Rule-based geographical relationship extraction in Dutch news articles

Antoon W. Meijer
11016914

Bachelor thesis
Credits: 18 EC

Bachelor Opleiding Kunstmatige Intelligentie

University of Amsterdam
Faculty of Science
Science Park 904
1098 XH Amsterdam

Supervisor
dr. M.J. Marx

Information and Language Processing Systems
Faculty of Science
University of Amsterdam
Science Park 904
1098 XH Amsterdam

February 7th, 2020

Abstract

Present-day geocoding models lack a method to accurately geolocate complex locations. Complex locations are locations that consist of multiple toponyms at the same geographical scope, making it difficult to locate for geographical search engines. This research aims to determine how accurately complex locations can be geolocated by understanding the relationship between toponyms through surrounding spatial adpositions. An analysis was conducted on a complex-location subset of Dutch traffic accident news articles to obtain the most common spatial identifier words. These spatial adpositions were then grouped by meaning, converted into predicates, and processed in predicate-specific ways. Each predicate-toponym pair results in a set of coordinates, that is then combined with all coordinate sets in an article to form the final article-wide location results. When run on an annotated complex-location testset, the model showed an accuracy of 26%, a median distance of 0.36 km, and an average distance of 5.95 km to the desired location. It can be concluded that the model sometimes shows promising results, but the variety in natural language makes it hard to perform consistently well. Further research is needed to make the model more generally applicable and enable it to cope better with more varying sentence structures.

Contents

1 Introduction

As the amount of published articles and other Dutch text on the Internet continues to increase, it would be helpful for researchers and others interested in location-specific articles to be able to search for articles based on locations mentioned in the article. It would enable them to spend less time sifting through irrelevant articles, and to spend more time doing their own research. A geolocation model could also be beneficial for other purposes, such as automatic traffic accident location statistics.

Olieman et al. (2015) have attempted to accommodate this need by creating an application, LocLinkVis, that plots Dutch input text on a map based on geographical entities in the text. By doing so, users can more quickly see if an article is relevant. The approach taken by Olieman et al. (2015) identifies toponyms in text by using a gazetteer populated with OpenStreetMap data. After toponym disambiguation, the most likely candidate is shown on a map. However, LocLinkVis does not take into account that some locations might consist of more than one toponym. In such a case, the application simply shows the location of all involved toponyms, instead of understanding how these toponyms are linked and need to be interpreted as one location.

1.1 Research question

This research is aimed at improving the accuracy of geolocating complex locations mentioned in Dutch news articles, through the use of toponyms and surrounding terms containing spatial information. As explained in Gritta et al. (2018), toponyms are most easily described as place names. The following excerpt is a definition set by the United Nations: "*A geographical name may also be referred to as a topographical name or toponym*", (UN Department of Technical Cooperation for Development et al., 1992). The surrounding terms containing spatial references about toponyms are mostly prepositions of place, but can also be other adjectives or nouns. In this research, this group of terms will be referred to as *spatial adpositions*.

Accurately geolocating complex locations is especially important, as non-complex locations are already handled with sufficient accuracy by commonly-used geographical search engines such as OpenStreetMap and Google Maps. In this research, complex locations are defined as locations that involve multiple toponyms to accurately describe. Toponyms can refer to various geographical levels, such as streets, cities or regions. What makes a location complex, is the fact that there is usually one geographical level to which more than one toponym refers, thus making it difficult for a model. To understand how these toponyms relate to each other, the previously defined spatial adpositions have to be analyzed. After analysis, it should become clear how all toponym coordinates can be combined to obtain the complex location. Below is an example of a complex location:

"De Burgemeester Bechtweg is tussen de Nederlandweg en de aansluiting met de A65 in beide richtingen afgesloten."

In the example above, the location is not the entire length of the road, but only between two other toponyms. Such locations are usually not accurately geolocated by the common geographical search engines. This research thus aims to answer the following question:

Using spatial adpositions to model the relationship between toponyms, how accurately can complex locations be geolocated?

1.2 Overview

After the introduction, the general background and common steps involved in geolocating natural language are set out in section 2. Section 3 starts with relevant statistics about the used dataset, and details the inner workings of the proposed model in this research structured by the steps outlined in section 2. In section 4, the evaluation method is explained, and the evaluation results are shown. Interpretation of the results is conducted in section 5, and the research is concluded in section 6.

2 Background

The process of geolocating text can be divided into several steps. These steps can all be accomplished in several ways. The first step, geoparsing, is to correctly detect toponyms. Subsequently, the detected toponyms are geographically disambiguated and placed on a map. This process is also referred to as reference resolution (Monteiro et al., 2016). The final step is to apply the gathered geographical data to a model in order to select, combine or otherwise manipulate it. This step is called spatial inference (Leidner and Lieberman, 2011), reference grounding (Monteiro et al., 2016) or geocoding. A common evaluation method is detailed in the final subsection.

