ABSTRACT

Title of dissertation: ESTA ES UNA NARANJA ATRACTIVA: ADVENTURES IN ADAPTING AN ENGLISH LANGUAGE GROUNDING SYSTEM TO NON-ENGLISH DATA

Caroline Kery, Master of Science, 2019

Dissertation directed by: Dr. Cynthia Matuszek and Dr. Francis Ferraro
Department of Computer Science

In this thesis I describe a multilingual grounded language learning system adapted from an English-only system. This system learns the meaning of words used in crowd-sourced descriptions by grounding them in the physical representations of the objects they are describing. My work compares the performance of the system between languages from different perspectives to identify modifications necessary to attain equal performance, with the goal of enhancing the ability of robots to learn language from a more diverse range of people. I first analyze Spanish using translated English data, and then extend this analysis to a new corpus crowd-sourced Spanish language data. I then take the insights gained from this analysis, and repeat the experiment using Hindi. I find that with small modifications, the system is able to learn color, object, and shape words with comparable performance between languages.

ESTA ES UNA NARANJA ATRACTIVA: ADVENTURES IN ADAPTING AN ENGLISH LANGUAGE GROUNDING SYSTEM TO NON-ENGLISH DATA

by

Caroline Kery

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
2019

Advisory Committee:
Dr. Cynthia Matuszek, Chair/Advisor
Dr. Francis Ferraro, Co-Advisor
Dr. Tim Oates

Acknowledgments

This paper could not have been completed without the help of many people. I would like to thank my advisors Dr. Matuszek and Dr. Ferraro for their guidance over the past year. In particular I would like to thank them for all the time and effort they put into answering my questions, for constantly pulling me out of linguistic rabbit holes, and for giving me really good advice (even when it took me a while to follow it). I would also like to thank Nisha Pillai, who not only provided the system I based my research off of, but was always willing to patiently sit with me and answer my questions. In addition, my analysis and collection of Hindi data could not have been completed without the work and insights of Rishabh Sachdeva. The rest of the students of the IRAL lab were also very helpful and supportive. I thank my family and friends for their support, and their willingness to listen to me try and explain my research. Finally, I would like to thank the many more people who helped me get to this point that are not listed here.

Table of Contents

List of Figures

List of Abbreviations

NLP	Natural Language Processing
GT	Google Translate
HIT	Human Intelligence Tasks
IDF	Inverse Document Frequency
GLS	Grounded Language System

Chapter 1: Introduction

With widespread use of products like Roombas, the Amazon Echo, and drones, robots are becoming commonplace in the homes of regular people. We can see a future where robots are integrated into homes to provide assistance in many ways. This could be especially beneficial to elders and people with disabilities, where having someone to help with basic tasks could be what allows them to live independently [1].

Natural language is an intuitive way for human users to communicate with robotic assistants [2]. In order for this communication to happen, robots need to first learn what language means, and how it maps to their surroundings. Grounded Language Acquisition is the concept of learning language by tying natural language inputs to concrete things one can perceive. This field of study looks to train language and perceptual skills simultaneously [3], in order to gain a better understanding of both. This is a concept that crops up often in human language learning. A child learns what the word "dog" means by encountering many examples of dogs and building an association between the word and the things they perceived. In robotics, work in this field is critical for building robots that can learn about their environments from the people around them.

For such a system to truly be useful for the average user, it is not enough to merely train a robot how to recognize everyday objects and actions in a lab. Much like toddlers who grow up in a family and surrounding culture, a service robot should be ideally able to learn the acronyms, nicknames, and other informal language that happens naturally in human interaction. It logically follows that a truly well-designed system should not only be able to handle vocabulary differences between users but also users that speak various languages. There are thousands of official languages spoken around the world, and many more dialects. In the United States alone, around 21 percent of residents speak a non-English language as their primary language at home [4].

While there has been past work to apply grounded language learning systems to multiple languages [5, 6] to my knowledge there have been few efforts in the space of non-English robot language learning where comprehensive analysis was done to diagnose differences in performance between languages and work to mitigate these differences. Grounded Language Acquisition takes many of its roots from Natural Language Processing, which in the past has had an unfortunate habit of focusing on English-centric methods. This often leads to systems that perform very well in English and "well enough" in other languages.

In this thesis, I take an existing grounded language acquisition system [7, 8] originally designed for English and examine what adaptations are necessary for it to perform equally well for two other languages. I start with the simpler task of extending the system to Spanish data, and analyze the performance of translated English-Spanish, and novel

Spanish data. I then follow this same formula with Hindi, a language that has both a different script and much different morphology to English. Through this analysis, I explore places where linguistic differences can impact performance, and discuss the overall transferability of the system.

Chapter 2: Background

2.1 Related Works

In this section, I describe relevant previous works in grounded language acquisition and multilingual natural language processing.

2.1.1 Grounded Language Acquisition

Language grounding can be done in many ways. There is a significant community within computer vision that works on object recognition with the help of captions [9, 10]. These efforts ground objects found in images with words and relations stated in the captions. A multilingual example of this by Gella, used images as pivots between English and German image descriptions. This thesis has a similar task of mapping language to images, but does so on a token level, and does not attempt to combine information between the language corpora. In addition, the image data I used includes depth information, as I am simulating the visual percepts of a robot. It must be noted that this differs from other works that use additional products of robotic percepts like video data, eye tracking, and other forms of gesture recognition [5, 6, 11, 12]. In the robotics space, many works tie language grounding to enable actions like pathfinding [2], household tasks [6], and building [13]. While performing practical tasks is the eventual goal of the grounded language system, the current system focuses on the first step: building representations of objects and how they are described (nouns and adjectives).

There are a few examples of language grounding in multiple languages [5, 6]. Several works tested their system in a language besides English and presented the results for both. While this showed that their systems could handle multiple languages, none provided an in-depth analysis into the differences in performance for their systems, or extrapolated past the two languages. My work seeks to examine and identify causes of differences in performance. While this thesis mainly concentrates on extending to Spanish and Hindi, it seeks to do so in an organized way that can be easily generalized to additional languages.

2.1.2 Multilingual Natural Language Processing

There is a strong multilingual community in the broader field of NLP working in many different aspects, such as machine translation [14] or multilingual question answering [10]. Some works dive deep into specific language pairs to evaluate how differences between the languages complicate translation [15–17]. Analysis such as these helped to shape my analysis when comparing the English, Spanish, and Hindi data performance.

There are quite a few examples in literature of taking a system designed for English and adapting it for multilingual use [18–21]. Sometimes this involves manually recreating aspects of the system to match the rules of the other language [19]. Other projects look to make an English system "language agnostic" (not biased towards any one language) by editing parts of the preprocessing algorithm [18, 22]. The first method introduces a lot of additional complications such as manual rule-building, so it may seem attractive to make a system that is completely language-blind. The problem with this is that even generalized preprocessing techniques are often still biased towards languages with English-like structures [23], and in avoiding specifying anything about the language one can miss out on common properties within language families that could increase performance. For this paper, I strive to find common ground between making my system as generalized as possible and taking specific linguistic structures into account if necessary.

One significant difference between my research and many works in grounded language acquisition is that the system is entirely trained off of noisy short descriptions collected without filtering. This has very different characteristics from the more common corpora built off of newswire and other forms of well-written text (a very common one is multilingual Wikipedia), or data that has been placed into structures like trees [24]. MY data is prone to errors in grammar and misspellings; in this regard, my data is most like that of works that use Twitter data [25]. However, in contrast to [25], the system I am using only uses token extraction to find the relevant images to extract features from, rather than extracting all features from just the language.

2.2 The Original Grounded Language System

In this thesis, instead of building a new grounded language system, I chose to start with an existing system presented by Pillai et al. [8] and expanded on in [26], which I will refer to throughout this thesis as the GLS (grounded language system). This system attempted to learn physical groundings of colors, shapes, and objects by tying color and depth data derived from images of various items with natural language descriptions of the images. The validity of these groundings was then tested with the downstream task of object recognition. My work concentrates on exploring how well this existing system can be extended to multiple languages, and identify potential complications that need to be considered when doing so.

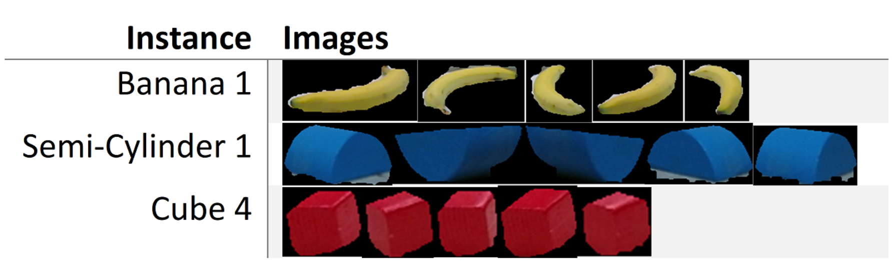

Figure 2.1: Examples of images of objects in the original dataset.

2.2.1 Data

Pillai et al. used a Kinect depth camera to collect images of fruit, vegetables, and children's blocks of various shapes (see figure 2.1 for examples). There were a total of 18 object types, with four instances of each object. Each instance had around five images taken using the depth camera. For each of these images, RGB and HMP-extracted kernel descriptors [27, 28] were extracted from the RGB-D data. The authors then collected descriptions of these images in English using Amazon Mechanical Turk. About 85 descriptions were collected for each instance, for a total corpus of about six thousand descriptions. As I discuss in section 2.4, my own data collection process replicated this setup.

2.2.2 Grounding System

The GLS learned classifiers for meaningful tokens in the categories of color, shape, and object in an unsupervised setting. The system used the Mechanical Turk descriptions to identify which images were positive or negative examples of each token. It used a "bag of words" approach, where it counted how many times a token appeared in any description of each instance, without reference to the context of its use. Images that were described with a particular token often were assumed to be examples of that token. To find negative examples of tokens, the GLS used document similarity metrics to compare the descriptions for instances in vector space [26]. The instances that were sufficiently far away in vector space from the identified positive instances for a token that had also never been described with that token were then chosen as negative examples of that token. For example, suppose the system were finding positive and negative instances for the token "orange." A positive instance identified might be "carrot 4." In the document vector space, the instances with the descriptions most different from "carrot 4" would be "arch 1" and "cuboid 4," while instances like "tomato 2" and "cucumber 3" are closer but still different enough to possibly qualify as negative examples of the token "orange."

Tokens that did not have any negative examples or had fewer than three positive examples were thrown away, with the assumption that there was not enough data to learn a classifier. The final classifiers were scored by how well they could identify new positive and negative examples of each token. This was done by presenting held-out images that were either positive or negative examples of that token, and checking how well the token's classifiers could identify the type of example this was.

2.3 Preprocessing Techniques

A number of relatively small changes had to be made to the data preprocessing system in the original paper to make it usable to other languages. Particular care was taken to select preprocessing steps that could be generalized across as many languages as possible. These steps fell into a few categories, which are discussed in more detail below.

It must be noted that along with the preprocessing system, small changes had to be made in the codebase itself to handle non-English data. File encodings had to be set to utf-8, in all file I/O operations and the Python files themselves. The default encoding is ascii, which does not include most non-English characters.

2.3.1 Basic Data Cleaning

In this section, “basic data cleaning” is referring to steps like removing punctuation or lower-casing words (when applicable). These steps are essential for proper token segmentation. Several small changes had to be made to the original system so that it could also clean Spanish and Hindi text. Originally, the English system simply removed all characters that were not alphanumeric using regular expressions. This was not trivial to extend to Hindi characters and Spanish accents, so instead a list of punctuation characters were iteratively removed. The original list came from nltk and this list was extended to include terms like the Spanish upside-down question mark, and the Hindi full stop.

2.3.2 Removing Stems

In many languages, words can be formed by taking a root “stem” of a word like “walk”, and conjugating it in different ways depending on context like “they are walking”, or “I walked.” In many natural language processing tasks, it is beneficial for different conjugations of a word to be recognized as being essentially the same word. To accomplish this, tools like Lemmatization and Stemming exist with the goal of taking conjugated words and reducing them to their non-conjugated forms.

2.3.2.1 Lemmatization

In the original English system, lemmatization was employed to remove conjugations from words. Lemmatizers are tools that attempt to replace conjugated words like “making” with correct English root words like “make”. These systems can be complex, and I found them to be difficult to find for other languages. It was for this reason that I replaced this step with the rougher, but more available step of Stemming.

2.3.2.2 Stemming

Stemmers, are the rougher, less sophisticated cousin of lemmatizers. They also seek to remove conjugation from words but do not make grantees that the results will be valid words. Stemmers can come in many forms, with the simplest being an algorithm that simply “chops off” affixes attached to words. In English, this sort of stemmer would reduce “making” to “mak”, and also probably reduce “make” to “mak” as well. This sort of system can work very well for conflating different conjugations of a word, but can fall short for unusual forms like “running”, which a very rough stemmer would likely reduce to “runn” which would not conflate with uses of “run”. In future chapters, I make use of the slightly more sophisticated nltk Snowball stemmer [29] when removing conjugation from Spanish and English text. Due to availability, a simple affix-chopping stemmer [30] is employed when removing conjugation from Hindi text.

2.3.3 Removing Stop Words

In natural language systems, "stop words" are defined as words that do not contribute to the meaning of the sentence they are in. In English, these are words like "the", "an", or "of", which are essential to creating readable English, but are mostly used to connect other words. Many lists of general stop words exist for different languages. The original English system made use of one such list from nltk to remove stop words from the English text. This was an important step, because it ensured that the system did not attempt to learn groundings for words like "and". At the same time, I found that there were a number of words like "object", "picture", or "color" that were used so often in the object descriptions that they held little physical meaning. These are designated as "domain-specific stop words", which refer to words that in general cases hold meaning, but for the particular domain have been rendered meaningless by their frequent and varied use. I found that these words could be identified by their inverse-document-frequency (IDF), which was found by taking the total number of instances where a token appeared in the descriptions, and multiplying its inverse by the total number of instances, then taking the log of the result. The comparison of the terms found this way to a general stop word list is shown in figure 2.2. The effects of stop word removal using each method are explored in Chapters 3 and 4.

NLTK Stop Words that appear in the English Data	English tokens with IDF < 0.75 sorted by IDF
after, against, all, am, an, and, any, are, as, at, be, because, been, before, being, both, but, by, can, do, does, down, each, few, for, from, had, has, have, here, his, if, in, into, is, it, its, just, more, most, no, not, of, off, on, only, or, other, out, re, so, some, such, than, that, the, then, there, these, they, this, those, to, too, under, up, very, was, we, what, when, where, which, will, with, you, your	picture, of, color, it, the, its, on, an, in, object, side, and, looks, like, image, are, with, that, be, laying, to, shaped, colored, these

Figure 2.2: Comparison of pre-built stop word list and tokens identified with IDF. Note that many regular stop words were not identified with the IDF form, which may indicate that a combination of manual and IDF-found stop words might be the most thorough method.

