

CHAPTER 5

METHOD

5.1 Introduction

In this chapter, I explain my system for rating and ranking responses automatically, where the goal is to approximate the benchmark rankings described in Chapter 4. The data-driven method used to analyze picture description task (PDT) responses throughout this dissertation represents an evolution from my own previous rule-based methods. In this chapter, I summarize my earlier work and the lessons I learned there, then lay out the current approach.

In short, my first approach was heavily rule-based and relied on strict matching with a pre-established set of acceptable responses. This found moderate success, leading to the improved current approach, which is data-driven and relies on more flexible methods of comparison.

5.2 Previous approach: Rule-based semantic triple matching

This section summarizes relevant work first presented in King and Dickinson (2013) and King and Dickinson (2014); please see those papers for deeper discussions. However, please note that some experiments included in those papers are omitted here, and thus some results involving the averaging or ranking of scores may differ slightly.

Like the current research, my previous work focused on analyzing English non-native speaker (NNS) responses to a PDT by comparison with native speaker (NS) responses. I was initially unsure if such a task would be within reach for a single researcher using off-the-shelf tools, so this study sought to uncover the challenges involved and determine whether variation in the form and content of responses could be manageable.

5.2.1 Rule-based matching method

My earlier method was inspired by research from areas such as sentiment analysis, topic modeling, and content assessment that used rule-based approaches to extract important elements from dependency-parsed text (Nastase et al., 2006; Bailey and Meurers, 2008; Di Caro and Grella, 2013). My idea was to extract a *verb(subj,obj)* triple from each sentence. Each NNS triple could be compared against the list of NS triples for a match; a NNS response with a NS triple match would be labeled “correct,” while a non-match would be labeled “incorrect.”

For these experiments, I chose or developed PDT images that present an event that I believed to be transitive in nature and likely to elicit responses with an unambiguous subject, verb and object.


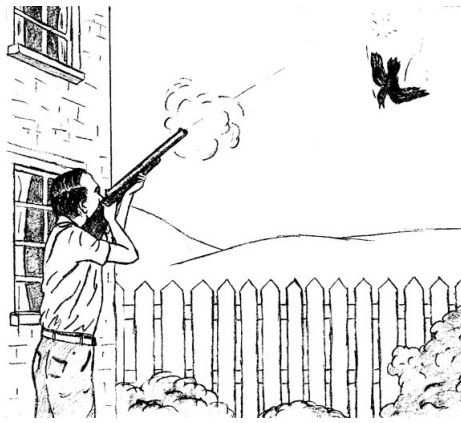
	
Response (L1)	Response (L1)
He is droning his wife pitcher. (Ar)	The man killing the beard. (Ar)
The artist is drawing a pretty women. (Ch)	A man is shutting a bird. (Ch)
The artist is painting a portrait of a lady. (En)	A man is shooting a bird. (En)
The painter is painting a woman's paint. (Sp)	The man shouted the bird. (Sp)

Figure 5.1: Example items from the previous study showing responses from native speakers of Arabic (Ar), Chinese (Ch), English (En) and Spanish (Sp).

The PDT consisted of 10 items (8 line drawings and 2 photographs) intended to elicit

a single sentence each; an example is given in Figure 5.1. Participants were asked to view the image and “describe the action in either past or present tense.” Responses were typed by the participants themselves in a computer lab with spell checking disabled. I collected responses from 53 participants for a total of 530 sentences. There were 14 NSs (non-linguistics undergraduate and graduate students) and 39 NNSs (university students enrolled in English as a Second Language courses).

My process was to parse a NNS sentence into a dependency representation and then extract and lemmatize a semantic triple from this parse to compare to a set of gold standard semantic triples similarly derived from the NS responses. As in my current research, I used the Stanford Parser for this task, trained on the Penn Treebank (de Marneffe et al., 2006; Klein and Manning, 2003).¹

Type	Description	Example	NS	NNS
A	Simple declar. trans.	The boy is kicking the ball.	117	286
B	Simple + preposition	The boy played with a ball.	5	23
C	No tensed verb	Girl driving bicycle.	10	44
D	No tensed verb + prep	Boy playing with a ball.	0	1
E	Intransitive (No object)	A woman is cycling.	2	21
F1	Passive	An apple is being cut.	4	2
F2	Passive with agent	A bird is shot by a man.	0	6
Ax	Existential A or C	There is a boy kicking a ball.	0	0
Bx	Existential B or D	There was a boy playing with a ball.	0	0
Ex	Existential E	There is a woman cycling.	0	0
F1x	Existential F1	There is an apple being cut.	0	1
F2x	Existential F2	There is a bird being shot by a man.	0	0
Z	All other forms	The man is trying to hunt a bird.	2	6