2.1 Geoparsing

Geoparsing is the process of detecting toponyms in text. This is considered to be a form of named entity recognition, where only geographical entities are deemed important. Geoparsing deals with two types of ambiguity: semantic and structural ambiguity. Semantic ambiguity refers to the different meanings that one entity can hold. Take for example "Balk". This is a Dutch noun signifying "beam", but it is also the name of a Frisian village, and might very well be someone's last name. Structural ambiguity occurs when it is unclear where a multi-word entity begins or stops. Consider the following example:

"Op de Baron van der Aaweg in Bokhoven bij ..."

The full road name here is "Baron van der Aaweg", but due to two non-capitalized words occurring in the middle of the toponym, the average named entity recognition model would struggle here.

All geoparsing methods fall into three categories: gazetteer-based, rule-based, or machine learning-based (Leidner and Lieberman, 2011). A gazetteer is a database filled with place names and other toponyms, and string matches all words in the text to detect toponyms. LocLinkVis (Olieman et al., 2015) is an example of a gazetteer-based geoparser, utilizing a database filled with OpenStreetMap data. Rule-based geoparsers recognize toponyms based on predefined rules looking at word shape features, by using filters such as first-letter capitalization. The popularity of rule-based geoparsers has declined due to the advent of supervised machine learning approaches, where rules are automatically created by processing annotated data examples.

2.2 Geographical disambigation

After toponym detection, toponyms are geographically disambiguated and the correct coordinates are retrieved. This disambiguation process is also called reference resolution (Monteiro et al., 2016). The ambiguity problem lies in the fact that many geographical locations share the same name, thus algorithms have to choose between several options. As explained by Buscaldi (2011), all approaches to toponym disambiguation can be divided into three main categories:

- map-based: the disambiguation process is mainly accomplished using coordinate processing.
- knowledge-based: disambiguation is achieved by static or dynamic external knowledge bodies such as gazetteers or ontologies.
- data-driven: disambiguation is performed through machine learning-trained models.

Woodruff and Plaunt (1994) uses a map-based approach, where all toponyms are plotted as polygons on a 3D-map. All polygons have the same height. If two polygons overlap, the height naturally increases. Using this method, the highest place on the 3D-map is the result location. An example of a knowledge-based solution is Rauch et al. (2003). Their approach uses context, textual proximity to other toponyms and popularity to geographically disambiguate toponyms. Adelfio and Samet (2013), for example, use a machine learning approach involving Bayesian classifiers for the disambiguation process.

Although supervised machine-learning solutions are gaining popularity, these approaches require an annotated dataset in the target language. The dataset used in this research is Dutch, and no suitable annotated dataset exists in Dutch. Knowledge-based and map-based solutions thus present a more achievable approach.

2.3 Geocoding

The geocoding step is where the model tries to make sense of the retrieved geographical locations. The way in which the model structures the data is very much dependent on the ultimate goal of the research (Monteiro et al., 2016). Some researchers try to find the region in which all toponyms lie to determine the geographical scope of a document, and thus return only one location (Ding et al., 2000). Others use data structures to gain insight into how multiple result locations relate to each other (Calazans Campelo and de Souza Baptista, 2008), and some just return all locations in random order (Goh et al., 2005), effectively skipping this step altogether.

Also of interest is a related research field commonly referred to as qualitative spatial reasoning. The goal of this field is to correctly formalize vague spatial relationships in natural language (Cohn and Hazarika, 2001). For example, Hall and Jones (2008) details an experiment where cardinal relationships (e.g. "north of", "east of") are formalized by studying geotagged images with a caption that includes such a cardinal relationship and a toponym. Other researchers do not study specific spatial relationship constructs, but instead deal with broader concepts such as how to formalize "nearness" (Worboys, 2001).

2.4 Evaluation

Evaluation in the geolocation field has not yet been standardized, due to the variety in source data and focus of researchers. Also, there has not been a sufficient standard evaluation dataset proposed because of the efforts involved in creating one (Leidner and Lieberman, 2011). Such a dataset would need to be manually annotated, and it would need to consist of documents that are all freely available. In order to still be able to compare research results, most researchers evaluate their approach on a self-annotated testset by calculating the distance between their model's answer and the annotated answer. The distances of all these answers are put together and the median and average values of this set are calculated. These results can then be compared. Such a comparison has been carried out by Melo and Martins (2017).

3 Method

This section details dataset characteristics and outlines the model constructed for the research. The section division is based on the steps explained in section 2.

3.1 Dataset

Even though the scope of the research is wider, the dataset that was used solely consists of traffic accident news articles. This choice was made because of the fact that traffic accident articles almost always contain toponyms to describe where the accident took place, with a relatively high degree of complex locations as well, thus making it interesting for this research.

