Statistical Language Models for Scene Understanding

Avinash Arutla, Laleth Indirani Nehrukumar, Sheela Hansda

Northeastern University

Boston, MA

arutla.a@northeastern.edu, indiraninehrukumar.l@northeastern.edu, hansda.s@northeastern.edu

December 2023

1. Objectives and significance

A significant problem in robotics is understanding sceneries, which calls for robots to be able to semantically comprehend spaces and their contents in addition to being able to navigate and localize in a variety of situations. For example, if a robot is told "go fetch a spoon," it should be able to accomplish the task because it is naturally familiar with the common items present in a kitchen. An underutilized strategy to deal with this is to employ language models, which, when trained on substantial amounts of text data, collect semantic information. As an illustration, a language model might come to correlate "Bathrooms contain..." with "toilets" rather than "stoves," exhibiting a degree of common sense that is essential for comprehending the setting. Furthermore, language's ability to convey commonsense questions, even those incorporating novel concepts is crucial for spatial perception since deployed robots can come across unexpected items and need to be able to conclude unknown object kinds.

This project aims to develop an image classification model, based on objects detected in the scene. We took motivation from the paper ‘Extracting Zero-shot Common Sense from Large Language Models for Robot 3D Scene Understanding’ [21], which utilizes a large language model and zero-shot learning approach to generalize to arbitrary room and object labels, including those that it has not seen during training. Similarly, this project aims to utilize Language Models employing the “Bag of Words” technique to classify rooms based on the objects detected within the scenes captured in images. Though we took a similar approach as [21] our approach is less complex and easy to deploy.

The significance of this approach is that it provides the model with a semantic understanding of an environment and the entities within it. To explain this, we can revisit the above example “go fetch a spoon,” the model should understand that a spoon is likely to be in

the kitchen rather than the bedroom. Another significance of this approach might have is to reduce the training time or complexity of Computer Vision models by limiting the training data to only lower-level classes i.e., specific objects. In the popular ImageNet challenges new and efficient models are being introduced every year for detection or classification tasks. These models almost always are trained on data that have labels on multiple hierarchical levels. For example, a class car could have sub-classes called Fiet, BMW, Audi, etc. This always makes the model more complex. In our approach, we hope to create a new method for classification models that use the lower-level classes to classify classes on higher levels.

2. Background

2.1. Language and Robotics

According to [19], research on using language models for semantic reasoning has grown significantly. Most of the recent research has concentrated on tasks such as fact completion [15] [16], question answering [17], and logical reasoning [18]. On the other hand, previous robotics research has mostly focused on the application of language for activities such as providing planning and execution instructions to robots. Natural language processing techniques have not been widely used for robotics tasks involving scene interpretation or room classification. Prior methods, which frequently depended on the most frequently occurring objects, have been used to categorize rooms based on identified objects, such as employing Bayesian probabilistic frameworks. These techniques, however, have trouble generalizing to new spaces and objects and frequently need unique statistical information for every new instance. In this work, we investigate the possibility of language models as intuitive tools to improve a robot's perception of its surroundings, overcome these constraints, and capitalize on the benefits of language.

2.2. Natural Language Processing

Implementing Natural Language Processing (NLP) systems involves a combination of linguistic knowledge, machine learning techniques, and advanced algorithms. At the core of many NLP applications are statistical models and machine learning algorithms that can learn patterns and relationships within large datasets of human language. Supervised learning is commonly used, where models are trained on labeled datasets that contain examples of input text and their corresponding desired outputs. This training process allows the model to learn

the inherent structures and patterns in language, enabling it to make predictions or classifications on new, unseen data.

Recent advancements in NLP have been driven by the development of deep learning models, especially transformer architectures like BERT (Bidirectional Encoder Representations from Transformers) and GPT (Generative Pre-trained Transformer). Pre-trained language models, trained on massive amounts of text data, have demonstrated remarkable capabilities in understanding context and generating coherent text. Fine-tuning these pre-trained models for specific tasks or domains further enhances their performance.