Table 5.1: Sentence type examples, with distributions of types for native speakers (NS) and non-native speakers (NNS)

I manually categorized the 530 sentences in the dataset into 11 types plus one catch-all category, as shown in Table 5.1. I established these types because each one corresponds to a basic sentence structure and thus has consistent syntactic features, leading to predictable patterns in the dependency parses. A sentence type indicates that the subject, verb, and ob-

¹<http://nlp.stanford.edu/software/lex-parser.shtml>

ject can be found in a consistent place in the parse, such as under a particular dependency label. For simple transitive sentences (type *A* in Table 5.1), for example, the dependents labeled *nsubj*, *root*, and *dobj* pinpoint the necessary information. Thus, the patterns for extracting semantic information—in the form of *verb(subj,obj)* triples—reference particular Stanford typed dependency labels, part-of-speech (POS) tags, and locations relative to word indices (see Figure 5.2).

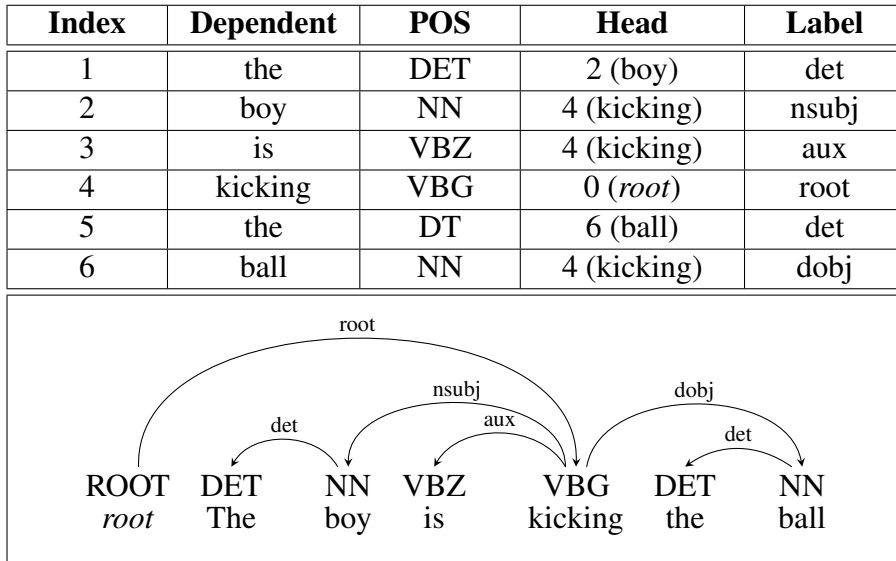


Figure 5.2: The dependency parse of an example NNS response in a standard format (CoNLL) and the corresponding visual representation.

Determining the sentence type was accomplished by arranging a small set of binary decisions into a tree, as shown in Figure 5.3. This decision tree checks for the presence of various dependency labels. The extraction rules for the particular sentence type were then applied to obtain the semantic triple. Finally, for each NNS response, the resulting triple was lemmatized and checked against the gold standard list of lemmatized NS triples. Ideally, each acceptable response will find a match, and unacceptable responses will not.

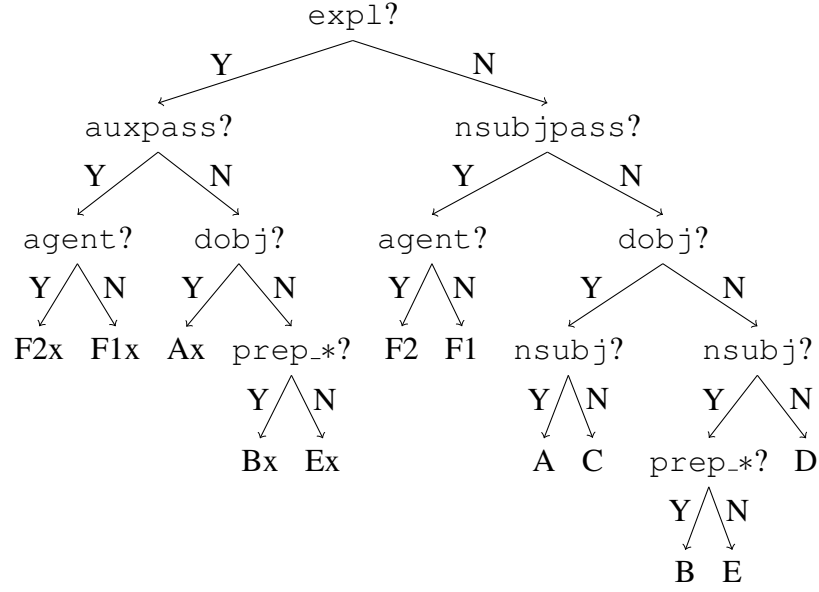


Figure 5.3: Decision tree for determining sentence type and extracting semantic triple based on the presence of syntactic dependency labels

5.2.2 Rule-based matching results

Evaluating this work required addressing two major questions. First, how accurately does this approach extract semantic information from potentially innovative sentences? Second, how many semantic forms does one need in order to capture the variability in meaning in NNS sentences? I operationalized this second question by asking how well a given set of NS semantic forms functions as a gold standard.

