A practical primer on processing semantic property norm data

Erin M. Buchanan[1], Simon De Deyne[2], & Maria Montefinese[3]

[1] Harrisburg University of Science and Technology

[2] The University of Melbourne

[3] University of Padua

Author Note

Correspondence concerning this article should be addressed to Erin M. Buchanan, 326 Market St., Harrisburg, PA 17101. E-mail: ebuchanan@harrisburgu.edu

Abstract

Semantic property listing tasks require participants to generate short propositions (e.g., <*barks*>, <*has fur*>) for a specific concept (e.g., dog). This task is the cornerstone of the creation of semantic property norms which are essential for modelling, stimuli creation, and understanding similarity between concepts. However, despite the wide applicability of semantic property norms for a large variety of concepts across different groups of people, the methodological aspects of the property listing task have received less attention, even though the procedure and processing of the data can substantially affect the nature and quality of the measures derived from them. The goal of this paper is to provide a practical primer on how to collect and process semantic property norms. We will discuss the key methods to elicit semantic properties and compare different methods to derive meaningful representations from them. This will cover the role of instructions and test context, property pre-processing (e.g., lemmatization), property weighting, and relationship encoding using ontologies. With these choices in mind, we propose and demonstrate a processing pipeline that transparently documents these steps resulting in improved comparability across different studies. The impact of these choices will be demonstrated using intrinsic (e.g., reliability, number of properties) and extrinsic measures (e.g., categorization, semantic similarity, lexical processing). This practical primer will offer potential solutions to several longstanding problems and allow researchers to develop new property listing norms overcoming the constraints of previous studies.

Keywords: semantic, property norm task, tutorial

A practical primer on processing semantic property norm data

Semantic properties are assumed to be, entirely or in part, the building blocks of semantic representation – the knowledge we have of the world - by a variety of theories (e.g., Collins & Quillian, 1969, @Jackendoff; Jackendoff, 1992; Minsky, 1975; Norman & Rumelhart, 1975; Saffran & Sholl, 1999; Smith & Medin, 1981) and computational models (Caramazza, Laudanna, & Romani, 1988; Farah & McClelland, 1991; Humphreys & Forde, 2001). Within this perspective, the meaning of a concept is conceived as a distributed pattern of semantic properties, which convey multiple types of information (Cree & McRae, 2003; Plaut, 2002; Rogers et al., 2004). For example, the concept HORSE can be described by encyclopedic (*<is a mammal>*), visual (*<is furry>*, *<has legs>*, *<has a tail>*, *<has a mane>*), functional (*<used for racing>*), and motor (*<gallops>*) information. Given the relevance of semantic properties in shaping theories of semantic representation, researchers have recognized the value of collecting semantic property production norms. Typically, in the property generation task, participants are presented with a set of concepts and are asked to list the properties they think are characteristic for each concept meaning. Generally, in this task, the concepts are called *cues*, and the responses to the cue are called *features*[1]. This method has a long history of use by researchers wishing to gain insight into semantic representations of concrete concepts and categories (McRae, Cree, Seidenberg, & McNorgan, 2005; Rosch & Mervis, 1975; Smith, Shoben, & Rips, 1974), and more recently, events and abstract concepts (Lebani, Lenci, & Bondielli, 2016; Vinson & Vigliocco, 2008; Wiemer-Hastings & Xu, 2005).

On the one hand, many studies adopted the property generation task itself to make inferences about word meaning and its computation (Recchia & Jones, 2012; Santos, Chaigneau, Simmons, & Barsalou, 2011; Wiemer-Hastings & Xu, 2005; Wu & Barsalou, 2009). On the other hand, researchers employed the property listing task in order to provide other researchers with a tool of standardized word stimuli and relative semantic measures.

[1]Throughout this article, features will be distinguished from cues using angular brackets.

Indeed, based on data obtained from the property production task, it is then possible to calculate numerous measures and distributional statistics both at the feature and the concept level. For example, these feature data can be used to determine the semantic similarity/distance between concepts, often by calculating the feature overlap or number of shared features between concepts (Buchanan, Valentine, & Maxwell, 2019; McRae et al., 2005; Montefinese, 2019; Montefinese, Zannino, & Ambrosini, 2015; Vigliocco, Vinson, Lewis, & Garrett, 2004), or how different types (Daniele Zannino, Perri, Pasqualetti, Caltagirone, & Carlesimo, 2006; Kremer & Baroni, 2011) and dimensions of feature informativeness, such as, distinctiveness (Garrard, Lambon Ralph, Hodges, & Patterson, 2001), cue validity (Rosch & Mervis, 1975), relevance (Sartori & Lombardi, 2004), semantic richness (Pexman, Hargreaves, Siakaluk, Bodner, & Pope, 2008), and significance (Montefinese, Ambrosini, Fairfield, & Mammarella, 2014) are distributed across concepts.

Efficient ways to collect data online have boosted the availability of large feature listing data sets. These semantic feature norms are now available across different languages: Dutch (De Deyne & Storms, 2008; Ruts et al., 2004), English (Buchanan, Holmes, Teasley, & Hutchison, 2013; Buchanan et al., 2019; Devereux, Tyler, Geertzen, & Randall, 2014; Garrard et al., 2001; McRae et al., 2005; Vinson & Vigliocco, 2008), German (Kremer & Baroni, 2011), Italian (Catricalà et al., 2015; Kremer & Baroni, 2011; Montefinese, Ambrosini, Fairfield, & Mammarella, 2013; Zannino, Perri, Pasqualetti, Caltagirone, & Carlesimo, 2006), Portuguese (Marques, Fonseca, Morais, & Pinto, 2007), and Spanish (Vivas, Vivas, Comesaña, Coni, & Vorano, 2017) as well as for blind participants (Lenci, Baroni, Cazzolli, & Marotta, 2013). However, these norms vary substantially in the procedure of data collection and their pre-processing, and this does not facilitate performing cross-language comparisons and, thus, making inferences about how semantic representations are generalizable across languages.

First, there is a lack of agreement in the instructions provided to the participants.

Indeed, while some studies use an open-ended verbal feature production (Buchanan et al., 2013, 2019; De Deyne & Storms, 2008; Montefinese et al., 2013) where participants can list the features related to the concept with any kind of semantic relation, other studies use a constrained verbal feature production (Devereux et al., 2014; Garrard et al., 2001) where participants were instructed to use specific semantic relations between cue concept and features, such as, for example, <*is …*>, <*has …*>, <*does …*>, <*made of …*>, and so forth. Moreover, some authors instruct the participants to produce a single word as a feature instead of a multiple-word description. This latter case could also determine a problem on subsequent coding steps that affect the identification of pieces of information. For example, if the participant listed the feature <*has four wheels*> for the concept CAR, there is no consensus if this feature should be divided into <*has wheels*> and <*has four wheels*>, under the assumption that the participant provided two bits of information, or rather if it should be considered as a unique feature. Second, some authors gave a time limit to provide the features descriptions (Kremer & Baroni, 2011; Lenci et al., 2013; Marques et al., 2007) or a limited number of features to be listed (De Deyne & Storms, 2008), with a possible influence on a number of feature-based measures (e.g., semantic richness or distinctiveness).

