Detection and Insertion of Implicit Subjects for Approaching Information System Challenges with Natural Language Processing

Lukas Rossi

Abstract

In the English language, the actor of an action is oftentimes only implied. For example, the phrase 'the data is kept up to date' requires additional context in order to determine who is keeping the data up to date. In this project work we introduce a framework for detecting the need for, and inserting these implicit actors back into the original phrase. We report a precision of 54% *and recall of* 83% *for the detection mechanism and* 27% *congruence when applied to a gold standard consisting of process descriptions.*

1 Problem Definition

Natural language often times allows for the omission of parts of a sentence, if the omitted fragment is clear from context. In the English language a commonly omitted fragment is the agent of an action as in the sentence 'The agent is omitted [by the author]'. We refer to this omitted agent as an *implicit subject*. For clarity, it should be noted that an implicit subject is not the same thing as the grammatical subject of a sentence. When referring to a subject within this text, we are referring to implicit subjects unless explicitly stated otherwise. It should also be noted that coreference resolution, i.e., determining if multiple words refer to the same entity, is a distinct problem from this and will not be discussed further here.

Though usually clear for a human reader, these implicit subjects can pose a problem for certain NLP tasks, for example when comparing two or more process descriptions. If both descriptions contain a fragment akin to 'the data shall be kept up to date' a naive comparison will find the descriptions to match. The responsibility of who should 'keep the data up to date' might differ though if for example one document is implicitly referring to the data subject and the other to the handling company.

In the following, we will investigate how implicit subjects can be detected and replaced by explicit ones for the English language following a rule-based approach. Our work focuses on business process descriptions but has wider applicability to general text.

2 Related Work

Though the need for dealing with implicit subjects in the setting of business process compliance has previously been identified in [11], little direct research in this field has been done. We would like to draw attention though to the similarity to problems encountered in the field of RDF-triplet extraction from unstructured text. Here also, implicit entities lead to triplets not being extractable. [3] provides a review of methods used for knowledge graph completion. KG completion though operates more on the level of an extracted graph as opposed to our approach focusing more on extraction from text. We would also like to highlight the differing emphasis between RDF-triplet extraction, which focuses more on precision and can afford lower recall due to usually employing larger corpora and thus restated relations [1], whereas the use case of business process descriptions requires the extraction of all possible relations in a text.

More generally, the extraction of implicit information from text has been the focus of much work. [8] examines ways of extracting relations from compound nouns instead of verbs using rule-based approaches. [10] focuses on extracting implicit relations for numerics using a bootstrapping approach.

As none of this work focuses specifically on the use-cases outlined above, we see a clear research gap worth investigating.

3 Methodology

3.1 Background

After a review of available literature, we manually inspected documents related to process descriptions. In detail, we examined the GDPR [13] as an example of a normative document for other process descriptions, and a set of synthetic process descriptions taken from [4]. From this four principal occurrences of implicit subjects were identified:

1. **The agent of a passive sentences**: The most common form of implicit subject is found when using the passive voice. The English language allows for the omission of the agent of a verb in the passive voice as is done in our initial example: 'The agent is omitted [by the author]'.

2. **Gerunds**: Another common class of implicit subjects are those accompanying gerunds (verbs inflected to end on *-ing*) as is the case in the sentence: '[The author] omitting the subject is common.'

3. **Nouns referring to actions**: Often times nouns referring to actions can also be supplemented by a subject[1] as is the case in the sentence 'The omission of a subject [by the author] is possible'. As the boundaries of when a noun can accept an agent are ill-defined we focus on a subset of these nouns, namely nominalized gerunds, e.g., 'The processing of data [by the controller] must be monitored'. As nominalized gerunds are always deduced from a verb, they can generally be assumed to implicitly reference an agent, if the reference is not explicit. The important task of widening the class of inspected nouns is left to future work.

4. **Imperatives**: Imperative verbs can also be considered as implicitly referring to a subject, namely the reader of the text, i.e., 'you', as can be seen in the sentence '[You] omit the agent'.

We do not claim this list to be exhaustive but find it to be sufficient for describing the inspected data. The implicit subject for each of these types can and must be deduced from the context in which the instance occurs. Each of these classes also differ in how the found subject is inserted into the corresponding sentence as we shall see in subsection 3.6.

