Cloud Gazing As A Pro

Mariana Montes

2021-03-05

Contents

Introduction

Here I'm starting the first draft ever of my PhD dissertation. There are reports scattered all over the place, but here I will try to write things already thinking of a Final Product. I need a layout (briefly discussed a week ago) and I will slowly start building the final text.

The original title (in any case, the title of my project) was *Methodological triangulation in corpus-based distributional semantics*, which indicates I'm going to compare several corpus-based methods.

I was going to do some behavioral profiles, but that might prove a bit tough right now. A bit too much. In any case, I did compare manual annotation with token-vector models.

What I really did do was develop the visualization, based on Thomas Wielfaert's original code, analyze the parameter settings in multiple ways, checking different combinations and exploring different avenues, and lead, check, study and compare manual annotation of 33 lemmas.

That is what I'm going to write about.

Part I

Visualization tool

Chapter 1

An interface to the world of clouds

In this part (which will have who knows how many chapters) I mean to describe the visualization tool. It was originally created by Thomas Wielfart, but around July 2019 I started to play around with the code and learn Javascript and D3, culminating in the present version.

This section will include the technical description of the workflow, as it pertains to the tool itself and to the processing work made before (the Python module, other Python and R functions), and a sort of manual of how it's used. It will be more or less redundant with the paper I wrote with Kris in December (Montes and Heylen, ming), more or less like vignettes for documentation (as of now, it is still not documented).

Chapter 2

Parameter settings

In this chapter I will describe the various parameter settings we have explored: which are the possible decisions, which ones we have set and which were looked at, why. This should be preceded by an explanation of the workflow itself.

2.1 First steps

Both the targets and the first and second order features are lemma/part-of-speech pairs, such as *haak/verb* (the verb *haken* 'to hook/crochet'), *beslissing/noun* (the noun *beslissing* 'decision'), *in/prep* (the preposition *in* 'in'). The features (or context words) can have any part of speech except for punctuation and have a minimum relative frequency frequency of 1 in 2 million (absolute frequency of 227) after discarding punctuation from the token count in the full QLVLNews corpus. There are 60533 such lemmas in the corpus.

(This threshold is more or less arbitrary, but we're assuming that words with a lower frequency won't have a rich enough vectorial representation.)

In the steps between defining corpus and types and obtaining a the token-level vectors, we have two main kinds of parameters to explore. **First-order parameters** influence which context features will be selected from the immediate environment of the target tokens, while the **second-order parameters** influence the shape of the vectors that represent such first-order features.

In order to visualize the tokens, we have performed dimensionality reduction, i.e. a process by which we try to represent relative distances between items in a low-dimensional space while preserving the distances in high-dimensional space as much as possible. This procedure will be described in [appropriate section].

2.2 First-order selection parameters

We call the immediate context of a token the **first order context**: therefore, first-order parameters are those that influence which elements in the immediate environment of the token will be included in modeling said token. This was made in two stages: one dependent on whether syntactic information was use, and one independent of it.

It goes without saying that the parameter space is virtually unlimited, and decisions had to be made regarding which particular settings would be explored. We tried to keep the parameter settings different enough from each other to have some variation. The decisions were based on a mix of literature (Kiela and Clark, 2014), linguistic intuition and generalizations over the annotation of our very targets. As part of the annotation task, the annotators had to select the items in the immediate context that had helped them select the appropriate tag. In order to remove noise from misunderstandings and idiosyncrasy, we only looked at pairs (or trios) of annotators that had agreed with each other and with our final annotation and ranked the context words over which they had agreed. The distance and dependency information of these context words were used to inform some of our decisions below.

On the first stage, the main distinction is made by `BASE`: between bag-of-words (`BOW`) based and dependency-based models (`LEMMAPATH` and `LEMMAREL`). The former are further split by window size (`FOC-WIN`), part-of-speech filters (`FOC-POS`) and whether sentence boundaries are respected (`BOUND`).

`FOC-WIN` (first order window) A symmetric window of 3, 5 or 10 tokens to each side of the target was used.
Of course, virtually any other value is possible [add references!]. Windows of 5 and 10 are typical in the literature [sources?], while 3 was enough to capture most of the context words tagged as informative by the annotators.

