KnowML

Knowledge Discovery
through
Machine Learning

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

Contents

Setting the stage

In their recent book “Rebooting AI: Building Artificial Intelligence we can trust,” Gary Marcus and Ernest Davis express what many researchers and practitioners of AI already realized: that *“... something fundamental is still missing”*. Despite progress in hardware and software and the sheer amount of data we can use and analyze, *“what we have for now are basically digital idiot savants.”*

According to the authors: *“What’s missing from AI today—and likely to stay missing, until and unless the field takes a fresh approach—is broad (or “general”) intelligence.”*
The authors then make it clear: *“To make progress, we need two things to get started: an inventory of what kind of knowledge a general intelligence should have, and an understanding of how this knowledge would be represented, clearly and unambiguously in a self-contained fashion, inside a machine.”*

We do not pretend to have found the ultimate solution to this problem, as it exists on the fundamental level of general intelligence and its absence from today’s AI/ML applications. However, we believe we have made a small step toward that goal. This small step is the orchestration of the interplay between Machine Learning techniques and the methods of Symbolic Knowledge Representation. This orchestration assumes, firstly, a provision of some amount of initial knowledge that helps AI to start its activity in a more meaningful way. Secondly – it assumes a proposition of a candidate for the representation of knowledge that is the output of the AI activity.

This whitepaper describes what we call above as the first “small step.” We focus here on the well-defined and restricted domain of financial regulations and their expression in legal texts that form a corpus of documents we have chosen for our analysis.

Challenge

The methodology described in this paper has been invented and implemented by MakoLab R&D team to address the fundamental challenge of practical knowledge management, which can be formulated as the following postulation:

Given a corpus of documents, extract a body of structured knowledge that these documents implicitly contain.

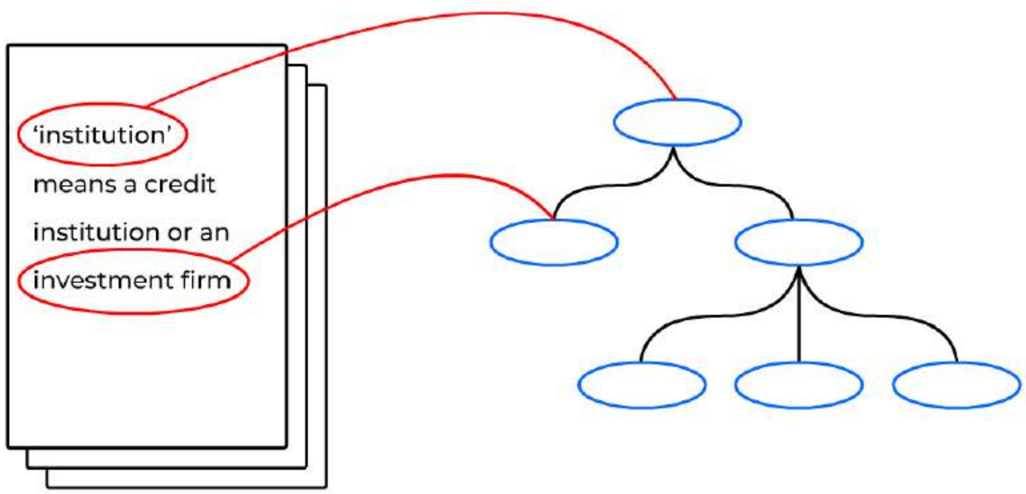


Figure 1 Challenge

By 'knowledge' we understand here an informational artefact that represents a network of concepts interconnected by domain-specific relations. A concept is understood here as a term with its definition whose specific content is co-defined by its role in such network. To be more specific, the extracted knowledge will be materialized as an OWL ontology as instances of `skos:Concept` class.

The methodology is now being used in the process of creation of the suite of products under an umbrella name: KnowML.

Our approach

To meet the challenge we propose to orchestrate an interplay between machine learning techniques and methods of symbolic knowledge representation. The latter provide us with ontologies, i.e., a formal representations of knowledge, which serve twofold purpose:

1. Identify concepts to be found in documents
2. Provide training data for machine learning driven recognition of concepts' definitions

On the other hand, Machine Learning algorithms:

3. Recognize new definitions and concepts
4. Generate structured data to the next step of ontology development.

As such, this interplay is an iterative process in which the last step of one cycle leads to the first step in the next one.

Obviously, the task at stake involves also more mundane goals, e.g., content extraction from the documents in scope (see: "Document content extraction" section)

Our approach at work

In order to show how our methodology works in practice we applied it to a set of documents (approx. 500) that specify legal measures related to the domain of financial reporting regulations in the European Union – a sample thereof is shown in Table 1. This set will serve here as an illustration of our approach.

Reporting framework	Type	Title	URL
BRRD	Directive	Directive 2014/59/EU of the European Parliament and of the Council of 15 May 2014 establishing a framework for the recovery and resolution of credit institutions and investment firms and amending Council Directive 82/891/EEC, and Directives 2001/24/EC, 2002/47/EC, 2004/25/EC, 2005/56/EC, 2007/36/EC, 2011/35/EU, 2012/30/EU and 2013/36/EU, and Regulations (EU) No 1093/2010 and (EU) No 648/2012, of the European Parliament and of the Council	https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex:32014L0059
BRRD	Delegated Act	COMMISSION DELEGATED REGULATION (EU) …/... of 18.3.2016 on classes of arrangements to be protected in a partial property transfer under Article 76 of Directive 2014/59/EU of the European Parliament and of the Council	https://ec.europa.eu/transparency/regdoc/rep/3/2016/EN/3-2016-1372-EN-F1-1.PDF
BRRD	Delegated Act	Commission Delegated Regulation (EU) 2015/63 of 21 October 2014 supplementing Directive 2014/59/EU of the European Parliament and of the Council with regard to ex ante contributions to resolution financing arrangements	https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=CELEX:32015R0063

Table 1 Exemplary documents in scope

Document content extraction

This process extracts textual contents from the documents in scope and partitions them into chunks that are based on the explicit document structure (provided that the documents have such identifiable structure.) This explicit structure may involve:

- documents' paragraphs
- documents' articles
- documents' chapters
- etc.

HTML documents

HTML documents are usually structured by means of the HTML tags, in particular by means of the paragraph tag '<p>'.

PDF documents

A file of PDF format can be parsed by various kinds of tools, each of which has its rendering technique. PDF formatting is also human dependent and may have a different organization of paragraphs. In general, in the structure of a given PDF document, the pages (labelled by page indexes) and the paragraphs (text sequences separated from each other) have been identified.

Storage

We store the textual contents of the processed documents in SOLR cores, which were later aggregated into a single common core for all documents' paragraphs.

The Apache SOLR is an open-source enterprise-search platform, written in Java, from the Apache Lucene project. Its major features include full-text search, hit highlighting, faceted search, real-time indexing, dynamic clustering, database integration, NoSQL features, and rich document (e.g., Word, PDF) handling.[1] SOLR stores indexed contents of its documents in cores (i.e. a running instance of an index that contains all the SOLR configuration files required).

