MEX Vocabulary: A Lightweight Interchange Format for Machine Learning Experiments

Diego Esteves
AKSW, University of Leipzig
Leipzig, Germany
esteves@informatik.uni-leipzig.de

Diego Moussallem
AKSW, University of Leipzig
Leipzig, Germany
moussallem@informatik.uni-leipzig.de

Ciro Baron Neto
AKSW, University of Leipzig
Leipzig, Germany
cbaron@informatik.uni-leipzig.de

Tommaso Soru
AKSW, University of Leipzig
Leipzig, Germany
tsoru@informatik.uni-leipzig.de

Ricardo Usbeck
AKSW, University of Leipzig
Leipzig, Germany
usbeck@informatik.uni-leipzig.de

Markus Ackermann
AKSW, University of Leipzig
Leipzig, Germany
ackermann@informatik.uni-leipzig.de

Jens Lehmann
AKSW, University of Leipzig
Leipzig, Germany
lehmann@informatik.uni-leipzig.de

ABSTRACT

Over the last decades many machine learning experiments have been published, giving benefit to the scientific progress. In order to compare machine-learning experiment results with each other and collaborate positively, they need to be performed thoroughly on the same computing environment, using the same sample datasets and algorithm configurations. Besides this, practical experience shows that scientists and engineers tend to have large output data in their experiments, which is both difficult to analyze and archive properly without provenance metadata. However, the Linked Data community still misses a lightweight specification for interchanging machine-learning metadata over different architectures to achieve a higher level of interoperability. In this paper, we address this gap by presenting a novel vocabulary dubbed MEX. We show that MEX provides a prompt method to describe experiments with a special focus on data provenance and fulfills the requirements for a long-term maintenance.

Keywords

Vocabulary, Data Provenance, Interchange Format, Machine Learning Experiments

1. INTRODUCTION

So far, we have seen a variety of publications on Semantic Web and Machine Learning (ML) topics, many of them contributing to the state of the art in their respective fields. However, in the last years we experienced a *knowledge gap* in the standardization of experiment results for scientific publications. This technological gap can be summed up by the following question: *"How to achieve interoperability among machine-learning experiments over different system architectures?"*. In particular, experimental results are often not delivered in a common machine-readable way, causing the information extraction and processing to be tricky and burdensome.

Moreover, recurring issues regarding the experiment could benefit from the existence of a public vocabulary. Reviewers of publications on Machine Learning often need to investigate basic information on experiments conducted, e.g., which implementation of an algorithm was used, its configuration or choices for related hyper-parameters. Several questions may arise during the reviews, such as *"Which kernel method did the authors use?", "What is the regularization constant value?", "How many folds were used for the cross-validation section?", "Did they normalize the data?", "Which data distribution was used?", "Have any hypothesis test been applied?"*.

This interpretation and look-up process is time-consuming and introduces misinterpretations which affect both comparability and reproducibility of scientific contributions. To address this problems, several *e-Science* tools and collaboration platforms have been developed to provide provenance and other metadata information for scientific experiments (e.g., *Wings* [6], *OpenTox* [18] and *MyExperiment* [16]).

However, these approaches were designed to collaborate within specific domains and are involved in non-generic scenarios (e.g., chemistry, bioinformatics, nanotechnology, neuroscience). Despite their noted achievements, we still do not have a consensus for a public format to achieve the interoperability for machine-learning experiments over any system implementation in a lightweight and simple format. In this sense, *Open ML* [20] emerges as the most promising idea. We discuss the current drawback in the Section 2.

The MEX vocabulary has been designed to reuse existing ontologies (i.e., *PROV-O* [11, 13], *Dublin Core* [21], and *DOAP* [4]). As previously stated, using a shared vocabulary and providing complete metadata information can help to tackle interoperability is-

SEMANTiCS '15, September 15-17, 2015, Vienna, Austria
© 2015 ACM. ISBN 978-1-4503-3462-4/15/09…$15.00
DOI: http://dx.doi.org/10.1145/2814864.2814883

sues, however our aim is not to describe a complete data-mining domain, which can be modeled by more complex and semantically refined structures, such as DMOP [9]. The aim of MEX is neither to re-design the existing scientific workflow approaches nor hold the whole set of existing scientific variables on its structure, but provide a simple and lightweight vocabulary for exchanging machine learning metadata to achieve a higher level of interoperability. In this context, Noy and McGuiness [14] pointed out that *"an ontology helps achieving this since it defines a common vocabulary for researchers who need to share information in a domain"*.