The dataset used originates from the website *flitsservice.nl* and has been edited by Hendriks (2019) to separate each article's body from its title. The titles of the articles will not be used in this research, because these never contain complex locations due to their short length, and title toponyms are usually repeated in the article. The dataset contains 7784 articles. For testing purposes, the dataset has been split in two parts. The first 4998 articles are used as training set, the last 2786 are used as testing set. The publishing dates of the articles vary between 2003 and 2019. Below, two figures show the distributions of general toponyms and highways specifically for both the trainingset and the testset. For figure 2, highways also include secondary roads (N-roads). From these distributions, the conclusion can be drawn that both datasets contain about the same distributions of general toponyms and highways.

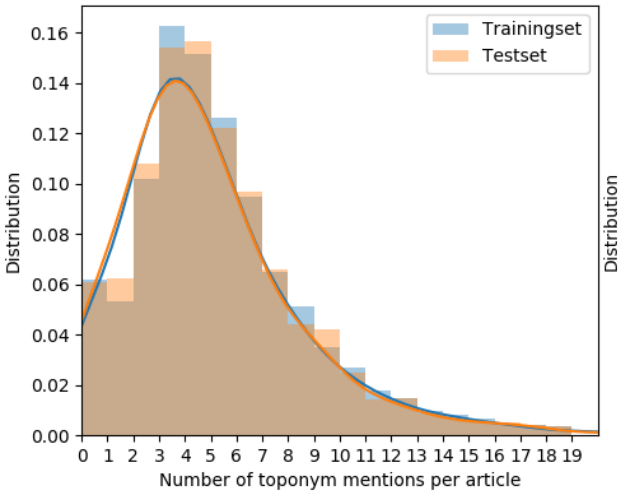

Figure 1: General toponym distribution.

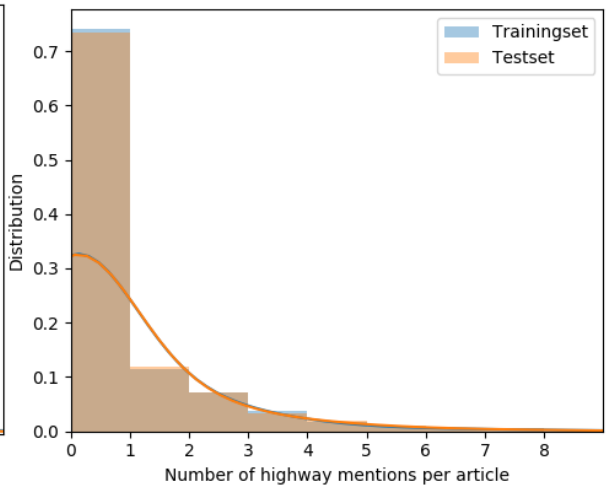

Figure 2: Highway distribution.

3.2 Geoparsing

3.2.1 Toponym detection

The first step to geolocating an article is to find the toponyms in the text. This was done using the model constructed by Overdijk (2020), which is based on LocLinkVis (Olieman et al., 2015). Using the named entity recognition features of spaCy was also considered, but because of superior accuracy results reported by Overdijk (2020), his model was chosen.

The model first tokenizes the input text using spaCy, a natural language processing toolkit available for Dutch language processing. Then, it detects municipalities mentioned in the tokenized text and searches for streets that are within the scope of the detected municipalities. There are some drawbacks to this method, such as when there is no municipality detected, other toponyms will also not be detected. This poses a problem for articles that contain locations from outside of the Netherlands. Also, the model does not recognize landmarks and other high-detail geographical features. However, the model does detect highways and secondary roads (N-roads) that are not tied to a specific municipality.

The model also classifies the toponyms into several categories, but these labels are not used in this research. When a toponym is detected, "LOC" is added to the toponym as a prefix. Below is an example sentence before and after processing by the model.

Before: *"Een 28-jarige man uit de gemeente Rijnwaarden is vrijdagochtend om het leven gekomen bij een scooterongeval op de Doesburgseweg tussen Zevenaar en Giesbeek."*

After: *"Een 28-jarige man uit de gemeente LOCRijnwaarden is vrijdagochtend om het leven gekomen bij een scooterongeval op de LOCDoesburgseweg tussen LOCZevenaar en LOCGiesbeek ."*

3.2.2 Toponym spans

After toponym detection, all sentences that contain no toponyms are deleted. In the sentences that do contain a toponym, all words further than four words away from the closest toponym are also deleted. This is called the *window size*. In section 4, different window sizes are tested to obtain the most optimal value. A toponym with its surrounding words will from now on be referred to as a *span*. In a span, there can be more than one toponym if there was more than one toponym in its source sentence. Below is the example sentence used in section 3.2.1, but now after span selection. This example contains two spans, separated by a comma.

"man uit de gemeente LOCRijnwaarden is vrijdagochtend om het, een scooterongeval op de LOCDoesburgseweg tussen LOCZevenaar en LOCGiesbeek ."