3.2 Pipeline

Our goal is to provide an end-to-end pipeline to first detect the absence of an explicit subject where one can be inserted, find the correct subject to insert from user provided context, and finally insert the identified subject into the corresponding sentence, creating again a well-formed sentence.

Based on these observations and goals we have devised a four step process depicted in Fig. 1. First, we identify instances of the types of implicit subjects outlined above. This instance will in the following be referred to as the *target*. In parallel to this, we create a list of possible candidates for insertion from the context. This list is created purely from the context without regard for the targeted implicit subject. In practice the set of candidates consists of all grammatical subjects and objects in the context as any of them might in theory be the searched for implicit subject. This step sacrifices precision for high recall. Only in the next step are the candidates run through an array of filters sensitive to the target. In theory, this filter chain may return any number of candidates. In practice though, it is designed in a way as to always provide exactly one candidate. Else we can conceptually pick at random. We will elaborate on the choice of filters in section 3.5. Lastly, this selected candidate must be inserted into the original sentence.

[1] Not in a grammatical sense but rather in a semantic sense. The explicit replacement of the implicit version is technically an object.

3.3 Candidate Extraction

Candidates are extracted by passing the context through a dependency parser and extracting every word that appears as either the subject or the object in a phrase. Each word is extracted alongside its dependents. We can thus differentiate between for example the word 'flow' appearing as the head of 'the process flow' or appearing as the head of 'a similar flow'. At this stage a very large number of false positives are provided as well as candidates that we do not want to use, for example due to them being a pronoun and thus, providing little additional information without a subsequent pass of a coreference resolution algorithm.

3.4 Implicit Subject Detection

For each of the classes mentioned in 3.1 distinct extraction rules were developed.

3.4.1 Passive sentences

Using a part-of-speech tagger, we can extract verbs in the past participle that have an auxiliary verb as a dependency. We remove verbs that already have an agent as a dependency and those that have a subject in the active voice. Though technically also verbs in the passive voice without subject, we also remove those that serve as an adverbial modifier as they tend to not take an agent in natural speech. Further, there exists a set of passive verbs that in general do not take an agent. For example, the construct 'based on' qualifies as a verb in the passive voice without an agent, but cannot be logically assigned an agent, which we blacklist from extraction.

3.4.2 Gerunds

Similarly, again using a POS tagger, we can directly detect present participles and filter those out that already have a subject as a dependency. We further prune this set by removing gerunds that act as adjectival modifiers or clauses, as their subject are the sentence fragment being modified.