An example of a paragraph from a document and its SOLR representation is presented below:

[1] https://lucene.apache.org/solr/guide/8_0/

```
<p>

Directive 2014/59/EU of the European Parliament and of the Council of 15 May 2014 establishing a framework for the recovery and resolution of credit institutions and investment firms and amending Council Directive 82/891/EEC, and Directives 2001/24/EC, 2002/47/EC, 2004/25/EC, 2005/56/EC, 2007/36/EC, 2011/35/EU, 2012/30/EU and 2013/36/EU, and Regulations (EU) No1093/2010 and (EU) No 648/2012, of the European Parliament and of the Council Text with EEA relevance<br/>

</p>
```

Source paragraph example

```
{

"id":"f119b168-fa8e-4972-b906-56b0c92cc6a3",

"document_id":"https://eur-lex.europa.eu/legal-content/EN/ALL/?uri=celex:32014L0059",

"content":"Directive 2014/59/EU of the European Parliament and of the Council of 15 May 2014 establishing a framework for the recovery and resolution of credit institutions and investment firms and amending Council Directive 82/891/EEC, and Directives 2001/24/EC, 2002/47/EC, 2004/25/EC, 2005/56/EC, 2007/36/EC, 2011/35/EU, 2012/30/EU and 2013/36/EU, and Regulations (EU) No 1093/2010 and (EU) No 648/2012, of the European Parliament and of the Council Text with EEA relevance",

"paragraph_corpus_id":"2",

"embedded_doc_id":"doc_id_0",

"version":1629327257241649152

}
```

SOLR document example

Discovery of concepts

Concepts in frames

We assume that concepts occur in special semantical contexts known as frames. The notion of frame in Frame Semantics has its conceptual roots in the notion of frame introduced to Artificial Intelligence by M. Minsky. A frame is understood as a system of concepts related in such a way that to understand any one of them you have to understand the whole structure in which it fits; when one of the things in such a structure is introduced into a text, or into a conversation, all of the others are automatically made available.[2]
In our approach we use this idea at two levels:

1. We represent systems of concepts by means of Semantic Web ontologies (or RDF(S) graphs), in which concepts and their definitions are present and correlated therebetween by means of domain specific relations.[3]
   - Thereby a frame is actually an RDF(S) subgraph of the whole ontology – an example of such frame is shown in Figure 2Figure 2.

2. We represent the basic definitional structure of a definition as the frames shown in Figure 3 and Figure 4.

---

[2] Fillmore, Ch. (2006). Frame semantics. In Geeraerts, D. (ed.), Cognitive Linguistics. Basic readings. Berlin and New York: Mouton de Gruyter. p. 373.
[3] We follow here such approaches as: J. Scheffczyk et al., Ontology-Based reasoning about lexical resources, [in:] Ontologies and Lexical Resources for Natural Language Processing, Cambridge 2008; E. Ovchinnikovaet al., Data-Driven and Ontological Analysis of FrameNet for Natural Language Reasoning, [in:] Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10); N. Ide, FrameNet and Linked Data, Proceedings of Frame Semantics in NLP: A Workshop in Honor of Chuck Fillmore (1929-2014), p. 18-21; P. Hauck et al., Supporting FrameNet Project with Semantic Web technologies, ONTOBRAS 2015.

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com
MakoLab SA is a joint-stock company organized and existing under the laws of Republic of Poland with a registered share capital of 707 473,00 PLN (Polish zlotys), identified in the National Court Register (Krajowy Rejestr Sądowy) conducted by the District Court for Lodz Srodmiescie in Lodz under the number KRS: 0000289179, Tax Identification Number (NIP): PL 7250015526, National Official Business Register (REGON): 47134311.