2.4 Amazon Mechanical Turk

This thesis makes liberal use of Amazon's crowd-sourcing platform Mechanical Turk. Crowd-sourcing is a method for data collection in which the researcher posts tasks to a large online community and asks them to complete them for compensation. This system has benefits and drawbacks. On one hand, it often allows researchers to collect a lot of diverse data in a short amount of time. On the other hand, this data is inherently noisy, and it can be hard to guarantee that the people completing the tasks have the relevant skills, or are even taking the tasks seriously.

The grounding system used in this thesis was designed with the motivation of handling unconstrained descriptions from regular users, so general noise caused by diverse

Spanish Description	Translation
Este es un pepino. El pepino puede rebanarse para usarlo encima de los ojos como humectante dentro de rutinas de belleza.	This is a cucumber. Cucumber can be sliced to be used over the eyes as a moisturizer in beauty routines
Un pepino.	A cucumber.
Un pepino verde que esta listo para comer.	A green cucumber that is ready to eat.
Un pepino maduro y de color verde.	A ripe and green cucumber.
El objeto es una verdura de forma alargada. Es de color verde.	The object is an elongated vegetable. It is green.

Figure 2.3: Sample actual descriptions of a cucumber, as collected on Mechanical Turk.

users who were giving an honest attempt at the task was actually desirable (see figure 2.3). In the original English data collection and my collection of Spanish and Hindi data, descriptions were only thrown out when the worker explicitly did not follow the directions (such as answering in the wrong language, or giving text unrelated to the image provided). For my data collection, additional checks had to be added to account for users utilizing automatic translation tools. These are talked about in sections 3.4 and 4.3.1. The directions and a sample Mechanical Turk task used for the Spanish data collection (called a HIT which stands for "human intelligence task") are shown in figures 2.4 and 2.5.

Instructions

Describa el objeto que se muestra en la imagen **en una o dos oraciones completas en español**. Al realizar este HIT, acepta que ha leído la descripción del estudio que se está realizando y da su consentimiento para que los datos que ingrese se utilicen para investigación. Por favor lea el formulario de consentimiento, si prefiere no participar en este experimento, devuelva este HIT.

Por favor haga lo siguiente:

- **Describa el objeto (no la imagen en sí)** que se muestra en las imágenes **usando oraciones completas en español como si lo estuviera describiendo a otra persona.**
- Si *no puede* reconocer el objeto, trate de describirlo con adjetivos.

Figure 2.4: The directions shown to the Spanish Mechanical Turk worker.

Figure 2.5: A sample of the form that was presented to the workers. Each task asked them to annotate five images.

Chapter 3: Expanding the System: Spanish

3.1 Introduction

The focus of my thesis is to examine the transferability of the English system to non-english text. To do this, I needed to choose a language to begin my comparison with. I decided to start with Spanish, for several reasons. The first was because Spanish is a very common language in the United States, and due to this I had some fluency in it. The second was that Spanish has very similar grammar, characters, and vocabulary to English, minimizing the potential causes of differences in the system's performance. Through the lens of analyzing this simpler problem, I sought to establish a general analytic framework, which would make it easier to then compare the performance of languages like Hindi that differed strongly from English.

When I looked for related works that analyzed language pairs, several worked with Spanish and English specifically, [24, 25]. These papers drew attention to differences in the languages that could potentially cause performance differences for NLP systems. For the GLS, it was the morphological richness of Spanish as well as the high rate of inflection [24] that ended up causing the most differences.

3.2 First Steps: Google Translate

For my preliminary experiments, I only had access to the English corpora. I wanted to get baselines in how a Spanish corpora might perform, so to get a rough estimate I translated the English descriptions to Spanish through Google Translate's API [14]. As a sanity check on the quality of translation, the translated text was then translated back into English, again using Google Translate. The English and back-translated English phrases were compared manually to see if their overall meanings were preserved. A total of 2,487 out of the 6,120 (around 40%) phrases remained exactly the same between translations. For the remaining 60%, five hundred back-translated phrases were randomly selected and manually compared to their original English version (see table 3.2 for examples). Approximately 87% of the phrases examined preserved their meaning between translations, so I estimated from this that about 90% of the phrases were translated accurately (shown in figure 3.1).

For those phrases that did not translate accurately back to English, I observed a number of patterns. Some of them were simply due to ambiguities with the meaning of a word where the wrong one was selected during one of the translations (as an example, for the bottom row of table 3.2, "forma" can mean "shape" or "way"). A common example of this was the phrase "this is a red cabbage" becoming "this is a red wire," which happened

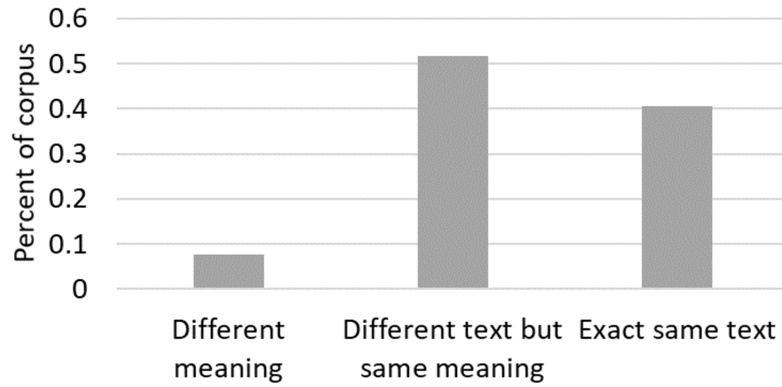

Figure 3.1: Breakdown of meaning preservation for English and English-Spanish-English translation.

Image ID	Original English	Spanish Translation (Google API)	Back-translated English (Google API	Same Meaning?
Orange 2	This fruit is called an orange	Esta fruta se llama naranja	This fruit is called orange	Yes
Cuboid 4	This is a picture of rectangular shaped blue coloured solid block	Esta es una foto de bloque sólido de color azul con forma rectangular.	This is a solid block photo of blue with rectangular shape.	No
Lime 2	It is a lime	Es una lima	It's a lime	Yes
Cuboid 2	This is a block The block is green The background is black The green block is laying on its side	Esto es un bloque El bloque es verde El fondo es negro El bloque verde está de lado	This is a block The block is green The background is black The green block is on its side	Yes
Cuboid 3	THIS IS A SHAPE	Esto es una forma	This is a way	No

Figure 3.2: Samples of English descriptions that were translated into Spanish and then back into English. The column on the right indicates if the meaning of the original English text matches the final back-translated English

six times out of the five hundred selected phrases. Another error that occurred three times was “laying on its side” becoming “set aside,” since the Spanish phrase “puesta de lado” can mean “put sideways” but also “set aside.”

Other translation errors could be related to differences in Spanish and English structures. The pronoun “it” commonly became “he,” as Spanish nouns are gendered. Phrases with many adjectives saw them switching places with each other and the nouns they were attributed to. For example: “This is a picture of rectangular shaped blue colored solid block” became “This is a solid block photo of blue with rectangular shape.” One could attribute this confusion to differences in the rules of adjective ordering between English and Spanish.

3.3 Scores for English and Google Translated Spanish

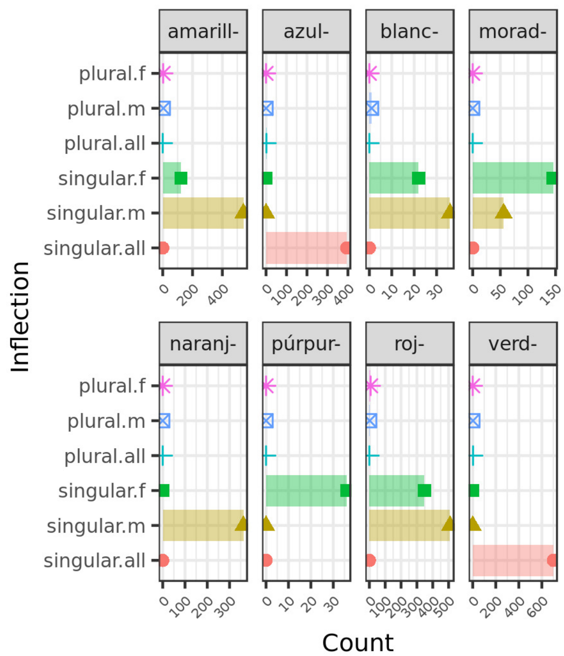

Figure 3.3: Proportion of color word forms in raw translated Spanish.

As a preliminary test of how the system could handle Spanish data, I ran the classifier on the translated Spanish and English corpora with minimal preprocessing (lowercasing and removing punctuation), and tested the color tokens only. My goal was to get a baseline for how the system would perform, using tokens that would be easy to compare between languages. It was expected that the translated Spanish corpus would preform worse, since it was not perfectly translated. When the tests were run, the translated Spanish did perform slightly worse (see figure 3.4), but an additional interesting issue emerged.

Spanish, unlike English, has adjective-noun agreement. This meant that a simple color word like “red,” could translate to “rojo,” “roja,” “rojos,” or “rojas” depending on the gender and plurality of the noun it was describing. This meant that the possible positive instances for color words could be split between the various forms, since different descriptions of the same object might use a different form depending on the structure of the sentence. One can see from figure 3.3, that in the overall translated corpus, the color words were split between different conjugations. This led to the hypothesis that some form of lemmatization or stemming would be necessary for Spanish, in a way that would have been less essential for English.

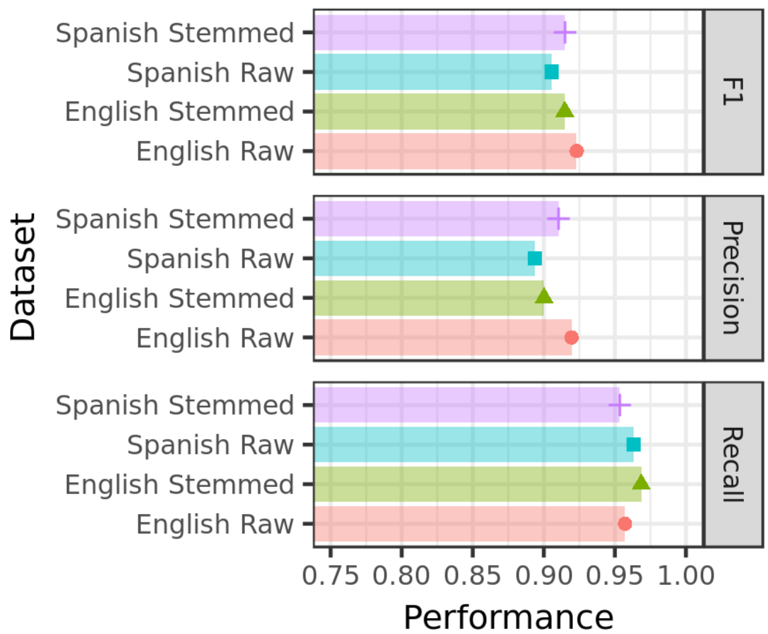

Figure 3.4: Average scores for color-related tokens between English and translated Spanish.

I decided to run both the translated Spanish and English descriptions through a Snowball stemmer [29]. I chose this stemmer as it is readily available for a wide variety of languages through the nltk library (see section 2.3.2.2 for more information). The results can be seen in figure 3.4.

One can see from figure 3.4 that applying stemming to the translated Spanish descriptions had a small positive effect on the F1-scores of the color classifiers. It also slightly raised the average number of positive instances per token, since stemming allowed instances that were split between small counts of several forms of a word to see them as the same word. One can see this in more detail in figures 3.6 and 3.7, which show the difference between the average of the scores for the various forms of color words in the unstemmed data (for example amarilla and amarillo would be averaged as amarill*), and the stemmed score of the stemmed form.

One can see in figure 3.6 that for the three colors shown, stemming always increased the average precision for that color, but could reduce recall. In addition from figure 3.7, one can see that some of the colors had a large increase in average positive instances, while others did not. This was likely due to a case where many instances labeled with

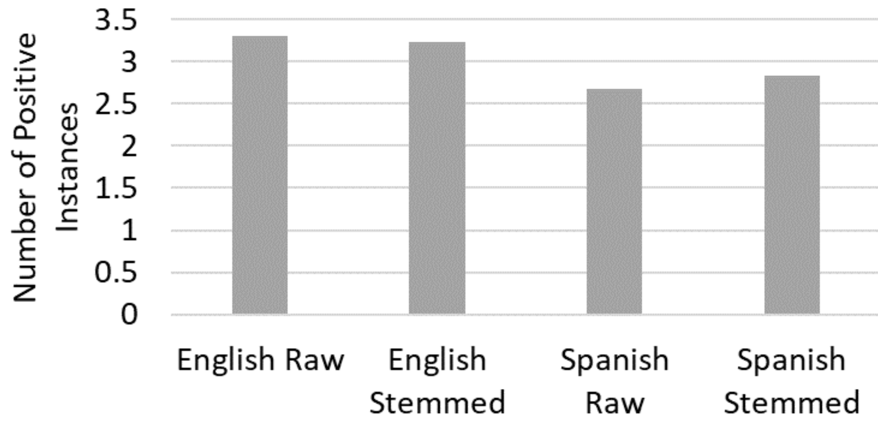

Figure 3.5: Average number of positive instances for color words.

"rojo" also saw enough "roja" that it was a positive instance for both. When looking at the counts per instance, I found that for the 23 instances that had the token "roj" in their stemmed descriptions, 16 were positive examples of both "roja" and "rojo" in the un-stemmed version. For objects like cabbages (coles) and plums (ciruelas), "roja" was used dramatically more, while for tomatoes (tomates), cubes (cubos), and cylinders (cilindros) "rojo" appeared more.

As a final check, I examined the number of occurrences over all descriptions of each instance of the stemmed and un-stemmed versions of color words. For most of the colors, instances were often split between possible conjugations. For "amarill" (yellow), there were five instances where the individual counts of both un-stemmed forms of yellow: "amarillo" and "amarilla" were less than the threshold for a positive instance, while the stemmed version "amarill" was able to overcome that threshold. This is shown in the more dramatic increase in number of positive examples in figure 3.7. The effect on the scores is more complicated, since very yellow instances often had 50 or more occurrences of "amarill." Because of the inherent messy nature of the data, instances with low but still significant counts of tokens (greater than five occurrences) are much more likely to be falsely positive examples that could damage a classifier. One can see this in figure 3.8 where the instance "eggplant 1" was called green seven times in the English data. This is clearly because the stem of the fruit is green. However, a simple classifier may be confused by this instance, since it is mostly purple. This could lead a "green" color classifier to incorrectly label purple objects as green.