Because the feature listing task is a verbal task and language is very productive (i.e., the same feature can be expressed in many different ways), few features will be listed in exactly the same way across participants. To be able to derive reliable quantitative measures, nearly all studies specify a series of pre-processing steps to group verbal utterances about the same underlying conceptual property together. The main problem is that there is no agreement about how to code/pre-process data derived from the feature listing task. Recoding features is sometimes done in manually (McRae et al., 2005) whereas others use semi-automatic procedures, especially for larger datasets (Buchanan et al., 2019). Further points of debate are related to the inclusion/exclusion of certain types of responses. For example, unlike previous semantic norms (McRae et al., 2005; Montefinese et al., 2013; Vivas et al., 2017), Buchanan et al. (2019) included idiosyncratic features (features produced only by one or a few number of participants) if they were in the top listed features, ambiguous words (words with multiple meanings), and created a special coding for affixes of the root words. Moreover, they discarded stop words, such as, the, an, of, and synonyms were treated as different entries.

While hand-coding features leads to features that concise, easily interpretable, and highly predictive of semantic behavior, the increasing scale of recent studies and more powerful natural language processing techniques make automatic procedures an attractive alternative. Moreover, building standard automatic procedures to process feature-listing data would not only add transparency to the process but would also prevent human errors and allow a generalization of the data across languages. For the first time, in this study we propose an automatic procedure to code the raw feature data derived from a semantic feature listing task (SFL). The next sections provide a tutorial on how raw feature data might be processed to a more compact feature output. The tutorial is written for *R* and is fully documented, such that users can adapt it to their language of choice (https://github.com/doomlab/FLT-Primer). Figure 1 portrays the proposed set of steps including spell checking, lemmatization, exclusion of stop words, and final processing in a multi-word sequence approach or a bag of words approach. After detailing these steps, the final data form will evaluated and compared to previous norms to determine the usefulness of this approach.

Materials and Data Format

You can load the entire set of libraries for this tutorial as shown below using `dependencies.R` found online[2].

[2]A `packrat` project compilation is available on GitHub for reproducibility (Ushey, McPherson, Cheng, Atkins, & Allaire, 2018)

```r
library(here)
library(dplyr)
#Spelling
library(hunspell)
library(tidytext)
library(stringi)
#Lemmatization
library(koRpus)
library(koRpus.lang.en)
library(tokenizers)
#Stopwords
library(stopwords)
```

The data can then be imported with `importData.R`. Additionally, the answers from participants may need to be normalized into lowercase for consistency.

```r
# Importing the raw feature lists
X <- read.csv("../raw_data/tidy_words.csv", stringsAsFactors = F)
## Lower case to normalize
X$answer <- tolower(X$answer)
```

The data for this tutorial includes 16544 unique concept-feature responses for 226 concepts from Buchanan et al. (2019). The concepts were taken from McRae et al. (2005), Vinson and Vigliocco (2008), and Bruni, Tran, and Baroni (2014). The concepts include 185 nouns, 25 verbs, and 16 adjectives. Concreteness ratings collected by Brysbaert, Warriner, and Kuperman (2014) were matched with the current data set. The concreteness ratings capture the difference between abstract (language-based) and concrete (experience-based) concepts and were measured on a five-point scale. Nouns were rated as most concrete: M = 4.59 (SD = 0.52), followed by adjectives: M = 3.78 (SD = 0.81), and verbs: M = 3.57 (SD = 0.79). The SFL data consist of a text file where concept-feature observation is a row and each column is a variable. An example of these raw data are shown in Table 1, where the `word` column is the cue, and the `answer` column denotes a single participant's response. The original data can be found at https://osf.io/cjyzw/.

The data was collected using the instructions provided by McRae et al. (2005), however, in contrast to the suggestions for consistency detailed above (Devereux et al., 2014), each participant was simply given a large text box to include their answer. Each answer includes multiple embedded features, and the tutorial proceeds to demonstrate potential processing addressing the data in this nature. With structured data entry for participants (e.g., asking participants to type one feature on each line), the suggested processing steps are reduced.

Spelling

The first step (see Figure 1) in processing the features consists of identifying and replacing spelling mistakes. Spell checking can be automated with the `hunspell` package in *R* (Ooms, 2018) using `spellCheck.R`. Each `answer` can be checked for misspellings across an entire column of answers, which is in the `X` dataset. Because participants were recruited in the United States, we used the default American English dictionary. The `hunspell` vignettes provide details on how to import your own dictionary for non-English languages. The choice of dictionary should also normalize between multiple varieties of the same language, for example, the `"en_GB"` would convert to British English spellings.

```
# Extract a list of words
tokens = unnest_tokens(tbl = X, output = token, input = answer)
wordlist = unique(tokens$token)
# Spell check the words
spelling.errors <- hunspell(wordlist)
spelling.errors <- unique(unlist(spelling.errors))
spelling.sugg <- hunspell_suggest(spelling.errors, dict = dictionary("en_US"))
```

The result from the `hunspell()` function is a list object of spelling errors for each row of data. For example, when responding to APPLE, a participant wrote <*fruit grocery store orchard red green yelloe good with peanut butter good with caramell*>, and the spelling errors were denoted as <*yelloe*> <*caramell*>. After checking for errors, the `hunspell_suggest()`

function was used to determine the most likely replacement for each error. For <*yelloe*>, both <*yellow*> <*yell*> were suggested, and <*caramel*> <*camel*> were suggested for <*caramell*>.