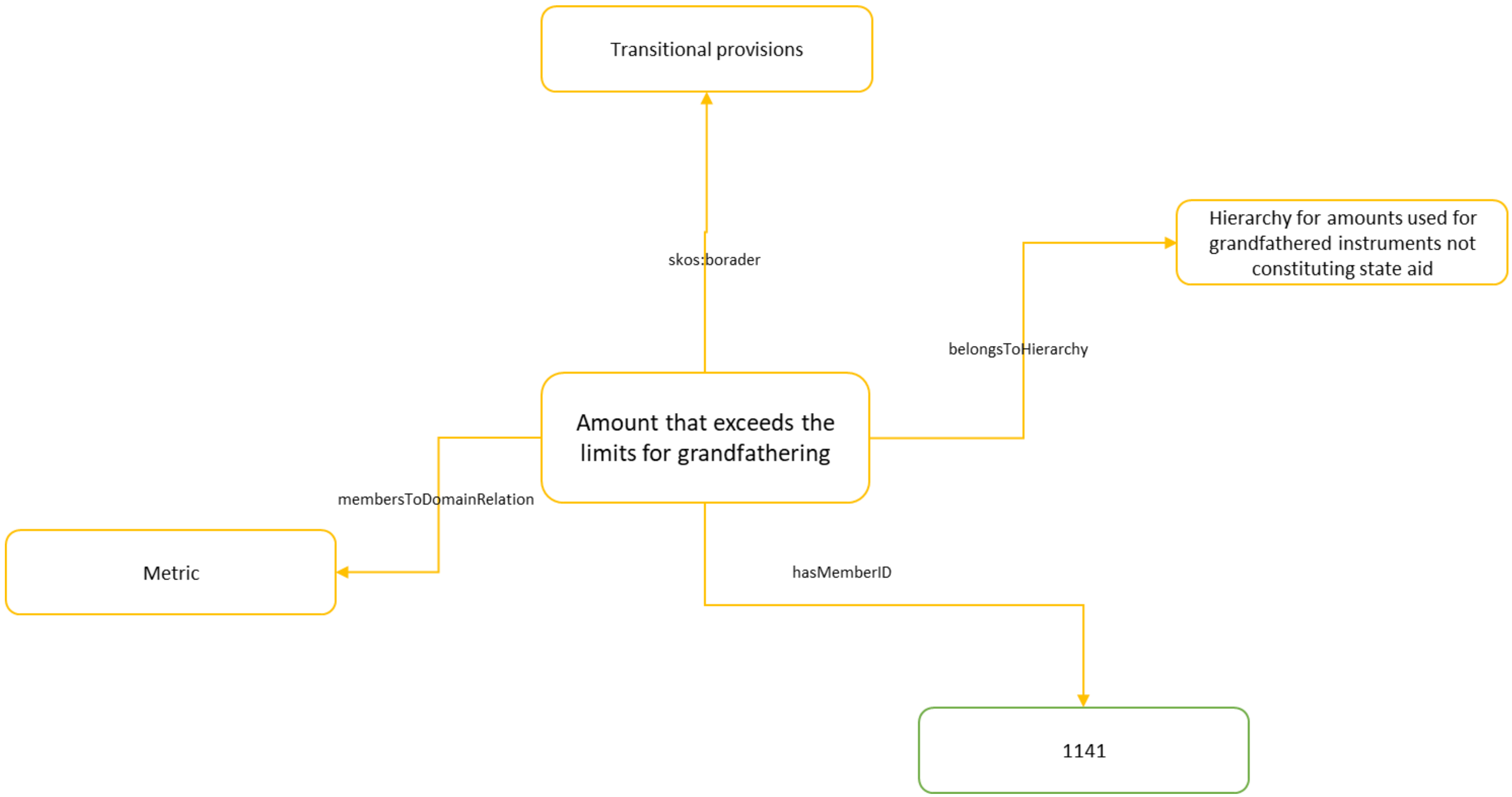


Figure 2 Frame - example

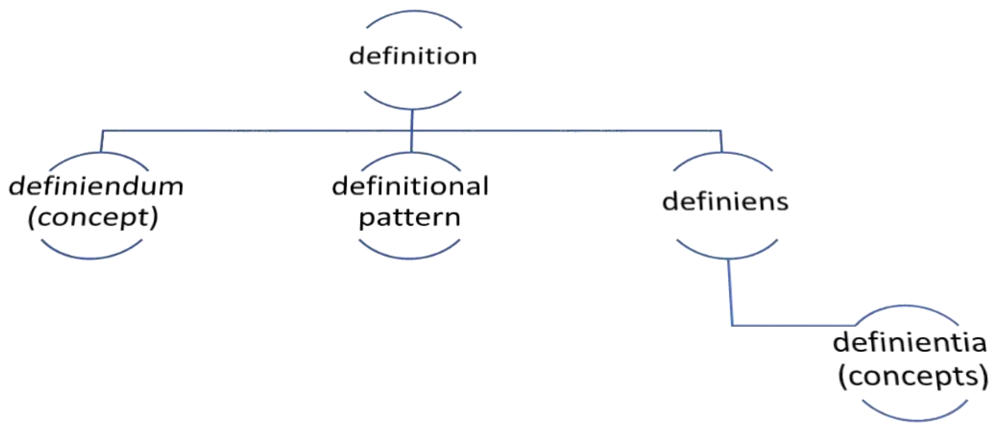


Figure 3 Frame for definitions

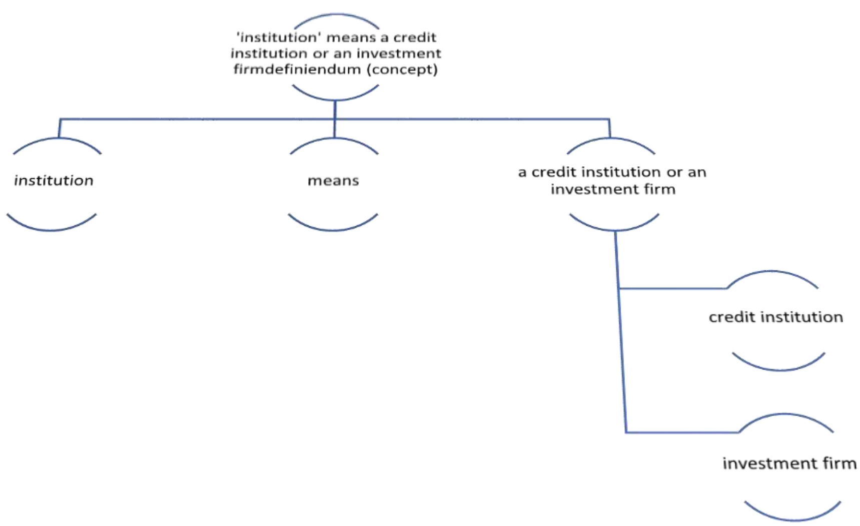


Figure 4 Frame for definitions - example

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

Our methodology requires that we concurrently develop a suite of domain specific ontologies and training datasets to machine-learning algorithms that recognize concepts and their definitions.

We will use ontologies in two capacities:

1. As the concepts' containers
2. As the sources for concept discovery.

Concept discovery in practice

In the particular case of the current example the ontology suite contains five domain ontologies:

1. EuroVoc
    a. 7372 concepts and concept schemes
    b. source: https://data.europa.eu/euodp/en/data/dataset/eurovoc
2. SKOS version of FIBO ontology
    a. 4058 concepts and 1695 object properties
    b. source: https://spec.edmcouncil.org/fibo/vocabulary
3. DPM (Data Point Model) Ontology
4. DTIF (Dictionary of Finance and Investment Terms) Ontology
5. ML (Machine Learned) Ontology

All five ontologies contain 22,461 concepts and 3879 properties in total.

EuroVoc and FIBO are open source ontologies ready to use.

DPM and DTIF ontologies were extracted from quasi-ontological sources as described below.

ML ontology was developed in an iterative process that meshes up Machine Learning methods with Symbolic Knowledge Representation approach.

DPM Ontology development

DPM ontology is based on the Data Point Model dictionary (DPM) from https://eba.europa.eu/risk-analysis-and-data/reporting-frameworks/reporting-framework-2.9

To build this ontology we have created five OWL classes representing DPM resources:

- DPM Member
- DPM Hierarchy
- DPM Domain
- DPM Dimension
- DPM Code.

DPM ontology consists of around 8,400 concepts, contains the hierarchy of concepts and relations between the concepts, domains, codes, and dimensions.

DTIF Ontology development

DTIF ontology was automatically extracted from Dictionary of Finance and Investment Terms by John Downes, Barron's Educational Series, Fifth Edition.

DTIF ontology consists of 4487 dictionary concepts. Each concept has the `skos:prefLabel` of the term that is defined (definiendum) and the `skos:definition` of the term (definiens). Some concepts are related by `skos:related` to related terms contained in the definiens. They all stand as independent ontological sources. A new ontology can be easily added to the set of ontologies provided it is expressed in SKOS ontology format. In particular, each concept should contain `skos:prefLabel` and `skos:definition` relations, and optionally `skos:broader`, `skos:narrower`, and `skos:related`.

ML Ontology development

The development of ML (Machine Learned) ontology consists of:

1. Automated extraction of definienda from a set of definitions
2. Validation of the definenda extraction
3. Creation of the ontology as an OWL file, i.e., import of the validated extractions to the OWL file

Definienda extraction

Generally speaking, the definiendum of a definition is identified as the first Part-Of-Speech (aka POS) in the definition's parsing tree, whose parts are identical to those of some ontological concept. Note that we do not require that the structure of the definiendum is the same as the structure of one of the concepts, but only that the former contains the same Parts-Of-Speech as one of the latter.

Algorithm

1. The inputs:
   a) ontologies from the ontology suite
   b) definitions – set of definitions
2. concepts – all concept labels retrieved from ontologies
3. For each concept in concepts:
   a) The parser finds its parsing tree, i.e., a part-of-speech tree.
   b) The tree is transformed into a "set_of_pos_tags_concept", i.e., the set is the collection of all POS tags from this tree
   c) This set is associated with the concept
4. For each definition in definitions:
   a) The parser finds its parsing tree and the transformation thereof into a "set_of_pos_tags_definition" (in the same way as in the case of concepts)
   b) If this "set_of_pos_tags_definition" is identical to a "set_of_pos_tag_concept" associated with some concept, then the whole definition is the definiendum (rare case!)
   c) Otherwise, all children of the parsing tree are collected and steps (a)-(c) are repeated for each child until we find the definiendum, or we reach all leaves of the tree in which case no definiendum is found.
5. The output:
   a) set of definitions associated with their definienda

