Machine learning using Natural Language Processing (NLP) in support of the GEO Knowledge Hub

Iulia Florea, Lionel Menard, Philippe Blanc, Sébastien Travadel

1. General context

This work has been conducted during a four-month internship in MINES ParisTech for Center "Observation, Impacts, Energie (O.I.E)" [6]. The activities of the O.I.E center focus on the temporal and geographical aspects of questions related to resources of renewable energy and the environmental impact of production and usage of energy. It develops ways of representing in a mathematical way the geographical reality, by observing, modelling and decisioning.

The work aims to investigate the use of Machine Learning (ML) technique and more precisely Natural Language Processing (NLP) approach in order to provide a first step toward a better and more accurate knowledge of inner content of solar observations-based repositories going beyond current existing search techniques.

1.1 GEO general context

GEO[1] is a partnership of more than 100 national governments and in excess of 100 Participating Organizations that envisions a future where decisions and actions for the benefit of humankind are informed by coordinated, comprehensive and sustained Earth observations. GEO is a unique global network connecting government institutions, academic and research institutions, data providers, businesses, engineers, scientists and experts to create innovative solutions to global challenges at a time of exponential data growth, human development and climate change that transcend national and disciplinary boundaries. The unprecedented global collaboration of experts helps identify gaps and reduce duplication in the areas of sustainable development and sound environmental management.

The goal of the GEO Knowledge Hub is to provide, in a single place, access to all documents associated to Earth observation applications. These documents include: (a) the research paper or report which describes the methods and results; (b) the software algorithm used for processing; (c) the in situ and imagery data used; (d) the results. One may also think of additional resources such information about networks of active researchers in the related domain. Nowadays, these components are stored and distributed in different ways. Papers and reports describe methods, generally shared as PDF files. Algorithms are mostly written in scripting languages embedded notably in Jupyter Notebooks or R-markdown files. Increasingly, satellite data is stored in the cloud and in situ data is available via trusted repositories. However, current search engines do not provide links or structured relationships between these components of a document, thus reducing their fitness for use. The knowledge hub aims at providing linkages between these components of an Earth observation application.

We define the GEO Knowledge Hub to be a set of curated and linked documents that contain relevant information for Earth observation applications. Examples of such documents include for example HTML files, PDF files (reports or papers), Jupyter Notebooks, R or python markdown files, GitHub pages, repository entries linking to dataset stores with an assigned Digital Object Identifier (DOI),

Amazon Web Services (AWS) or other links to datasets, OGC service links for data, video, etc. (see Figure 1). We also use the term "document set" to describe a set of documents linked to the same application.

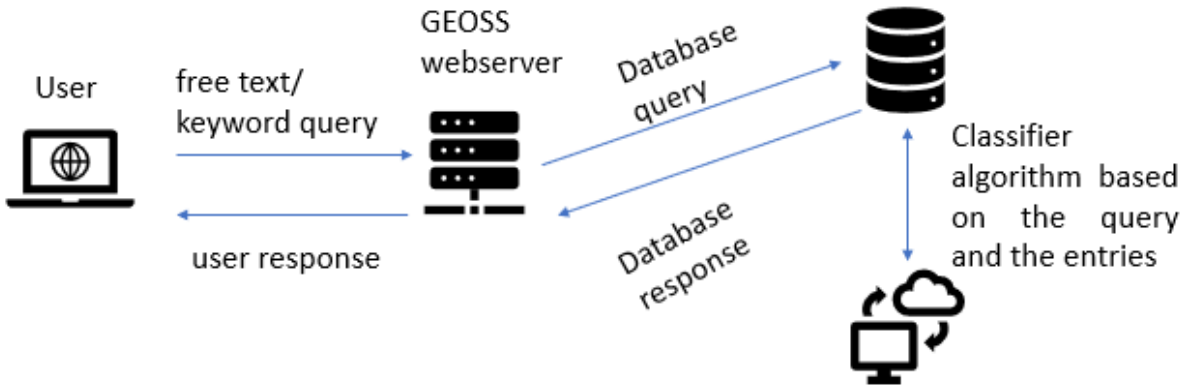
1.2 Building a new search system

The geoportal provides a basic search system, which looks for entries in the GEOSS database and retrieves them without taking the relevance into consideration. Most of the entries are metadata records, which describe access to web services and maps related to solar observations, typical meteorological years, solar potential or data related to atmosphere. Users can apply multiple filters related to the data type, the organization that provided the data or the keywords. A new system of distributed search is in test, which tries to gather data from additional sources, in order to obtain a single entry point for knowledge related to solar observations and global resources. At this point, it looks through scientific papers and metadata records for web services or geo maps.

Apart from retrieving specific results from the database, we propose a new solution that also orders the entries in a hierarchy depending on their relevance, provides entries that are not automatically found by a simple search and which may be relevant for an expert. Furthermore, our solution aims to connect, in the future, similar resources from multiple sources. Another benefit is also getting in-situ data through that unique entry point.

The main scope of the project is to build a "smart" search system, which is able to associate scientific papers with web services, to get data from the multiple sources in a coherent way and to show the results based on their relevance. For users, this process is transparent, they start by searching in the webserver using keywords or free text and, in the backend, there is a classifier that provides not only the results obtained by querying, but also entries related to the subject, which may be useful for an expert.

The following image explains the big picture of the process presented by this paper:



2. System architecture

The work aims to investigate the use of Machine Learning (ML) technique and more precisely Natural Language Processing (NLP) approach in order to provide a first step toward a better and more accurate knowledge of inner content of EO based repositories going beyond current existing search techniques. The scope of the techniques described below has been tested on two different curated and linked document repositories:

- webservice-energy.org an Open Geospatial Consortium (OGC) Catalog Service for the Web (CSW) GEOSS complient catalogue containing over 1600 metadata records encoded in ISO 19139 and dedicated to renewable energy.
- Zenodo.org a general-purpose open-access repository developed under the European OpenAIRE program and operated by CERN.