3.3 Geographical disambiguation & geocoding

3.3.1 Spatial identifier selection

When dealing with complex locations, the key to accurate geolocation lies in understanding the relation between the toponyms specifying the location. This relation is expressed in spatial adpositions surrounding the toponyms. To obtain a better understanding of what words are important for spatial identification, a part of the training set was manually classified as either containing complex locations or non-complex locations. Subsequently, an analysis was conducted to visualize the difference in word distributions between all articles and complex-marked articles.

The complex/non-complex annotated dataset contains the toponym spans of the first two hundred articles of the training set. Eighty-two articles are marked as containing complex locations, the rest are marked non-complex. To get a sense of how accurately this subset of two hundred articles represents the entire training set, figure 3 was plotted. The figure shows that both toponym distributions correspond quite well, with the two hundred-article subset only having a bit more articles with average numbers of toponyms, and relatively fewer articles with a lot of toponyms.

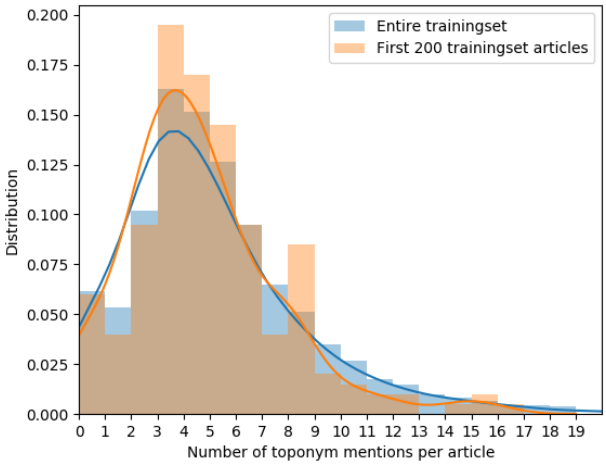

Figure 3: General toponym distribution in first 200 trainingset articles and entire trainingset.

Figure 4 shows the distribution of the most common thirty-five words in the complex-article subset, and the distribution of that word in the first two hundred articles. Because spatial adpositions are never behind the toponym they are referring to, the words located after the last toponym in a span are not counted in this analysis. Toponyms themselves are also removed.

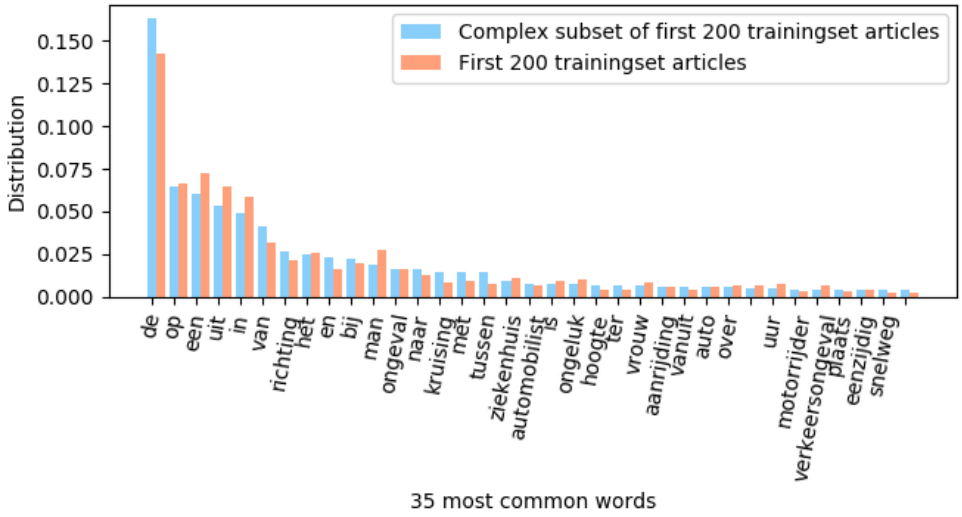

Figure 4: Frequency of thirty-five most common words in complex location articles, together with frequency of said word in all of first two hundred articles.

Drawing from Figure 4, word distributions are mostly the same between complex-location articles and average articles, with some minor exceptions. When only looking at words that contain geographical information, there are some words with larger frequency differences, such as “kruising”,“tussen” and “hoogte”. Even though these words do not appear in the training set very often, they appear to be important to complex locations. Of course, there are also spatial adpositions occurring with similar frequency in both categories that are still very important: “op”, “uit”, “in”, “richting”, “bij” and “naar”.

3.3.2 Converting spatial identifiers into predicates

Now that the most common spatial adpositions are selected, they can be converted into specific rules to make geocoding possible. In the table below, the resulting predicates can be found on the left-hand side, with the selected input words on the right-hand side. The value behind the predicate represents its arity. Some input words, such as "kruispunt" and "te", are not among the most common thirty-five words, but can easily be added to an existing predicate since they have the same meaning. The word "hoogte" might not sound like a spatial adposition, but it occurs near toponyms frequently in the form "ter hoogte van", in which case it is a spatial adposition. Sometimes, there occurs more than one of the selected prepositions within the window size before the toponym. In this case, either "INTERSECT(1/2)" is selected if it occurs, or if it does not, the preposition closest to the toponym is selected.