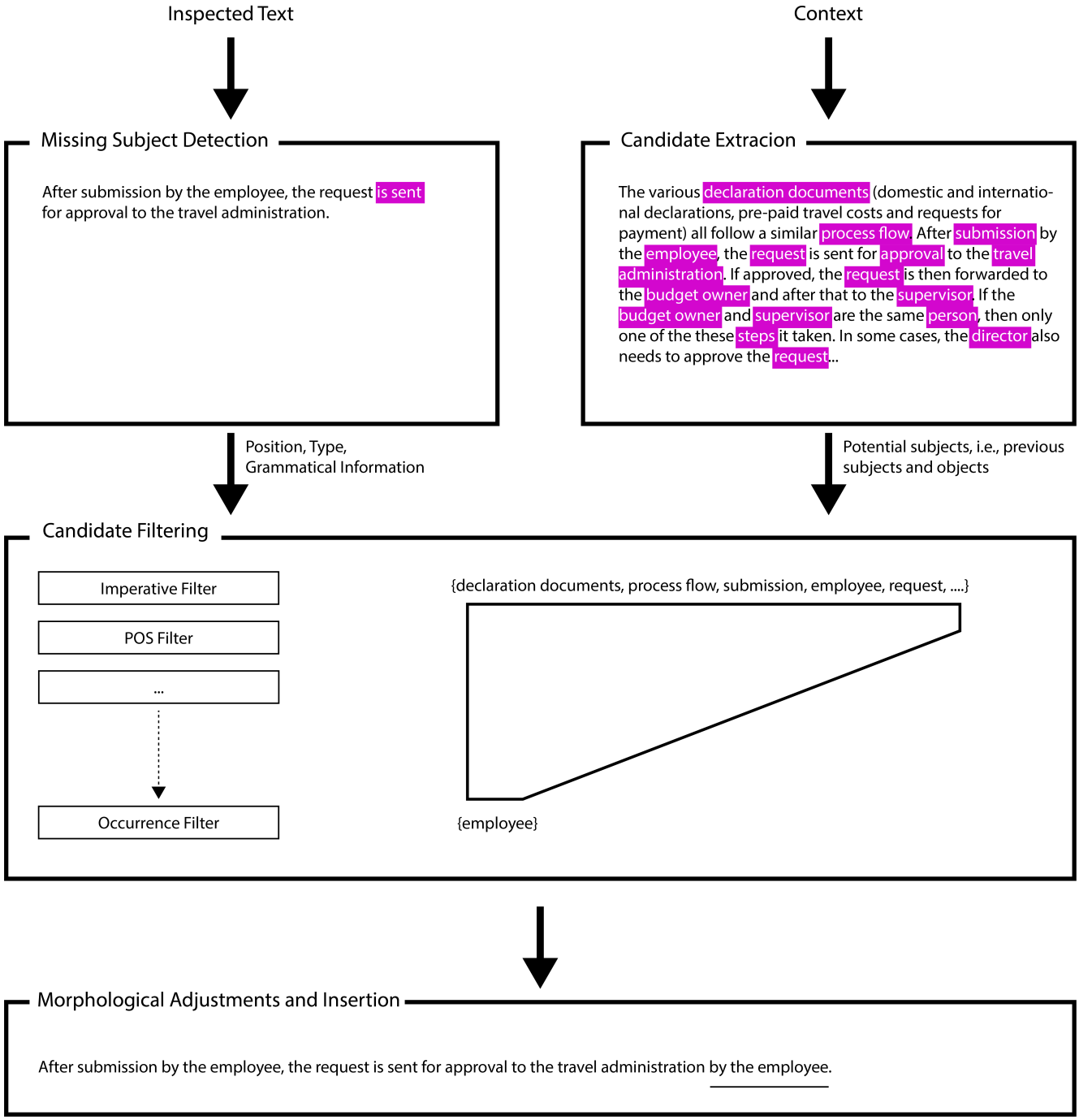

Figure 1. A holistic overview of the pipeline used for implicit subject detection and insertion.

3.4.3 (Gerundive) Nouns

To detect instances of gerundive nominalization, we search for nouns with suffix *-ing*. We then inspect the stem obtained by removing this suffix. If this stem is in turn a verb, we can assume the original noun to be a nominalized gerund and be able to take a subject. Note how we must rely solely on the orthography of the stem. We thus use a predefined list of English verbs taken from [12]. As we will see in the evaluation, this reliance on orthography can cause problems, for example with homographs. Note also that this approach may be generalized by using a general stemmer on the inspected noun instead of only removing the *-ing* suffix. We found this approach to detect too many nouns though.

3.4.4 Imperative

Imperatives are again easily detected by a POS tagger. We simply filter for verbs in their base form and disregard those that have an auxiliary verb accompanying them.

3.5 Candidate Filtering

Conceptually, the candidate filtering can be thought of as a structured ensemble model. Multiple filtering steps are chained together to produce a final output. In the following, we will present some possible filters.

3.5.1 Imperative Filter

We begin with a simple filter to demonstrate the modus operandi of our filters in general, namely the imperative filter. If the target detection is an imperative, no elaborate search must be undertaken as the subject in question is always 'you'. Thus, if an imperative is detected we can filter the list of candidates to only those whose text is 'you', creating a new 'you' candidate if needed. If the inspected target is not an imperative we simply pass on the entire set of candidates to the next filter in the chain. As this filter has virtually perfect recall, it is best placed at the beginning of the chain.

3.5.2 Part-of-Speech Filter

Before taking the context or any semantics into consideration, we further filter the candidates based on their part-of-speech. More concretely, we disregard candidates that are not nouns. This filters out mostly pronouns. We argue that we can do this without major loss of information as the noun referred to by each filtered pronoun should still be present in the context. We do though loss information on both the quantity and the context with which the candidate entity was referred to within the text. An alternative would be to apply some form of coreference resolution to replace instead of simply disregard pronouns. We deem the added complexity to not be worth this marginal information gain.

We make an exception for 'you'-candidates and do not filter them as they are needed for imperatives.