2.2.1 Bag of Words

The approach is simple and flexible and can be used in many ways for extracting features from documents. A bag-of-words is a representation of text that describes the occurrence of words within a document. The method mainly incorporates two concepts: 1. A vocabulary of known words. 2. A measure (count) of the presence of known words.

It is called a “bag” of words because any information about the order or structure of words in the document is discarded. The model is only concerned with whether known words occur in the document, not where in the document.

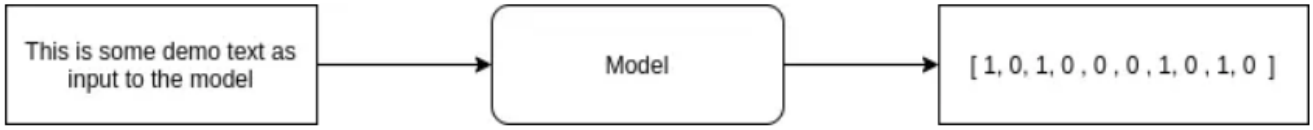


Fig 1.0 - How a Bag-of-Words model work

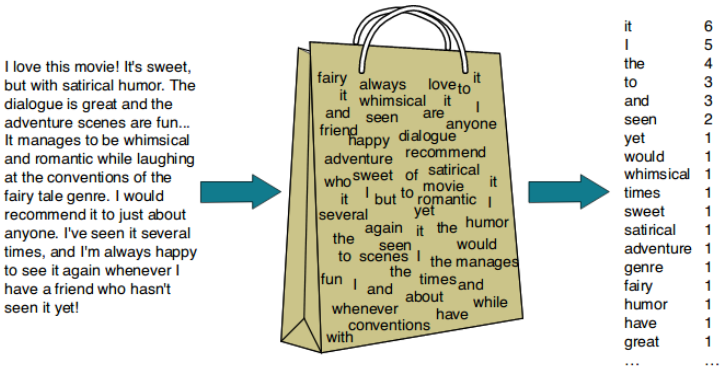


Fig 2.0 - Simple representation of Bag of Words. The bag represents the vocabulary of new words the model encounters. The right-most column is the BoW representation that contains the count of each word in vocabulary that was encountered in the document.

The Bag-of-Words approach is implemented by following these steps:

Step 1: Tokenization -

The first step involves breaking down the text into individual words or tokens. This process is called tokenization. Punctuation and stop words (simple words like "the," "is," etc.) are often removed during this stage.

Step 2: Vocabulary Construction -

A vocabulary is created by compiling a list of unique words from the entire corpus of documents. Each unique word in the vocabulary is assigned a unique index or position.

Step 3: Vectorization -

For each document in the corpus, a vector is created. The length of the vector is equal to the size of the vocabulary, and each element represents the count (or sometimes binary occurrence) of the corresponding word in the document. If a document contains the word represented at index i in the vocabulary, the i-th element of the vector is incremented.

Step 4: Sparse Representation -

Since most documents only use a small subset of the entire vocabulary, the resulting vectors are often sparse, meaning that most elements are zero. This sparse representation is efficient for storage and computation.

Step 5: Final Feature Representation-

The vectors for all documents form a matrix, where each row corresponds to a document, and each column corresponds to a word in the vocabulary. This matrix serves as the feature representation for the entire corpus.

2.3 Previous Works

Previous papers like Tellex et al. [21], Sharma et al. [18], and more [10, 19, 8, 6, 12] have leveraged language for communicating goals or instructions to a robot to plan around or execute. Past works like Chaves et al. [3] instead use an explicit Bayesian probabilistic framework for determining room categories based on detected objects. However, these methods remain hard to generalize to new room and object categories, with Chaves et al. [3] only considering five room labels. In the paper 'Extracting Zero-shot Common Sense from Large Language Models for Robot 3D Scene Understanding' [21], Large Language Models are used as common-sense mechanisms for robot scene understanding. It uses a heuristic method for picking out a small number of semantically informative present within a room, constructing a query string with those object labels acting as a description of the room, and passing the string through a language model to infer the room's label. However, using an LLM [18,21] makes the model more complex in both cost and time in training and computation. Also, the accuracies of these models are low (average: 65%). Our approach, however, leverages statistical language model for a change in representation of the scene from pixels to a simple sparse vector which is used to train classifiers without using any Deep Neural Networks. This makes our approach simpler and more accurate (average: 83%) than another similar approaches [18,21].

