RESEARCH ARTICLE

A practical primer on processing semantic property norm data

Erin M. Buchanan[1] · **Simon De Deyne**[2] · **Maria Montefinese**[3,4]

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 modeling, stimuli creation, and understanding similarity between concepts. 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 preprocessing (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 long-standing problems and allow researchers to develop new property listing norms overcoming the constraints of previous studies.

Keywords Semantic · Property norm task · Tutorial

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 and Quillian 1969; Jackendoff 1992, 2002; Minsky 1975; Norman and Rumelhart 1975; Saffran and Sholl 1999; Smith

✉ Erin M. Buchanan
  ebuchanan@harrisburgu.edu

1 Harrisburg University of Science and Technology, 326 Market St., Harrisburg, PA 17101, USA

2 The University of Melbourne, Melbourne, Australia

3 University of Padova, Padua, Italy

4 University College London, London, UK

and Medin 1981) and computational models (Caramazza et al. 1988; Farah and McClelland 1991; Humphreys and 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 and 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] While the method is most frequently used to study semantic representations of concrete concepts and categories (McRae et al. 2005; Rosch and Mervis 1975; Smith et al.

---

[1] Throughout this article, *<features>* will be distinguished from CUES using angular brackets and italic font.

1974), it has also been used for other types of concepts, corresponding to verbs (Vinson and Vigliocco 2008), events, and abstract concepts (Lebani et al. 2016; Recchia and Jones 2012; Wiemer-Hastings and Xu 2005).

On the one hand, many studies adopted the property generation task itself to make inferences about word meaning and its computation (Recchia and Jones 2012; Santos et al. 2011; Wiemer-Hastings and Xu 2005; Wu and 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. Indeed, based on data obtained from the property production task, it is then possible to calculate numerous measures and distributional statistics at both the feature and the concept levels. 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 et al. 2019; McRae et al. 2005; Montefinese et al. 2015, 2018; Vigliocco et al. 2004), or how different types (Kremer and Baroni 2011; Zannino et al. 2006a) and dimensions of feature informativeness, such as distinctiveness (Duarte et al. 2009; Garrard et al. 2001), cue validity (Rosch and Mervis 1975), relevance (Sartori and Lombardi 2004), semantic richness (Pexman et al. 2008), and significance (Montefinese et al. 2014), are distributed across concepts.

Efficient ways to collect data online have boosted the availability of large feature listing datasets. These semantic feature norms are now available across different languages: Dutch (De Deyne et al. 2008; Ruts et al. 2004), English (Buchanan et al. 2013, 2019; Devereux et al. 2014; Garrard et al. 2001; McRae et al. 2005; Vinson and Vigliocco 2008), German (Kremer and Baroni 2011), Italian (Catricalà et al. 2015; Kremer and Baroni 2011; Montefinese et al. 2013; Zannino et al. 2006b), Portuguese (Marques et al. 2007), and Spanish (Vivas et al. 2017) as well as for blind participants (Lenci et al. 2013). However, these norms vary substantially in the procedure of data collection and their preprocessing, and this does not facilitate performing cross-language comparisons, 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 et al. 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 *<is ...>*, *<has ...>*, *<does ...>*, and *<made of ...>*. Moreover, authors could instruct the participants to produce a single word as a feature instead of a multi-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 pieces of information, or rather if it should be considered as a unique feature. Second, some authors gave a time limit to provide the feature descriptions (Kremer and Baroni 2011; Lenci et al. 2013; Marques et al. 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 preprocessing 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/preprocess data derived from the feature listing task. Recoding features is sometimes done manually (McRae et al. 2005), whereas others use semiautomatic 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 stopwords, such as *the*, *an*, and *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 for assistance in processing language data. Moreover, building standard automatic procedures to process feature listing data would not only add transparency to the process but also reduce 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. 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,

Fig. 1 Flowchart illustrating how feature lists are recoded to obtain a standard feature format
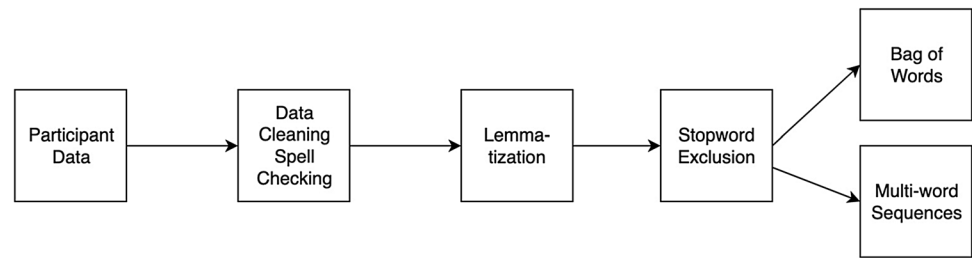
lemmatization, exclusion of stopwords, and final processing in a multi-word sequence approach or a bag-of-words approach. After detailing these steps, the final data form will evaluated and compared to previous norms to determine the usefulness of this approach.

