A practical primer on processing semantic property norm data

|  |
| --- |
| Erin M. Buchanan1, Simon De Deyne2, & Maria Montefinese3 |
| 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](mailto: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

Word count:

**Introduction**

Semantic property 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, 1990, 2002; Minsky, 1975; Norman & Rumelhart, 1975; Saffran & Sholl, 1999; Smith & Medin, 1981) and computational models (e.g., Caramazza & Shelton, 1998; 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 (e.g., Cree & McRae, 2003; Plaut, 2002; Rogers et al., 2004). For example, the concept horse can be described by encyclopaedic (<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]](#footnote-1)*.

This method has a long history of use by researchers wishing to gain insight into semantic representations of concrete concepts and categories (e.g., Rosch & Mervis1975;Hampton, 1979; McRae et al., 2005; Smith, Shoben & Rips, 1974), and more recently, events and abstract concepts (e.g., Lebani, Bondielli & Lenci, 2015; Wiemer-Hastings & Xu, 2005; Vinson &Vigliocco, 2008).

On the one hand, many studies adopted the property generation task itself to make inferences about word meaning and its computation (e.g., Recchia & Jones, 2012; Wiemer-Hastings & Xu, 2005; Wu & Barsalou, 2009; Santos et al., 2011). On the other hand, many researchers employed the property listing task in order to provide other researchers with a tool of standardized word stimuli and relative semantic measures. 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 (McRae et al., 2005; Vigliocco et al., 2004; Montefinese et al., 2015; 2019), or how different types (Kremer & Baroni, 2011; Zannino et al., 2006) and dimensions of feature informativeness, such as, distinctiveness (Garrard et al., 2001), cue validity (Rosch & Mervis, 1975), relevance (Sartori et al., 2004), semantic richness (Pexman et al., 2008) and significance (Montefinese et al., 2014) are distributed across concepts.

Previous studies showed that all the measures derived from a property generation task appear to account for a variety of behavioural phenomena, in both healthy participants and neurological patients. For example, the similarity between concepts has been shown to predict reaction times in lexical decision (Vigliocco et al., 2004; ) and recognition memory (Montefinese et al., 2015; 2019) tasks in healthy participants, as well as the degree of difficulty in identifying concepts in patients with Alzheimer’s disease (Zannino et al., 2006b). Concepts with many semantic features (i.e., with higher semantic richness value) are recognized faster across different lexical and semantic processing (Grondin et al., 2009; Yap et al., 2012; Pexman, Holyk, & Monfils, 2003; Pexman, Lupker, & Hino, 2002). While distinctive features (e.g., the feature <has stripes> which allows to distinguish a zebra from a horse) have a privileged role in healthy participants (Cree et al., 2006; Garrard et al., 2001), they are more vulnerable to the damage in neurodegenerative diseases compared to shared features because they lack reinforcing connections with the other features (Alathari et al., 2004; Catricalà et al., 2015; Duarte et al., 2009; Garrard et al., 2005; Giffard et al., 2001, 2002; Laisney et al., 2011; Perri, Zannino, Caltagirone, & Carlesimo, 2011, 2013). Moreover, the relevance and significance measures (composite measures of feature informativeness for a given concept) are good predictors of the performance in feature verification and name retrieval tasks in healthy participants and patients with semantic disorders (Montefinese et al., 2014; Sartori & Lombardi, 2004; Sartori, Lombardi & Mattiuzzi, 2005).

Finally, semantic features are also used as a dependent measure. Both psychologists and computational linguists use subjective features to evaluate language-based distributional models of meaning (add refs).

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 (Ruts et al., 2004; De Deyne & Storms, 2008); English (Buchanan et al., 2013; Buchanan et al., 2019; Deveraux et al., 2004; Garrard et al., 2001; McRae et al., 2005; Vigliocco & Vinson, 2008), German (Kremer & Baroni, 2011), Italian (Catricalà et al., 2015; Kremer & Baroni, 2011; Montefinese et al., 2013; Zannino et al., 2006), Portuguese (Marques, 2007) and Spanish (Vivas, Vivas, Comesana, 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 (e.g., Buchanan et al., 2013; 2019; Montefinese et al., 2013; De Deyne et al., 2008) where participants can list the features related to the concept with any kind of semantic relation, other studies use a constrained verbal feature production (e.g., Garrard et al., 2001; Deveraux et al., 2004) 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 (e.g., Kremer & Baroni, 2011; Lenci et al., 2013; Marques, 2007) or a limited number of features to be listed (De Deyne et al., 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 (e.g. McRae et al., 2005) whereas others use 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 (e.g., Vivas et al., 2017; McRae et al., 2005; Montefinese et al., 2013), Buchanan and colleagues (2019) included idiosyncratic features (features produced only by one or a few number of participants), ambiguous words (words with multiple meanings) and 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 behaviour (), 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 this 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. 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.