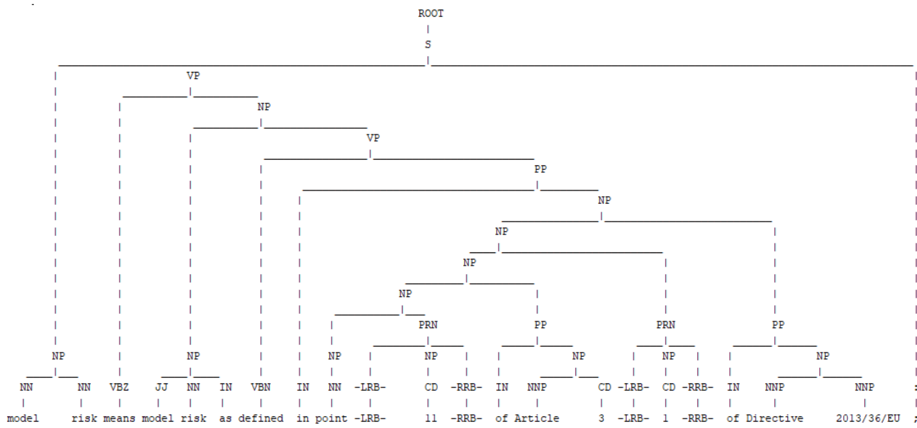
Examples

Parsing tree for the *concept* 'model risk'

```
          |
          NP
     _____|___
    NN        NN
    |         |
  model      risk
```

Parsing tree for the *definition* of 'model risk'

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com



ML ontology creation

Definienda were converted into instances of `skos:Concept`. Each concept has its `skos:prefLabel`, at least one definiens and the whole definition. The process of building the ML ontology is incremental. Each time we have a new list of definitions pre-processed as described above, we add them to the ontology.

Human-assisted ML algorithms

Training data

In general, the training data for the ML algorithm is a set of paragraphs, where each paragraph is binary classified either as a definition or as a non-definition.

Path finding

To find the training set of definitions, we decided to make use of the two sources thereof:

1. The definitions from the ontologies we have developed
2. The definitions of the relevant concepts from the documents in scope

Finding concepts in practice

In the context of our current example to get to the latter part, we initially followed the process:

1. We extracted the paragraphs from the documents' sections that are titled 'Definitions'

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

2. Browsing through these paragraphs, we found out the following patterns of the definitional verb 'mean' and the paragraphs where this verb is used in this capacity.
3. We found the following patterns:
    1. 'X' means Y
    2. X' shall mean Y
    3. 'X' means Y as defined in point Z
    4. 'X' means Y is referred to in Article Z
    5. 'X' means Y in accordance with Article Z
    6. 'X' means either of the following:
    7. 'X' means the requirements laid down in Article(s) Z
    8. 'X' means Y within the meaning of point Z

In order to find the training set of non-definitions:

1. We found a set of paragraphs that are not "mean" – definitions.
2. We took a random subset of this set so that its size is comparable to the size of the subset of definitions.

Thus, the set of non-definitions might have contained paragraphs with the word "mean" in a non-definitional role or paragraphs that happen to be definitions other than "mean" definitions described above. Then we used the training data in the algorithms described below and validated their predictions against the evaluation benchmark.

ML algorithms

To train the ML agents to recognize definitions, we tested six approaches described below, differing in the input data transformations and the machine learning models. The models tested in the experiments were built by means of three different techniques used in the present-day machine learning systems:

- Inductive learning,
- Deep learning,
- Gradient boosting.

The main goal of inductive learning is to discover rules (e.g., in the IF-THEN form) representing the knowledge hidden in data containing the training cases. The extracted rules can be used for classification of new cases. We have used three algorithms in the experiments:

1. LEM2[4]

[4] J. W. Grzymala-Busse, A New Version of the Rule Induction System LERS. Fundamenta Informaticae 31(1997): 27-39.

2. AQ[5]
3. CN2[6]

The final classification output was obtained by means of a simple voting mechanism.

Deep Learning is a specific subfield of Machine Learning. Deep Learning covers a broader family of the machine learning methods based on neural networks. In general, Deep Learning models consist of multiple successive layers to progressively extract higher level features from data consisting of training cases. We have tested the Deep Learning models based on two main architectures: Densely Connected Neural Networks (without memory) and Recurrent Neural Networks (with memory). In case of the Recurrent Neural Networks, each text sequence is processed by iterating through the sequence elements.[7]

Gradient Boosting is a machine learning technique in which a prediction model is built in the form of an ensemble of weak prediction submodels (e.g., decision trees). The weak prediction submodels are trained in a gradual, additive and sequential manner.[8]

For each machine learning model, proper input data were prepared be means of transformations (used in natural language processing) of texts into numerical vectors used in natural language processing. In the end we tested 6 algorithms described in Table 1.

[5] R.A. Michalski, R.A. Theory and methodology of inductive learning. Machine Learning 1(1983): 83-134
[6] P. Clark, T. Niblett The CN2 induction algorithm. Machine Learning 3(1989):261-283
[7] https://livebook.manning.com/book/deep-learning-with-r/chapter-1/
[8] https://www.kaggle.com/dansbecker/xgboost

Names	Type	Input data transformation	Model
Binary Vectors + XGBoost	Gradient Boosting	paragraphs were transformed into binary vectors (one-hot binary representations)	ensemble of models as decision trees implemented in XGBoost library
Sequences + Deep Learning: dense layers	Deep Learning	paragraphs were transformed into sequences of indices of words in paragraphs	neural network of one embedding layer and four dense layers implemented using Keras library
Sequences + LSTM	Deep Learning	paragraphs were transformed into sequences of indices of words in paragraphs	LSTM: recurrent neural network implemented using Keras library
TF + rules	Inductive Learning	paragraphs were transformed into numerical vectors of words' term frequency coefficients	IF-THEN rules generated using three algorithms, LEM2, AQ, and CN2, implemented in Rough Sets package
TF-IDF + deep learning: dense layers	Deep Learning	paragraphs were transformed into numerical vectors of words' the term frequency–inverse document frequency coefficients	neural network of one embedding layer and four dense layers implemented using Keras library
TF-IDF + XGBoost	Gradient Boosting	paragraphs were transformed into numerical vectors of words' the term frequency–inverse document frequency coefficients	ensemble of models as decision trees implemented in XGBoost library

Table 1 Machine learning algorithms tested

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

Concepts in action

In order to enhance search for concepts in the documents in scope, we use MakoLab Search Insights (SI) software. The SI takes as its input an ontology and SOLR documents (containing paragraphs from Euro Lex documents. In other words, it is a concept-based search engine developed by MakoLab, where the search is realized not by a simple document-to-keyword matching, but by more robust and precise content-of-the-document-to-concept matching. To obtain this level of coordination, SI first applies Natural Language Processing algorithms (lemmatization and parsing) to create an internal index of all syntactically analyzed documents. Then, SI takes advantage of the ontologies provided in the project. Once an index of documents and the ontologies are provided, SI creates a mapping between the documents and the ontological concepts. The function of the mapping is to understand the content of a document by its conceptual map.

After the user sends a query to SI, first, it is syntactically analyzed, and then, SI initiates the process of the query "understanding" by selecting the ontological concepts corresponding to the query's content. In the next step, SI applies the matching between the content-of-the-document and the selected concepts and finds documents that form a set of answers to the query.

Conclusions

We found out that the fine-tuned orchestration of symbolic AI and ML approach may produce the synergic effect in terms of the scope and quality of the informational artifacts it produces.