An accurate extraction was defined as one in which the extraction rules chose the desired subject, verb, and object given the sentence at hand and without regard to the PDT image. Accuracy was 92.3% for NNS responses and 92.9% for NS responses. I attribute the high extraction scores to the constrained nature of the task and the relatively small range of sentence types it elicits. As seen in Table 5.1, only three sentence types account for more than 90% of all responses.

Assessing the coverage of NNS forms required first manually determining which extracted triples *should* be matched given a hypothetical perfect gold standard set of triples.

To separate the problem of coverage from extraction, I first removed any incorrectly extracted triples from the NNS set and the NS gold standard. I then labeled the remaining responses as “appropriate” or “inappropriate.” Note that this label applies to the response, not the triple. My annotation guidelines for this labeling were minimal: *Given the prompt, would the response be acceptable to most English speakers?*

I called an appropriate NNS triple found in the gold standard set a **true positive (TP)** (i.e., a correct match), and an appropriate NNS triple *not found* in the gold standard set a **false negative (FN)** (i.e., an incorrect non-match), as shown in Table 5.2. I used standard terminology here (TP, FN), but because this was an investigation of what *should be* in the gold standard, these were considered false negatives and not false positives. To address the question of how many NS sentences are needed to obtain good coverage, **coverage** was defined as recall: $TP/(TP+FN)$. I reported 23.5% coverage for unique triple *types* and 51.0% coverage for triple *tokens*.

		NNS	
		+	−
NS	Y	TP	FP
	N	FN	TN

Table 5.2: Contingency table comparing presence of NS forms (Y/N) with appropriateness (+/−) of NNS forms

I defined an inappropriate NNS triple (i.e., a content error) *not found* in the gold standard set as a **true negative (TN)** (i.e., a correct non-match). **Accuracy** based on this gold standard—assuming perfect extraction—is defined as $(TP+TN)/(TP+TN+FN)$.² I reported 46.4% accuracy for types and 58.9% accuracy for tokens.

The immediate lesson taken from this was this: given a strict matching approach, NS data alone does not make for a sufficient gold standard, in that many appropriate NNS answers are not counted as correct. I explored expanding the set of NS triples by separating

²Accuracy is typically defined as $(TP+TN)/(TP+TN+FN+FP)$, but false positives (FPs) are cases where an incorrect NNS response was in the gold standard; by removing errors from the NS responses, I prevented this scenario (i.e., $FP=0$).

individual subjects, verbs and objects from NS triples and recombining them into the various possible combinations. However, this recombination generates a lot of nonsensical triples and degrades the gold standard. Consider, for example, *do(woman,shirt)*—an incorrect triple derived from the correct NS triples, *wash(woman,shirt)* and *do(woman,laundry)*. This could be improved somewhat by evaluating new combinations with a language model, but this would both complicate the approach and diverge from my vision of content analysis driven by real speaker behavior. Instead, my current work has attempted to improve coverage by prompting NSs to give an initial PDT response, followed by a second alternative.

A related concern was that, even when only examining cases where the meaning is literally correct, NNSs produced a wider range of forms to describe the prompts than NSs. For example, for a picture showing what 100% of NSs described as a *raking* action, many NNSs referred to a man *cleaning* an area. Literally, this may be true, but it does not align with a NS gold standard. This behavior was expected, given that learners are encouraged to use words they know to compensate for gaps in their vocabularies (Agustín Llach, 2010). This also parallels the observation in SLA research that while second language learners may attain native-like grammar, their ability to use pragmatically native-like language is often much lower (Bardovi-Harlig and Dörnyei, 1998). These findings highlighted the need for a more flexible approach.

Moreover, evaluating this strict matching approach required an annotator to decide whether a given response is correct or incorrect. Partial matching is not allowed; this is an inherent weakness of the approach, because while a complete triple gives some indication of the meaning of the sentence, any single element of the triple taken alone does not provide enough context to indicate meaning. This inflexibility means that using this approach would effectively require the manual curation of a robust gold standard set of acceptable responses, which is counter to my goal of producing an approach that can be expanded to new PDT items simply by crowdsourcing a gold standard from NSs.

