \in R^{N_p \times 134}$. Here $p_i \in R^3$ are the coordinates, $r_i \in R^3$ are the colors and $f_i \in R^{128}$ are the additional multi-view features. N_p shows the total number of points.

To train the captioning module, a data of natural language is required. We use the ScanRefer [3] dataset which introduces object descriptions to ScanNet scans. In total, there are 51,583 descriptions for 11,046 objects in 800 scenes. These descriptions contain information about object appearances, as well as spatial relations between neighbouring objects. We use GLOVE [11] embeddings to get the numerical representation of the ScanRefer vocabulary.

4.2. Detection Backbone

The baseline detection backbone is VoteNet, which uses PointNet++ [13] for feature learning and Deep Hough Voting for generating object proposals [12]. Chen et al. [4] altered the model to process the desired inputs and output object bounding boxes and feature vectors optimized for the dense captioning goal.

In this work, we use PointGroup [6], the state-of-the-art instance segmentation model for extracting object features and creating bounding boxes. One major challenge of using PointGroup as the backbone is the complexity of the model. Even though PointGroup utilizes sparse convolu-

tions for achieving more efficient convolutions, the model uses the gain in efficiency to create a larger network. The resulting model is significantly larger than the baseline backbone VoteNet. Another difference is that PointGroup uses a volumetric grid approach. The point cloud is voxelized and each feature dimension becomes a depth channel. The voxels are then convolved with the kernels in the UNets of Figure 1. This significantly increases the number of computations despite the sparse convolutions, hence making a forward pass through the network more complex. These differences cause a relatively harder hyper-parameter optimization and architecture exploration.

Since PointGroup is an instance segmentation network, it cannot be used for the object detection task out-of-the-box. We make two major modifications to it to integrate it into the captioning pipeline: calculating the bounding boxes for each instance cluster and retrieving cluster feature vectors.

First, we start with calculating bounding boxes from the predicted instance segmentation clusters. For this, we follow a naive approach to use the maximum and the minimum values in all directions and set these values as the boundaries of the bounding box for the corresponding cluster.

Second and lastly, we utilize the so called ScoreNet in the PointGroup model to get features for each cluster. For this, we modify the PointGroup model to increase the minimum number of feature channels for each object during the training. This is an essential step, as we want to optimize the representability of the features without increasing the computational complexity. As shown in Figure 1, we retrieve a 64-dimensional feature vector for each object proposal from the ScoreNet. We use a simple linear layer to double the dimension and thus get the 128-dimensional vector for each cluster similar to the baseline detection backbone.

4.3. Captioning Module

As the captioning part of our network, we use a simple GRU cell. We set the shapes of its input u_t and hidden state h_t to the shape of GLOVE embeddings, which is $\in R^{300}$. The 128-dimensional features coming out of the detection backbone are first mapped to the embedding size by a linear layer, and set as the initial GRU hidden state h_0. To keep the training of the captioning module stable and accurate, we only consider the features of good proposals that achieve a larger IoU than 0.25. At each time step t one word is generated. The recurrence is continued until the final step $T = 32$, which is the maximum sentence length we define plus the SOS (start of sentence) and EOS (end of sentence) tokens.

During training, we apply *teacher forcing*, where the ground truth object description is fed word-by-word as the GRU input at each time step. Only the hidden state is transferred to the next step. In the inference phase, the module builds the sentence by itself. At time step t, the GRU input is its output word from previous step $t-1$, making the process completely recurrent.

5. Experiments and Results

As described in Section 3, we divide the evaluation into two topics: object detection and captioning. In the following, we describe the experiment conditions as well as the results of the corresponding experiments.

5.1. Experiment Conditions

We build our model using PyTorch. Due to the complexity of 3D computer vision, we use a CUDA graphic processor as the computation power. We use ScanRefer train and validation sets for training and validating the results. For each scene, we randomly sample 40,000 points if there are more than that to keep the overall data size feasible. We train the networks over 15 epochs and choose the best performing epoch. One epoch is roughly equal to 14 runs over the 561 scenes of ScanRefer training set.

There are 3 preparation epochs in the model with PointGroup in contrast to the model with VoteNet. This is because PointGroup works optimally when its semantic prediction backbone is trained several epochs before activating the clustering module.

5.2. Object Detection

As an object detection performance metric, we use the mean average precision (mAP) thresholded by the IoU values 0.25 and 0.5. Table 1 shows the results of both backbones. We see that the PointGroup backbone outperforms the VoteNet backbone for both of the threshold values.

	mAP@IoU0.25	mAP@IoU0.5
VoteNet	49.91	32.21
PointGroup (ours)	**51.01**	**34.66**

Table 1. Object detection performance of each of the backbones.

These results show that PointGroup results in a better understanding of the objects in the scene. We hypothesize that this may lead to more representative object features, thus better captioning performance. In the next section of the results, we inspect this in detail.

5.3. Captioning

In this section, we evaluate the captioning performance of the models both quantitatively and qualitatively.

5.3.1 Quantitative Analysis

For the quantitative analysis of the captioning performance, we use BLEU [9], CIDEr [14], METEOR [2], and ROUGE [7] metrics thresholded with IoU values of the prediction and ground truth bounding boxes. This means that if the

	BLEU-4@IoU0.5	ROUGE@IoU0.5	METEOR@IoU0.5	CIDEr@IoU0.5
VoteNet + GRU	21.42	41.33	34.31	20.13
PointGroup + GRU (ours)	**22.41**	**44.25**	**34.63**	**21.38**

Table 2. Dense captioning performance of the end-to-end architectures using each of the backbones.