When deployed, the former provides high-quality training data for the latter, and the latter may extend the coverage of the symbolic representations. In order to increase this effect, the reiteration of the orchestrated process is recommended. Secondly, although most of the processing can now be automated, we confirmed the current widespread assumption to the effect that domain expert assistance is still required in AI.

We also learned that one needs to find the proper balance between the precision and recall parameters in the evaluation of ML models. Finally, we discovered that the advantages of such an approach could be fully appreciated when it is applied at the level of operational end-user application, where the symbolic representation of concepts is visualized employing knowledge graphs.

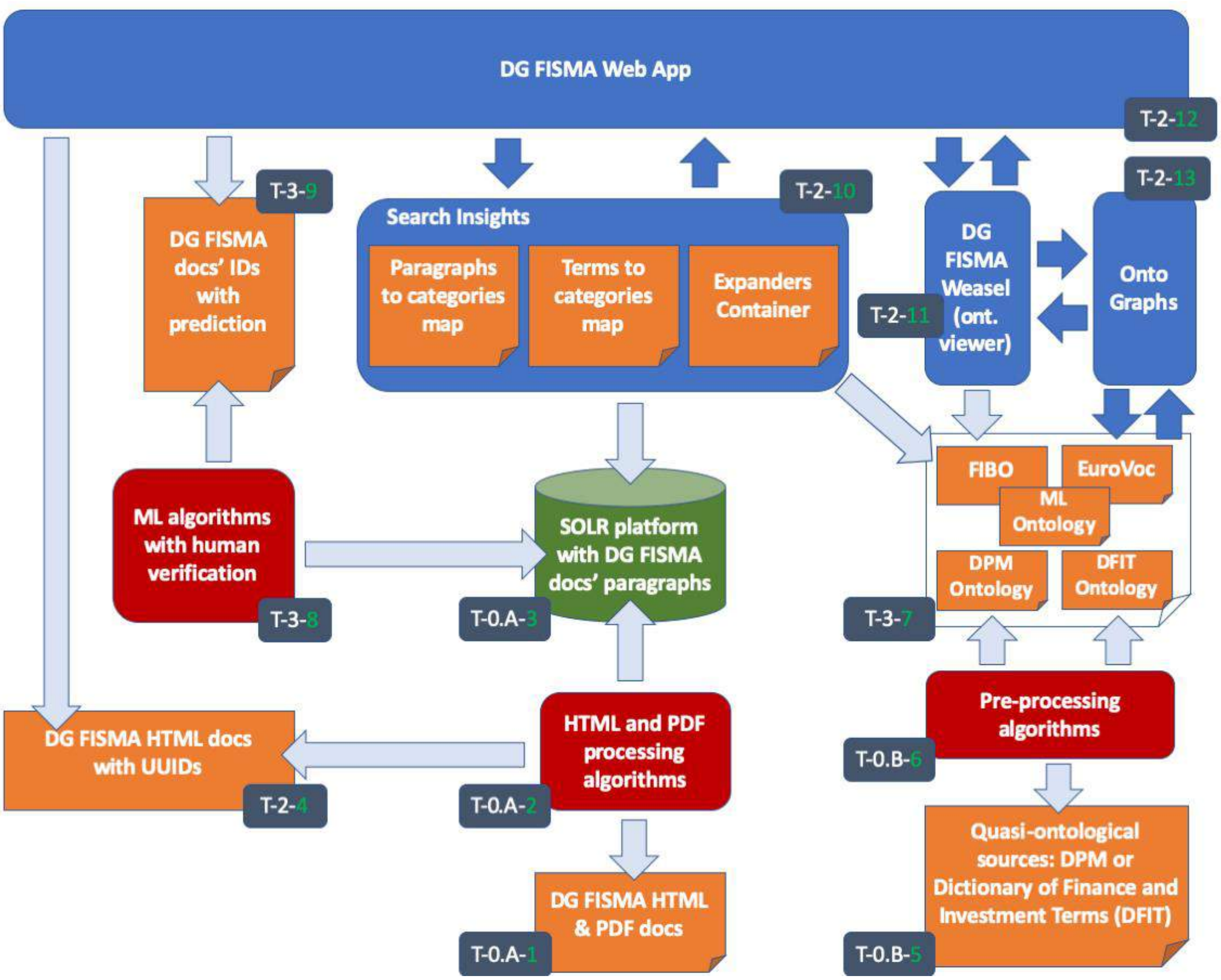
Appendices

Implementation example

For the purpose of the current example specified above we developed a web application described in this section.

System architecture

The diagram below presents the main components of the software system developed in the project.



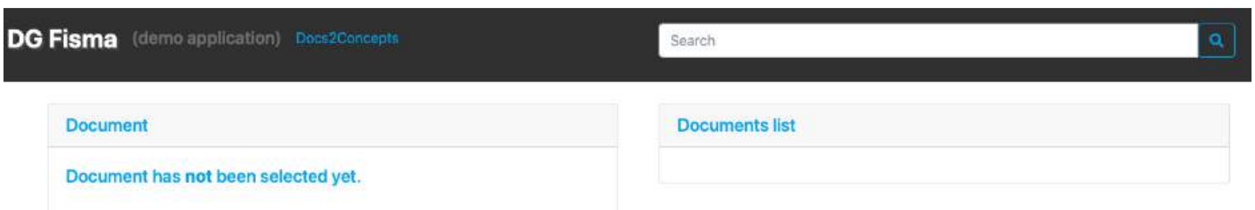
MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

To facilitate the presentation of results embedded in the application, the numerical identifiers of the system's components appear in the names of the forthcoming subsections.

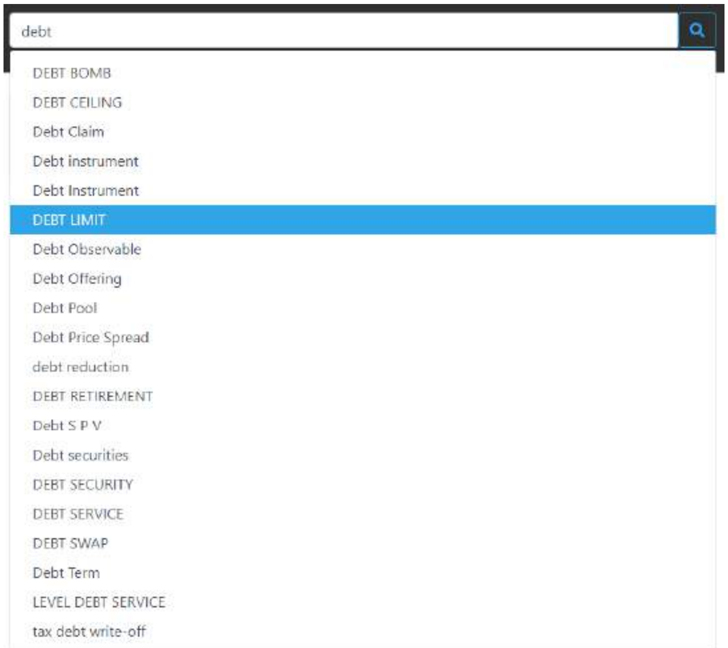
System walk-through guide

Based on the design specified above we developed a web application available at https://fisma.makodev.pl.