The added value of ML algorithms is the amount of information that can be efficiently retrieved beyond the merge of separated searches. Indeed, it does not imply simply distributed search through multiple sources, but it also adds results that cannot be provided directly. By default, a search in the database gets only the entries that contain the words in the query. Our solution implies adding entries from the database that contain similar concepts, synonyms of the words in the query. In case of webservice-energy catalog, ML can provide extra metadata records for a query, which can be interesting from an expert's point of view. For instance, when looking for "solar potential", the outcome may provide results related to this that do not contain the expression. In order to achieve that, we focused on computing text similarity and tried to classify the entries in the database into classes.

To obtain that, we need to create lexical fields for multiple words related to solar observations, renewable energies or common expression that can be used to determine similarities between texts. We need to create a world model with an adapted vocabulary, so we gathered scientific papers from Zenodo. We retain as many papers as possible related to the fields contained by the GEOSS database, so natural language processing algorithms can better determine similarities between words. We choose this repository since it provides an endpoint API, which can be used to retrieve as much information as needed.

While trying to extend knowledge, we focused on classifying the entries in the database while using classification based on text similarity. For this, there are two main classes of algorithms that can be used: unsupervised and supervised. The most general solution is to use an unsupervised algorithm. Its advantage is that it does not depend on the content of the database and it can be run in the same manner for any field of application. Also, it does not require human expertise or labelled document. On the other hand, this general approach may not retrieve good results in every case. It needs a much larger amount of information and its accuracy depends on it. In this case, a large corpus is mandatory for training.

A supervised classifier assisted by an expert, can provide more accurate results, especially in case of lower amount of information. Starting from several correctly classified entries, a model can be trained and then applied to similar data with much better results.

In this case, we ran representative algorithms for classification, while trying to understand their advantages and limitations, especially since they depend the most on to the amount of input data.

For the both approaches, the first step is to preprocess the input data, in order to have coherent results. For an algorithm, for instance, the singular and plural forms on a noun are different entities. In this case, all the words should be brought to their basis form. Some of the state-of-the-art solutions are based on the number of word occurrences. Therefore, since prepositions, adverbs and other connecting words are the most common in phrases, they must be removed, in order to emphasize the

focus on the important words. This preprocessing consists in removing stop-words from the vocabulary.

In case of unsupervised machine learning, there is no additional input from the users, so the connections between words are built automatically, using the *corpus* made of metadata records. On the other hand, when using hybrid or supervised algorithms, it is sometimes necessary to compute similarities between words. This can be done using *word2vec* library, presented in section 2.2.1 and it requires an accurate language model. One of the best way to obtain a language model is by using technical papers in the required field. In this case, using a general-purpose open-access repository helps obtaining the necessary vocabulary easily.

2.1 Data Science in the context of text classification

Text classification is the task of assigning a set of predefined categories to free text, received as input. It can be used to organize structure and categorize any type of information. In order to do this in a faster and more cost-effective way, techniques such as machine learning and natural language processing are necessary.

In case of automatic text classification, there are three different types of systems:

- Rule-based systems
- Machine Learning based systems
- Hybrid systems

Rule-based approaches classify text into organized groups by using a set of handcrafted linguistic rules. These instruct the system to use semantically relevant elements of a text to identify relevant categories based on its content. In order to obtain it, it is important to have some predefined language and categories set by an expert. Each rule consists of an antecedent or pattern and a predicted category. The advantage of this system is that is human comprehensible and can be developed in time. On the other hand, it may not scale well, since it requires new rules each time and also human analysis.

ML based systems start from samples of predefined labels and data and they build a model that can be used for prediction. In this case, the user has to build some rightly classified data samples and run a feature extraction algorithm that will build associations between the categories and the text. In this case, the number and the quality of the samples are important. The more they are, the easier is for the algorithm to build up patterns to further classify correctly. The main advantage is that they are scalable, once the model is built, in can run with minimal human intervention.

Hybrid models combine base machine learning classifiers with rule-based systems, used to adapt to changes in time. In this case, whenever the model is returning false results, new sets of rules can be added to fix wrong matches.

Building a classification algorithm for text requires making language understandable for computers. Natural Language Processing is the field of Computer Science that identifies and extracts rules in order convert unstructured language data in a form that the computers understand. Optimally, the algorithm should understand the meaning of a sentence and collect the essential data.

NLP uses two main techniques: syntactic and semantic analysis. The former refers to the arrangement of words in sentences in order to make sense from a grammatically point of view and the latter is used to assess how the grammatical rules are used in case of natural language.

Syntax analysis refers to breaking text into distinct units, such as sentences, then words, identifying parts-of-speech and reducing various inflections of words into a single form. On the other hand, semantic analysis involves recognizing entities, such as persons, organizations or locations, multiple senses of words, depending on the context and generating inflections in order to have human understandable text. This is the complex part of NLP applications, the one that is still in progress.

For this project, we focused mostly the syntactic analysis. Since the texts that we analyzed are short and technical, the words are unlikely to have multiple radically different senses. Furthermore, since classification is done using domains and themes from the energy sector, we did not focus on the retrieving information related to names and locations. In this case, the purpose of the application is to determine the main subject of a small sentence or paragraph. In other words, we assumed that obtaining links between words is enough to find which are dominant in a sentence and may influence the subject.

3. Implementation

In order to make text comprehensible for computers, it is necessary to apply the following transformations to achieve better results.

3.1 Data preprocessing

In order to have coherent results and reduce noise in the data, the first step is to clean up the text and transform it so an algorithm can digest. In general, this consists of four parts [7]:

- Cleaning: all numbers, punctuations and special characters are removed, together with the stop-words, which connect parts of the sentence. These are common in the text and affect the quality of the results. For example, words like “the” or “and” can be removed by comparing the input text to an existing list of stop-words. NLP libraries provide such lists. Also, words capitalization is creating bias, especially because of the uppercase words at the beginning of a sentence. The most common approach is to reduce everything to lower case. Also, cases like changing meaning, such as transforming “US” to “us” should be taken into consideration.
- Annotation: it consists of the application of a scheme to texts such as structural markup and part-of-speech tagging. In this case, the first step is tokenization, meaning that paragraphs will be split into sentences and, then, sentences into words. Then, if necessary, parts of speech can be added to each word. NLP libraries provide tagging and it is a step that can make a difference when determining the meaning of a sentence.
- Normalization: words should be lemmatized (set the present tense from the past tense, change from the plural to the singular, change the word to its base). Also, all the abbreviations should be changed to their corresponding word sentence, so the algorithm would learn the English words and not lose any technical information.
- Analysis: it consists of statistically probing, manipulating and generalizing from the dataset for feature analysis. This is the step where a language model can be built for further use.