3.4 Collection of Real Spanish Data

Exploring comparisons between English and translated Spanish enabled me to get a basic idea of how Spanish descriptions might differ from English. However, in order to truly compare the languages, I needed to collect real Spanish data. I attempted to follow the methods used by Pillai et al. [8] as closely as possible to obtain comparable Spanish data to their English data. I utilized Amazon Mechanical Turk (see section 2.4)

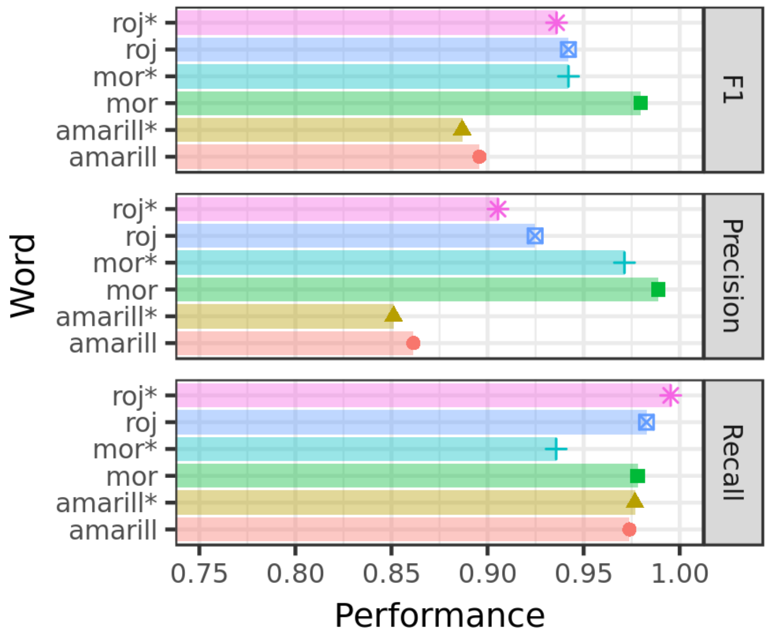

Figure 3.6: Comparing the average of the unstemmed scores for various word conjugations and their stemmed score.

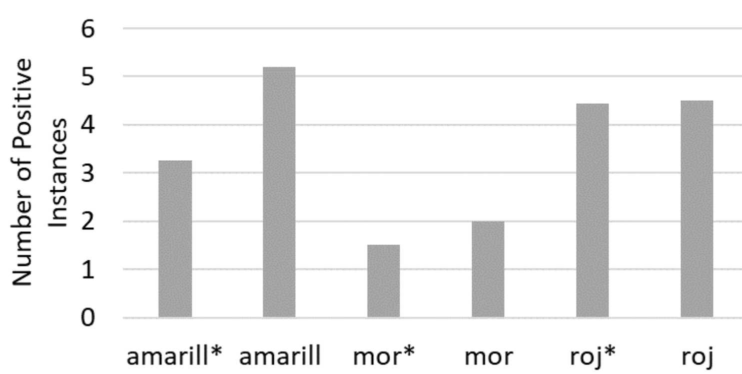

Figure 3.7: Comparing the number of positive instances between raw word forms and stemmed.

Instance	Occurrences of “green”	Images
Cube 2	75	
Cucumber 2	31	
Eggplant 1	7	

Figure 3.8: Sample of instances that had more than five occurrences of “green” in the English corpora.

to collect Spanish descriptions of the images in the database.[1] In addition, workers were required to have at least fifty HITs accepted before being eligible to work on my HITs. To avoid biasing the workers towards a particular type of description, I provided no example descriptions.

I excluded data from a small number of workers who did not follow the directions (for example, responding in English or randomly selecting adjectives) and obtained additional high quality data to replace their submissions. All other submissions were accepted. This allowed for a wide variety of answers. One worker might simply name a carrot, while another would describe how it tastes, what foods it goes well in, or where it comes from (see figure 2.3 in section 2.4 for examples). The English dataset was similarly noisy. This is desirable, as a robot that is trying to learn language from an average user must be able to handle the many ways in which a user might choose to introduce a new object.

One possible danger in collecting Spanish data that was considered was that someone might be responding in English and using a translation tool. I attempted to check for this by comparing my real Spanish data to the translated Spanish data. I found that short descriptions like “Esto es un limón” (this is a lemon) had a large amount of overlap, but in general most of the real Spanish descriptions were longer and did not mirror any of the translated results. More work was put towards this issue with the Hindi data (see section 4.2). In the future, it would likely be useful to employ a more sophisticated method of ensuring Spanish fluency.

In the end, the total number of Spanish descriptions per object type was on average slightly lower than in the English corpus (see figure 3.9). I controlled for this in my analysis 3.5 by taking several random subsets of both corpora such that each instance had an equal number of Spanish and English descriptions and averaging the results.

3.5 Comparison of Spanish and English

3.5.1 Overall Scores

In figure 3.10, one can see the averaged F1-Score for the color, shape, and object classifiers between the original English and the collected Spanish descriptions. Each

[1] This was accepted as an IRB exempt study

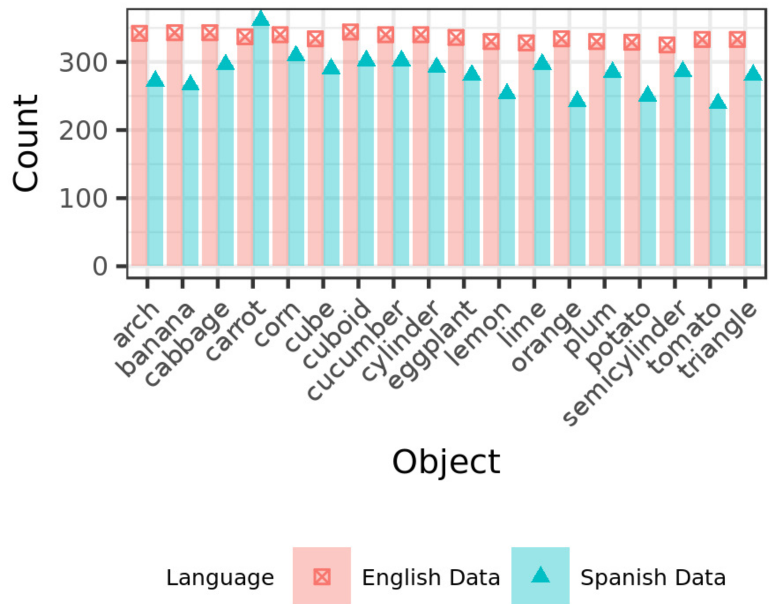

Figure 3.9: Total number of descriptions collected per object type.

score was found by averaging the results of twenty evaluation runs each of ten train-test splits. These scores were averaged across all tokens learned, without specifically subsetting for the tokens that naturally represented colors, shapes, or objects. In general, the scores were fairly similar, varying between 0.8 and 0.84. From the small differences one can see that stemming appeared to benefit the Spanish data for learning object and shape classifiers, but slightly hurt the performance for color classifiers. Un-stemmed English performed better than either Spanish version for color and object classifiers, but worse for shape classifiers. Much like with Spanish, stemming appeared to help the shape and object classifiers, and hurt the color ones.

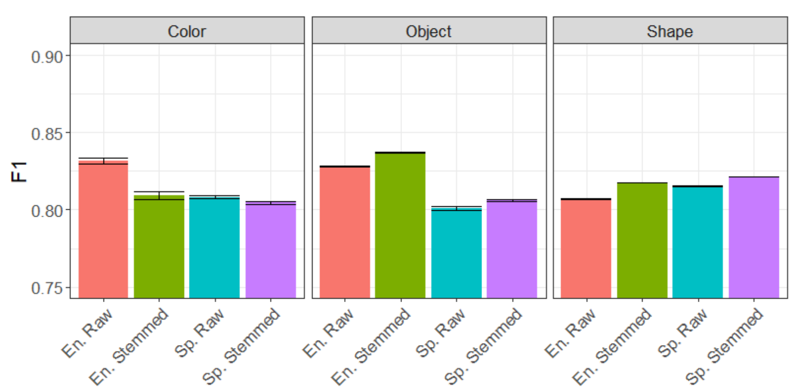

Figure 3.10: Average F1 scores for English and Spanish classifiers of each type.

3.5.2 The Effect of Stemming

As one can see from figure 3.10, the effect of Stemming on the F1-Scores of the English and Spanish classifiers was not consistent. For both the object and shape classifiers, stemming appeared to either benefit or have little impact on the object recognition task. For the color tokens, stemming either barely impacted or lowered the scores.

Stemming can cause words to be conflated correctly or incorrectly. Incorrect stemming can certainly cause problems, where tokens are conflated that shouldn't be, or words that should be conflated are not. However, as discussed earlier, it is also possible for correct stemming to cause an instance to barely meet the threshold for being a positive example of a particular token 3.8, when perhaps that instance is not a good example of that token in reality. This is a particularly likely occurrence due to the inherent messiness of crowd-sourced data and the fact that the GLS was basing its classification label off of these messy descriptions. Due to this, and the high amount of conjugation in Spanish, it was decided that stemming would likely be a good step to employ with Spanish irregardless of the scores.

	Token	Count	F1-Score	Token	Count	F1-Score	Token	Count	F1-Score
English	corn	261	0.926508	banana	261	0.82946	lemon	252	0.907535
Spanish (accented)	maíz	117	0.802675	plátano	90	0.65640	limón	165	0.898421
Spanish (no accent)	maiz	65	0.793374	platano	47	0.66132	limon	118	0.866587
Spanish stemmed	maiz	182	0.835578	platan	140	0.65773	limon	283	0.840359

Figure 3.11: Object Scores for words that could be written with and without accents.

3.5.3 Accents

Another interesting difference that stood out when examining the real Spanish data with translated data was the use of accents. Unlike with the translated data, the real Spanish data was inconsistent with its usage of accents. While a majority of workers used accents where they were supposed to go, a not-insignificant percentage of them left them out (see figure 3.11 for examples). This is likely because those workers did not have easy access to a keyboard with accented characters, and thus chose to leave them off. One can see in figure 3.11 that for common accented words, this had the effect of splitting the data. Luckily, the snowball stemmer [29] automatically removed these accents. One can see in figure 3.11 that after stemming, the counts for the accented and unaccented versions of the token were combined. The combined classifier did not always have a higher score on the testing data, for similar reasons to those discussed in section 3.5.2.

3.5.4 Stop Words

Without employing stop word removing during preprocessing, the system learned a total of ten words that could be classified as general stop words for English and eight for Spanish (see figure 3.12). This happened because for each of these words there was at least one instance where the word did not appear in any description. For Spanish, the tokens “de,” “es,” “una,” “y,” and “se,” and for English the tokens “this,” “is,” and “a” all had zero negative instances and were appropriately removed.

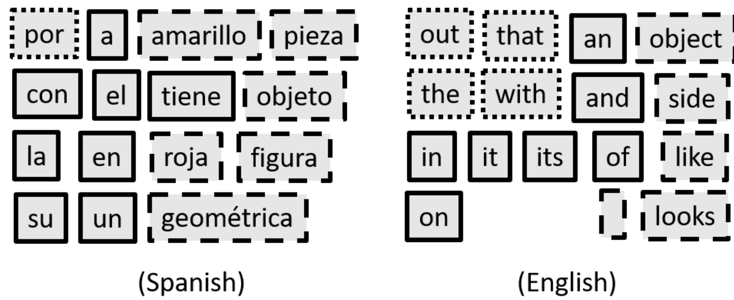

Figure 3.12: Stop words that appeared often enough to have classifiers trained on them. A dotted border indicates a stop word from the language’s nltk stop word list. A dashed border indicates this token was in the top 2% tokens by ascending IDF score. A solid border means the token appeared in both lists.

Figure 3.12 also shows tokens that appeared in the bottom 2% of tokens when sorted by IDF score. This was my way of estimating “domain-specific stop words”. Note that there were quite a few nltk stop words that also had very low IDF scores. The IDF method identified tokens like “object”, or “looks” which were used very often in the English descriptions and ideally should be ignored. Figure 3.13 shows how removing each type of stop word impacted the scores of the raw classifiers. For both languages, the greatest impact appeared to come from removing both general purpose stop words and low-IDF tokens, though the impact was small in all cases.

It must be noted that for the Spanish data, the tokens “amarillo” (yellow) and “roja” (red) were included in the bottom 2% of tokens by IDF score. These tokens were common due to the prevalence of red and yellow objects in the dataset, and this suggests it might be beneficial to explore a more nuanced approach to finding domain-specific stop words in the future.

3.5.5 Token-Level Comparison

Figure 3.10 shows the average difference in F1-scores between the English and Spanish data. This section digs deeper into the performance differences by identifying objects, shapes, and colors that individually had significantly higher classifier scores in one language than the other, and exploring the differences in how the English and Spanish systems grounded these concepts.

It must be noted that for the sections below, I concentrate on objects, shapes, and colors that were used often enough in both the Spanish and English datasets as to be learned. Each language had a number of words where the direct translation was not used very often or at all in the other language. I did attempt to pair words together even if they were not direct translations if they were used in exactly the same context (for example

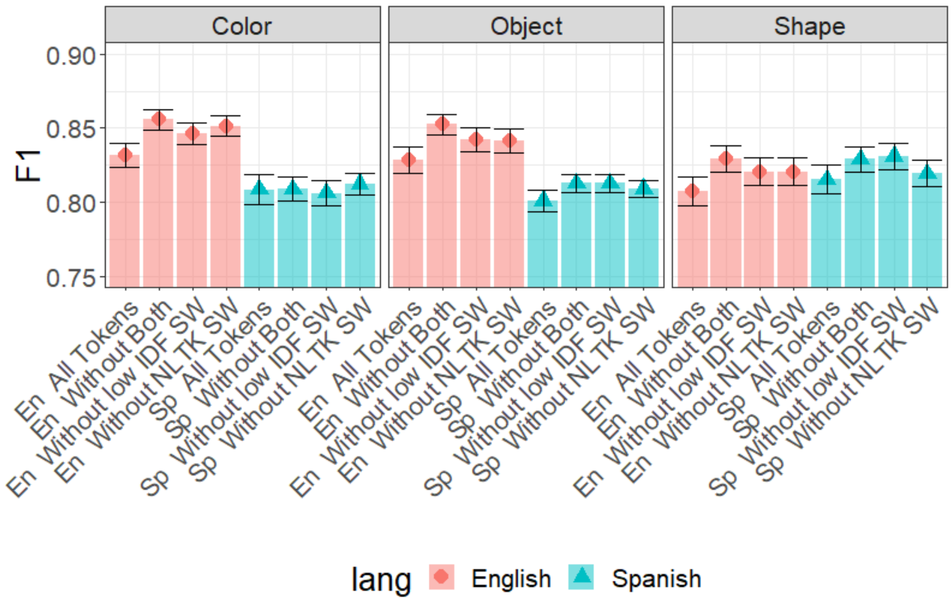

Figure 3.13: The graph demonstrates the impact on the average F1-score of removing nltk stopwords versus removing the lowest 2% tokens by IDF score for English and Spanish. The error bars show how the variance of these scores among the tokens averaged.

"espiga" means spike in English, but it is used in the same way an English speaker would use "cob" with "cob of corn"). In addition, for each section I first identified the tokens with greater than 0.05 difference in the F1-Score between Spanish stemmed and English stemmed. I then subset this set of tokens by removing stopwords (including domain-specific words like "figure") or words that were not representatives of that section of words. As an example: for the object scores, I ignored differences in words like "round", because the shape classifier scores would be far more informative as to how well that concept was learned.

3.5.5.1 Object Tokens

For this section of classifiers, I identified object-related tokens with large (greater than 0.05) performance differences between the Spanish and English versions. Four English tokens were identified, corresponding to seven Spanish tokens, as three of the English tokens had multiple translations. The scores for these tokens and their number of occurrences in each dataset are shown in figure 3.14. For all three tokens, the precision was what caused the difference in F1 scores. That is, the Spanish tokens were more likely to incorrectly label their negative instances as positive. For all object classifiers learned, the average precision was 0.049 points lower for Spanish stemmed than English stemmed, while the recall was about the same.