Our scientific contributions are: (1) definition of a lightweight public vocabulary for interchanging basic information regarding machine-learning experiments over different system implementations; (2) provision of a minimal set of meta-information needed to reproduce a (*Classification, Regression or Clustering*) machine learning problem; (3) the opportunity to increase collaboration on existing scientific workflow platforms, which can make use of MEX for interchange information among them; (4) reduced risk of misinterpretation on the design of machine learning experiments that publish appropriate MEX descriptions.

The paper is structured as follows: In the next section, we present the related works. Section 3 presents the MEX Vocabulary in more detailed terms and explicates how it is related with the machine learning workflow and its iterations. In Section 4, we describe some implementations as examples for the vocabulary. Finally, we give a outlook on the further development and possibilities for MEX in Section 5.

2. RELATED WORKS

The main aim of MEX vocabulary is to provide missing pieces for researchers to achieve higher levels of interoperability for data of machine learning experiment.

State-of-the-art workflow approaches have been successfully developed to face the current problems derived from the experimental process [10, 16, 18]. Moreover existing provenance information generated by the experimental process, such as *"datasets"*, *"algorithms and implementations"*, *"cross-validation and data splits process"*, *"versioning"*, *"hardware and network issues"*, *"preprocessing"* and *"features analysis"* can be managed into these workflow systems.

However, many machine learning experiments are developed using general purpose programming languages, such as *Java*, *C#*, *Python*, *C++*, with or without use of libraries like *Weka*[1], *scikit-learn*[2] or *Shogun*[3]. In this context, one of the requirements is to implement some machine-readable way for interchange results over complex architecture systems. Examples of state-of-the-art formats and patterns for interchanging are: *Comma-Separated Values (CSV)*, *eXtensible Markup Language (XML)*, *Value-Object (VO)*, *Data-Transfer-Objects (DTO)*. However, the drawbacks here are threefold: (1) the technology dependence for the implementation of the certain design pattern; (2) the possible lack of schema information on the implementation of certain format; (3) the lack of semantic information in both cases. Table 1 sums up the described related works on provenance meta-data for scientific experiments, whereas Table 2 shows the related works on data mining ontology.

[1]http://www.cs.waikato.ac.nz/ml/weka/

[2]http://scikit-learn.org/stable/

[3]http://www.shogun-toolbox.org

Table 1: The state-of-the-art platforms for e-science workflows

Platform	Description
MyExperiment [16]	It is a collaborative environment where scientists can publish their workflows and experiment plans
Wings [6]	A Semantic Approach to creating very large scientific workflows
OpenTOX [18]	An interoperable predictive toxicology framework
Open ML [20]	A frictionless, collaborative environment for exploring machine learning

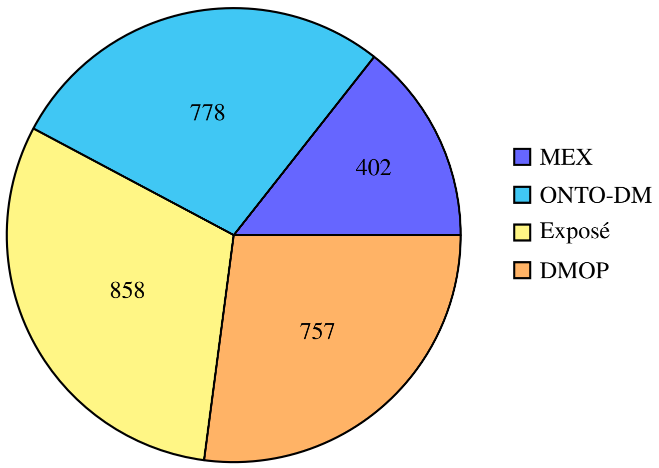
Table 2: The related (heavy-weight) ontologies for data mining and their respective conceptualizations

Platform	Description
The Data Mining OPtimization Ontology (DMOP) [9]	It supports informed decision-making at various choice points of the data mining process
OntoDM-KDD [15]	Ontology for representing the knowledge discovery process
Exposé [19]	An ontology for data mining experiments used in conjunction with *experiment databases* (ExpDBs [2])

2.1 Boundaries

Despite the peculiarity and different characteristic of each domain representation, we provide a simple quantity indicator (*number of existing classes*), comparing the number of classes existing on each related ontology as follows. Currently, 402 classes are represented by three MEX layers. Further, a complete experiment (machine learning execution) can be fully represented by MEX by implementing 7 main classes for *mexalgo*, 14 for *mexcore* and finally 5 classes of *mexperf* (considering the representation of a classification problem, for instance), totaling 26 referenced classes. The architecture of each layer is described in the Section 3.