I followed up this work with a modification that included language models and spell checking tools to attempt to identify and fix misspellings that lead to downstream problems (King and Dickinson, 2014). I omit this discussion because it is not applicable to the current work; I now take a simpler approach—respondents use spell checking during the task. This is because in most contexts where my system would be used, like a language tutoring application or game, spelling instruction is not the objective, and a built-in spell checker would likely be available. Moreover, omitting this step removes a layer of analysis—and importantly, a potential source of errors—and allows the research to focus more directly on meaning.

5.3 Recent work: Generalized, similarity-based approaches

In subsequent work (King and Dickinson, 2016), I began looking for a “sweet spot” of semantic analysis (cf. Bailey and Meurers, 2008) for image-based learner productions. I applied new methods to the same dataset, and the current dissertation research applies a refined version of these methods to the new dataset discussed in Chapters 3 and 4. In particular, using available NLP tools, I moved away from discrete representations of correctness in the form of a gold standard set of semantic triples to a more continuous notion of correctness using a model comprised of smaller, overlapping pieces of information (section 5.3.1). This obviates the need for a rule-based extraction of semantic triples, which must be customized for a limited range of expected sentence types. It also allows for graded scoring of results, meaning that a response is not outright rejected because only one argument of a triple is not found. On the other hand, it means that the system does not provide a discrete “thumbs up” or “thumbs down” decision for each response, which means that downstream tasks like assessment, providing feedback or effecting game outcomes would require more careful consideration of what to do with the system output.

I should note, in this context, that I am discussing semantic analysis given a reliable set of NS sentences. Image processing tasks often rely on breaking images into semantic

primitives (see, e.g., Gilberto Mateos Ortiz et al., 2015, and references therein), but for NNS data, I want to ensure that I can account not just for correct semantics (the *what* of a picture), but natural expressions of the semantics (the *how* of expressing the content). In other words, the goal is to reason about meaning based on specific linguistic forms.

A second issue regarding content analysis, beyond correctness, stems from using an incomplete gold standard. The productive nature of language means that a sentence can be expressed in countless ways, and thus a gold standard can never really be “complete.” Examining the degree of this variability both for NSs and NNSs is necessary to determine whether a crowd-sourced gold standard can account for a sizable portion of test responses. Additionally, it can offer insights into theoretical research on variability in learner language (cf. Ellis (1987), Kanno (1998)). With regard to the current work, analyzing variability can also help determine the most effective parameters for an NLP system for image based responses. By applying a set of new approaches to my older data, I was able to select the best performing approaches for refinement and further study with the larger, current dataset.

5.3.1 Generalizing the methods

The previous work assumed that the assessment of NNS responses involves determining whether a NS gold standard set contains the same semantic triple that the NNS produced, i.e., whether a triple is *covered* or *non-covered*. In such a situation the gold standard need only be comprised of *types* (not *tokens*) of semantic triples. Here the gold standard is comprised of the small set of NS responses—only 14. This means that exact matching is going to miss many cases, and indeed as discussed in Section 5.2.2, coverage was only 51%. Even with a much larger gold standard, we can expect responses to follow Zipf’s law; a sample of language data will always be incomplete because it will not contain all “long tail,” low-frequency phenomena.

Additionally, relying on matching of triples limits the utility of the method to specific

semantic constraints, namely transitive sentences. By dropping the exact matching approach and instead comparing the frequencies of elements in the NNS response with those of the gold standard, I moved into a gradable, or ranking, approach to the analysis. For this reason, I shift terminology here from NS *gold standard* to NS *model*. A gold standard is roughly akin to an answer key, which was appropriate for my strict triple-matching approach. A model is typically a richer data structure containing statistics from observations of known correct data. This distinction and its relevance will be more apparent with the discussion of response representations in Section 5.3.2.

My goal is to emphasize the degree to which a NNS response conveys the same meaning as the set of NS responses, necessitating an approach which can automatically determine the importance of a piece of information among the set of NS responses. This required two major decisions: 1) how to **represent** each response as a set of sub-elements, and 2) how exactly to **score** these sub-elements via comparison with the NS data. In Section 5.3.2, I detail how I represented the information and in Section 5.3.3, I discuss comparing NNS information to NS information, which allowed me to rank responses from most to least similar to the NS model. In Section 5.3.4, I outline the system parameters that I combined to generate scores, and in Section 5.3.5 I present and interpret the results of the various settings.