3.2 Building a language corpus for a technical field

In order to have functional classifiers, we need to use the build the language model used for lexical fields. We do that by preprocessing scientific papers related to entries in the GEOSS database and sending them as input to the Word2vec algorithm presented below.
Word2vec[2] is a two-layer neural network meant to process text, as presented in the image below.

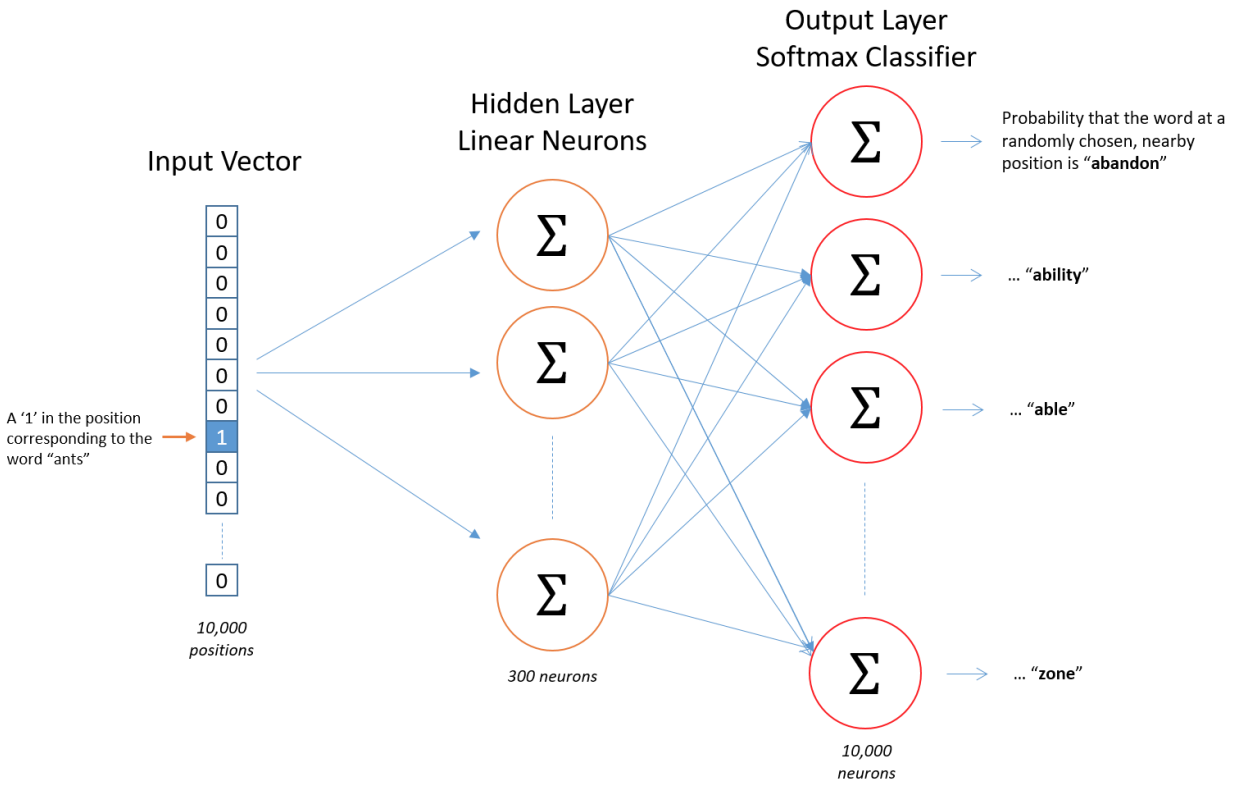


Figure 1: General architecture of the hidden layers of a classifier

Its input is a text *corpus* and its output are a set of vectors: feature vectors for words in that corpus. The transformation from words to vectors is called "word embedding". main reason of doing this for ML algorithms is to apply algebra and vectoral formulas instead of processing words. In this approach, each word will have a representation in the (high-dimensional) vector space. For example, in a 4-dimension model, the word "cake" can be represented as a [0.5, 0.7, 0.25, 0.1] and the word 'dog' will be [0.1. 0.2, 0.4, 0.5]. The key concept of Word2Vec is to locate words which share common contexts in the training corpus, in proximity —or high correlation— in the vector space. So, the words having common meaning will be closer to each other in the space, while the others will be further.
As mentioned above, the purpose of word2vec is to represent words by dense vectors of real-valued numbers. In order to this, it must learn relationships between words on a large corpus and to return embeddings.

A model has been trained on part of Google News dataset (about 100 billion words). It contains 300-dimensional vectors for 3 million words and phrases. Due to the data dimensions, this model is reliable for application for generally used words.

On the other hand, in case of a technical field, there are a lot of specialized terms that are hard to be found in the everyday language. In this case, the solution is to build a new corpus, based on published papers in a certain domain. In this case, it is possible to use a big database, such as Zenodo or Google Scholar, which provide a large range of articles.

For instance, for our specific corpus of interest in the framework of GOE, we have used the Zenodo database, which exposes a connection endpoint without limiting searches. In this case, we set a list of queries related to renewable energy and solar observations, which returned a final list of almost 1500 unique papers. Based on them and using the Gensim implementation of word2vec[3], a model of embedding words is obtained.

3.3 Classifying entries in the database

3.3.1 Supervised classification based on metadata keywords

A first approach to classification is to get the most relevant keywords in the corpus and use them as labels for supervised classification.

The first step is to build a keyword list that can be used for text classification. In order to do this, we get the list of keywords from the metadata records and select the ones that are the similar to the search keyword (above a predefined threshold). Also, the must be then must be lemmatized in order to avoid differences between singular, plural word or case-sensitivity.

For each metadata record, we are looking at the keywords that are attached and, if they are also contained in the list of the best keywords, then the preprocessed metadata record is written in the training file, together with its corresponding best keywords, as labels. The same text will be written also in the test file, so the classifier will be able to find the best category for each metadata record.