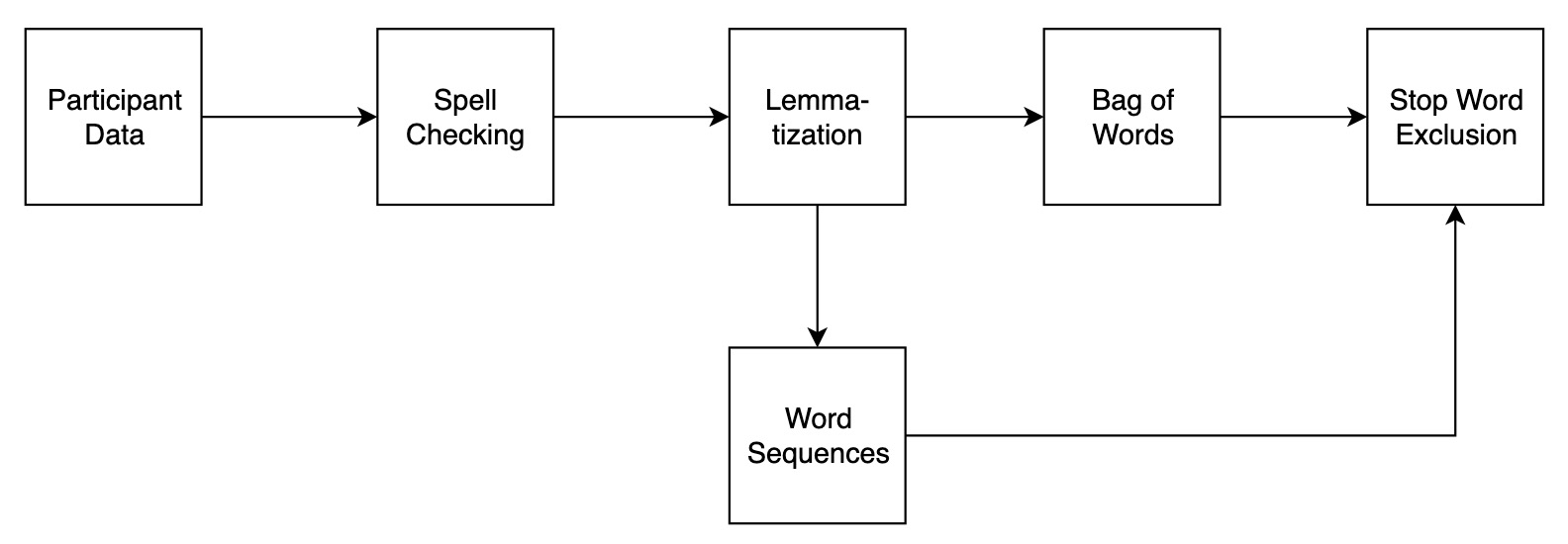


Figure 1: Flow chart illustrating how feature listings are recoded to obtain a standard feature format.

## Materials and Data Format

The data for this tutorial includes 16,544 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.1 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 ??. The original data can be found at <https://osf.io/cjyzw/>.

This 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). Each answer can be checked for misspellings across an entire column of answers, which is in the master dataset. Because participants were recruited in the United States, we used thee 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.

library(hunspell)

# Lower case to normalize  
master$answer <- tolower(master$answer)  
## Install the hunspell package if necessary  
#install.packages("hunspell")  
 # Check the participant answers  
## The output is a list of spelling errors for each line  
spelling\_errors <- hunspell(master$answer, 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.

## Check for suggestions  
spelling\_suggest <- lapply(spelling\_errors, hunspell\_suggest)

For *yelloe*, both *yellow yell* were suggested, and *caramel caramels caramel l camellia camel* were suggested for *caramell*. The suggestions are presented in most probable order, and using a few loops with the substitute (gsub()) function, we can replace all errors with the most likely replacement in a new dataset spell\_checked. 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. 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).

