APPROVAL SHEET

Title of Thesis: SPEECH VS TEXTUAL DATA FOR GROUNDED LANGUAGE LEARNING

Name of Candidate: Rishabh Sachdeva
Master of Science
Computer Science, 2020

Thesis and Abstract Approved: ________________________________
Dr. Cynthia Matuszek
Assistant Professor
Computer Science and Electrical Engineering

Date Approved: ________________

Curriculum Vitae

Name: Rishabh Sachdeva

Degree and date to be conferred: Masters in Computer Science, May 2020

Collegiate institutions attended:

University of Maryland, Baltimore County,
MS in Computer Science, May 2020

S.G.S. Institute of technology and Science, Indore, India
Bachelor of Engineering, May 2015

Field of Study: Computer Science

Professional positions held:

1. Graduate Teaching Assistant, UMBC (Fall 2019 – Spring 2020)
2. Technical Intern, AT&T Research labs,
   San Ramon, CA, USA (May 2019 – July 2019)
3. Associate Software Engineer, Nuance Communications,
   Pune, India (Nov 2017 – June 2018)
4. Software Developer, Amdocs LLP,
   Pune, India (July 2015 – Nov 2017)

ABSTRACT

Title of dissertation:	SPEECH VS TEXTUAL DATA FOR GROUNDED LANGUAGE LEARNING
	Rishabh Sachdeva, Master of Science, 2020
Dissertation directed by:	Dr. Cynthia Matuszek Department of Computer Science

In this thesis, we describe the compatibility of audio data with the Grounded Learning system adopted from text-only systems. My thesis work lies in the junction of NLP, Speech, and Robotics. First, we conduct in-person user studies to collect audio descriptions of household objects in a controlled environment. In this work, we use category-based Grounded Learning System [9]. This system learns the meaning of words used in crowd-sourced descriptions by grounding them in the physical representation of the objects that the workers describe. We compare the performance of the category-based model with the in-lab collected speech data and crowd-sourced text data. We find that the system can learn color, object, and shape words with comparable performance. To expand the analysis, we collect natural language descriptions both in textual as well as speech format for various kitchen, office, and household items using the crowd-sourced platform. Our work involves an in-depth comparative and qualitative analysis of crowd-sourced speech and textual data. We compare the F1-scores generated for learned tokens using the category-based model for speech and text data collected using AMT. We find that the final averaged F1

scores of all the individual tokens learned are comparable in the two cases with no significant difference.

SPEECH VS TEXTUAL DATA
FOR GROUNDED LANGUAGE LEARNING

by

Rishabh Sachdeva

Dissertation submitted to the Faculty of the Graduate School of the
University of Maryland, Baltimore County in partial fulfillment
of the requirements for the degree of
Master of Science
2020

Advisory Committee:
Dr. Cynthia Matuszek, Chair/Advisor
Dr. Francis Ferraro
Dr. Don Engel

© Copyright by
Rishabh Sachdeva
2020

Acknowledgments

I owe my gratitude to all the people who helped me in the journey and made this thesis possible. First, I would like to give my deepest thanks to my advisor, Dr. Cynthia Matuszek. I would like to thank her for all the time and effort she put in to guide me and for always making sure that I am on the right track. I highly value all the advice and technical insights she has provided to help me reach this point. I want to extend my appreciation to the rest of my committee - Dr. Don Engel and Dr. Frank Ferraro.

I want to extend my sincere thanks to John Winder. John's insights and attention have helped me grow and shape the quality of my writing and thought process, which has been especially crucial in these past few months dedicated to the completion of this thesis. I would especially like to thank Nisha Pillai for patiently answering my endless questions and helping me with her technical expertise.

Additionally, I would like to thank Caroline Kery for working with me last year and helping me to lay the foundation of my thesis, although the duration of our work together was short but was full of learnings. I would like to sincerely extend my thanks to Pat Jenkins, Luke Richards, Padraig Higgins, without whom this work could not have been completed. I would also like to thank all the IRAL members; it has been my privilege working alongside you. Finally, I would like to thank my family and friends for their endless support.

Table of Contents

List of Figures

List of Abbreviations

AMT	Amazon Mechanical Turk
CB-GLS	Category-based Grounded Learning System
CF-GLS	Category-free Grounded Learning System
GLD	Grounded Learning dataset (pronounced as GOLD)

Chapter 1

Introduction

We are privileged to be living in a time where technology is present almost everywhere to assist us and make our lives easier. Due to technological advancements, robots today are more productive, compact, and easy to use. With the widespread use of products like Amazon Echo, Siri, and Roombas, robots are getting more and more common in households. The United States is currently witnessing a rapid increase in the elderly population. Robotics can play a vital role when it comes to physical and cognitive assistance to the elderly and can help them to lead a comfortable life [6].

Robots must communicate and understand the language, and should know how to perceive the environment. Grounded Language Acquisition is a concept to bind the natural language with environmental surroundings. Grounded Language Learning is concerned with learning the meaning of language as it applies to the physical world [7]. Matuszek et al. [5] came up with a joint model of language and perception for grounded attribute learning, expanded in [9], and [10]. They design and train Machine Learning models using image summarization text data. It is necessary to introduce spoken language in training to make robots ready for the real world. As robots become more capable and ubiquitous, there is an increasing need for non-specialists to interact with and control them [7], and spoken natural language is a most flexible and intuitive way to communicate. The goal of our work is to analyze how speech description differs from written ones and learn the compatibility of text-only grounded learning systems with speech data.

Notable contributions of our work:

1. Introduce a robust multi-model Grounded Language dataset with image summarizations in speech and text formats.

2. Comparative data analysis of speech and text corpus.

3. Adopt text-only grounded learning systems and determine compatibility with speech data.

Chapter 2

Related work

In this section, we discuss the literature associated with Grounded Language Learning and the corresponding datasets.

2.1 Grounded Language Learning Datasets

There exist a variety of benchmark datasets to facilitate language and visual tasks [17, 18]. Grounded language learning requires multimodal datasets-language and visual, for development and evaluation of computational models. Sometimes, the dataset is created to tackle a specific task that research attempts to achieve. This usually leads to narrow applications. Notwithstanding such challenges, several datasets are crafted for Grounded Language Learning. In 2013, Krishnamurthy and Kollar [57] presented a SCENE dataset consisting of segmented images of indoor environments containing several ordinary objects and collected language descriptions via AMT, where each image acts as an environment, and each boundary segment as an entity. They introduced logical semantics with a perception model for grounded learning that learns to map natural language statements to their referents in a physical environment. In 2014, Lin et al. [19] introduced widely known Microsoft COCO: Common Objects in Context, an image dataset depicting everyday scenes of familiar objects in their natural context. It contained a total of 2.5 million labeled instances in 328k images and was traditionally designed to focus on vision-related problems like object recognition and scene understanding. In 2017, Shekhar et al. [25], came up with FOIL-COCO, which is an extension to MS COCO that introduced incorrect or foiled captions with the only difference of a single mistake or foil-word to the original ones. Their work showed how the Language and Vision models could fall into traps with such data and emphasized on a fine-grained understanding of the relationship between image and language. In the same year, Chrupala et al. [26] extended MS-COCO by adding generated synthetic spoken captions using Google Text-to-Speech and presented a visually grounded model of speech perception that projects spoken language and visual features to joint semantic space. However, speaker variability is limited, as only one voice was used for speech synthesis. William et al. [20] augmented the MS-COCO and added speech captions using Voxygen text-to-speech to images and came up with the Speech-COCO dataset with more than 600k spoken captions paired with images. Extensions to these datasets in other languages, like Japanese, is also introduced in [29]. Our dataset differs from them in a way how speech data is collected. In our work, we do not synthesize but collect actual audio descriptions of an object in the form of wav files using AMT. Flickr8k [21] and Flickr30k [22] datasets contains approximately 8,000 and 30,000 images from Flickr, respectively. Both the datasets contain five descriptions per image collected using AMT. Harwath and Glass [23] augmented the Flikr8k dataset by collecting 40,000 spoken captions using AMT. Krishna et al. [24] presented a visual genome dataset to achieve success in cognitive tasks, containing textual image descriptions, objects, attributes, relationships, and question-answer pairs. In total, it contains over 108K images where each image has an average of 35 objects, 26 attributes, and 21 pairwise relationships between objects. In [34], Zellars et al. authors proposed novel adversarial filtering to construct SWAG, a dataset with 113 multiple-choice questions. Their idea was to build a large-scale adversarial dataset by oversampling the potential counterfactuals using language models. Gaspers et al. described a German multimodal corpus containing parallel data from multiple speakers, including speech, visual, and body posture data. In [28], Bisk et al. discuss problem-solution sequence (PSS) data, containing a sequence of frames that the robot sees while fulfilling a goal. They focused on understanding the relationship between natural language and complex actions and goals or from sequences of actions to natural language. Most of the work we discussed above utilized crowd-sourced platforms like AMT for data collection. AMT was created is 2015, and today is a prominent crowdsourcing platform that brings tens of thousands of people together to accomplish tasks [30]. Goodman et al. investigated the differences between AMT participants and traditional samples on several dimensions. Even though. They found that AMT participants generally produced reliable results; they recommended using screening procedures to maintain quality [31]. In [32], Lane et al. presented the tools for collecting crowd-sourced speech corpora. The authors presented two different ways for data collection, first, using the mobile application, and second, using AMT by integrating external JAVA applet. In our work, we use AMT to collect speech descriptions for objects; however, modify the internal AMT image summarization task by including recording features.

2.2 Grounded Language Acquisition

Grounded Language Acquisition, the task of mapping the natural language to a representation in the physical world [7], has attracted tremendous interest in recent times. Mooney [2] in 2008, discussed this integrative AI problem of learning the connection between language and perception. Some of the pioneering work in this area include ([1], [5], [59]).

Precisely, Matuszek et al. [5] in 2012 came up with the joint model of Language and Perception for Grounded Attribute Learning, which is built on the existing work of probabilistic combinatorial categorical grammar for semantic parsing [3], [4] and visual attribute classification using depth kernel descriptors [70]. The work was further extended by her in joint work with Pillai and others [44]. The key idea was to incorporate the advantages of interactive labeling that uses active learning to ask annotations from the human subject. [8], [9] focus on obtaining negative examples of natural language annotations. Using semantic similarity, [10] described an unsupervised system that learns visual classifiers associated with the words from the corpus of perceptual and linguistic data. Chen and Mooney [59], with the objective to decide the important events to comment on, from sportscasts of the simulated soccer game, presented a novel commentator system that uses Iterative Generation Strategy Learning (IGSL) algorithm.

Navigation task (NT): the task of translating the natural language instructions into a formal intermediate path description language also received substantial attention. In 2006, Wong and Mooney [61] introduced a novel Word Alignmentbased Semantic Parsing (WASP) algorithm to construct a formal representation of a sentence using statistical machine translation (SMT) techniques. Matuszek et.al. [62] used WASP to develop probabilistic translation model to follow navigational instructions without prior linguistic knowledge. Following this, Chen and Mooney [63] presented a way to accomplish NT without prior linguistic knowledge, and only by observing human behavior in complex synthetic indoor environments. In 2013, Matuszek et al. [64] presented a system to learn grounding relations from data without any predefined-mappings, that executes the required commands in the previously unseen environment.

Thompson et al. [36], [37] presented clarification dialogues as a potential strategy to acquire perceptual concepts on-the-fly. She and Chai [38] came up with an interactive learning approach where robots actively engage with humans to acquire models of grounded verb semantics. Further, Chai et al. [39] extended the work by utilizing Reinforcement learning to determine when to ask what question to maximize the long-term reward.

Some works move past beyond using just visual percepts, and utilize properties like weight or sound to establish groundings [42], [40], [41], [43]. Motivated by human concept acquisition, Kiela and Clarke [40] came up with the approach of learning the meaning of the object (say, some musical instrument) not only by visual properties but also by auditory information like sound, pitch, and timber. Chen and Ballard [43] present a way to accomplish a complex task of word learning by utilizing the speaker's intentional body movements like gaze, head and hand movements, to establish relations between word and their grounded meanings. Thompson et al. [35]

came up with an interesting approach of utilizing haptic, auditory and proprioceptive data, along with visual details to learn the groundings of natural language sentences and words via human-robot I spy game. Some of the other works focus on binding event sequence or actions to natural language [43, 45, 46, 47].

Attempts are made to perform grounded learning in multiple languages in order to reach a wider audience [68, 69]. Kery et al. [66] adopted an English-only Grounded Learning system [5, 9, 10], and examined the adaptations necessary for it to perform equally well with other languages. Our current work adopts the same Grounded Learning system, with the motive of measuring compatibility with spoken annotations.

Chapter 3

Background

3.1 Google speech-to-text

Google speech-to-text[1] is an easy-to-use API which allows conversion of speech to text format in real time using powerful neural network models. The API supports more than 120 languages and its variants. It has the ability to automatically transcribe proper nouns and context-specific formatting. The API can be used for real-time streaming, immediately returning resulting text as the user speaks. Another variation is to generate text from audio stored in a file. It comes along with some pre-developed models that can be used according to the use-case:

- command_and_search: best for small queries.
- phone_call: best for transcribing audio generated via phone calls.
- video: best for transcribing audio generated from a video.
- default: best for other cases.

The API accepts various parameters that can be modified as per use-case. Some of the important ones are:

[1] https://cloud.google.com/speech-to-text

Portions of this page are reproduced from work created and shared by Google and used according to terms described in the Creative Commons 4.0 Attribution License.

- ENCODING : Encoding of audio data sent to API. We are using LINEAR16 encoding.

- SAMPLE_RATE_HERTZ : Sample rate of the audio data sent (in hertz). Valid values are 8000-16000. Lower sampling rate may reduce accuracy. It is suggested to keep this 16000 Hz for optimal performance.

- LANGUAGE_CODE : language of the supplied audio. "en-US" is used for English.

- ENABLE_AUTOMATIC_PUNCTUATION : Boolean parameter to enable basic punctuation (full stop, question mark, comma) in returned text.

- MAX_ALTERNATIVES : Integer parameter referring to the number of recognition hypothesis to be returned. Default is one. Please note that server may return fewer responses than this parameter value. The generated text should be in qualifying confidence value to be included in response.

- CHUNK : Audio Recording Parameter: In the python file, chunk refers to frame size. Streaming Recognition recognizes live audio as it is captured using microphone. The audio stream is split into chunks (or frames), and sent in consecutive messages. Larger frames are more efficient, but can cause high latency. 100 ms frame size is the suggested trade off between accuracy and latency.

3.2 Category-based Grounded Learning System

Category-based Grounded Learning system is ”word-as-a-classifier” based model presented in [5] and expanded further in [9] and [10]. This system attempts to learn the meaning of words used in crowd-sourced descriptions by grounding them in the physical representation of the objects that the workers describe. We will be using this system in our work to analyze the compatibility of this system with spoken language descriptions. We call it Category-based because it binds natural language descriptions with RGB-D features and learns groundings in the form of color, shape, and object categories. We will be referring to this model by the name of CB-GLS throughout this thesis. Figure 3.1 shows the flow of the system from data collection to model evaluation.

- Image Feature Extraction: Pillai et al. used Microsoft Kinect to gather RGB-D data of the ordinary household objects. Both color and depth features are crucial for color, shape, and object classifiers. Originally, the authors used an image dataset containing 18 object categories, each category comprised of 4 instances. Further, each instance had 4-5 images. We use a subset of this image dataset to define the baseline for our study.

- Image Description Collection: The second step is to gather the descriptions for the images generated. The authors used AMT to collect the language descriptions in written English format and gather over 3000 descriptions. We use the subset of this language dataset in our baseline study.

- Positive and Negative Data Extraction: After data cleaning and pre-processing, the next step is to identify positive and negative images associated with meaningful tokens. The authors used the TF*IDF strategy to identify the tokens that should be learned by the system. The positive images related to the specific token are determined using the "bag-of-words" strategy. If the token appears in the descriptions of images more than a threshold number of times, the image is considered as a positive example. For negative data points, authors use Semantic Similarity between the descriptions [9]. The cosine similarity between the vectors generated from descriptions is used as a distance metric.

- Learning Classifiers: Image and depth features are now clubbed together with the identified meaningful tokens. RGB-D data of the positive and negative examples are used to train three binary classifiers (color, shape, and object) associated with each useful token. RGB data is used to train the color classifier, depth data is used for shape classifier, and both RGB and depth data is used to train object classifier. The idea behind learning three classifiers is that it is not known beforehand what the token might be describing - color, shape, or object.

- Evaluation: In this step, test images are evaluated against all the learned classifiers. Classifiers are scored based on how well they can report whether the image is positive or negative instance associated with the concerned token.

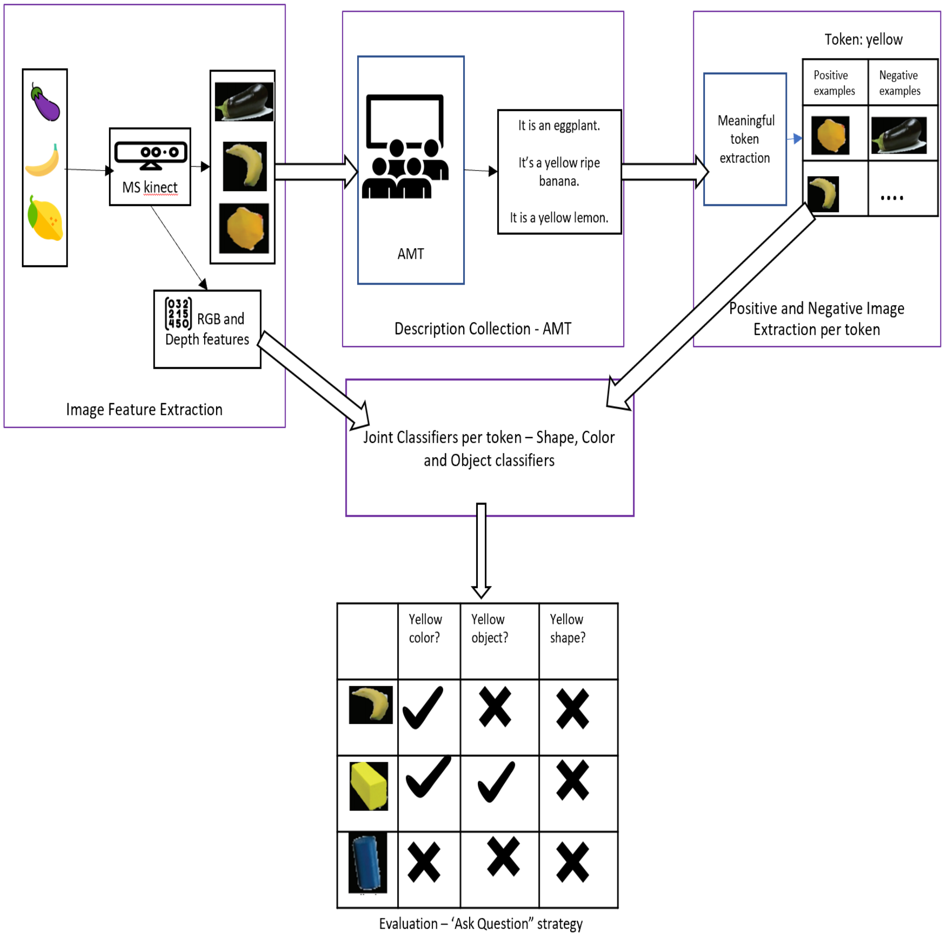

MS kinect
RGB and Depth features
Image Feature Extraction
AMT
It is an eggplant.
It's a yellow ripe banana.
It is a yellow lemon.
Description Collection - AMT
Token: yellow
Meaningful token extraction
Positive examples
Negative examples
....
Positive and Negative Image Extraction per token
Joint Classifiers per token – Shape, Color and Object classifiers
Yellow color?
Yellow object?
Yellow shape?
Evaluation – 'Ask Question" strategy

Figure 3.1: Flow of Category-based Grounded Learning System [5], [9] and [10]. Icons made by Freepik from `www.flaticon.com`

3.3 Category-Free Grounded Learning System

Pillai et al. presented a learning system in which language is grounded in visual precepts without pre-defined category constraints [11]. It's named Category-Free because it is a more general approach that learns grounding without pre-specifying the category constraints (see figure 3.2 for the design diagram). Instead of learning three different classifiers per meaningful token as in CB-GLS (yellow-as-color, yellow-as-shape, and yellow-as-object), this approach attempts to learn a single classifier (yellow-classifier). The approach involved the following steps:

- Concatenation of all visual features (RGB-D) into a single vector.
- Generating latent embeddings for the concatenated vectors. The authors used a deep generative model of variational autoencoder [48] for the purpose.
- Combine visual embeddings and language descriptions to learn one classifier per token.

3.4 Pre-processing techniques

Data cleaning and pre-processing is the first step after language data collection. Basic data cleaning involves removing punctuations and converting the descriptions into lower case. Sometimes, removing multiple spaces between th ethe words is also required for smooth tokenization. Audio files may need some modifications to ensure the compatibility with Automatic speech recognition systems. We use FFmpeg[2] to

[2] https://www.ffmpeg.org/

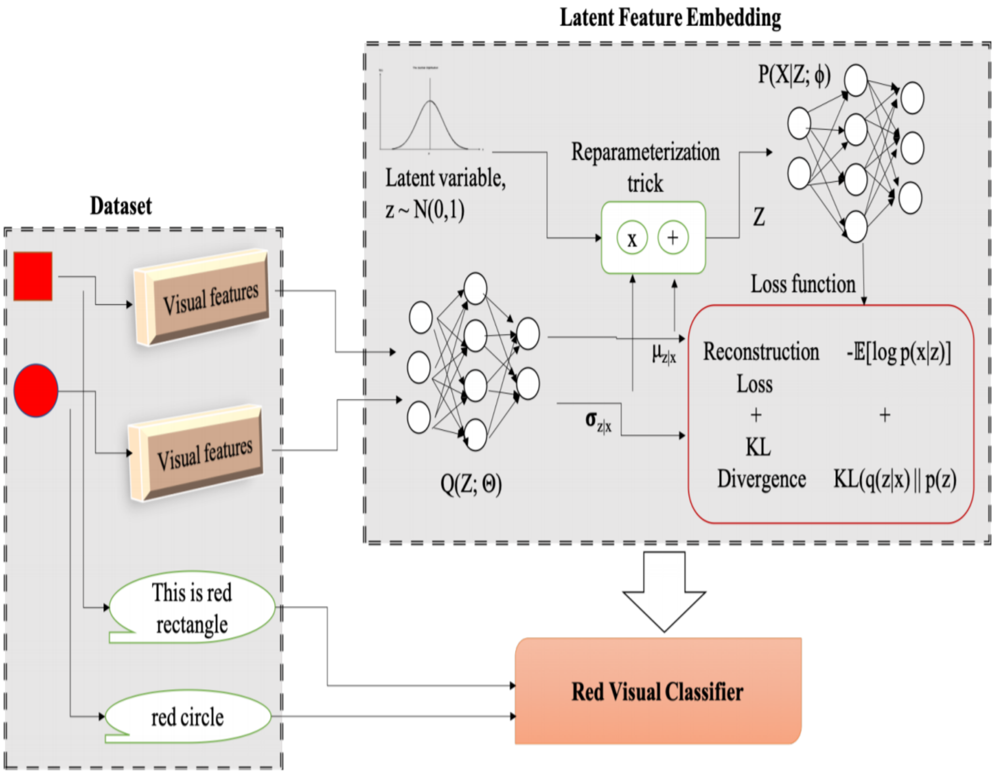


Figure 3.2: Design Diagram for Category-Free Grounded Language Acquisition. Reprinted from [11].