Thus we have the four tokens: banana, cabbage, carrot, and vegetable, which performed better in English than in Spanish. Figure 3.14 shows that the Spanish tokens tended to have lower counts, but this does not mean that the English system had more to learn from. The grounding system does not care about how many times an instance is

English token stemmed	Count	F1 Score	Spanish stemmed token	Count	F1 Score
banana	**266.6667**	0.82165	**banan**	**45**	0.667662
banana	-	-	**platan**	**132**	0.657732
cabbag	**237**	0.929735	**col**	**28**	0.835175
cabbag	-	-	**repoll**	**113**	0.829441
carrot	**267.3333**	0.910907	**zanahori**	**151**	0.780711
veget	**66.66667**	0.953333	**vegetal**	**97**	0.766956
veget	-		**verdur**	**62**	0.80199

Figure 3.14: Object tokens with large (greater than 0.5) differences in F1 score between Spanish and English.

described with a token, so long as it exceeds a given threshold. It makes more sense to then look at the positive and negative instances identified for each token.

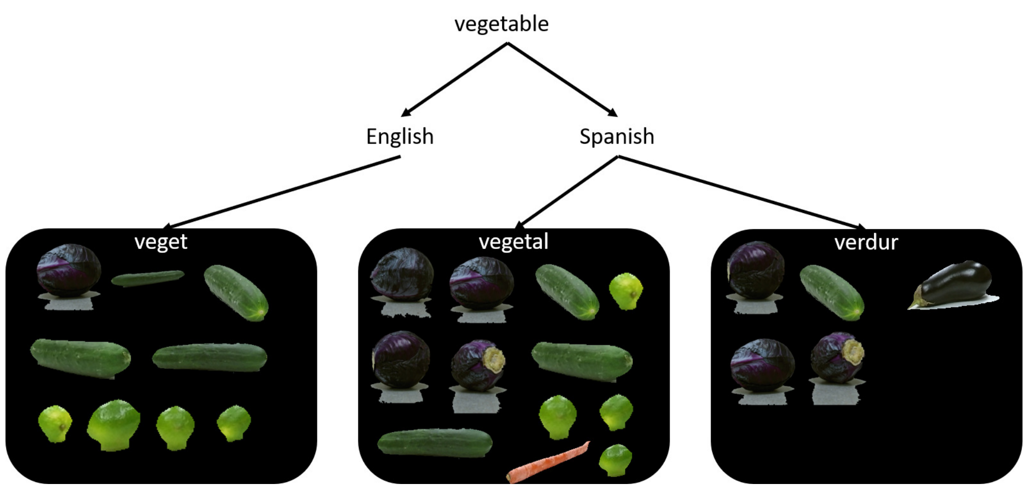

Figure 3.15: Positive instances found for the vegetable tokens in Spanish and English (stemmed).

Figure 3.15 shows the various positive instances identified for vegetable tokens in Spanish and English. Since the dataset has many vegetables, there is a variety of images being used. These images are the "ground truth" for the tokens shown. That is, the scores from the tokens are partially based on how well the token classifiers trained on some subset of these images (along with negative examples) can recognize the rest of the images as positive examples. The images in figure 3.15 help us to see where the score difference might come from. The English token is trained and tested on mostly cucumbers and limes, while the Spanish "vegetal" token must also learn cabbages and carrots as positive examples and "verdur" simply has fewer examples overall to learn from. This is likely due to differences in word usage between the languages. Simply put, the Spanish-speaking workers were more likely to mention that something was a vegetable than the English-speaking ones. In this, the Spanish classifier for "vegetal", while performing worse, was actually better since it had a more thorough understanding of the underlying definition of a vegetable.

For the other three objects: carrots, cabbages, and bananas, we see from figures 3.18, 3.16, and 3.17 that the Spanish and English tokens shared the same positive instances. For these three objects, I instead looked at their negative instances. What instances were identified as examples of what the objects were not? It was interesting to see that for all three objects, there was some overlap in the negative examples found for each language, so word usage between the languages had enough parallels that some instances that had very different descriptions from the object in English had similarly different descriptions in Spanish.

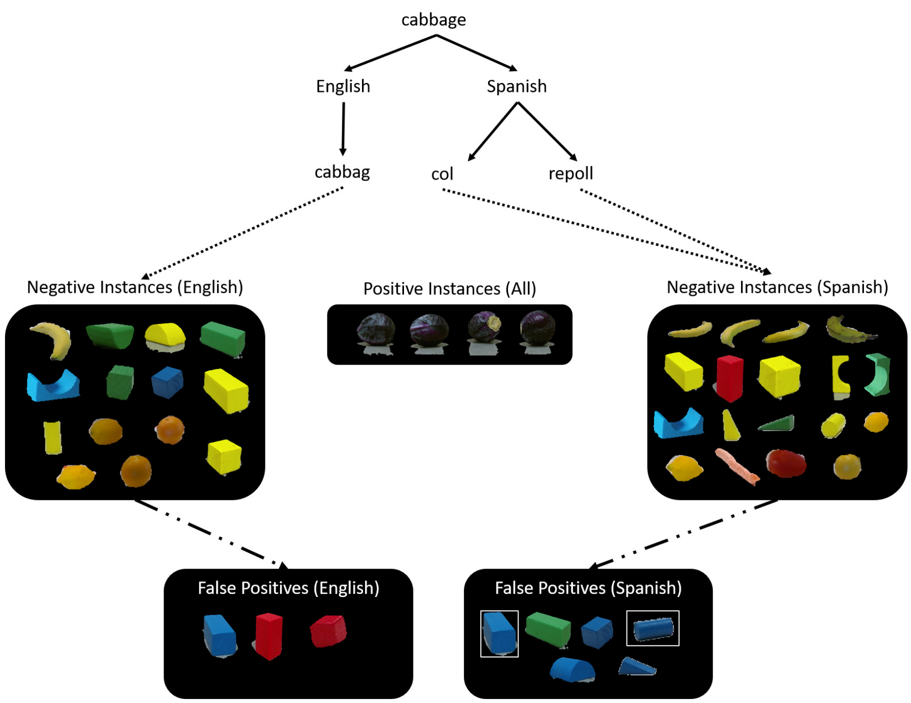

Figure 3.16: Positive and negative instances found for the cabbage tokens in Spanish and English (stemmed). In the False Positive squares, instances that were false positives multiple times are outlined in white, with a thicker border indicating the mistake happened more often. Note that the Spanish "col" and "repoll" classifiers had a hard time with blue objects.

3.18, 3.16, and 3.17. It is interesting to note that for the three objects shown, images that were false positives for only one language were often identified as negative examples during training for the other. As an example, the Spanish cabbage classifiers were often fooled by blue objects of a particular shade. The English classifier had far fewer problems, and this may be because a blue cube was identified as a negative instance in training.

A few properties of the system could be causing these results. The negative instances found during training are those instances from the training split that are furthest away from positive instances. The negative instances found in testing are the instances in

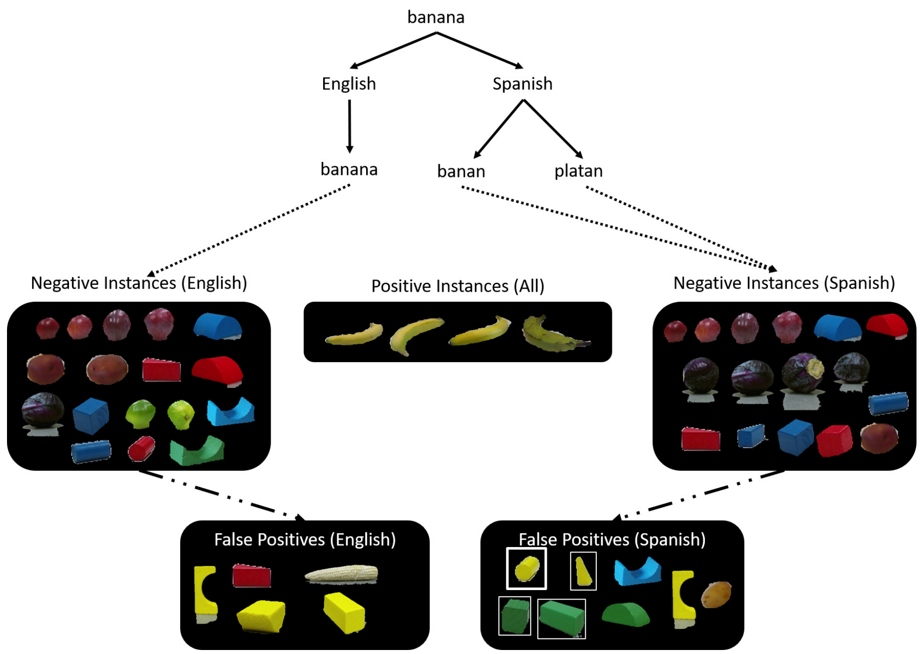

Figure 3.17: Positive and negative instances found for the banana tokens in Spanish and English (stemmed). Note that the Spanish "platan" and "banan" classifiers also miss-classified green objects.

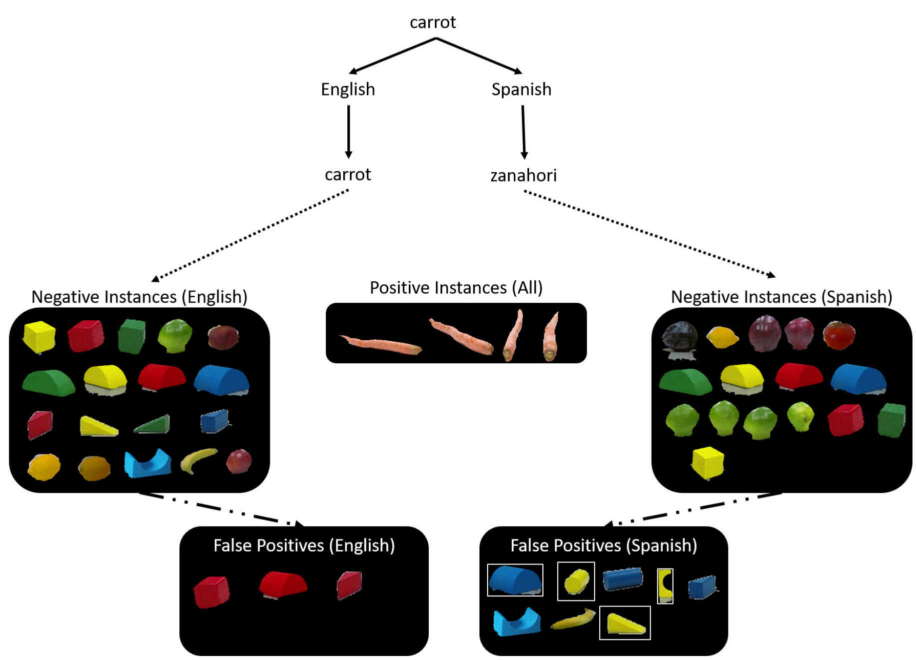

Figure 3.18: Positive and negative instances found for the carrot tokens in Spanish and English (stemmed). Note that the "carrot" classifier was trained on many of the negative examples that fooled the "zanahori" classifier.

the testing split that are furthest away. Since the test split chooses exactly one instance for each object, this forces stronger diversity in the negative instances tested. For the three objects shown, the English versions appeared to have identified the more confusing negative instances during training, allowing the classifiers to have fewer errors in testing. This could indicated that the underlying English descriptions are a bit more diverse between instances for these objects.

3.5.5.2 Shape Tokens

In this section, I identify a few shape-related tokens that performed differently on their shape classifiers between Spanish and English. The previous section looked at object classifiers, which are concerned with both shape and color. The shape classifiers ignored color features. Note from figure 3.10 that the average shape classifier scores were very close for Stemmed Spanish and Stemmed English tokens. The Spanish tokens actually performed slightly better on average, though the for the specific shapes below the English tokens performed better.

English token stemmed	Count	F1 Score	Spanish stemmed token	Count	F1 Score
squar	**61.33333**	0.853391	**cuadr**	**120**	0.782551
triangl	**88.33333**	0.904916	**triangul**	**149**	0.794936

Figure 3.19: Shape tokens with large (greater than 0.5) differences in F1 score between Spanish and English.

The two shapes identified were square and triangle. Figures 3.20 and 3.21 show the positive, negative, and false positive instances that were found for the Spanish and English versions of the words. For the square tokens, we see that the Spanish version had far fewer negative instances to go off of in training, and some misleading positive instances, both of which could cause confusion in the classifier. For the triangle tokens, it is a bit less clear. However, one can note that corn instances seemed to be identified as negative instances to train and test the "triangul" classifier on, while they don't appear to have been found for "triangle", and seem to be easily mistaken in shape for triangles. In addition, carrots were not identified during training as good negative instances for "triangul", and during testing it had problems identifying them as negative. In general, a lot of the performance differences seem to come down to differences in word choices across the corpora that caused more or less informative negative examples to be chosen during training.

3.5.5.3 Color Tokens

This section examines color tokens that performed differently between Spanish and English. These classifiers were only trained on the color features of the images.

On average, there was little difference between the color scores of the Spanish and English stemmed tokens. For the tokens found in this section (see figure 3.22), Spanish

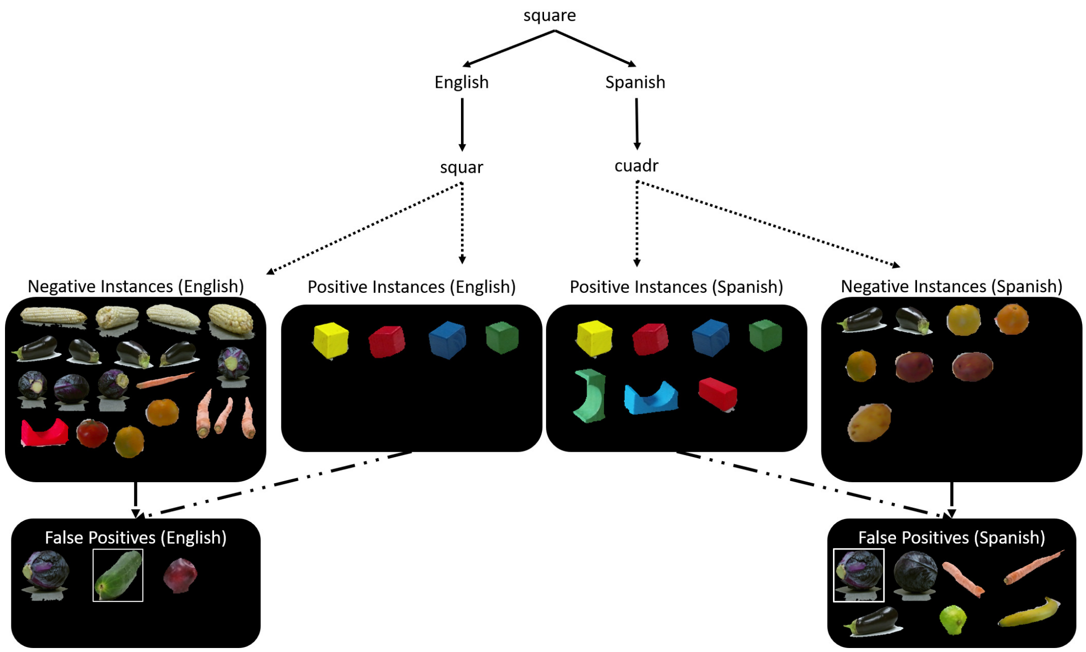

Figure 3.20: Positive and negative instances found for the square tokens in Spanish and English (stemmed).In the False Positive squares, instances that were false positives multiple times are outlined in white. For the sake of space, only one image of each instance is shown, but the system was trained and tested on these instances from different angles.