## Replace with most likely suggestion  
spell\_checked <- master  
### Loop over the dataframe  
for (i in 1:nrow(spell\_checked)){  
 ### See if there are spelling errors  
 if (length(spelling\_errors[[i]]) > 0) {  
 ### Loop over all errors  
 for (q in 1:length(spelling\_errors[[i]])){  
 ### Replace with the first answer  
 spell\_checked$answer[i] <- gsub(spelling\_errors[[i]][q],   
 spelling\_suggest[[i]][[q]][1],  
 spell\_checked$answer[i])  
 }  
 }  
}

## 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*)[[2]](#footnote-2).

Lemmatization is performed using the TreeTagger (Schmid, 1994) which is implemented in the koRpus packages (Michalke, 2018). TreeTagger is a trained tagger designed to annotate part of speech and lemma information in text, and parameter files are available for multiple languages.

library(koRpus)  
library(koRpus.lang.en)

lemmas <- spell\_checked  
## Find all types for faster lookup  
all\_answers <- tokenize(lemmas$answer, format = "obj", tag = F)  
all\_answers <- unique(all\_answers)

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.

## This example has both suppressWarnings & suppressMessages  
## You should first view these to ensure proper processing  
temp\_tag <- suppressWarnings(  
 suppressMessages(  
 ## Note: the NULL option is to control for the <unknown> that appears  
 ## to occur with the last word in each text  
 treetag(c(all\_answers, "NULL"),   
 ## Control the parameters of treetagger  
 treetagger="manual", format="obj",  
 TT.tknz=FALSE, lang="en",  
 TT.options=list(path="~/TreeTagger", preset="en"))))

This function returns a tagged corpus object, which can be converted into a dataframe of the token-lemma information. The goal would be to replace inflected words with their lemmas, and therefore, unknown values, number tags, and equivalent values are filtered from the dataset. Table ?? portrays the results from TreeTagger.

## Remove all tags not using  
replacement\_lemmas <- temp\_tag@TT.res  
replacement\_lemmas <- subset(replacement\_lemmas,   
 #ignore punctuation  
 wclass != “punctuation” &  
 #unknown values  
 lemma != “<unknown>” &   
 #numbers  
 lemma!= “@card@" &   
 #token should change more than case  
 tolower(token) != tolower(lemma))

Similar to spelling correction, stri\_replace\_all\_regex() is used to replace the wordforms with their corresponding lemmas from the stringi package (Gagolewski & Tartanus, 2019). Table ?? shows the processed data at this stage.

## Install the stringi package  
#install.packages("stringi")  
library(stringi)  
## Replace all the original tokens with new lemmas using \\b for word boundaries  
lemmas$answer <- stri\_replace\_all\_regex(str = lemmas$answer,   
 pattern = paste("\\b", replacement\_lemmas$token, "\\b", sep = ""),  
 replacement = replacement\_lemmas$lemma,  
 vectorize\_all = F, list(case\_insensitive = TRUE))

## 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 (e.g., (e.g., Garrard et al., 2001; Deveraux et al., 2004), 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 mode (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 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 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 table() function is used to tabulate the final count of n-grams and their frequency.