NLTK English stopwords
’ourselves’, ’hers’, ’between’, ’yourself’, ’but’, ’again’, ’there’, ’about’, ’once’, ’during’, ’out’, ’very’, ’having’, ’with’, ’they’, ’own’, ’an’, ’be’, ’some’, ’for’, ’do’, ’its’, ’yours’, ’such’, ’into’, ’of’, ’most’, ’itself’, ’other’, ’off’, ’is’, ’s’, ’am’, ’or’, ’who’, ’as’, ’from’, ’him’, ’each’, ’the’, ’themselves’, ’until’, ’below’, ’are’, ’we’, ’these’, ’your’, ’his’, ’through’, ’don’, ’nor’, ’me’, ’were’, ’her’, ’more’, ’himself’, ’this’, ’down’, ’should’, ’our’, ’their’, ’while’, ’above’, ’up’,’ both’. ’to’, ’ours’, ’had’, ’she’, ’all’, ’no’, ’when’, ’at’, ’any’, ’before’, ’them’, ’same’, ’and’, ’been’, ’have’, ’in’, ’will’, ’on’, ’does’, ’yourselves’, ’then’, ’that’, ’because’, ’what’, ’over’, ’why’, ’so’, ’can’, ’did’, ’not’, ’now’, ’under’, ’he’, ’you’, ’herself’, ’has’, ’just’, ’where’, ’too’, ’only’, ’myself’, ’which’, ’those’, ’i’, ’after’, ’few’, ’whom’, ’t’, ’being’, ’if’, ’theirs’, ’my’, ’against’, ’a’, ’by’, ’doing’, ’it’, ’how’, ’further’, ’was’, ’here’, ’than’

Table 3.1: NLTK English stop-words

add the required metadata before transcribing them using google speech.

3.4.1 Removing Stop Words

As discussed in sections 3.2 and 3.3, we are concerned with learning tokens as classifiers. Stop words are the tokens that do not contribute to the meaning of the sentence. It is necessary to remove such tokens to avoid our system to learn such classifiers. We use NLTK English stop word corpus[3] to identify and remove them from language descriptions (see table 3.1).

[3] http://www.nltk.org/nltk_data/

3.4.2 Lemmatization

Lemmatization is a technique to find the normalized form of the word. The strategy is to identify a lemma, or "basic form" for each token present in the document [49], [50]. Lemmatizers attempts to replace the various variants of the word (baking, baked) to their corresponding meaningful root (bake).

3.4.3 Stemming

Stemmers do not aim to replace various variants of a word by a correctly spelled root word. Instead, it removes the affixes attached to the word. The core idea is to reduce a word to their stem/root/base form makes it suitable for different languages. One example can be converting "baking,baked,bake" to "bak". We use Snowball stemmer [51] in our work; it's specifically designed for creating stemming algorithms for use in Information Retrieval [54]. It is partly based on the familiar Porter stemmer for English [52].

3.4.4 TF*IDF

TF-IDF stands for Term frequency-inverse document frequency, it's one of the most popular weighting scheme in information retrieval systems.TF-IDF is a statistical measure used to evaluate the importance of that word to the document in corpus. TF refers to the frequency of a word occurred in the document. Normalization is required to avoid bias between long and short documents. In our work, we consider each object description as a document. IIDF or Inverse Document Frequency defines how often a token appears in all the descriptions of all the objects; it determines how important the word is. The more often token appears, IDF parameter decreases. The strategy is to weigh down the words which appear in most of the documents or descriptions. In our study, we collected descriptions for object kept on a turntable. So, many users used "turntable" in the descriptions, such tokens do not contribute any information about the object itself and hence are called domain-specific stop-words [55], [56].

$$TF(t) = \frac{Number\ of\ times\ term\ t\ appears\ in\ the\ document}{Total\ number\ of\ terms\ in\ the\ document} \tag{3.1}$$

$$IDF(t) = \log\left(\frac{Total\,number\,of\,documents}{Number\,of\,documents\,with\,term\,t\,in\,it}\right) \tag{3.2}$$

Chapter 4

Crowd-Sourced Data Collection and Analysis - Speech and Text

4.1 Introduction

In this section, we present a multi-model Grounded Language Dataset (GLD) containing images of everyday household objects and language descriptions in multiple formats: text, audio, and transcribed speech, and can be found in `https://github.com/iral-lab/UMBC_GLD`. We elaborate on our methods for creating visual corpus as well as collecting the spoken and textual language descriptions. We attempt to analyze and study the nuances between the speech and text descriptions.

4.2 Image Data Corpus

We use Microsoft Azure Kinect[1], widely known as Kinect 3 to collect image and depth features. Figure 4.1 shows the data collection setup where Kinect 3 is mounted on a tripod, and the object is placed on the turntable to collect images from various angles. We gather raw images and point cloud data from 47 classes of objects across the five high-level categories - home, office, medical, tools, and food items. Table 4.1 shows all the object classes grouped by their high-level category. Each class of object contains roughly 4 or 5 instances, and each instance approximately contains four images taken from different angles. Each object class and the corresponding number

[1] https://docs.microsoft.com/en-us/azure/kinect-dk

of instances are mentioned in table 4.2. Figure 4.2 show different instances of the same object class, and figure 4.3 presents images taken from different angles for the same object instance. Images of the same object from various angles make our data set more diverse and robust. In total, the data set comprises of 47 classes, 207 instances, and 825 images.

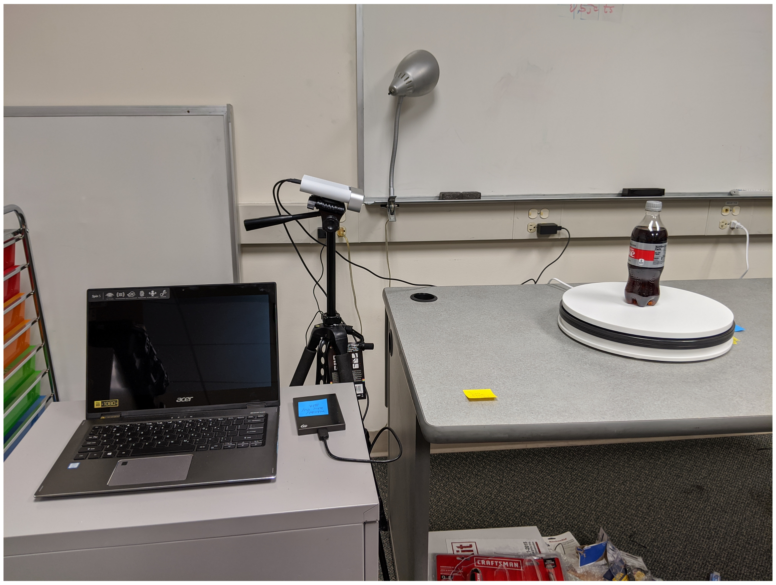


Figure 4.1: Data collection setup for RGB-D collection- Kinect 3 is mounted on tripod. Object shown is soda bottle placed on turn table, visual features are collected as turn table rotates for 90 seconds.

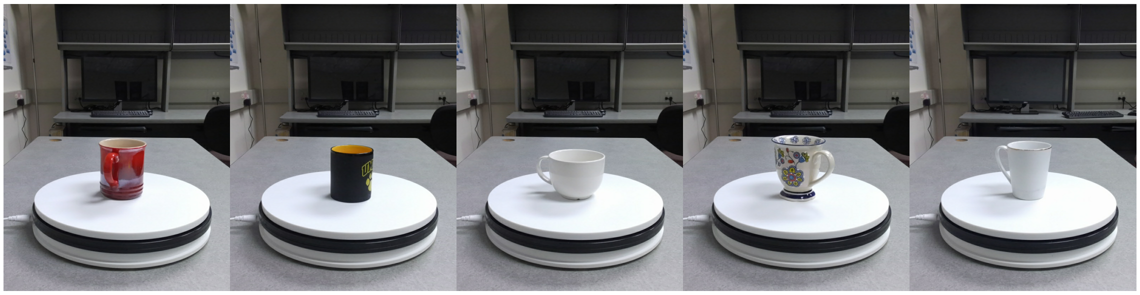


Figure 4.2: Five instances of Coffee Mug object class.

Category	**Classes of Objects**
home	*book, can opener, eye glasses, fork, shampoo, sponge, spoon, toothbrush, toothpaste, bowl, cap, cell phone, coffee mug, hand towel, tissue box, plate*
office	*mouse, pencil, picture frame, scissors, stapler, marker, notebook*
medical	*band aid, gauze, medicine bottle, pill cutter, prescription medicine bottle, syringe*
tool	*allen wrench, hammer, measuring tape, pliers, screwdriver, lightbulb*
food	*potato, soda bottle, water bottle, apple, banana, bell pepper, food can, food jar, lemon, lime, onion*

Table 4.1: The classes of objects grouped by their high-level category.

4.3 Data Collection - Textual

We split the raw image videos into frames and select frames that capture the object from different angles. We collect text descriptions for the images using Amazon Mechanical Turk. Workers are asked to describe the object in a sentence or two as if they are explaining it to someone else. The instructions mentioned not to describe the background or turntable. A sample Mechanical Turk HIT is shown in figure 4.4. Each HIT contains five objects to be described in a text format. We

Object Class	**Number of Instances**
hammer, marker, scissors	6
allen wrench, apple, banana, band-aid, bell pepper, book, bowl, can opener, coffee mug, food can, food jar, fork, lemon, lightbulb, lime, measuring tape, medicine bottle, onion, pencil, picture frame, plate, pliers, potato, screwdriver, shampoo, soda bottle, spoon, stapler, syringe, toothbrush, toothpaste, water bottle	5
eye glasses	4
cell phone, flashlight, hand towel, mouse, prescription medicine bottle	3
cap, gauze, notebook, sponge	2
pill cutter, tissue box	1

Table 4.2: Object classes and number of instances per class.

assign five assignments per task and collect 8125 text descriptions in total. Collecting descriptions for images taken from different angles is crucial. Consider a case when a person talking to a robot has a partial view or understanding of the object; it is necessary to accommodate these groundings in our model. Consider figure 4.3, where one of the mug images has a paw print on it, and the other two have either a partial view or no such design. But all these images are of the same object instance; hence, it is essential to gather descriptions for images taken from different angles.

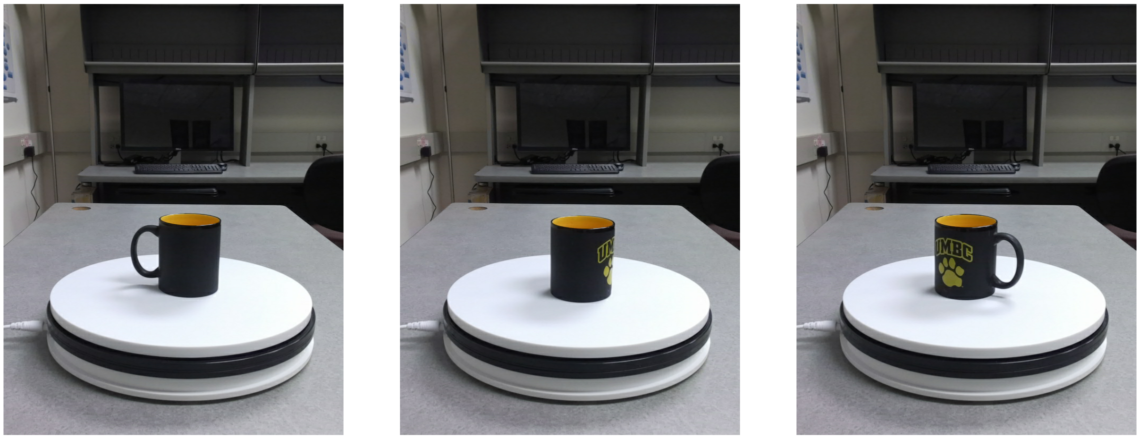
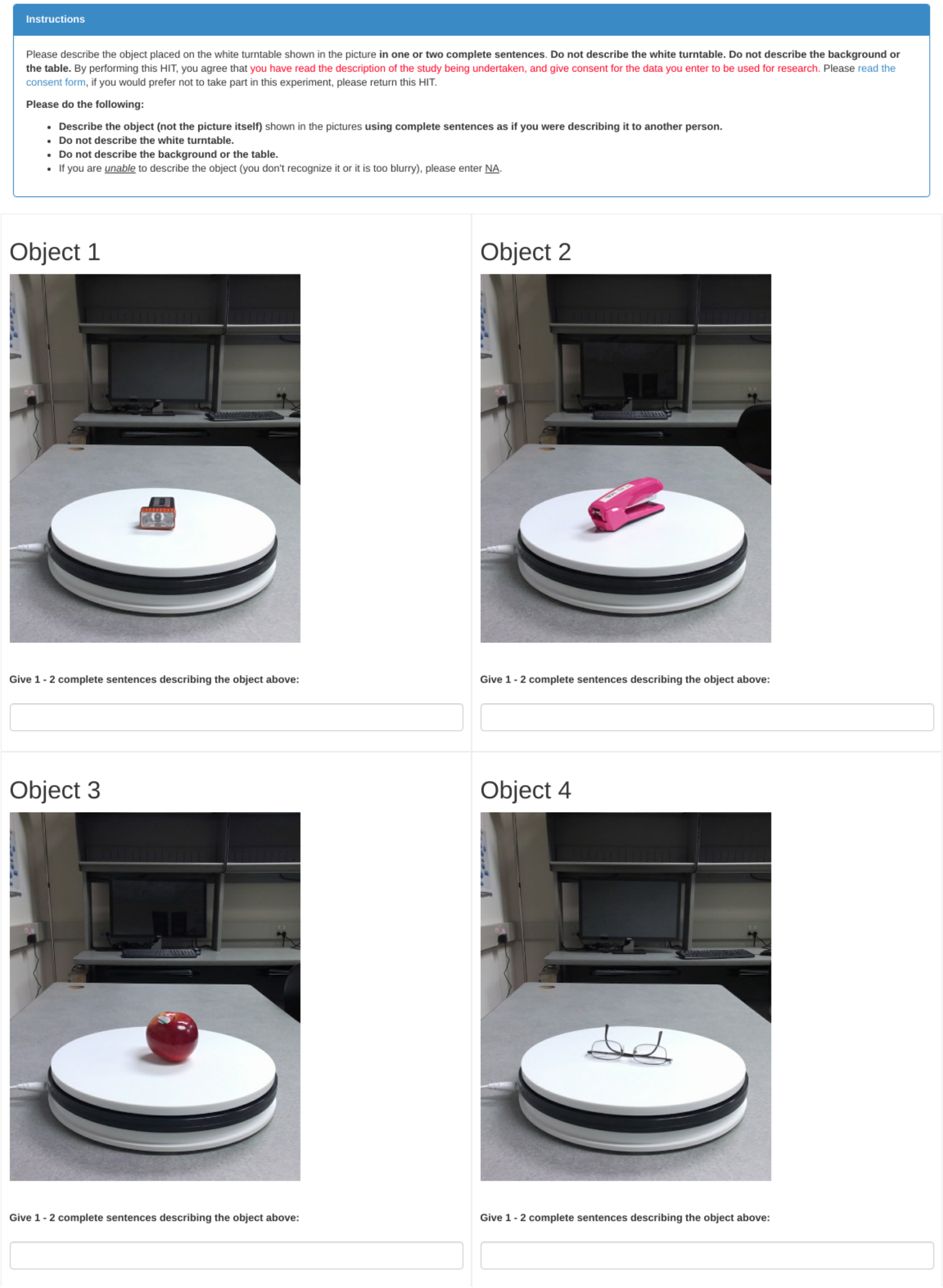


Figure 4.3: Images taken from different angles for same instance of coffee mug. First image has no paw print, second and third image has either partial or full paw print.

4.4 Data Collection - Speech

The motive of collecting audio data is to capture the nuances between spoken and written language. It is common practice to restructure sentences before writing them, but while speaking, we do not have the liberty to re-frame or restructure them. Therefore, spoken sentences might be not very well framed or can be grammatically incorrect. We support speech with body gestures, eye gaze, expressions or pitch of



Instructions

Please describe the object placed on the white turntable shown in the picture **in one or two complete sentences**. **Do not describe the white turntable. Do not describe the background or the table.** By performing this HIT, you agree that you have read the description of the study being undertaken, and give consent for the data you enter to be used for research. Please read the consent form, if you would prefer not to take part in this experiment, please return this HIT.

Please do the following:

- **Describe the object (not the picture itself)** shown in the pictures **using complete sentences as if you were describing it to another person.**
- **Do not describe the white turntable.**
- **Do not describe the background or the table.**
- If you are *unable* to describe the object (you don't recognize it or it is too blurry), please enter NA.

Object 1

Give 1 - 2 complete sentences describing the object above:

Object 2

Give 1 - 2 complete sentences describing the object above:

Object 3

Give 1 - 2 complete sentences describing the object above:

Object 4

Give 1 - 2 complete sentences describing the object above:

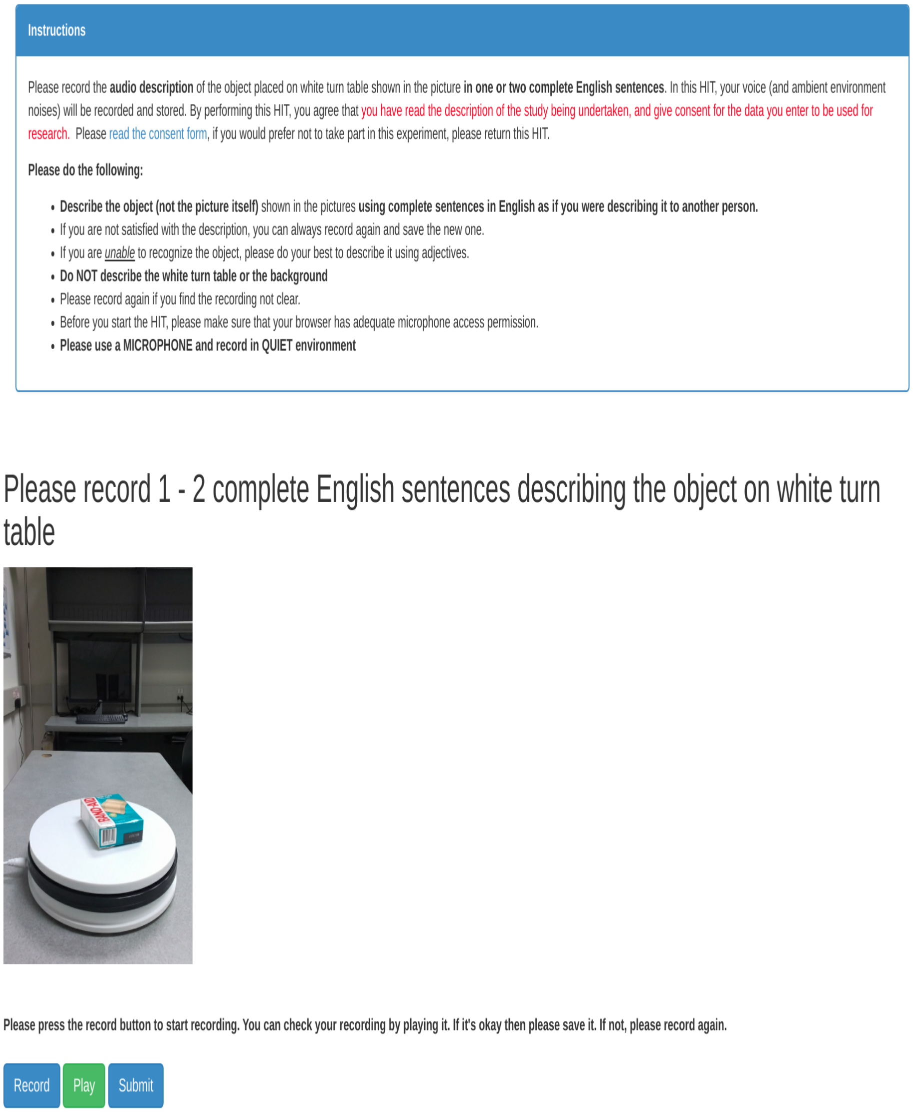
Figure 4.4: User Interface for text data collection embedded in Amazon MTurk.

the voice, which on the contrary are missing in written text. Experienced writers may be able to overcome these differences while communicating. However, these people usually hold formal education [14]. So, to enhance human-robot interaction for a broader group of end users, it will be necessary to train robots with spoken data. Moreover, it is possible that this unorganized and spontaneous obtained data can prepare the robot even better for real-world scenarios. We develop a user interface to collect spoken natural language data using MediaStream recording API.[2] Further, the audio clips are stored in Amazon S3[3] bucket, which is a cloud storage service. A sample HIT is shown in figure 4.5. Workers can play the recorded audio and if not satisfied can record it again. Each HIT contains one object to be described in spoken language and we assigned 5 assignments per task. We collect 4059 audio descriptions in total. A similar approach is reported in recent work [15, 16] to collect data using web and mobile application-based systems. We embed the interface into Amazon Mechanical Turk, and the recorded audio files are collected from these tasks. We use the FFmpeg library[4] to add the missing metadata from the audio files to make them compatible with ASR systems. The audio files are then converted to text using Google's Speech to Text API.

[2] https://developer.mozilla.org/en-US/docs/Web/API/MediaStream_Recording_API

[3] https://aws.amazon.com/s3/

[4] https://www.ffmpeg.org/



Instructions

Please record the **audio description** of the object placed on white turn table shown in the picture **in one or two complete English sentences**. In this HIT, your voice (and ambient environment noises) will be recorded and stored. By performing this HIT, you agree that you have read the description of the study being undertaken, and give consent for the data you enter to be used for research. Please read the consent form, if you would prefer not to take part in this experiment, please return this HIT.