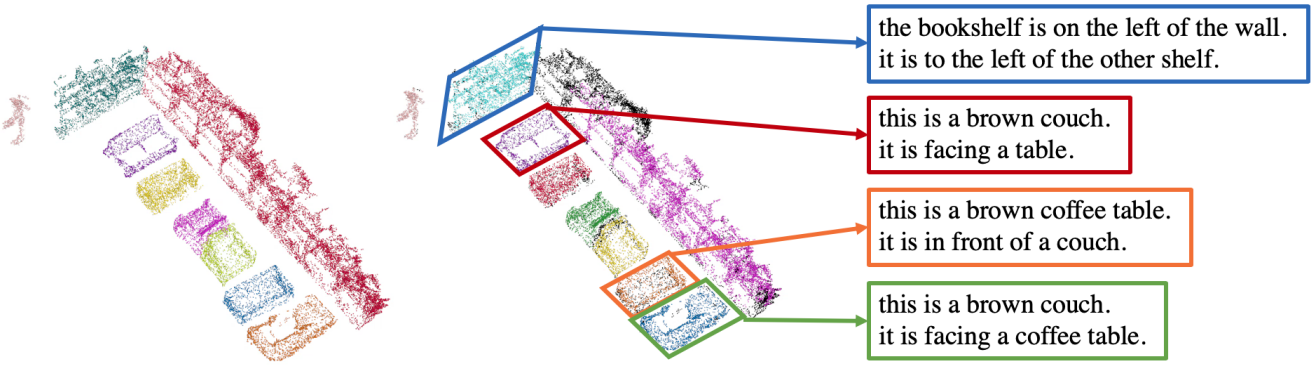

Figure 2. Sample output of the end-to-end captioning pipeline using a scene from the validation set. On the left is the ground truth instance segmentation, on the right is the instance segmentation prediction and the generated captions assigned to four of the predictions.

IoU value for a bounding box prediction is less than the threshold, the corresponding captioning metric is set to zero. After applying the threshold, we calculate the average score over all predictions.

Table 2 shows the captioning results of the experiment. Our captioning model with the PointGroup backbone outperforms the baseline model in all 4 metrics.

5.3.2 Qualitative Analysis

Looking at the qualitative performance of the the generated captions by our model, there are some key points to mention about the general performance. First of all, we observe that some objects (e.g., bed, table, chair, cabinets) are successfully described in most cases while some other objects (e.g., keyboard, mouse, telephone) are sometimes left without a caption. In the results with VoteNet backbone, this issue is seen on a larger scale. We can reason this to the distribution in the dataset. The successfully described objects have many examples, while the others are more uncommon. Also considering a single scene, these objects cover a larger area. So, despite some improvement, small and rare objects still remain as a challenge.

Another observation we make is the similarity among mistaken objects. In some cases, the model confuses refrigerators with cabinets or refrigerators with white doors. This is not very surprising, as the named objects are sometimes hard to discriminate even with the human eye.

As the final matter, when describing objects, our model can understand and use the spatial relations between the target and a few local neighbors. Figure 2 shows a sample output of the end-to-end pipeline. As seen in the predicted captions, the model successfully captures the spatial relations between neighboring couches and tables. We also observe that it almost never uses the global context of the whole scene. In example, it can say "this is a brown table. it is surrounded by chairs.", and rarely uses the phrases as "in the middle of the room" or "at the corner". Incorporating the global context of the scene into the captions remains a future task that can further improve the results.

6. Conclusion

In this work, we explored the effect of better object detection in the 3D dense captioning task. We integrated PointGroup as an object detection backbone to the captioning pipeline and compared the resulting model with the existing baseline that uses VoteNet. Our model outperformed the baseline model both in detecting objects and generating captions, which we quantitatively measured using bounding box mAP, BLEU, CIDEr, ROUGE, and METEOR metrics thresholded by IoU.

7. Future Work

Due to time and resource limitations, the PointGroup backbone has not been applied to the current state-of-the-art 3D dense captioning model Scan2Cap by Chen et al. [4]. The integration of PointGroup into the Scan2Cap model remains a future work. However, this work comparing PointGroup and VoteNet provides strong evidence that the Scan2Cap model using the PointGroup backbone will outperform the existing model.

References

[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. *16th European Conference on Computer Vision (ECCV)*, 2020. 1

[2] Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*, pages 65–72, 2005. 3

[3] Dave Zhenyu Chen, Angel X Chang, and Matthias Nießner. ScanRefer: 3D Object Localization in RGB-D Scans using Natural Language. *arXiv preprint arXiv:1912.08830*, 2019. 1, 2

[4] Dave Zhenyu Chen, Ali Gholami, Matthias Nießner, and Angel X. Chang. Scan2cap: Context-aware dense captioning in rgb-d scans, 2020. 1, 2, 4

[5] 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. 1, 2

[6] Li Jiang, Hengshuang Zhao, Shaoshuai Shi, Shu Liu, Chi-Wing Fu, and Jiaya Jia. PointGroup: Dual-Set Point Grouping for 3D Instance Segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4867–4876, 2020. 1, 2

[7] Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81, 2004. 3

[8] Baoyuan Liu, Min Wang, Hassan Foroosh, Marshall Tappen, and Marianna Pensky. Sparse convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 806–814, 2015. 1

[9] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002. 3

[10] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016. 2

[11] 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. 2

[12] 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 International Conference on Computer Vision*, pages 9277–9286, 2019. 1, 2

[13] Charles R Qi, Li Yi, Hao Su, and Leonidas J Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. *arXiv preprint arXiv:1706.02413*, 2017. 2

[14] Ramakrishna Vedantam, Zitnick C Lawrence, and Devi Parikh. Cider: Consensus-based image description evaluation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4566–4575, 2015. 3