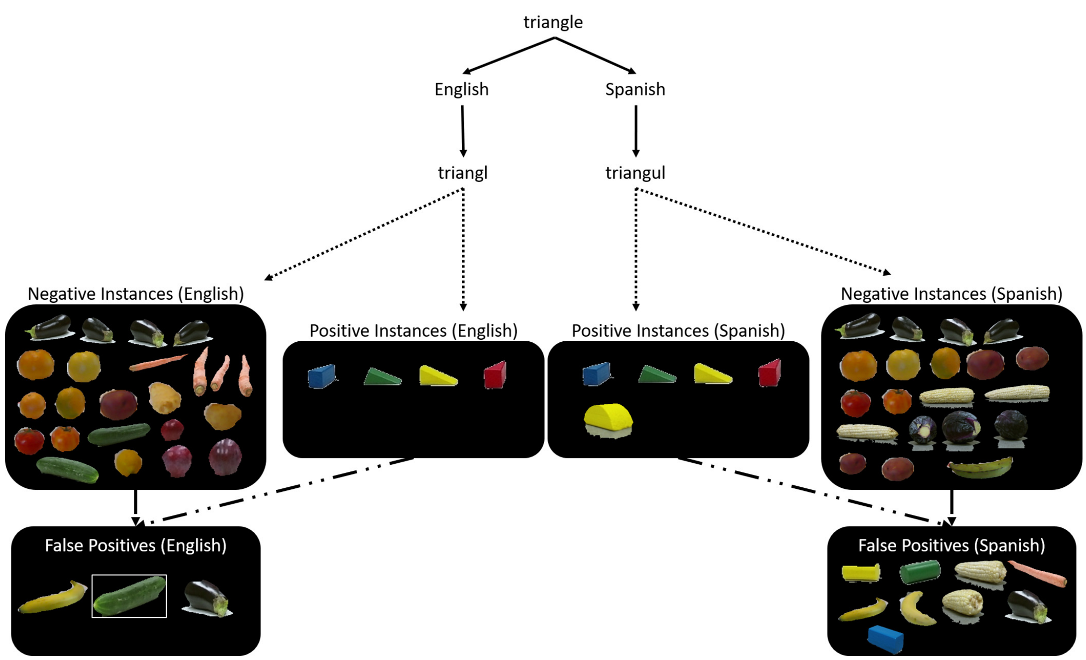

Figure 3.21: Positive and negative instances found for the triangle tokens in Spanish and English (stemmed). Note that corn instances were chosen as negative examples to train and test more often for the Spanish token.

English token stemmed	Count	F1 Score	Spanish stemmed token	Count	F1 Score
yellow	**562**	0.844851	**amarill**	**648**	0.932648
green	**599.6667**	0.870932	**verd**	**674**	0.735079

Figure 3.22: Color tokens with large (greater than 0.5) differences in F1 score between Spanish and English.

performed better for yellow, and English performed better for green.

For the yellow tokens, we see that there were fewer negative instances found for “amarill”, and in general “amarill” appeared more often in a more diverse list of instances than “yellow” did (“yellow” had 44 instances where it was never used in any descriptions, as compared with “amarill” which had 19). However, we see that the threshold was crossed for a similar list of instances in both languages. The restriced possible negative example set would cause the “amarill” classifier to be tested on fewer things, which could explain the higher score.

For the green tokens, there is a less dramatic but similar situation, where “green” only had 37 possible negative instances while “verd” had 44. Both classifiers were confused by yellow and blue instances.

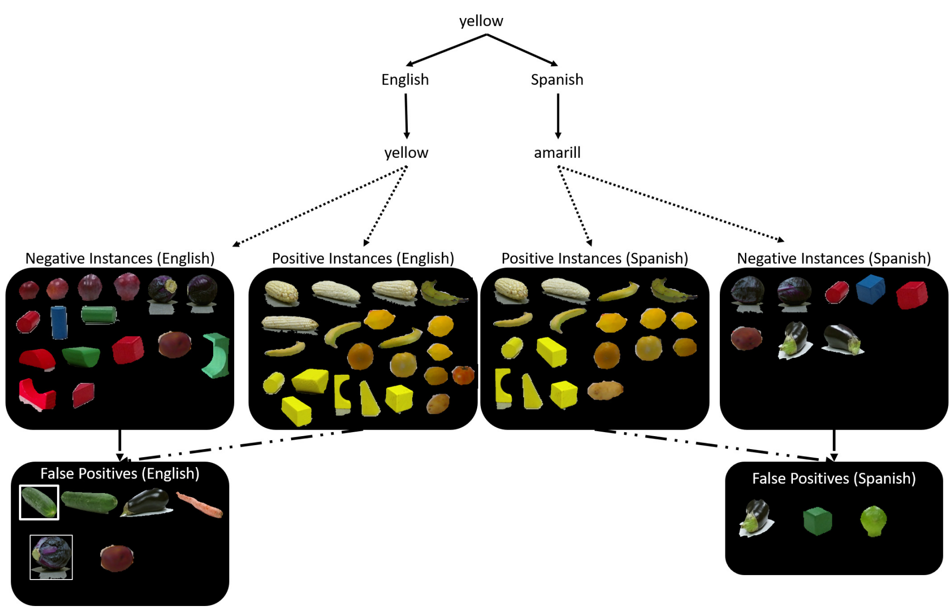

Figure 3.23: Positive and negative instances found for the yellow tokens in Spanish and English (stemmed).In the False Positive squares, instances that were false positives multiple times are outlined in white, with a thicker border indicating the mistake was made more often. Note that “amarill” had fewer negative tokens than “yellow”, indicating that there were fewer instances overall that did not see “amarill” at least once.

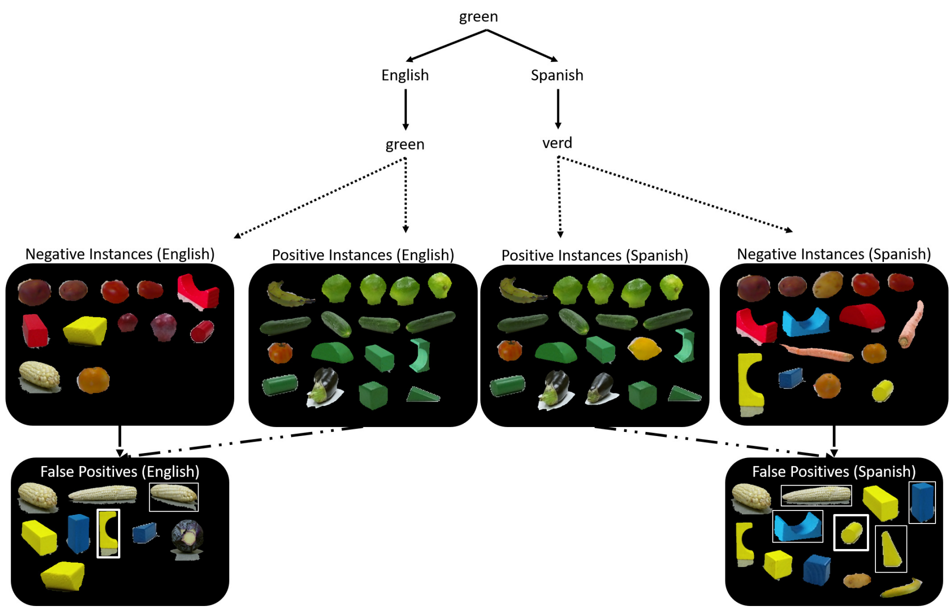

Figure 3.24: Positive and negative instances found for the green tokens in Spanish and English (stemmed). Note that yellow and blue instances were rarely chosen as negative instances to train on, and were responsible for most false positives.

Chapter 4: Expanding the System: Hindi

4.1 Introduction

Hindi is the native language for hundreds of millions of people [31]. It is from a different language family than English or Spanish, has a wide variety of dialects with small linguistic differences, and uses its own script. Just as Spanish was chosen as the first language to apply the system to due to its similarities to English, Hindi was chosen as a language significantly different from the ones applied to the system before. It was also an ideal language for my situation, as there were several students in the same lab who spoke the language that I could consult with.

To inform this work, I looked into recent Natural Language Processing research using Hindi, mainly on the tasks of entity recognition [32] and text summarization [33–35]. NLP work in Hindi is complicated due to the variety of dialects found within the language, and the lack of large annotated corpora [32]. In addition, tasks like entity recognition are complicated due to the language's free word order, lack of a concept of capitalization, and variation in the spelling of proper nouns [32]. The grounding system my work is based off of uses a bag-of-words approach, so the free word order would likely not cause a problem (though if the system were ever to be updated to take word order into account, this would be something to watch out for). The system also generally ignores capitalization, so it is only the possible spelling differences that might effect how well groundings are learned. For preprocessing steps, many of the papers used pre-made lists of stop words [33–35]. Stemming was also popular, and was accomplished using either Hindi WordNet, or a simple largest-suffix removal stemmer [30]. Since stemming turned out to be very necessary for Spanish, and was used so often in general Hindi NLP work, I decided to add it in from the beginning when comparing the English and translated Hindi performances as shown in figure 4.2.

4.2 Google Translated Data

As with the Spanish data, I chose to start my analysis of Hindi with a translated version of the English corpus using the Google translate API [14]. This allowed me to quickly test the system with data in Hindi characters, and get an idea for places where properties of Hindi might cause differences in system performance.

As before, once the English data had been translated, I translated it back to English and ran a check on what meaning was retained between the original English descriptions and the English-to-Hindi-to-English descriptions. There were a number of common word replacements, such as "side" becoming "shore," "lime" becoming "color", "child" becoming "wild," and "oblong" becoming "rectangular." These could all be explained by the Hindi translation of the first word having several possible meanings. A more interesting effect of the translation was adjectives and descriptions of objects simply being removed. For example, the phrase "this is a carrot laying on its side" became "this is a carrot." In addition, the back-translation often corrected misspellings in the English text, where the Spanish translator had usually just left them untranslated. An example is the token "eggplanet" which was used often enough as a misspelling for "eggplant" that it learned its own classifiers. The Spanish translator generally left this word alone, treating it as a proper noun, while the Hindi translator attempted to derive the intended word and produced "eggplant" upon back-translation. It appeared that when the translator could not derive and English word, it attempted to write out the sounds for the English word in Hindi characters. Both the spelling correction and the propensity to chop off phrases were likely caused by differences in the underlying systems Google uses to translate Hindi versus Spanish.

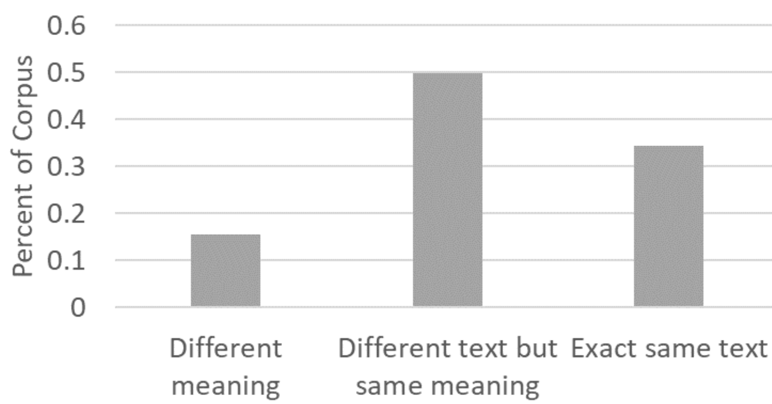

Figure 4.1: Back-Translation meaning retention between English and Hindi.

Overall, translation of the English text to Hindi and back left around a third of the descriptions exactly the same. Out of the ones that did change, five hundred descriptions were randomly selected to be examined and about three quarters of those retained their meaning (see figure 4.1). This meant that the expected percent of the descriptions that were correctly translated is around 84% (compared with the 90% for Spanish). I expected that the translated Hindi translation should perform slightly worse due to translation error, but this mostly proved to be false.

Figure 4.2 shows that for the color classifiers, Hindi scored comparably to the English data, with small differences that could be due to mis-translation. The main differences were in the object classifiers, where stemming appeared to decrease the Hindi scores, and the shape classifiers where the Hindi scores started out higher. It was interesting to see that stemming appeared to impact the Hindi translated text in similar but less dramatic ways than the English version.

Unlike with the Spanish data, I attained real Hindi data fairly soon after the Google Translate version. Since the translated data was only ever an approximation of the real thing, the bulk of the Hindi analysis is done using the collected data.

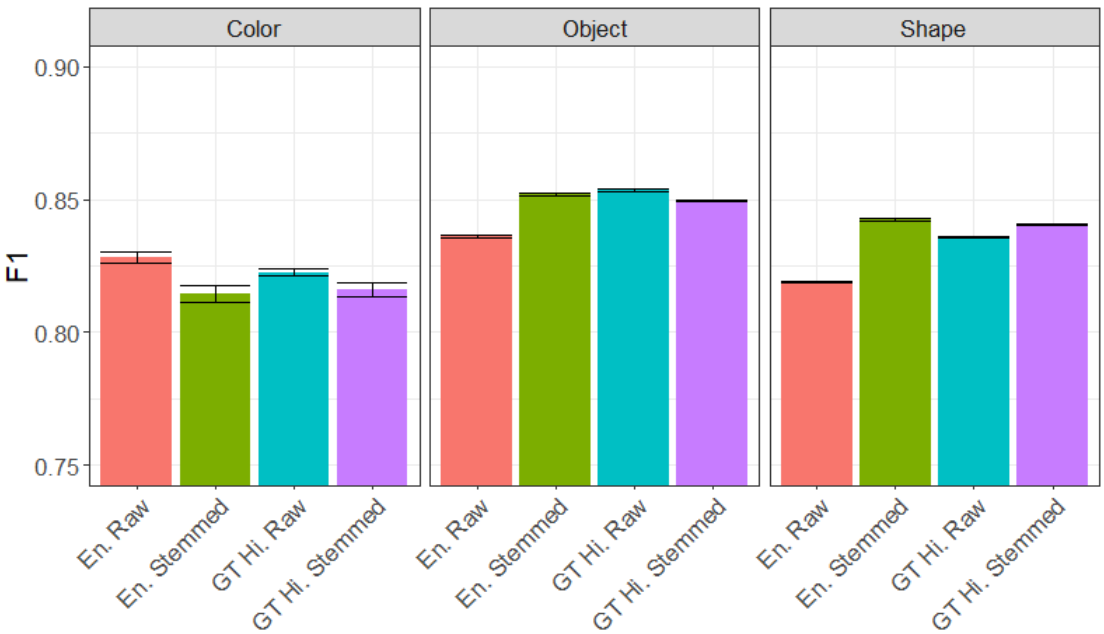

Figure 4.2: Average classifier scores for English and Google Translated Hindi stemmed and un-stemmed. The error bars show the variation in the scores across runs.