Answers are provided in the most probable order, therefore, the first suggestion is selected as the correct answer. These answers are compiled into a spelling dictionary, which is saved for reproducibility. In addition to the hunspell dictionary, an auxiliary dictionary with pre-coded error responses and corrections could also be added at this stage to catch any false positives by adding entries to the `spelling.dict`. Other paid alternatives, such as Bing Spell Check, can be a useful avenue for datasets that may contain brand names (i.e, *apple* versus *Apple*) or slang terms and provides context sensitive corrections (e.g., keeping *Apple* as a response to computer, but not as a response to green).

```
# Pick the first suggestion
spelling.sugg <- unlist(lapply(spelling.sugg, function(x) x[1]))
spelling.dict <- as.data.frame(cbind(spelling.errors,spelling.sugg))
spelling.dict$spelling.pattern <- paste0("\\b", spelling.dict$spelling.errors, "\\b")
# Write out spelling dictionary
write.csv(x = spelling.dict, file = "../output_data/spelling.dict.csv",
          fileEncoding = "utf8", row.names = F)
```

As noted, data was collected with a large text box, allowing participants to free respond to the target cue. Participants often used extra spacing, tabs or other punctuation to denote separate answers to the cue. The `unnest_tokens()` function from `tidytext` can be used to split their answers into separate response lines and `trimws()` to remove all extra white spaces (De Queiroz et al., 2019).

```
# Parse features
tokens <- unnest_tokens(tbl = X, output = token,
                        input = answer, token = stringr::str_split,
                        pattern = "  |\\, |\\.|\\,|\\;")
tokens$token <- trimws(tokens$token,
                       which = c("both", "left", "right"),
                       whitespace = "[ \t\r\n]")
```

To finalize our data cleaning, we can remove blank lines, and use `stri_replace_all_regex()` is used to replace the spelling errors with their corrections from the `stringi` package (Gagolewski & Tartanus, 2019). The spell checked dataframe is then output to a comma delimited file to preserve each workflow step.

```
# Remove empty features
tokens <- tokens[!tokens$token =="",]
tokens$corrected <- stri_replace_all_regex(str = tokens$token,
                                           pattern = spelling.dict$spelling.pattern,
                                           replacement = spelling.dict$spelling.sugg,
                                           vectorize_all = FALSE)
# Rename columns
tokens <- tokens %>%
  rename(cue = word,feature = corrected) %>%
  select(cue, feature)
# Write processed file
write.csv(x = tokens,file = "../output_data/spellchecked.features.csv",
          fileEncoding = "utf8",row.names = F)
```

Lemmatization

The next step groups different word forms that share the same lemma. The process of lemmatizing words uses a trained dictionary to convert all tokens part of a a lexeme set (i.e., all words forms that have the same meaning, *am, are, is*) to a common lemma (i.e., *be*)[3]. Lemmatization is performed using the `TreeTagger` program (Schmid, 1994) and implemented through the `koRpus` package in *R* (Michalke, 2018). TreeTagger is a trained tagger designed to annotate part of speech and lemma information in text, and parameter

[3]We mainly focus on lemmatization and do not proceed stemming the word because it introduces additional ambiguity. More specifically, stemming involves processing words using heuristics to remove affixes or inflections, such as *ing* or *s*. The stem or root word may not reflect an actual word in the language, as simply removing an affix does not necessarily produce the lemma. For example, in response to AIRPLANE, <*flying*> can be easily converted to <*fly*> by removing the *ing* inflection. However, this same heuristic converts the feature <*wings*> into <*w*> after removing both the *s* for a plural marker and the *ing* participle marker.

files are available for multiple languages. We will create a unique set of tokenized words to lemmatize to speed computation, as shown in `lemmatization.R`.

```
# Open the spell checked data
X <- read.csv("../output_data/spellchecked.features.csv", stringsAsFactors = F)
# Extract the list of updated tokens
tokens <- unnest_tokens(tbl = X, output = word, input = feature)
cuelist <- unique(tokens$cue)
```

The `treetag()` function calls the installation of TreeTagger to provide part of speech tags and lemmas for each token. Importantly, the `path` option should be the directory of the TreeTagger installation.

```
# Create a dataframe for lemmas
tokens.tagged <- data.frame(doc_id=character(),
                            token=character(),
                            wclass=character(),
                            lemma=character(),
                            stringsAsFactors=FALSE)
# Loop over cues and create lemmas + POS tags
for (i in 1:length(cuelist)){
  temp.tag <- suppressWarnings(
    suppressMessages(
      treetag(c(X$feature[X$cue == cuelist[i]], "NULL"),
              treetagger="manual", format="obj",
              TT.tknz=FALSE, lang="en", doc_id = cuelist[i],
              # These parameters are based on your computer
              TT.options=list(path="~/TreeTagger", preset="en"))))
  temp.tag <- temp.tag@TT.res %>%
    mutate_if(is.factor, as.character)
  tokens.tagged <- tokens.tagged %>%
    bind_rows(temp.tag %>%
                select(doc_id, token, wclass, lemma))
  }
```

This function returns a tagged corpus object, which can be converted into a dataframe of the token-lemma information. TreeTagger will return `<unknown>` for unknown values and `@card@` for numbers, and these values were replaced with the original token. Table 2 portrays example results from TreeTagger.

```
tokens.tagged <- tokens.tagged %>%
  rename(cue = doc_id, feature = token, pos = wclass)
# Clean up unknown lookups
tokens.tagged$lemma[tokens.tagged$lemma == "<unknown>"] <- tokens.tagged$feature[tokens.tagged$lemma == "<unknown>"]
tokens.tagged$lemma[tokens.tagged$lemma == "@card@"] <- tokens.tagged$feature[tokens.tagged$lemma == "@card@"]
tokens.tagged$lemma <- tolower(tokens.tagged$lemma)
# Write processed file
write.csv(x = tokens.tagged, file = "../output_data/lemmatized.features.csv",
          fileEncoding = "utf8", row.names = F)
```

Stopwords

As shown in Figure 1, the next stage of processing would be to exclude stopwords, such as *the, of, but*. The `stopwords` package (Benoit, Muhr, & Watanabe, 2017) includes a list of stopwords for more than 50 languages. At this stage, the `feature` (original tokens, not lemmatized) or `lemma` (lemmatized tokens) column can be used depending on researcher selection. This code is included in `stopWordRemoval.R`.

```
# Open the lemmatized data
X <- read.csv("../output_data/lemmatized.features.csv", stringsAsFactors = F)
# Remove punctuation and stopwords from lemmas
X$lemma <- gsub("\\-", " ", X$lemma)
X$lemma <- gsub("^$|\002", NA, trimws(X$lemma))
X.nostop <- X %>%
  filter(!grepl("[[:punct:]]", lemma)) %>%
  filter(!lemma %in% stopwords(language = "en", source = "snowball")) %>%
  filter(!is.na(lemma))
# Write processed file
write.csv(x = X.nostop, file = "../output_data/nostop.lemmas.csv",
          fileEncoding = "utf8", row.names = F)
```