Spatial adposition	**Predicate**
op over	ON(1)
in te	IN(1)
bij nabij hoogte	AT(1)
tussen	BETWEEN(2)
kruising kruispunt splitsing T-splitsing	INTERSECT(1/2)

Table: Spatial adposition to predicate conversion table

Any toponyms with no spatial adposition, or with a spatial adposition that is not in the table above, will not be processed. In order to convert spatial adpositions to the predicates listed above, the algorithm below is used as a first step. This algorithm condenses spans to adposition-toponym pairs. The innermost loop of "extractADPLOC"[1] is described, where spans are processed one by one.

Algorithm 1 extractADPLOC

for word in reversed span **do**
  if word is toponym **then**
    if already a toponym in adposition-toponym pair **then**
      1. Save the full adposition-toponym pair to result list.
      2. Create new adposition-toponym pair and add toponym to this pair.
    else
      1. Add toponym to adposition-toponym pair.
    end if
  else if word is spatial adposition **then**
    if already a toponym in adposition-toponym pair **then**
      1. Add spatial adposition to adposition-toponym pair.
    end if
  end if
  1. Add adposition-toponym pair to result list.
end for

[1] `https://github.com/toonmeijer/geocoding-nlp/blob/master/implementation/spatial_identifier_extraction.py`

Below is the same example sentence used in section 3.2.1 and 3.2.2, but now further condensed. Note that the toponym "Rijnwaarden" is not processed by the algorithm, because it has a spatial adposition, "uit", that is not supported.

Before: *"Een 28-jarige man uit de gemeente LOCRijnwaarden is vrijdagochtend om het leven gekomen bij een scooterongeval op de LOCDoesburgseweg tussen LOCZevenaar en LOCGiesbeek ."*

After: *[[['op'], ['Doesburgseweg']], [['tussen'], ['Zevenaar', 'Giesbeek']]]*

Unlike in the example above, there can be multiple adpositions in one adposition-toponym pair. The function "ADPLOCtoPredicate"[1] further handles the conversion from adpositions to predicates by selecting the best predicate from the list, if there is more than one. If there are no adpositions, the toponym is removed. The best predicate is the closest one to the toponym, i.e. the last one in the list, unless a spatial adposition covered by "INTERSECT" is in the list. Then that adposition has priority and is selected. Below is the same example mentioned above, but the spatial adpositions are now converted to predicates.

ON('Doesburgseweg'), BETWEEN('Zevenaar', 'Giesbeek')

3.3.3 Converting toponym-predicate pairs into coordinates

After converting spatial adpositions to predicates, each toponym is converted to coordinates in a way that depends on the predicate. This step gives two types of results: a list of coordinate points from road toponyms, processed by "ON(1)" or "INTERSECT(1/2)", or a simple two- or four-coordinate bounding box from other toponyms, processed by the other predicates. Road toponym results are added to the *road list*, and bounding box toponym results are added to the *bounding box list*. Algorithm 2 essentially converts a dataset of toponym-predicate pairs into a structured list of coordinates per article. How each predicate is converted to coordinates is explained in the following subsections. The two merging steps are detailed in algorithm 3. The reason for performing both sentence-based and article-based merging instead of only article-based merging, is that toponyms in the same sentence are more likely to be related than toponyms in the article, thus preventing wrong matches. All algorithms can be found on GitHub[2].

Algorithm 2 findLocations

for article in dataset **do**
  for sentence in article **do**
    for toponym-predicate pair in sentence **do**
      1. Fetch coordinates for toponym based on predicate.
      2. Add coordinates and predicate type to sentence bounding box list or sentence road list.
    end for
    1. Merge sentence bounding box list and sentence road list using algorithm 3.
    2. Append both lists to their respective article-wide list.
  end for
  1. Merge article bounding box list and article road list using algorithm 3.
  2. Return both article lists.
end for

3.3.3.1 Coordinate source & search method

To retrieve the coordinates associated with a toponym, data is extracted from OpenStreetMap. This is done using Nominatim and Overpass API. Nominatim is used to search for cities and villages with a clear center and surrounding bounding box, whereas the Overpass API is used to search for streets. Nominatim is the preferred search method due to its speed and ease-of-use, but it shows poor recall of highway segments and other long road segments. Both search methods are programmed to primarily search for coordinates in the Netherlands, with Overpass excluding any results abroad, and Nominatim preferring results in the Netherlands. For both methods, functions were written called OverpassSearch[2], OverpassQuery[2] and NominatimSearch[2].

[2] https://github.com/toonmeijer/geocoding-nlp/blob/master/implementation/geocoding.py

3.3.3.2 ON(1)

Through observation, the toponym following the "ON" predicate has been determined to mostly be a street of some kind. Therefore, the toponym coordinates are searched for through Overpass. No bounding box is given other than the Netherlands, so search results usually include multiple streets throughout the country if the street name happens to be a commonly used one. The toponym and result coordinates are added to the road list, together with the road type "street".