5.3.2 Representing responses

I first represented each NNS response as a list of dependencies taken directly from the parse. This eliminates the complications of extracting semantic triples from dependency parses, which only handled a very restricted set of sentence patterns and resulted in errors in 7–8% of cases, as discussed in Section 5.2.2. Operating directly on individual dependencies from the overall tree also means the system can allow for “partial credit;” it distributes the matching over smaller, overlapping pieces of information rather than a single, highly specific triple. The words within the dependencies were lemmatized using the pre-trained

Stanford CoreNLP lemmatizer (Manning et al., 2014). Lemmatizing was used here to minimize data sparsity. For example, the main verb in the forms *kicks*, *kicked*, *has kicked* and *is kicking* was reduced to *kick* in all cases, increasing the likelihood of finding matches.

The boy is kicking the ball.				
ldh (labeled)	xdh (unlab.)	lxh	ldx	xdx (word)
det(the,boy)	<i>x</i> (the,boy)	det(<i>x</i> ,boy)	det(the, <i>x</i>)	<i>x</i> (the, <i>x</i>)
nsubj(boy,kick)	<i>x</i> (boy,kick)	nsubj(<i>x</i> ,kick)	nsubj(boy, <i>x</i>)	<i>x</i> (boy, <i>x</i>)
aux(is,kick)	<i>x</i> (is,kick)	aux(<i>x</i> ,kick)	aux(is, <i>x</i>)	<i>x</i> (is, <i>x</i>)
root(kick,root)	<i>x</i> (kick,root)	root(<i>x</i> ,root)	root(kick, <i>x</i>)	<i>x</i> (kick, <i>x</i>)
det(the,ball)	<i>x</i> (the,ball)	det(<i>x</i> ,ball)	det(the, <i>x</i>)	<i>x</i> (the, <i>x</i>)
dobj(ball,kick)	<i>x</i> (ball,kick)	dobj(<i>x</i> ,kick)	dobj(ball, <i>x</i>)	<i>x</i> (ball, <i>x</i>)

Table 5.3: Given the example sentence above, the updated approach represents responses in the dependency formats shown: *ldh* (for *label*, *head*, *dependent*; i.e., labeled dependencies), *xdh* (unlabeled dependencies), *lxh* (label+head), *ldx* (label+dependent), or *xdx* (word, or more technically, *dependent*).

Next, I obtained five different representations from the lemmatized dependencies, as shown in Table 5.3. I refer to this variable as **term representation**, in keeping with *term* frequency-inverse document frequency, discussed in Section 5.3.3. The five term representations are then variations on dependencies. The full form is the **labeled dependency** and includes the **label**, **dependent** and **head**, so I refer to it as **ldh**. The remaining four forms abstract over either the label, dependent and/or head. I refer to these forms as **xdh** (i.e., unlabeled dependency), **lxh** (label+head), **ldx** (label+dependent), and **xdx** (dependent only, roughly equivalent to *word* in a bag-of-words approach).

The goal in choosing these five representations was to find the optimal combination of dependency features and the right level of detail to obtain the best system performance. The bag-of-words representation was implemented largely to provide a baseline by which to compare the others.

This processing was applied to the collection of NS responses as well. For each item, the dependencies from all NS responses was pooled into a single flat list—a “bag-of-dependencies.” From this list, a copy in each of the five term representations was produced,

allowing for comparison with the corresponding NNS data.

5.3.3 Scoring responses

Taking the five term representations, my next step was to score them in a way which ranks responses from most to least appropriate. I devised four scoring approaches, each using one of two methods to **weight** response terms combined with one of two methods to **compare** the weighted NNS terms with the NS data.

For weighting, I used either a simple relative frequency measure or one based on *term frequency-inverse document frequency* (*tf-idf*) (Manning et al., 2008, ch. 6). Most commonly, a *term* for *tf-idf* purposes would be a *word*. However, *term* here refers to a single syntactic dependency, and this is a central conceit of both the study in reference and the larger dissertation: the dependency, which captures aspects of semantic and syntactic relationships, is an ideal atomic unit for evaluating meaning by comparing distributions in crowdsourced data. As discussed, the dependencies were represented in one of the five term representations—some combination of label, dependent and head. For the NNS data, each response was treated as a document. For the NS data, the entire collection of NS responses was treated as one document. The relative frequency measure was simply the token count of a given term in a document normalized by the total count of tokens in the document.

I used *tf-idf* as a measure of a term’s importance with the expectation that it would reduce the impact of semantically less important terms—e.g., determiners like *the*, frequent in the responses, but mostly unnecessary for evaluating the semantic contents—and to upweight terms which may be salient but infrequent, e.g., only used in a handful of NS sentences. For example, for an item depicting a man shooting a bird (see Table 5.4 and Figure 5.1), of 14 NS responses, 12 described the subject as *man*, one as *he* and one as *hunter*. Since *hunter* is relatively infrequent in English, even one instance in the NS responses should get upweighted via *tf-idf*, and indeed that was the effect; in the bag-of-

words approach, the term *hunter* is weighted among the highest, and the same is true for dependencies containing the word *hunter* among the other term representations. This is valuable, as numerous NNS responses used *hunter*.