`FOC-POS` (first order part-of-speech) A restriction was placed to only select (common) nouns, adjectives, verbs and adverbs (`lex`) in the surroundings of the token. If no restriction is placed, the value of this parameter is `all`.
Of course, other selections are possible. [add reference] distinguish between `nav`, which only includes common nouns, adjectives, and verbs, and `nav-nap`, which expand the selection to proper nouns, adverbs and prepositions.
A more detailed research on different combinations would be material for further research. As we will see, the `lex` filter is often redundant with the one based on association strength.

`BOUNDARIES` Given information on the limits of sentences (e.g. in corpora annotated for syntactic dependencies), we can exclude context words beyond the sentence of the target (`bound`) or include them (`nobound`).
This parameter seems to be virtually irrelevant. It was thought as a way of leveling the comparison with the dependency-based models, which by

definition don't include context words beyond the sentence, but they don't seem to make a difference.

The distinction between BOW- and dependency-based model doesn't rely so much on which context words are selected but on how tailored the selection is to the specific tokens. For example, a closed-class element like a preposition may be distinctive of particular usage patterns in which a term might occur. However, such a frequent, multifunctional word could easily occur in the immediate raw context of the target without actually being related to it. Unfortunately, just narrowing the window span doesn't solve the problem, since it would also drastically reduce the number of context words available for the token and for any other token in the model. In contrast, we could also have context words that are directly linked to the target but separated by many other words in between, and enlarging the window to include them would imply too much noise for this token and for any other token in the model.

A dependency-based model, instead, will only include context words in a certain syntactic relationship to the target, regardless of the number of words in between. The actual selection process takes two forms in our case: by path length and by relationship. The former, which we call `LEMMAPATH`, is similar to a window size but counts the steps in a dependency path instead of slots in a bag-of-words window. The latter, `LEMMAREL`, matches the dependency paths to specific templates inspired by the context words tagged as informative by the annotators.

To exemplify, let's look at (1) and take *herhalen* 'to repeat' as the target.

(1) *De geschiedenis rond Remmelink herhaalt zich.* 'The history around Remmelink repeats itself.'

`LEMMAPATH` This set of dependency-based models selects the features that enter a syntactic relation with the target with a maximum number of steps.
The possible values we have included are `selection2` and `selection3`, which filter out context words more than two or three steps away, respectively, and `weight`, which gives a larger weight to context words that are closer in the dependency path.
A one-step dependency path is either the head of the target or its direct dependent. Such features are included by both `selection2` and `selection3` and receive a weight of 1 in `weight`. In (1) this includes the subject, *geschiedenis* 'history', and the reflexive pronoun *zich*, which depend directly on it. If the target was *geschiedenis* 'history', *herhalen* 'to repeat', its head, would be selected.
A two-step dependency path is either the head of the head of the target, the dependent of its dependent, or its sibling. Such features are included by both `selection2` and `selection3` and receive a weight of 2/3 in `weight`. In (1) this includes the determiner *de* and the modifier *rond* 'around' directly depending on a *geschiedenis* 'history'.

A three-step dependency path is either the head of the head of the head of the target, the sibiling of the head of its head, the dependent of the dependent of its dependent, or the dependent of a sibling. A typical case of the last path is the subject of a passive construction with a modal, where the target is the verb in participium (*belastingen* 'taxes' in *de belastingen moeten geheven worden* 'the taxes must be levied'). Such features are included in `selection3` but excluded from `selection2` and receive a weight of 1/3 in `weight`. In (1) this corresponds to *Remmelink*, the object of *rond* 'around'.

Features more than 3 steps away from the target are always excluded. While some features four steps away can be interesting, such as passive subjects of a verb with two modals, they are not that frequent and may not be worth the noise included by accepting all features with so many steps between them and the target. To catch those relationships, `LEMMAREL` is a more efficient method. There are no context words more than three steps away from the target in (1).