3.5.3 Dependent Of Same Sentence Filter

Another simple filter that can be applied is disregarding candidates that already act as the object of the implicit subject construction, specifically those that are already the grammatical subject of a target predicate in the passive voice. Though technically an entity can be both the grammatical subject and agent of a verb in the passive voice, in practice such a structure is typically made explicit, for example through the use of a reflexive pronoun.

3.5.4 Perplexity Filter

Oftentimes, a candidate can be disregarded simply from it being semantically incompatible with the target. For example, when examining the phrase 'the house is built' we can exclude words such as 'requirements' and 'draft' from consideration as the subject of 'built' as they do not fit semantically, regardless of the given context. We attempt to filter out these words by using the probability of the sentence with inserted candidate being generated by a large language model[2]. More accurately, we use the perplexity the model assigns to the candidate sentence [6]. To compare different candidates, we examine the ratio of the perplexity of the target sentence to that of the sentence with the candidate inserted ρ. We found selecting only candidates with $\rho \leq 1.5$ to offer a good tradeoff between selecting too many candidates and filtering out the correct one. A higher ratio may be picked to filter out fewer candidates and vice versa.

3.5.5 Similarity Filter

Another approach is based on the fact that making the implicit subject explicit should not change the meaning of the sentence. As a consequence, we can rank the candidates by how semantically similar they are to the original sentence. For this we inspect the cosine similarity of the sentence embeddings using embeddings generated by Google's Universal Sentence Encoder [2].

3.5.6 Previously Mentioned Relation Filter

A straight forward approach for determining the correct subject for insertion is to search the context for previous occurrences of each candidate standing in relation to the currently inspected target. For example, if the sentence 'After

[2]We chose GPT-2 [9], as inference can easily be done on consumer hardware.

submission, [something happens]' is preceded by the sentence 'The employee submits [something]', we can assume that the correct candidate for 'submission' is 'by the employee'. More concretely, we can search through the candidates and keep those, whose stem of the lemma of their predicate is equal to that of the implicit subject target's. We further filter these candidates to those that appear either as the subject of a phrase in the active voice or an agent of a phrase in the passive voice.

3.5.7 Occurrence Filter

An indication for which candidate is likely to be the most fitting is the candidate's prevalence in the context. If the candidate occurs often in the context, it is likely to also be relevant for the target implicit subject. From this we develop a primitive tie breaking mechanism employed at the end of our pipeline: We simply count the number of times a candidate's text occurs in the list of candidates and choose the most frequent one. Though not an efficient filter on its own, it provides better performance than picking a candidate at random at the end of the filter chain.

3.6 Insertion

Though providing only the implicit subject detection and the selected candidate is deemed sufficient for most downstream tasks, implementing an insertion mechanism has both the advantage of creating a full end-to-end process as well as allowing us to use filter mechanisms that require the candidate to be inserted into the sentence to assess its relevance. We must preface this by stating that inserting a subject into a sentence may not have a unique solution. The sentences 'You are fined by the police after 30 days' and 'You are fined after 30 days by the police' are effectively equivalent. We will discuss the repercussions of this in the evaluation.

For now, we simply focus on creating syntactically and semantically correct sentences. We may again fall back on our classification of implicit subject types beginning with passive sentences. The goal is to insert the selected candidate. For this we simply extract the noun chunk of which the candidate is a part, prefix it with 'by' and insert it immediately after the verb. No inflection of the verb is necessary. In certain situations, we cannot insert the candidate immediately after the verb but rather after adpositional constructs surrounding the verb. For example, we the phrase 'the data is kept by the controller up to date' though understandable is not as well formed as 'the data is kept up to date by the controller'. The same insertion mechanism can be used for implicit subjects based on nouns without any adjustments.

Inserting 'you' into imperative sentences can be done by inserting the pronoun in front of verb[3]. The verb must be inflected to match the second person singular[4]. Again, situations may arise where 'you' cannot be inserted immediately preceding the verb. In the case of imperatives this may happen when the verb appears after an adverbial modifier. If this adverbial modifier is not part of an adverbial clause, we insert the 'you' in front of this adverbial modifier instead. We thus obtain '[You] *always* select the cheapest parts' but '*After checking the price*, [you] select the cheapest parts'.

Gerunds can be handled in a very similar way, the only difference being the verb needing to be inflected according to the number and person of the selected candidate and the candidate being inserted instead of 'you'. We always choose the present tense for inflection as it is the most fitting for our domain of process descriptions. Further work is needed to generalize this approach.