Calculating tf-idf relies on both *term frequency* (tf) and *inverse document frequency* (idf). Term frequency is simply the raw count of a term within a document. Inverse document frequency is derived from some reference corpus of documents, and it is based on the notion that appearing in more documents makes a term less informative with respect to distinguishing between documents. The formula is in (3) for a term t , where N is the number of documents in the reference corpus, and df_t is the number of documents featuring the term ($idf_t = \log \frac{N}{df_t}$). A term appearing in fewer documents will thus obtain a higher idf weight, and this should readjust frequencies based on semantic importance.

$$(3) \quad tfidf(t) = tf_{GS} \log \frac{N}{df_t}$$

After this frequency counting or tf-idf weighting, the scores were then either **averaged** to yield a response score, or NNS term weights and NS term weights were treated as vectors and the response score was the **cosine distance** between them. This yields the four approaches:

Frequency Average (FA). Within the set of NS responses, the relative frequency of each term is calculated. The NS *model* here is simply each NS term and its relative frequency. Each term in the NNS response is then given a score equal to its frequency in the NS model; terms missing from the NS model are scored zero. The response score is the average of the term scores, with higher scores closer to the NS model.

Tf-idf Average (TA). This involves the same averaging of term scores as with approach FA, but here the term scores are the tf-idf scores. The NS model here is thus each NS term and its tf-idf score.

Frequency Cosine (FC). Each term score is taken as its relative frequency calculated within its document: either the NS response set or the single NNS response. The NS model is then the set of all NS terms and their scores. The term scores are then treated as vectors—one vector of the NS term scores (i.e., the NS model here)—and one vector of the NNS term scores. Each vector is an ordered list of term scores for each term observed in either the NS document or the NNS document. In other words, each vector represents term scores for the sorted union set of NS and NNS terms. Naturally, many of the term scores are zero for the much shorter NNS document. The response score is the cosine distance between the vectors, with lower scores being closer to the NS model.

Tf-idf Cosine (TC). This involves the same distance comparison as with approach FC, but now the term scores in the vectors are tf-idf scores. The NS model here is thus the vector representing the union set of terms for the NS and NNS document, populated with the tf-idf scores from the NS document.

In many natural language processing scenarios where high dimensional vectors are involved, such as sentence encoders or word embeddings, methods for dimensionality reduction are employed (Devlin et al., 2018; Mikolov et al., 2013). This improves efficiency by reducing the storage and computing power needed. In my FC and TC approaches, however, the number of word types (and in turn term types) remained small enough that the raw vectors representing dependencies' tf-idf scores can be processed easily with an ordinary PC. Not only does this simplify the task, it means that the process remains transparent. There are no transformers or attention mechanisms to produce compressed and unexplainable representations. If desired, each sentence vector can be examined value by value, where each number maps to a real syntactic dependency. This is important because it leaves the door open for meaningful feedback on each response. For example, one might choose to identify the most salient dependencies in the NS model and use them to guide an ICALL user from a low scoring response to a better response.

5.3.4 System Parameters

I ran a total of 30 experiments, with each representing a combination of these system parameters for processing responses (see also Table 5.6):

Term representation As discussed in section 5.3.2, the terms can take one of five representations: **ldh**, **x dh**, **l x h**, **l d x**, or **x d x**.

Scoring approach. As discussed in section 5.3.3, the NNS responses can be compared with the NS models via approaches **FA**, **TA**, **FC**, or **TC**.

Reference corpus. The reference corpus for deriving tf-idf scores can be either the **Brown Corpus (Brown)** (Kucera and Francis, 1967) or the Wall Street Journal (**WSJ**) Corpus (Marcus et al., 1993). This is only relevant to approaches TA and TC, so na is used where approaches FA and FC are involved. The corpora are divided into as many documents as originally distributed (WSJ: 1640, Brown: 499). WSJ is larger, but Brown has the benefit of containing more balance in its genres (vs. newstext only). Considering the narrative nature of PDT responses, a reference corpus of narrative texts would be ideal, but as no such reliably parsed corpus is available, I chose the widely used, pre-parsed Brown and WSJ corpora. The corpora were converted from their standard dependency parse format to each of the five term representations used in order to be compatible with the NS and NNS data for tf-idf.

5.3.5 Experiments and Results

Evaluation metrics

I ran 30 response scoring experiments, each with different system settings (section 5.3.4). Within each experiment, I ranked the 39 scored NNS responses from least to most similar to the NS model.