Please do the following:

- **Describe the object (not the picture itself)** shown in the pictures **using complete sentences in English as if you were describing it to another person.**
- If you are not satisfied with the description, you can always record again and save the new one.
- If you are *unable* to recognize the object, please do your best to describe it using adjectives.
- **Do NOT describe the white turn table or the background**
- Please record again if you find the recording not clear.
- Before you start the HIT, please make sure that your browser has adequate microphone access permission.
- **Please use a MICROPHONE and record in QUIET environment**

Please record 1 - 2 complete English sentences describing the object on white turn table

Please press the record button to start recording. You can check your recording by playing it. If it's okay then please save it. If not, please record again.

Figure 4.5: User Interface for speech data collection embedded in Amazon MTurk.

4.5 Comparative Analysis - Speech and Text corpus

In this section, we compare the text and speech corpus in various aspects. We compare the two based on the number of words and characters in the description and presence of nouns, adjectives, and verbs. We note the behavior when stop words are removed. We attempt to measure the accuracy of transcriptions developed using google speech-to-text using different metrics like WER and BLEU scores. We perform pilot studies for quality evaluation and calculate Fleiss' kappa scores for inter-reliability before moving to a more extensive set.

4.5.1 Most Frequent Terms

Table 4.3 shows the most frequent tokens in text and spoken data. Most of the tokens are consistent in both cases. The color appears as the most common choice to describe the objects. Note that the difference in frequency of tokens in two cases is because the textual data is almost double when compared to speech data. We notice some interesting observations in both cases. People tend to use filler words when describing the objects using speech. For example, the term "like" appears 166 times in speech data, whereas it was not significant in the text data. We also observe the frequency of the word "used" is high in both cases, which are typically used to describe the functionality of certain objects.

Token	**Frequency**
black	1073
object	924
white	817
blue	784
red	746
bottle	732
yellow	718
small	482
used	449
pair	436
green	432
plastic	341
box	310
silver	265
metal	220
pink	219
picture	188
orange	174
large	173
jar	164

Token	**Frequency**
black	599
white	545
blue	427
bottle	385
red	360
yellow	353
object	268
green	231
used	223
handle	210
small	185
color	171
like	166
box	163
silver	163
pair	153
plastic	151
looks	131
pink	109
light	102

Table 4.3: The top 20 most frequent words and their frequencies in the textual descriptions (left) and in the transcribed speech data (right).

4.5.2 Sentence length Analysis - Number of words

We hypothesized, initially, that people would use more number of words while describing the objects in speech when compared to text because the effort consumed in talking is less as compared to typing. We calculate the mean number of tokens used by workers to describe the objects in both cases. We use all the speech and text descriptions for these calculations. We find that the average is around 8.72 in speech and 8.38 in the text corpus. After the removal of NLTK English stop words, the mean of speech data length drops to 4.52, and in the case of text data, the average comes down to 4.38 (see figures 4.6a and 4.6b for distribution plots). We expected earlier that people would use more number of stop words in speech than written descriptions. However, the mean drop difference is not significant. We observe that the drop in mean difference is around 4.19 in speech and 3.99 in text data. The distribution plots of sentence length show that the significant chunk lies below 20 words per sentence in both the cases (see figures 4.6 and 4.8). The difference lies in the number of high word count sentences. In the case of text data, the number of sentences with token length greater than 30 is 22, whereas it's 116 in the case of speech data. Percent wise, the sentences with word count higher than 30 make 2 percent of the speech corpus, whereas this percent value is 0.54 in case of text description. Consider 20 as a threshold measure; this percent value is 6.6 percent in speech corpus and 3.3 percent in the text corpus. We observe a similar behavior towards the lower end of the spectrum. The number of descriptions with length less than 4 is 327 in the text corpus and 750 in speech corpus. Percent wise, these sentences make 18 percent in speech and 8 percent in the text corpus.

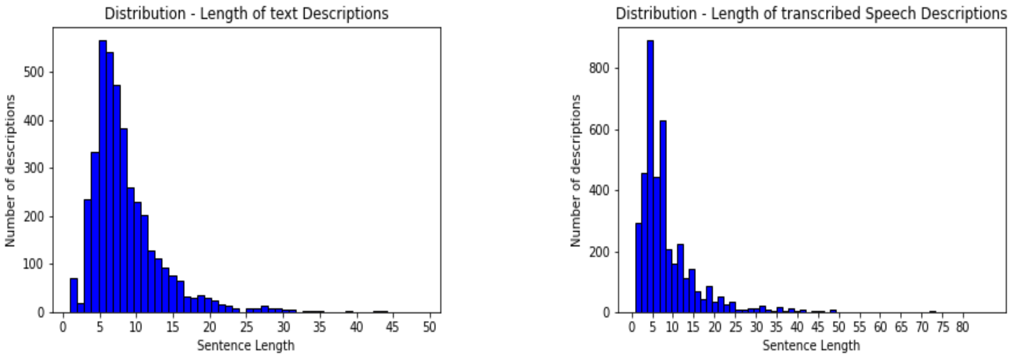
When we remove stop words, the distribution plots show that most of the descriptions contain less than 15 tokens (see figures 4.7b and 4.7b). In this case 269 speech descriptions contains more than 10 tokens and 138 of textual descriptions comprises more than 10 tokens (see figures 4.7 and 4.9).

The middle value or median lies close to each other in both the data sets. In the text corpus, the median is 7, and in speech, it is 6. When we remove stop words, median values drop to 4 and 3 for text and speech data sets, respectively (see figure 4.11).

We find that the length of text descriptions are less spread out when compared to speech ones. We observe this behavior in the original descriptions as well as the ones with no stop words. The standard deviation of lengths of text descriptions is 5.14, and in the case of speech, it is around 8. When we remove stop words, the standard deviation comes down to 2.58 for text data and 3.96 for speech data (see figure 4.10).

4.5.3 Sentence length Analysis - Number of characters

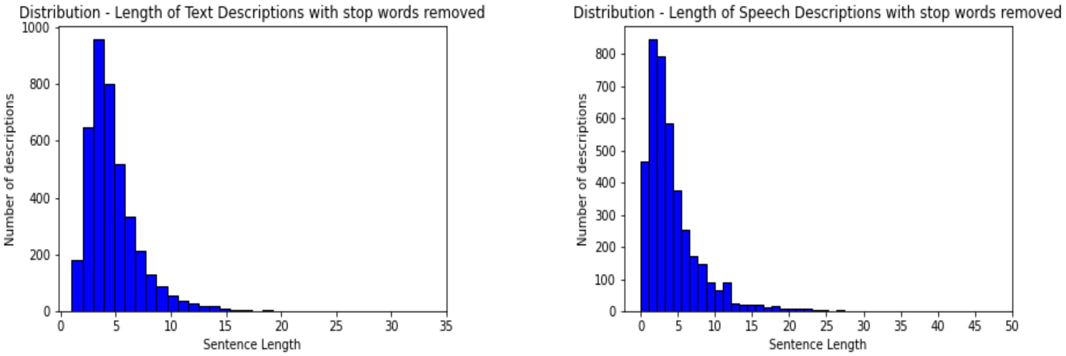
In this section, we compare the text and speech corpus based on the number of characters in the description. When the number of characters is concerned, the mean and median of the data sets are in similar lines. The mean number of characters 41.2 and 42.4 in text and speech corpus, respectively 4.12. Median is on higher-end in the case of the text corpus. It is 34 for text and 31 in the case of speech descriptions.



(a) Text distribution

(b) Speech distribution

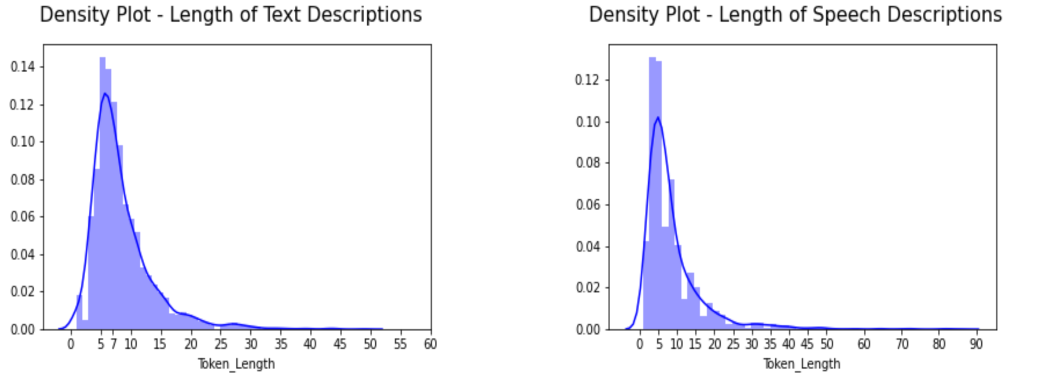
Figure 4.6: Distribution of sentence length in Text and Speech data.



(a) Text distribution

(b) Speech distribution

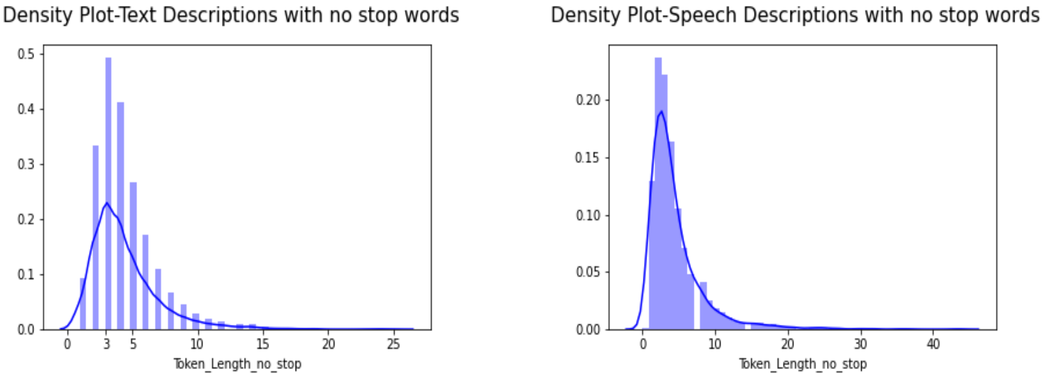
Figure 4.7: Distribution of sentence length in Text and Speech data with stop words removed.



(a) Text distribution

(b) Speech distribution

Figure 4.8: Density Probability Distribution of sentence length in Text and Speech data.



(a) Text distribution

(b) Speech distribution

Figure 4.9: Density Probability Distribution of sentence length in Text and Speech data with stop words removed.

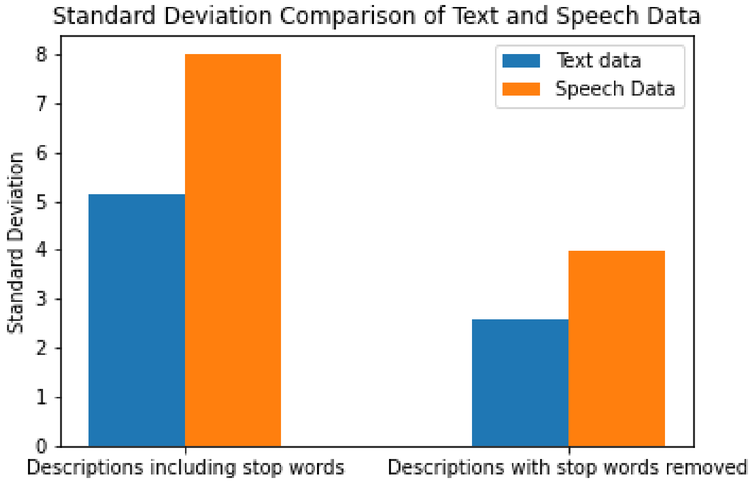


Figure 4.10: Standard deviation Comparison.

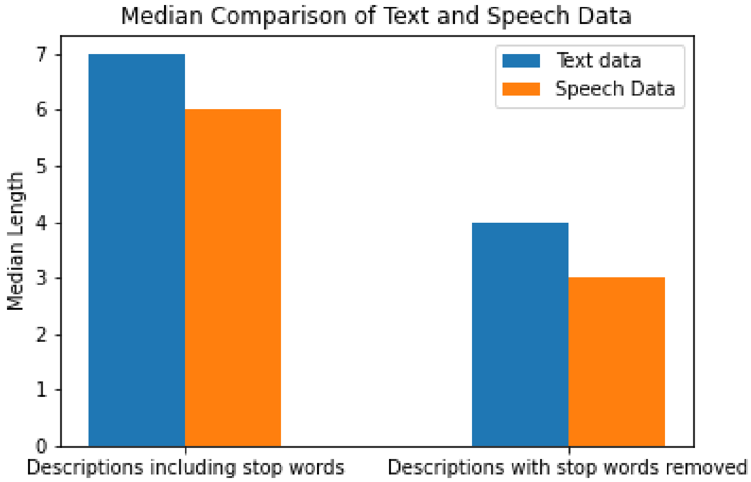
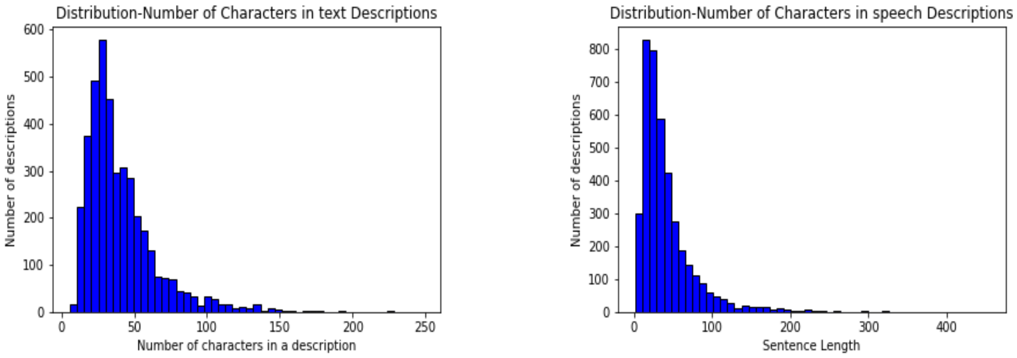


Figure 4.11: Median Comparison.

The interesting observation here is the difference in standard deviation between the two sets. The text corpus shows the standard deviation of 25.9, whereas it is 39.2 in the case of speech set.

We observe high variations in the standard deviation of the number of characters and number of words in both the data sets. Hence, we see a greater variety in descriptions when the length is concerned or the number of tokens used. Sometimes, google speech-to-text collapse multiple mumbled words into a single long term. It also sometimes breaks a single complex word into multiple words. It happens when the input speech is not very clear. This kind of behavior may occur because of the quality of the microphone, background noise, or accent of the user. These reasons may also contribute to high fluctuations in the length of descriptions in speech corpora.



(a) Text distribution

(b) Speech distribution

Figure 4.12: Distribution-Number of characters in text and speech corpus.

4.5.4 Nouns, Adjectives and Verbs in Descriptions

We use Stanford Part-of-Speech (POS) Tagger [5] to count the number of nouns, adjectives, and verbs in the descriptions. We are interested in evaluating the occurrence of nouns, verbs, and adjectives because we believe they play a central role in defining groundings associated with any object. We find that the mean number of noun tokens in text descriptions is slightly higher than the speech one. The same is the case with adjectives. When verbs are concerned, occurrence in speech descriptions is marginally higher. When we remove nouns, adjectives, and verbs from the sentence, mostly determiners, articles, pronouns, conjunctions, and auxiliary tokens are left. We find that the mean occurrence of these tokens is around 10.4 percent higher in speech than in text corpora (see table 4.4). The results show that people tend to use more such words while describing the objects in spoken language. This may also occur due to transcriptions produced by google speech-to-text. Consider one such example mentioned below.

Google transcribed text: *again used for containing food*

User said: *a can used for containing food*

In the above example, google transcribed text missed a noun (can) which is an important associated grounding to the object (food can). These errors may also affect the noun and adjective count in the speech corpora.

[5] https://nlp.stanford.edu/software/tagger.shtml

	Text Descriptions	**Speech Descriptions**
Mean Noun occurrence	2.59	2.49
Mean Adjective occurrence	1.25	1.17
Mean Verb occurrence	0.52	0.62
Other token mean occurrence	4.02	4.44

Table 4.4: Averaged number of noun, adjective and verb occurrence in text and speech data.

4.6 Accuracy of Google transcribed Responses

In this section, we aim to measure the accuracy of the collected speech transcribed descriptions. We observed that the standard deviation of the length of google transcribed text of the collected speech corpora is on the higher end when compared to the text data set. It is necessary to evaluate the transcriptions as we further use this data with the category-based grounded learning system. To measure transcription accuracy, we manually rate the randomly picked 100 audio samples, which comprises around 2.4 percent of the whole speech data set. We also measure the accuracy of transcriptions using Bilingual Evaluation Understudy or BLEU metric.

4.6.1 Pilot studies to rate transcriptions

Before moving to a more extensive set, we conducted two pilot experiments where three raters evaluated the quality of transcriptions. In the first experiment, we randomly picked nine audio files to be rated. Raters are asked to evaluate the quality out of 4 using the following scheme:

- Rating 1: wrong transcription or gibberish/unusable sound file,
- Rating 2: slightly wrong transcription (missing keywords/concepts),
- Rating 3: pretty good transcription (main object correctly defined),
- Rating 4: perfect transcription (accurate transcription and no errors).

Also, raters are asked to mark the audio file as Incomplete when the audio file is blank or unusable. The overall agreement between the three raters is about 75 percent. We calculate the Fleiss' kappa statistic [6] to assess the reliability of the agreement [12].

- N = num of cases (sound files = 9)
- n = num pf raters (3)
- k = num of categories (4)

In table 4.5, the categories are presented in column and the subjects (transcription) are presented in the rows. Each cell in the table lists the number of raters who assigned the indicated subject (row) to the indicated category (column).

[6] https://en.wikipedia.org/wiki/Fleiss_kappa

n_{ij}	**RATING #1**	**RATING #2**	**RATING #3**	**RATING #4**	$\sum_{j=1}^{k} n_{ij}^2$	P_i
Transcription 1	0	0	1	2	**5**	**0.3333333333**
Transcription 2	0	0	0	3	**9**	**1**
Transcription 3	0	3	0	0	**9**	**1**
Transcription 4	0	1	2	0	**5**	**0.3333333333**
Transcription 5	0	3	0	0	**9**	**1**
Transcription 6	0	0	0	3	**9**	**1**
Transcription 7	0	0	0	3	**9**	**1**
Transcription 8	0	0	3	0	**9**	**1**
Transcription 9	0	1	1	1	**3**	**0**
TOTAL	**0**	**8**	**7**	**12**		
p_j	**0**	**0.2962962963**	**0.2592592593**	**0.4444444444**		

Table 4.5: kappa calculation for pilot 1.

$$p_j = \frac{1}{Nn}\ sum_{i=1}^{N} n_{ij} \tag{4.1}$$

$$\sum_{j=1}^{k} p_j = 1 \tag{4.2}$$

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} n_{ij}(n_{ij}-1)) \tag{4.3}$$

$$P_i = \frac{1}{n(n-1)} \sum_{j=1}^{k} (n_{ij}^2 - n_{ij}) \tag{4.4}$$

$$P_i = \frac{1}{n(n-1)} [(\sum_{j=1}^{k} n_{ij}^2) - n] \tag{4.5}$$

$$\bar{P} = \frac{1}{N} \sum_{i=1}^{N} P_i \tag{4.6}$$

$$\bar{P} = \frac{1}{Nn(n-1))} (\sum_{i=1}^{N} \sum_{j=1}^{k} n_{ij}^2 - Nn) \tag{4.7}$$

$$\bar{P}_e = \sum_{j=1}^{k} p_j^2 \tag{4.8}$$

$$\kappa = \frac{\bar{P} - \bar{P}_e}{1 - \bar{P}_e} \tag{4.9}$$

$$\sum_{i=1}^{N} P_i = 0.3333333333 + 1 + 1 + 0.3333333333 + 1 + 1 + 1 + 1 + 0 \tag{4.10}$$

$$\sum_{i=1}^{N} P_i = 6.666666667 \tag{4.11}$$

$\bar{P} = 0.7407407407$ (from equations 4.6, 4.7 and 4.11)

$\bar{P}_e = 0.3525377229$ (from table 4.5 and equation 4.8)

$\kappa = 0.5995762712$ (from equation 4.9)

Hence, the kappa statistic for pilot 1 study is about 0.6, which is considered to be moderate/substantial agreement between the raters.

In the second pilot experiment, we picked ten random transcriptions, and three

n_{ij}	**RATING #1**	**RATING #2**	**RATING #3**	**RATING #4**	$\sum_{j=1}^{k} n_{ij}^2$	P_i
Transcription 1	0	0	0	3	9	1
Transcription 2	0	0	0	3	9	1
Transcription 3	0	0	0	3	9	1
Transcription 4	0	0	2	1	5	0.3333333333
Transcription 5	0	0	0	3	9	1
Transcription 6	0	1	2	0	5	0.3333333333
Transcription 7	3	0	0	0	9	1
Transcription 8	2	1	0	0	5	0.3333333333
Transcription 9	2	1	0	0	5	0.3333333333
Transcription 10	0	0	0	3	9	1
TOTAL	7	3	4	16		
pj	0.2333333333	0.1	0.1333333333	0.5333333333		

Table 4.6: kappa calculation for pilot 2.

different raters are asked to evaluate the transcriptions using the same quality scale (out of 4). We improved the interface used for evaluation in terms of usability and clarity according to the feedback gathered in the first experiment. In this interface, the user can click on the link present in the spreadsheet to play the audio clip directly. In the previous study, a manual search in google drive was required.

From table 4.6 and equations 4.6

$$\sum_{i=1}^{N} P_i = 1 + 1 + 1 + 0.333 + 1 + 0.333 + 1 + 0.333 + 0.333 + 1 \tag{4.12}$$

$$\sum_{i=1}^{N} P_i = 6.666666667 \tag{4.13}$$

$\bar{P} = 0.7333333333$ (from equations 4.6, 4.7 and 4.13)

$\bar{P_e} = 0.3666666667$ (from table 4.5 and equation 4.8)

$\kappa = 0.5789473684$ (from equation 4.9)

The kappa statistic for the second pilot is around 0.58. Hence both the pilot results represent moderate/substantial agreement among the raters. Although the kappa scores are not high in both the pilot rounds, we observe that disagreement was not more than one on a quality scale between the raters. In multiple cases, we find that if two raters evaluated one of the subjects as 1, and the third rater marked it as 2. It never occurred that the third rater evaluated it on other extremes (4 in this case). However, kappa scoring does not incorporate such factors in the evaluation. So, we believe the kappa scores are sufficient to move further with evaluating a broader set.

