Computational Construction Grammar and Collostructional Analysis on a PropBank-annotated Corpus

Thomas Moerman

This paper uses a Computational Construction Grammar (CCG) approach and collostructional analysis on a PropBank-annotated corpus. It was written during an internship at the VUB AI-Lab. The corpus and dataset were accessed through this internship.

Table of contents

List of Figures

List of Tables

1 Introduction

Construction Grammar (CxG) is a general term for a family of theories used for describing and interpreting language. It emphasises the importance of regular patterns of language use, known as constructions. It argues that these constructions capture all linguistic knowledge in the shape of form-meaning pairs. In other words, in CxG the traditional concepts of lexicon and grammar are combined into one. There is no distinction between the two as form is inherently connected to meaning (**allthecxgboysandgirls?**). Collostructional analysis, developed by (**griesstefall?**), is a combination of several quantative (statistical) methods for examining the relationship between words (lemmas) and structures. The term collostruction, a blend of the words collocation and construction, describes the method of measuring the level of attraction or repulsion that words have towards specific syntactic constructions.

Most studies using collostructional analysis have focused on the relationship between verbs and constructions that convey information about argument structure ((**gries2013?**)). This present paper follows in those footsteps but will take a slightly different approach. It still studies the patterns of co-occurrence between verbs and specific English argument constructions. However, instead of only examining the lemma, this paper focuses on the role-set or word sense of a specific lemma in relation to its place in an argument structure construction. It does this by performing collustructional analysis (**specifywhichones?**) on a data set taken from the propbank-annotated OntoNotes 5.0 corpus. What follows is a brief description of the corpus and data set used.

OntoNotes 5.0 (**Weischedel2013?**) is a corpus consisting of a subset of English, Chinese and Arabic texts. It can be described as a broad-coverage corpus as it spans several genres including religious texts, telephone conversations, news articles and weblogs. In total it consists of 2.9 million words. The corpus is annotated with a number of different layers of information. These are presented in the corpus as follows:

-> sample-paper_files/figure-html/example-ontonotes-annotation.png/) (**VanEeckePPT?**)

In the above example, the utterance "With their unique charm, these well-known cartoon images once again caused Hong Kong to be a focus of worldwide attention." is annotated with a treebank, propbank, word sense annotation, ontology, coreference and entity names annotation layer. A complete description of these annotations can be found in (**Xue2012?**). For the purpose of the present study, the propbank annotation layer is of particular interest. The propbank layer (https://propbank.github.io/) annotates the argument structure of verbs. It does this by providing a list of possible roles for each verb. These roles are called arguments. The word sense annotation layer provides a list of possible word senses for each word. These word senses are also called role sets. The propbank layer is used to identify the argument structure. The word sense layer is used to identify the specific word sense of a verb. To illustrate how these annotations should be interpreted, consider the following utterance "She gave Peter a watch." as shown in Table 1:

Table 1: Example of the PropBank annotation for the verb "give.01" in the utterance "She gave Peter a watch."

roleType	pos	string	indices	roleset	lemma
arg0	np	She	0		
v	v	gave	1	give.01	give
arg2	np	Peter	2		
arg1	np	a watch	3		

In this utterance, the verb "gave" would be annotated with the lemma "give" and the role set "give.01" which is described as "transfer" in https://propbank.github.io/v3.4.0/frames/. The verb here is the frame evoking element (FEE). The role set "give.01" has three arguments: "Arg0", "Arg1" and "Arg2". In propbank annotation this means that "Arg0" is the giver, "Arg1" is the thing given and "Arg2" is the entity given to. In traditional grammar, these arguments would be described as the agent (Arg0), patient (Arg1) and instrument/benefactive/attribute (Arg2). This paper will use the propbank terminology. This means that the above utterance would be annotated as follows: "She(Arg0-NP) gave(FEE-V) Peter(Arg2-NP) a watch(Arg1-NP)". In other words, the argument structure construction in which the roleset "give.01" is embedded is "Arg0-NP FEE-V Arg2-NP Arg1-NP". This can be recognised as a ditransitive construction. This paper focusses broadly on ditransitive constructions in its various manifestations.