For assessing these settings themselves, I relied on the past annotation, which counted unacceptable responses as errors (see section 5.2.2). As the lowest numerical rank indicates the greatest distance from the NS model, a good system setting should position the unacceptable responses among those with the lowest rankings. To evaluate this discriminatory power, I used **(mean) average precision ((M)AP)** (Manning et al., 2008, ch. 8).

Rank	Score	Sentence	Error
1	1.000	she is hurting.	1
	1.000	man mull bird	1
3	0.996	the man is hurting duck.	1
4	0.990	he is hurting the bird.	1
11	0.865	the man is trying to hurt a bird	1
12	0.856	a man hunted a bird.	0
17	0.775	the bird not shot dead.	1
18	0.706	he shot at the bird	0
19	0.669	a bird is shot by a un	1
20	0.646	the old man shooting the birds	0
37	0.086	the old man shot a bird.	0
38	0.084	a old man shot a bird.	0
39	0.058	a man shot a bird	0
Average Precision: 0.75084			

Table 5.4: Example item rankings for the best system setting (TC, Brown Corpus, and labeled dependencies (ldh) based on average precision scores. Note that not all 39 responses are shown.

For average precision (AP), one calculates the precision of error detection at every point in the ranking, lowest to highest. Consider Table 5.4, which presents an excerpt of ranked sentence responses for one PDT item. The precision for the first cut-off (1.000) is 1.0, as two responses have been identified, and both are errors ($\frac{2}{2}$). At the 11th- and 12-ranked response, precision is 1.0 ($=\frac{11}{11}$) and 0.917 ($=\frac{11}{12}$), respectively, precision dropping when the item is not an error. AP averages over the precisions for all m responses ($m = 39$ for the NNS data), as shown in (4), with each response notated as R_k . Averaging over all 10 items results in the Mean AP (MAP).

$$(4) \quad AP(item) = \frac{1}{m} \sum_{k=1}^m Precision(R_k)$$

Best system parameters

To start the search for the best system parameters, it may help to continue with the example in Table 5.4. The best setting, as determined by the MAP metric, uses the tf-idf cosine (TC) approach with the Brown Corpus (Brown), and the full form of the labeled dependencies (ldh). It ranks highest because errors are well separated from non-errors; the highest ranked of 17 total errors is at rank 19. Digging a bit deeper, one can see in this example how the verb *shoot* is common in all the highest-ranked cases shown (#37–39), but absent from all the lowest, showing both the effect of the NS model (as all NSs used *shoot* to describe the action) and the potential importance of even simple representations like lemmas. In this case, the labeled dependency (ldh) representation is best, likely because the word *shoot* is not only important by itself, but also in terms of which words it relates to, and how it relates (e.g., *dobj(bird,shoot)*).

Rank	MAP	Approach	Reference	Term rep.
1	0.5168	TC	Brown	ldh
2	0.5128	TC	WSJ	ldh
3	0.5124	TC	Brown	x dh
4	0.5109	TC	Brown	l x h
5	0.5102	TC	WSJ	x dh
26	0.4826	FA	<i>na</i>	l d x
27	0.4816	TA	Brown	x d x
28	0.4769	FC	<i>na</i>	l x h
29	0.4721	TA	WSJ	x d x
30	0.4530	FA	<i>na</i>	l x h

Table 5.5: Based on Mean Average Precision, the five best and five worst combinations of system parameters across all 10 PDT items.

Table 5.5 shows the five best and five worst combinations of system settings averaged across all 10 PDT items, as ranked by MAP. The table clearly indicates that the TC approach is superior, occurring in all of the top five combinations, while approaches FA and FC compete for worst. Brown appears among top scoring combinations more often than does WSJ. Finally, for the term representation, labeled (ldh) and unlabeled (x dh) dependencies

Approach		Term rep.		Ref. corpus	
0.50630	TC	0.50499	ldh	0.50461	Brown
0.49609	TA	0.50405	x dh	0.49777	WSJ
0.49471	FC	0.49287	ldx		
0.48247	FA	0.49190	xdx		
		0.49115	lxh		

Table 5.6: Individual system parameters ranked by Mean Average Precision for all 10 PDT items.

score are used among the top scoring combinations.

I also summarize the rankings for each isolated parameter, presented in Table 5.6. For a given parameter, e.g., *ldh*, I averaged the scores from all settings including *ldh* across all 10 items. Generally, the same trends appear salient. Notably, TC outperformed the other models, with FC and TA close behind (and nearly tied). Performance fell for the simplest model, FA, which was in fact intended as a baseline. With **TC > FC** and **TA > FA**, tf-idf weighting seems preferable to basic frequencies. Likewise, with **TC > TA** and **FC > FA**, for my term based scoring, comparing score vectors outperformed simply comparing score averages.