Materials and data format

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

```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$feature_response <- tolower(X$feature_response)
```

The data for this tutorial include 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 et al. (2014). The concepts include 185 nouns, 25 verbs, and 16 adjectives. The concepts were both abstract and concrete, and to describe the concepts, the concreteness ratings collected by Brysbaert

et al. (2014) can be used. In their study, they asked participants to rate words on a scale ranging from 1—abstract (language-based)—to 5—concrete (experience-based)—concepts. 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 feature listing 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 is shown in Table 1, where the `cue` column is the cue, and the `feature_response` column denotes a single participant's response. The original data can be found at https://osf.io/cjyzw/.

The data were 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 additional challenges in unstructured data of this nature. Figure 1 portrays the suggested data processing steps. With structured data entry for participants (e.g., asking participants to type one feature on each line), the multi-word sequence step would be implemented within the data collection design, rather than post-processing. This tutorial

---

[2] A `packrat` project compilation is available on GitHub for reproducibility (Ushey et al. 2018), and this manuscript was written in Rmarkdown with `papaja` (Aust and Barth 2017).

Table 1 Example of data formatted for tidy data

Cue	Participant 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

presents the more difficult scenario to be applicable to more data collection methods.

Spelling

The first step (see Fig. 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 `feature_response` can be checked for misspellings across an entire column of answers, which is in the `X` dataset. Because participants were recruited in the USA, we used the 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 variants of the same language; for example, the "en_GB" would convert to British English spellings.

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>* and *<caramell>*. After checking for errors, the `hunspell_suggest()` function was used to determine the most likely replacement for each error. For *<yelloe>*, both *<yellow>* and *<yell>* were suggested, and *<caramel>* and *<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 and can be used to manually check the validity of the suggestions in a final (optional) step. 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`. For example, by examining `spelling.dict`, we found entries that would need to be corrected: *tast* became *tacit*, *frends* became *fends*, and *musles* became *mules*. Since the spelling dictionary is saved, this will facilitate the additional step of manually examining the output for incorrect suggestions and to add their own corrections. This file could then be reloaded and used in the step below to provide adjusted spelling corrections. Other paid alternatives, such as Microsoft's 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).

```r
# Extract a list of words
tokens <- unnest_tokens(tbl = X, output = token, input = feature_response)
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"))
```

```r
# 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 were collected with a large text box, allowing participants to list multiple features 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 = feature_response, 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 `stri_replace_all_regex()` is used to replace the spelling errors with their corrections from the `stringi` package (Gagolewski and Tartanus 2019). If the `spelling.`

```
# 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)
```

`dict` output file was manually edited, it can be (re)loaded here with `read.csv` to update with the adjusted spelling corrections.[3] 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(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 lexeme set

(i.e., all word forms that have the same meaning, *am, are, is*) to a common lemma (i.e., *be*).[4] 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 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`.

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.

---

[3] For transparency, the updated csv file should be renamed, which also practically keeps one from overwriting their adjustments if they rerun their code. The csv should be loaded as `spelling.dict` to continue with the code below.

[4] 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* for a participle marker.

```
# 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.

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. Within the filter command, we

```
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$feature[tokens.tagged
tokens.tagged$lemma[tokens.tagged$lemma == "@card@"] <- tokens.tagged$feature[tokens.tagged
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 Fig. 1, the next stage of processing would be to exclude stopwords, such as *the, of,* and *but.* The stopwords package (Benoit et al. 2017) includes a list of stopwords for more than 50 languages. At this

have excluded all lemmas in the stopword list provided by the stopwords library. Using stopwords (language = "en", source = "snowball"), one can view the stopword list and edit it for their own needs.