Multi-word Sequences

Multi-word sequences are often coded to mimic a Collins and Quillian (1969) semantic network, where words are nodes and edges are labelled with relations such as "is-a" or "has-a". Some instructions specify the use of specific relation types (Devereux et al., 2014; Garrard et al., 2001), in which case pre-encoded the following step can be omitted. A potential solution for processing unstructured data involves identifying patterns that mimic "is-a" and "has-a" strings. Examples of such an approach is the Strudel model (Baroni, Murphy, Barbu, & Poesio, 2010) in which meaningful relations are grouped together using a small set of highly specific regular expressions. An examination of the coding in McRae et al. (2005) and Devereux et al. (2014) indicates that the feature tags are often adverb-adjective, verb-noun, or verb-adjective-noun sequences. Using TreeTagger on each concept's answer set, we can obtain the parts of speech in context for each lemma. With `dplyr` (Wickham, Francios, Henry, Muller, & Rstudio, 2019), new columns are added to tagged data to show all bigram and trigram sequences. All adverb-adjective, verb-noun, and verb-adjective-noun combinations are selected, and any words not part of these multi-word sequences are treated as unigrams. Finally, the `count()` function is used to tabulate the final count of n-grams and their frequency (`multiwordSequences.R`).

```r
# Open the no stop words data
X <- read.csv("../output_data/nostop.lemmas.csv", stringsAsFactors = F)
# Combine lemmas and POS
X <- X %>%
  mutate(two.words = paste(lemma, lead(lemma), sep = " "),
         three.words = paste(lemma, lead(lemma),
                             lead(lemma, n = 2L), sep = " "),
         two.words.pos = paste(pos, lead(pos), sep = "."),
         three.words.pos = paste(pos, lead(pos),
                                 lead(pos, n = 2L), sep = "."))
# Patterns
adverb.adj <- grep("\\badverb.adj", X$two.words.pos)
verb.nouns <- grep("\\bverb.noun", X$two.words.pos)
verb.adj.nouns <- grep("\\bverb.adjective.noun", X$three.words.pos)
# Use combined and left over lemmas
X$combined.lemmas <- NA
X$combined.lemmas[c(adverb.adj, verb.nouns)] <- X$two.words[c(adverb.adj,verb.nouns)]
X$combined.lemmas[verb.adj.nouns] <- X$three.words[verb.adj.nouns]
X$combined.lemmas[-c(verb.nouns, verb.nouns+1, verb.adj.nouns,
                     verb.adj.nouns+1, verb.adj.nouns+2)] <- X$lemma[-c(verb.nouns, verb.nouns+1,
```

```
                                                                verb.adj.nouns, verb.adj.nouns+1,
                                                                verb.adj.nouns+2)]
#Create cue-lemma frequency
multi.words <- X %>%
  filter(!is.na(combined.lemmas)) %>%
  group_by(cue) %>%
  count(combined.lemmas)
# Write processed file
write.csv(x = multi.words, file = "../output_data/multi.nostop.lemmas.csv",
          fileEncoding = "utf8", row.names = F)
```

This procedure produces mostly positive output, such as FINGERS <*have fingernails*> and COUCHES <*have cushions*>. One obvious limitation is the potential necessity to match this coding system to previous codes, which were predominately hand processed. Further, many similar phrases, such as the ones for *zebra* shown below may require flexible regular expressions to ensure that the different codings for <*is a horse*> are all combined together, as shown in Table 3.

Bag of Words

To be able to evaluate the role of identifying multi-word sequences, we now describe an approach where this information is not retained. This bag of words approach simply treats each token as a separate feature to be tabulated for analysis. After stemming and lemmatization, the data can be processed as single word tokens into a table of frequencies for each cue word. The resulting dataframe is each cue-feature combination with a total for each feature from `bagOfWords.R`. Table 4 shows the top ten most frequent responses to *zebra* given the bag of words approach.

```
# Open the no stop words data
X <- read.csv("../output_data/nostop.lemmas.csv", stringsAsFactors = F)
# Create cue-lemma frequency
bag.words <- X %>%
  group_by(cue) %>%
```

```
  count(lemma)
# Write processed file
write.csv(x = bag.words, file = "../output_data/bag.nostop.lemmas.csv",
          fileEncoding = "utf8", row.names = F)
```

Descriptive Statistics

The finalized data now represents a processed set of cue-feature combinations with their frequencies for analysis. The data from Buchanan et al. (2019) was collected over multiple years with multiple sample sizes. The sample size for each cue was then merged with the finalized cue-feature information to control for differences in potential maximum frequency. Table 5 includes descriptive statistics for the processed cue-feature set.

Number of response types. First, the number of cue-feature combinations was calculated by taking the average number of cue-feature listings for each cue. Therefore, the total number of features listed for ZEBRA might be 100, while APPLE might be 45, and these values were averaged. More cue-feature combinations are listed for the multi-word approach, due to differences in combinations for some overlapping features as shown in Table 3. The large standard deviation for both approaches indicates that cues have a wide range of possible features listed. For example for the cue ZEBRA, we find a total of 196 features, whereas for APPLE we find 134 features. We expect that the number of different responses tokens is a function of the number of times a cue was presented in the study. To investigate this relation, we calculated the correlation provided represents the relation between sample size for a cue and the number of features listed for that cue. These values are high and positive, indicating that the number of unique features increases with each participant.

Idiosyncratic responses. Potentially, many of the cue-feature combinations could be considered idiosyncratic. The next row of the table denotes the average number of cue-feature responses listed by less than 10% of the participants. This percent of responses is somewhat arbitrary, as each researcher has determined where the optimal criterion should be. For example, McRae et al. (2005) used 16% or 5/30 participants as a minimum standard, and Buchanan et al. (2019) recently used a similar criteria. Many cue-features are generated by a small number of participants, indicating that these are potentially idiosyncratic or part of long tailed distribution of feature responses with many low frequency features. The advantage to the suggested data processing pipeline and code provided here is the ability of each researcher to determine their own level of response necessary, if desired.

Response strength. The next two lines of Table 5 indicate cue-feature combination frequencies, such as the number of times ZEBRA <*stripes*> or APPLE <*red*> were listed by participants. The percent of responses is the frequency divided by sample size for each cue, to normalize over different sample sizes present in the data. These average frequency/percent can be seen as a measure of response strength and were calculated for each cue, and then averaged over all cues. The correlation represents the average response strength for each cue related to the sample size for that cue. These frequencies are low, matching the results for a large number of idiosyncratic responses. The correlation between frequency of response and sample size is positive, indicating that larger sample sizes produce items with larger frequencies.