4. Evaluation and Discussion

To evaluate our approach a basic prototype was implemented[5]. We use Python 3.10 [14] with heavy reliance on the NLP library spaCy [5] for dependency parsing and POS-tagging. Verb inflection is based on the pattern library [12] The quantitative evaluation was done using a hand-crafter gold standard (73 sentences), based on a mix of synthetic process descriptions [4] and the GDPR [13]. The inspected text is always a single sentence from each corpus. The context for the process descriptions is always the entire process description and for the GDPR the entire article from which the inspected sentence is taken.

4.1 Quantitative Evaluation

Our best results were achieved using only a subset of the proposed filters in the following order: imperative filter, POS filter, dependent of same sentence filter, perplexity filter and occurrence filter.

We conduct two quantitative evaluations targeting first only the implicit subject detection an then the entire end-to-end pipeline. Our implicit subject detection mechanism achieves a precision of 54% at a recall of 83% resulting in an F1-score of 66%. For the end-to-end evaluation we compare each sentence from the gold standard to the generated sentence by checking if both sentences contain the same tokens with the same dependencies. We match the gold standard in 27% of cases. As we will see in the next subsection,

[3] One may argue that additionally inserting a auxiliary verb, e.g., 'should', would capture the semantics of the imperative. We decide against this though to be more inline with our evaluation data.

[4] The second person singular is in most cases orthographically equivalent to the imperative. An example exception is the highly irregular verb 'to be'.

[5] `https://github.com/l-rossi/is-with-nlp-implicit-subject`

we believe these values to not truly capture the quality of the pipeline.

4.2 Qualitative Evaluation

A qualitative analysis of the pipeline's outputs will provide a better indication of its strengths and weaknesses. As little can be said to the candidate extraction step, we group the discussion thereof with that of the candidate filter step.

4.2.1 Implicit Subject Detection

As seen in the quantitative analysis, our detection mechanism has a tendency of detecting more implicit subjects than the gold standard. A common problem was the detection of verbs in the passive voice that cannot logically take an agent, often referred to as 'impersonal passive'. In the constructs 'provided that', referred to in' and 'based on' the verb provided cannot take an agent in any meaningful way. An ad hoc solution would be to blacklist verbs that we know can only appear in the impersonal passive. There are two major problems with this though. Firstly, maintaining such a list would be a substantial undertaking and out of the scope of this project. Secondly, there exist verbs that depending on context can either require the impersonal passive or be able to take an explicit subject. We thus found no effective way of detecting these constructs and leave such a task to future work. A similar thing can happen with certain instances of gerund detections. For example, the phrase 'taking something into account' is a general statement which again cannot take a subject.

In rare cases, the dependency parser employed by us makes minor mistakes. For example, in the sentence 'The information shall be provided by electronic means' the propositional object 'by electronic' means might be identified as an agent instead. In general, the dependency parser is very reliable though and we see little room for improvement here.

Not all of these detections are strictly erroneous though. Oftentimes it is not clear if a construct should be amended by a subject or not. An example for this is the sentence 'If approved, the request is then forwarded to the budget owner and after that to the supervisor.' It is not clear if the second verb 'forwarded' should take a subject and if so, which it should be. Our method detects this as a passive verb in need of a subject, the gold standard though does not insert a subject here.

A seldomly occurring problem that though highlights some of the limitations of our approach is that of nouns being detected as nominalized gerunds due to them being homographs of nominalized gerunds. In our case, the word 'evening' has two meanings, the time of day and the nominalized gerund of the verb 'to even'. As our nominalized gerund detection approach only considers orthography, a nonsensical subject is added to the fragment 'in the evening'.

Though false negatives are not considered a major problem of our approach as can be seen from the comparatively high recall, some interesting observations can still be made. As mentioned in 3.1 we decided to focus on a subset of nouns referring to actions, namely gerundive nouns. This ignores other nouns referring to actions such as 'the withdrawal', 'the renewal', 'the omission', etc. As previously mentioned in 3.4.3 we can generalize our approach to try and include these nouns as well. This though leads to an unfavorable tradeoff between precision and recall, falling and rising to 23% and 88% respectively.