4.6.2 Transcription Accuracy of 100 randomly selected descriptions

We choose 100 audio descriptions, which comprises of 2.4 percent of the whole speech data set and evaluate the quality. We rate these transcriptions out of 4 using the same rating scheme as in the pilot experiments. We also manually transcribe these 100 audio clips to measure the quality precisely. We find that 77 percent of the sound clips are high-quality, i.e., rated as 3 or 4. 13 percent of the clips are rated as 1, which is considered as gibberish/unusable (see figure 4.13).

To measure the quality of transcription, we use the BLEU metric; it is origi-

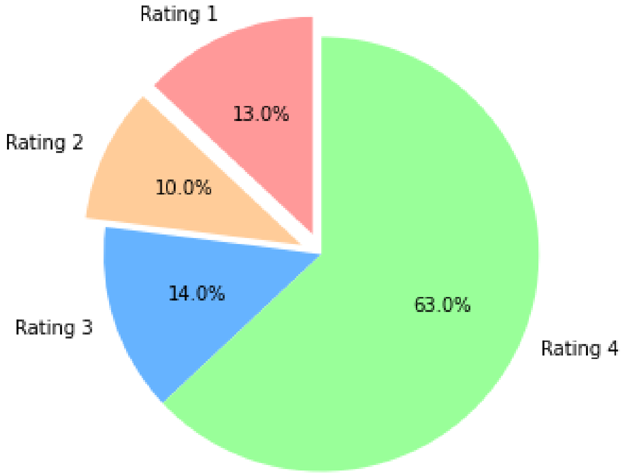


Figure 4.13: Quality rating wise distribution.

RATINGS	OBJECT DESCRIBED	GOOGLE TRANSCRIPTION	MANUAL TRANSCRIPTION
1	Toothpaste	Institute best	It's a toothpaste
1	Spoon	did Persephone used to serving before	this is spoon made up with wood used for serving food
1	Soda bottle	lovesick 100 African Buffalo	it is a plastic one and half liter bottle of coke
2	Stapler	this is the stuff inside mechanical device which joins Legends of paper	this is a stapler, it is a mechanical device which joins pages of paper
2	Can opener	emmanuel 10 opener with a blue handle	A manual tin opener with a blue handle
2	Hand towel	its a folded great owl	it's a folded gray towel
3	Shampoo bottle	what is a bottle of shampoo	that is a bottle of shampoo
3	Mouse	Addison black color Mouse can be used in laptop or system	it is a black color Mouse can be used in laptop or system
3	Coffee mug	Arizona white coffee mug	There is a white coffee mug

Table 4.7: Some example with transcription-quality ratings.

nally a method for automatic evaluation of machine translation. We use this metric as it is quick, and inexpensive that correlates highly with human assessment, and has a little marginal cost per run [13]. The core idea behind BLEU is the closer machine translation is to human translation, the better it is. BLEU is a widely used metric to measure the accuracy of language translations based on string similarity; we adopt this system to evaluate the goodness of transcriptions. This method rate the readings out of 1. Higher the score, the better the translation. The metric looks for the presence and absence of the concerning tokens. The method works by finding n-gram overlaps between machine translation and reference translations (see 4.14). N-grams are a contiguous sequence of n tokens present in a sentence. Unigram or 1-gram represents matching a singe token occurrence, and bigram corresponds to matching two adjacent words. BLEU metric system requires two inputs:

- Human reference translations (In our case, human transcriptions)
- Automated translation output of the same data set (In our case, google transcriptions)

$$BLEU = min(1, \frac{output - length}{reference - length})(\prod_{i=1}^{4} precision_i)^{\frac{1}{4}} \tag{4.14}$$

We find the BLUE scores of all the transcriptions in the randomly chosen sample of 100 speech descriptions. We find the mean BLEU score of the sample set to be 0.58, and the median is 0.76 (see figure 4.14 for distribution). When we only consider the transcription rated 3 or 4, the BLEU mean rises to 0.73, as expected. We observe an interesting relationship between Quality Ratings, BLEU scores, and the length of descriptions (see figure 4.16). We find that if the number of tokens in the description is low, the probability of incorrect transcription produced by google speech-to-text is higher (see figures 4.17 and 4.18). When the number of words in

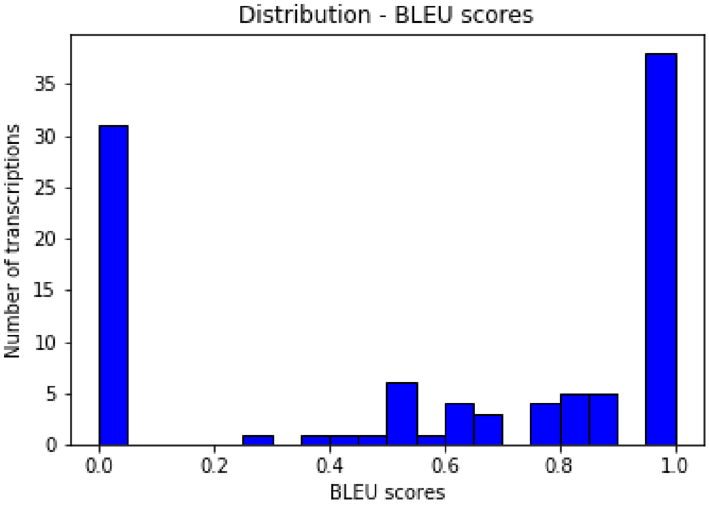


Figure 4.14: Distribution of BLEU scores.

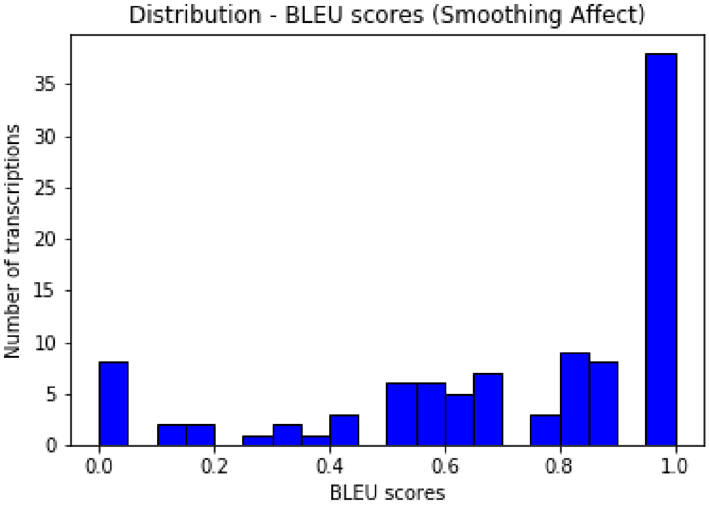


Figure 4.15: Distribution of BLEU scores after applying smoothing function.

the description is less than 3, the mean Rating is 1.62, and the average BLEU score drops to zero. The BLEU metric we used is not suitable for short descriptions, especially ones with length less than 4. In such cases, the score will always be zero because 4-gram never appears, causing the product of precisions to zero. To mitigate this effect, we apply smoothing function [78] to add 1 to both numerator and denominator while calculating precision. In this case, we find the mean BLEU scores to be 0.71 (figure 4.15 shows the distribution). Plots 4.17 and 4.18 shows the impact of sentence length on accuracy of transcriptions. We observe the short sentences with length less than 4 terms show poor transcription accuracy. This may occur due to the following reasons:

- When the number of words uttered is less, the google speech-to-text doesn't work as expected.

- Sometimes, google speech-to-text collapses multiple mumbled words into a single long term, resulting in a decrease in sentence length as well as accuracy.

Word Error Rate (WER) is widely accepted as the de facto metric for ASR. It works by calculating the distance between the system's results - called the hypothesis - and manually transcribed text - called the reference. It is derived from the Levenshtein distance, working at the word level instead of the phoneme level. The Levenshtein distance is a numerical value of the cost of the least expensive set of insertions, deletions, or substitutions that would be needed to transform one string into another [71]. The WER metric computes the minimum-edit distance

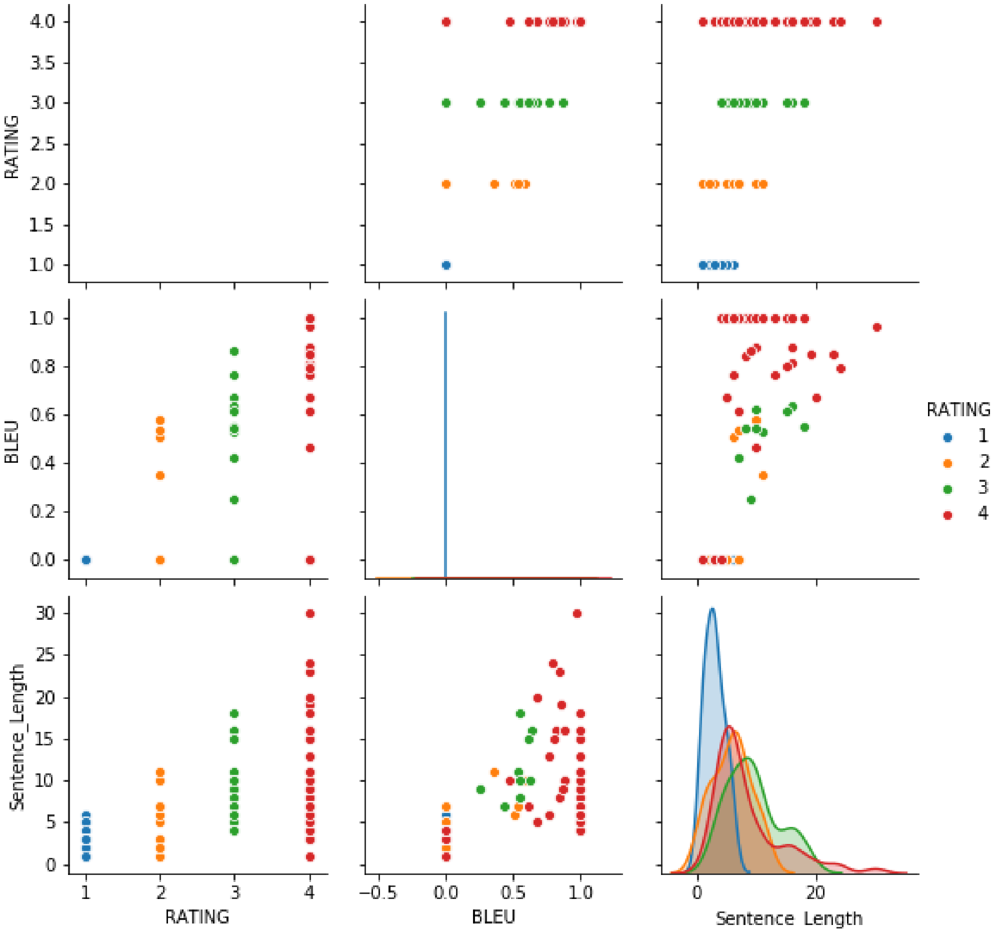


Figure 4.16: Relationship between BLEU score, Ratings and Sentence length.

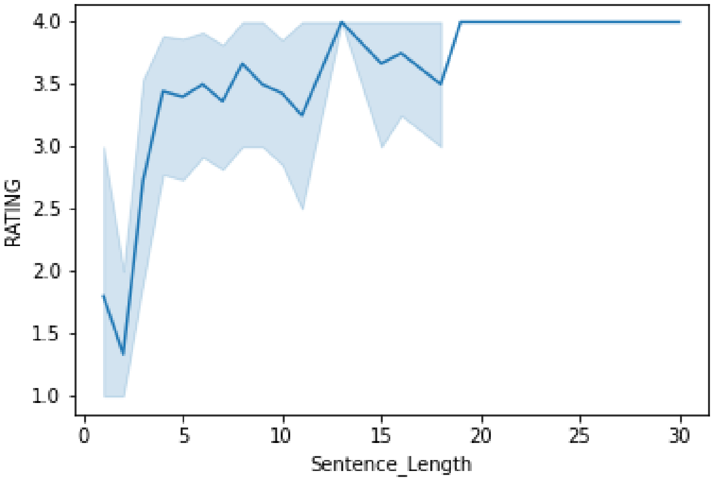


Figure 4.17: Relationship between Ratings and Sentence length.

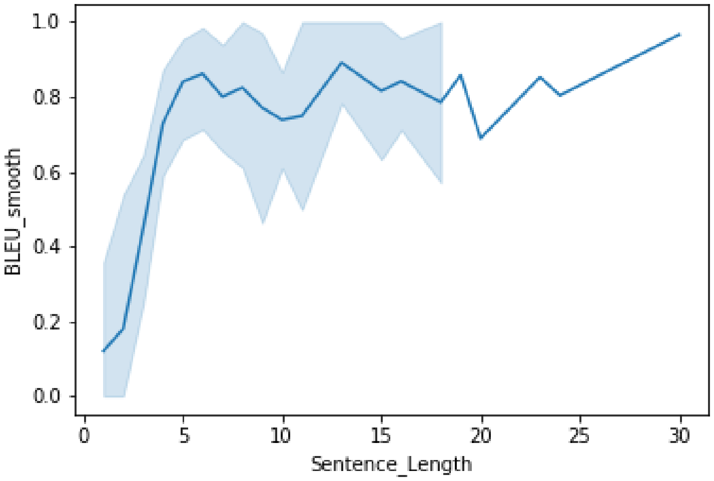


Figure 4.18: Relationship between BLEU scores and Sentence length.

between the ground-truth sentence and the hypothesis sentence of a speech-to-text API. WER can be computed as:

$$WER = 100 * \frac{S + D + I}{N} \tag{4.15}$$

where

- S = number of substitutions,
- D = number of deletions,
- I = number of insertions,
- C = number of correct words,
- N = number of words in reference (N= S+D+C).

We evaluate WER for the same subset of 100 randomly picked audio descriptions. We find that the mean per audio description WER is roughly 21.3 percent. Figure 4.19 shows the distribution of WER in the subset. Out of 100 audio descriptions, 42 showed the error rate exactly 0, that means the manual transcription and speech-to-text output matched precisely. For other 58, there exists atleast some transcription error.

4.7 Role of Accent and Gender

In the same random set of 100 audio descriptions as discussed above, we also note the gender and accent of the speaker. We tried to make these decisions as accurately as possible, but we understand there could be marginal errors involved. We find that the number of female speakers was slightly higher than male participants. Accent wise, we find 23 percent of the descriptions are exhibit Non-American En-

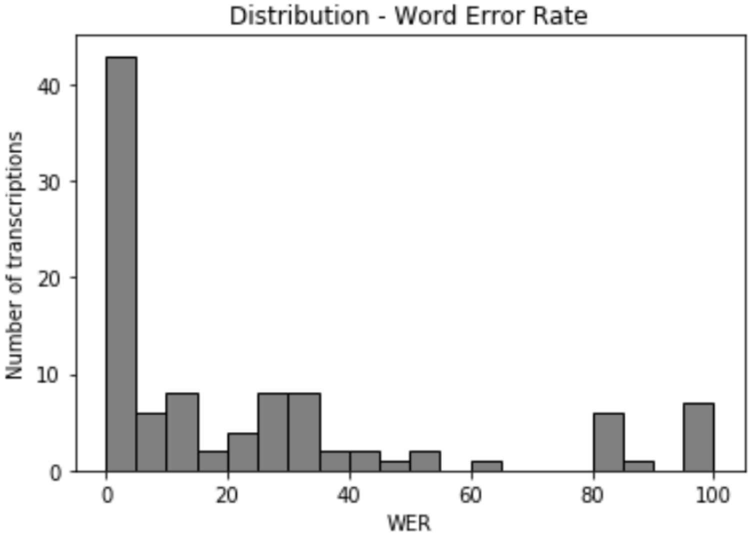


Figure 4.19: Distribution of Word Error Rate in 100 speech descriptions.

Accent Type	**Mean Rating**	**Mean BLEU**	**Mean WER**
American Accent	3.63	0.9	14.2
Non American Accent	2.086	0.47	54.3

Table 4.8: Variation in Transcription Quality with accent

glish. We observe that there is a substantial change in the quality of transcription because of the variation in accent. When we compare the quality of descriptions in American accent with non-American ones, we find that mean rating drop roughly by 42.5 percent, and the mean BLEU scores by 60 percent, and Word Error Rate jump by 282 percent. Table 4.8 shows the mean ratings, BLEU scores, and WER in the two cases. However, we witness no such significant quality distinctions when gender is considered.

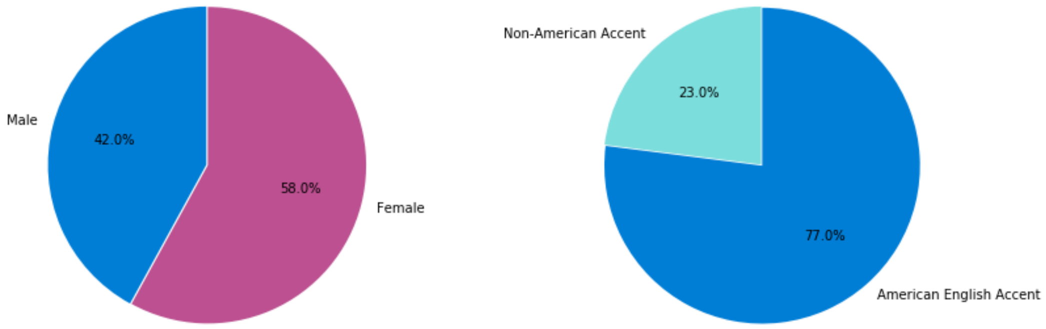


Figure 4.20: Gender and Accent variety in Mturk workers

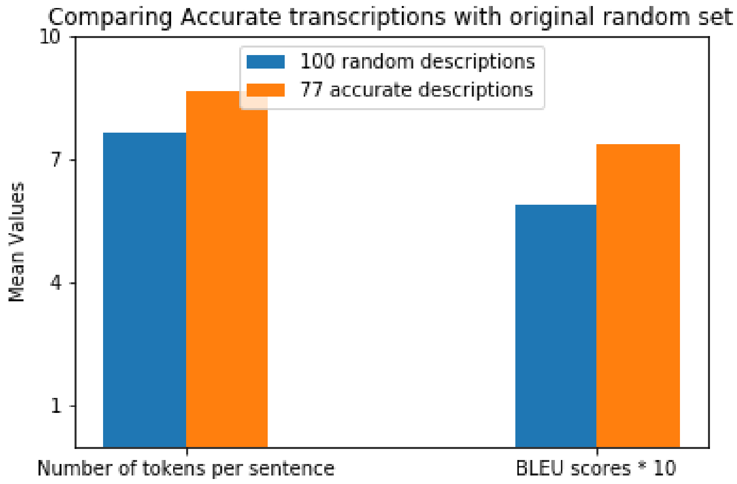


Figure 4.21: BLEU scores and Sentence length in accurate and original random set.

4.8 Analysis - Accurate Transcriptions

In this section, we use the same 100 randomly chosen speech descriptions. We study the behavior of high-rated (3 or 4) descriptions and compare them with the original random set. In this set, 77 descriptions are rated as 3 or 4, and 23 are rated either 1 or 2. BLEU scores show a rise from 0.79 to 0.95, and WER roughly declines

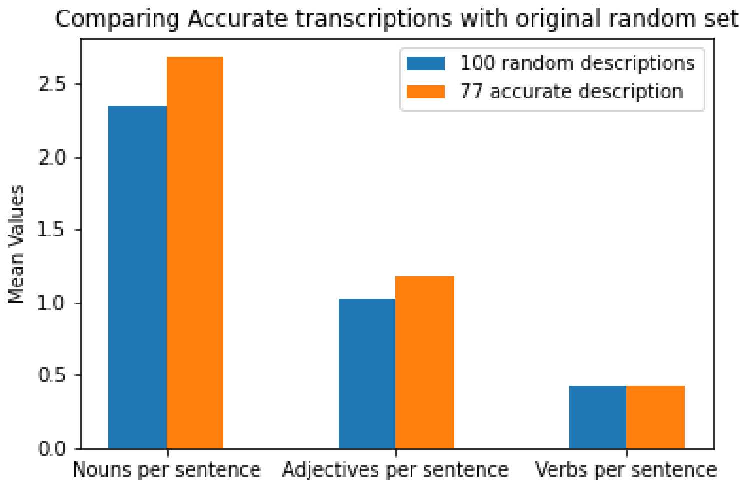


Figure 4.22: Noun, adjective and verb occurrence in accurate and original random set.

from 21 to 9.8 percent, as expected. We see a spike in average sentence length in the accurate transcriptions, when compared to the whole set. The average number of tokens in the accurate transcriptions is 8.66, whereas, in the original sample, the mean number of words per sentence turns out to be 7.64 (see figure 4.21). The accurate transcriptions also perform better when the occurrence of nouns and adjectives are considered (see figure 4.22). We use Stanford Part-of-Speech (POS) Tagger [7] to count the number of nouns, adjectives, and verbs in the descriptions. The presence of the average number of nouns per sentence shows a rise from 2.34 to 2.68, and the average adjective count increases from 1.01 to 1.18. Hence, the noun and adjective presence heighten up by around 17 and 14.5 percent, respectively.

[7] https://nlp.stanford.edu/software/tagger.shtml

4.9 Some observations from pilot study

As we discussed above, there are challenges involved when working with speech-to-text technologies. ASR systems are still not mature enough to deal with factors like accent, background noise, and microphone quality. In speech data collected in controlled environment, we found various gibberish translations. Some of such examples are:

- Cucumber: Ambar, Akumal, Humber, skookum bird, Green ombre, Goomba, Cucamonga, car bomber
- corn: coin
- fruit: flute
- cube: green tube

In the same study, we find certain descriptions captured that are completely irrelevant to the objects. The speech engine misinterpreted the descriptions uttered on the whole sentence level. Some of these are:

- Chimera assembly news in Scotland
- But one of our flight Carlos Agassi and Jackal country
- Is opium Malaysian cleaners kind of color
- A green coconut oil used for along with lunch or dinner in silence
- This is a banana and it is slightly Android and you can be laminated

Another observation is the impact of the experimenter's presence on the length of object descriptions. Interestingly, we find that subjects are more comfortable while performing user studies in the absence of any authoritative figure. In the absence of an experimenter, the descriptions collected are much more detailed and lengthy.

Chapter 5

Approach

In this section, we compare the performance of CBGLS (section 3.2) with speech and written data. We transcribe the speech descriptions using google speech API to feed the models. We start with a small pilot study to develop a baseline for our experiments. We further expand the study and use more advanced GLD dataset (section 4.2) with crowd-sourced language descriptions (sections 4.3, 4.4).

5.1 Implementation

In this section, we elaborate on the implementation details of CBGLS and the inputs accepted by the system. In the training Summary section, we illustrate how meaningful tokens are determined and further trained as classifiers. In the validation summary, we elaborate on the process of calculating F1-scores, Precision, and Recall to measure the system's performance.