The welcome screen of the application is minimalistic in its design and draws user attention directly to the search on the top right-hand corner:

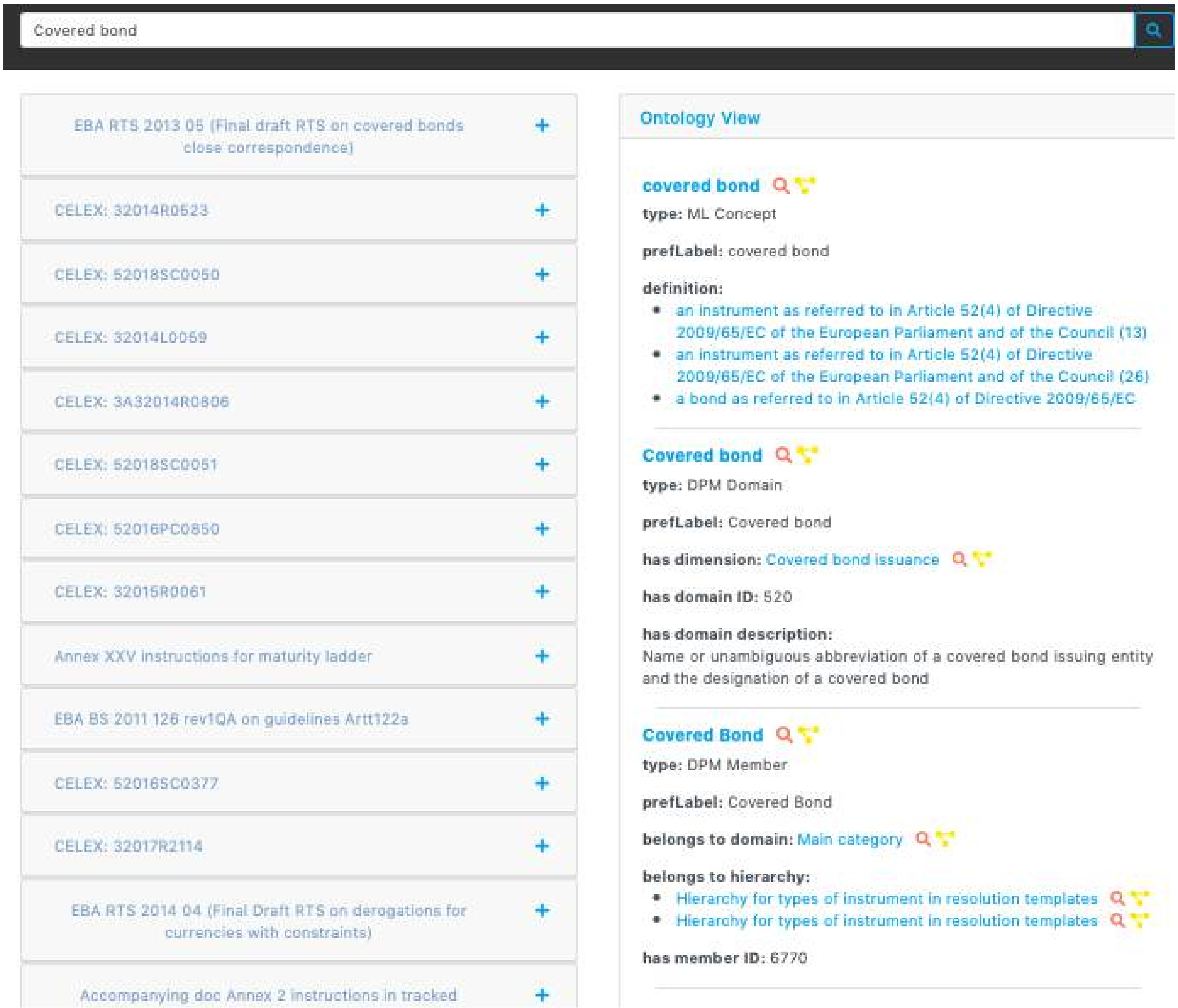


When the user starts typing the desired concept the application immediately suggests the matching concept from the list of available ones – that is from the ontologies and dictionaries. The search is a full-text one, so the application looks for a desired term in the beginning, middle or at the end of the matching phrase.



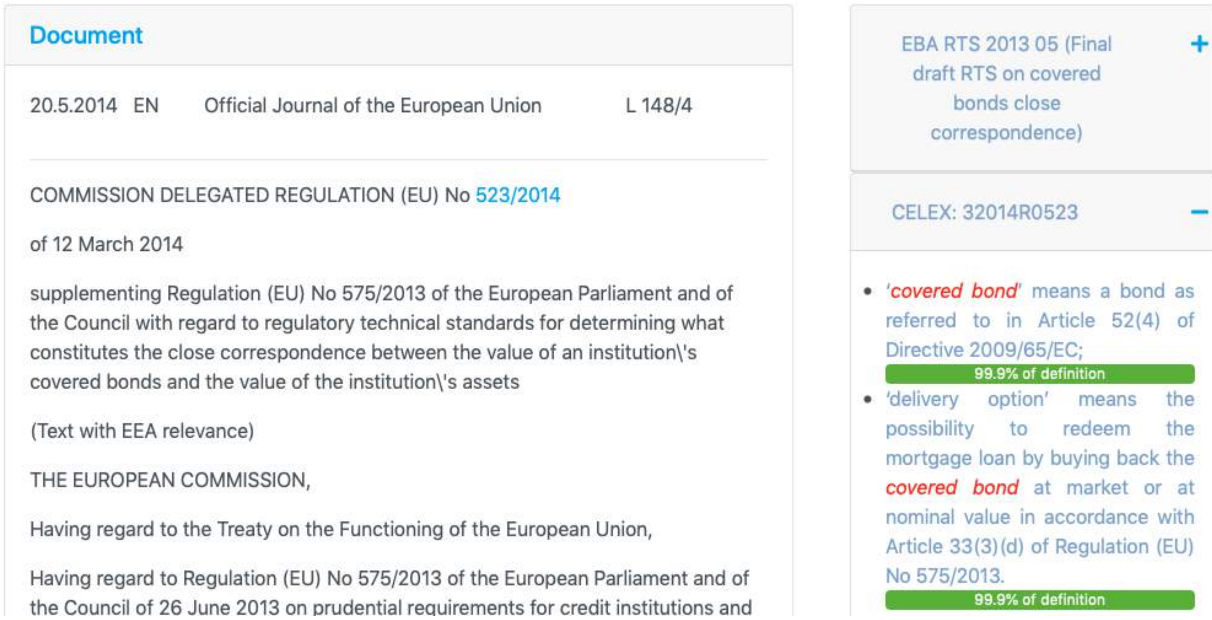
After typing or choosing the selected term, the application presents the two main extensions on the screen: the document view (on the left) and the ontology view (on the right). Those two views are separate, in a sense that they can be browsed separately.