Additionally, the correlation between response strength and sample size is negative, suggesting that larger sample sizes are often paired with more items with smaller response strengths. Figure 2 displays the correlations for the average cue-frequency responses and the response strength by sample size. It appears that the relationship between sample size and percent is likely curvilinear, rather than linear. The size of the points indicates the variability (standard deviation of each cue word's average frequency or percent). Variability appears to increase linearly with sample size for average frequency, however, it is somewhat mixed for average percent. These results may imply a necessity to discuss common sample sizes for data collection ($ns \sim 30$) to determine the optimal sample size for an appropriate body of data for each cue word.

Internal Comparison of Approach

In this section, we show that the bag of words approach matches the data from McRae et al. (2005), Vinson and Vigliocco (2008), and Buchanan et al. (2019), which compares data processed completely through code to datasets that were primarily hand coded. In Buchanan et al. (2019), the McRae et al. (2005) and Vinson and Vigliocco (2008) datasets were recoded in a bag of words approach, and the comparison between all three is provided below. The multi-word sequence approach would be comparable if one or more datasets used the same structured data collection approach or with considerable hand coded rules for feature combinations. The data from open ended responses, such as the Buchanan et al. (2019), could potentially be compared in the demonstrated multi-word sequence approach, if the raw data from other such projects were available.

Cosine is often used as a measure of semantic similarity, indicating the feature overlap between two sets of cue-feature lists. These values can range from 0 (no overlap) to 1 (perfect overlap). Two cosine values can be derived from the Buchanan et al. (2019) data: the raw cosine, which included all features as listed and the cosine for lemmatized responses. Each cue in the sample data for this project was compared to the corresponding cue in the Buchanan et al. (2019). If data were processed in an identical fashion, the cosine values would be nearly 1 for Buchanan et al. (2019) data or match the cosine values found for McRae et al. (2005) and Vinson and Vigliocco (2008) in the Buchanan et al. (2019) results: original feature cosine = .54-.55, and lemmatized[4] features = .66-.67. However, all previous datasets have been reduced by eliminating idiosyncratic features at various points, and therefore, we might expect that noise in the data would reduce the average cosine values.

[4] These results were lemmatized by creating a lookup dictionary from the features listed in the Buchanan et al. (2019) norms

Table 6 shows the role of using a cut-off for low-frequent or idiosyncratic responses by calculating the cosine values when using varying cut-offs or stopword filtering. On the left, the cosine values with stopwords are provided for both the original feature listed (i.e., no lemmatization) and the lemmatized features. The right side of the table includes the cosine values once stopwords have been removed. The removal of stopwords increases the match between sets indicating how removing these terms can improve prediction. The cosine values for no stopwords indicate a somewhat comparable set of data, with lower values for McRae et al. (2005) than previous results in the original feature sets. These values indicate that the data processed entirely in *R* produces a comparable set of results, albeit with added noise of small frequency features.

External Comparison of Approach

The MEN dataset (Bruni et al., 2014) contains cue-cue pairs of English words rating for similarity by Amazon Mechanical Turk participants for stimuli taken from the McRae et al. (2005) feature norms. In their rating task, participants were shown two cue-cue pairs and asked to select the more related pair of the two presented. Each pair was rated by 50 participants, and thus, a score of 50 indicates high relatedness, while a score of 0 indicates no relatedness. The ratings for the selected set of cues provided in this analysis was 2 - 49 with an average rating of 25.79 ($SD = 12.00$). The ratings were compared to the cosine calculated between cues using the bag of words method with and without stopwords. The correlation between bag of words cosines with stopwords and the MEN ratings was $r = .54$, 95% CI [.42, .63], $N = 179$, indicating agreement between raters and cosine values. The agreement between ratings and bag of word cosine values was higher when stopwords were excluded, $r = .70$, 95% CI [.61, .76].

Discussion

Semantic feature listing tasks are used across various disciplines and are likely to remain an important source of information about the subjective meaning of concepts. In this article we have outlined a workflow to process large datasets where features consist of unstructured short propositions derived from written language. The advantage to this workflow is two-fold. First, science practices are shifting to open procedures and practices (Nosek et al., 2015), and reproducible research is key (Peng, 2011). Second, automated processing provides faster data analysis than hand-coded systems, and the ability to examine how processing steps affect results. We have shown that the automated procedure provides a comparable set of results to the hand-coded systems from Buchanan et al. (2019), McRae et al. (2005), and Vinson and Vigliocco (2008). The addition of specialized lemmas and other word exclusions (i.e., <*sometimes*>, <*usually*>, <*lot*> or idiosyncratic features) would provide more reduction, and thus, more overlap between hand and automated processing. Further, the automated data processing showed strong correlations with external subjective ratings of cue-cue relatedness in the MEN dataset. We suggest the workflow shown in Figure 1 and the suggested *R* code can provide a framework for researchers to use on their own data.

Extending the approach. An attractive property of the subjective feature listing task is that it results in transparent representations. As a result, many researchers have taken additional steps to group specific types of knowledge together, depending on semantic relations (e.g., taxonomy relations) or their mapping onto distinct brain regions (Fairhall & Caramazza, 2013). Typically this involves applying a hand-crafted coding scheme, which requires a substantial effort. One of the common ontologies is the one developed by Wu and Barsalou (2009). The ontology is structured as a hierarchical taxonomy for coding categories as part of the feature listing task. It has been used in several projects, notably the McRae et al. (2005). Examples of the categories include taxonomic (synonyms, subordinates), entity (internal components, behavior, spatial relations), situation (location, time), and introspective properties (emotion, evaluation). Coding ontology may be best performed systematically with look-up rules of previously decided upon factors, however, clustering analyses may provide a potential avenue to explore categorizing features within the current dataset. One limitation to this method the sheer size of the idiosyncratic features as mentioned above, and thus, features smaller in number may be more difficult to group.

Potentially, a simple ontology can be mapped using an approach similar to Strudel (structured dimension extraction and labeling, Baroni et al., 2010). Strudel is a corpus-based semantic model wherein cue words are found in a large text corpus and matched to nouns, verbs, and adjectives that appear near a concept. Using specific patterns of expected feature listing, Baroni et al. (2010) were able build a model of English concepts and their properties that aligned with semantic feature production norms. From this model, they were able to cluster properties based on their lexical patterns. For example, if a sentence included the phrase *fruit, such as an apple*, this lexical pattern would be classified as *such_as+right*, indicating that the concept (apple) was found to the right of the property (fruit) with the phrase such as connecting them. Using clustering, Baroni et al. (2010) was able to assign four ontology labels to properties: part, category, location, and function. Using these results, we can match 2279 of the bag of words features (5%). These features were predominately parts (39.7), followed by function (30.7), location (24.2), and category (5.4). Table 7 indicates ten of the most frequent cue-feature pairs for each ontology label, excluding duplicate features across cues. An examination of the top results indicates coherent labels (parts: ZEBRA <*stripe*>, location: SHOE <*foot*>, and category: FURNITURE <*table*>); however, there are also a few mismatches (location: SCISSORS <*cut*>, function: LEAF <*green*>). This model represents an area in which one might begin to automate the labeling process, likely combined with other pre-defined rule sets.