```
# 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 labeled with relations such as *<is-a>*

```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.adj.nouns,
                                                                        verb.adj.nouns

#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)
```

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 (structured dimension extraction and labeling; Baroni et al. 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 (*<usually-sweet>*), verb–noun (*<made-wood>*), or verb–adjective–noun (*<requires-lighting-source>*) sequences. Using TreeTagger on each concept's answer set, we can obtain the parts of speech in context for each lemma. With dplyr (Wickham et al. 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).

This procedure produces appropriate 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

Table 2 Lemma and part of speech (POS) 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

`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)
```

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 response 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.

Descriptive statistics

The finalized data now represent a processed set of cue feature combinations with their frequencies for analysis. The data from Buchanan et al. (2019) were 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

Table 3 Multi-word sequence examples for Zebra

Cue	Combined lemmas	N
zebra	horse	27
zebra	horselike	1
zebra	look similar horse	1
zebra	relate horse	2
zebra	resemble small horse	1
zebra	stripe similar horse	1

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) have recently used 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 how many low-frequency features should be included.

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*>, as 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. This average frequency/percent can be seen as a measure of response strength and was 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 approximates the data from McRae et al. (2005), Vinson and Vigliocco (2008) and Buchanan et al. (2019), thus comparing data processed completely through code to datasets that were primarily hand-coded. These 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 similarity is often used as a measure of semantic similarity, indicating the feature overlap between two sets of cue feature lists. For each concept or cue, it provides an estimate of similarity based on a vector consisting of features with magnitudes corresponding to their frequency. The formula is identical to a Pearson product correlation when the vectors are centered to have mean zeros. First, matching feature (1) frequencies of cues A and B are multiplied and then summed, and this value is divided by products of the vector length of A and B for all features:

$$\frac{\sum_{i=1}^{n} A_i \times B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \times \sqrt{\sum_{i=1}^{n} B_i^2}}$$

As all of the frequencies are positive, 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). The example participant responses provided in this tutorial are a subset of the Buchanan et al. (2019) data, and therefore, if the participant responses were processed in an identical fashion, the cosine values would be nearly 1. Additionally, if the processing detailed here matches the hand coding in the Buchanan et al. (2019), the overlap with the McRae et al. (2005) and Vinson and Vigliocco (2008) should be similar. These values were: original feature cosine = .54−.55 and lemmatized[5] 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.

Table 6 shows the role of using a cutoff for low-frequency or idiosyncratic responses by calculating the cosine values when using varying cutoffs 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. When stopwords were excluded, cosine values indicated somewhat comparable set of data, with lower values for McRae et al. (2005) than previous results in the original feature sets. These values portray that the data processed entirely in *R* produce 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 rating 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

---

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