## Create an empty dataframe   
multi\_words <- data.frame(Word=character(),  
 Feature=character(),   
 Frequency=numeric(),   
 stringsAsFactors=FALSE)   
## Create unique word list to loop over   
unique\_concepts <- unique(lemmas$word)  
## Install dplyr  
#install.packages("dplyr")  
library(dplyr)  
## Loop over each word  
for (i in 1:length(unique\_concepts)){  
 ## Create parts of speech for clustering together  
 temp\_tag <- suppressWarnings(  
 suppressMessages(  
 treetag(c(lemmas$answer[lemmas$word == unique\_concepts[i]], "NULL"),   
 ## Control the parameters of treetagger  
 treetagger="manual", format="obj",  
 TT.tknz=FALSE, lang="en",  
 TT.options=list(path="~/TreeTagger", preset="en"))))  
 ## Save only the dataframe, remove NULL  
 temp\_tag <- temp\_tag@TT.res[-nrow(temp\_tag@TT.res) , ]  
 ## Subset out information you don't need  
 temp\_tag <- subset(temp\_tag,   
 wclass != "comma" & wclass != "determiner" &   
 wclass != "preposition" & wclass != "modal" &  
 wclass != "predeterminer" & wclass != "particle" &  
 wclass != "to" & wclass != "punctuation" &   
 wclass != "fullstop" & wclass != "conjunction" &   
 wclass != "pronoun")  
 ## Create a temporary tibble   
 temp\_tag\_tibble <- as\_tibble(temp\_tag)  
 ## Create part of speech and features combined  
 temp\_tag\_tibble <- mutate(temp\_tag\_tibble,   
 two\_words = paste(token,   
 lead(token), sep = "\_"))  
 temp\_tag\_tibble <- mutate(temp\_tag\_tibble,   
 three\_words = paste(token,   
 lead(token), lead(token, n = 2L),   
 sep = "\_"))  
 temp\_tag\_tibble <- mutate(temp\_tag\_tibble,   
 two\_words\_pos = paste(wclass,   
 lead(wclass), sep = "\_"))  
 temp\_tag\_tibble <- mutate(temp\_tag\_tibble,   
 three\_words\_pos = paste(wclass,   
 lead(wclass), lead(wclass, n = 2L),   
 sep = "\_"))  
 ## Find adjective, noun, verb combinations to cluster on  
 verb\_nouns <- grep("\\bverb\_noun", temp\_tag\_tibble$two\_words\_pos)  
 adj\_nouns <- grep("\\badjective\_noun", temp\_tag\_tibble$two\_words\_pos)  
 verb\_adj\_nouns <- grep("\\bverb\_adjective\_noun", temp\_tag\_tibble$three\_words\_pos)  
 ## Use combined and left over features  
 features\_for\_table <- c(temp\_tag\_tibble$two\_words[verb\_nouns],   
 temp\_tag\_tibble$two\_words[adj\_nouns],   
 temp\_tag\_tibble$three\_words[verb\_adj\_nouns],  
 temp\_tag\_tibble$token[-c(verb\_nouns, verb\_nouns+1,   
 adj\_nouns, adj\_nouns+1,  
 verb\_adj\_nouns, verb\_adj\_nouns+1,   
 verb\_adj\_nouns+2)])  
 ## Create a table of frequencies  
 word\_table <- as.data.frame(table(features\_for\_table))  
 ## Clean up the table  
 word\_table$Word <- unique\_concepts[i]  
 colnames(word\_table) = c("Feature", "Frequency", "Word")  
 multi\_words <- rbind(multi\_words, word\_table[ , c(3, 1, 2)])  
}

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 ??.

## 

## Bag of Words

To be able to evaluate the role of identifying multiword 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.

## Create an empty dataframe   
bag\_words <- data.frame(Word=character(),  
 Feature=character(),   
 Frequency=numeric(),   
 stringsAsFactors=FALSE)   
## Loop over each word  
for (i in 1:length(unique\_concepts)){  
 ## Create a table of frequencies  
 word\_table <- as.data.frame(table(  
 ## Tokenize the words  
 tokenize(  
 ## Put all answers together in one character string  
 paste0(lemmas$answer[lemmas$word == unique\_concepts[i]], collapse = " "),   
 format = "obj", tag = F)))  
   
 ## Clean up the table  
 word\_table$Word <- unique\_concepts[i]  
 colnames(word\_table) = c("Feature", "Frequency", "Word")  
   
 bag\_words <- rbind(bag\_words, word\_table[ , c(3, 1, 2)])  
}  
## Remove punctuation  
bag\_words <- bag\_words[-c(grep('^[[:punct:]]',bag\_words$Feature)), ]

Table ?? shows the top ten most frequent responses to *zebra* given the bag of words approach. The top ten features in zebra indicate a match to the multi-word sequence approach but the inclusion of words such as *be, in, a* indicate the need to remove irrelevant words listed with features.

## Stopwords

As shown in Figure 1, the next stage of processing would be to exclude stopwords, such as *the, of, but*, for either the multi-word sequence or bag of word style processing. The stopwords package (Benoit, Muhr, & Watanabe, 2017) includes a list of stopwords for more than 50 languages. For multi-word sequence processing, these values can be removed by filtering stopwords as unigrams.