However, it is worth noting the that lower coverage of MEX to provide data mining contexts. Accordingly, MEX provides a lightweight format for machine learning iterations instead of providing support for all decision-making steps that have an impact on the outcome of the knowledge discovery process. Thus, many additional machine learning issues are not covered. A discussion is introduced as following:



A. **Confusion Matrix**: There is no formal representation of the matrix (confusion matrix in supervised learning or matching matrix in unsupervised learning), however this information could be derived from the performance layer (*mexperf*).

B. **Pre-processing**: The basic information can be stored into the vocabulary as a string, which does not provide a semantic level of information though (mainly due to its complexity level, e.g.: *feature selection, feature extraction, parameter optimization*).

C. **Model Induction**: Due the complexity of each generated model, its meta-data is not represented into the MEX vocabulary, although it is possible to derive the needed information through the execution's parameters.

D. **Complex Algorithm Representation**: The level of representation for each algorithm relies on: *Algorithm Class*, *Algorithm Learning Method*, *Algorithm Learning Problem* and *Algorithm Implementation*. A more complex network structure to deeply represent the semantics for each classifier is not covered also due to complexity.

These issues require a more comprehensive analysis and are well represented by state-of-the-art ontologies (Table 2).

Besides, a more generic machine learning platform and very interesting approach for representing machine learning experiments is *OpenML* [20], which misses an ontology for representing the metadata[4] and runs over *XML* schema representation[5]. For this scenario, a decoupled vocabulary could help to achieve a higher level of interoperability. Thus, an integration is desired as future work to transform and import a *mex* file to this repository.

Finally, most of the machine learning libraries already provide a well defined machine readable meta-data to manage the information provided. In fact, the problem in this case is the missing of a common format to interchange the useful data among many systems and teams, which would oblige the needing of implementing different wrappers in order to define a common format. Therefore, the motivation is to provide a decoupled and lightweight language-independent format for achieving the higher level of interoperability designed specifically for representing the minimal set of flows existing on machine learning problems (*Classification, Regression and Clustering problems*).

3. MEX VOCABULARY: A GENERIC AND LIGHTWEIGHT INTERCHANGE FORMAT FOR ML

The MEX vocabulary has been designed to tackle the problem of sharing provenance information particularly on the basic machine learning iterations in a lightweight format. We extended the *W3C PROV Ontology (PROV-O)* since it provides an excellent model for representing, capturing and sharing provenance information on the Web. The *PROV-O* provides three main classes, *Entity*, *Agent* and *Activity*, as well as other classes and properties enriching the provenance representation. Also, is endorsed by W3C [1]. The MEX vocabulary is composed as three sub-vocabularies:

A. **MEX-Core**: formalizes the key entities for representing the basic steps on machine learning executions, as well as the provenance information for linking between the published paper and the produced meta-data (Figure 4).

[4]https://github.com/openml/OpenML

[5]http://www.openml.org/r/454640/output/description

B. **MEX-Algorithm**: representing the context of machine learning algorithms and their associated characteristics, such as learning methods, learning problem and class of the algorithm (Figure 5).

C. **MEX-Performance**: provides the basic entities for representing the experimental results of executions of machine learning algorithms (Figure 6).

The first release of MEX vocabulary aims to provide an embracing formalization to define the basics of a generic machine learning configuration (*"the algorithm and its parameters, the input features of given dataset, the sampling method and the hardware environment"*) as well as the representation of its results (*"measures"*) .

The following sub-sections (3.3, 3.4 and 3.5) describe each layer in a more refined way. A detailed mapping between the basic machine learning workflow and MEX Vocabulary is presented into the sub-section 3.2

3.1 PROV-O: Expressing PROV Data Models

The PROV-O (Figure 1) is a high-level standard for provenance endorsed by W3C [1], providing a set of classes, restrictions and properties that for representation and interchanging provenance data produced in many different systems and contexts.

"It provides a set of classes, properties, and restrictions that can be used to represent and interchange provenance information generated in different systems and under different contexts. It can also be specialized to create new classes and properties to model provenance information for different applications and domains" [1].