3.3.3.3 IN(1)

Toponyms following "IN" are largely cities, villages, landmarks and public buildings. Toponyms are thus searched for using Nominatim. Search results of the class "boundary" or "place" are preferred, because both results contain accurate bounding box of a city or village. If these classes are not found, the first search result is returned. The resulting coordinates are added to the bounding box list.

3.3.3.4 AT(1)

The toponyms co-occurring with "AT" are very similar to toponyms co-occurring with "IN". Consequently, the seach engine used is Nominatim. The code behind this predicate is also exactly the same as "IN". The only difference between the two is that locations on a highway near a city or other landmark are mostly referred to with "AT", and not with "IN". One might think "AT" needs an enlarged bounding box compared to "IN", but since Nominatim has a tendency to return oversized bounding boxes, the model uses the standard bounding box.

3.3.3.5 BETWEEN(2)

Toponyms occurring after "BETWEEN" can be of any kind, though most are cities and villages. Both toponyms are looked up through Nominatim, with the result of class "place" given priority. In case there is no result of class "place", the first result is returned.

The bounding box between these two toponyms takes the shape of a square, where both toponyms are at opposite ends of a diagonal in the square. The other two points forming the bounding box are located at an angle of forty-five degrees from both sides of the diagonal, according to the properties of a square. To determine the distance between either point and either toponym, the length of the side of the square needs to be calculated. This is done by using the Pythagorean equation below on the left, where d is the diagonal and s represents the side of the square. The length of the diagonal is thus divided by the square root of two in order to obtain the square edge length.

$$d^2 = s^2 + s^2$$
$$d = \sqrt{s^2 + s^2}$$
$$d = s\sqrt{2}$$
$$s = \frac{d}{\sqrt{2}}$$

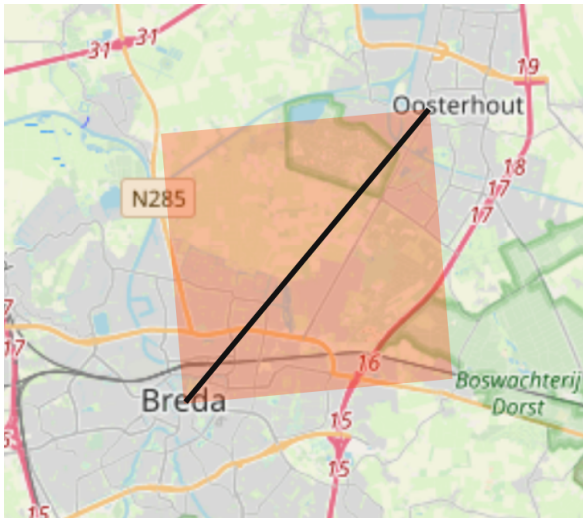

Figure 5: Example of bounding box created by "BETWEEN('Breda','Oosterhout')". Red zone is bounding box, black line is diagonal between toponyms.

3.3.3.6 INTERSECT(1/2)

Toponyms occurring in this predicate are assumed to be streets. This predicate sometimes occurs with only one toponym. In this case, the other toponym that makes up the intersection is usually located somewhere else in the sentence or in a neighbouring sentence. Therefore, "INTERSECT(1)" is not processed, but immediately added to the road list, marked with road type "intersect". This predicate will be dealt with later on, when the two resulting lists are merged.

In the case of "INTERSECT(2)", both toponyms are looked up through the Overpass API. The Overpass API searches for nodes where both toponyms overlap and it returns the coordinates of all nodes where this occurs. The resulting coordinates are added to the road list, with road type "crossroads".

3.3.4 Predicate result merging

Because street toponym predicates retrieve coordinates for all streets in the Netherlands with the same toponym name, these need to be filtered in order to be of any use. Algorithm 3 attempts to merge elements from both lists in order to narrow down road search results. The still unprocessed "INTERSECT(1)" predicate-toponym pair is also matched with other results using "INTERSECT(2)". Any entries in both result lists that do not match with any other entry, remain in the list. To see how and where this algorithm is used, view algorithm 2.

Algorithm 3 mergeLists

```
for road in road list do
  if road type is street or crossroads then
    for bounding box in bounding box list do
      if any part of road falls within bounding box then
        1. Remove parts of the road that lie outside of bounding box.
        2. Remove bounding box from bounding box list.
      end if
    end for
  else if road type is highway then
    for bounding box in bounding box list do
      if bounding box predicate is not "IN(1)" then
        if any part of road falls within bounding box then
          1. Remove parts of the road that lie outside of bounding box.
          2. Remove bounding box from bounding box list.
        end if
      end if
    end for
  else if road type is intersect then
    for road2 in road list do
      if road2 is does not equal road and road2 type is not crossroads then
        if road2 and road intersect then
          1. Remove road and road2 from road list.
          2. Add new intersection to road list.
        end if
      end if
    end for
  end if
end for
```

4 Results

To evaluate the model described in section 3, fifty articles were selected from the testset. Each article contained one complex location. For each complex location, its coordinates were manually annotated, including a precision radius in meters. For example, when a complex location is specified by two intersecting streets, the precision radius is small, set at one hundred meters. But when a complex location is vague, such as when an accident has happened on a highway between two cities, the precision radius can be as large as five thousand meters. A location suggestion by the model is classified as correct of false based on the distance to the annotated location. If the distance is larger than the annotated radius, it is considered false, and vice versa. It is described more in-depth by algorithm 4 below. Code can be found here[3].