Then, we look through the list and using word2vec to determine the distance between words, select only those who have a similarity from the search query that passes through a certain threshold. For example, for catalogue, for a search query of “solar energy” and a similarity index of 0.5, I get the following keywords that will be used as labels: 'energy', 'solar energy', 'solar radiation', 'solar', 'surface solar irradiation', 'electricity generation', 'surface solar radiation', 'solar august', 'solar march', 'energy resource', 'solar monitoring', 'photovoltaic', 'renewable energy', 'marine energy', 'energy marine', 'solar electricity', 'solar annual', 'annual energy', 'energy generator simple', 'energy generator', 'energy ministry' and 'wind energy project'.

When the similarity is higher, then the list of keywords would be shorter, containing only closer synonyms.

All the text from the ISO 19139 metadata records, including title, abstract and keywords, is then preprocessed and will be used for either training or testing. Each metadata record, if it has at least one keyword that belongs in the selected list of best keywords, will be written it in the training file, together with its best keywords, which are set as labels. For example, the following example is part of the training file:

__label__solar-energy surface, solar, irradiation, daily, exposure, diffuse, year, average, monthly, mean, receive, horizontal, plane, always, face, copyright, port, say, university, faculty, science, department, physic, mine, developed, method, convert, image, map, store, database, version, compute, least, valid, month, complete, miss, take, account, value, day, atmosphere, contains, gap, fill, available, component, direct, normal, ray, use, publish, empirical, model, data, provide, soda, service, since, use, academic, teach, research, energy, environment, climate, others, company, sit, plant, size, monitoring, production, French, company, charge, provide, also, series, user, tailor, service, similar, information, publication, archive, doc, help

If the metadata record does not have any keyword in the list, it is written to the test file to be labeled through a machine learning algorithm.

In order to determine the best label for each entry, we used fastText[3] classifier trained on the labeled file.

FastText[3] is an extension to Word2Vec, proposed by Facebook in 2016. Instead of using individual words as input for the neural network, FastText breaks words into several n-grams, of various sizes, including the word itself. For instance, the tri-grams for the word *home* are *{hom, ome}*. The bigrams are *{ho, om, me}* and the 4-gram is *{home}*. The word embedding vector for each word will be the sum of all these n-grams. After training the Neural Network, we will have word embeddings for all the n-grams in the training dataset. Rare words can now be properly represented since it is highly likely that some of their n-grams also appears in other words. FastText permits both supervised and unsupervised representations of words and sentences, which can be used for different applications, from data compression to initializers for transfer learning.

Another usage of fastText is related to text classification. We have used the supervised version on metadata records, where each text paragraph in the training file is connected to one or more labels, representing a category.

3.3.1.1 Results

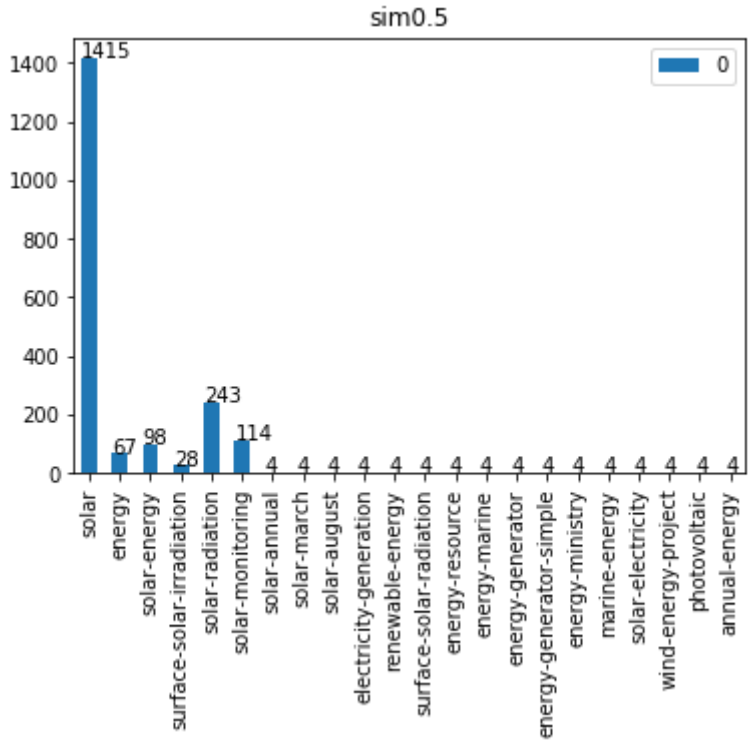
Based on the categories obtained above, we extracted for each metadata record the best label, computed by the neural network.

We took all the metadata records in the webservice-energy catalog and run against a search query defined by “solar energy”. In this case, all the keywords in the list should relate to the query by a certain level of similarity.

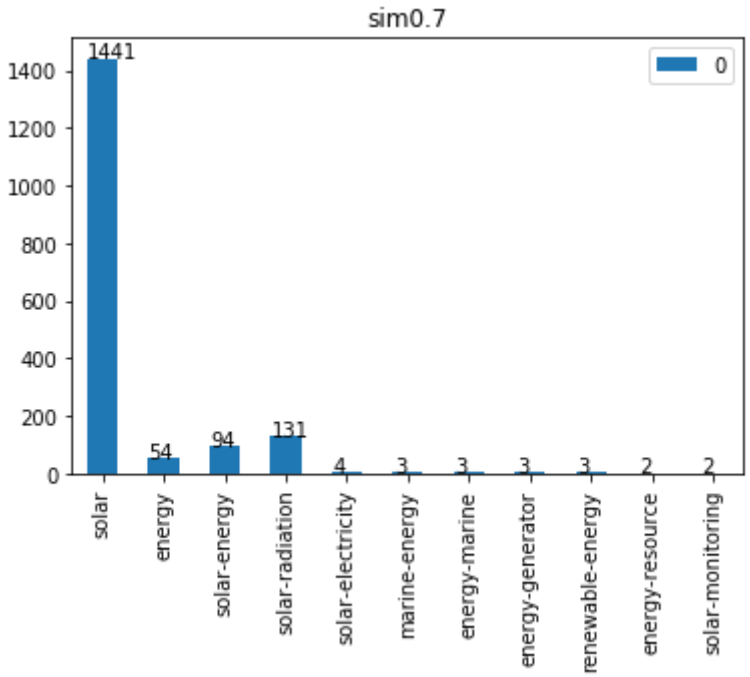
We tested the list of keywords, starting from a similarity index of 0.5 and going up to the point where all the entries are attached to a single label.