Therefore, its characteristics are suitable for the depiction of a generic machine learning experiment (Figure 2) regarding algorithms and hyper-parameters *mex-algo*, the measures for classification, regression, clustering problems, as well as the most used statistical measures for describing performances *mex-perf*. Moreover, the runs are defined by the *mex-core* layer.

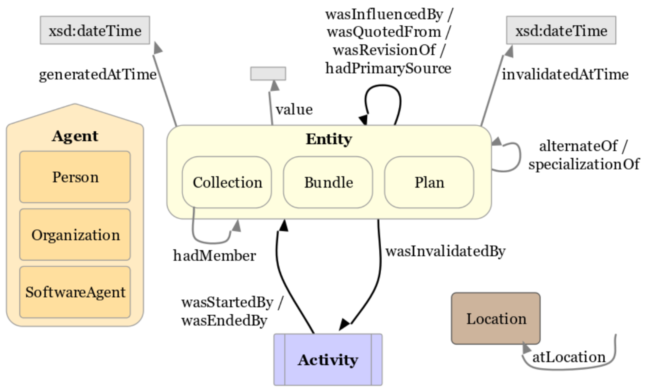


Figure 1: Expanded Terms: Entities as yellow ovals, Activities as blue rectangles, and Agents as orange pentagons [1].

The MEX Vocabulary has been fully designed reusing the PROV-O entities.

3.2 The Machine Learning Iterations

The overall goal is to provide a lightweight vocabulary to represent the executions generated by machine learning algorithms and its associated variables.

Figure 2 depicts the most common iterations for a set of executions of a machine learning problem (*Classification, Regression and Clustering*), excluding preprocessing characteristics mainly derived from complex data mining techniques, such as *feature analysis*, *denoising*, *data normalization* or *skewness*, for instance. Here, a real world problem (*ML Problem (A)* rounded circle) could be represented by a learning problem (e.g.: *Classification Problem* rounded circle) as well as split into many sub-problems ($SP_1, \ldots, SP_n$).

One of the key-factors to design a simple and lightweight format is to represent the entities and their relationships as simple as possible. In order to guarantee a clear level of representativeness, the vocabulary has been designed to perform each sub-problem as one experiment (*Experiment* rounded circle), i.e., each sub-problem can be represented by one *mex* file.

The mappings between the classical machine learning iterations (Figure 2) and the MEX Vocabulary classes are described as follows:

(A) ML Problem → collecting the basic information regarding the context of an experiment: `mexcore:Context` and `mexcore:ApplicationContext`.

(B) (Classification) Problem → definition of existing learning technique: `mexperf:PerformanceMeasure` and `mexperf:AlgorithmClass`.

(C) Experiment → basic information for a given experiment as well as its existing logical sub-experiments: `Experiment` and `ExperimentConfiguration`.

(D) Sampling Method → the applied experiment sampling method: `mexcore:SamplingMethod`.

(E) Hardware Configuration → `mexcore:HardwareConfiguration`.

(F) Execution → the core class to represent each run (`mexcore:SingleExecution` or `:OverallExecution`) of an algorithm. It connects the variables for a given execution: `Execution`.

(G) Model → represents the basic information regarding the generated (train) or the used (test) model/classifier `mexcore:Model`.

(H) DataSet → definition and representation of given dataset and its individuals: `mexcore:Dataset`, `mexcore:Example` and `mexcore:ExampleCollection`.

(I) Phase → the phase of given execution: train, validation, test: `mexcore:Phase`.

(J) Algorithm → the machine learning algorithm, its characteristics and its implementation: `mexalgo:Algorithm`, `mexalgo:LearningMethod`, `mexalgo:LearningProblem`, `mexalgo:AlgorithmClass`, `mexalgo:Implementation`.

(K) Parameters → the hyperparameters for given algorithm: `mexalgo:AlgorithmParameter`.

(L) Example Performance → Example measures (predicted and real values/labels) are represented by `mexperf:ExamplePerformance` and `mexperf:ExemplePerformanceCollection` (respective collection class).

(M) Overall → the *Classifier* overall measures are represented by the `mexperf:PerformanceMeasure` class, which has as subclasses: `:ClassificationMeasure`, `:RegressionMeasure`, `:StatisticalMeasure`, `:ClusteringMeasure`, `:UserDefinedMeasure`.

(N) Measure → the set of measures is represented by `mexperf:ExecutionPerformance`.

The following subsections (3.3, 3.4 and 3.5) detail each vocabulary's component.