## Install the stopwords package or use tm  
#install.packages("stopwords")  
library(stopwords)  
## Remove stop words from either processing approach  
multi\_words\_nostop <- subset(multi\_words,   
 !(Feature %in% stopwords(language = "en",   
 source = "snowball")))  
bag\_words\_nostop <- subset(bag\_words,   
 !(Feature %in% stopwords(language = "en",   
 source = "snowball")))

## Descriptive Statistics

The finalized data now represents a processed set of cue-feature combinations with their frequencies for analysis. Given the differences in sample size across data collection points from Buchanan et al. (2019), this information was merged with the sample data. Table ?? 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 ??. The large standard deviation for both approaches indicates that cues have a wide range of possible features listed. For example for the cue xx we find a total of yy features, whereas for xx2 we find yy2 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 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. Additionally, feature weighting using statistics such as pointwise mutual information could be implemented to discount rare features without excluding them.

*Response strength.* The next two lines of Table ?? 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.

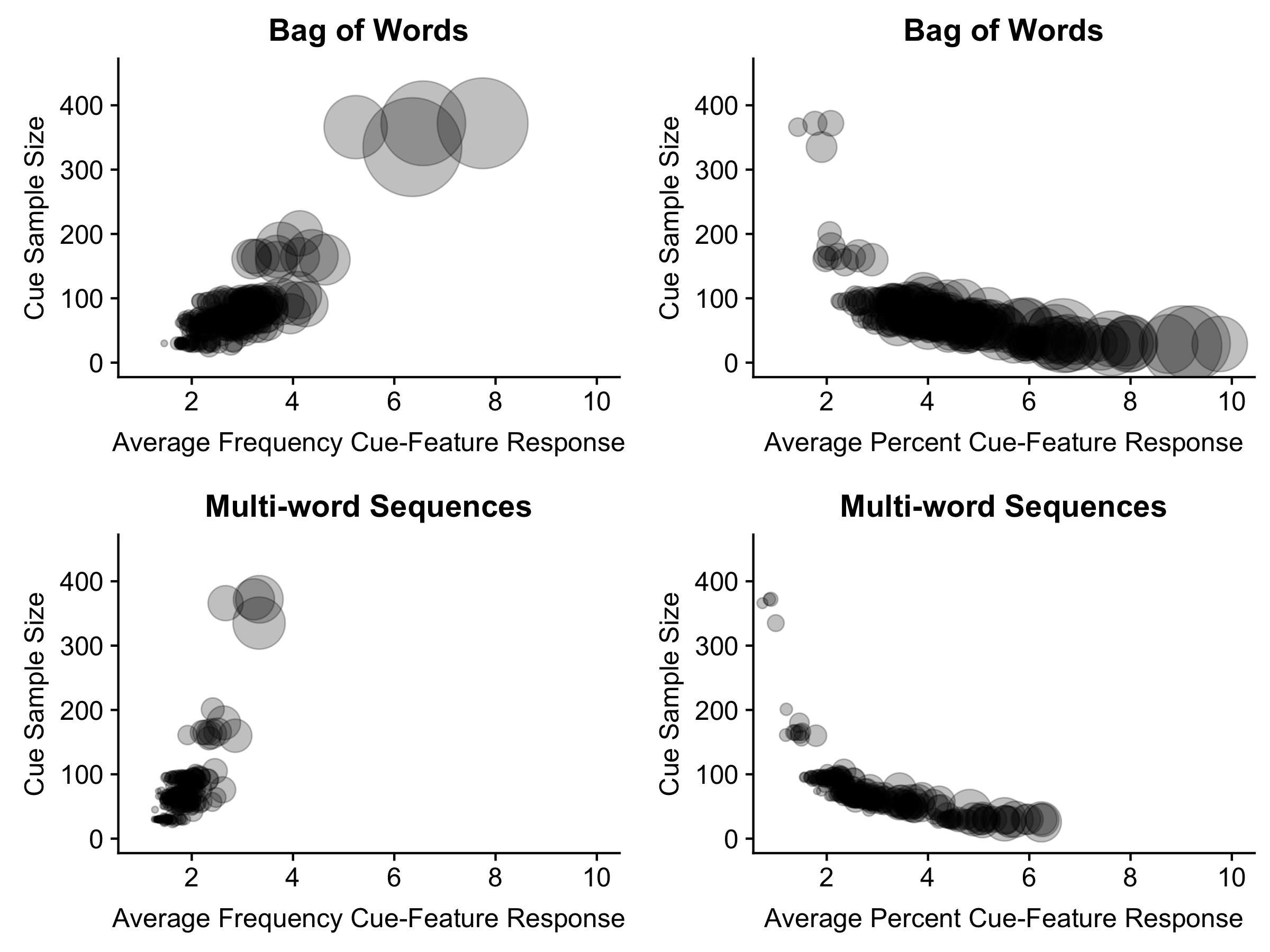


Figure 2: Correlation of sample size with the average cue-feature frequency (left) and percent (right) of response for each cue for both processing approaches. Each point represents a cue word, and the size of the point indicates the variability of the average frequency (left) or percent (right).