5.1.1 Inputs to the System

5.1.1.1 Parameters

1. **Visual features:** We use RGB-D features of images for training. We elaborate the process of feature extraction in the experiment sections.
2. **Annotation file:** This file contains language descriptions of images. Each line

in the annotation file contains the name of an object, corresponding instance, and description. Below is an example of one such line where *coffee_mug* is an object, and *coffee_mug_2* is one of its instances.

coffee_mug/coffee_mug_2: *it is a black and yellow coffee mug with paw print on it*

3. **Category Types [*color, shape, object*]:** The system accepts at least one and a maximum of all three categories. To train classifiers as color type, RGB features are utilized, and depth features are used for shape category. Concatenated RGB-D features are used for training classifiers as the object category. The classifier corresponding to all the tokens are learned according to categories provided. The underlying idea behind training three classifiers per token was that a new token might be representing a color, shape, or object, and a robot with no prior knowledge would not intrinsically know which category the token should belong to.

5.1.1.2 Hyperparameters:

1. **MINIMUM TOKENS PER INSTANCE:** As the system follows a word-as-a-classifier approach, this parameter is used to decide the positive training images per token. To associate token to an image, we need strong association evidence. In a Mturk task, if a user describes an onion instance as a tomato by mistake, or speech-to-text generates some gibberish tokens, we do not want our system to relate such terms to the image. So, if the term appears in descriptions of an instance atleast *MIN_TOKEN_PER_INST* number of times,

then only the instance is considered as a positive data example for classifier corresponding to a particular token.

2. **THRESHOLD NUMBER OF POSITIVE INSTANCES:** This parameter regulates the number of tokens that the system learns as classifiers. To train a classifier (corresponding to a token), we need a fair amount of positive data images, without which system might perform poorly. To avoid such behavior, when a token maintains higher than *THRESHOLD_POSITIVE_INSTS* number of positive examples, then only it is qualified to be learned.
3. **NEGATIVE SAMPLE PORTION:** The system choose negative examples by learning a paragraph vector, for instance-descriptions from the annotation file and using cosine similarity to find the most distant paragraph vectors. This parameter regulates the number of negative instances per token-as-classifier. The system uses oversampling to balance the data before training; we find in our experiments that the system performance varies significantly when we tune this parameter.

5.1.2 Training Summary

In this section, we summarize the implementation details of CBGLS involved in training. In our work, we introduce some modifications in the system to enhance the performance with the GLD dataset, which we discuss in the experiments. See figure 5.1 for a diagram version of steps below.[1]

1. **Inputs** = RGB-D features, language annotations conf file [*tomato/tomato_1:*

[1] created using `app.diagrams.net`

language_description], and categories for training [*color, shape, object*].

2. **Fetch Tokens per instance:** Find all associated tokens corresponding to each instance using the input annotation file. Example: *coffee_mug/coffee_mug_1: mug, coffee, drink, milk, cup, white.*
3. **Find Instance set** Find all object classes and the corresponding instances from the annotation file [*coffee_mug: coffee_mug_1, coffee_mug_2,..*].
4. **Split the test – training instances:** Randomly select one instance as a test instance from each object class. For example, consider an object class (coffee_mug) with five instances. Each of the five instances has concatenated RGB-D features of multiple images. A random instance (out of five) is selected to be a test instance, and each instance possesses about 4-5 images. Therefore, approximately 4-5 images of each object are part of the test set. For one of the experiments, we modify the test-training strategy and incorporate 6-fold cross-validation.
5. **Positive and Negative training data per token:** Find positive and negative instances for tokens. Only training instances are considered.
   (a) **Finding Positives:** If a token appears >= *MIN_TOKEN_PER_INST* for an instance, then the instance is considered as a positive example for the corresponding token. If no positive instances are found, the token is not learned.
   (b) **Finding Negatives:** Steps to find negative instances for a token:
      i. For each pair of instances, get the distance between their descriptions using the doc2vec and cosine similarity metric. Store it in a

dictionary (let's call it negCandidateScores).

ii. Filter out pairs containing any test instances.

iii. Sort the remaining pairs by distance, and keep only two-third of them (ignoring rest).

iv. Now, choose N most dissimilar ones from the remaining.

$N = int(math.ceil(float(len(negCandidateScores.keys()))*$
$NEGATIVE\_SAMPLE\_PORTION))$

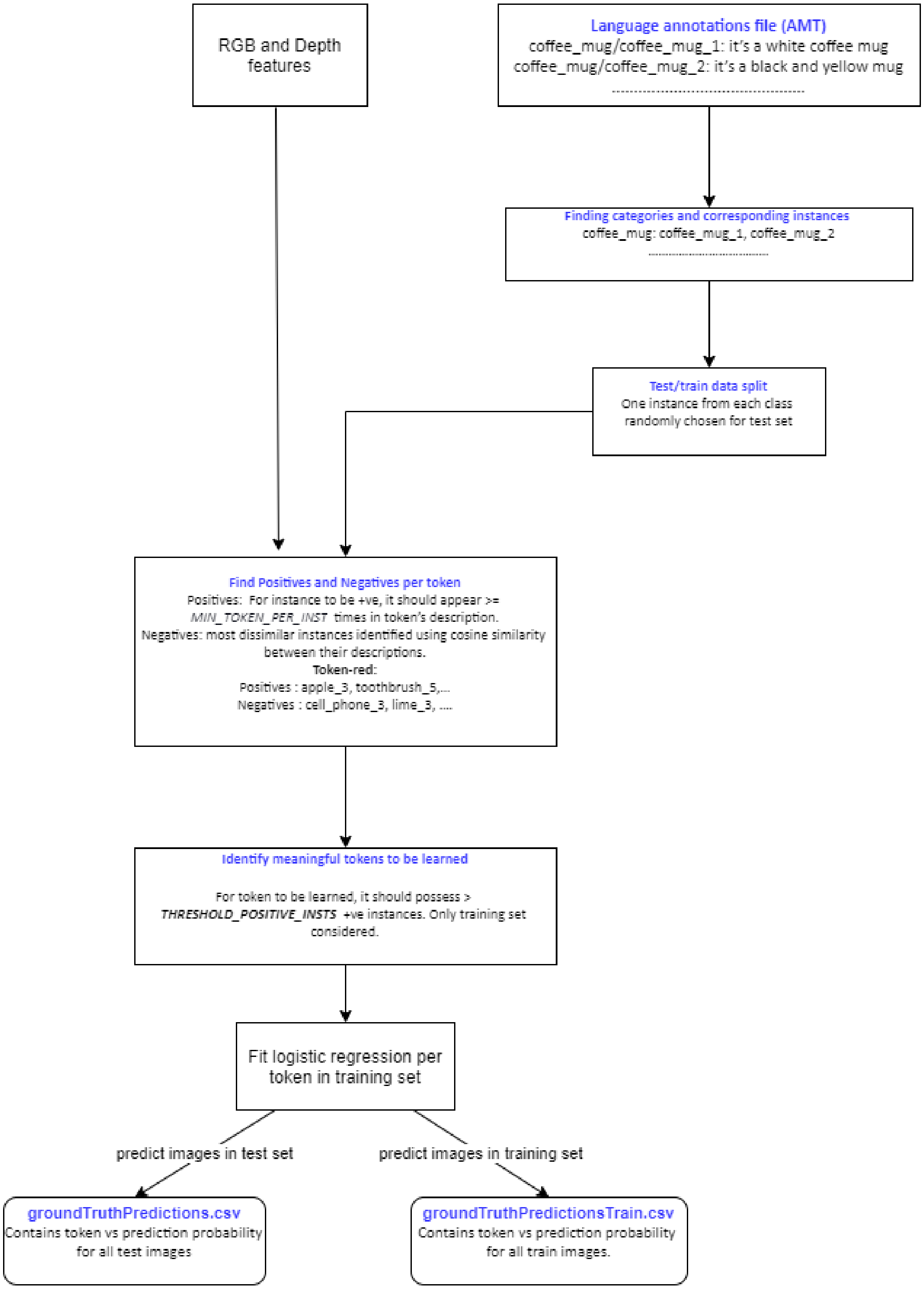
6. **Finding Meaningful Tokens:** The input is the conf file containing language annotations. The next step is to identify the meaningful tokens that the system should learn. For a token to be considered useful, it should possess more than *THRESHOLD_POSITIVE_INSTS* number of positive instances. In simple terms, the token should be present in the descriptions of more than 3 instances. The meaningful tokens are identified based on the training language corpus (descriptions of the training instances only).

7. **Training classifier per token:** Now, fit the Logistic Regression model on each token (from step 4) using identified positives and negative. Output:

   (a) **groundTruthPredictionTrain.csv:** contains probability results of applying the classifiers on training data.

   (b) **groundTruthPrediction.csv:** contains probability results of applying the classifiers on test data.

Figure 5.1: Training Code Flow Diagram for CBGLS


RGB and Depth features
Language annotations file (AMT)
coffee_mug/coffee_mug_1: it's a white coffee mug
coffee_mug/coffee_mug_2: it's a black and yellow mug
....
Finding categories and corresponding instances
coffee_mug: coffee_mug_1, coffee_mug_2
....
Test/train data split
One instance from each class randomly chosen for test set
Find Positives and Negatives per token
Positives: For instance to be +ve, it should appear >= MIN_TOKEN_PER_INST times in token's description.
Negatives: most dissimilar instances identified using cosine similarity between their descriptions.
Token-red:
Positives : apple_3, toothbrush_5,...
Negatives : cell_phone_3, lime_3, ...
Identify meaningful tokens to be learned
For token to be learned, it should possess > THRESHOLD_POSITIVE_INSTS +ve instances. Only training set considered.
Fit logistic regression per token in training set
predict images in test set
predict images in training set
groundTruthPredictions.csv
Contains token vs prediction probability for all test images
groundTruthPredictionsTrain.csv
Contains token vs prediction probability for all train images.

5.1.3 Validation Summary

In this section, we discuss the implementation details for the validation, the approach that system use to measure the performance by calculating F1 scores, Precision, and Recall. We adopt the changes presented by Kery [66] to select the true negative examples per token. See figure 5.2 for diagram version.[2]

1. **Inputs:** Annotation file and results from training (groundTruthPredictions.csv).
2. **Find Test Instances and Classifiers:** Fetch the tokens and prediction probabilities from groundTruthPredictions.csv. This file has the tokens, the ground truth, and the probability of each test instance when the token for that classifier is applied to it.
3. **Find True Positives and True Negative Instances per token learned in training:** Positives and negatives are identified in the same fashion as in training time, descriptions of **only test instances** are considered. Serialize the TP and TN in two separate files that are used in next step.
    (a) **Finding True Positives:** If a token appears >= *MIN_TOKEN_PER_INST* number of times for an instance, then the instance is considered as a positive example for the corresponding token.
    (b) **Finding True Negatives:** Steps to find negative instances for a token:
        i. For each pair of test instances, get the distance between their descriptions using the doc2vec and cosine similarity metric. Store it in a dictionary (let's call it negCandidateScores).

[2] created using `app.diagrams.net`

ii. Sort the remaining pairs by distance, and keep only two-third of them.

iii. Now, choose N most negative ones of the remaining.

$N = int(math.ceil(float(len(negCandidateScores.keys()))* NEGATIVE\_SAMPLE\_PORTION))$

4. **F1, Precision, Recall per token-as-classifier:** Calculate F1, Precision, recall and store it in CSV. For a token, select p positive and n negative instances from the test set.

   (a) p = number of positive instances.

   (b) n = number of negative instances

   (c) p is a randomly chosen integer out of [1,2,3] and n is decided accordingly.

      i. $p = random([1, 2, 3])$

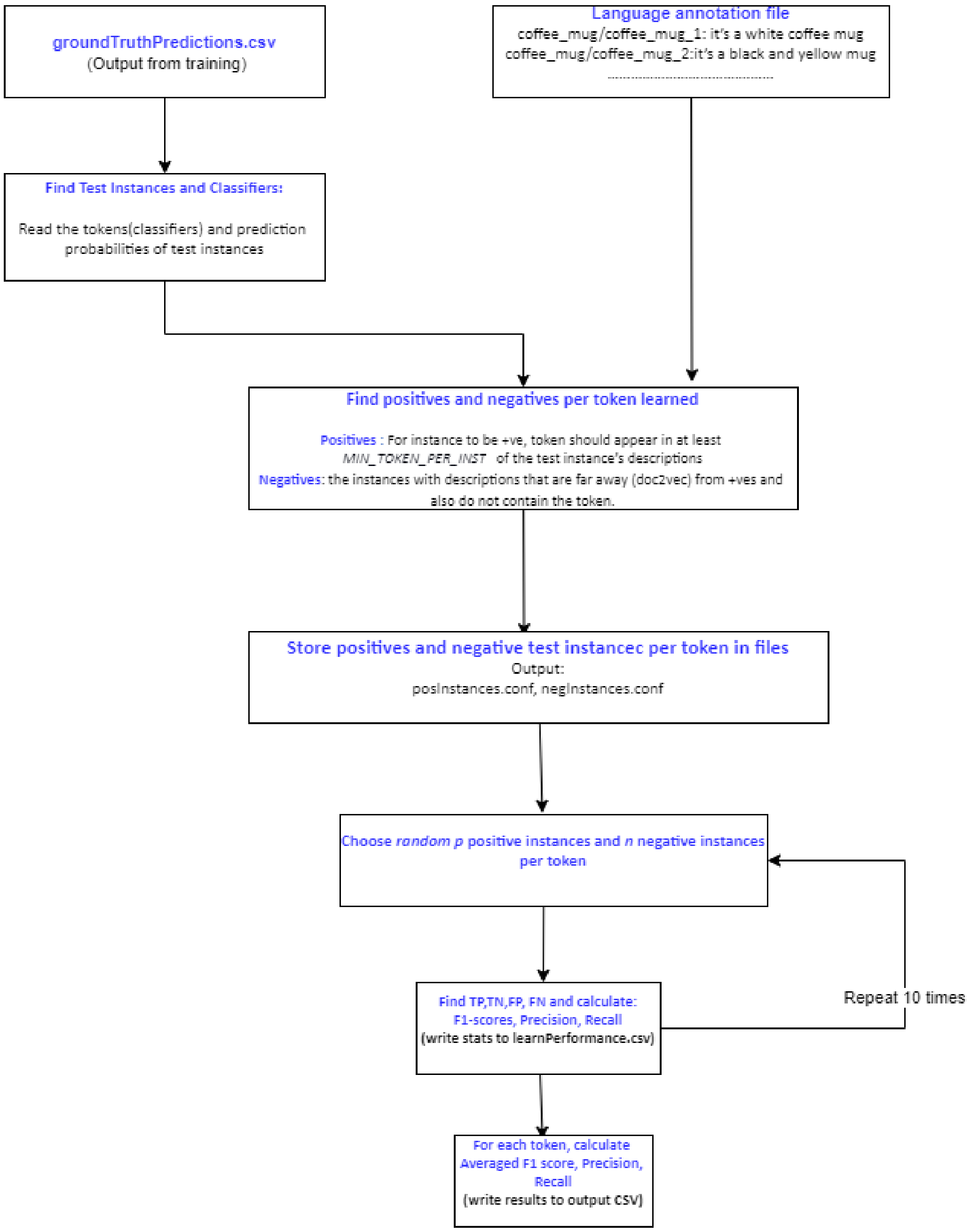
      ii. $n = random([4, 5, 6]) - p - 1$

5. For each token, repeat step-4 ten times. Further, evaluate the average statistics (10 runs) per token. Table 5.1 shows the sample output for token “bulb” after one validation run. Similarly, the F1 scores, Precision, and recall are generated for each of the tokens, and validation is executed ten times. The final stats are reported in experiments as the average of scores from all the individual tokens.

Test Object Images	**Ground Truth**	**Selected by Classifier**	**Precision**	**Recall**	**F1-Score**
lightbulb/lightbulb_2 _1, band_aid/band_aid_2_7, lightbulb/lightbulb_2_6, apple/apple_3_1, medicine_bottle/medicine_bottle_3_7	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6	1	1	1
apple/apple_3_7, medicine_bottle/medicine_bottle_3_1, lightbulb/lightbulb_2_6, coffee_mug/coffee_mug_5_1, band_aid/band_aid_2, lightbulb/lightbulb_2_11	lightbulb/lightbulb_2_6, lightbulb/lightbulb_2_11	coffee_mug/coffee_mug_5_1, lightbulb/lightbulb_2_6 lightbulb/lightbulb_2_11	0.67	1	0.8
lightbulb/lightbulb_2_1, apple/apple_3_13, lightbulb/lightbulb_2_16	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_16	lightbulb/lightbulb_2 _1, lightbulb/lightbulb_2_16	1	1	1
lightbulb/lightbulb_2_11, lightbulb/lightbulb_2_16 apple/apple_3_1, band_aid/band_aid_2_7	lightbulb/lightbulb_2_11, lightbulb/lightbulb_2_16	lightbulb/lightbulb_2_11 lightbulb/lightbulb_2_16	1	1	1
medicine_bottle/medicine_bottle_3_13, lightbulb/lightbulb_2 _1, apple/apple_3_19 (nan) band_aid/band_aid_2_1, coffee_mug/coffee_mug_5_16	lightbulb/lightbulb_2_1	lightbulb/lightbulb_2_1, coffee_mug/coffee_mug_5_16	0.5	1	0.67
lightbulb/lightbulb_2_11, apple/apple_3_7 lightbulb/lightbulb_2_6	lightbulb/lightbulb_2_11, lightbulb/lightbulb_2_6	lightbulb/lightbulb_2_11, lightbulb/lightbulb_2_6	1	1	1
lightbulb/lightbulb_2_1 apple/apple_3_13, lightbulb/lightbulb_2_6 lightbulb/lightbulb_2_16	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6. lightbulb/lightbulb_2_16	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6 lightbulb/lightbulb_2_16	1	1	1
medicine_bottle/medicine_bottle_3_19, lightbulb/lightbulb_2_11, coffee_mug/coffee_mug_5_16, band_aid/band_aid_2_13, bell_pepper/bell_pepper_4_19, apple/apple_3_19	lightbulb/lightbulb_2_11	coffee_mug/coffee_mug_5_16 lightbulb/lightbulb_2_11	0.5	1	0.67
lightbulb/lightbulb_2_6, apple/apple_3_13 lightbulb/lightbulb_2_1	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6	lightbulb/lightbulb_2_1, lightbulb/lightbulb_2_6	1	1	1
medicine_bottle/medicine_bottle_3_1, apple/apple_3_1, coffee_mug/coffee_mug_5_1, band_aid/band_aid_2_19, lightbulb/lightbulb_2_11, bell_pepper/bell_pepper_13	lightbulb/lightbulb_2_11	coffee_mug/coffee_mug_5_1 lightbulb/lightbulb_2_11	0.5	1	0.67
AVG			0.81	1	0.88

Table 5.1: Sample output for token "bulb" after one round of validation.

Figure 5.2: Validation Code Flow Diagram for CBGLS


groundTruthPredictions.csv
(Output from training)
Language annotation file
coffee_mug/coffee_mug_1: it's a white coffee mug
coffee_mug/coffee_mug_2:it's a black and yellow mug
....
Find Test Instances and Classifiers:
Read the tokens(classifiers) and prediction probabilities of test instances
Find positives and negatives per token learned
Positives : For instance to be +ve, token should appear in at least MIN_TOKEN_PER_INST of the test instance's descriptions
Negatives: the instances with descriptions that are far away (doc2vec) from +ves and also do not contain the token.
Store positives and negative test instancec per token in files
Output:
posInstances.conf, negInstances.conf
Choose random p positive instances and n negative instances per token
Find TP,TN,FP, FN and calculate:
F1-scores, Precision, Recall
(write stats to learnPerformance.csv)
Repeat 10 times
For each token, calculate Averaged F1 score, Precision, Recall
(write results to output CSV)

5.2 Pilot Study to establish baseline

The focus of my thesis is to examine the compatibility of audio data with the Grounded Learning systems. We start by collecting audio-transcribed data and compare the performance of the Category-based Grounded Learning system. We run small pilot studies to gather the spoken natural language descriptions and analyze the performance of CB-GLS with the spoken language data and textual descriptions collected using AMT presented by Pillai et al. [9]. The motive of this study is to establish a baseline before moving towards a more advanced dataset.

5.2.1 Data Corpus

For this study, we use the visual data corpus presented by Pillai et al. [9]. The images are collected using the kinect2 sensor, and crowd-sourced descriptions are gathered via Amazon Mechanical Turk. The dataset contains 72 objects divided into 18 categories. Each category has 3-4 instances, and each instance has 3-4 images taken from different angles (see figure 5.3). Language-wise, the dataset contains 3055 descriptions provided by AMT workers corresponding to the images shown. For this study, we use a subset of this data set containing a total of 4 object categories, where each category has 3-4 instances, and each instance has a single randomly chosen image. The reason to use only four object categories is to maximize the data per object to run a detailed analysis. The categories used for this pilot study are:

- Corn
- Cube

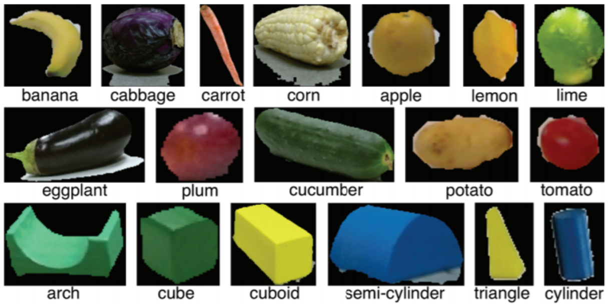


Figure 5.3: Sample RGB images taken using a kinect2 camera and presented to AMT workers. Reprinted from [9].

- Banana
- Cucumber

The subjects for this study are graduate students from the University of Maryland, Baltimore County, and the attempt is to collect data from people with different accents, keeping it gender-balanced. The user is presented with sixteen images to be described using a microphone. Each object category has four instances. At the end of the study, we collect around 29 descriptions corresponding to each instance, comprising a total of 464 speech transcribed descriptions.