4.3 Real Hindi Data: Collection and Analysis

4.3.1 Additional Complications

Just like with Spanish, my next step was to collect real Hindi descriptions using Amazon Mechanical Turk. From discussions with Hindi-speaking students, several potential complications were identified. First, it was noted that the language name Hindi is used to describe a wide variety of dialects, which could lead to differences in spelling and word usage between workers. In addition, it was noted that many Hindi speakers in India do not bother with a Hindi keyboard, but rather communicate online with roman characters. I decided against allowing workers to submit Hindi written with roman characters, since the spelling of the converted text can vary from person to person. However, I was concerned that unfamiliarity in using Hindi keyboards might encourage workers to use a translation tool to get phrases in Hindi characters.

I decided to run a small test batch of thirty HIT's (of five descriptions each) with five assignments per HIT (so a minimum of five different workers would complete each HIT). In this batch, each answer space had an additional hidden variable that noted if the field had been pasted into. From this, I hoped to get an idea for how much people were writing with a built-in Hindi keyboard, though it was noted that a person might use an online keyboard and paste the results from that, which would be perfectly valid. The results showed around 75% of the fields had been pasted into. There were 19 workers in total who worked on HITs. Out of this, two of them never used copy/paste. However, 52/150 of the HITs (from 7 workers) had at least one field that was not copy/pasted into.

This means that while there was a decent amount of copy/paste happening, around half of the workers had at least partially completed the HITs without another source. This was promising, since it suggested that workers were indeed available that were familiar with Hindi keyboards.

4.3.2 Data Collection

The Hindi data was collected in one large batch using the same HIT design as described in section 2.4, where each HIT was given 18 assignments to encourage diversity in the answers (18 assignments means at least 18 unique workers had to complete that HIT). 56 unique workers contributed to the dataset. Out of these two had to have their work rejected for not following directions. Several additional workers had their work accepted but were blocked from completing additional HITs, as their descriptions were too general (for example putting "it is healthy" for all vegetables and fruit). Around 30% of the HITs were completed with at least one description filled out without any copy/pasting. This is consistent with the results found in the pilot batch (see figure 4.3).

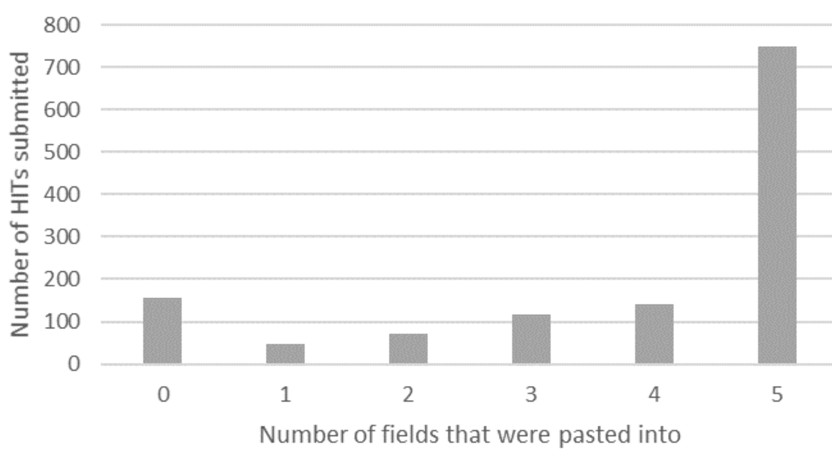

Figure 4.3: This figure shows the percentage of copy pastes used per HIT. Note that for slightly over thirty percent of the HITs, at least one field was manually filled in.

In total, I collected 6,283 descriptions, which was slightly larger than the English corpus of 6,045. After initial analysis, two additional workers were found to have submitted problematic results, and thus their submissions had to be thrown out (see section 4.3.6 for a discussion of this). The average number of descriptions per object after these workers were removed was 318, which was slightly lower than the average for English which was 335 (see figure 4.4). For the Spanish data, this average had been 283, which was low enough that for the analysis I subset the Spanish and English data so they were of comparable sizes. Though the Hindi data was much closer in size to the English one, I decided to also compare subsets, so that the scores could be compared across all three languages.

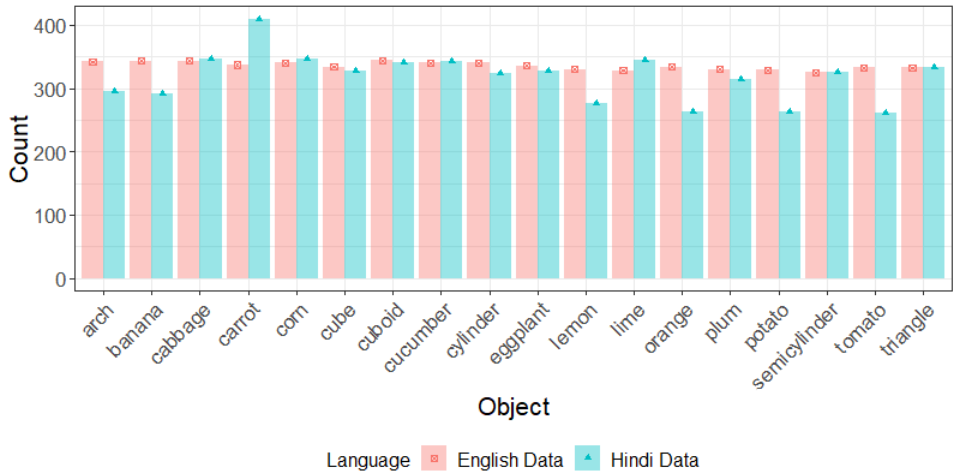

Figure 4.4: This shows the number of descriptions per object for English and Hindi. We can see that in their counts were mostly fairly close, with the exception of carrot, where the larger number of images per instance caused far more Hindi data to be collected than English.

4.3.3 Overall Scores

The average token scores for Hindi versus English are shown in figure 4.5. In general, the scores were pretty similar to English, though the system had slightly higher scores for Hindi shape classifiers, and slightly lower for object and color classifiers. Interestingly, the score differences between English and real Hindi were pretty similar to the score differences with translated Hindi. Once again, stemming appeared to negatively impact the color scores and positively impact the shape scores for both languages. The impact of stemming is examined in closer detail in section 4.3.5.

4.3.4 Stop Words

To dig into the score differences between English and Hindi, I first wanted to identify Hindi stop words the system trained classifiers for. Since nltk did not have a stop word list for Hindi, I chose to use the list found in [36] for the list of generic stop words. The total list of stop words found through this list and those tokens in the lowest 2% by IDF is shown in figure 4.6. I found that the system had learned classifiers for more stop words in Hindi than the other languages. Unlike with Spanish, there were no obviously useful tokens found in the bottom 2% of tokens by IDF. Figure 4.7 shows how removing the generic and low-IDF stop words impacted the scores. One can see that removing stop words barely effected the color classifier scores for Hindi, but had a noticeable positive effect on the object and shape scores.

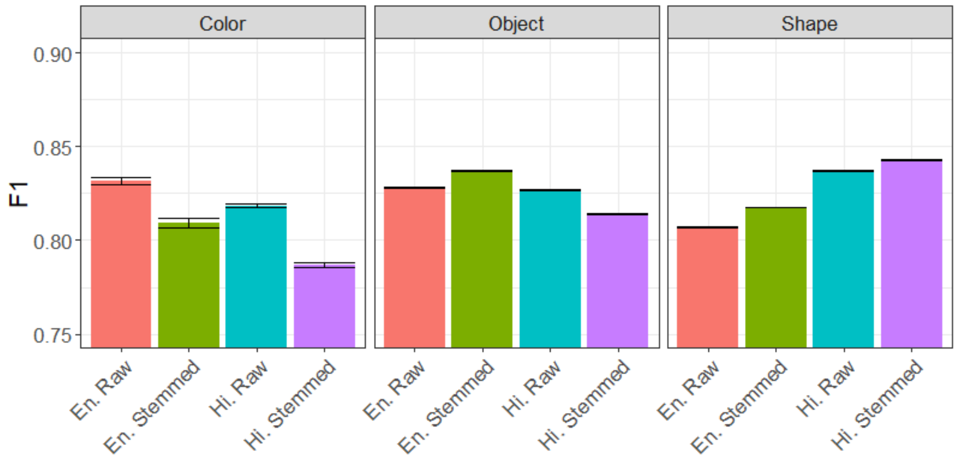

Figure 4.5: The average F1-Scores of all tokens trained in the Hindi and English data with and without stemming. The error bars show the variance in these scores across runs. Note that the variance was on a whole very low.

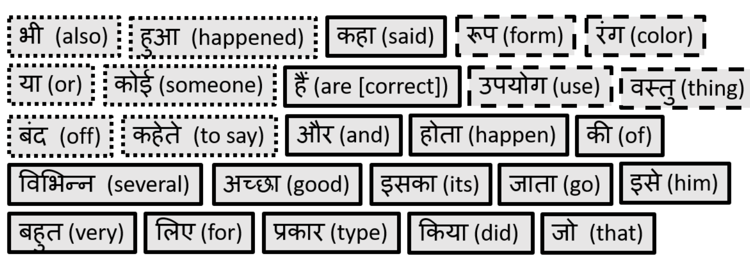

Figure 4.6: These were the tokens learned by the grounded language system on the Hindi data. The dashed border indicates the token was in the bottom 2% by IDF score. The dotted border indicates the token was found in the generic list. The solid border indicates the token appeared in both lists.

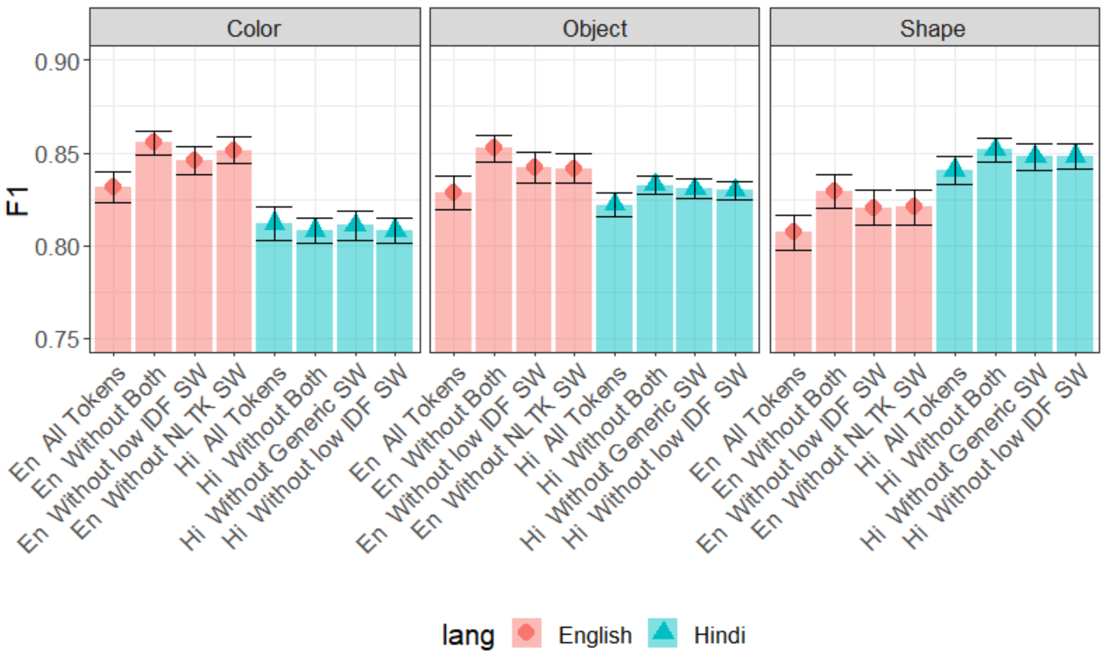

Figure 4.7: The graph demonstrates the impact on the average F1-score of removing generic stop words versus removing the lowest 2% tokens by IDF score for Hindi and English. The error bars show how the variance of these scores among the tokens averaged.

4.3.5 The Impact of Stemming: Colors

In the color section of figure 4.5, stemming appeared to negatively impact the scores. This section seeks to examine this impact more closely.

Context	Token
Masculine singular	नीला
Feminine singular	नीली
Plural	नीले

Figure 4.8: This shows the three ways in which an adjective might be conjugated based on the gender and plurality of the noun.

In Hindi, like in Spanish, nouns can have genders and adjectives must be conjugated to match the gender of the noun. Color-related tokens learned by the system could appear in four possible forms. They could be inflected to match either singular masculine, singular feminine, or plural nouns as shown in figure 4.8. They might also remain the same for all nouns, as is the case with the "red" token. Figure 4.9 shows how often color-related tokens appeared in various forms across the un-stemmed dataset.

As was the case with Spanish, for most colors one form was more popular to use than the others. For Hindi, this was most often the plural form. The GLS learned classifiers for multiple forms of blue, green, and yellow. Figures 4.10 and 4.10 show how the

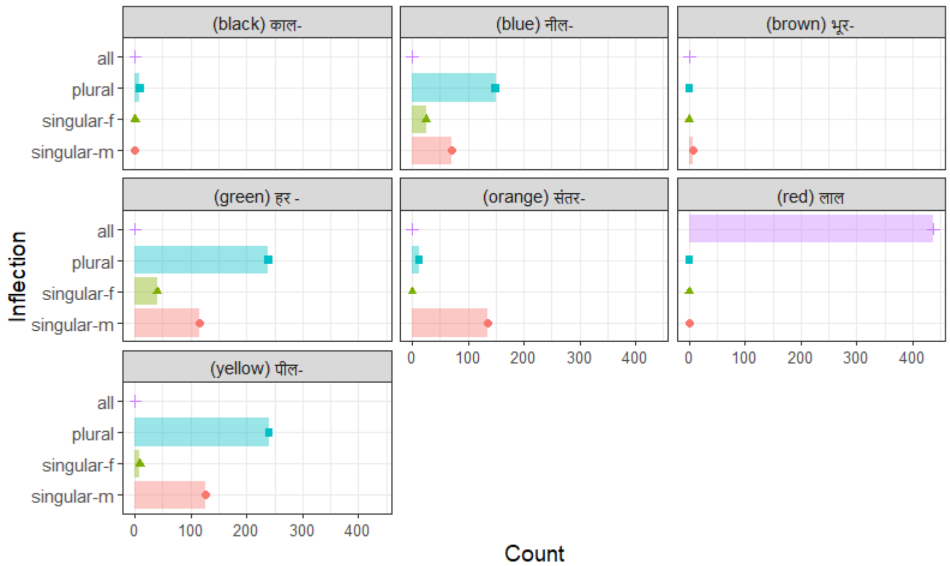

Figure 4.9: This shows the counts for the various conjugations of color words in the Hindi dataset. Note that for green, yellow, and blue, the plural conjugation was used most often, which is very different from the Spanish dataset.

scores and number of positive examples found for the stemmed versions of these colors compared with the average scores of the un-stemmed versions. The positive examples were higher for stemmed green and yellow than for their raw versions, indicating that the positive examples from different inflections were combined. One can see that the stemmed version almost always had lower precision, indicating that combining the examples from different inflections made a classifier that incorrectly labeled more things as positive. This gives some insight into the score difference between the raw and stemmed color classifiers, highlighting a situation where necessary stemming caused a decrease in scores.