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

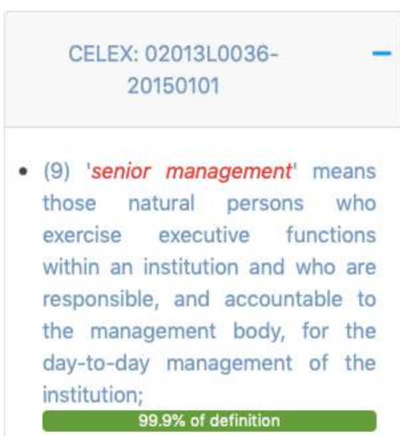


The document view presents the list of the documents the term appears in. They are sorted from the highest to the lowest certainty that a given document contains the definition. After clicking plus sign (+) or the name of the document: CELEX: 32014R0523, the whole text with its structure preserved appears on the left, while under the name of the document, the excerpt with the desired concept is shown – highlighted in red.

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com



Each bullet point represents a paragraph in which the application found the desired concept. Some of them may not be a definition in a strict sense. Certainty of particular paragraph being a definition is represented by a visual bar at the end of each paragraph (in the example above – 99%). The scale is from 0% – certainly not a definition, to 100% – certainly a definition.



Clicking at the chosen paragraph moves the user to the particular place where it appears, highlighted in green.

In the parsed PDF documents, clicking on the paragraph points to the exact place in the text, however, the text is not highlighted.

The ontology view presents the searched term in ontological manner, that is, with certain attributes connected with it, various for different ontologies.

As mentioned before the ontology view is separated from the document view, therefore, the user may browse through it without changing the previewed document. To do so, the user can click on the desired term and the application will display it.

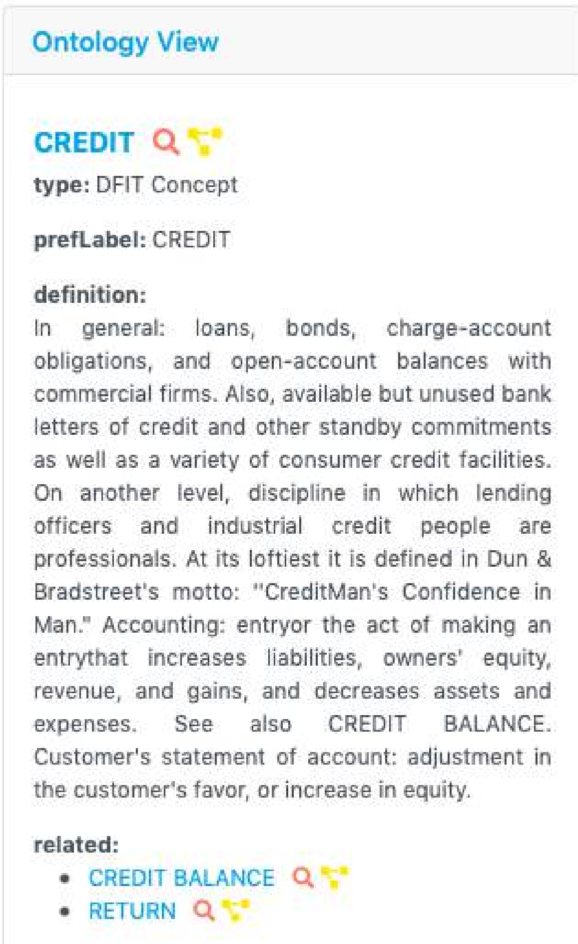
MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com

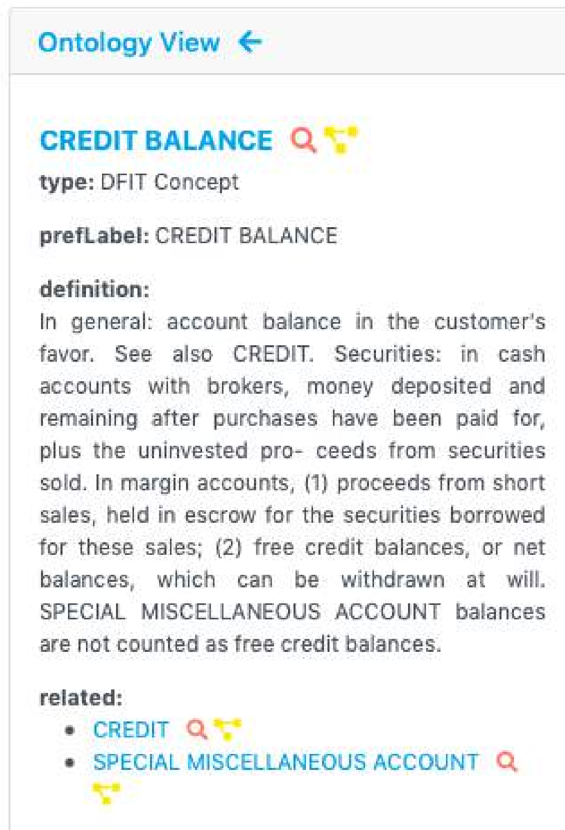
Definitions

The following definitions shall apply:

(1) ‘covered bond’ means a bond as referred to in Article 52(4) of Directive 2009/65/EC;

(2) ‘delivery option’ means the possibility to redeem the mortgage loan by buying back the covered bond at market or at nominal value in accordance with Article 33(3)(d) of Regulation (EU) No 575/2013.

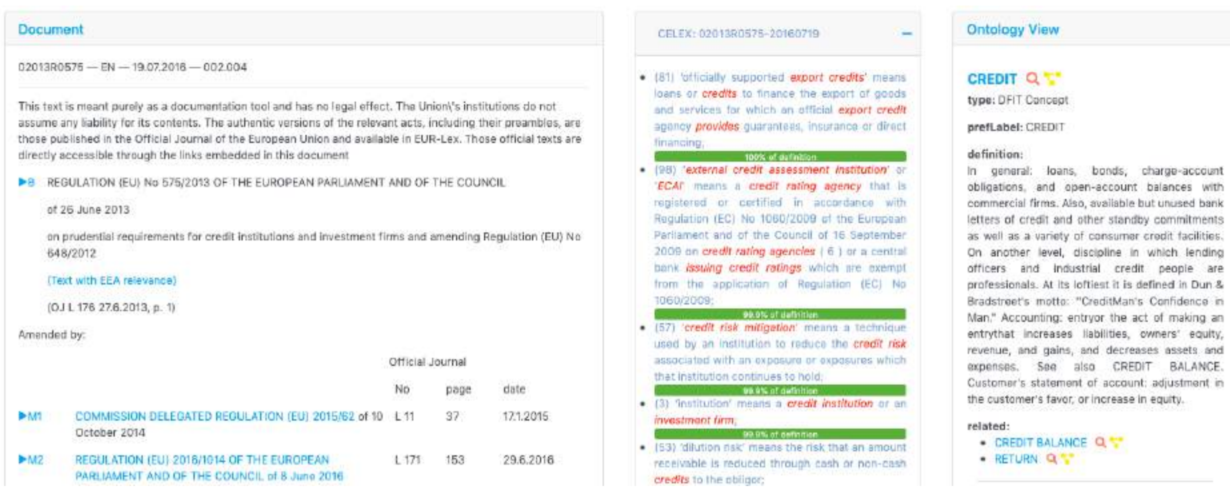




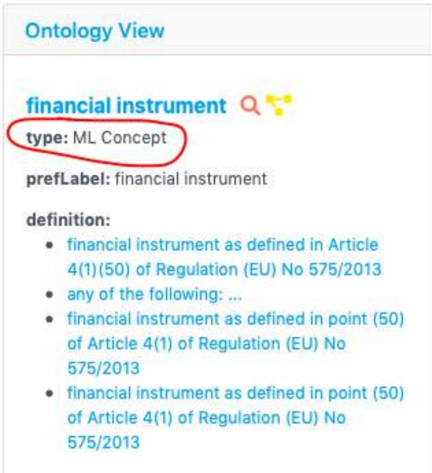
The terms may be browsed broader or narrower and related concepts are suggested (depending on the ontology used). The arrows on the top bar allow to move back and forth in-between the terms.