As explained in section 3.3.3, the model returns a road list and a bounding box list. Because of the way the model works, the road list actually contains all interesting, more specific locations. The bounding box list is merely used as a way to specify elements in the road list. This is why all fifty annotated complex locations happen to fall into the road list category. Because of this, only elements in the road list are compared to the annotated coordinates.

Algorithm 4 evaluateModel

for annotated article in testset **do**
1. Calculate bounding box result list and road result list using algorithm 2
for road result in road result list **do**
1. Calculate centroid of road result coordinates.
2. Calculate distance between centroid and annotated coordinates.
3. Add distance to distance list.
end for
1. Get smallest distance from distance list, thus extracting best road result.
if smallest distance < annotated precision radius **then**
1. Increment correct counter
end if
1. Append smallest distance to distance result list
end for
return distance result list **and** (correct counter / testset length)

This evaluation structure has been chosen because of its common use in this research field, explained in section 2.4. Because the model occasionally fails to suggest any locations for an article, the rightmost column is included. If the model does not return any locations for an article, there is no distance value given. This means that the given median and average values are based on fifty articles minus the articles where no location is detected. For result optimization, various window sizes[3.2.2] were tested.

	Median value (km)	Average value (km)
Annotated precision radius	0.100	0.622

Table 1: Annotation statistics

Window size	Accuracy	Median distance (km)	Average distance (km)	Locations not found
4	0.26	0.36	6.46	18
5	0.26	0.36	5.95	16
6	0.26	0.36	5.95	16
7	0.26	0.36	5.95	16

Table 2: Model results on annotated complex location testset

[3] https://github.com/toonmeijer/geocoding-nlp/blob/master/implementation/evaluations/results_complex_classification.py

5 Discussion

5.1 Result analysis

This research set out to answer the following question:

Using spatial adpositions to model the relationship between toponyms, how accurately can complex locations be geolocated?

The results indicate that, using the most commonly occurring spatial adpositions, complex locations can be geolocated with 26% accuracy. The described model occasionally shows the desired complex location, but often fails due to some misstep in the algorithm. Also, the conclusion can be drawn that a bigger window size leads to better performance.

Reflecting on the different window size results, it makes sense that on average, the bigger the window, the better the result in the case of this model implementation. This is because the window size simply increases the amount of words that are considered as spatial adposition. In some sentences, adpositions can be far removed from their toponym, thus falling outside of the window. Below is an example of such a sentence, where "Nieuw Amsterdamsestraat" is the toponym. With a window size of four, the spatial adposition "over" would not be detected, and the toponym would not be used since there is no other spatial adposition in the window.

"... fietste het meisje rond 20.30 uur over het fietspad van de Nieuw Amsterdamsestraat ..."

However, after a window size of five, the results do not improve. This is because a spatial adposition will rarely be further than five words away from its toponym. However, in a bigger testset, an improvement could perhaps be noticed moving from a window size of five to six.

As can be seen in table 1, the model does not suggest any road location for at least sixteen out of fifty times. Because the entire model entails a number of steps and algorithms, there are various reasons for an incorrect location suggestion. Firstly, a toponym is not detected by the geoparser. Secondly, a toponym is detected, but the spatial preposition is not supported by the model. Below is an example of such a case. The model will pick up the first toponym, "Staringlaan", but it will not use the second toponym.

"De automobilist [...] reed over de Staringlaan en sloeg linksaf om de H. Heijermanslaan in te rijden."

The third option is when the toponym and supported spatial adposition are detected, but the toponym is not found on OpenStreetMap. This occurs when spelling mistakes are made by the author of a news article, or when a toponym name has been changed since publishing.

5.2 Implications & limitations

Compared to the average and median distance results of related research, of which Melo and Martins (2017) have created an overview, the proposed model outperforms all others. However, these results cannot be directly compared, because each approach focuses on a different use case with varying levels of generalizability. The proposed model has been created with complex locations in Dutch traffic accident news articles in mind, and these articles always have the same structure. Further testing is required to see how well it would perform on more general Dutch text.

Applying the proposed model to a different but related language, with the necessary spatial adposition conversions, would also not guarantee the same results. As explained in Han et al. (2014), toponym disambiguation is much more difficult for a geographically-diverse language such as Spanish and English, compared to geographically-focused languages such as Japanese and Dutch. The geographical disambiguation would most likely need to be improved in order for the model to perform well on geographically-diverse languages.