These findings largely confirmed my expectations. The TC approach was intended to evaluate responses by focusing comparison on the most salient content. The scores here prove that to be successful. Regarding the top term representations, labeled and unlabeled dependencies both capture the relationship between dependents and their heads, making them an ideal unit for analyzing “who did what to whom” in the context of a PDT. Finally, given the subject matter and narrative style of the task, it is unsurprising that *Brown* serves as a better tf-idf reference than *WSJ*.

The trends noted in these averages were strong overall, but a closer look at individual PDT items revealed some exceptions. For example, for the item depicting a man raking leaves, the *ldx* and *xdx* term representations were the top performers. As discussed in Section 5.2.2, 100% of NSs used the verb *rake*, but only 3/39 NNSs used *rake*. Consider

the response, “The man rakes leaves.” We can expect the main verb of a transitive sentence to appear in at least three dependencies: *root*(rakes, ROOT), *nsubj*(man, rakes), and *dobj*(leaves, rakes). Note that in two of these three, the verb is the syntactic head. Thus, by omitting heads, the *ldx* and *xdx* representations increase the likelihood of overlap between the NS response and the NNS model. For example, “The man rakes leaves” and “The man sweeps leaves” both result in the *ldx* terms *nsubj*(man, *x*) and *dobj*(leaves, *x*).

A similar pattern emerged for an item that 12/14 NSs described as some version of “a woman is riding a bike/bicycle,” using the same subject, verb and object. 7/39 NNSs simply framed this as an intransitive, e.g., “The woman is biking.” 2/39 chose another verb—*pedal* or *drive*. Finally, while 30 NSs used the verb *ride*, six of these misspelled it—*rid*/*ridding* or *ridging*. It is also worth noting that 7/37 instances of *bike* or *bicycle* were misspelled. For this item, the top system performance came from combining the FC approach with the *xdx* (bag-of-words) representation. The combination of noisy (misspelled) NNS data and the small, homogenous set of NS data meant that the least granular term representation (*xdx*) worked best. The more sophisticated tf-idf based approaches suffer here due to the level of noise, allowing the FC to win out.

In the next section (5.4), I discuss how I applied these findings in the design of the current study, including the data collection and the methods of content analysis used on the much larger corpus described in Chapters 3 and 4.

5.4 Current method

The work discussed in Section 5.4 relied on a shaky implementation of “correctness” or “appropriateness” for responses, and this needed improvement, first and foremost. Developing a better and more reliable annotation scheme was a key goal for me in expanding the work to the current dissertation, in order to give the work more meaning and context and make my corpus useful for a broad range of uses. The annotation planning discussed in Section 4.1 was a direct result of the challenges working with an inadequate annotation

scheme.

As discussed in Chapter 4, in the current corpus, annotations no longer encode for errors, but instead give a binary score for five different features, which are then weighted and combined to produce a score between 0.0 and 1.0. This means the evaluation cannot take the form of MAP, but must instead compare response rankings, i.e., how well does the system rank responses in comparison to an ideal ranking based on manual annotations? Spearman rank correlation is used to provide these scores.

Because the annotations and evaluation are much different in the current work, it does not exactly follow that findings from the previous work will hold true. However, I believe that the previous work has highlighted some of the system parameters that are most likely to perform well, and I chose to focus my experiments on some of the best performing settings. For example, all of the current experiments rely on the tf-idf cosine (TC) approach, as this generally outperformed the others. As discussed in Section 5.3.5, the TC performance suffers for items where the NNS data is noisiest (with regard to spellings) and the NS data is homogenous (particularly with regard to verb choice). Rather than continue experimenting with the underperforming approaches, I chose to address these issues directly instead. As discussed in Chapter 3, to address the spelling noise, I made sure that data collection participants had access to spelling correction while typing their responses. To address the uniformity of the NS responses, I surveyed a much larger group of participants and instructed each of them to provide two responses per PDT item.

The current work retains just three of the five term representations: `ldh`, `x dh`, and `x dx`. The `ldh` and `x dh` forms performed best with the older dataset. Moreover, as these represent labeled and unlabeled dependencies, their use in linguistics is standard. The `x dx` representation is kept here as a rough equivalent of a bag-of-words model, which is useful as a baseline for comparing the other two.

Finally, as `Brown` overwhelmingly outperformed the `WSJ` as a tf-idf reference corpus, the current work relies exclusively on the Brown Corpus. The old and new datasets follow

a similar narrative style, so I expect Brown would again be the best option here.