3. Methods

3.1 Dataset

In this project, we used the publicly available LSUN (Large Scale Understanding) [22]. The dataset is a collection of images designed for scene understanding and scene parsing tasks in computer vision. It comprises millions of images that depict various indoor and outdoor scenes, offering a diverse range of environments, including bedrooms, living rooms, kitchens, classrooms, etc. The dataset includes 59 million images for 20 object categories selected from PASCALVOC2012 and 10 million images for 10 scene categories selected from the SUN database.

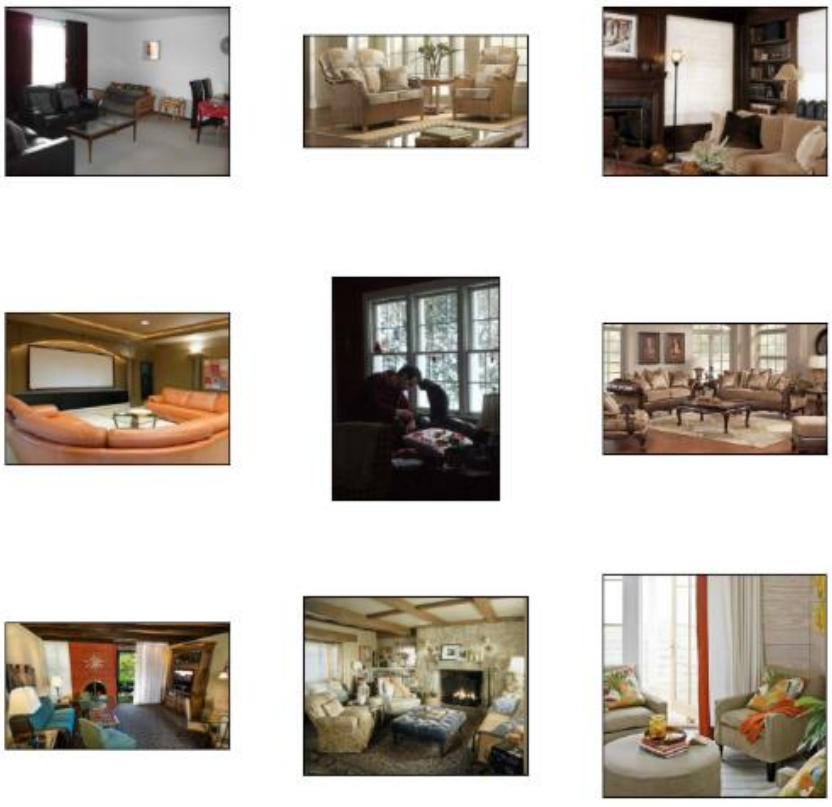


Fig 3.0 Sample dataset for living room.

In this project we used 20,000 images per class for five classes which includes bedroom, classroom, living room, dining room and kitchen for implementation. The dataset also provides a validation set of 300 images per class which was used for evaluating the classifiers.

3.2 Methodology

In this project we took an approach merging object detection and Bag-of-Words (BoW) representation to perform image classification using language models. Initially, the code employs a pre-trained object detection model to detect objects within images, establishing a vocabulary that stores unique detected objects across all images. It then converts the detected objects into BoW representations, creating numerical vectors based on the vocabulary. These BoW vectors serve as features, encoding the presence and frequency of objects in the images. Several classifiers are trained using these BoW representations and associated class labels, enabling the prediction of image labels based on the trained classifiers. For evaluation, BoW representations for the validation images are computed and used on the trained classifiers to predict the labels.

3.2.1 Object Detection Model

We use YOLO (You Only Look Once) [23] object detection model that excels in real-time object detection tasks. YOLO models are designed to detect and localize objects within images or video frames, while also classifying those objects into predefined categories. In this project, we use the pre-trained YOLOv8 model to get a list of objects present in images. The model is capable of identifying objects belonging to 80 classes. This is used as the maximum length (80) of vocabulary in BoW representation. This ensures that the object vectors or BoW representations remain consistent across all classes and will be easier to train classifiers without modifying the representations.