3.3 MEX Core (`mexcore`)

- `:ApplicationContext`: Basic project information.
- `:Context`: Scientific context of an experiment, e.g.: stock market, link predictions, part-of-speech tagging, logistics.
- `:Experiment`: The main class to define one experiment and its basic information.
- `:Execution`: The smallest representation of a run of an algorithm. Each execution carry the associated values for each existing variable.
- `:ExperimentConfiguration`: Each set of executions should be grouped by one configuration i.e., for one experiment we could have many executions, in different hardware environments or over different algorithm configurations.
- `:HardwareConfiguration`: Basic hardware configuration information.
- `:SamplingMethod`: The sampling method applied.
- `:Phase`: `Train`, `Test` or `Validation`.
- `:Model`: The generated *Classifier*.
- `:DataSet`: The dataset used on the experiment.
- `:Example`: Each object represents the identification of an attribute (`dct:Identifier`) and the corresponding value (`prov:value`) (Figure 3). Predicted classes (labels) are represented by the `mexperf` layer.
- `:ExampleCollection`: It represents a row (instances/example) used as input for an execution.
- `:Feature`: Each object represents a used feature.

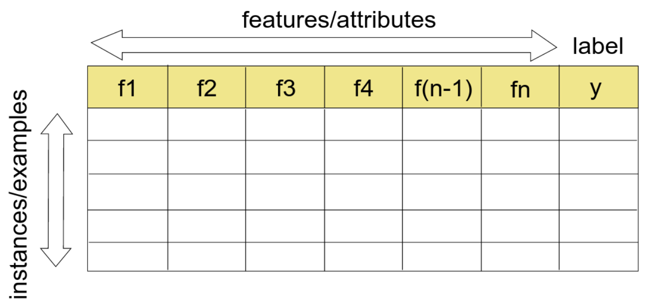


Figure 3: A classic input dataset for a machine learning execution

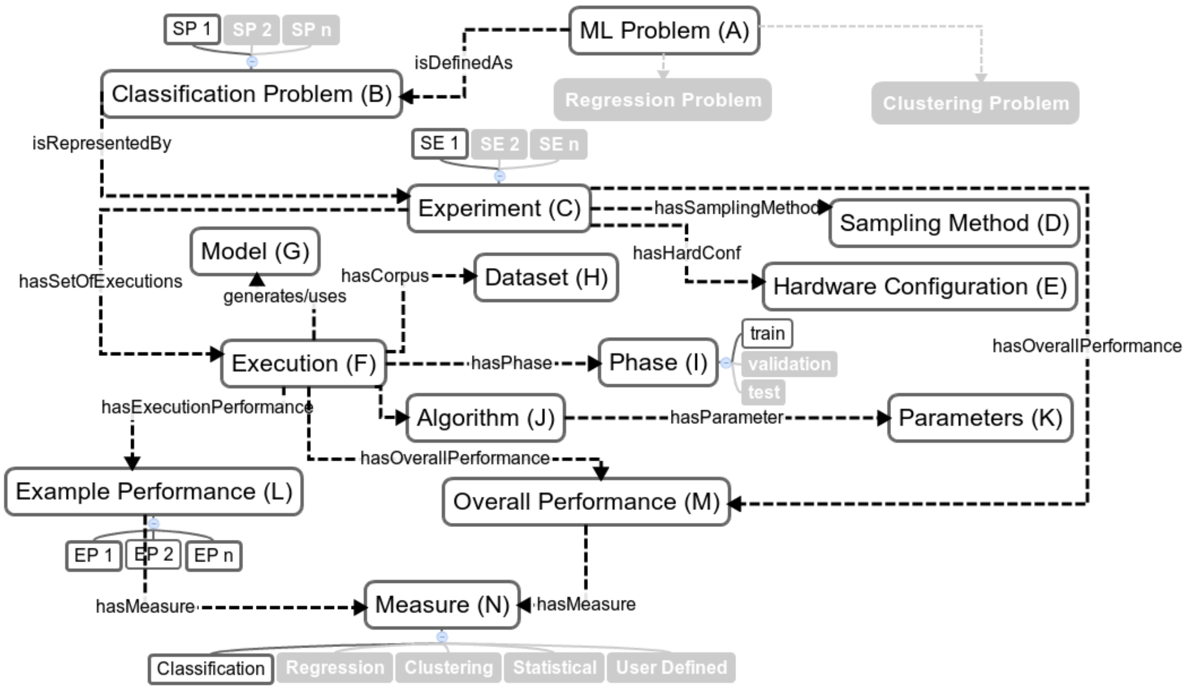


Figure 2: The diagram representing the classical iterations for an execution of a machine learning problem (*Classification, Regression or Clustering*). The white rounded rectangles representing a complete path for a *Classification* problem as well as its input (*Model*, *Corpus*, *Phase* and *Algorithm*) and outputs variables (*Example Performance* and *Overall Performance*)