The data set used for this analysis is a subset of the OntoNotes 5.0 corpus. This data set was extracted from the corpus using the (**CCxG?**) Explorer (**VanEecke2018?**). The CCxG Explorer is a tool developed by the Evolutionary & Hybrid AI (EHAI) research team at the VUB Artificial Intelligence Lab. Its goal was twofold. From a broad-coverage corpus, it wanted to, first, gain linguistic insights from large-scale construction grammar analyses regarding the English argument structure and, secondly, show the application potential of CCxG by operationalising it on a large scale (Beuls & Van Eecke, 2021). As a result, the CCxG Explorer allows

usage-based linguists to search for corpus examples that match a particular semantic structure. This is useful because it provides the option to find examples of morphosyntactic phenomena without the need to identify them explicitly ((**Beuls?**) & Van Eecke, submitted). The CCxG Explorer can be accessed on used on the web at https://ehai.ai.vub.ac.be/ccxg-explorer/. Its source code is publically available on Gitlab as part of the babel toolkit (**babelrepo?**). The method and accuracy of the CCxG Explorer has not yet been established. This is currently being investigated. This is why this present paper takes a cautious approach to the results.

To extract the ditransitive constructions, the following schema was used in the CCxG Explorer: "Arg0-NP FEE-V Arg1-NP Arg2". The precise order is not taken into account which means that, for example, the following constructions are also extracted: "Arg0-NP FEE-V Arg2-NP Arg1", "Arg1-NP FEE-V Arg0-NP Arg2" and "Arg1-NP FEE-V Arg2-NP Arg0". The reason for this is that the order of the arguments is not fixed in ditransitive constructions and this schema allows for a more broad exploration. Further, it should be noted that there is no part of speech specified in the Arg2 slot. This is because the Arg2 slot can be filled by not only a NP but also, for example, a PP which would result in the dative alternation of the ditransitive construction.

The search results for this particular schema contained 9339 utterances. They were downloaded as a .json file. This .json file was the raw data for this analysis. It contains the string of the utterances and the roles that are defined in that utterance. The roles are further specified by roleType, part of speech (pos), string, indices, role set and lemma. In those 9339 utterances, there were 844 unique lemmas, 924 unique rolesets and 87 unique argument structure constructions which appeared in the previously mentioned schema.

It is not this study's aim to come to any definitive conclusions about the relationship between word sense and argument structure constructions. Instead, its goal is explore the possibilities of using propbank-annotated corpora and Computational Construction Grammar (CCxG) in a collostructional analysis.

This paper consists of several sections. First, a short overview of the (theoretical) background regarding the CCxG Explorer is provided. In addition, the research methodology is presented, specifically on how the data set was analysed. This is followed by a presentation, analysis and discussion of the relevant data. The analysis section consists of two parts. First, the cleaned data is examined and second, the results from the Distinctive Collexeme Analysis (DCA) are presented and discussed.

2 Data Processing and Methodology

The raw data was processed in R using the 'jsonlite' package (**jsonlite?**). The data was then converted to a data frame and the relevant columns were selected. The raw data file does not contain the full argument structure construction in one string. To identify and classify the argument structure construction from each utterance a function was used. The resulting data

Table 2: Raw corpus data used for the subsequent analyses (first 10 rows)

arg_struc_cxn	roleset	lemma
arg0(np)-v(v)-arg1(np)-arg2(c("s", "vp"))	invite.01	invite
arg0(np)-v(v)-arg1(np)-arg2(pp)	force.01	force
arg0(np)-v(v)-arg1(np)-arg2(pp)	connect.01	connect
arg0(np)-v(v)-arg1(np)-arg2(pp)	receive.01	receive
arg0(np)-v(v)-arg1(np)-arg2(pp)	draw.02	draw
arg0(np)-v(v)-arg1(np)-arg2(pp)	lay.01	lay
arg0(np)-v(v)-arg1(np)-arg2(c("s", "vp"))	order.01	order
arg0(np)-v(v)-arg1(np)-arg2(np)	call.01	call
arg0(np)-v(v)-arg1(np)-arg2(pp)	deal.02	deal
arg0(np)-v(v)-arg1(np)-arg2(adjp)	turn.02	turn

frame contains the argument structure construction, role set and lemma. This data frame contains all the data used for the analysis. This is shown in Table 2.

Table 2 shows a small sample of data used for subsequent analyses. The table contains three columns: "arg_struc_cxn", "roleset", and "lemma". The "arg_struc_cxn" column lists the argument structure construction that the roleset occurs in. The "roleset" column lists the specific roleset that the lemma takes on. The "lemma" column lists the lemma of the verb.