3.2.2 Pseudo Code for computing Bag of Words Representation

Class RoomClassifierBoW:

Initialize:

Initialize object detection model

Initialize empty vocab, word_counts, and obj_list variables

Set maximum vocabulary length (max_len_vocab)

Method fit(paths, label):

Initialize empty objects_detected and ground_truth lists

For each image path in paths:

Detect objects in the image using object detection model

For each detected object:

Add object to obj_list and update vocab and word_counts

Add obj_list to objects_detected list

Add label to ground_truth list

Initialize empty representation list

For each obj_list in objects_detected:

Initialize empty obj_vec of length max_len_vocab

For each object in obj_list:

If object exists in vocab:

Increment corresponding index in obj_vec

Add obj_vec to representation list

Return representation, ground_truth

Function main():

Initialize image paths and labels

Split data into paths for each class

Initialize RoomClassifierBoW instance

For each class and its paths:

Generate BoW representation, vocabulary and true labels using the class instance

Concatenate representations and labels

3.3 Experiments

The experiments conducted were mainly composed of training and evaluating classifier models based on BoW representations. The following six classification algorithms are chosen for the experiments.

1. Naïve Bayes
2. Logistic Regression
3. Decision Tree
4. Random Forest
5. K-Nearest Neighbours (KNN)
6. Support Vector Classification

Each model is trained individually and evaluated using the following metrics.

1. Accuracy of the model – Number of correct prediction / Total number of predictions
2. Precision – Calculated for each class
3. Recall – Calculated for each class
4. F1 Score – Calculated for each class
5. ROC AUC score

4. Results

4.1. Bag of Words Representation

The first step in our model involves identifying objects present in the images and using them to create a vocabulary of objects and creating a BoW representation for each image. The following table represents the objects that are detected in the training set which was used for creating the vocabulary. We identified that the object detection model sometimes detects unlikely objects (represented in red in Table 1.0) for example all the training images consist only of indoor scenes, but the model may sometime return bus, elephant in some cases and sometimes returns an empty list. We hope to try other object detection models to overcome this problem.

Table 1.0 - Final vocabulary that was constructed based on objects detected in training images. Index represents the index of the object represented in BoW representation array.

Word (Object)	Index	Word (Object)	Index
couch	0	bench	1
bed	2	microwave	3
chair	4	bottle	5
backpack	6	person	7
potted plant	8	vase	9
toilet	10	sink	11
suitcase	12	dog	13
tv	14	cake	15
sheep	16	bird	17
cup	18	dining table	19
bowl	20	motorcycle	21
refrigerator	22	train	23
clock	24	donut	25
car	26	cat	27
book	28	umbrella	29
laptop	30	pizza	31
teddy bear	32	oven	33
handbag	34	horse	35
keyboard	36	cell phone	37
remote	38	wine glass	39
mouse	40	skateboard	41
sandwich	42	tie	43
fire hydrant	44	hot dog	45
sports ball	46	apple	47
banana	48	elephant	49
airplane	50	orange	51
frisbee	52	bicycle	53
giraffe	54	toothbrush	55
truck	56	traffic light	57
boat	58	knife	59
spoon	60	carrot	61
baseball bat	62	toaster	63
zebra	64	kite	65
cow	66	broccoli	67
parking meter	68	bus	69
surfboard	70	scissors	71
tennis racket	72	baseball glove	73
fork	74	skis	75
stop sign	76	snowboard	77

Table 2.0 - Example of training images, their BoW representation of class "Bedroom".

Image	BoW Representation
	[0, 0, **1**, 0, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]
	[0, 0, **0**, 0, 0, 0, 0, 2, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

As observed from Table 2.0, every bedroom is different and contains different entities. And due to bias or nature of training images used for the object detection model not all the objects of same class can be detected. For example, the second image in the table contains a bunk bed which is not detected by the model and hence the index for **bed (2)** in the corresponding BoW representation remains 0, where the index for **person (7)** has the value 2. It was observed that, for each image the object detection model returns at most 5-6 objects.