Some limitations. So far we have not investigated to what extend the automatic procedure leads to equally good representations for different types of concepts. More specifically, abstract concepts tend to have a larger number of features, and especially for these types of concepts, pooling together features might improve the quality of the final representation. Potentially, this might require additional steps in which features are not only grouped based on surface properties but might also benefit from grouping synonymous words. Within this framework, the properties could be added within a lookup dictionary to further promote an open and transparent coding for data processing.

References

Baroni, M., Murphy, B., Barbu, E., & Poesio, M. (2010). Strudel: A corpus-based semantic model based on properties and types. *Cognitive Science*, *34*(2), 222–254. doi:10.1111/j.1551-6709.2009.01068.x

Benoit, K., Muhr, D., & Watanabe, K. (2017). stopwords: Multilingual stopword lists. Retrieved from https://cran.r-project.org/web/packages/stopwords/index.html

Bruni, E., Tran, N. K., & Baroni, M. (2014). Multimodal distributional semantics. *Journal of Artificial Intelligence Research*, *49*, 1–47. doi:10.1613/jair.4135

Brysbaert, M., Warriner, A. B., & Kuperman, V. (2014). Concreteness ratings for 40 thousand generally known English word lemmas. *Behavior Research Methods*, *46*(3), 904–911. doi:10.3758/s13428-013-0403-5

Buchanan, E. M., Holmes, J. L., Teasley, M. L., & Hutchison, K. A. (2013). English semantic word-pair norms and a searchable Web portal for experimental stimulus creation. *Behavior Research Methods*, *45*(3), 746–757. doi:10.3758/s13428-012-0284-z

Buchanan, E. M., Valentine, K. D., & Maxwell, N. P. (2019). English semantic feature production norms: An extended database of 4436 concepts. *Behavior Research Methods*. doi:10.3758/s13428-019-01243-z

Caramazza, A., Laudanna, A., & Romani, C. (1988). Lexical access and inflectional morphology. *Cognition*, *28*(3), 297–332. doi:10.1016/0010-0277(88)90017-0

Catricalà, E., Della Rosa, P. A., Plebani, V., Perani, D., Garrard, P., & Cappa, S. F. (2015). Semantic feature degradation and naming performance. Evidence from neurodegenerative disorders. *Brain and Language*, *147*, 58–65. doi:10.1016/J.BANDL.2015.05.007

Collins, A. M., & Quillian, M. R. (1969). Retrieval time from semantic memory. *Journal of Verbal Learning and Verbal Behavior*, *8*(2), 240–247. doi:10.1016/S0022-5371(69)80069-1

Cree, G. S., & McRae, K. (2003). Analyzing the factors underlying the structure and computation of the meaning of chipmunk, cherry, chisel, cheese, and cello (and many other such concrete nouns). *Journal of Experimental Psychology: General*, *132*(2), 163–201. doi:10.1037/0096-3445.132.2.163

Daniele Zannino, G., Perri, R., Pasqualetti, P., Caltagirone, C., & Carlesimo, G. A. (2006). Analysis of the semantic representations of living and nonliving concepts: a normative study. *Cognitive Neuropsychology*, *23*(4), 515–540.

De Deyne, S., & Storms, G. (2008). Word associations: Norms for 1,424 Dutch words in a continuous task. *Behavior Research Methods*, *40*(1), 198–205. doi:10.3758/BRM.40.1.198

De Queiroz, G., Hvitfeldt E, Keyes O, Misra K, Mastny T, Erickson J, … Silge J. (2019). tidytext: Text mining using 'dplyr', 'ggplot2', and other tidy tools. Retrieved from https://cran.r-project.org/web/packages/tidytext/index.html

Devereux, B. J., Tyler, L. K., Geertzen, J., & Randall, B. (2014). The Centre for Speech, Language and the Brain (CSLB) concept property norms. *Behavior Research Methods*, *46*(4), 1119–1127. doi:10.3758/s13428-013-0420-4

Fairhall, S. L., & Caramazza, A. (2013). Category-selective neural substrates for person- and place-related concepts. *Cortex*, *49*(10), 2748–2757. doi:10.1016/j.cortex.2013.05.010

Farah, M. J., & McClelland, J. L. (1991). A computational model of semantic memory impairment: Modality specificity and emergent category specificity. *Journal of Experimental Psychology: General*, *120*(4), 339–357. doi:10.1037/0096-3445.120.4.339

Gagolewski, M., & Tartanus, B. (2019). stringi: Character string processing facilities. Retrieved from https://cran.r-project.org/web/packages/stringi/index.html

Garrard, P., Lambon Ralph, M. A., Hodges, J. R., & Patterson, K. (2001). Prototypicality, distinctiveness, and intercorrelation: Analyses of the semantic attributes of living and nonliving concepts. *Cognitive Neuropsychology*, *18*(2), 125–174. doi:10.1080/02643290125857

Humphreys, G. W., & Forde, E. M. (2001). Hierarchies, similarity, and interactivity in object recognition: "category-specific" neuropsychological deficits. *The Behavioral and Brain Sciences*, *24*(3), 453–476.

Jackendoff, R. (1990). On Larson's treatment of the double object construction. The MIT Press. doi:10.2307/4178683

Jackendoff, R. (1992). *Semantic structures*. Boston, MA: MIT Press.

Kremer, G., & Baroni, M. (2011). A set of semantic norms for German and Italian. *Behavior Research Methods*, *43*(1), 97–109. doi:10.3758/s13428-010-0028-x

Lebani, G. E., Lenci, A., & Bondielli, A. (2016). You are what you do: An empirical characterization of the semantic content of the thematic roles for a group of Italian verbs. *Journal of Cognitive Science*, *16*(4), 401–430. doi:10.17791/jcs.2015.16.4.401

Lenci, A., Baroni, M., Cazzolli, G., & Marotta, G. (2013). BLIND: A set of semantic feature norms from the congenitally blind. *Behavior Research Methods*, *45*(4), 1218–1233. doi:10.3758/s13428-013-0323-4

Marques, J. F., Fonseca, F. L., Morais, S., & Pinto, I. A. (2007). Estimated age of acquisition norms for 834 Portuguese nouns and their relation with other psycholinguistic variables. *Behavior Research Methods*, *39*(3), 439–444. doi:10.3758/BF03193013

McRae, K., Cree, G. S., Seidenberg, M. S., & McNorgan, C. (2005). Semantic feature production norms for a large set of living and nonliving things. *Behavior Research Methods*, *37*(4), 547–559. doi:10.3758/BF03192726

Michalke, M. (2018). koRpus: An R package for text analysis. Retrieved from https://cran.r-project.org/web/packages/koRpus/index.html

Minsky, M. (1975). A framework for representing knowledge. In P. H. Winston (Ed.), *The psychology of computer vision* (pp. 211–277). Winston, NY: McGraw Hill.