5.2.2 Experiments

In this section, we evaluate the performance of CBGLS (described in section 3.2) with collected speech descriptions in the pilot study and AMT text descriptions. As this is a small study to define a baseline and no crowdsourcing platform is used, hence the spoken descriptions are far less than written data. To avoid bias, we randomly choose a subset of AMT textual data containing 460 descriptions. The number of descriptions per object is approximately the same in both cases. Below are the inputs to the system:

1. **RGB-D features:** To feed the model, we use RGB-D features extracted by Pillai et al. [9]. For each image, Kinect returns a typical color image and a 3-dimensional version where each point had a color and a location in 3-D space. Authors extracted the RGB features from the color channels and used kernel descriptors to extract shape and object features from the depth channel [74], [75].
2. **Annotation files:** We work with two types of annotation files in the following experiments. In first experiment, annotation file contains the stemmed object descriptions and the second one contains lemmatized descriptions.
3. **Hyperparameters:** For details about hyperparameters, please refer to section 5.1.1.2. In the following experiments, we utilize the same parameters as Kery [66] used, as both the works use the same image corpus.
   (a) MINIMUM TOKENS PER INSTANCE = 5
   (b) THRESHOLD NUMBER OF POSITIVE INSTANCES = 2
   (c) NEGATIVE SAMPLE PORTION = 0.25

We perform experiments with lemmatized as well as stemmed descriptions. We clean the descriptions to remove punctuation, stop-words, and convert them descriptions to lower-case. We utilize Snowball English Stemmer [51] for the purpose. Stemming is particularly helpful when users misspell terms while writing. For lemmatization, we use WordNet lemmatizer [53]. In both the experiments, learned classifiers worked better with textual data. However, the performance difference is not significant in the two cases. We initially hypothesized that F1 Scores for Spoken data would be around at least 80 percent of the scores obtained using written data. Results show that the spoken data performed well above that figure; hence, the CBGLS system shows promising results with the spoken descriptions.

5.3 Category-based Grounded Learning with GLD

In this section, we use more advanced visual dataset (described in section 4.2) and crowd-sourced language descriptions (sections 4.4, 4.3). We use Richards' approach to extract the visual features [73], [72]. The approach uses layered Artificial Neural Network to condense high-dimensional inputs to the meaningful representation of features.

5.3.1 Data Preprocessing

For audio data, we first transcribe all the descriptions using Google's speech-to-text API. For both speech-transcribed and text descriptions, we perform necessary data cleaning, transform them to lower-case and remove punctuation. For visual features, we concatenate all the extracted RGB-D features (NumPy arrays) of images per instance to feed the system. As each object instance approximately owns 3-4 images, each feature vector corresponding to an instance contains multiple RGBD vectors concatenated in a single file. We perform three experiments and compare the F1-scores, precision, and recall for learned object category classifiers with both datasets. First, we use raw descriptions with stop-words removed. The second experiment is with stemmed descriptions and no stop-words. In the third experiment, we use lemmatized descriptions. We also note the effect of stemming and lemmatization on learned object classifiers. We collected 4059 speech transcribed descriptions and 8250 textual descriptions via AMT. We randomly select 4059 textual descriptions in experiments performed to avoid bias.

5.3.2 Inputs to System

1. **RGB-D features**: We use Microsoft Kinect 3 RGB-D camera to collect color and depth images of the data corpus described in Section 4.2. To gather RGB-D features, we use Richard's approach [73], which is originally based on Eitel's work [72]. The method combines the benefits of transfer learned RGB models to both RGB and depth images for object recognition. Transfer learning is established as a useful technology in computer vision for leveraging rich labeled data in the source domain to build an accurate classifier for the target domain [79], [80]. Precisely, two convolutional neural networks are trained separately, one for each modality - color and depth. The wide-RESNET50 Convolutional networks [76] pre-trained on ImageNet are used for data from both sensor types. The next step is to combine the outputs of two CNNs to a fully connected fusion layer. Eitel et al. use a softmax function as the last layer to perform object classification task, removing this layer exhibit RGB-D visual features which are further paired with language descriptions and used

for grounded learning.

2. **Annotation Files**: In the following experiments, we clean the image descriptions collected via AMT and work with three different types of configuration files: raw descriptions with stop words removed, stemmed descriptions, and lemmatized descriptions. In total, we create six annotation files - three for text and three for speech-transcribed descriptions, one for each type.

3. **Hyperparameters**:

   (a) MINIMUM TOKENS PER INSTANCE = 5. We decide to keep this value as five after analyzing the language corpus. We collected five audio descriptions per image. As each instance approximately has four images, this makes the descriptions count to be around 20 per object instance. After observing the language data, we believe that the token should be present in at least one-fourth of the descriptions before an instance can be deemed as a positive example. If the token appears less than five times, we believe it may not be relevant enough for the corresponding instance, see figure 5.2 for an example. The less frequent tokens can arise due to the user's mistake or Google's speech errors.

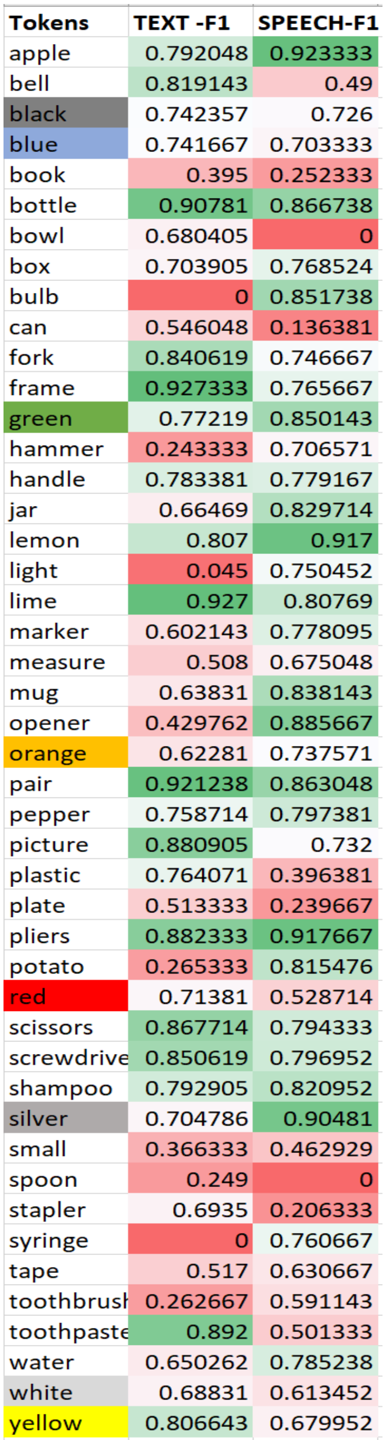
   (b) THRESHOLD NUMBER OF POSITIVE INSTANCES = 3. This threshold value is used to regulate the number of tokens learned by the system. In previous experiments, we used two as the threshold because of the nature of the dataset. GLD dataset has an increased number of instances per object, which helps to accommodate the tokens which are specific to a particular object class in current settings. When we reduce this threshold, the system attempts to learn certain tokens (for example: like, use), which are not informative.

(c) NEGATIVE SAMPLE PORTION = 0.1. We decide to keep this cutoff value as 10 percent, because, with the higher value, we find that some token classifiers with a few positive examples are overfitted and hence perform poorly with test instances.

coffee mug		cell phone	
Tokens	**Frequency**	**Tokens**	**Frequency**
black	14	smartphone	11
coffee	12	phone	8
mug	11	black	5
cup	6	Moriarty	1
nutria	1	space	1
Mississippi	1	kansas	1

Table 5.2: Few tokens and their frequency appeared in speech-transcribed descriptions of coffee-mug and cell-phone instance. Low-frequency words are either gibberish produced by Google speech-to-text, or, do not present information about the object.

NOTE: The results reported in the following experiments are the final medi-



Tokens	TEXT -F1	SPEECH-F1
apple	0.792048	0.923333
bell	0.819143	0.49
black	0.742357	0.726
blue	0.741667	0.703333
book	0.395	0.252333
bottle	0.90781	0.866738
bowl	0.680405	0
box	0.703905	0.768524
bulb	0	0.851738
can	0.546048	0.136381
fork	0.840619	0.746667
frame	0.927333	0.765667
green	0.77219	0.850143
hammer	0.243333	0.706571
handle	0.783381	0.779167
jar	0.66469	0.829714
lemon	0.807	0.917
light	0.045	0.750452
lime	0.927	0.80769
marker	0.602143	0.778095
measure	0.508	0.675048
mug	0.63831	0.838143
opener	0.429762	0.885667
orange	0.62281	0.737571
pair	0.921238	0.863048
pepper	0.758714	0.797381
picture	0.880905	0.732
plastic	0.764071	0.396381
plate	0.513333	0.239667
pliers	0.882333	0.917667
potato	0.265333	0.815476
red	0.71381	0.528714
scissors	0.867714	0.794333
screwdrive	0.850619	0.796952
shampoo	0.792905	0.820952
silver	0.704786	0.90481
small	0.366333	0.462929
spoon	0.249	0
stapler	0.6935	0.206333
syringe	0	0.760667
tape	0.517	0.630667
toothbrush	0.262667	0.591143
toothpaste	0.892	0.501333
water	0.650262	0.785238
white	0.68831	0.613452
yellow	0.806643	0.679952

Figure 5.4: Experiment 1: Common tokens and the F1 scores.

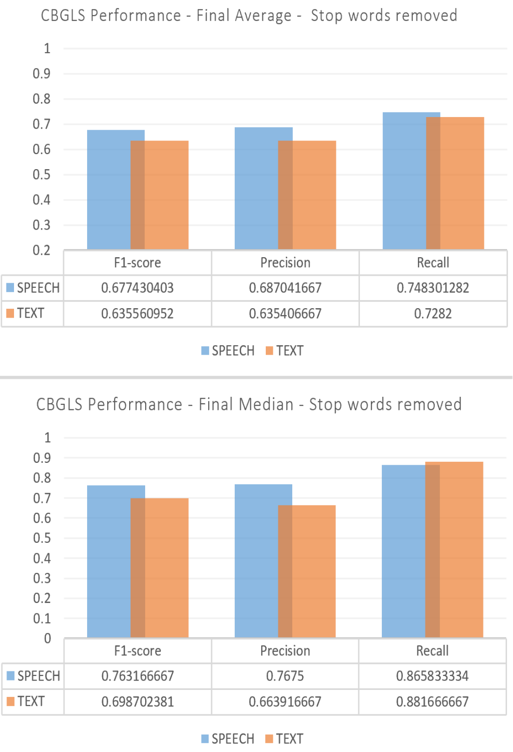


Figure 5.5: Experiment 1: F1, Precision, Recall.

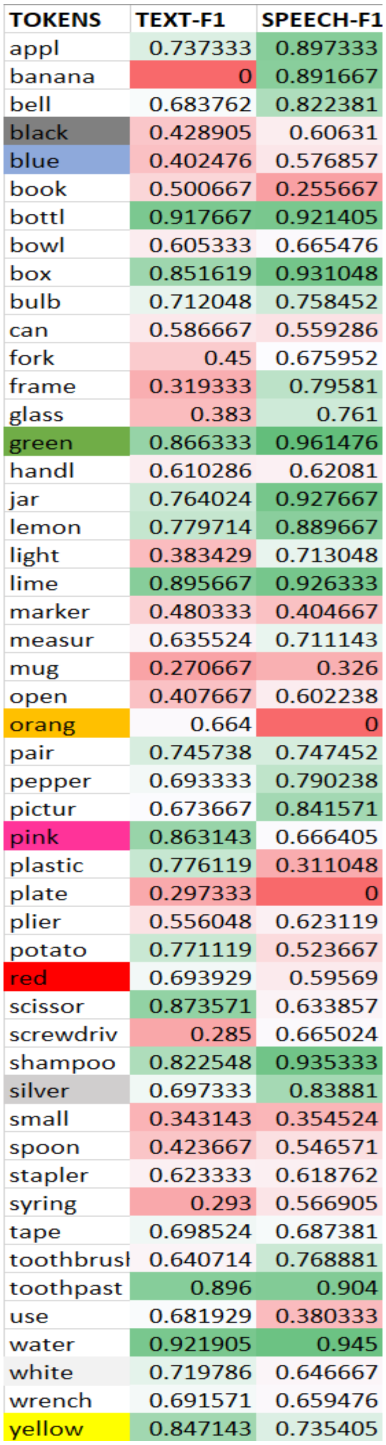
ans and means of all the averaged scores corresponding to the individual classifiers obtained after various validation runs. For some classifiers, precision is higher than recall, and vice versa. So, it is possible that the presented F1 scores do not lie in the range [Precision, Recall].

5.3.3 Experiment 1 - Cleaned Descriptions with stop-word removed

In this section, we perform necessary data cleaning and remove stop words from speech transcribed and textual descriptions. In both cases, the system learned almost the same number of tokens (50 in text, 52 in speech). The system learned 46 common tokens in the two cases, see table 5.4. The F1 scores of classifiers corresponding to color tokens are in similar lines (highlighted in 5.4). Figure 5.5 shows the differences in the average and median F1 scores in the two cases. We see that the system performs moderately better with speech transcribed descriptions. The final median of per-token average F1 scores is around 0.76 in speech and 0.70 with textual data. The hyperparameters to the system make sure to ignore the gibberish terms generated by google speech-to-text API. We see interesting behavior with descriptions corresponding to “band-aid” object instances. When textual data is fed, system learns two related tokens - *bandages* and *bandaids* both exhibiting approximately the same F1 score around 0.64. In case of speech data, the system learns two different tokens - *band* and *aids* with F1 scores 0.68 and 0.65 respectively. In the case of the text-based AMT task, people used *band aids* or *bandaids* to describe the object, but the use of hyphen was rare. In the case of speech data, the speech-to-text API always transcribed the term as band-aids, which is the correct way of writing it. As we remove punctuation, the system now learns two separate groundings, which convey a different meaning than what it is supposed to do. We observe similar behavior with other compound terms like toothbrush, toothpaste, and lightbulb.

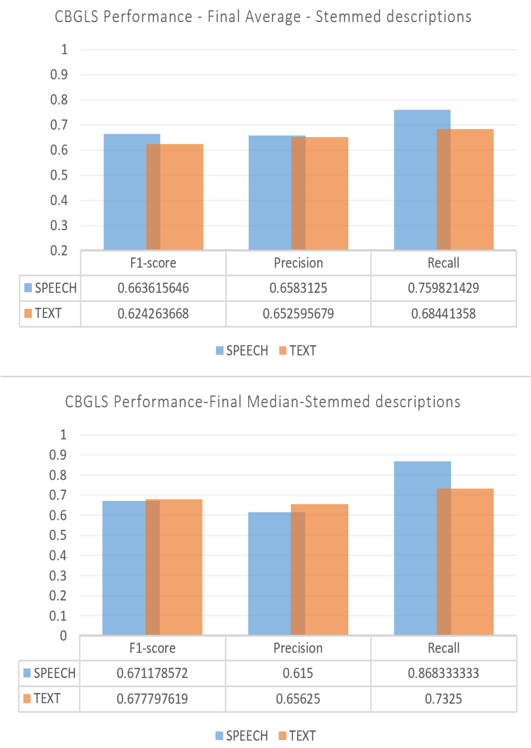
5.3.4 Experiment 2 - Effect of Stemming

In this experiment, we remove stop words and use stemmed annotation files. With textual data, the system learned 54 tokens and 56 in speech-transcribed data, among which 50 are common (see figure 5.6). Stemming sometimes helps an instance to meet threshold criteria in becoming a positive instance for a token. As expected, the number of tokens slightly increased when compared to experiment 1. As one can see in figure 5.7, there is no statistically significant F1-score difference in the two cases. When compared to Experiment 1, stemming has some negative impact on the object recognition task. Stemming can cause words to impact correctly or incorrectly. Incorrect stemming can undoubtedly cause problems, where tokens are stemmed that should not be, or words that should be stemmed are not. One example could be term - *opener* that was widely used by workers in descriptions of the "can opener" object, it is stemmed to *open.* Such behavior can sometimes increase positive cases for classifiers, which the system should not ideally include.



TOKENS	TEXT-F1	SPEECH-F1
appl	0.737333	0.897333
banana	0	0.891667
bell	0.683762	0.822381
black	0.428905	0.60631
blue	0.402476	0.576857
book	0.500667	0.255667
bottl	0.917667	0.921405
bowl	0.605333	0.665476
box	0.851619	0.931048
bulb	0.712048	0.758452
can	0.586667	0.559286
fork	0.45	0.675952
frame	0.319333	0.79581
glass	0.383	0.761
green	0.866333	0.961476
handl	0.610286	0.62081
jar	0.764024	0.927667
lemon	0.779714	0.889667
light	0.383429	0.713048
lime	0.895667	0.926333
marker	0.480333	0.404667
measur	0.635524	0.711143
mug	0.270667	0.326
open	0.407667	0.602238
orang	0.664	0
pair	0.745738	0.747452
pepper	0.693333	0.790238
pictur	0.673667	0.841571
pink	0.863143	0.666405
plastic	0.776119	0.311048
plate	0.297333	0
plier	0.556048	0.623119
potato	0.771119	0.523667
red	0.693929	0.59569
scissor	0.873571	0.633857
screwdriv	0.285	0.665024
shampoo	0.822548	0.935333
silver	0.697333	0.83881
small	0.343143	0.354524
spoon	0.423667	0.546571
stapler	0.623333	0.618762
syring	0.293	0.566905
tape	0.698524	0.687381
toothbrusl	0.640714	0.768881
toothpast	0.896	0.904
use	0.681929	0.380333
water	0.921905	0.945
white	0.719786	0.646667
wrench	0.691571	0.659476
yellow	0.847143	0.735405

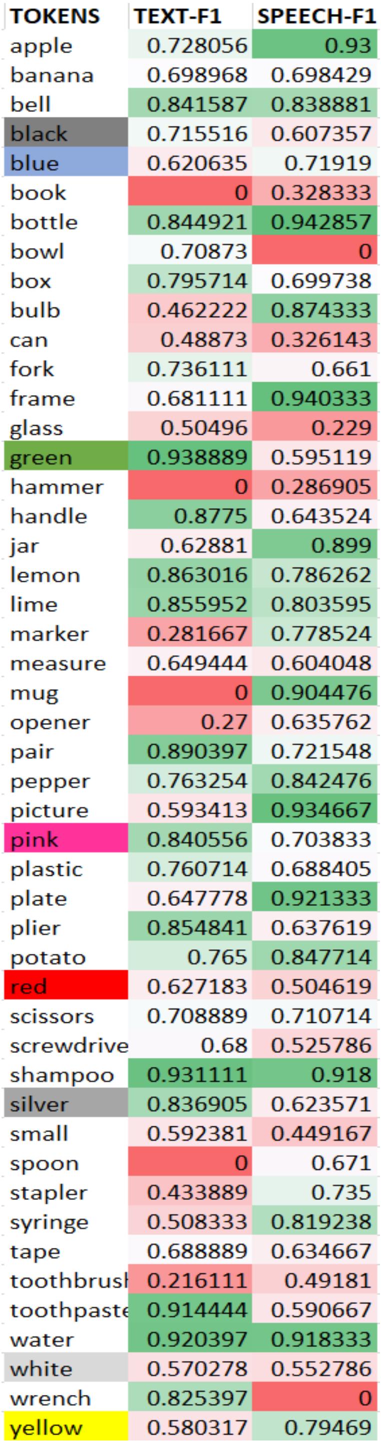
Figure 5.6: Experiment 2: Common tokens and their F1 scores.



	F1-score	Precision	Recall
SPEECH	0.663615646	0.6583125	0.759821429
TEXT	0.624263668	0.652595679	0.68441358

	F1-score	Precision	Recall
SPEECH	0.671178572	0.615	0.868333333
TEXT	0.677797619	0.65625	0.7325

Figure 5.7: Experiment 2: F1, Precision, Recall.



TOKENS	TEXT-F1	SPEECH-F1
apple	0.728056	0.93
banana	0.698968	0.698429
bell	0.841587	0.838881
black	0.715516	0.607357
blue	0.620635	0.71919
book	0	0.328333
bottle	0.844921	0.942857
bowl	0.70873	0
box	0.795714	0.699738
bulb	0.462222	0.874333
can	0.48873	0.326143
fork	0.736111	0.661
frame	0.681111	0.940333
glass	0.50496	0.229
green	0.938889	0.595119
hammer	0	0.286905
handle	0.8775	0.643524
jar	0.62881	0.899
lemon	0.863016	0.786262
lime	0.855952	0.803595
marker	0.281667	0.778524
measure	0.649444	0.604048
mug	0	0.904476
opener	0.27	0.635762
pair	0.890397	0.721548
pepper	0.763254	0.842476
picture	0.593413	0.934667
pink	0.840556	0.703833
plastic	0.760714	0.688405
plate	0.647778	0.921333
plier	0.854841	0.637619
potato	0.765	0.847714
red	0.627183	0.504619
scissors	0.708889	0.710714
screwdrive	0.68	0.525786
shampoo	0.931111	0.918
silver	0.836905	0.623571
small	0.592381	0.449167
spoon	0	0.671
stapler	0.433889	0.735
syringe	0.508333	0.819238
tape	0.688889	0.634667
toothbrus	0.216111	0.49181
toothpaste	0.914444	0.590667
water	0.920397	0.918333
white	0.570278	0.552786
wrench	0.825397	0
yellow	0.580317	0.79469

Figure 5.8: Experiment 3: Common tokens and the F1 scores.

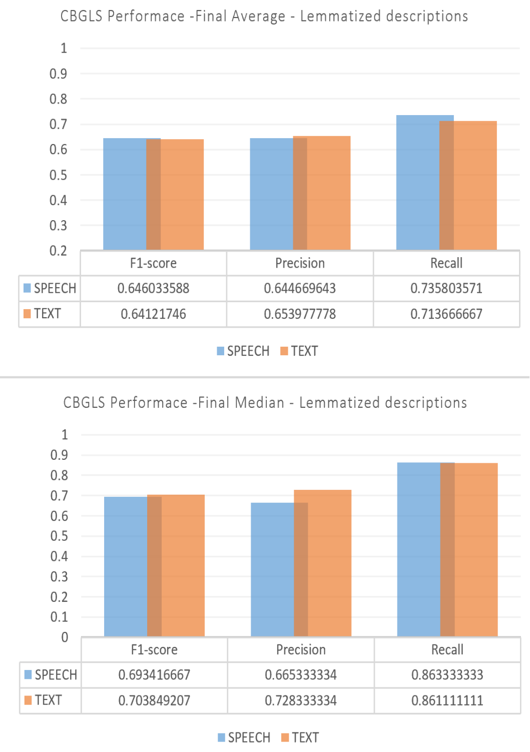


Figure 5.9: Experiment 3: F1, Precision, Recall.