`LEMMAREL` This set of dependency-based models selects the features that enter in a certain syntactic relation with the target. They are tailored to the part-of-speech of the target, and each group expands on the selection of the group before it. The specific selections are listed in Table 2.1.

groups	nouns
1	modifiers and determiners of the target, items of which the target is modifier or de
2	conjuncts of the target (with or without conjunction), objects of the modifier of the
3	objects and modifiers of items of which the target is subject or modifier, subjects a

2.2.1 PPMI weighting

The `PPMI` parameter is taken outside the set of first-order parameters because it can both filter out first-order features and reshape their vector representations. In truth, the choice of **p**ositive **p**ointwise **m**utual **i**nformation (PPMI) over other weighting mechanisms, as well as setting a threshold or not, is already a parameter setting, which in these circumstances is set to PPMI and a threshold of 0. In all cases, the PPMI was calculated based on a 4-4 window (that could also be a variable parameter).

This parameter can take three values. `selection` and `weight` mean that only the first-order features with a PPMI > 0 with the target type are selected, and the rest discarded, while `no` does not apply the filter. The difference between

`selection` and `weight` is that the former only uses the value to filter the context features, while the latter also weighs their vectors with that value.

2.2.2 Second-order selection

The selection of second-order features influences the shape of the vectors: how the selected first-order features are represented. While the frequency transformation and the window on which such values were computed could be varied, they were set to fixed values, namely PPMI and 4-4 respectively. The parameters that were varied across, although we don't expect drastic differences between the models, are vector length and part-of-speech.

`SOC-POS` (second order part-of-speech) This parameter can take two values: `nav` and `all`. In the former case, a selection of 13771 lect-neutral nouns, adjectives and verbs made by Stefano is taken as the set of possible second-order features. In the latter, all lemmas with frequency above 227 and any part-of-speech are considered.

`LENGTH` Vector length is the number of second-order features and therefore the dimensionality of the matrices on which the distance matrices are based, although the amount is not all that changes. It is applied after filtering by part-of-speech.

We have selected two values: `5000` and `FOC`. The former includes the 5000 most frequent elements of the possible features, while the latter takes the intersection between the possible second-order-features and the first-order-features, regardless of frequency. With `SOC-POS:all`, `FOC` will include all first-order features of that model, while with `SOC-POS:nav`, only those included in Stefano's selection.

The actual number of dimensions resulting from `FOC` depends on the strictness of the first order filter. This information can be found on the plots that, for each staal, show how many first order context words are left after each combination of first order filters.

2.2.2.1 FOC as SOC

What does it mean to use the same first-order context words as second-order context words?

First, depending on the number of target tokens and the strictness of the filter, there could be a different number of context words, ranging in the hundreds or low thousands.

Second, the context words will be compared based on their co-occurrence with each other. The behaviour of a context word outside the context of the target will be largely ignored: of course, the association strength between two items has to do with their co-occurrence across the whole corpus, as well as their

non-co-occurrence, but it will only be included in the second order vector of the first item if the second is also among the first order context words.

2.3 Medoids

The multiple parameters return a huge number of models, and while purely quantitative methods might be able to process and compare them, it is not feasible for a human to look at hundreds of clouds and stay sane enough to make out anything from them. A more efficient –and easier on the human mind– way to approach this is, instead, to look at representative models.

[Describe PAM?]

This method requires us to choose a number of medoids beforehand, which is not an easy task. If we wanted the medoids to represent the best clustering solution, we could run the algorithm with different values of k and compare the results with measures such as silhouette width, as suggested by Levshina (2015). However, that is not necessarily our goal. We want to be able to see as much variation as possible, while keeping the number of different models manageable (i.e. below 9). It is not particularly problematic if these models are redundant, as long as we can ensure that all the phenomena that we are interested in are represented in them.