Montefinese, M. (2019). Semantic representation of abstract and concrete words: a minireview of neural evidence. *Journal of Neurophysiology*, *121*(5), 1585–1587. doi:10.1152/jn.00065.2019

Montefinese, M., Ambrosini, E., Fairfield, B., & Mammarella, N. (2013). Semantic memory: A feature-based analysis and new norms for Italian. *Behavior Research Methods*, *45*(2), 440–461. doi:10.3758/s13428-012-0263-4

Montefinese, M., Ambrosini, E., Fairfield, B., & Mammarella, N. (2014). Semantic significance: a new measure of feature salience. *Memory & Cognition*, *42*(3), 355–369. doi:10.3758/s13421-013-0365-y

Montefinese, M., Zannino, G. D., & Ambrosini, E. (2015). Semantic similarity between old and new items produces false alarms in recognition memory. *Psychological Research*, *79*(5), 785–794. doi:10.1007/s00426-014-0615-z

Norman, D. A., & Rumelhart, D. E. (1975). *Explorations in cognition.* San Francisco, CA: Freeman.

Nosek, B. A., Alter, G., Banks, G. C., Borsboom, D., Bowman, S. D., Breckler, S. J., . . . Yarkoni, T. (2015). Promoting an open research culture. *Science*, *348*(6242), 1422–1425. doi:10.1126/science.aab2374

Ooms, J. (2018). The hunspell package: High-Performance Stemmer, Tokenizer, and Spell Checker for R. Retrieved from https://cran.r-project.org/web/packages/hunspell/

Peng, R. D. (2011). Reproducible research in computational science. *Science (New York, N.Y.)*, *334*(6060), 1226–7. doi:10.1126/science.1213847

Pexman, P. M., Hargreaves, I. S., Siakaluk, P. D., Bodner, G. E., & Pope, J. (2008). There are many ways to be rich: Effects of three measures of semantic richness on visual word recognition. *Psychonomic Bulletin & Review*, *15*(1), 161–167. doi:10.3758/PBR.15.1.161

Plaut, D. C. (2002). Graded modality-specific specialisation in semantics: A computational account of optic aphasia. *Cognitive Neuropsychology*, *19*(7), 603–639. doi:10.1080/02643290244000112

Recchia, G., & Jones, M. N. (2012). The semantic richness of abstract concepts. *Frontiers in Human Neuroscience*, *6*, 315. doi:10.3389/fnhum.2012.00315

Rogers, T. T., Lambon Ralph, M. A., Garrard, P., Bozeat, S., McClelland, J. L., Hodges, J. R., & Patterson, K. (2004). Structure and deterioration of semantic memory: A neuropsychological and computational investigation. *Psychological Review*, *111*(1), 205–235. doi:10.1037/0033-295X.111.1.205

Rosch, E., & Mervis, C. B. (1975). Family resemblances: Studies in the internal structure of categories. *Cognitive Psychology*, *7*(4), 573–605. doi:10.1016/0010-0285(75)90024-9

Ruts, W., De Deyne, S., Ameel, E., Vanpaemel, W., Verbeemen, T., & Storms, G. (2004). Dutch norm data for 13 semantic categories and 338 exemplars. *Behavior Research Methods, Instruments, & Computers*, *36*(3), 506–515. doi:10.3758/BF03195597

Saffran, E., & Sholl, A. (1999). Clues to the function and neural architecture of word meaning. In P. Hogoort & C. Brown (Eds.), *The neurocognition of language*. Oxford University Press.

Santos, A., Chaigneau, S. E., Simmons, W. K., & Barsalou, L. W. (2011). Property generation reflects word association and situated simulation. *Language and Cognition*, *3*(1), 83–119. doi:10.1515/langcog.2011.004

Sartori, G., & Lombardi, L. (2004). Semantic relevance and semantic disorders. *Journal of Cognitive Neuroscience*, *16*(3), 439–452. doi:10.1162/089892904322926773

Schmid, H. (1994). Probabilistic part of speech tagging using decision trees. doi:10.1.1.28.1139

Smith, E. E., Shoben, E. J., & Rips, L. J. (1974). Structure and process in semantic memory: A featural model for semantic decisions. *Psychological Review*, *81*(3), 214–241. doi:10.1037/h0036351

Smith, E., & Medin, D. L. (1981). *Categories and concepts (Vol. 9)*. Cambridge, MA: Harvard University Press.

Ushey, K., McPherson, J., Cheng, J., Atkins, A., & Allaire, J. (2018). packrat: A dependency management system for projects and their R rackage dependencies. Retrieved from https://cran.r-project.org/web/packages/packrat/index.html

Vigliocco, G., Vinson, D. P., Lewis, W., & Garrett, M. F. (2004). Representing the meanings of object and action words: The featural and unitary semantic space hypothesis. *Cognitive Psychology*, *48*(4), 422–488. doi:10.1016/j.cogpsych.2003.09.001

Vinson, D. P., & Vigliocco, G. (2008). Semantic feature production norms for a large set of objects and events. *Behavior Research Methods*, *40*(1), 183–190.

doi:10.3758/BRM.40.1.183

Vivas, J., Vivas, L., Comesaña, A., Coni, A. G., & Vorano, A. (2017). Spanish semantic feature production norms for 400 concrete concepts. *Behavior Research Methods*, *49*(3), 1095–1106. doi:10.3758/s13428-016-0777-2

Wickham, H., Francios, R., Henry, L., Muller, K., & Rstudio. (2019). dplyr: A grammar of data manipulation. Retrieved from https://cloud.r-project.org/web/packages/dplyr/index.html

Wiemer-Hastings, K., & Xu, X. (2005). Content differences for abstract and concrete concepts. *Cognitive Science*, *29*(5), 719–736. doi:10.1207/s15516709cog0000_33

Wu, L.-l., & Barsalou, L. W. (2009). Perceptual simulation in conceptual combination: Evidence from property generation. *Acta Psychologica*, *132*(2), 173–189. doi:10.1016/j.actpsy.2009.02.002