## 

## Internal Comparison of Approach

In this section, we show that the bag of words approach processed completely through code matches a bag of words approach that was hand coded from Buchanan et al. (2019). 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, lemmatized features[[3]](#footnote-3) = .66-.67). However, all previous datasets have been reduced by eliminating idiosyncratic features at various points, and therefore, we might expect that noise in this data to reduce the average cosine values. Table ?? shows the role of using a cut-off for low-frequent or idiosyncratic responses by calcuating 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 , 95% CI , , *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, , 95% CI , .

## 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. [SAY SOMETHING ABOUT HANDCRAFTED VS AUTOMATIC PROCEDURE PERFORMANCE WITH REGARDS TO ABOVE]

*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 2259 of the bag of words features (5%). These features were predominately parts (39.9), followed by function (30.5), location (24.0), and category (5.5). Table ?? 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 rulesets.

*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.

# Discussion

* this sort of thing is great for replication purposes, which is pretty important because of the garden of forking paths which applies not just to statistical analyses but also to processing.
* we’ve provided a workflow suggestion that a researcher can use to format their work, along with functions that can be detailed to match any hand processing results.
* weave this to match introduction
* how concrete or abstract the words are

# References

Ashcraft, M. H. (1978). Property norms for typical and atypical items from 17 categories: A description and discussion. *Memory & Cognition*, *6*(3), 227–232. doi:[10.3758/BF03197450](https://doi.org/10.3758/BF03197450)

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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/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](https://doi.org/10.3758/s13428-019-01243-z)

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](https://doi.org/10.1016/S0022-5371(69)80069-1)

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](https://doi.org/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](https://doi.org/10.1016/j.cortex.2013.05.010)

Gagolewski, M., & Tartanus, B. (2019). stringi: Character String Processing Facilities. Retrieved from <https://cran.r-project.org/web/packages/stringi/index.html>

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](https://doi.org/10.3758/s13428-010-0028-x)

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](https://doi.org/10.3758/s13428-013-0323-4)

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](https://doi.org/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>

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](https://doi.org/10.3758/s13428-012-0263-4)

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/vignettes/intro.html{\#}setting{\\_}a{\\_}language](https://cran.r-project.org/web/packages/hunspell/vignettes/intro.html%7b\#}setting{\_}a{\_}language)

Reverberi, C., Capitani, E., & Laiacona, E. (2004). Variabili semantico lessicali relative a tutti gli elementi di una categoria semantica: Indagine su soggetti normali italiani per la categoria “frutta". *Giornale Italiano Di Psicologia*, *31*, 497–522.

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](https://doi.org/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](https://doi.org/10.3758/BF03195597)

Schmid, H. (1994). Probabilistic Part-of-Speech Tagging Using Decision Trees. doi:[10.1.1.28.1139](https://doi.org/10.1.1.28.1139)

Stein, L., & de Azevedo Gomes, C. (2009). Normas Brasileiras para listas de palavras associadas: Associação semântica, concretude, frequência e emocionalidade. *Psicologia: Teoria E Pesquisa*, *25*, 537–546. doi:[10.1590/S0102-37722009000400009](https://doi.org/10.1590/S0102-37722009000400009)

Toglia, M. P. (2009). Withstanding the test of time: The 1978 semantic word norms. *Behavior Research Methods*, *41*(2), 531–533. doi:[10.3758/BRM.41.2.531](https://doi.org/10.3758/BRM.41.2.531)

Toglia, M. P., & Battig, W. F. (1978). *Handbook of semantic word norms*. Hillside, NJ: Earlbaum.

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](https://doi.org/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](https://doi.org/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>

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](https://doi.org/10.1016/j.actpsy.2009.02.002)

1. Throughout this article, features will be distinguished from cues using angular brackets. [↑](#footnote-ref-1)
2. 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. [↑](#footnote-ref-2)
3. In this study, the responses were lemmatized “by hand”. [Explain better]. [↑](#footnote-ref-3)