For example, in the webservice-energy catalogue, the number of results per keywords is the following, taking the consideration the similarity between the search query and the list of the best keywords. The similarities are computed using a cosine distance, so the results will be in the interval [-1, 1]. The higher the similarity, the more connected to words are as synonyms.

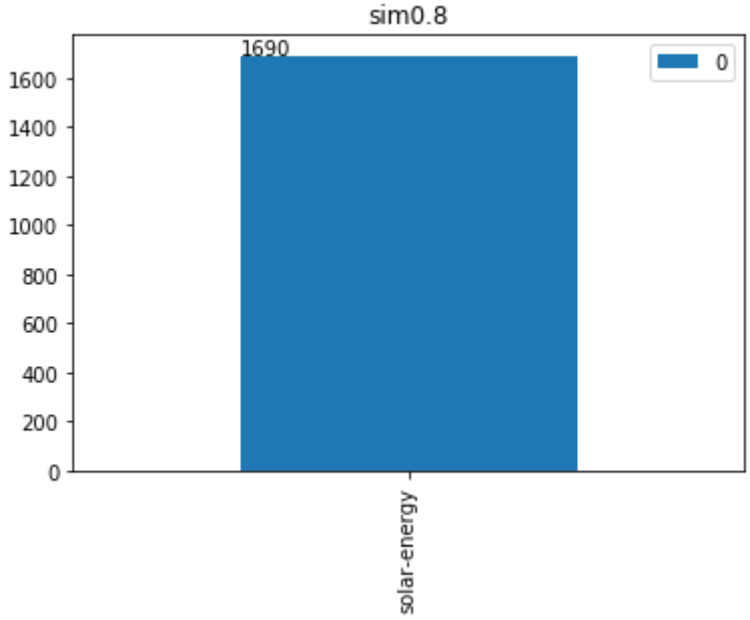
For a similarity of 0.5 we have the following distribution, meaning that 1415 records correspond to “solar” keywords, 243 to “solar radiation”, 3 to “photovoltaic” and so on.



For a similarity of 0.7, we have the distribution below. What can be observed is that the list of keywords has shortened, since they are considered too far from the search query in the vector space.



In case of the similarity set to 0.8, the only entry in the list is “solar energy” and all the records are attached to it.



3.3.1.2 Conclusions on the approach

The main advantage of this approach is that it can be applied automatically in any field, if the information has keywords attached. Building connections between keywords require a preprocessed language model in the field, but, if it is there, the algorithm can be applied easily for any type of data.

On the other hand, this approach depends on the quality of the keywords. If they are too general or missing, the quality of the results will be affected. The common issues related to this is that sometimes, the keywords are acronyms that must be transformed in proper words, or instead of separate words there are concatenations, which cannot be translated into English automatically. Also, multiple levels of similarities must be tested, together with the approach of having one or multiple categories attached to each record.

3.3.2 Building a supervised classifier, using an expert point of view

In case of short texts, such as metadata records, the best approach is to build up a hierarchy of predefined words related to the topic and automatically assign each text to these categories. In this case, an expert view is needed, in order to provide the necessary vocabulary and the hierarchy.

In this case the tree was built from general to more specific information related to energy, starting by the most general level, there is “Theme” →”Domain” → “Sub-domain” → “Information” → “Measures”.

At the “Theme” level, we have “Energy”.
At the “Domain” level, there is “Renewable Energy”.
At the “Sub-domain” level, there is “Solar Energy”.
At the “Information” level, there are “Solar resource”, “Atmosphere and meteorology”, “Ground topography”, “Meteorological Year” and “Solar potential”, together with the most prominent terms of each category.
Each “Measures” entry is linked to each category above and provides further information on the technical vocabulary for each case. Words such as “time sampling”, “time resolution”, “numerical weather prediction model” are considered relevant for this level.

At this point, we are using the data of the "Information" layer. As future work, measures should also be added in the vocabulary.

For example, words like the ones in the following table are considered relevant from an expert's point of view.

Important characteristics	Acronyms
time sampling ; time resolution ; temporal resolution ; temporal average ; time period ; time coverage ; temporal coverage ; spatial resolution ; uncertainty ; model ; model-based ; satellite ; satellite-based ; numerical weather prediction model ; in-situ ; ground ; ground-based ; pyranometer ; pyranometric sensor ; pyrheliometer ; shaded pyranometer ; thermopile, photopile ; rotating shadow irradiometer ; quality check*[,ed] ; quality control*[,ed] ;	UV = ultraviolet; IR = infrared ; TIR = thermal infrared ; GHI = global horizontal irrad*[iance, iation] ; DHI = diffuse horizontal irrad*[iance, iation] ; DNI = direct normal irrad*[iance, iation] ; BNI = beam normal irrad*[iance, iation] ; GTI = global tilted irrad*[iance, iation] ; PoA = Plane of Array ; KT = clearness index ; KC = clearsky index ; KD = diffuse fraction ; QC = quality *[control, check] ; NWP = numerical weather prediction
time sampling ; time resolution ; temporal resolution ; temporal average ; time period ; time coverage ; temporal coverage ; spatial resolution ; uncertainty ; model ; model-based ; satellite ; satellite-based ; numerical weather prediction model ; in-situ ; ground ; ground-based ; quality check*[,ed] ; quality control*[,ed] ;	WV = water vapor ; OZ = ozone ; O3 = ozone ; RH = relative humidity ; AOD = aerosol optical depth ; AOT = aerosol optical thickness ; SSA = single scattering albedo ; CBH = cloud base height ; CTH = cloud top height ; CMV = cloud motion vector ;

The approach in this case is the following:

- Run an unsupervised classifier for short texts to obtain several topics.
- Build a similarity matrix between each set of expert labels and the obtained topics.
- Add the similarities between each text and its topic to the matrix.
- Add information from words not considered by the automatic topics.
- For each topic, arrange the results in descending order, based on the similarity.