3.4 MEX Algorithm (mexalgo)

- :LearningMethod: The algorithm learning method (e.g.:Reinforcement or Supervised).
- :LearningProblem: The algorithm learning problem (e.g.:Meta-heuristic or Association).
- :AlgorithmClass: The algorithm class (e.g.:ArtificialNeuralNetwork or Boosting).
- :AlgorithmParameter: the representation of each algorithm parameter and its associated values.
- :AlgorithmParameterCollection: A simple collection class for :AlgorithmParameter.
- :Implementation: Represents an implementation for an algorithm, i.e., the reference for a library or software that provides the specific algorithm (e.g.: Weka or SPSS).

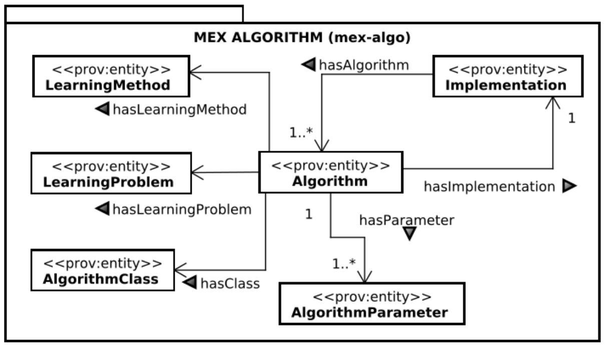

MEX ALGORITHM (mex-algo)
<<prov:entity>> LearningMethod
◀ hasLearningMethod
◀ hasAlgorithm
<<prov:entity>> Implementation
1..*
1
<<prov:entity>> LearningProblem
<<prov:entity>> Algorithm
hasImplementation ▶
◀ hasLearningProblem
1
hasParameter
<<prov:entity>> AlgorithmClass
◀ hasClass
1..*
<<prov:entity>> AlgorithmParameter

Figure 5: *mexalgo* layer: describing the algorithm information for certain machine learning experiment

3.5 MEX Performance (mexperf)

- :PerformanceMeasure: The root class for representing the measures. It is applicable whenever an overall execution (mexcore:OverallExecution) is being described.
- :ClassificationMeasure: usual measures for machine learning classification problems.
- :RegressionMeasure: usual measures for machine learning regression problems.
- :StatisticalMeasure: usual statistical measures.
- :ClusteringMeasure: usual measures for machine learning clustering problems.
- :UserDefinedMeasure: a special case for representing specific measures for given domain, which tends to be unusual for a generic machine learning context, but necessary. Examples are particular indicators and metrics for representing thresholds in models of stock market prediction.
- :UserDefinedMeasureCollection: A collection for the :UserDefinedMeasure class.
- :ExamplePerformance: used whenever a single execution is being described (mexcore:SingleExecution). It represents the predicted class or value for those specific execution as well as the real class or value (Figure 3). It is linked to the mexcore:Example class.