4.2. Classifier Evaluation Results

The first metric we chose to evaluate our approach is to calculate the accuracy of the predicted labels. For comparison the following tables provide top 1% accuracy of [18] and other models presented in 2016 LSUN classification challenge.

Table 3.0 - Test-split accuracies for all language-only approaches. Highest test split accuracies are bolded.[18]

	Baselines		Zero-shot		Embedding-based		Structured-language			
					RoBERTa		With Position		No Position	
	Statistical	GraphSage	GT	Proxy	GT	Proxy	GT	Proxy	GT	Proxy
nyuClass	57.7% (50.6%)	64.2%	52.2% (53.6%)	27.2% (28.2%)	65.5%	57.6%	**69.1%**	68.7%	67.3%	67.7%
mpcat40	44.7% (46.9%)	59.1%	48.9% (50.1%)	27.5% (27.3%)	63.9%	58.5%	**68.3%**	67.7%	63.9%	63.7%

Table 4.0 - Test-split room- and image-wise accuracies for all language and vision approaches. Highest test split accuracies are bolded.[18]

	Zero-shot							Baseline		Fine-tuning		
	CLIP	BLIP VQA			BLIP-2 Captioner			ResNet-50		BLIP VQA		
	None	None	nyuClass	mpcat40	None	nyuClass	mpcat40	From Pretrained	From Scratch	None	nyuClass	mpcat40
Room-wise	36.5% (32.7%)	37.8% (36.2%)	37.0% (36.6%)	37.1% (37.5%)	47.5% (45.7%)	47.9% (45.7%)	48.0% (46.2%)	51.1%	26.2%	53.2%	**68.6%**	65.3%
Image-wise	26.5% (25.9%)	30.1% (28.8%)	35.6% (34.9%)	34.4% (34.3%)	40.1% (39.3%)	45.0% (43.1%)	44.5% (43.0%)	36.8%	22.6%	47.0%	**67.9%**	64.1%

Table 5.0 - Classification accuracy of different teams at the LSUN challenge 2016. [24]

Rank	Team	Year	Top1 Accuracy
1	SIAT_MMLAB	2016	**91.6%**
2	SJTU-ReadSense	2016	90.4%
3	TEG Rangers	2016	88.7%
4	ds-cube	2016	83.0%
1	Google	2015	91.2%

Table 6.0 - Accuracy of classifiers trained in this project.

Classifier	Naïve Bayes	Logistic Regression	Decision Tree	Random Forest	KNN	SVC
Accuracy %	81	82.87	81.53	83	83.13	82.73

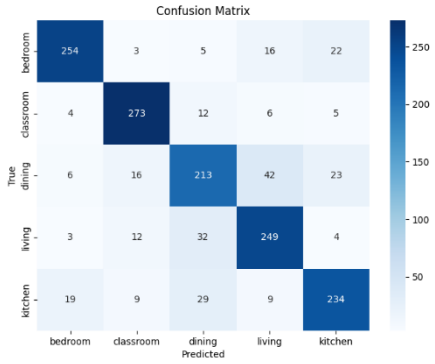
As observed from Table 6.0, our approach has considerable accuracy when compared with the top models presented at the LSUN Classification Challenge 2016 without using complex neural networks. We deduced that we could further improve the accuracy, by using a more effective and accurate object detection model to create an improved BoW representation. The following table shows the other metrics we used for evaluation such as precision, recall, F1 score and ROC AUC score.

Table 7.0 - Evaluation metrics of all classifiers. Field marked with '-' are scores for the model.