In order to find the document with the desired term from ontology, the user should click on the magnifying glass icon, on the right of the term. This will result in finding and displaying all the documents that have the term. There may be a situation that a given concept from the ontology doesn't have its representation in the documents or definitions. This is valid since the list of concepts in many ontologies is much broader than the ones existing currently in the documents.

Next steps are identical to normal concept searching, so clicking on the documents and finding the definition.



Whenever a term comes from the ontology type: ML Concept, (i.e. from the set of concepts pulled out from the documents by the Machine Learning algorithms), the application has a direct link between the concept, the definition and the document it is located in.
In the *Ontology View* (type**:** ML Concept), in the definition section, all the applicable definitions of a given concept are presented. Clicking on one, the application opens the respective document on the proper paragraph.



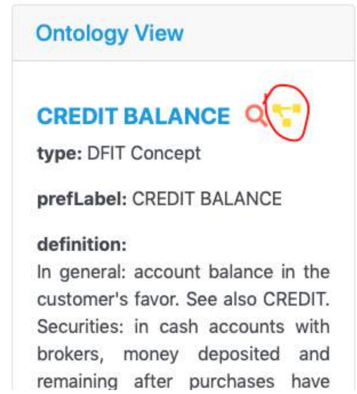
The application allows to browse concepts found in the particular documents, going from the document to list of concepts. This can be done through Docs2Concepts functionality. It is available through button on the top bar - next to DG FISMA title.

After clicking the button Docs2Concepts, the list of processed document appears. Expanding one of them presents a list of concepts in this particular document - sorted descending by the weight given them by the Search Insights. The number represents the relevancy of this term to the document.

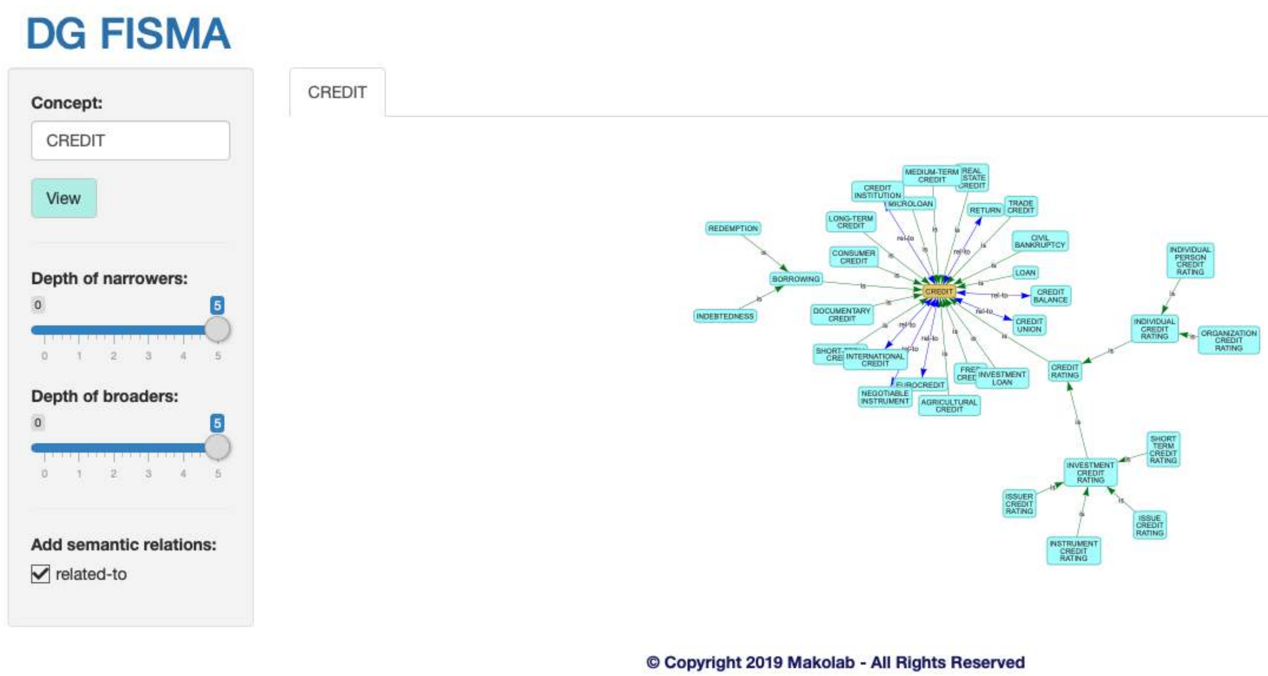
Clicking the term redirects the user to the given concept like it was typed in the search field.

The graph icon next to the term and magnifying glass represents a new functionality – a graphical representation of the term in relation to its broader and narrower terms.

MakoLab SA, Ogrodowa 8, 91-062, Łódź, t: 48 42 239 28 50, office@makolab.com, www.makolab.com



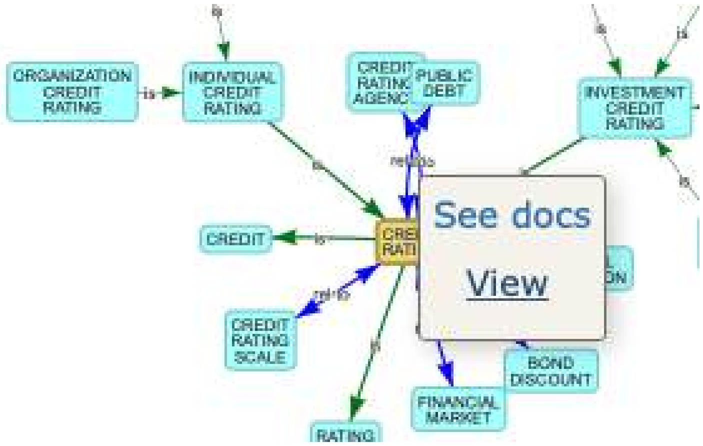
Clicking on it opens a new web application at [http://binsem.makolab.pl/ov/:](http://binsem.makolab.pl/ov/)



The application displays the searched term and its relations: broader and narrower accordingly, pointed with green arrows. The depth of the aforementioned can be adjusted using the bars on the left.

Checkbox "related-to", shows the terms related to the given term (pointed by blue arrows).

The graph can be zoomed in and out using mouse scroll. Moreover, each node of the graph can switch to parent-node by clicking on it and selecting "View".



Clicking "See docs" transfers the user to the main web application with a selected term displayed as a queried concept with all referred documents displayed. This way the user can go back and forth browsing and displaying the concepts in the ontology and documents.

Exemplary Ontology Suite

The suite of ontologies developed for the sake of the running example is available from http://graphchain.io/KnowML/mlknow_ontology_dg_fisma.ttl .