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 fair 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 twofold. 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 positive correlations with external subjective ratings of cue–cue relatedness in the MEN dataset. We suggest the workflow shown in Fig. 1, and the suggested R code can provide a framework for researchers to use on their own data. In closing, the use of automated procedures will depend on specific use cases and cannot entirely replace careful human annotation (e.g., in the case of spell-checking). It can, however, greatly facilitate such checking.

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 and Caramazza 2013). Typically this involves applying a handcrafted 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 lookup 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 to build a model of English concepts and their properties that aligned with semantic feature production norms. From this

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 for multi-word sequences and bag-of-words approaches

Statistics	Multi-word sequences			Bag of words		
	Mean	SD	r	Mean	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

The correlation (r) represents the relation between frequency of response and sample size

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) were 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 predominantly 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. Taxonomic labeling often represents a large time demand on a researcher or team and by shifting the burden of the taxonomic labeling to a semiautomatic process, this time may be reduced. With the addition of ontology labels to property norm data, theoretical questions about semantic representation can be explored (Jones and Golonka 2012; Santos et al. 2011).

Table 6 Cosine overlap with previous data collection

Statistic	With stopwords		No stopwords	
	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

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

Some limitations

So far, we have not investigated to what extent 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. This result can be explained by the larger context variability of these concepts, but could also reflect to the level of detail in the specific ontologies used to code these features (Recchia and Jones 2012). Pooling together features might improve the quality of the final representation, especially for these types of concepts. 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

Fig. 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)
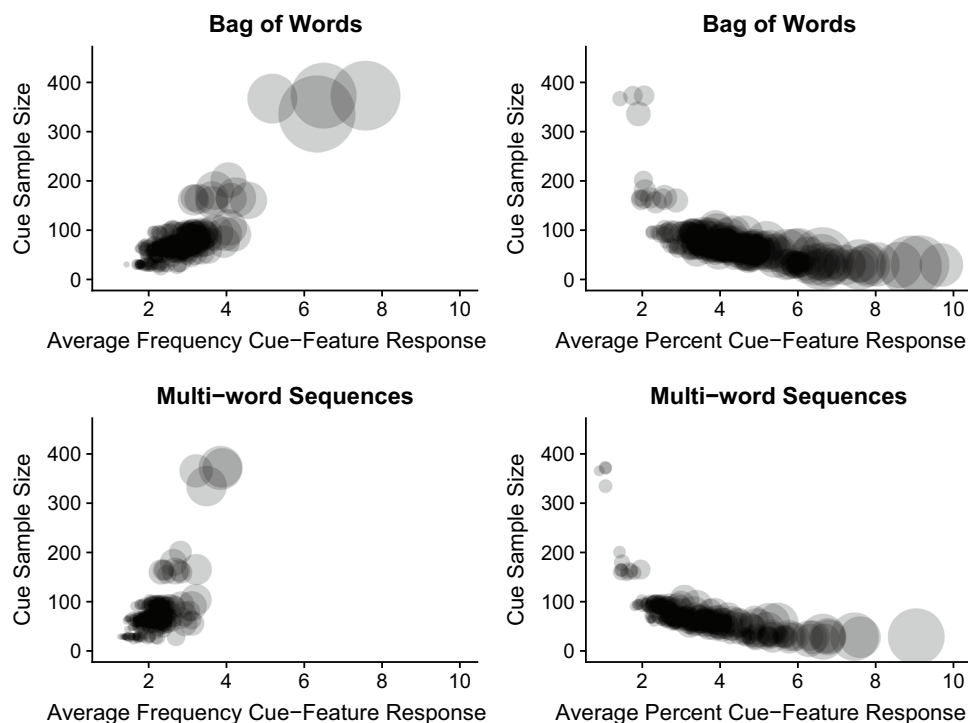
Table 7 Top ten ontology labels

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

within a lookup dictionary to further promote an open and transparent coding for data processing.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflict of interest.

Ethical approval All procedures performed in studies involving human participants were in accordance with the ethical standards of the institutional and/or national research committee (include name of committee + reference number) and with the 1964 Helsinki Declaration and its later amendments or comparable ethical standards.

References

Aust F, Barth M (2017) papaja: create APA manuscripts with R Markdown. https://github.com/crsh/papaja. Accessed 15 Oct 2019

Baroni M, Murphy B, Barbu E, Poesio M (2010) Strudel: a corpus-based semantic model based on properties and types. Cognit Sci 34(2):222–254. https://doi.org/10.1111/j.1551-6709.2009.01068.x

Benoit K, Muhr D, Watanabe K (2017) stopwords: multilingual stopword lists. https://cran.r-project.org/web/packages/stopwords/index.html. Accessed 15 Oct 2019

Bruni E, Tran NK, Baroni M (2014) Multimodal distributional semantics. J Artif Intell Res 49:1–47. https://doi.org/10.1613/jair.4135

Brysbaert M, Warriner AB, Kuperman V (2014) Concreteness ratings for 40 thousand generally known English word lemmas. Behav Res Methods 46(3):904–911. https://doi.org/10.3758/s13428-013-0403-5

Buchanan EM, Holmes JL, Teasley ML, Hutchison KA (2013) English semantic word-pair norms and a searchable web portal for experimental stimulus creation. Behav Res Methods 45(3):746–757. https://doi.org/10.3758/s13428-012-0284-z

Buchanan EM, Valentine KD, Maxwell NP (2019) English semantic feature production norms: an extended database of 4436 concepts. Behav Res Methods 51(4):1849–1863. https://doi.org/10.3758/s13428-019-01243-z