Classifier	***Class***	*Decision Tree*	*KNN*	*Logistic Regression*	*Naïve Bayes*	*Random Forest*	*SVC*
Precision	Bedroom	0.89	0.90	0.92	0.85	0.89	0.89
	Classroom	0.87	0.87	0.91	0.88	0.89	0.91
	Dining	0.73	0.79	0.79	0.73	0.78	0.79
	Living	0.77	0.75	0.73	0.81	0.78	0.73
	Kitchen	0.81	0.84	0.82	0.78	0.80	0.82
Recall	Bedroom	0.85	0.84	0.84	0.86	0.85	0.84
	Classroom	0.91	0.92	0.88	0.88	0.91	0.90
	Dining	0.71	0.74	0.74	0.75	0.72	0.75
	Living	0.83	0.85	0.88	0.77	0.86	0.87
	Kitchen	0.78	0.80	0.81	0.79	0.81	0.78
F-1 Score	Bedroom	0.87	0.87	0.88	0.86	0.87	0.88
	Classroom	0.89	0.89	0.89	0.88	0.90	0.91
	Dining	0.72	0.77	0.77	0.74	0.75	0.77
	Living	0.80	0.80	0.79	0.79	0.82	0.79
	Kitchen	0.79	0.79	0.81	0.79	0.81	0.79
ROC AUC Score	-	0.87	0.89	0.86	0.88	0.88	0.87

4.3. Confusion Matrix

This section provides an in-depth look into the performance of the model by looking at the confusion matrices for each classifier



.