4.3.6 Impact of Excluded workers: The “science guys”

In the original examination of the Hindi data, two workers were identified as potentially problematic. Both workers tended to submit long sentences that appeared to be taken straight out of scientific articles about the objects they were being asked to describe. Initially, these descriptions were left in as acceptable noise. However, this ended up having a drastic impact on the learning system. The Hindi system learned 294 tokens, a full 187 additional tokens to what the system would learn without those descriptions. The reason for this seemed to be that the “science guys” always submitted the same long sentences of science jargon for objects of the same type. Since these two workers did many HITs, this introduced a large number of tokens that were tied to specific objects, causing the system to train fairly accurate classifiers for them. This inflated the F1-score for the Hindi tokens as shown in figure 4.12. Since the descriptions submitted were mostly related but not directly describing the objects, and the workers always submitted “I can’t

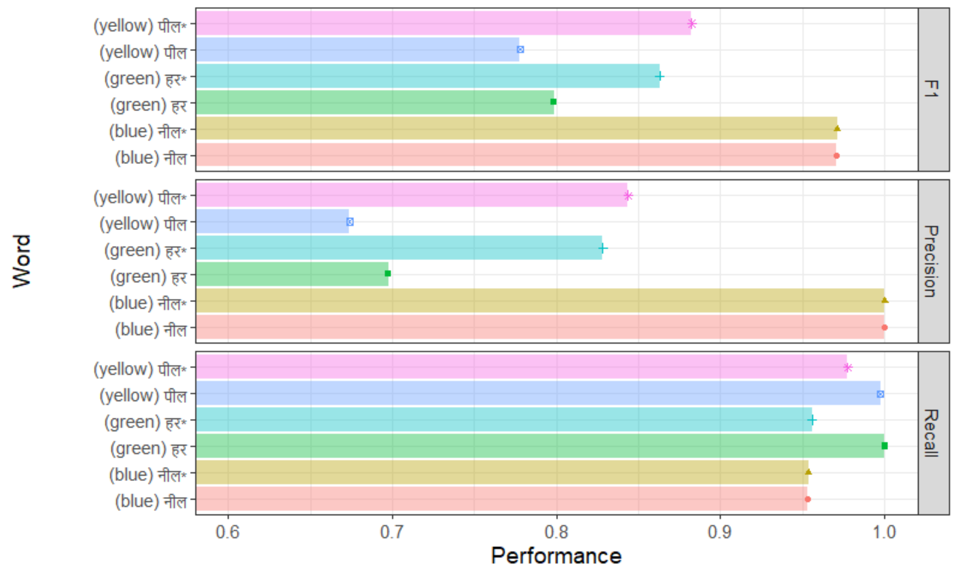

Figure 4.10: This shows the difference between the average of the scores for different conjugations of color words and the stemmed score. Note that the precision dropped in most cases.

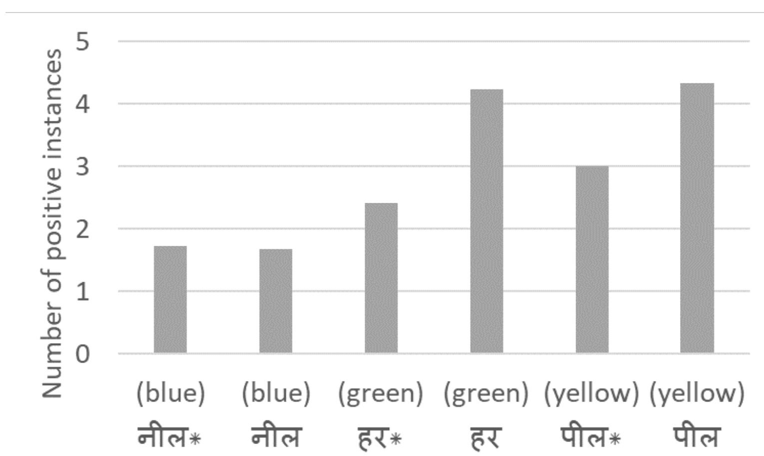

Figure 4.11: This shows the difference between the average of the number of positive instances for different conjugations of color words and the stemmed score. Note that the stemmed versions tended to have more positive instances, showing that stemming did help the classifier get more examples for colors.

tell what the image is" for any object that was not a fruit or vegetable, it was decided that these submissions would not be included in the final analysis. Nonetheless, this is an interesting example of how the grounded language system can be sensitive outliers.

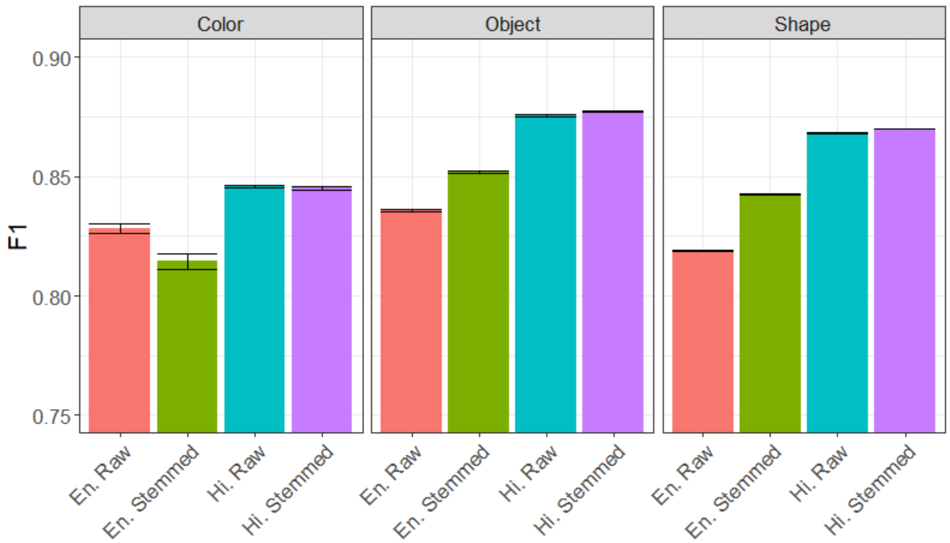

Figure 4.12: The average F1-Scores where the problematic workers have not been removed. Note that Hindi scored much higher than English, partly due to the addition of many tokens only used by those two workers for specific objects.

Chapter 5: Analysis

5.1 The Three Datasets

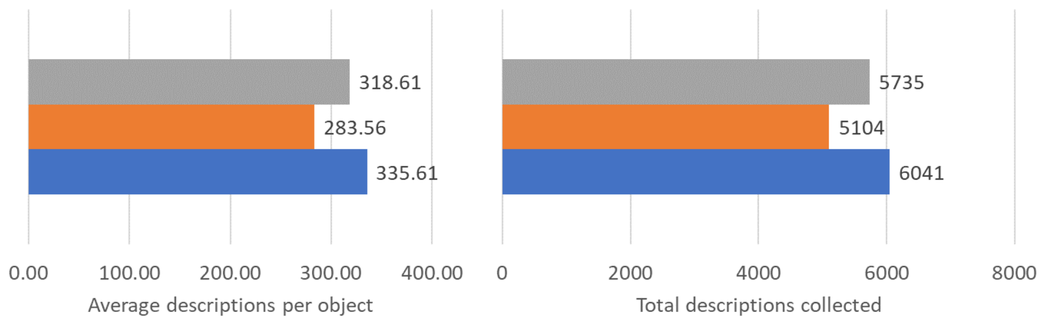

Figure 5.1: The left chart shows the average number of descriptions per object collected for each language. The right graph gives the total number of descriptions collected.

For my thesis, I collected object descriptions in both Spanish and Hindi. I purposefully imitated the procedures used to collect the English data, to maximize the comparability of the system's performance between languages. While enough batches were run to collect a comparable number of descriptions in Hindi and Spanish to the ones in English, both results ended up smaller than English because problematic workers were identified after the fact and their submissions had to be thrown out. Figures 5.2 and 5.1 show that the Hindi dataset ended up much closer in size to the English dataset than the Spanish one did. This is partially because the Spanish data was collected first, so lessons were learned about proper vetting of worker submissions and the correct number of assignments to give per HIT, which minimized the damage from problematic workers in the Hindi data. In general, there were many similarities in the kinds of descriptions given for the objects despite lack of priming. This is likely due to the simplicity of the objects, and the propensity of workers to describe the objects in as few words as possible. Nevertheless, each dataset had its own outliers from workers who gave valid but unusual answers, which contributed noise and meant that the system did not learn exactly the same terms for each language.

5.2 Aggregate Analysis

Previous chapters compared how the grounded language system performed between pairs of languages. This section pulls this together to look at the performance across all

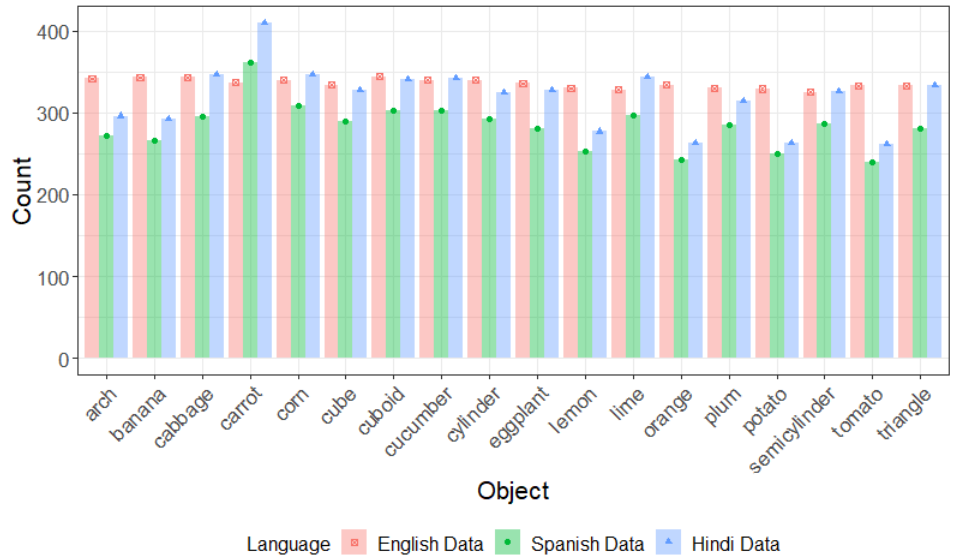

Figure 5.2: The number of descriptions collected per object for the three datasets.

three languages. Figure 5.3 shows the scores across the three languages with and without each of their respective stemmers. One can see that, overall, the system did not appear to have higher scores for one language than the others for all categories, and stemming tended to impact scores in similar ways across languages.

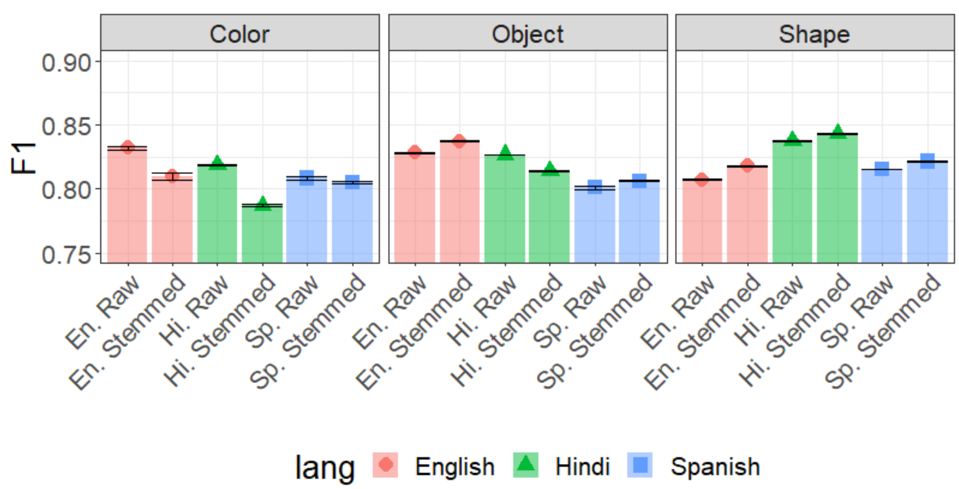

Figure 5.3: Overall scores of all three languages. Note that to allow for fair comparison, the Hindi, English, and Spanish datasets have been subset to have equal amounts of descriptions per instance.

There are two things that are important to note about these scores. Firstly, the values are the averages across all tokens learned, whether or not those tokens belong to that category. Secondly, as has been seen in earlier sections, the value of the F1-score for a particular token does not necessarily correspond to a well-trained classifier. One can see from the Hindi and Spanish analyses in sections 3.5.2 and 4.3.5 that stemming was certainly necessary for tokens specifically relating to colors, due to the high rate of inflection, but the F1-score might decrease with stemming. Spanish had lower object scores than English, but when comparing individual objects where the Spanish version had a lower score (see section 3.5.5) there were times where the examples chosen for the Spanish classifier were more representative of the underlying meaning of the word.

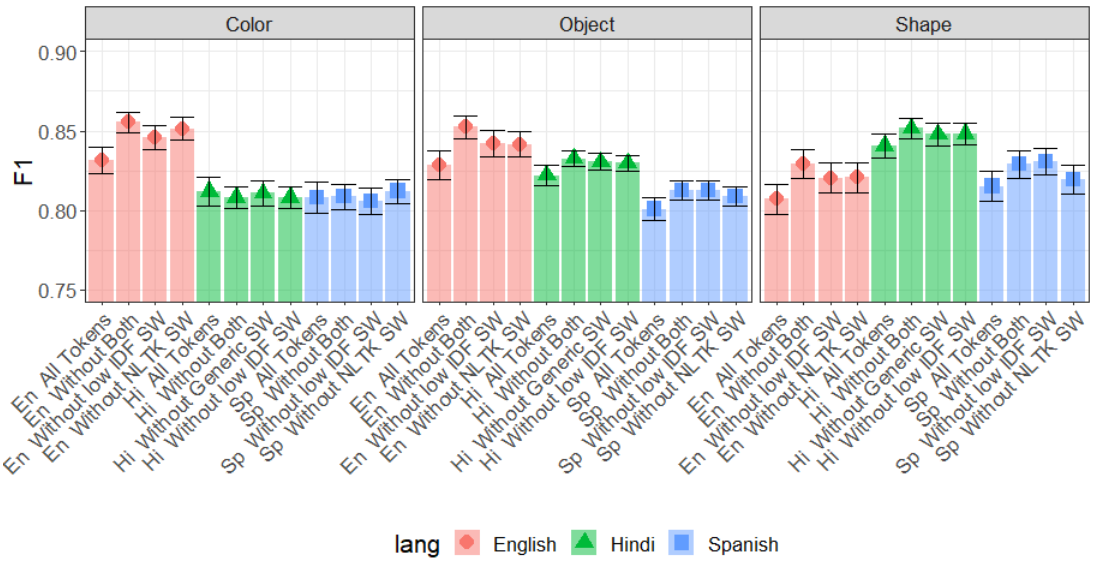

Figure 5.4: The impact of stopwords on the three languages (subset for equal dataset size).