As previously mentioned, the type of analysis used for this paper is a collustructrional analysis. According to Stefanowitsch (**Steffi2013?**), there are several types of collustructrional analysis. There is a Simple Collexeme Analysis (SCA), Distinctive Collexeme Analysis (DCA) and Co-varying Collexeme Analysis (CCA). The type of analysis used for this paper is the Distinctive Collexeme Analysis. Since this is the only relevant type of analysis for this paper, the other types of analysis will not be discussed further.

DCA (**GriesStef2004a?**) compares all words that occur in a slot of two similar constructions. It is based on the frequency of the word and constructions it occurs in. (**StefGries2013?**) presents the following Table 3 to illustrate which frequency information is needed for a DCA.

Table 3: Frequency information needed for a distinctive collexeme analysis

	Word l of Class L	Other Words of Class L	Total
Construction c1 of Class C	Frequency of L(l) in C(c1)	Frequency of L(-l) in C(c1)	Total frequency of C(c1)
Construction c2 of Class C	Frequency of L(l) in C(c2)	Frequency of L(-l) in C(c2)	Total frequency of C(c2)
Total	Total frequency of L(l) in C(c1,c2)	Total frequency of L(-l) in C(c1,c2)	Total frequency of C(c1,c2)

In this table, the frequency of a word (l & -l) and construction (c1 & c2) are mapped to each other to create a contingency table containing the frequency of l in c1, l in c2, -l in c1, -l in c2. These are then combined to create a total frequency. Such a contingency table can then be used to perform a contingency test to return association measures like, for example, the Fisher-Yates p score. The Fisher-Yates p score is the score that is most used in these types of analyses (**allDCAref?**) and will, therefore, be discussed more thoroughly than the other assocation measures. It is used as a measure of the strength of association between, in this case, a word and a construction. the p-value represents the probability of obtaining a test statistic as extreme or more extreme than the one observed. A small p-value (typically less than 0.05) suggests that the null hypothesis can be rejected, and that the difference in co-occurrence frequencies between the two words being compared is not due to chance. Conversely, a large p-value (typically greater than 0.05) suggests that there is not enough evidence to reject the null hypothesis and that the difference in co-occurrence frequencies may be due to chance. In other words, a small p-value means that the difference in co-occurrence frequencies between the two words is statistically significant and it is unlikely that the observed difference is due to chance Levshina (2015). It is important to note that in DCA there is a focus on the differences between constructions. To uncover similarities between construction a SCA can be used (**GriesStef2004a?**) but this is not done here given the limited scope of the present research.

An extension of DCA is the possibility to perform it on a data set that has more than two types of constructions. This is referred to as Multiple Distinctive Collexeme Analysis (MDCA) (**StefGries2004a?**). MDCA is a method that can be used to compare the distinctive collexemes of multiple constructions. It is based on the same principles as DCA, but instead of comparing two constructions, it compares multiple constructions. In order to perform a MDCA on a data set a multidimensional contingency table is required. (**GriesStef2013?**) gives the following Table 4 to illustrate which frequency information is needed for a MDCA.:

Table 4: Frequency information needed for a multiple distinctive collexeme analysis

	Word l of Class L	Other Words of Class L	Total
Construction c1 of Class C	Frequency of L(l) in C(c1)	Frequency of L(-l) in C(c1)	Total frequency of C(c1)
Construction c2 of Class C	Frequency of L(l) in C(c2)	Frequency of L(-l) in C(c2)	Total frequency of C(c2)
...	...	...	...
Construction c(n)	Freq. of L(l) in C(cn)	Freq. of L(-l) in C(cn)	Total frequency of C(cn)
Total	Total frequency of L(l) in C(c1,c2, ...n)	Total frequency of L(-l) in C(c1,c2, ...n)	Total frequency of C(c1,c2, ...n)

As can be observed in the above table, the difference is that in the MDCA contingency table there is an n number of constructions represented in the columns. Due to the limited scope of this paper, only the DCA will be discussed in the analysis section. However, the MDCA was still performed and the results are available in the appendix because it could prove useful for future research.