Fig 4.0 - Confusion Matrix – Decision Tree

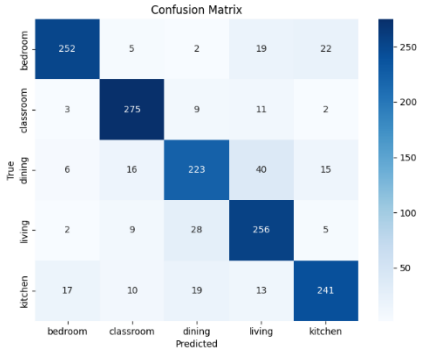


Fig 5.0 - Confusion Matrix – KNN

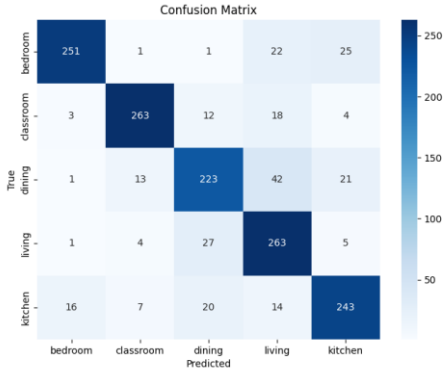


Fig 6.0 - Confusion Matrix – Logistic Regression

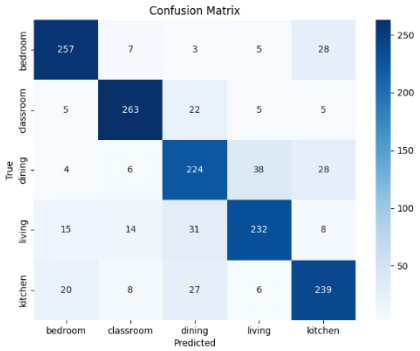


Fig 7.0 - Confusion Matrix – Naïve Bayes

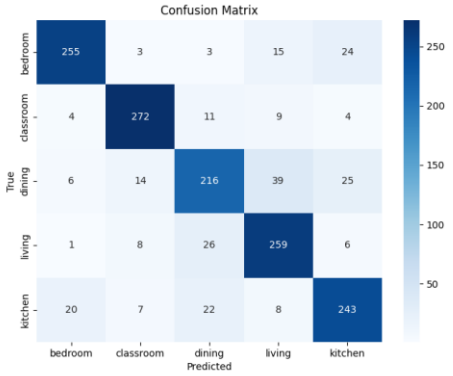


Fig 8.0 - Confusion Matrix – Random Forest

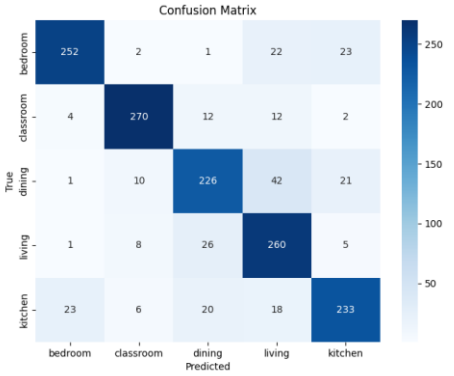


Fig 9.0 - Confusion Matrix – SVC

From these confusion matrices we can see that classes that are likely to contain unique objects such as bedroom, classroom have high number of true positives and classes that share similar objects such as kitchen, dining room have high number of false positives. In simpler terms we can see that in all classifiers a considerable number of dining rooms are classified as kitchens and vice versa. This is a behavior we expected the model to have since the classification is based on the objects identified. This shows the possibility of the model to learn and classify based on reasoning rather than just training based on image features. We could also see that a similar number of bedrooms are classified as kitchen and living rooms. This happens because the object detection model sometimes returns false objects that could skew the BoW representation. This can be rectified by using a more accurate object detection model or fine tune the model to detect specific objects in the training set that could provide more information regarding the scene and avoid unnecessary data. We can also see that the classifiers are unlikely to predict false positives for completely different classes. For example, we can see that across all classifiers we see a smaller number of classrooms and dining rooms are classified as bedrooms.

5. Conclusions

Our primary objective is to create a resilient algorithm to precisely classify 2D images into higher level classes like using low level classes. Our strategy was to adopt a comparatively simpler, yet effective method using representation vectors to classify images. Bag of Words as an algorithm is actively used in computer vision tasks for various applications, but most of them are modified to create vocabulary for image features such as pixels. But we intended to use the algorithm as a statistical language model and use it for computer vision tasks such as image classification. As outlined earlier, we employed multiple classifiers for result comparison, revealing that their accuracies are similar across them. From the confusion matrix we can see that the model could understand basic concepts such as a if a room has a bed, it can be classified as bedroom and a classroom cannot be a bedroom. Some future work that we hope to do is to try a different object recognition model and see if it improves the BoW representation and can the accuracy be improved. We also plan to analyze in-depth what the models could learn and try to build a model that could serve as an alternate to full on image processing model that could understand the scene and can be used for streamline tasks such as classification and detection. Further study by comparing the performance of this model and state of art image classification model need to be done to analyze whether this approach can improve the field.

6. Individual Tasks

Sheela Hansda was responsible for data collection and annotation. The task included gathering, curating, and annotating a dataset of indoor scenes in various environments, along with associated textual descriptions. She also performed the preprocessing and data cleaning, to ensure consistency in annotations.

Laleth Indirani Nehrukumar was responsible for designing the architecture of the model. He created the program for converting images into representation vector using Bag of Words and implemented the classifier models that effectively identified room classifications from the image datasets.

Avinash Arutla developed metrics and evaluation procedures to assess the model's performance. This included tasks like calculating accuracy, precision, and recall for scene understanding.

Apart from these specific tasks, all the team members were involved in the end-to-end understanding of the project, i.e., object recognition, room classification, or inferring spatial relationships. All team members were responsible for documenting the project's progress, methodologies, and results. This included creating reports, user guides, and potentially presenting findings.

7. References

[1] S. Bowman, N. Atanasov, K. Daniilidis, and G. Pappas, “Probabilistic data association for semantic SLAM,” in IEEE Intl. Conf. on Robotics and Automation (ICRA), 2017, pp. 1722–1729.

[2] N. Hughes, Y. Chang, and L. Carlone, “Hydra: a real-time spatial perception engine for 3D scene graph construction and optimization,” in Robotics: Science and Systems (RSS), 2022, (pdf).

[3] A. Rosinol, A. Violette, M. Abate, N. Hughes, Y. Chang, J. Shi,A. Gupta, and L. Carlone, “Kimera: from SLAM to spatial perception with 3D dynamic scene graphs,” Intl. J. of Robotics Research, vol. 40, no. 12–14, pp. 1510–1546, 2021, arXiv preprint arXiv: 2101.06894, (pdf).

[4] J. Dong, X. Fei, and S. Soatto, “Visual-Inertial-Semantic scene representation for 3D object detection,” in IEEE Conf. on Computer Vision and Pattern Recognition (CVPR), 2017.

[5] L. Nicholson, M. Milford, and N. Sünderhauf, “QuadricSLAM: Dual quadrics from object detections as landmarks in object-oriented SLAM,” IEEE Robotics and Automation Letters, vol. 4, pp. 1–8, 2018.

