Fraunhofer FOKUS

FRAUNHOFER-INSTITUT FÜR OFFENE KOMMUNIKATIONSSYSTEME FOKUS

DUPLICATE EVALUATION - POSITION PAPER BY FRAUNHOFER FOKUS

Simon Dutkowski (simon.dutkowski@fokus.fraunhofer.de), Andreas Schramm (Andreas.schramm@fokus.fraunhofer.de)

Berlin, 23.09.2015

1 European Data Portal and Duplicates

Today, the worlds open data ecosystem is organized in a hierarchical structure, where datasets and their metadata are usually published on leaf nodes. These nodes are in most cases local area portals for specific regions or specialized portals for a specific class or type of data, like geo or statistic portals. This constellation potentially results in duplicate datasets in a portal that is in a higher position in the hierarchy. Many publishers are often not sure where to publish and finally they simply publish their data twice, once on the local portal and additionally e.g. on the national geo portal. A national open data portal probably harvests the local and the geo portal and, if no measures are taken, will finally contain the same data twice with slightly different metadata descriptions.

Another source for duplicates arises when datasets are harvested from several different portals which harvest from one portal. The European Data Portal (EDP), which is relatively high in the harvesting hierarchy (if not on top), is facing now the fact that there are really many possible duplicates, even harvested from single sources, as in most cases no other portal has a proven mechanism to avoid duplicates.

2 Why it is important to avoid Duplicates

In the beginning of the EDP development the topic of identifying and eliminating duplicates was not of high priority. There are some good reasons why we should try to avoid duplicates. Most important, we should give the end user the best possible experience when using an open data portal. That means whenever someone queries for datasets during a search, the result should not contain duplicates, which in the best case is just annoying. Second, we assume that the amount of duplicates does also have a noticeable impact on the performance and the resource consumption of the portal that could be avoided, at least in the European portal, due to missing mechanisms or any kind of direct support through the DCAT-AP standard, e.g. either using a mandatory identifier or extended provenance information. We think that there is no hundred percent safe method to prevent duplicates as long as there are more than one metadata schema is involved, therefore we approach another way to identify possible duplicates. This approach can be seen as a complementary mechanism to find also duplicates where all other mechanisms failed. Because it is only a way to determine the magnitude of equality it should not be used to simply eliminate duplicates. The final decision if two dataset are really duplicates and may be eliminated should be made by an appropriate human authority.

Our opinion is that metadata schemas like DCAT-AP could play an important role for the prevention of duplicates. Mandatory unique identifiers or any kind of provenance information in one schema are obviously not sufficient, as they could be lost during transitions while harvested. Of course, at first we need to take the data providers at hand and educate them to publish their metadata appropriately. This can be done e.g. by providing good best practice guide lines. But from the pure technical view, how can we improve the metadata schema or vocabularies to minimize the appearance of duplicates? Again, we think the "heterogeneous" ecosystem in the current situation will always require an approach like introduced below, but we also think we should discuss any possible improvements e.g. for identifiers or provenance. May be these improvements could even support any approaches which are similar to ours.

3 Approach for Duplicate Evaluation

Duplicate datasets in the European Data Portal typically origin from multiple harvesting of the same original dataset, but possibly in different versions. Thus their titles and descriptions may vary a bit. In addition, different (automatic) translations may be involved, resulting in similar but not identical texts.
The first approach was to carry out pairwise comparisons of texts, viz. by the "Term Frequency/Inverse Document Frequency" (TF/IDF) method and the vector space model with a cosine similarity measure. Although this is a sound approach to evaluate text similarities, it turned out to be ways too slow. With respect to speed, we refined our requirement as follows:

- The number of EDP datasets is, or will eventually be, in the order of million(s).
- Use case 1: From the interactive GUI, one individual dataset (i.e., its title and description) is to be compared with all others, in acceptable time.
- Use case 2: Every dataset is compared with every other one, in batch mode, within hours.

From this we drew the following design decisions:

1. A single comparison must not last more than a few µs.
2. Use case 2 will only compare datasets with the same language mark of the containing catalogue (in EDP, there are 70+ dataset catalogues with ~35 different language marks).

The second decision reduces the number of comparisons, which is quadratic to the number of datasets, back to a manageable order of magnitude.
So at this point in time, the challenge was to carry out a single comparison in a few µs. Within this time, no real text comparisons can be carried out; something faster needs to be found. Here the concept of "Locality Sensitive Hashing" (LSH) comes into play. Regular hashing functions are supposed to yield results that look random, and to yield possibly different results for different inputs, no matter how similar. In contrast, LSH functions are designed to exhibit some concept of continuity, i.e., to throw similar inputs to similar results. Thus, using LSH, the similarity of two datasets can be estimated by comparing the hash values of their respective titles and descriptions. These hash values need to be calculated in advance, which is no problem as it only takes linear effort.
Various approaches to LSH can be found, with varying complexity. The most promising for our purpose seemed to be TLSH by Jonathan Oliver, Chun Cheng, and Yanggui Chen [1]. It is algorithmically relatively simple. It takes a character string as input, and its result consists of some header information and b two-bit numbers, where b is a design parameter (the number of "buckets", without going into details). In order to get a distance measure of two strings, one simply checks how many of their b two-bit numbers differ. In addition, since these numbers essentially are length-agnostic, the lengths of the two strings have to be compared. All this can be implemented very efficiently, which we did.
First experiments with TLSH yielded disappointing results. The warning that TLSH would need strings of some length in order to give meaningful results turned out to be legitimate. In the average, the titles and descriptions of the EDP datasets were too short for TLSH with its original parameters to be useful. Thus, in order to cater for our relatively short texts, we reduced the number of buckets, *b*, from originally 256 to 64. With this minor adaptation,

[1] TLSH – A Locality Sensitive Hash, Jonathan Oliver, Chun Cheng and Yanggui Chen, Trend Micro, North Ryde, NSW, 2113, Australia, jon_oliver@trendmicro.com

TLSH worked very well even for our short texts. (Some waste of precision for longer texts might be the consequence, but this is bearable.)
Thus, a comparison of two texts is carried out by a comparison of two 16-byte TLSH fingerprints plus a comparison of the text lengths. This can be implemented very efficiently, the basic step being x'-oring' two respective bytes and making a table look up. The computation time consumed by one comparison is below 1 µs; so our self-imposed time constraint has been met.

At this point, after experimenting and identifying some duplicate datasets, we discovered that text similarity is not a reliable indicator for datasets being duplicates, which after all is what we are after. We saw examples of almost identical texts, e.g. with the word "gas" in one of them and "electricity" in the other, and other examples of that kind. Obviously, these do not refer to duplicates but to genuinely distinct datasets which contain data for distinct energy carriers. So for the time being, detecting datasets with similar titles and descriptions is only the starting point for duplicate detection; the ultimate decision has to be taken by a human. As an outlook, one might imagine to automate this distinction, but this would require distinguishing related words from synonyms, and this might constitute too costly a departure from the algorithmic simplicity of TLSH.