The DCA is used to compare the distinctive collexemes of the two most frequent constructions in the data set. The MDCA can be used to compare the distinctive collexemes of all constructions in the data set. To perform both types of analyses, a R script developed by (**GriesStef2013?**) was used and adapted to suit the data set in this paper. With regards to the DCA, this script calculates the association measures for all words (lemma / role set) in relation to the two most frequent argument structure constructions from the data set and returns a table containing the association measures for each of those word. In total, eightDE association measures are given. They are presented in Table 5. Not all of them are used in the analysis but they are presented for completeness and to give a general overview of the measures that are available when using DCA on a PropBank-annotated corpus. A comparison between these measures is not examined in this paper.

Table 5: Association measures used in the DCA

Association Measure	Full Name	Score Indication
LLR	Log-Likelihood Ratio	The higher the value, the stronger the association between the two variables.
PRES	Pearson Residual	The closer the value is to 0, the stronger the association between the two variables.
LOR	Log Odds Ratio	The higher the value, the stronger the association between the two variables.

Association Measure	Full Name	Score Indication
MI	Mutual Information	The higher the value, the stronger the association between the two variables.
DPC2W	Difference in Probability of Construction to Word	The higher the value, the stronger the association between the two variables.
DPW2C	Difference in Probability of Word to Construction	The higher the value, the stronger the association between the two variables.
FYE	Fisher-Yates Exact Test	The lower the value, the stronger the association between the two variables.

3 Results and Discussion

In the following section, an exploratory analysis will be conducted on the cleaned corpus data and on the resulting data from the DCA. The analysis departs from an overview of the frequency of the lemmas and rolesets in specific argument constructions. In Table 6, a random subset of an alphabetically sorted list is shown. The argument structure constructions are ordered based on the frequency in which they appear in the data set.

What Table 6 shows is that a lemma manifests itself in different rolesets in different constructions. The logical consequence of this is that there are more role sets than there are lemmas. In other words, a verb can be used in different senses and these senses are potentially linked to a specific construction. For example, the lemma "catch" appears in role sets "catch.02" and "catch.03" and the lemma "charge" appears in the role sets "charge.01", "charge.04" and "charge.05". Table 7 illustrates these role sets with their semantic meaning and an example utterance from the corpus.

Table 7: Example of the role sets and their semantic meaning

Lemma	Roleset	Meaning	Utterance	Arg_Struc_Cxn
catch	catch.02	come upon, find	"We **caught** them cheating"	arg0(np)-v(v)-arg1(np)-arg2(v)
	catch.03	trap	"...asking locals not to [...] **catch** them in nets."	arg0(np)-v(v)-arg1(np)-arg2(pp)
charge	charge.01	asking price	"He **charge** you a fortune?"	arg0(np)-v(v)-arg1(np)-arg2(pp)
	charge.04	buy on credit	"...car buyers **charge** [...] their purchase on the [...] card"	arg0(np)-v(v)-arg1(np)-arg2(pp)
	charge.05	make an allegation	"...they indicated to **charge** Mr. Noriega himself..."	arg0(np)-v(v)-arg1(np)-arg2(pp)

Table 6: Alphabetical sorted summary of the data set (random 20 rows and first 8 columns).

lemma	roleset	total	arg0(np)-v(v)-arg1(np)-arg2(pp)	arg0(np)-v(v)-arg2(np)-arg1(np)	arg0(np)-v(v)-arg1(np)-arg2(c("s","vp"))	arg0(np)-v(v)-arg1(np)-arg2(vp)	arg0(np)-v(v)-arg1(np)-arg2(np)
cast	cast.02	2	2				
castigate	castigate.01	3	3				
catapult	catapult.01	1	1				
catch	catch.02	5	1			3	
catch	catch.03	3	3				
cede	cede.01	2	2				
challenge	challenge.01	5	3		2		
change	change.01	22	21		1		
channel	channel.01	2	2				
characterize	characterize.01	8	6				
charge	charge.01	3	1	2			
charge	charge.04	1	1				
charge	charge.05	21	21				
charter	charter.01	2			1		
chastise	chastise.01	3	3				
chide	chide.01	2	2				
choose	choose.01	3	1				
cite	cite.01	12	11				
cite	cite.02	1	1				
claim	claim.01	1	1				
clean	clean.01	2	2				

Table 7 shows information about the different senses of a single lemma (in this case "catch" and "charge"). Each row represents a different sense of the lemma, identified by its role set. The columns in the table provide information about the meaning of the sense, as defined on the PropBank website, an example utterance from a dataset, and the argument structure construction (arg_struc_cxn) of that utterance. To further clarify how the argument structure construction is applied to the utterance, Table 8 is given:

Table 8: Example of the argument structure construction of the utterance "We caught them cheating"

roleType	pos	string	indices	roleset	lemma
arg0	np	We	0		
v	v	caugth	1	catch.02	catch
arg1	np	them	2		
arg2	v	cheating	3		

Table 8 is showing the information of the sentence "We caught them cheating." as annotated in the OntoNotes corpus. The lemma is "catch", the roleset is "catch.02" and the roleType, pos (part of speech), string and indices of the words that fill the role are given in the table. The word 'We' is filling the role of arg0, the word 'caught' is filling the role of v, the word 'them' is filling the role of arg1 (entity) and the word 'cheating' is filling the role of arg2 (attribute).

The next Table 9 shows a similar list to Table 6 but now no longer alphabetically sorted but sorted based on the frequency of the lemmas and rolesets in the data set. The argument structure constructions are still ordered based on the frequency in which they appear in the data set.

Table 9 indicates the frequency of the different rolesets in different constructions as represented by the column names. The "total" column represents the sum of all the values in that row or, in other words, the total frequency of the specific roleset. It is not a surprise that the lemma "give" in the sense "give.01" is by quite a margin the most frequent in the previously defined argument structure schema. This has been well-established in previous research (**giveresearch?**). It could be expected that due to "give.01" its high frequency in the "arg0(np)-v(v)-arg2(np)-arg1(np)" construction, this construction is one of the most frequent constructions in the data set. This can be examined by looking at Table 10 which shows the frequency of the different argument structure constructions in the data set.

Table 10 that shows the distribution of different argument structures (arg_struc_cxn) in the data set. Each row in the table represents a different argument structure and the "Freq" column shows the number of times that argument structure was found. It shows the ten most frequent argument structure constructions. The three most frequent argument structure in the data set are "arg0(np)-v(v)-arg1(np)-arg2(pp)" (5497 occurances), "arg0(np)-v(v)-arg2(np)-arg1(np)" (1180 occurances) and "arg0(np)-v(v)-arg1(np)-arg2(c("s", "vp"))" (741

Table 9: Frequency ordered overview of the data set (first 10 rows and 8 columns).

lemma	roleset	total	arg0(np)-v(v)-arg1(np)-arg2(pp)	arg0(np)-v(v)-arg2(np)-arg1(np)	arg0(np)-v(v)-arg1(np)-arg2(c("s", "vp"))	arg0(np)-v(v)-arg1(np)-arg2(vp)	arg0(np)-v(v)-arg1(np)-arg2(np)
give	give.01	976	224	662			1
put	put.01	443	355				2
use	use.01	381	76		284		
bring	bring.01	243	175	27			4
take	take.01	225	189		3	1	1
call	call.01	195	3			2	130
tell	tell.01	192	19	150			
send	send.01	188	111	49		1	3
spend	spend.02	134	49		74	2	
force	force.01	118	22		2	80	

Table 10: Frequency ordered overview of the argument structure constructions (first 10 rows).

arg_struc_cxn	Freq
arg0(np)-v(v)-arg1(np)-arg2(pp)	5497
arg0(np)-v(v)-arg2(np)-arg1(np)	1180
arg0(np)-v(v)-arg1(np)-arg2(c("s", "vp"))	741
arg0(np)-v(v)-arg1(np)-arg2(vp)	315
arg0(np)-v(v)-arg1(np)-arg2(np)	226
arg0(np)-v(v)-arg2(pp)-arg1(np)	142
arg1(np)-arg0(np)-v(v)-arg2(np)	119
arg0(np)-v(v)-arg2(rp)-arg1(np)	113
arg0(np)-v(v)-arg1(np)-arg2(sbar)	111
arg1(np)-arg0(np)-v(v)-arg2(pp)	110

occurances). The "arg0(np)-v(v)-arg2(np)-arg1(np)" is the construction in which "give.01" was most frequent with 662 occurances. In other words, roughly 56% of the times "arg0(np)-v(v)-arg2(np)-arg1(np)" appeared it was in context of "give.01". This is important to keep in mind when interpreting the results of the DCA analysis. This analyis was performed on the two most frequent type of argument structure construction occuring in the data set. The high frequency of a verb in a specific argument structure construction therefore has an impact on the results of the DCA analysis, more specifically on which types of constructions are analysed. This could be an indication that perhaps a MDCA is a better approach to analyse these types of data set which contain a large number of different argument structure constructions conforming to a specific schema. However, as mentioned before the results of the MDCA are not discussed in this paper but the raw data of the MDCA is available in the appendix for further research.