[6] K. Ok, K. Liu, and N. Roy, “Hierarchical object map estimation for efficient and robust navigation,” in 2021 IEEE International Conference on Robotics and Automation (ICRA), 2021, pp. 1132–1139.

[7] J. Behley, M. Garbade, A. Milioto, J. Quenzel, S. Behnke, C. Stachniss, and J. Gall, “SemanticKITTI: A Dataset for Semantic Scene Understanding of LiDAR (Light Detection and Ranging) Sequences,” in Intl. Conf. on Computer Vision (ICCV), 2019.

[8] M. Grinvald, F. Furrer, T. Novkovic, J. J. Chung, C. Cadena, R. Siegwart, and J. Nieto, “Volumetric Instance-Aware Semantic Mapping and 3D Object Discovery,” IEEE Robotics and Automation Letters, vol. 4, no. 3, pp. 3037–3044, 2019.

[9] R. Rosu, J. Quenzel, and S. Behnke, “Semi-supervised semantic mapping through label propagation with semantic texture meshes,” Intl. J. of Computer Vision, 06 2019.

[10] C. Li, H. Xiao, K. Tateno, F. Tombari, N. Navab, and G. D. Hager, “Incremental scene understanding on dense SLAM,” in IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2016, pp. 574–581.

[11] J. McCormac, R. Clark, M. Bloesch, A. Davison, and S. Leutenegger,
“Fusion++: Volumetric object-level SLAM,” in Intl. Conf. on 3D Vision (3DV), 2018, pp. 32–41.

[12] J.-R. Ruiz-Sarmiento, C. Galindo, and J. Gonzalez-Jimenez, “Building multiversal semantic maps for mobile robot operation,” Knowledge-Based Systems, vol. 119, pp. 257–272, 2017.

[13] C. Galindo, A. Saffiotti, S. Coradeschi, P. Buschka, J. Fernández- Madrigal, and J. González, “Multi-hierarchical semantic maps for mobile robotics,” in IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems (IROS), 2005, pp. 3492–3497.

[14] F. Petroni, T. Rocktäschel, P. Lewis, A. Bakhtin, Y. Wu, A. H. Miller, and S. Riedel, “Language models as knowledge bases?” 2019.

[15] F. Petroni, P. Lewis, A. Piktus, T. Rocktäschel, Y. Wu, A. H. Miller, and S. Riedel, “How context affects language models’ factual predictions,” in Automated Knowledge Base Construction, 2020.

[16] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, “Qa-gnn: Reasoning with language models and knowledge graphs for question answering,” 2021.
[Online]. Available: https://arxiv.org/abs/2104.06378

[17] A. Creswell, M. Shanahan, and I. Higgins, “Selection-inference: Exploiting large language models for interpretable logical reasoning,” 2022.
[Online]. Available: https://arxiv.org/abs/2205.09712

[18] W. Chen, S. Hu, R. Talak, L. Carlone,” Leveraging Large Language Models for Robot 3D Scene Understanding”. [Online]. Available: https://doi.org/10.48550/arXiv.2209.05629.

[19] https://arxiv.org/abs/1709.06158v1.

[20] https://matterport.com/partners/facebook.

[21] W. Chen, S. Hu, R. Talak, L. Carlone,” Extracting Zero-shot Common Sense from Large Language Models for Robot 3D Scene Understanding”. [Online]. Available: https://arxiv.org/abs/2206.04585

[22] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, & Jianxiong Xiao. (2016). LSUN: Construction of a Large-scale Image Dataset using Deep Learning with Humans in the Loop. [Online]. Available: https://doi.org/10.48550/arXiv.1506.03365.

[23] https://github.com/ultralytics/ultralytics.

[24] Wang, L., Guo, S., Huang, W., Xiong, Y., & Qiao, Y. (2017). Knowledge Guided Disambiguation for Large-Scale Scene Classification With Multi-Resolution CNNs. IEEE Transactions on Image Processing, 26(4), 2055–2068. [Online]. Available: https://doi.org/10.48550/arXiv.1610.01119.