3.3.2.1 Unsupervised classification

In this case, LSA cannot find sets of different concepts for short texts, where the words related to the topic can occur only once or twice in the text. Generally, the technical words are not used often in the same paragraph and they are usually ignored by the LSA algorithm. Even if the stop-words are removed. there are still English words in text that occur more often. Even if the unsupervised classifier doesn't bring the best results, it is used as an intermediate step to get the final similarity. Beside the topics, it also returns a matrix of similarity between each document and each topic.

Considering:
- D = total number of documents in corpus
- T_1 = total number of automatic topics

The number of topics is set to a predefined number, but the algorithm may find a lower number and return the last topics empty.

T = the number of topics obtained as a result, it may be T_1 or less

The results to be kept are the top words for each topic and the matrix of similarity between the documents and the topics of size D x T.

3.3.2.2 Building similarity matrices between topics and predefined labels

Considering the following definitions:
A topic t with n_t words $t = \{w_1^t, \dots, w_{n_t}^t\}$
A label l with n_l words $l = \{w_1^l, \dots, w_{n_l}^l\}$
Let us note $F_s(w, w')$ the similarity between the words w and w'.
For cosine similarity between these two words, first a transformation T is needed from a word w to a $n \times 1$ vector $T(w)$. Then
$F_s(w, w') = \frac{\langle T(w), T(w') \rangle}{\|T(w)\| \|T(w')\|}$
Where $\langle u, v \rangle$ is the scalar product between u and v and $\|u\| = \langle u, u \rangle^{0.5}$ is the norm of u.
For one given topic t and a given label l, a $n_t \times n_l$ matrix M of similarity is first computed:
$M = \left[F_s\left(w_i^t, w_j^l\right)\right]_{i,j}$
For the topic t, the $n_t \times 1$ vector of best similarity among all labels is then determined:
$v = \left[\max_j\left(M_{i,j}\right)\right]_i$
From this vector v, the similarity $S(t, l)$ between the topic t and the label l is finally computed as:
$S(t, l) = \frac{\|v\|}{n_t}$
In case of T topics and L labels, a $T \times L$ similarity matrix $S_{T,L}$ is then defined:
$S_{T,L} = [S(t, l)]_{t,l}$

Apart from this, it is necessary to tune the algorithm for better results. For an expert, it is possible to categorize an entry to a certain topic by looking only at the title. Based on this, we can add a weight to the words in the title, based on the topics defined by the experts. Normalizing the total values is important since we are comparing sentences and titles of different length.

3.3.2.3 Building similarity matrices between topics and predefined labels

For each document, we have a list of similarities to each automatic topic, normalized and obtained by the LSA algorithm. The property of each document, the sum of probabilities in 1.

In order to compute the similarity between document and pre-defined label, we multiply the similarity matrix of the first step with the one already generated by the LSA.

3.3.3.4 Add information from words not considered by the automatic topics

The main issue related to automatic topics is that they do not take technical words into consideration, since they don't appear often in the text and so, they are not in the list of relevant words. In order to avoid these issues, we take the list of words at the "Information" level and keep the ones that are not in the list of relevant words for the automatic labeling at the first step.

At this step, we are interested in finding out the similarity between the metadata records and the words from the labels provided by the expert that have not been considered relevant by the automatic unsupervised classifier. So, we took the list of labels, the five categories that we already used and removed the words that were already found by the unsupervised classifier and used at the first step. Then, we apply the same steps at the subchapter 5.2.1.2 and we are looking for similarities between the document corpus and the filtered lists of words in the labels. We obtain a second matrix of similarity between each metadata record in the corpus and each label. This will have a small impact on the results above, since the length of text is higher and the similarities are lower. Taking rare words into consideration, such as specialized technical data, they will be found only few times in the corpus

and will have higher distances from commonly used terms. On the other hand, they are important words from an expert's point of view and cannot be ignored.

3.3.2.5 Add title weights

Titles are also important for the classification process, since for an expert is sometimes easy to categorize a metadata record by simply looking at its titles and the upper layers of classification. So, the first step into building a classification project is to use the words from the most general fields in the expert's hierarchy and to compute similarities between them and the words in the title. In this case, keywords are also relevant since they may reveal some strong connection to a certain topic.

3.3.2.6 Results

We tested the above algorithm to get entries from the following topics: solar resource, c, ground topography, meteorological year and solar potential.

For each topic, the entries having the highest scores are the following:

1. Solar potential
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7523&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=848&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7566&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=857&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7526&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7424&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7429&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7481&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7712&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7634&currTab=simple
2. Atmosphere and meteorology
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4081&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6919&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4307&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4297&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4302&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4305&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4301&currTab=simple

- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4303&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4290&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4287&currTab=simple

3. Ground topography
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=1242&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6751&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4062&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4212&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4085&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4102&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4103&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4121&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4220&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4214&currTab=simple
4. Meteorological year
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7245&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7264&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6667&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7646&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6643&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=7536&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6646&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6645&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6644&currTab=simple
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6787&currTab=simple
5. Solar potential
    - http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4107&currTab=simple

- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=6752&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4401&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4662&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4658&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4162&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4657&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4245&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4388&currTab=simple
- http://geocatalog.webservice-energy.org/geonetwork/srv/eng/metadata.show?id=4797&currTab=simple

3.3.2.7 Conclusions on the approach

The best results are obtained for the solar related entries, where the number of words in the records related to the subject is higher. On the other hand, when it comes to complementary entries, where there are only few words, the results are slightly worse. Some words which are not relevant for the subject may influence the results and there are also some words such as "field" with multiple meaning, that may influence the final classification results. Not taking into consideration the meaning of the words may lead to slightly poor results, especially in case of short texts.

3.3.3 K-means unsupervised classification

K-means clustering is one of the basic and generic unsupervised classification algorithms of elements in a vectorial space. Its objective is to group similar data points together (clusters), by following patterns. The first step is to define a number *k*, which represents the number of clusters that will be formed. The center of each cluster is called "centroid". When analyzing data, the algorithm identifies *k* centroids and allocates each data point to the nearest cluster.

At the beginning, the algorithm starts with a first group of random centroids, and then it performs repetitive calculations to optimize their positions. It stops whether the pre-set number of iterations has been reached or if there is no change in the positions of the centers.