The next part of the analysis section will depart from the DCA lemma and role set table, which contains all calculated association scores. The analysis starts with giving a general overview of the most frequent and distinctive collostructions found in the corpus. Then, it will proceed to examine the specific association measures in order to gain insight into the patterns and preferences in the data. Examples of the collostructions will be provided to assist in understanding the findings and interpreting the results. The full results of the analyses are available in the appendix. The results of the DCA are presented in Table 11:

In these tables, the first column "WORD" shows the lemma / roleset of the word. The next two columns show the frequency of the word in the two different constructions: "arg0(np)-v(v)-arg1(np)-arg2(pp)" and "arg0(np)-v(v)-arg2(np)-arg1(np)". The next columns "LLR", "PEARSONRESID", "LOGODDSRATIO", "MI", "DELTAPC2W", "DELTAPW2C" and "FYE" are association measures that gives information on the strength of the association between the word and the construction it appears in.

A first observation that can be made is that the two most frequent argument structure constructions and were thus used for the DCA are the arg0(np)-v(v)-arg1(np)-arg2(pp) and arg0(np)-v(v)-arg2(np)-arg1(np). These could, in more traditional terms, be described as the propositional dative and ditransitive construction. Additionally, there are three categories on how the role sets / lemmas are related to the two constructions. They can appear exclusively in either arg0(np)-v(v)-arg1(np)-arg2(pp) or arg0(np)-v(v)-arg2(np)-arg1(np), or they appear in both. The following tables show

To interpret the results of the DCA they are compared to a study by (**GriesStef2004a?**) where a DCA was used to compare collexemes in a ditransitive and to-dative construction. The results of this study are presented in the following table:

——————— results table stef gries 2004

discussion DCA

The results of the MDCA are presented in the following table. It shows

Table 11: Results DCA analysis.

WORD	arg0.np	arg0.arg1.np	PRED	arg2.np.arg1.np	PRES	LOR	MI	DPC2W	DPW2C	FYE
put.01	355	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	142	3.7	27	0.28	0.06	0.19	16.0
take.01	189	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	75	2.7	26	0.28	0.03	0.18	8.6
accuse.01	100	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	39	1.9	26	0.28	0.02	0.18	5.1
use.01	76	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	30	1.7	26	0.28	0.01	0.18	3.8
receive.01	75	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	29	1.7	26	0.28	0.01	0.18	4.0
keep.04	73	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	29	1.7	25	0.28	0.01	0.18	3.8
get.01	71	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	28	1.6	25	0.28	0.01	0.18	4.0
throw.01	68	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	27	1.6	25	0.28	0.01	0.18	3.9
thank.01	57	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	22	1.5	25	0.28	0.01	0.18	3.4
describe.01	52	0	arg0(np)-v(v)-arg1(np)-arg2(pp)	20	1.4	25	0.28	0.01	0.18	3.0

```
dataset <- read_tsv(here("register-analysis.tsv"), show_col_types = FALSE)
fa <- dataset %>% data.frame(row.names = "filename") %>%
  as.matrix() %>% factanal(factors = 4, scores = "regression")
```

To interpret the results of the MCA

4 Conclusion

The conclusion should recap the paper, summarizing what was said in each section and how everything ties together. It might feel redundant, in particular in relation to the introduction, but that's precisely the point: for someone who has not heard your ideas, redundancy is key to understand. Be clear about how each section contributes to your main point and what the take-home message is.

Of course, the language of the paper should be formal and academic (I appreciate puns and jokes, but don't lower the register too much). Coherence in the ideas, cohesion between the sentences and appropriate use of the technical vocabulary and of connectors (e.g. *however*, *in contrast*, *while*…) are important and **will be evaluated**. Not because of "language" evaluation but because these aspects are crucial for understanding, and if the reader needs to read a sentence many times and/or have previous knowledge of your process in order to understand the text, it's not well written. I also recommend checking out the `{spelling}` package to run some spelling checks on your files!

References

Levshina, Natalia. 2015. *How to do linguistics with R: Data exploration and statistical analysis.* Amsterdam; Philadelphia: John Benjamins Publishing Company.