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

Catricalà E, Della Rosa PA, Plebani V, Perani D, Garrard P, Cappa SF (2015) Semantic feature degradation and naming performance. Evidence from neurodegenerative disorders. Brain Lang 147:58–65. https://doi.org/10.1016/J.BANDL.2015.05.007

Collins AM, Quillian MR (1969) Retrieval time from semantic memory. J Verbal Learn Verbal Behav 8(2):240–247. https://doi.org/10.1016/S0022-5371(69)80069-1

Cree GS, 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). J Exp Psychol Gen 132(2):163–201. https://doi.org/10.1037/0096-3445.132.2.163

De Deyne S, Verheyen S, Ameel E, Vanpaemel W, Dry MJ, Voorspoels W, Storms G (2008) Exemplar by feature applicability matrices and other Dutch normative data for semantic concepts. Behav Res Methods 40(4):1030–1048. https://doi.org/10.3758/BRM.40.4.1030

De Queiroz G, Hvitfeldt E, Keyes O, Misra K, Mastny T, Erickson J et al (2019) tidytext: text mining using 'dplyr', 'ggplot2', and other tidy tools. https://cran.r-project.org/web/packages/tidytext/index.html. Accessed 15 Oct 2019

Devereux BJ, Tyler LK, Geertzen J, Randall B (2014) The centre for speech, language and the brain (CSLB) concept property norms. Behav Res Methods 46(4):1119–1127. https://doi.org/10.3758/s13428-013-0420-4

Duarte LR, Marquié L, Marquié JC, Terrier P, Ousset PJ (2009) Analyzing feature distinctiveness in the processing of living and nonliving concepts in Alzheimer's disease. Brain Cognit 71(2):108–117. https://doi.org/10.1016/j.bandc.2009.04.007

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

Farah MJ, McClelland JL (1991) A computational model of semantic memory impairment: modality specificity and emergent category specificity. J Exp Psychol Gen 120(4):339–357. https://doi.org/10.1037/0096-3445.120.4.339

Gagolewski M, Tartanus B (2019) stringi: character string processing facilities. https://cran.r-project.org/web/packages/stringi/index.html. Accessed 15 Oct 2019

Garrard P, Lambon Ralph MA, Hodges JR, Patterson K (2001) Prototypicality, distinctiveness, and intercorrelation: analyses of the semantic attributes of living and nonliving concepts. Cognit Neuropsychol 18(2):125–174. https://doi.org/10.1080/02643290125857

Humphreys GW, Forde EM (2001) Hierarchies, similarity, and interactivity in object recognition: "category-specific" neuropsychological deficits. Behav Brain Sci 24(3):453–476

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

Jackendoff R (2002) Foundations of language (brain, meaning, grammar, evolution). Oxford University Press, Oxford

Jones LL, Golonka S (2012) Different influences on lexical priming for integrative, thematic, and taxonomic relations. Front Hum Neurosci 6:205. https://doi.org/10.3389/fnhum.2012.00205

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

Lebani GE, 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. J Cognit Sci 16(4):401–430. https://doi.org/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. Behav Res Methods 45(4):1218–1233. https://doi.org/10.3758/s13428-013-0323-4

Marques JF, Fonseca FL, Morais S, Pinto IA (2007) Estimated age of acquisition norms for 834 Portuguese nouns and their relation with other psycholinguistic variables. Behav Res Methods 39(3):439–444. https://doi.org/10.3758/BF03193013

McRae K, Cree GS, Seidenberg MS, McNorgan C (2005) Semantic feature production norms for a large set of living and nonliving things. Behav Res Methods 37(4):547–559. https://doi.org/10.3758/BF03192726

Michalke M (2018) koRpus: an R package for text analysis. https://cran.r-project.org/web/packages/koRpus/index.html. Accessed 15 Oct 2019

Minsky M (1975) A framework for representing knowledge. In: Winston PH (ed) The psychology of computer vision. McGraw Hill, Winston, pp 211–277

Montefinese M, Ambrosini E, Fairfield B, Mammarella N (2013) Semantic memory: a feature-based analysis and new norms for Italian. Behav Res Methods 45(2):440–461. https://doi.org/10.3758/s13428-012-0263-4

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

Montefinese M, Zannino GD, Ambrosini E (2015) Semantic similarity between old and new items produces false alarms in recognition memory. Psychol Res 79(5):785–794. https://doi.org/10.1007/s00426-014-0615-z

Montefinese M, Vinson D, Ambrosini E (2018) Recognition memory and featural similarity between concepts: the pupil's point of view. Biol Psychol 135:159–169. https://doi.org/10.1016/J.BIOPSYCHO.2018.04.004