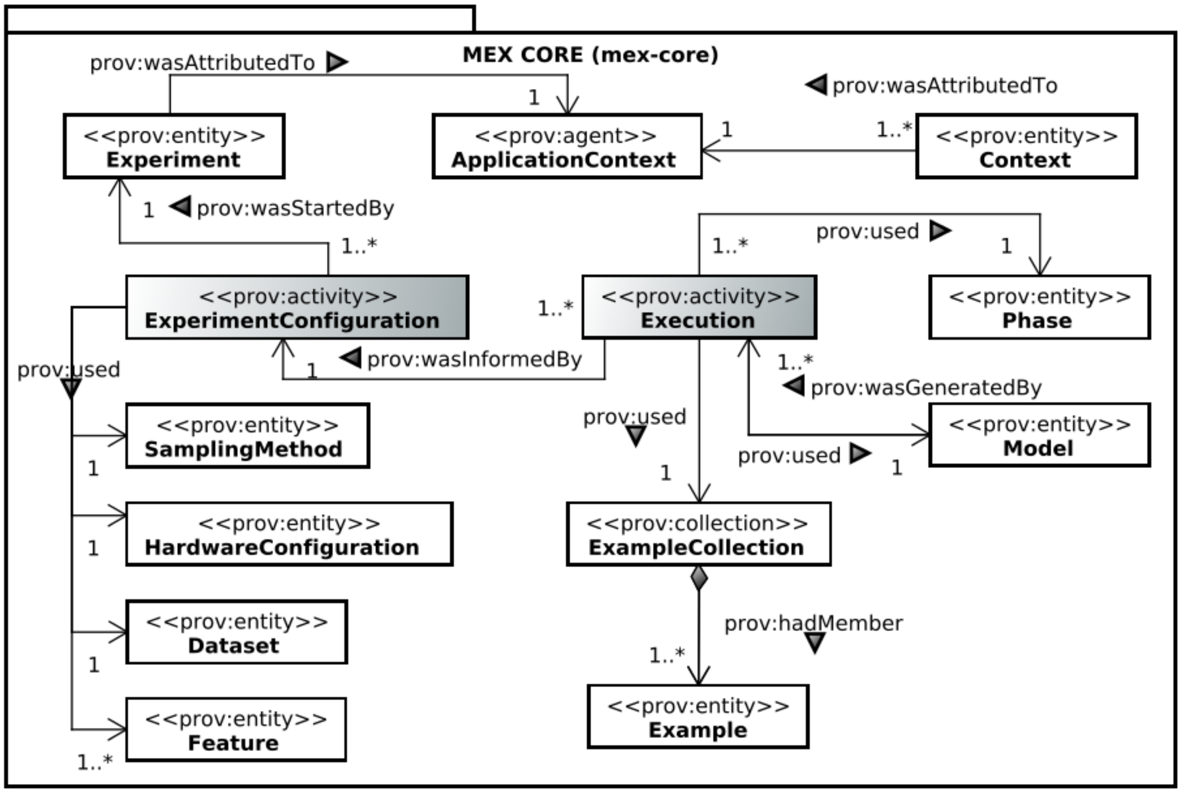


Figure 4: *mexcore* layer: blue classes defined as *prov:activity*, describing the core information for certain machine learning experiment

- `:ExamplePerformanceCollection`: A collection for `:ExamplePerformance` class.
- `:ExecutionPerformance`: represents the class to control the performance of each execution as well as interlink the MEX layers.

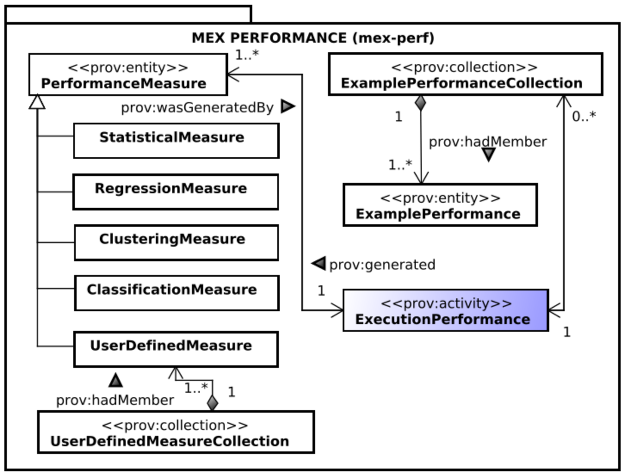

MEX PERFORMANCE (mex-perf)
1..*
<<prov:entity>> PerformanceMeasure
<<prov:collection>> ExamplePerformanceCollection
prov:wasGeneratedBy
1
0..*
StatisticalMeasure
prov:hadMember
1..*
RegressionMeasure
<<prov:entity>> ExamplePerformance
ClusteringMeasure
prov:generated
ClassificationMeasure
1
<<prov:activity>> ExecutionPerformance
1
UserDefinedMeasure
prov:hadMember
1..*
1
<<prov:collection>> UserDefinedMeasureCollection

Figure 6: mexperf layer: describing the performance of the experiment

3.6 MEX Layers: linking the information

The Figure 7 depicts the most important relationships existing on MEX to connect and represent the flow of information among the three layers. Besides, it represents the maximum level of abstraction for given machine learning problem: a machine learning classifier (`mexalgo:Algorithm`) runs over an existing set of parameters (`mexcore:SingleExecution` or `mexcore:OverallExecution`) and produces a particular set of measures (`mexperf:ExecutionPerformance`).