5.3.5 Experiment 3 - Effect of Lemmatization

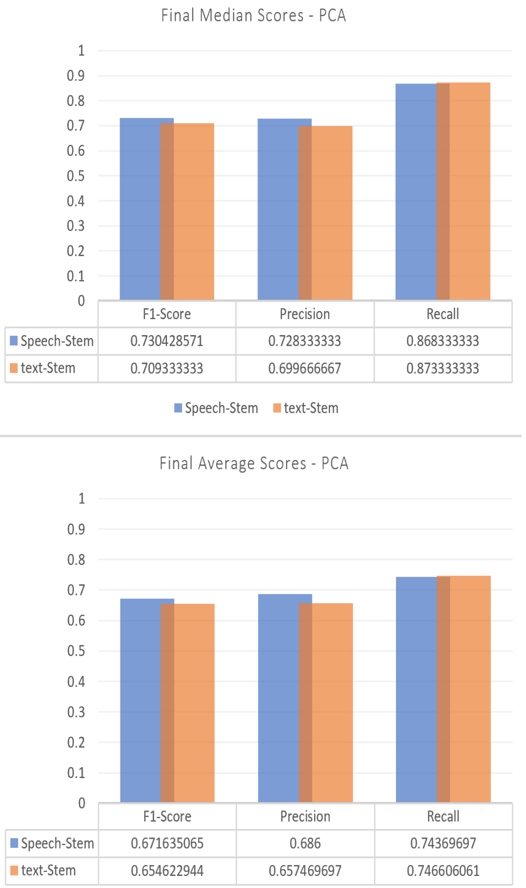
In this experiment, we remove stop words and use lemmatized annotation files. The system learned 50 tokens with textual descriptions and 56 with speech-transcribed annotations, among which 48 are common (refer to figure 5.8). Most of the tokens learned are common, and the final averaged F1 scores are approximately equal in the two cases. The final median and average F1 rating of all the learned classifiers are around 0.64 and 0.7, respectively, in both cases (see figure 5.9).

In the above experiments, we observe no significant difference among F1 scores of the system with textual and speech-transcribed descriptions. Most of the tokens learned are also common in all the experiments performed. Sometimes, we see a massive difference in the F1-score of the same token learned in different experiments. As the system chooses random test instances in each experiment, randomization plays a crucial role in causing such behavior. Also, we observe that some tokens exhibit zero F1-scores in the experiments. It implies that the corresponding classifier was unable to predict True Positive even once in multiple validation runs, and this odd behavior serves as a motivation for our next experiment.

5.3.6 Experiment 4 - Effect of PCA

Principal Component Analysis has been widely used for the representation of shape, appearance, and motion, and PCA representation is an established technique to tackle problems like object recognition, tracking, and detection [82, 83, 84, 85, 86]. It is a feature extraction technique that finds linear transformations of data that retain the maximal amount of variance to preserve as much statistical information as possible [81]. In the case of high dimensional and very sparse data, overfitting can become a severe problem, and PCA can help overcome it [87]. In the GLD dataset, each RGB-D image vector has a dimension of 4096; we use the PCA technique to transform the original feature set with the objective of preserving about 98 percent of variance exhibited by the data. We observe that 98 percent of variance can be achieved by looking at roughly 72 number of components on an average. Simply put, we add an extra PCA layer in the system before classification training of each token, which reduced the dimension of RGB-D image vectors from 4096 to 72 on an average per token.

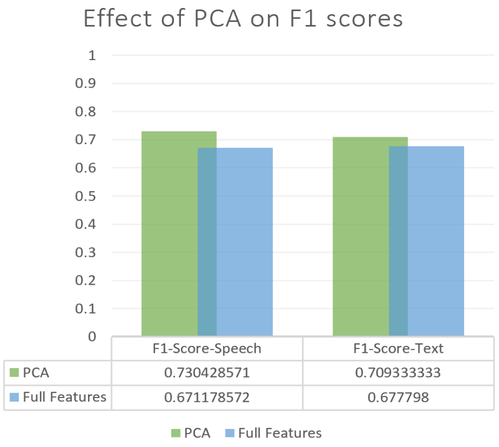
In this experiment, we use stemmed textual and speech-transcribed annotation files. Figure 5.10 shows the final median and average of F1-score, Precision, and Recall per token classifier. All the experiments described are conducted in a 64-bit Windows 10 machine with an Intel Core i-7 processor running at 1.99 GHz using 16 GB of RAM. In the current settings, we see a decrease in training time by around 23 percent. The total training time roughly reduced from 4 hrs 10 minutes to 3 hrs 12 minutes with the same input parameters. Another improvement is the number of zero-f1 classifiers. In previous experiments, we see some classifiers in almost every experiment exhibiting zero F1 scores, Precision and Recall (see figure 5.12). Mostly, such behavior is observed in token-classifiers having less number of positive instance examples. As per the current functioning of the system, the number of negative examples is generally much more than positives. The system uses oversampling to balance the data, but that didn't help. According to our understanding, such zero-f1 tokens appear because of overfitting. After adding the PCA layer, the system performed slightly better with the test instances resulting in more consistent F1 scores, as demonstrated in figure 5.11.



	F1-Score	Precision	Recall
🟦 Speech-Stem	0.730428571	0.728333333	0.868333333
🟧 text-Stem	0.709333333	0.699666667	0.873333333

	F1-Score	Precision	Recall
🟦 Speech-Stem	0.671635065	0.686	0.74369697
🟧 text-Stem	0.654622944	0.657469697	0.746606061

Figure 5.10: Experiment 4: Final Median and Averaged F1, Precision, Recall.



	F1-Score-Speech	F1-Score-Text
🟩 PCA	0.730428571	0.709333333
🟦 Full Features	0.671178572	0.677798

Figure 5.11: Change in final Median F1 scores after PCA

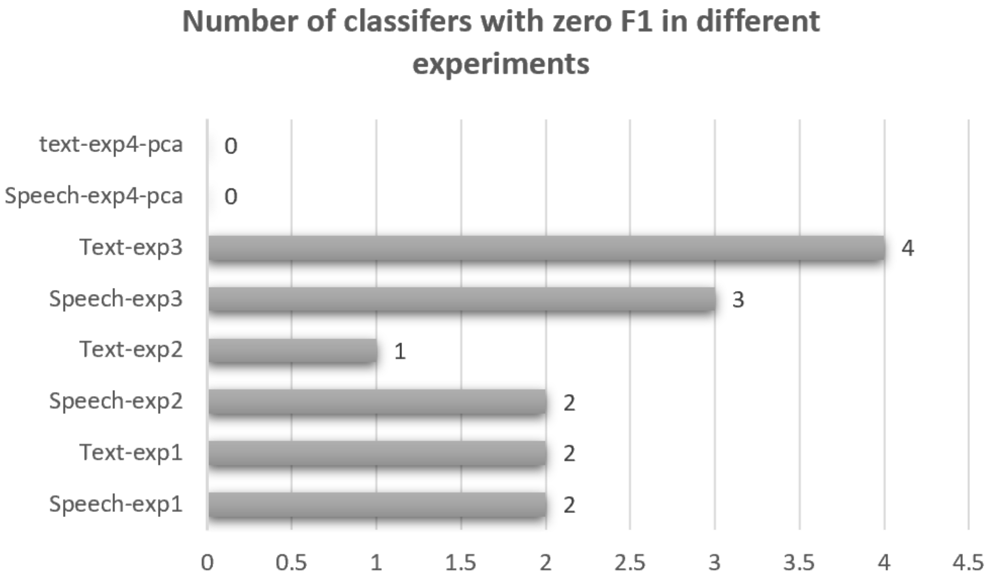


Figure 5.12: Number of classifiers exhibiting zero F1 scores. In experiment 4, no classifiers showed zero F1 scores.

5.3.7 Experiment 5 - K-fold Cross Validation for multi-class imbalanced data

In the above experiments, we observe that the resulting F1 scores are highly sensitive to positive and negative examples. As the system randomly chooses one test instance from each class, such randomization leads to high fluctuations in F1 scores even for the same tokens learned n different settings. If we train the system just once, the results may not be appropriate indicators of the performance. In this experiment, we attempt to mitigate such fluctuations caused by randomization by incorporating an k-fold cross-validation based strategy to handle our multi-class imbalanced data. In k-fold cross-validation, the initial sample is partitioned randomly into k same sized subsamples. Among the k subsamples, a single subsample is retained as the

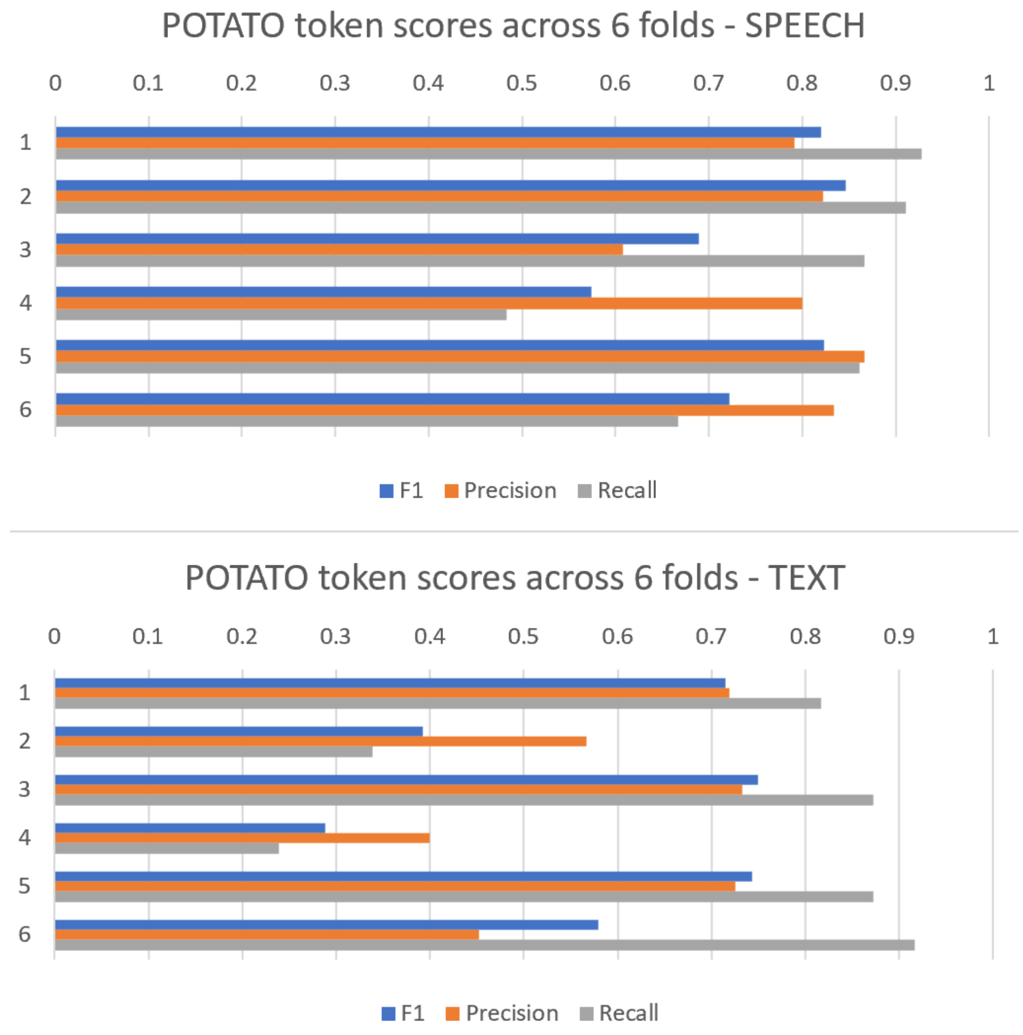


Figure 5.13: F1 scores, Precision and Recall of Potato token-as-classifier across different folds with Text and Speech annotations. x-axis represents numerical scores and y-axis denotes different folds in 6-cross validation. Note that the reported scores are averaged after multiple validation runs.

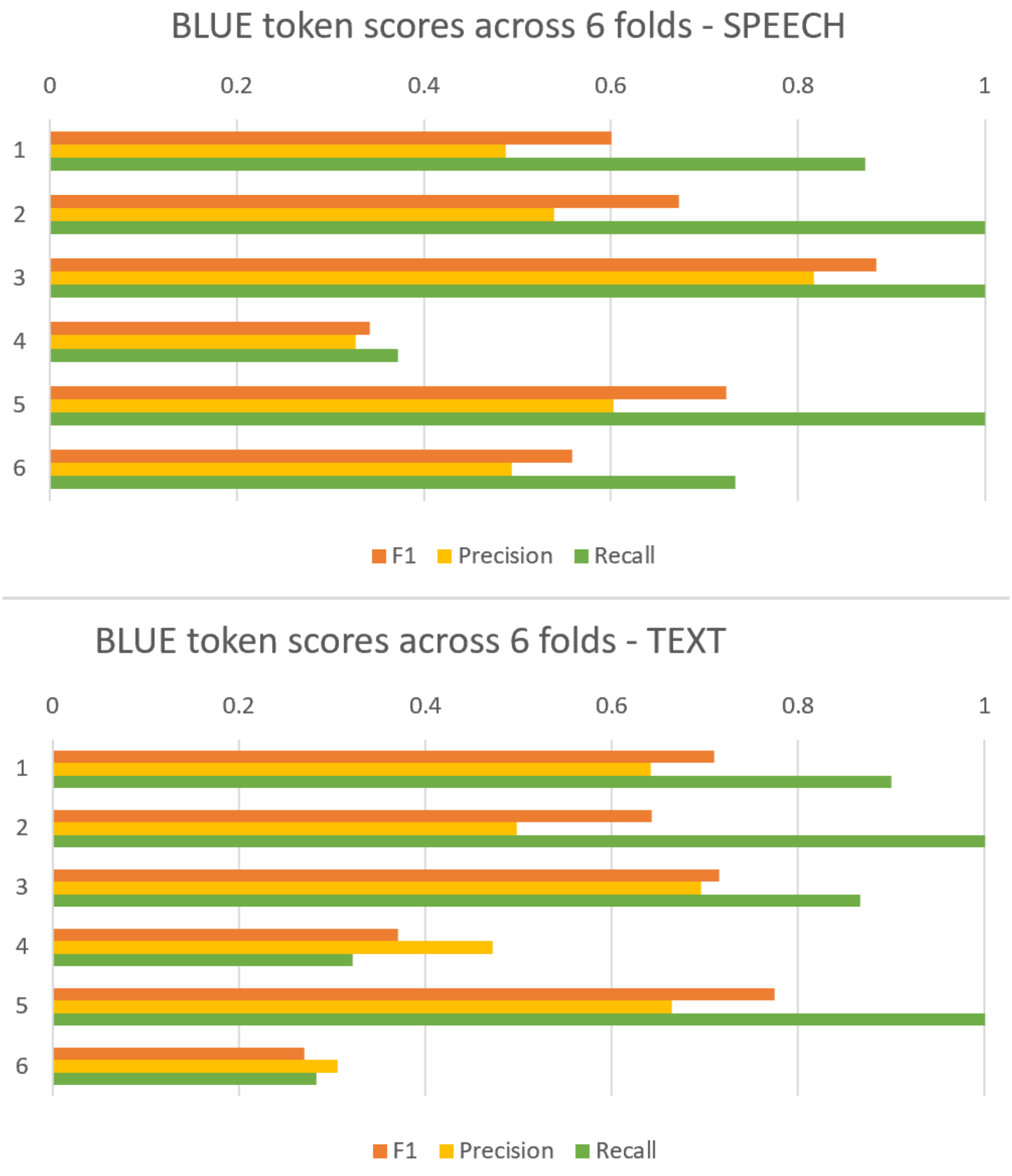


Figure 5.14: F1 scores, Precision and Recall of Blue token-as-classifier across different folds with Text and Speech annotations. x-axis represents numerical scores and y-axis denotes different folds in 6-cross validation. Note that the reported scores are averaged after multiple validation runs.

validation data for testing the model, and the remaining subsamples are then used for training. The cross-validation method is then repeated k times, with each of the k subsamples used exactly once as the testing data. The k results can then be averaged to generate a single estimation as the performance indicator. Originally, the system randomly chooses exactly one instance from each class as a test instance. In our work, we modify the system to make sure that each instance from every class is chosen as a test instance at least once. This way, every instance across all classes will appear in training as well as the testing set at least once; hence, mitigating the fluctuations in results caused by randomization. As shown in table 4.2 of Chapter 4, our dataset is imbalanced when the number of instances per class is concerned, as the number of instances per class differs from one to six. To accommodate our multi-class imbalanced dataset, we choose six-folds for cross-validation. One of the reasons for choosing k as six is because the maximum number of instances a class can have is six in our dataset. For each fold, a random instance from every class is chosen to be a part of the test set such that it has not been selected in previous folds. If all the instances are previously seen, then the random one is chosen from the original set of instances of that class. There exist two classes (pill cutter, tissue box) with only one instance each, which are handled separately. For these two classes, the corresponding single instance is chosen as part of the training for three folds, and as part of testing for the other half. We understand that these two examples are overrepresented in our cross-validation scheme since they appear in half of the folds. Yet, we make this tradeoff for the sake of resolving the instance imbalance issues that cause variance in precision and recall. To summarize, we enhance the traditional k-fold cross-validation approach to handle our multi-class imbalanced dataset with the following train-test split strategy.

1. For classes with exactly k number of instances, every instance is selected once randomly in each fold to be the part of the test set.
2. For classes with n instances ($1 < n < k$), every instance is selected once randomly in n folds. For the remaining $k - n$ folds, an instance is selected randomly from the original set of n instances to develop the test set.
3. For classes with only one instance, the single instance is selected as part of the training set for three folds and as part of the testing set for the remaining folds.

In this experiment, we train the system with six different training sets and validate on corresponding test sets, as discussed above. The system is then validated (as described in section 5.1.3) to calculate the averaged F1 scores per token-as-classifier. The final averaged F1 scores of all the individual classifiers are reported in table 5.3 across different training and validation datasets. Interestingly, when the single instance of "tissue box" appeared in training, the system learned a few extra tokens like "orange". Including such instances in training helped these tokens to meet the threshold to be determined meaningful by the system. We performed training with lemmatized as well as stemmed annotations, and we find that the averaged F1 scores across folds are slightly higher with speech-transcribed annotations, but the difference is marginal. We do find that the system is sensitive to the positives and negatives examples selected for training. Figures 5.13 and 5.14 shows the variation in F1, Precision and Recall of tokens "blue" and "potato" respectively

FOLD	Text-F1- lemmatized	Speech-F1-lemmatized	Text-F1-Stemmed	Speech-F1-Stemmed
1	0.622907509	0.635311471	0.60542034	0.60550558
2	0.618393592	0.659618764	0.62242210	0.61234263
3	0.630200659	0.608668763	0.63875180	0.62799088
4	0.579425338	0.618521458	0.56048941	0.614180375
5	0.578813933	0.595554113	0.5525	0.591252834
6	0.593548535	0.631324675	0.59747828	0.582365196
AVG	**0.605948206**	**0.623534914**	**0.596177**	**0.6056062**

Table 5.3: F1 scores with 6-fold cross-validation

across different folds. Hence k-fold cross-validation is a necessary step to perform a reasonable comparison. We find that more than 90 percent of tokens learned are the same in the two cases, and the final averaged scores are not significantly different.

5.3.8 Variation in F1 scores by Negative Sample Portion

In all the experiments conducted, we used 0.1 as the cutoff value of Negative Sample Portion. We decided on a 0.1 cutoff value after conducting several tests. We started with a 0.25, which is used by Kery [66]. With this cutoff value, F1 scores are not satisfactory; moreover, around 15-20 percent of the tokens learned exhibited zero F1 scores. Table 5.4 shows the number of such classifiers with different cutoff scores with speech and text annotations. The token-classifiers with a high number of positives performed satisfactorily, the ones with less positive examples suffered the most. We tried pilot experiments after increasing the cutoff value to 0.5 and

0.75, and the results were worse. Figure 5.15 demonstrates the variation in F1 scores as the cutoff varies. We achieved promising results with 0.1 cutoff with a very few number of zero-f1 tokens, which we finally able to eliminate in Experiment 4, as discussed in 5.3.6

NEGATIVE_SAMPLE_PORTION	**0.25**	**0.15**	**0.1**
Zero-F1s-Speech	8	5	2
Zero-F1s-Text	10	5	2

Table 5.4: Number of token-classifiers exhibiting zero F1 scores as the Negative Sample Portion value decrease. The values are from experiments conducted using annotation files with stop-words removed (as in Experiment 1).

5.4 Summary

We performed experiments with different annotation files, and we found no statistically notable difference in F1 scores, Precision, and Recall of the system. As we saw earlier, Google's speech-to-text produced a considerable number of low-quality transcriptions, especially with non-American English speakers, we initially hypothesized that the system would perform significantly better with text annotations. However, the speech annotations performed slightly better in almost all cases. We found that the crowd-sourced text descriptions are prone to typos and spelling mistakes that sometimes can affect the positive example for an instance. We found a few such occurrences like *ttothpaste, banannas, vitimans, towl, ren, ttowel*

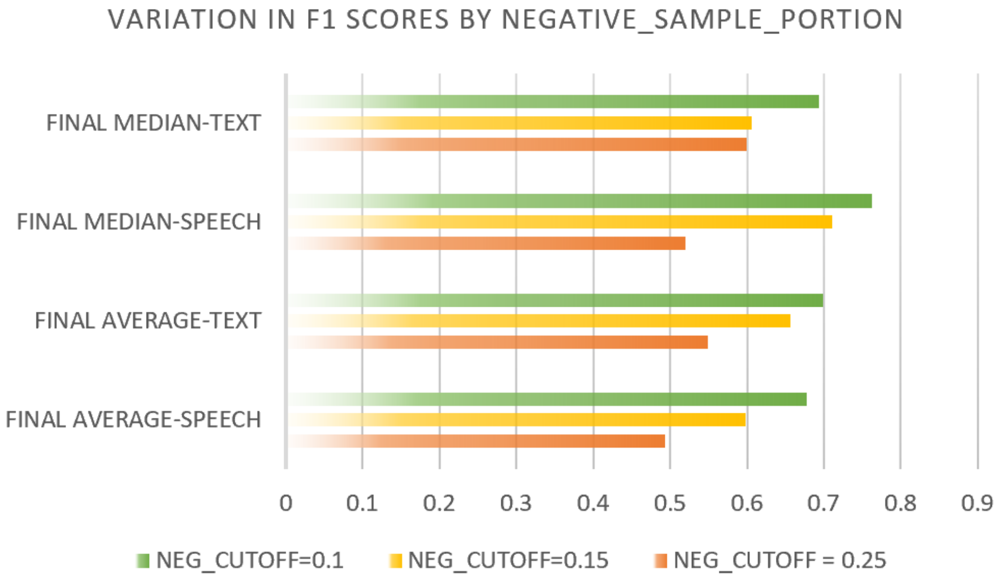


Figure 5.15: Variation in final median and average F1-scores as Negative_Sample_Portion value varies. The values are from experiments conducted using annotation files with stop-words removed (as in Experiment 1).

in textual descriptions. As the CBGLS follows the bag-of-words approach, robust hyperparameters can mitigate the adverse effect of such data. One positive point of using ASR systems is reducing the possibility of such typos to almost zero. We found that the F1-scores were very sensitive to the specific positive and negative instances chosen by the system. This is the reason why the same tokens' F1 scores varied significantly in different experiments. To mitigate such variance impact, we incorporate k-fold cross-validation strategy. We find that the averaged F1-scores across different folds are comparable in both the cases. Also, people tend to write the same word in different ways; this especially happened with compound and hyphenated words like *toothbrush, toothpaste, and lightbulb.* Some variants of *band-aids* like band aids, band-aids, and bandaids, adversely influencing the positive examples of the corresponding classifiers. With the advancements in NLP technologies, current ASR systems are mature enough to perform such mistakes. Therefore, such terms are either not learned with textual annotations or performed poorly.