In case of natural language processing, word2vec embeddings are used in order to determine first the position of each word in the multidimensional space.

The algorithm consists of several steps. Firstly, the corpus is transformed into array of real numbers and data is positioned in a multi-dimensional space. These dimensions can go from 2, in case of a small amount of data to hundreds, in case of a large corpus. Then, two phases are repeated for a fixed number of iterations: cluster assignment and centroid move. The algorithm goes through each of the data points and depending on which cluster is closer, it assigns the data point to it. It then calculates the average of all the points in a cluster and moves the centroid to that average location.

For the implementation, we used the Gensim library in Python 3.6.

The preprocessed corpus is set as input for a word2vec implementation that transforms the words into vectors, so that the distance or similarity between the words can be computed. Then the K-Means implementation is tested with a . The basic idea behind partitioning methods, such as k-means

clustering, is to define clusters such that the total intra-cluster variation [or total within-cluster sum of square (WSS)] is minimized [6].
In order to get the optimal number of centroids, there are several methods that can be tested. The first one is the elbow method. This method consists in looking at the total WSS as a function of the number of clusters. The algorithm should be run for different numbers of clusters (for instance, having a varying k from 1 to 20), compute the total within-cluster sum of square for each k and plot the curve of WSS according to the number of clusters. When visualizing the plot, there is a bend, which is considered as an indicator of the appropriate number of clusters.
Another solution is running a silhouette test. Briefly, it measures the quality of a clustering. That means that it determines how well each object lies within its cluster. A high average silhouette width indicates a good clustering.

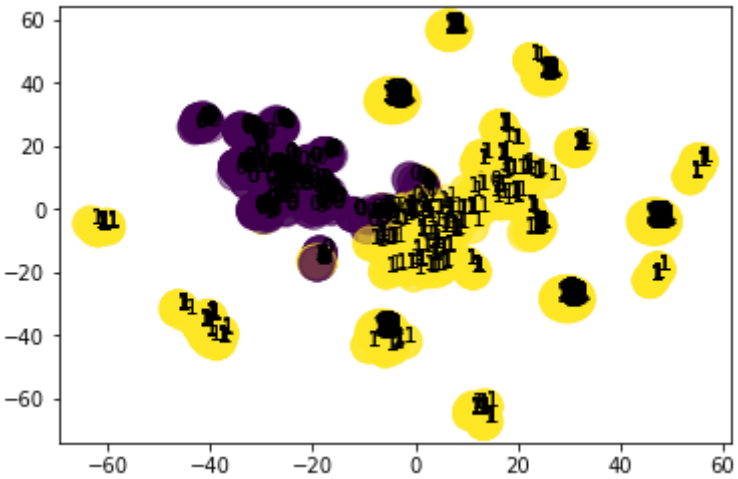
After the optimal number of topics in obtained, we also assess the way the metadata records belong to each cluster in terms of similitude, coherency and effectiveness.

As a comparison test, we also tried a hierarchical approach of running K-Means, by building a binary tree of categories. Starting from the entire corpus, we divided it into 2 clusters, then, for each of them, we kept dividing again into subcategories. Each time, the prominent words were set by using LSA.

3.3.3.1 Results
In case of the hierarchical approach, firstly, the corpus was divided into 2 clusters:

Cluster 0
- number of metadata records: 444 documents
- most important words that determine the topic:
['source', 'spatial', 'resolution', 'description', 'layer', ‘unknown', 'line', 'power', 'transmission', 'boundary']

Cluster 1
- number of metadata records: 1246 documents
- most important words that determine the topic:
['solar', 'monthly', 'energy', 'data', 'model', 'institute', 'average', 'dni', 'horizontal', 'available']



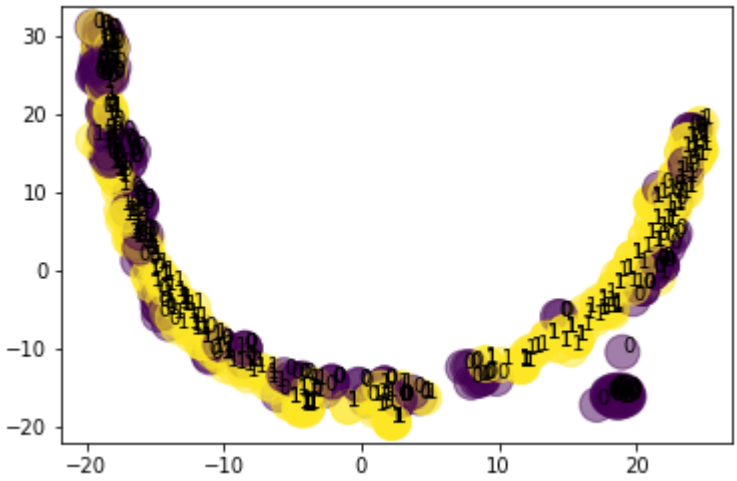
Then, selecting the first cluster, will lead to new classification of the 444 documents:

Cluster 0:
- number of metadata records: 182 documents
- most important words that determine the topic:
['source', 'resolution', 'spatial', 'line', 'description', 'transmission', 'layer', 'power', 'data', 'resource']

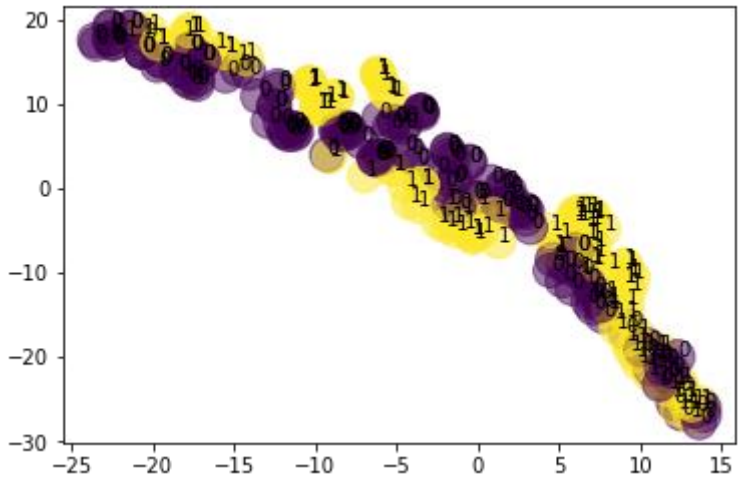
Cluster 1:

- number of metadata records: 262 documents
- most important words that determine the topic:
['source', 'spatial', 'resolution', 'description', 'layer', 'unknown', 'boundary', 'point', 'area', 'power']

The K-Means algorithm returns the following graphical results:



Continuing the hierarchy, we can select the second cluster and we obtain the below representation, with the following keywords:



Cluster 0

- number of metadata records: 141 documents
- most important words that determine the topic:
[('layer', 0.06738688), ('spatial', 0.06738688), ('description', 0.06687246), ('source', 0.06687246), ('resolution', 0.06635806), ('boundary', 0.047325112), ('unknown', 0.030349774), ('country', 0.025205757), ('river', 0.024691353), ia('administrative', 0.024176959)]
Cluster 1

- number of metadata records: 121 documents
- most important words that determine the topic:

[('source', 0.05332084), ('resolution', 0.05285316), ('description', 0.052385416), ('spatial', 0.0523854), ('layer', 0.045369476), ('power', 0.035079516), ('plant', 0.034611803), ('unknown', 0.030402238), ('area', 0.025724981), ('protect', 0.023854073)]

In case of the first approach, with the optimal numbers of clusters, we get to 7 as the best solution.

3.3.3.2 Conclusions on the approach

The algorithm can be easily used with minimal knowledge on the field. It implies only preprocessing the text and using a K-Means library. It can be configured to determine the optimal number of clusters and it can group the database entries in multiple clusters. For better results, it requires a large dataset, so the word2vec implementation can get connections between words.
The main drawback of this solution is that it cannot build a topic per cluster, so another algorithm should be used to get the most important words. In this case, we used, for each cluster, an LSA implementation for topic modeling. For each cluster, we gathered the list of metadata records belonging to them and ran LSA with the number of topics set to 1, in order to get the most important words, the ones that define the cluster.

3.3.4 Latent Semantic Analysis

Topic modeling is a text mining technique which provides methods for identifying co-occurring keywords to summarize large collections of textual information. In the natural language processing field, one of the most common techniques for automatically obtaining topics is Latent Semantic Analysis (LSA). It builds relationships between a set of documents and the terms they contain by producing a set of concepts related to the documents and terms. As input, it received a corpus of documents and returns several different topics from it, together with the most prominent words of each. It uses Bag of Word (BoW) model. In this case, the text is represented as the multiset of its words, without taking into consideration links or order of them in the sentence, only the number of occurrences of each word. This results in a term-document matrix (occurrence of terms in a document). It builds a term frequency-inverse document frequency (tf-idf) where each position in the vector corresponds to a different word and a document is represented by the number of times each word appears. So, the most important words will be the ones that appear the most often in the documents. The LSA algorithms improve the process by also considering synonymy between words. It learns latent topics by performing a matrix decomposition on the document-term matrix using Singular value decomposition (SVD). This is a matrix factorization method that represents a matrix in the product of two matrices.

Since LSA is already used to determine the main topics obtained when clustering using K-Means for each group of documents. Another approach was to run the algorithm on the entire database of metadata records, in order to obtain the topics. Their optimal number was obtained from the silhouette test of K-Means clustering.

3.3.4.1 Conclusions on the approach

LSA is a state-of-the-art method for topic retrieval, but in case of short text, related to the same theme, it is hard to identify distinct information of subclasses. Taking the examples of concept like “solar”, “irradiation”, “wind”, they can be found in the same text in case of digital elevation models, but also in different records in case of more specific web services. “Bag of Words” is also not the best approach when it comes to metadata records on web services, since the technical words that define a subject may not appear often in the text.

4. Conclusion

Analyzing the results above, the best approach related to clustering metadata records of solar observations is a hybrid approach including a mix of manual labels and machine learning approaches. In this case, a list of predefined labels is necessary, since the themes, domains or requested information are difficult to be discovered automatically. The main challenge of classifying the entries is related to the length of the texts and the technical concepts. It is highly important to know beforehand what is relevant from an expert's perspective and the machine learning algorithm can provide a new insight over text similarities and interesting results. We can get metadata records that are not automatically returned by the database when querying. We can use a ranking system, based on the search text, so the most relevant results will be shown first, for a better user experience. Thanks to NLP and ML algorithm, we can connect in the future resources from multiple sources, so gathering information in a field will be easily obtained from a single entry point.

5. Recommendations and perspectives

In the context of the GEOSS Knowledge Hub, the ideal context would be for a user to easily get different type of results, when looking for something. The main purpose is to retrieve resources from multiple sources, to be able to automatically connect research papers and metadata records obtained from the same organizations or related to the same places and the same period of time. Also, a "smart" knowledge hub can automatically compute similarities between articles, to facilitate research on a field or to easily gather data for a single area. The existing algorithm can provide complex connection between distinct resources, to provide a better user experience for both experts and new users.

6. Bibligraphy

[1] https://www.earthobservations.org/geoss.php
[2] Mikolov, Tomas, et al. "Distributed representations of words and phrases and their compositionality." *Advances in neural information processing systems*. 2013.
[3] https://radimrehurek.com/gensim/models/word2vec.html
[4] https://fasttext.cc/
[5] https://www.datanovia.com/en/lessons/determining-the-optimal-number-of-clusters-3-must-know-methods/
[Figure 1]: Image taken from http://mccormickml.com/2016/04/19/word2vec-tutorial-the-skip-gram-model/
[6] http://www.oie.mines-paristech.fr/Accueil/
[7]https://towardsdatascience.com/pre-processing-in-natural-language-machine-learning-898a84b8bd47

List of terms:
Label: In case of a label provided by an expert, it is a sequence of words that defines a category. In case of supervised classification, such as fastText, a label is the category the text belongs to
Corpus: the content of a database used as input for a machine earning algorithm. In this case, it can be the content of the metadata records of the webservice-energy for the classification part or the list of texts obtained from Zenodo when building the language model
Topic: a list of words, obtained automatically, that summarize short texts.