4.2.2 Candidate Selection

Candidate selection is by far the most challenging part of the pipeline. The most reliable candidate selection mechanism is that for imperatives, functioning correctly in virtually every case. The filter chain can be holistically viewed as first removing obviously false candidates (Imperative Filter, Part-of-Speech Filter, Dependent of Same Sentence Filter) then trying to find a semantically viable candidate (Perplexity Filter) and finally narrowing down the selection to a single candidate by any means possible (Occurrence Filter). Consequently, the first group of filters tend to cause little problem. An overview of the distribution of where candidates are filtered out in the chain can be found in Fig. 2. We observe how the first four filters have perfect recall, but consequently much lower precision. This trend reverses the further we go through the chain.

A major drawback of the 'perplexity filter' is its disregard for the wider context of each inspected target. Filtering is done purely dependent on the probability of a sequence of tokens being generated by the LLM, which leads to a blatant bias towards rare words. This becomes apparent with proper nouns which are often disregarded too eagerly. For example, 'Blizzard' in the sentence '[It] starts with *Blizzard* checking whether you have [an] account' receives $\rho > 1.7$ although it is the correct candidate.

Though the 'occurrence filter' delivers decent results for such a simple selection mechanism, its lack of context awareness can lead to naive choices. This problem is exacerbated by large context sizes as is the case for the articles in the GDPR. A possible solution would be to use a discounted count of occurrences based on the distance to the target. One could thereby disregard frequent occurrences of a candidate which are deemed to be too distant from the currently inspected text. The details of this are once again left to future work.

Lastly, we would like to discuss the 'semantic similarity filter' and the 'previously mentioned relation filter', which

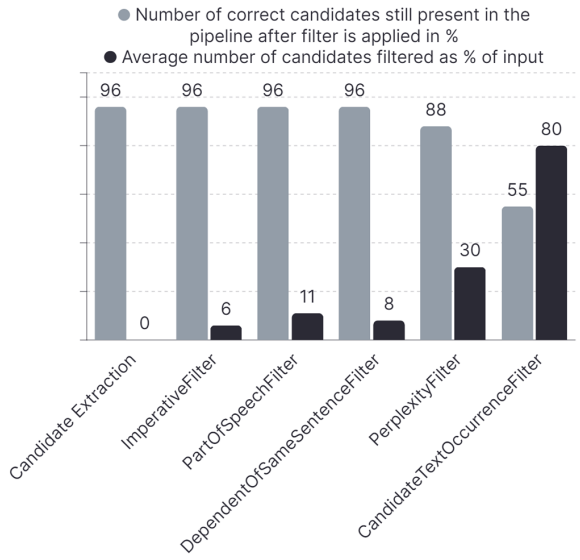

Figure 2. Am overview of where correct candidates are lost within the filter chain and how many candidates are filtered where in the chain. Only true positives from the detection step are taken into consideration.

were not included in our final pipeline. The 'semantic similarity filter' provided subpar results both when including the context in the similarity calculations and when excluding it. We believe this to be due to the difficulty of calculating meaningful similarity metrics between large documents and the relative minor effect of inserting a single subject into these documents. We do believe though, that this avenue of research is still promising and leave it to future work to find a way of implementing similarity metrics. The 'previously mentioned relation filter' works well on toy examples but provided no added value to the pipeline. The few occurrences in which it correctly identified a candidate were more than offset by it selecting candidates too eagerly, most often due to the candidate being a dependency of a very common verb, for example 'need'.

4.2.3 Insertion

The insertion mechanism is deemed to work close to flawlessly. We only see one areas of minor improvement, handling verb tense. Luckily for us, in the English language, the present tense can be used in close to all cases. Nonetheless, close to all the mismatches caused by the insertion mechanism between the gold standard and the results from our approach are caused by a mismatch in tense. Take for example the phrase 'when craving'. Our methods turn this into 'when you crave' whereas a technically better match would be 'when you are craving'. This though is a minor mistake which would require a lot of work to fix and we thus see little room for improvement here.

5 Outlook

With this project we have provided a framework for solving the problem of detecting and inserting implicit subjects into text using a rule-based approach. We have provided sample implementations of each step and provided a prototypical, programmatic implementation to perform an evaluation of our methods. We do believe that there are still some promising areas of potential improvements to this approach, namely:

1. **Adding new classes of implicit subjects**: As previously discussed, the four classes introduced in this paper are sufficient for describing a majority of implicit subject instances. Nonetheless, extending this set of classes may be necessary depending on the corpus to which these methods are applied.