Zannino, G. D., Perri, R., Pasqualetti, P., Caltagirone, C., & Carlesimo, G. A. (2006). (Category-specific) semantic deficit in Alzheimer's patients: The role of semantic distance. *Neuropsychologia*, *44*(1), 52–61. doi:10.1016/J.NEUROPSYCHOLOGIA.2005.04.008

Table 1

Example of Data Formatted for Tidy Data

word	answer
airplane	you fly in it its big it is fast they are expensive they are at an airport you have to be trained to fly it there are lots of seats they get very high up
airplane	wings engine pilot cockpit tail
airplane	wings it flys modern technology has passengers requires a pilot can be dangerous runs on gas used for travel
airplane	wings flys pilot cockpit uses gas faster travel
airplane	wings engines passengers pilot(s) vary in size and color
airplane	wings body flies travel

Table 2

Lemma and Part of Speech Information from TreeTagger

cue	feature	pos	lemma
airplane	is	verb	be
airplane	fast	adverb	fast
airplane	they	pronoun	they
airplane	are	verb	be
airplane	expensive	adjective	expensive
airplane	they	pronoun	they

Table 3

Multi-Word Sequence Examples for Zebra

cue	combined.lemmas	n
zebra	horse	27
zebra	horse like	1
zebra	look similar horse	1
zebra	relate horse	2
zebra	resemble small horse	1
zebra	stripe similar horse	1

Table 4

Bag of Words Examples for Zebra

cue	lemma	n
zebra	stripe	64
zebra	black	63
zebra	white	61
zebra	animal	54
zebra	horse	32
zebra	africa	28
zebra	zoo	22
zebra	leg	20
zebra	life	20
zebra	eat	17

Table 5

Descriptive Statistics of Text Processing Style

	Multi-Word Sequences			Bag of Words		
Statistics	*M*	*SD*	*r*	*M*	*SD*	*r*
Number of Cue-Features	192.27	99.14	.78	173.44	77.21	.67
Frequency of Idiosyncratic Response	183.29	97.38	.80	160.57	74.26	.69
Frequency of Cue-Feature Response	2.09	3.39	.65	2.70	4.76	.83
Percent of Cue-Feature Response	3.41	5.10	-.64	4.30	4.76	-.62

Note. Correlation represents the relation between the statistic listed for that row and the sample size for the cue.

Table 6

Cosine Overlap with Previous Data Collection

	With Stopwords		No Stopwords	
Statistic	Original	Translated	Original	Translated
B Mean	.55	.58	.69	.74
B SD	.16	.16	.16	.15
M Mean	.33	.50	.39	.59
M SD	.15	.13	.18	.13
V Mean	.50	.50	.60	.59
V SD	.18	.18	.18	.19

Note. Translated values are hand coded lemmatization from Buchanan et al. (2019). B: Buchanan et al. (2019), M: McRae et al. (2005), V: Vinson & Vigliocco (2008). N values are 226, 61, and 68 respectively.

Table 7

Top Ten Ontology Labels

Parts	Function	Location	Category
brush use	brush hair	scissors cut	flute instrument
lawn grass	river water	snow cold	snow white
snail shell	branch tree	farm land	elephant animal
river stream	chair sit	cabin wood	cabbage green
radio music	leaf plant	rocket space	dagger knife
elephant trunk	kitchen food	breakfast day	apple fruit
zebra stripe	hammer nail	stone rock	hammer tool
river flow	garden flower	bacon pig	lion king
door open	oven cook	shoe foot	cabbage vegetable
dragon fire	leaf green	toy play	furniture table

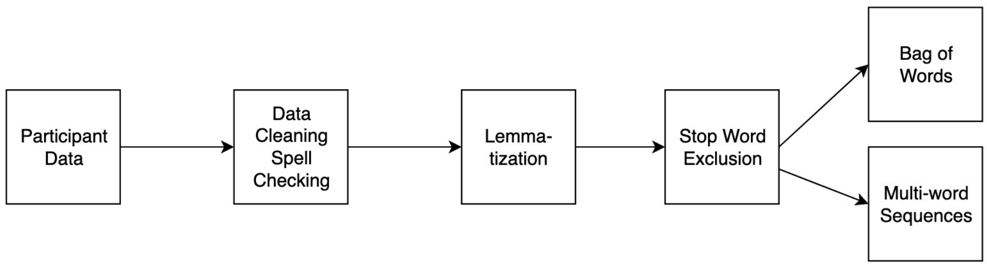

Figure 1. Flow chart illustrating how feature listings are recoded to obtain a standard feature format.

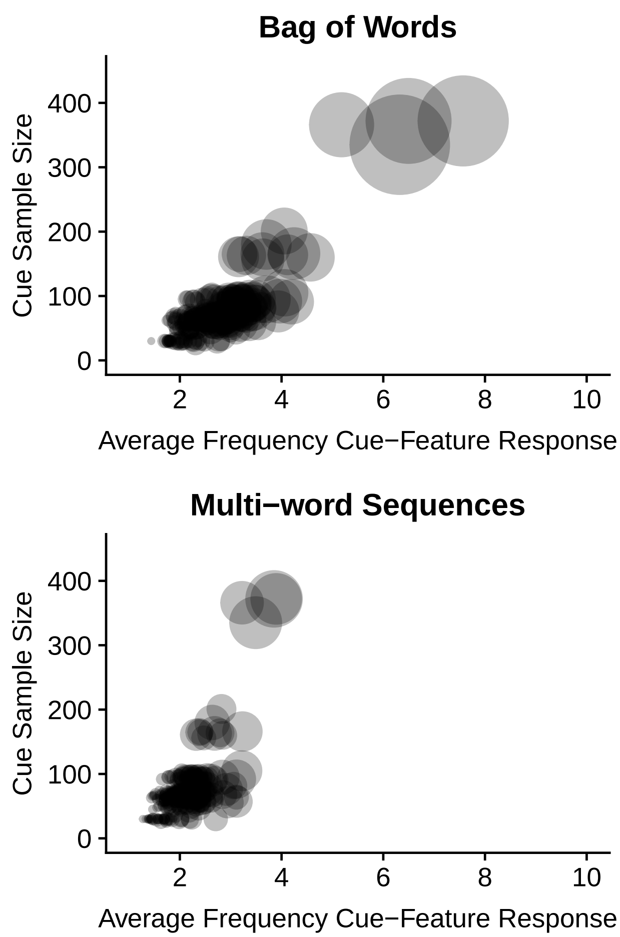

Figure 2. Correlation of sample size with the average cue-feature frequency (left) and percent (right) of response for each cue for both processing approaches. Each point represents a cue word, and the size of the point indicates the variability of the average frequency (left) or percent (right).