One effect that extended across all three languages was that the removal of both generic stop words and low-IDF-score tokens was helpful for the system. In this, the F1-score was a decent indicator, since the scores for tokens that did not have meaning in the intended task were indeed lower. Figure 5.4 shows that the effect of removing these tokens was consistent for the object and shape classifiers. One can also see from figure 5.5 that other than the two color words unfortunately found for Spanish, there were a lot of similarities in the stop words that were only found through low IDF score in all three languages. All three lists had some token that a person might use when talking about a generic item. This fulfills the intended purpose of finding low-IDF tokens on top of regular stop words, which is to stop the GLS from trying to ground words that don't refer to anything specific.

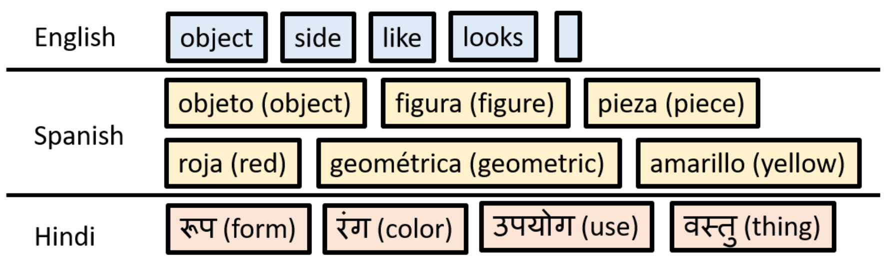

Figure 5.5: Low IDF stop words found for each language that were not also generic stop words.

Chapter 6: Conclusion

6.1 Conclusion

In this thesis I have proposed adaptations to expand an existing unsupervised grounded language acquisition system [8] to work with Spanish and Hindi data. I discussed my initial observations using Google translate, and explored the extent to which these observations could be extended to real data collected through Amazon Mechanical Turk. Through my experiments, I was able to identify several differences between the three languages that should be addressed in the system to attain comparable results. At the same time, I did not find that either Hindi or Spanish did significantly worse than English even before applying additional steps. In general, the existing system with slight modifications seems to work fairly well for all three languages, which is promising when considering its applicability to real-life situations.

6.2 Future Work

This thesis sought to examine the performance of an existing system in the context of a particular data domain. There are many possible ways in which the different aspects of the system or data could be expanded on in the future.

Firstly, the data used for my experiments had a very limited scope, and the images being described were very simple with only one object in each image. A good next step would be to expand the image dataset to include more items and more complex images. The descriptions collected had a lot of redundancy, partially because the same worker might see different angles of the same object many times. For future data collection, it might be beneficial to add the constraint that an individual worker not see the same object more than once or twice.

Secondly, though my work did discuss several preprocessing steps common to NLP (lemmatization, stemming, and stop word removal), there are many more that were left out since they were left out in the original system. In particular, spelling correction could be very helpful in mitigating some of the noise introduced by different workers. Other techniques like entity recognition or part-of-speech tagging could also help to identify meaningful concepts within the descriptions. At the same time, I believe it would be most beneficial to examine these techniques while still keeping multilingual data in mind, as there are likely to be varying levels of resources depending on the language.

One reason why the GLS was able to transfer so well across languages was because it was a fairly simple system. The classifiers were trained using logistic regression, and the tokens were identified without respect to the context they were used in. It would be a simple next step to experiment with different classification algorithms, to see if logistic regression is really the best fit. The bag-of-words approach sidestepped several complications that could have been introduced by different word orderings, but also potentially threw away meaningful information. In the future, it would be interesting to explore more sophisticated ways of tying the language to images. One way would be to use word embeddings to train features of images on features of words, rather than directly mapping image features to words. This could help to utilize commonalities between different tokens. One could also make use of the similar data collected in different languages to tie tokens and phrases together using the images as pivots, as was done in [37]. It would be interesting to examine the possibility of learning groundings of concepts where the tokens span across languages.

The possible future work listed above represents a small fraction of the ways the research presented in this thesis could be expanded. Multilingual Grounded Language Acquisition is a maturing field with many fascinating challenges left unsolved. As we move forward into the future, work in this area will be essential for making future robotic assistants accessible and adaptable, allowing them to be enjoyed by a diverse population.

Appendix A: Other Modifications: Generalizing the System to Choose Positive and Negative Examples

A.1 Introduction

The system to select positive and negative examples of tokens described in section 2.2.2 is a generalization of the approach used in [8] and [26]. This section describes the modifications that I made to the original system described in those papers. These modifications were made after it was noted that the procedure to identify positive and negative instances was different between training and evaluation in the original code base. Positive and negative instances are found twice in the code, first to choose which examples to train each token classifier on, and then to choose which examples to evaluate those classifiers on. In the original training code, instances were positive examples of a token if that token appeared once in descriptions, while a cutoff of 10 was enforced for evaluation. Negative instances were found using a threshold on the cosine similarity value for training, but the evaluation code merely took the last 2/3rds of negative instance candidates by distance. In addition, the training code enforced that a negative instance must be above the threshold of distance from all positive instances of that token, and that the token must never have appeared in any description of that instance, neither of which were enforced in evaluation. I decided to merge these methodologies together and edit the code so both training and evaluation used the same system.

A.2 Finding Positive Instances

In the end, the method for identifying positve instances was kept fairly close to what was already in the code. The only difference was that a cutoff of 5 was implemented for both training and evaluation. This meant that an instance was a positive example of a token if the token had appeared at least five times in descriptions for that instance. The value of 5 was found experimentally, by seeing what positive instances were found for different tokens at various cutoffs.

A.3 Finding Negative Instances

The final method that was decided upon for finding negative instances used elements from both original methods. It was decided that enforcing that a negative instance had never seen the token in any of its descriptions would be a good way to narrow down candidates. In addition, I combined the concepts of choosing some portion of negative instance candidates by distance (instead of using a threshold) with choosing the negative instances that were the furthest away from all positive instances for a token. This was done by first having each positive instance identify the top 2/3rds most different negative instance candidates from itself. The final scores for each negative instance came from the sum of the distances between the candidate and all positive instances where the candidate was in the list for that instance. The negative instance candidates were then sorted by this score, and the top 25% of the candidates were chosen as negative examples. This 25% value was chosen experimentally by examining the F1-scores at different cutoffs in both training and evaluation using the English data (see figure A.1). It must be noted that during these experiments, a token only had to appear once in descriptions of an instance for that instance to be a positive example of the token. This caused the scores to be much lower than those reported for English in other parts of the thesis.

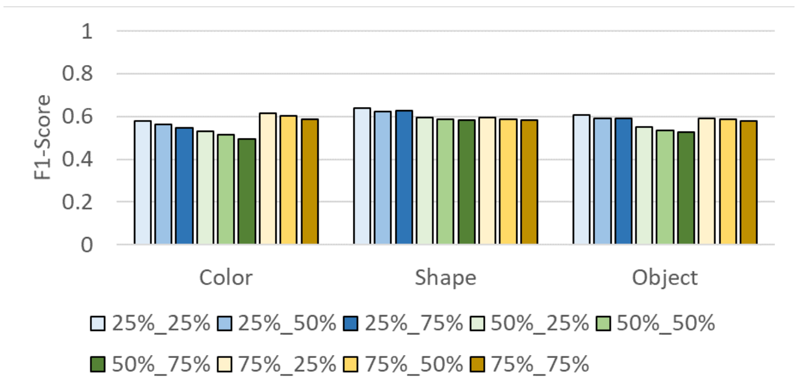

Figure A.1: The F1-scores when token classifiers were trained on different cutoffs of negative examples. The score for 25%,50% means that the top 25% of instances were chosen as negative examples for training, while the top 50% were chosen during evaluation.

Bibliography

[1] Joost Broekens, Marcel Heerink, Henk Rosendal, et al. Assistive social robots in elderly care: a review. *Gerontechnology*, 8(2):94–103, 2009.

[2] Cynthia Matuszek, Nicholas FitzGerald, Evan Herbst, Dieter Fox, and Luke Zettlemoyer. Interactive learning and its role in pervasive robotics. In *ICRA Workshop on The Future of HRI*, St. Paul, MN, 2012.

[3] Raymond Mooney. Learning to connect language and perception. In *Proceedings of the 23rd AAAI Conference on Artificial Intelligence (AAAI)*, pages 1598–1601, Chicago, IL, 2008.

[4] United States Census Bureau, US Department of Commerce. American community survey, 2017. Data collected from 2012-2016.

[5] David Chen, Joohyun Kim, and Raymond Mooney. Training a multilingual sportscaster: Using perceptual context to learn language. *J. Artif. Intell. Res. (JAIR)*, 37:397–435, 01 2010.

[6] 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 (AAAI-17)*, 2017.

[7] 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.

[8] Nisha Pillai and Cynthia Matuszek. Unsupervised selection of negative examples for grounded language learning. In *Proceedings of the 32nd National Conference on Artificial Intelligence (AAAI)*, New Orleans, USA, 2018.

[9] Ranjay Krishna, Yuke Zhu, Oliver Groth, Justin Johnson, Kenji Hata, Joshua Kravitz, Stephanie Chen, Yannis Kalantidis, Li-Jia Li, David A Shamma, et al. Visual genome: Connecting language and vision using crowdsourced dense image annotations. *International Journal of Computer Vision*, 123:32–73, 2017.

[10] 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*, pages 2296–2304, 2015.

[11] Thomas Kollar, Stefanie Tellex, Deb Roy, and Nick Roy. Grounding verbs of motion in natural language commands to robots. In *Experimental Robotics. Springer Tracts in Advanced Robotics*, Springer, Berlin, Heidelberg, 2014.

[12] Chen Yu and Dana H. Ballard. A multimodal learning interface for grounding spoken language in sensory perceptions. In *ACM Transactions on Applied Perception*, pages 57–80, 2004.

[13] Jake Brawer, Olivier Mangin, Alessandro Roncone, Sarah Widder, and Brian Scassellati. Situated human–robot collaboration: predicting intent from grounded natural language. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 827–833, 2018.

[14] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. In *CoRR*, 2016.

[15] Jawharah Alasmari, J Watson, and ES Atwell. A comparative analysis between arabic and english of the verbal system using google translate. In *Proceedings of IMAN'2016 4th International Conference on Islamic Applications in Computer Science and Technologies*, Khartoum, Sudan, 2016.

[16] Ekta Gupta and Shailendra Shrivastava. Analysis on translation quality of english to hindi online translation systems- a review. In *International Journal of Computer Applications*, 2016.

[17] Hadis Ghasemi and Mahmood Hashemian. A comparative study of" google translate" translations: An error analysis of english-to-persian and persian-to-english translations. *English Language Teaching*, 9:13–17, 2016.

[18] Joachim Daiber, Max Jakob, Chris Hokamp, and Pablo N Mendes. Improving efficiency and accuracy in multilingual entity extraction. In *Proceedings of the 9th International Conference on Semantic Systems*, pages 121–124. ACM, 2013.

[19] Michael Gamon, Carmen Lozano, Jessie Pinkham, and Tom Reutter. Practical experience with grammar sharing in multilingual nlp. *From Research to Commercial Applications: Making NLP Work in Practice*, 1997.

[20] Craig Macdonald, Vassilis Plachouras, Ben He, Christina Lioma, and Iadh Ounis. University of glasgow at webclef 2005: Experiments in per-field normalisation and language specific stemming. In *Workshop of the Cross-Language Evaluation Forum for European Languages*, pages 898–907, 2005.

[21] Massimo Poesio, Olga Uryupina, and Yannick Versley. Creating a coreference resolution system for italian. In *International Conference on Language Resources and Evaluation*, Valletta, Malta, 2010.

[22] Joonatas Wehrmann, Willian Becker, Henry EL Cagnini, and Rodrigo C Barros. A character-based convolutional neural network for language-agnostic twitter sentiment analysis. In *2017 International Joint Conference on Neural Networks (IJCNN)*, pages 2384–2391. IEEE, 2017.

[23] Emily M Bender. Linguistically naïve!= language independent: why nlp needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, 2009.

[24] Joseph Le Roux, Benoit Sagot, and Djamé Seddah. Statistical parsing of spanish and data driven lemmatization. In *ACL 2012 Joint Workshop on Statistical Parsing and Semantic Processing of Morphologically Rich Languages (SP-Sem-MRL 2012)*, pages 6–pages, 2012.

[25] Ferran Pla and Lluís-F Hurtado. Political tendency identification in twitter using sentiment analysis techniques. In *Proceedings of COLING 2014, the 25th international conference on computational linguistics: Technical Papers*, pages 183–192, 2014.

[26] 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.

[27] Liefeng Bo, Kevin Lai, Xiaofeng Ren, and Dieter Fox. Object recognition with hierarchical kernel descriptors. In *Computer Vision and Pattern Recognition*, 2011.

[28] 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: Research Topics and Applications*, pages 167–192, 2013.

[29] Martin F. Porter. Snowball: A language for stemming algorithms. *Retrieved March*, 1, 01 2001.

[30] Ananthakrishnan Ramanathan and Durgesh D Rao. A lightweight stemmer for hindi. In *the Proceedings of EACL*, 2003.

[31] India: Office of the Registrar General & Census Commissioner. Comparative speakers' strength of scheduled languages -1971, 1981, 1991 and 2001, 2015. Archived 2007-11-30.

[32] Shilpi Srivastava, Mukund Sanglikar, and DC Kothari. Named entity recognition system for hindi language: a hybrid approach. *International Journal of Computational Linguistics (IJCL)*, 2(1):10–23, 2011.

[33] Chetana Thaokar and Latesh Malik. Test model for summarizing hindi text using extraction method. In *2013 IEEE Conference on Information & Communication Technologies*, pages 1138–1143. IEEE, 2013.

[34] Vishal Gupta. Hybrid algorithm for multilingual summarization of hindi and punjabi documents. In *Mining Intelligence and Knowledge Exploration*, pages 717–727. Springer, 2013.

[35] Manjula Subramaniam and Vipul Dalal. Test model for rich semantic graph representation for hindi text using abstractive method. *International Research Journal of Engineering and Technology (IRJET)*, 2(2), 2015.

[36] Sifatullah Siddiqi and Aditi Sharan. Construction of a generic stopwords list for hindi language without corpus statistics. *International Journal of Advanced Computer Research*, 8(34):35–40, 2018.

[37] Spandana Gella, Rico Sennrich, Frank Keller, and Mirella Lapata. Image pivoting for learning multilingual multimodal representations. *arXiv preprint arXiv:1707.07601*, 2017.