2. **Widening the class of detected nouns**: We have already demonstrated both the need for and an exemplary but insufficient solution to the problem of detecting all nouns that can take an implicit subject. This problem is thus still open.

3. **Improving the occurrence filter**: As mentioned in the evaluation section, although simply counting the occurrences of each candidate provides a good heuristic for the importance of a candidate to the text at large,

a more sophisticated approach is deemed to offer an avenue of improvement. This could be done either by finding a way of discounting each occurrence of a candidate using some metric or instead using more general text summarization tools.

4. **Improving the similarity filter**: Although we did not manage to make the similarity filter perform to a satisfactory degree, we still believe the general idea to hold merit.

5. **Impersonal passive voice**: As already discussed, we were not able to detect the impersonal passive voice adequately. Consequently, the precision of our pipeline suffers.

6. **Incorporating generative AI**: Recent developments have shown the great proficiency shown by LLMs when solving NLP tasks. We believe that the capabilities of LLMs to work with semantics would complement the more rigid rule-based approach. As a proof-of-concept, our implementation contains the possibility of integrating generative AI [7] in the filter chain (ChatGPTFilter). We do this by using the insertion step of our pipeline to generate a sentence for each candidate that has not been previously filtered out. We then prompt the AI to tell us which of these sentences is the most fitting given the user provided context. Though this filter does not generally outperform the filters presented in this paper, it shows how one may make use of generative AI in a structured manner. Further research into how these two approaches can be combined is thus deemed promising.

By combining these improvements with our current approach, we believe we can provide a significant contribution to natural language understanding, specifically in the context of business processes.

References

[1] M. Banko, M. J. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, page 2670–2676, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

[2] D. Cer, Y. Yang, S. yi Kong, N. Hua, N. L. U. Limtiaco, R. S. John, N. Constant, M. Guajardo-Céspedes, S. Yuan, C. Tar, Y. hsuan Sung, B. Strope, and R. Kurzweil. Universal sentence encoder. In *In submission to: EMNLP demonstration*, Brussels, Belgium, 2018. In submission.

[3] Z. Chen, Y. Wang, B. Zhao, J. Cheng, X. Zhao, and Z. Duan. Knowledge graph completion: A review. *IEEE Access*, 8:192435–192456, 2020.

[4] Eduardo Breitembach. Repo for nlp bpm datasets. `https://github.com/EduardoBre/nlp-bpm-data`, 2024.

[5] M. Honnibal, I. Montani, S. Van Landeghem, and A. Boyd. spaCy: Industrial-strength Natural Language Processing in Python. 2020.

[6] D. Jurafsky and J. H. Martin. *Speech and Language Processing*. 2023.

[7] OpenAI. Gpt-3.5 turbo. `https://www.openai.com/`, 2022. Accessed: January 22, 2024.

[8] H. Pal and Mausam. Demonyms and compound relational nouns in nominal open IE. In J. Pujara, T. Rocktaschel, D. Chen, and S. Singh, editors, *Proceedings of the 5th Workshop on Automated Knowledge Base Construction*, pages 35–39, San Diego, CA, June 2016. Association for Computational Linguistics.

[9] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners. 2019.

[10] S. Saha, H. Pal, and Mausam. Bootstrapping for numerical open IE. In R. Barzilay and M.-Y. Kan, editors, *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 317–323, Vancouver, Canada, July 2017. Association for Computational Linguistics.

[11] C. Sai, K. Winter, E. Fernanda, and S. Rinderle-Ma. Detecting deviations between external and internal regulatory requirements for improved process compliance assessment. In M. Indulska, I. Reinhartz-Berger, C. Cetina, and O. Pastor, editors, *Advanced Information Systems Engineering*, pages 401–416, Cham, 2023. Springer Nature Switzerland.

[12] T. D. Smedt and W. Daelemans. Pattern for python. *Journal of Machine Learning Research*, 13(66):2063–2067, 2012.

[13] The European Parliament and the Council of the European Union. Regulation (eu) 2016/679 of the european parliament and of the council of 27 april 2016. *Official Journal L119*, page 1, 2016.

[14] G. Van Rossum and F. L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009.