Chapter 6

Conclusion and Future Work

In this work, we presented the extensive and robust Grounded Language Dataset (GLD) with RGB and depth point cloud images of 47 image classes and their crowd-sourced language descriptions in multiple formats - text, audio, and transcribed speech. We compare the written and spoken descriptions using parameters like sentence length and presence of nouns, verbs, and adjectives. Using the BLEU metric and Word Error Rate, we attempted to measure the accuracy of transcribed speech. We also note the role of accent in the accuracy of the transcriptions.

In this thesis, we have proposed adaptations of unsupervised grounded language acquisition system [9] to work with speech-transcribed English descriptions. After cleaning the descriptions, we perform experiments with text and speech data using three different types of annotation files - first, with stop words removed, second, with stemmed annotations and third, lemmatized descriptions. We discussed the effect of various hyperparameters on the performance of the system, and present the optimal hyperparameters to our dataset. Even with a significant amount of inaccurate speech transcriptions, there was no substantial difference in final F1 scores in the two cases. We also note how Principle Component Analysis can have a positive impact on learning tokens. To summarize, the primary contributions of our work include:

1. Presented multi-modal Grounded Learning dataset with language descriptions in various formats.[1]
2. Performed in-depth detailed comparative analysis of spoken and textual annotations.
3. Proposed adaptation of Grounded Learning System [9] with speech annotations to reach a wider audience.

We sought to examine the existing Grounded Learning system with speech transcribed annotations. The system use thresholds or value counts of tokens to evaluate the importance, and we found that these thresholds play a vital role in the system's performance, and can vary significantly across datasets. Hence to broaden the scope, some other techniques like part-of-speech tagging or entity recognition in language sentences can help identifying meaningful tokens more reliably.

Secondly, we found a significant amount of noisy data in annotations collected, especially the audio descriptions. Such behavior was prominent with short descriptions when the number of words in a sentence is below 4. If we can force the time limit requirements, we can collect better annotations, but such solutions come with their challenges. With the textual data, spell-checks can be an effective strategy.

The system we used trains the classifiers using Logistic Regression. Practically, any other classifier can also be used for the purpose. Using others like SVM would be a simple yet interesting next step. Moving to other, more sophisticated deep learning-based models (like [11]) would be another exciting step to gain more

[1] present in `https://github.com/iral-lab/UMBC_GLD`

insights. Due to the bag-of-words approach, the performance of the current system was not adversely affected because of gibberish transcriptions. Still, such data can impact the possible positive and negative examples for token classifiers up to an extent. To overcome such challenges, incorporating the "Confirmation Dialogue Strategy" in the data collection process could be an effective approach [36]. Dialogues are challenging to implement; maintaining the context of the conversation and speaking the right thing at the right time is challenging with robots. But, we believe this strategy has the potential to overcome the troublesome behavior of speech-to-text.

Appendix A

AMT Consent Form

Below we provide the consent form presented to the workers to collect the audio descriptions via Amazon Mechanical Turk.

Consent form for mechanical turk participants in "Teaching a robot by active learning." This consent form describes an experiment in which a participant provides feedback by describing objects using spoken natural language on the Mechanical Turk crowdsourcing platform.

Welcome to the research project on teaching robots using spoken natural language!

DESCRIPTION: *We are researchers at the University of Maryland, Baltimore County, doing a research study to develop interfaces, communication models, and educational tasks. Our goal is to learn how users can productively, comfortably, and efficiently teach a robot about real world objects and actions, in this case, by using spoken natural language.*

All data collected in this study are for research purposes only. Participants will be asked to provide natural language descriptions of objects, which will be spoken into a microphone and recorded. Descriptions are used to help a robot learn about objects in the world. Participants will not be asked additional questions beyond being asked for descriptions of objects. Participation should take approximately 30 seconds or *less per object.*

RISKS and BENEFITS: *The risks to your participation in this online study are those associated with basic computer tasks, including boredom, fatigue, mild stress, or breach of confidentiality. The benefit to you is the learning experience from participating in a human robot interaction research, and the compensation associated with completing each HIT. The benefit to society is the contribution to scientific knowledge.*

COMPENSATION: *6¢- 12¢ per question*

PLEASE NOTE: In this study, your voice (and possibly sound in your environment) will be recorded and stored.

CONFIDENTIALITY: *Your Mechanical Turk Worker ID will be used to distribute payment to you. Please be aware that your MTurk Worker ID can potentially be linked to information about you on your Amazon public profile page, depending on the settings you have for your Amazon profile, but we will not be accessing any personally identifying information about you that you may have put on your Amazon public profile page. Your Worker ID will never be included in publications or presentations, and will be used only to disburse payment and correlate your responses. Any reports and presentations about the findings from this study will not include your name or any other information that could identify you, with the possible exception of your voice. Study data may be stored and shared for use in future research studies. If we share data with other researchers doing studies, we will not include any personally identifiable information.*

SUBJECT'S RIGHTS: *Your participation is voluntary. You may stop partici-*

pating at any time by closing the browser window or the program, or by returning the HIT.

For additional questions about this research, you may contact:

Dr. Cynthia Matuszek, cmat@umbc.edu

ITE 325 B, CSEE , University of Maryland, Baltimore County

1000 Hilltop Circle, Baltimore, MD 21250

I have read and understand this consent form, and I understand that by working on these HITS, I am participating in this online research study. I am aware that I can contact the study author at any time for additional information, and that I may withdraw my participation at any time by returning or not working on Mechanical Turk HITs.

Bibliography

[1] Stevan Harnad. The symbol grounding problem. *Physica D: Nonlinear Phenomena* 42.1-3 (1990): 335-346.

[2] R.J. Mooney, Learning to connect Language and Perception. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, 2008.

[3] Tom Kwiatkowski, Luke Zettlemoyer, Sharon Goldwater, and Mark Steedman. Lexical generalization in CCG grammar induction for semantic parsing. In *Proceedings of the conference on empirical methods in natural language processing*, pp. 1512-1523. Association for Computational Linguistics, 2011.

[4] Luke S. Zettlemoyer, and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In *UAI* (2005).

[5] Cynthia Matuszek, Nicholas FitzGerald, Luke Zettlemoyer, Liefeng Bo, and Dieter Fox. A joint model of language and perception for grounded attribute learning. In *Proceedings of the 2012 International Conference on Machine Learning*, Edinburgh, Scotland, 2012.

[6] Jodi Forlizzi, DiSalvo Carl, and Gemperle Francine. Assistive robotics and an ecology of elders living independently in their homes. *Human–Computer Interaction* 19.1-2, 2004.

[7] Cynthia Matuszek. Grounded Language Learning: Where Robotics and NLP Meet. *IJCAI*, 2018.

[8] Pillai, Nisha, and Cynthia Matuszek. Identifying Negative Exemplars in Grounded Language Data Sets. *UMBC Student Collection*, 2017.

[9] Nisha Pillai and Cynthia Matuszek. Unsupervised selection of negative examples for grounded language learning. In proceedings of *Thirty-Second AAAI Conference on Artificial Intelligence*, New Orleans, USA, 2018.

[10] Nisha Pillai, Francis Ferraro, and Cynthia Matuszek. Optimal semantic distance for negative example selection in grounded language acquisition. *Robotics: Science and Systems Workshop on Models and Representations for Natural Human-Robot Communication*, 2018.

[11] Nisha Pillai, Francis Ferraro, and Cynthia Matuszek. Deep Learning for Category-Free Grounded Language Acquisition. *UMBC Student Collection*, 2019.

[12] Joseph L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin* 76.5 (1971): 378.

[13] 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 on association for computational linguistics.* Association for Computational Linguistics, 2002.

[14] Jim Miller. Bas Aarts et al., editors, The Hand-book of English Linguistics, chapter 28, pages 673–675. *Blackwell Publishing Ltd.*

[15] Ian Lane, et al. Tools for collecting speech corpora via Mechanical-Turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with amazon's mechanical turk.* Association for Computational Linguistics, 2010.

[16] Kong Aik Lee, Anthony Larcher, Guangsen Wang, Patrick Kenny, Niko Brümmer, David van Leeuwen, Hagai Aronowitz et al. The RedDots data collection for speaker recognition. In *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[17] Firman, Michael. RGBD datasets: Past, present and future. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2016.

[18] Jing Zhang, Wanqing Li, Philip O. Ogunbona, Pichao Wang, and Chang Tang. RGB-D-based action recognition datasets: A survey. *Pattern Recognition 60*, 2016.

[19] 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, Cham, 2014.

[20] Havard, William, Laurent Besacier, and Olivier Rosec. Speech-coco: 600k visually grounded spoken captions aligned to mscoco data set. In *ISCA Workshop on Grounding Language Understanding* (GLU2017), 2017.

[21] Micah Hodosh, Peter Young, and Julia Hockenmaier. Framing image description as a ranking task: Data, models and evaluation metrics. In *Journal of Artificial Intelligence Research*, pages 853-899, 2013.

[22] Peter Young, Alice Lai, Micah Hodosh, and Julia Hockenmaier. From image descriptions to visual denotations: New similarity metrics for semantic inference over event descriptions. *Transactions of the Association for Computational Linguistics 2*, pages 67-78, 2014.

[23] David Harwath, and James Glass. Deep multimodal semantic embeddings for speech and images. *2015 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, 2015.

[24] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision* 123, pages 32-73, 2017.

[25] Ravi Shekhar, Sandro Pezzelle, Yauhen Klimovich, Aurélie Herbelot, Moin Nabi, Enver Sangineto, and Raffaella Bernardi. FOIL it! Find One mismatch between Image and Language caption. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 255–265. Association for Computational Linguistics, 2017.

[26] Grzegorz Chrupała, Lieke Gelderloos, and Afra Alishahi. Representations of language in a model of visually grounded speech signal. In *Association for Computational Linguistics*, 2017.

[27] Judith Gaspers, Maximilian Panzner, Andre Lemme, Philipp Cimiano, Katharina J. Rohlfing, and Sebastian Wrede. A multimodal corpus for the evaluation of computational models for (grounded) language acquisition. In *Proceedings of the 5th Workshop on Cognitive Aspects of Computational Language Learning (CogACLL)*, pages 30-37. 2014.

[28] Yonatan Bisk, Daniel Marcu, and William Wong. Towards a dataset for human computer communication via grounded language acquisition. *Workshops at the Thirtieth AAAI Conference on Artificial Intelligence*, 2016.

[29] Yuya Yoshikawa, Yutaro Shigeto, and Akikazu Takeuchi. Stair captions: Constructing a large-scale japanese image caption dataset, 2017.

[30] Matt Lovett, Saleh Bajaba, Myra Lovett, and Marcia J. Simmering. Data quality from crowdsourced surveys: A mixed method inquiry into perceptions of

amazon's mechanical turk masters. *Applied Psychology* 67, no. 2, pages 339-366, 2018.

[31] Joseph K. Goodman, Cynthia E. Cryder, and Amar Cheema. Data collection in a flat world: The strengths and weaknesses of Mechanical Turk samples. *Journal of Behavioral Decision Making* 26, no. 3, pages 213-224, 2013

[32] Ian Lane, Alex Waibel, Matthias Eck, and Kay Rottmann. Tools for collecting speech corpora via Mechanical-Turk. In *Proceedings of the NAACL HLT 2010 workshop on creating speech and language data with amazon's mechanical turk*, pages 184-187, Association for Computational Linguistics, 2010.

[33] Adam J. Berinsky, Gregory A. Huber, and Gabriel S. Lenz. Evaluating online labor markets for experimental research: Amazon. com's Mechanical Turk. *Political analysis* 20, no. 3, pages 351-368, 2012.

[34] Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2018.

[35] Jesse Thomason, Jivko Sinapov, Maxwell Svetlik, Peter Stone, and Raymond J. Mooney. Learning Multi-Modal Grounded Linguistic Semantics by Playing I Spy. In *IJCAI*, pages 3477-3483, 2016.

[36] Jesse Thomason, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond J. Mooney. Improving grounded natural language understanding through human-robot dialog. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 6934-6941, IEEE, 2019.

[37] Thomason, Jesse, Aishwarya Padmakumar, Jivko Sinapov, Nick Walker, Yuqian Jiang, Harel Yedidsion, Justin Hart, Peter Stone, and Raymond Mooney. Jointly improving parsing and perception for natural language commands through human-robot dialog. *Journal of Artificial Intelligence Research* 67, pages: 327-374, 2020.

[38] Lanbo She, and Joyce Chai. Interactive learning of grounded verb semantics towards human-robot communication. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1634-1644, 2017.

[39] Joyce Y. Chai, Qiaozi Gao, Lanbo She, Shaohua Yang, Sari Saba-Sadiya, and Guangyue Xu. Language to Action: Towards Interactive Task Learning with Physical Agents. In *IJCAI*, pages 2-9, 2018.

[40] Douwe Kiela, and Stephen Clark. Multi-and cross-modal semantics beyond vision: Grounding in auditory perception. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 2015.

[41] Michael Fleischman, and Deb Roy. Grounded language modeling for automatic speech recognition of sports video. In *Proceedings of ACL-08: HLT*, 2008.

[42] Thomason, Jesse, Aishwarya Padmakumar, Jivko Sinapov, Justin Hart, Peter Stone, and Raymond J. Mooney. Opportunistic active learning for grounding natural language descriptions. In *Conference on Robot Learning*, pages 67-76, 2017.

[43] Chen Yu, and Dana H. Ballard. A multimodal learning interface for grounding spoken language in sensory perceptions. *ACM Transactions on Applied Perception (TAP)* 1.1, pages 57-80, 2004.

[44] Nisha Pillai, Karan K. Budhraja, and Cynthia Matuszek. Improving grounded language acquisition efficiency using interactive labeling. *UMBC Student Collection*, 2016.

[45] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nicholas Roy. Grounding verbs of motion in natural language commands to robots. In *Experimental robotics*, pages 31-47, Springer, Berlin, Heidelberg, 2014.

[46] Dilip Arumugam, Siddharth Karamcheti, Nakul Gopalan, Edward C. Williams, Mina Rhee, Lawson LS Wong, and Stefanie Tellex. Grounding natural language instructions to semantic goal representations for abstraction and generalization. *Autonomous Robots* 43, no. 2, pages 449-468, 2019.

[47] Muhannad Alomari, Paul Duckworth, David C. Hogg, and Anthony G. Cohn. Natural language acquisition and grounding for embodied robotic systems. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[48] Durk P. Kingma, Shakir Mohamed, Danilo Jimenez Rezende, and Max Welling. Semi-supervised learning with deep generative models. In *Advances in neural information processing systems*, pages 3581-3589, 2014.

[49] Tuomo Korenius, Jorma Laurikkala, Kalervo Järvelin, and Martti Juhola. Stemming and lemmatization in the clustering of finnish text documents. In

Proceedings of the thirteenth ACM international conference on Information and knowledge management, pages 625-633, 2004.

[50] Joël Plisson, Nada Lavrac, and Dunja Mladenic. A rule based approach to word lemmatization. In *Proceedings of IS*, vol. 3, pages 83-86, 2004.

[51] Martin F Porter. Snowball: A language for stemming algorithms, 2001.

[52] Martin F. Porter An algorithm for suffix stripping, *Program 14.3*, pages 130-137, 1980.

[53] George A Miller. WordNet: a lexical database for English. *Communications of the ACM* 38.11, pages 39-41, 1995.

[54] Jaap Kamps, Christof Monz, Maarten De Rijke, and Börkur Sigurbjörnsson. Language-dependent and language-independent approaches to cross-lingual text retrieval. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 152-165. Springer, Berlin, Heidelberg, 2003.

[55] Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 1972.

[56] Salton, Gerard, Edward A. Fox, and Harry Wu. Extended Boolean information retrieval. textitCommunications of the ACM 26, no. 11, pages 1022-1036, 1983

[57] Jayant Krishnamurthy, and Thomas Kollar. Jointly learning to parse and perceive: Connecting natural language to the physical world. *Transactions of the Association for Computational Linguistics*, pages: 193-206, 2013.

[58] Roy, Deb K. Learning visually grounded words and syntax for a scene description task. *Computer speech and language* 16.3-4, pages 353-385, 2002.

[59] David L. Chen, and Raymond J. Mooney. Learning to sportscast: a test of grounded language acquisition. In *Proceedings of the 25th international conference on Machine learning*, 2008.

[60] Yuk Wah Wong, and Raymond Mooney. Learning synchronous grammars for semantic parsing with lambda calculus. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 960-967, 2007.

[61] Wong, Yuk Wah, and Raymond J. Mooney. Learning for semantic parsing with statistical machine translation. In *Proceedings of the main conference on Human Language Technology Conference of the North American Chapter of the*

Association of Computational Linguistics, pages 439-446. Association for Computational Linguistics, 2006.

[62] Cynthia Matuszek, Dieter Fox, and Karl Koscher. Following directions using statistical machine translation. In *5th ACM/IEEE International Conference on Human-Robot Interaction (HRI)*, pages 251-258. IEEE, 2010.

[63] David L. Chen, and Raymond J. Mooney. Learning to interpret natural language navigation instructions from observations. *Twenty-Fifth AAAI Conference on Artificial Intelligence*, 2011.

[64] Cynthia Matuszek, Cynthia, Evan Herbst, Luke Zettlemoyer, and Dieter Fox. Learning to parse natural language commands to a robot control system. In *Experimental robotics*, pages 403-415, Springer, Heidelberg, 2013.

[65] Caroline Kery, Francis Ferraro, and Cynthia Matuszek. ¿ Es un plátano? exploring the application of a physically grounded language acquisition system to Spanish. *Proceedings of the Combined Workshop on Spatial Language Understanding (SpLU) and Grounded Communication for Robotics (RoboNLP)*, 2019.

[66] Caroline Kery. Esta Es Una Naranja Atractiva: Adventures in Adapting an English Language Grounding System to Non-English Data. Order No. 13878394 University of Maryland, Baltimore County, 2019. Ann Arbor: ProQuest. Web. 7 Apr. 2020.

[67] Haoyuan Gao, Junhua Mao, Jie Zhou, Zhiheng Huang, Lei Wang, and Wei Xu. Are you talking to a machine? dataset and methods for multilingual image question. In Advances in neural information processing systems, pp. 2296-2304. 2015.

[68] David L. Chen, Joohyun Kim, and Raymond J. Mooney. Training a multilingual sportscaster: Using perceptual context to learn language. *Journal of Artificial Intelligence Research*, pages 397-435, 2010.

[69] Ákos Kádár, Desmond Elliott, Marc-Alexandre Côte, Grzegorz Chrupala, and Afra Alishahi. ”Lessons Learned in Multilingual Grounded Language Learning.” *Association for Computational Linguistics*, 2018.

[70] Bo, and Cristian Sminchisescu. Efficient match kernel between sets of features for visual recognition. In *Advances in neural information processing systems*, pages 135-143, 2009.

[71] Joseph B. Kruskal, Mark Liberman, and J. Kruskal. The symmetric time-warping problem: from continuous to discrete, 1999.

[72] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. Multimodal deep learning for robust RGB-D object recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 681-687, IEEE, 2015.

[73] Luke Richards, and Cynthia Matuszek. Learning to understand non-categorical physical language for human robot interactions. From the *RSS Workshop on AI and its Alternatives in Assistive and Collaborative Robotics*, Vol. 6, 2019.

[74] Liefeng Bo, Kevin Lai, Xiaofeng Ren, and Dieter Fox. Object recognition with hierarchical kernel descriptors. In *CVPR* 2011, pages 1729-1736. IEEE, 2011.

[75] Kevin Lai, Liefeng Bo, Xiaofeng Ren, and Dieter Fox. RGB-D object recognition: Features, algorithms, and a large scale benchmark. In *Consumer Depth Cameras for Computer Vision*, pages 167-192, Springer, London, 2013.

[76] Zagoruyko, Sergey, and Nikos Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.

[77] Gopalakrishnan, Kasthurirangan, Siddhartha K. Khaitan, Alok Choudhary, and Ankit Agrawal. Deep Convolutional Neural Networks with transfer learning for computer vision-based data-driven pavement distress detection. *Construction and Building Materials* 157, pages 322-330, 2017.

[78] Chin-Yew Lin, and Franz Josef Och. Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Association for Computational Linguistics, 2004.

[79] Mingsheng Long, Jianmin Wang, Guiguang Ding, Jiaguang Sun, and Philip S. Yu. Transfer feature learning with joint distribution adaptation. In *Proceedings of the IEEE international conference on computer vision*, pages 2200-2207, 2013.

[80] Sinno Jialin Pan, and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering* 22, no. 10, pages 1345-1359, 2009.

[81] Ian T. Jolliffe, and Jorge Cadima. Principal component analysis: a review and recent developments. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 374, no. 2065, 2016.

[82] De la Torre, Fernando, and Michael J. Black. Robust principal component analysis for computer vision. In *Proceedings Eighth IEEE International Conference on Computer Vision*, vol. 1, pages 362-369, IEEE, 2001.

[83] Timothy F. Cootes, Gareth J. Edwards, and Christopher J. Taylor. Active appearance models. In *European conference on computer vision*, pages 484-498. Springer, Berlin, Heidelberg, 1998.

[84] Michael J. Black, Yaser Yacoob, Allan D. Jepson, and David J. Fleet. Learning parameterized models of image motion. In *Proceedings of IEEE computer society conference on Computer vision and pattern recognition*, ppages 561-567, IEEE, 1997.

[85] Baback Moghaddam, and Alex Pentland. Probabilistic visual learning for object detection. In *Proceedings of IEEE international conference on computer vision*, pages 786-793, IEEE, 1995.

[86] Hiroshi Murase, and Shree K. Nayar. Visual learning and recognition of 3-D objects from appearance. *International journal of computer vision* 14.1, 1995.

[87] Alexander Ilin, and Tapani Raiko. Practical approaches to principal component analysis in the presence of missing values. *Journal of Machine Learning Research* 11, pages 1957-2000, 2010.

[88] Yu-Dong Zhang, and Lenan Wu. An MR brain images classifier via principal component analysis and kernel support vector machine. *Progress In Electromagnetics Research* 130, pages 369-388, 2012.

[89] Dietrich Klakow, and Jochen Peters. Testing the correlation of word error rate and perplexity. *Speech Communication* 38, no. 1-2, pages: 19-28, 2002