# Method

For achieving the goal of tying Dutch properties to the corresponding English ontology member, it has to be established that these two are indeed each other’s counterparts. There are several ways to do this, such as automatic translation or assuming they are the same when they have the same value. As both of these methods have their flaws and benefits, the second way is the most universal applicable without relying on external sources. Also, it is believed that an automatic mapping system could profit from large collections of <entity, attribute, value> triples harvested from Wikipedia templates (Bouma, et al.).

To justify the assumption that the same value belongs to the same attribute in different languages, as much doubt or ambiguity has to be taken away first in order to make sure that the matching values belong to the same entity. For example, it is not unimaginable that one person died on the day that another person was born. Therefore the descriptions of the values in this research will only be matched if the DBpedia URL corresponds. The risk that two equal values show up on the same page is considered naught, and is favoured of the risk of getting wrong translations using the other method.

The matching process of the value in the two languages will involve a *Python*[[1]](#footnote-1)script that loops through the two documents and creates a dictionary with the value and page URL as key, and the English and Dutch name for it as value. As the attributes of entire DBpedia take up several Gigabytes, a fast and fluent approach is needed in order to keep the runtime manageable. The only way to do this is by keeping all the data in one single dictionary, as otherwise two large files have to be compared line by line. This results in reading one file line by line for every line in the other file, which results in exponential longer runtime if more data is added.   
Due to the fact that Dutch properties can have multiple meanings in English, and only one can have the ‘SameAs’ relation that will be assigned, the type of the page also has to be included. For example, the word ’doop’ can mean baptism for a child and christening of commission date of a ship. With the use of the type, both can be kept in the mappings file. This way the method is wider applicable, as it is not restricted to one single type of enitity, such as a person.

All necessary information can be found in the files of the DBpedia dump[[2]](#footnote-2), where three Dutch files contain all required input:

* **Mappingbased-properties\_nl.nt**

Contains the Dutch enitity URL, the ontology attribute and the value

* **Infobox-properties\_nl.nt**

Contains the Dutch enitity URL, the Dutch property attribute and the value

* **Instance-types\_nl.nt**

Contains the Dutch entity URL, the RDF Type identifier and the ontology attribute

Combining all three of these files will create a symmetrical mapping file that links the Dutch property to the official English DBpedia ontology for a certain type of page. For the development of the system not all entries in these files are needed, as the full runtime will be too long for convenience. An arbitrary selection will be sufficient, for instance all attributes that contain a date.

The evaluation of the experiment will be done by comparing two methods: manually annotate how many cases are mapped right in the raw output files, versus filtering single cases out and then annotate. After all, it is reasonable to assume that if a mapping is wrong, it will be because the value was with the wrong attribute. These cases will be very rare mistakes, and will probably not happen more than once per page type.

1. https://www.python.org [↑](#footnote-ref-1)
2. http://downloads.dbpedia.org/2015-04 [↑](#footnote-ref-2)