Norman DA, Rumelhart DE (1975) Explorations in cognition. Freeman, San Francisco

Nosek BA, Alter G, Banks GC, Borsboom D, Bowman SD, Breckler SJ, Yarkoni T (2015) Promoting an open research culture. Science 348(6242):1422–1425. https://doi.org/10.1126/science.aab2374

Ooms J (2018) The hunspell package: high-performance stemmer, Tokenizer, and spell checker for R. https://cran.r-project.org/web/packages/hunspell/. Accessed 15 Oct 2019

Peng RD (2011) Reproducible research in computational science. Science (New York, N.Y.) 334(6060):1226–1227. https://doi.org/10.1126/science.1213847

Pexman PM, Hargreaves IS, Siakaluk PD, Bodner GE, Pope J (2008) There are many ways to be rich: effects of three measures of semantic richness on visual word recognition. Psychon Bull Rev 15(1):161–167. https://doi.org/10.3758/PBR.15.1.161

Plaut DC (2002) Graded modality-specific specialisation in semantics: a computational account of optic aphasia. Cognit Neuropsychol 19(7):603–639. https://doi.org/10.1080/02643290244000112

Recchia G, Jones MN (2012) The semantic richness of abstract concepts. Front Hum Neurosci 6:315. https://doi.org/10.3389/fnhum.2012.00315

Rogers TT, Lambon Ralph MA, Garrard P, Bozeat S, McClelland JL, Hodges JR, Patterson K (2004) Structure and deterioration of semantic memory: a neuropsychological and computational investigation. Psychol Rev 111(1):205–235. https://doi.org/10.1037/0033-295X.111.1.205

Rosch E, Mervis CB (1975) Family resemblances: studies in the internal structure of categories. Cognit Psychol 7(4):573–605. 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. Behav Res Methods Instrum Comput 36(3):506–515. https://doi.org/10.3758/BF03195597

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

Santos A, Chaigneau SE, Simmons WK, Barsalou LW (2011) Property generation reflects word association and situated simulation. Lang Cognit 3(1):83–119. https://doi.org/10.1515/langcog.2011.004

Sartori G, Lombardi L (2004) Semantic relevance and semantic disorders. J Cognit Neurosci 16(3):439–452. https://doi.org/10.1162/089892904322926773

Schmid H (1994) Probabilistic part of speech tagging using decision trees. https://www.cis.uni-muenchen.de/~schmid/tools/TreeTagger/data/tree-tagger1.pdf

Smith E, Medin DL (1981) Categories and concepts, vol 9. Harvard University Press, Cambridge

Smith EE, Shoben EJ, Rips LJ (1974) Structure and process in semantic memory: a featural model for semantic decisions. Psychol Rev 81(3):214–241. https://doi.org/10.1037/h0036351

Ushey K, McPherson J, Cheng J, Atkins A, Allaire J (2018) packrat: a dependency management system for projects and their R rackage dependencies. https://cran.r-project.org/web/packages/packrat/index.html. Accessed 15 Oct 2019

Vigliocco G, Vinson DP, Lewis W, Garrett MF (2004) Representing the meanings of object and action words: the featural and unitary semantic space hypothesis. Cognit Psychol 48(4):422–488. https://doi.org/10.1016/j.cogpsych.2003.09.001

Vinson DP, Vigliocco G (2008) Semantic feature production norms for a large set of objects and events. Behav Res Methods 40(1):183–190. https://doi.org/10.3758/BRM.40.1.183

Vivas J, Vivas L, Comesaña A, Coni AG, Vorano A (2017) Spanish semantic feature production norms for 400 concrete concepts. Behav Res Methods 49(3):1095–1106. 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. https://cloud.r-project.org/web/packages/dplyr/index.html. Accessed 15 Oct 2019

Wiemer-Hastings K, Xu X (2005) Content differences for abstract and concrete concepts. Cognit Sci 29(5):719–736. https://doi.org/10.1207/s15516709cog0000_33

Wu L-L, Barsalou LW (2009) Perceptual simulation in conceptual combination: evidence from property generation. Acta Psychol 132(2):173–189. https://doi.org/10.1016/j.actpsy.2009.02.002

Zannino GD, Perri R, Pasqualetti P, Caltagirone C, Carlesimo GA (2006a) Analysis of the semantic representations of living and nonliving concepts: a normative study. Cognit Neuropsychol 23(4):515–540. https://doi.org/10.1080/02643290542000067

Zannino GD, Perri R, Pasqualetti P, Caltagirone C, Carlesimo GA (2006b) (Category-specific) semantic deficit in Alzheimer's patients: the role of semantic distance. Neuropsychologia 44(1):52–61. https://doi.org/10.1016/j.neuropsychologia.2005.04.008