For example, given a lemma with multiple senses, it might be the case that some models group the tokens of one sense, and others group the tokens of another: we would like to see representatives of both kinds. [add example]

There is no guarantee that the method with the best silhouette returns all the variation we are interested in –our goal is, rather, to limit the number of different models we need to examine from the total number, say 200, to a more manageable amount, like 8. In the same terms, there is also no guarantee that when we identify something interesting in a medoid, i.e. an island for a particular usage pattern, all the models in the cluster of that medoid, and only those models, will share that characteristic. In order to check that, we can look at random samples (again, of 8 or 9 models) of each of the clusters and visually compare them to their medoids. This doesn't need to be as thorough an examination as that of the medoids themselves: it suffices to check if the random sample is not too different and seems to share the characteristic of interest. [add example]

In general terms, for the characteristics identified in the case studies that make up this investigation, we can be quite confident that the medoids are representative of the models in their clusters. However, depending on the concreteness of the phenomena, the variation across models, the clarity of the visualization and the wishful thinking that might lurk in the researchers' minds, it might be the case that something found or assessed in a medoid is not shared by the models in its cluster. The comparison needed with the random sample should be fast

and honest and is strongly recommended: if the medoids are representative, you can see it in an instant; if they are not, it just takes a bit longer to admit it. It is *not* the same as actually studying and comparing 64 different models.

[add an example of something not working, like hoekig medoid 3?]

Chapter 3

Exploring parameter settings

In this chapter I would give a brief review of why some parameters where fixed to certain values and why we explored others along a certain range.

This would include procrustes/euclidean, addition/average, the range for PPMI, the values set because Kiela + Clark…

Chapter 4

NephoVis

In this chapter I will describe the visualization tool itself, as was done in the paper with Kris. It should include a brief introduction mentioning D3.js and Github Pages and describe the current features. I should leave the data format to the end because I expect I will change it.

Part II

The language of clouds

Chapter 5

Nonsense or no senses?

In this part, which will have who knows how many chapters, I will delve into the theroetico-methodological insights –the theoretical impact, if you will– of my analyses.

I will have to describe the annotation procedure (probably) but, more importantly, discuss the main theoretical observations derived from my studies.

The main points, for now, are:

- From cloud interpretation, what we see are not necessarily senses, but rather "common/shared/similar" contexts of usage.
- There is no single optimal solution for every item.
- Because of the second point, what do the different parameter settings do, for specific items? (What kind of info do they pick up?)
    - This is also material for the first part of the project's monograph

Currently (but there is still much research to do) I think that different parameters offer different perspectives, but picking out different aspects of the context. Those perspectives won't be relevant for all lemmas –prepositions, if dependency-informed, are relevant for *hoop* but not for other nouns. Even that relevance is gradual. We might be able to generalize, classifying items depending on which parameter settings are relevant to them, or depending on what they would look like with certain or all kinds of parameters.

Say, *horde* and *heffen* look quite good with just any setting, while *haten* looks awful always. Still, this is based on the current way of computing vectors, which adds the type-level vectors of the tokens instead of averaging over them (which is what I thought they did). And I still have to look at the adjectives and revise nouns and verbs using the medoids based on euclidean distances.

Will these analyses change my current conclusions? Chan chan chaaaannn...

In any case, I will try to look at all of that in the last two weeks of February and then discuss with DG the content for this part .

Chapter 6

The nature of clouds

Clouds don't necessarily show senses but usage patterns –the senses are well distinguished insofar they match those usage patterns. But we could have one usage pattern for multiple senses, multiple usage patterns for one sense, and, more importantly, different degrees of definition of each usage pattern.

It is not a matter of frequency we can have infrequent senses/patterns that are nonetheless strong enough to form a clear group; and
extremely frequent senses can also be grouped separately, they don't necessarily absorb the minor senses/patterns, if their pattern is characteristic enough.

Bibliography

Kiela, D. and Clark, S. (2014). A Systematic Study of Semantic Vector Space Model Parameters. In *Proceedings of the 2nd Workshop on Continuous Vector Space Models and their Compositionality*, pages 21–30, Gothenburg. ACL.

Levshina, N. (2015). *How to do linguistics with R: data exploration and statistical analysis*. John Benjamins Publishing Company, Amsterdam ; Philadelphia.

Montes, M. and Heylen, K. (Forthcoming). Visualizing Distributional Semantics. Mouton De Gruyter.