6 Conclusion

This research aimed to quantify how accurately complex locations can be geolocated by using spatial adpositions to extract the relationship between toponyms. This has been attempted by linking an existing geoparser to the model proposed in this research, that is based on converting common spatial adpositions to more general predicates for further processing. The results indicate that the model occasionally works as expected, but often returns no or low-accuracy results. The accuracy reported by the model on an annotated complex location testset is 26%.

As has been discussed in the introduction, the need for addressing complex location geocoding was made clear by previous research into Dutch text geocoding (Olieman et al., 2015). This approach was taken because it seemed like the most straightforward way to tackle the problem. By looking at the most common words appearing near toponyms in complex location articles and selecting all spatially relevant ones, the model obtains its input. The selected spatial adpositions are then converted into a fitting predicate. Each predicate is translated into sets of coordinates in a predicate-specific way, and at last, coordinate sets are merged to form the end result. In retrospect, the approach should have also included a way to geocode toponyms with an unsupported spatial adposition. This could have improved the overall accuracy of the model.

There are various ways in which this research can be taken a step further. Whereas this approach has focused on single words, it would most likely benefit the reported accuracy if spatial adposition combinations were also detected. There are also a few spatial adpositions such as “uit” and “richting” that were supposed to be supported by the model, but were ultimately not. Detection support for “richting” can be found in the code, but it is unused in the geocoding step. Adding these to the model would require more research, because the meaning of these words depend on sentence context.

7 References

Adelfio, M. D. and Samet, H. (2013), Structured toponym resolution using combined hierarchical place categories, *in* 'Proceedings of the 7th workshop on geographic information retrieval', pp. 49–56.

Buscaldi, D. (2011), 'Approaches to disambiguating toponyms', *Sigspatial Special* **3**(2), 16–19.

Calazans Campelo, C. E. and de Souza Baptista, C. (2008), Geographic scope modeling for web documents, *in* 'Proceedings of the 5th Workshop on Geographic Information Retrieval', pp. 11–18.

Cohn, A. G. and Hazarika, S. M. (2001), 'Qualitative spatial representation and reasoning: An overview', *Fundamenta informaticae* **46**(1-2), 1–29.

Ding, J., Gravano, L. and Shivakumar, N. (2000), 'Computing geographical scopes of web resources'.

Goh, D. H.-l., Lim, E.-p., Sun, A., Wu, D. and Zong, W. (2005), On assigning place names to geography related web pages, *in* 'Proceedings of the 5th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'05)', IEEE, pp. 354–362.

Gritta, M., Pilehvar, M. T., Limsopatham, N. and Collier, N. (2018), 'What's missing in geographical parsing?', *Language Resources and Evaluation* **52**(2), 603–623.

Hall, M. M. and Jones, C. B. (2008), Quantifying spatial prepositions: an experimental study, *in* 'Proceedings of the 16th ACM SIGSPATIAL international conference on Advances in geographic information systems', pp. 1–4.

Han, B., Cook, P. and Baldwin, T. (2014), 'Text-based twitter user geolocation prediction', *Journal of Artificial Intelligence Research* **49**, 451–500.

Hendriks, B. (2019), 'Retrieving, cleaning and analysing dutch news articles about traffic accidents'.

Leidner, J. L. and Lieberman, M. D. (2011), 'Detecting geographical references in the form of place names and associated spatial natural language', *SIGSPATIAL Special* **3**(2), 5–11.

Melo, F. and Martins, B. (2017), 'Automated geocoding of textual documents: A survey of current approaches', *Transactions in GIS* **21**(1), 3–38.

Monteiro, B. R., Davis Jr, C. A. and Fonseca, F. (2016), 'A survey on the geographic scope of textual documents', *Computers & Geosciences* **96**, 23–34.

Olieman, A., Kamps, J. and Claros, R. M. (2015), 'Loclinkvis: A geographic information retrieval-based system for large-scale exploratory search', *arXiv preprint arXiv:1509.02010* .

Overdijk, C. (2020), 'Location recognition and geocoding in dutch traffic news articles'.

Rauch, E., Bukatin, M. and Baker, K. (2003), A confidence-based framework for disambiguating geographic terms, *in* 'Proceedings of the HLT-NAACL 2003 workshop on Analysis of geographic references', pp. 50–54.

UN Department of Technical Cooperation for Development et al. (1992), *United Nations Conference on the Standardization of Geographical Names*, Vol. 2, United Nations, p. 9.

Woodruff, A. G. and Plaunt, C. (1994), 'Gipsy: Automated geographic indexing of text documents', *Journal of the American Society for Information Science* **45**(9), 645–655.

Worboys, M. F. (2001), 'Nearness relations in environmental space', *International Journal of Geographical Information Science* **15**(7), 633–651.