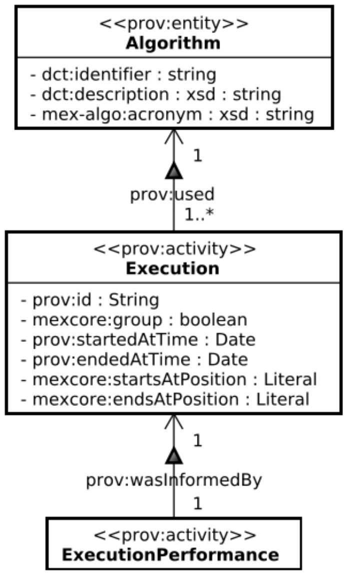

<<prov:entity>> Algorithm
- dct:identifier : string
- dct:description : xsd : string
- mex-algo:acronym : xsd : string
1
prov:used
1..*
<<prov:activity>> Execution
- prov:id : String
- mexcore:group : boolean
- prov:startedAtTime : Date
- prov:endedAtTime : Date
- mexcore:startsAtPosition : Literal
- mexcore:endsAtPosition : Literal
1
prov:wasInformedBy
1
<<prov:activity>> ExecutionPerformance

Figure 7: The interlinking among layers (*mex-core*, *mex-algo* and *mex-perf*)

4. USE CASES

In this section, we will illustrate how the MEX ontology can be used in real scenarios. Figure 8 shows an excerpt of the RDF graph applied to an experiment dubbed *Labour Negotation Contract Offer Judgement Predictions*. The experiment's aim was to document the application of various classification methods implemented in the WEKA toolkit to judge about offers for Canadian labour contracts. As can be seen, the `Execution` instance named `exec-A` is the central hub of the subgraph. Simply, each execution `exec-A` has one or more performance measure, grouped by the `ExecutionPerformance` class. Data provenance information is ensured through `prov:used` properties, which link the individual above to `training-` and `testSplit`, namely the working datasets, the `wekaLADtree` and `alternatingDecisionTree` algorithms, the `adtTreeModel` and the `hardware` configuration. Datatype values are not displayed in the sub-graph due to lack of space.

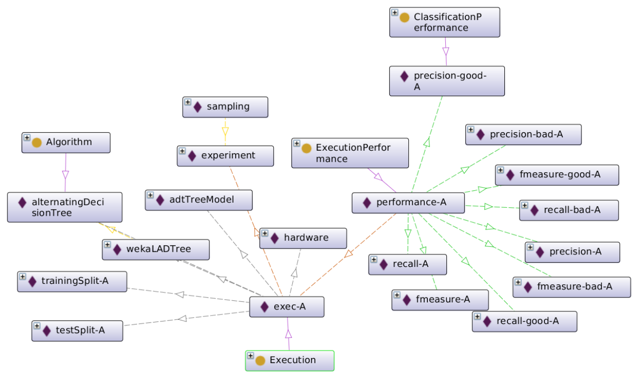


Figure 8: An excerpt of the RDF graph from the Labour use-case. Full arrows are inverse `rdf:type` properties, whilst dashed arrows are `prov:used` properties.

A second real world example is the creation of provenance information for the *ACL Wiki page for part-of-speech (POS) taggers*[6] which does not provide the desired level of information for a benchmark. For example, the features are not clearly stated as well as the algorithms (just the respective implementations). Some approaches were defined with no machine-readable information, such as the defined cross-validation method for the *TnT* system. Figure 9 shows an excerpt of the RDF graph for representing the *French TreeBank* results. Here, the *mex* file could be used in order to provide the needed metadata information to achieve a reasonable level of comparison between new approaches.

As example, we defined an instance of the `mexcore:ExperimentConfiguration` in order to group the three existing systems (*Morfette, SEM and MElt*) per *Sampling Method* (`sm1`) method and *Dataset* (`ds1`), both defined on the current benchmark. Each execution (`exec1, exec2 and exec3`) points to an execution performance instance (`ep1, ep2 and ep3`, respectively). It is worth noting that the POS models are commonly evaluated, among others, by the accuracy of *"unknown words"*. Although these specific measure is not represented into the *MEX Performance* layer, we could represent it in a *"string level"*, making use of the `mexperf:UserDefinedMeasure` class. As future work we plan to represent the *Natural Language Processing* scenarios extending MEX with existing state-of-the-art Vocabularies, such as NIF [8], creating a subclass of `mexperf:PerformanceMeasure` class for representing NLP measures.

[6]http://aclweb.org/aclwiki/index.php?title=POS_Tagging_(State_of_the_art)

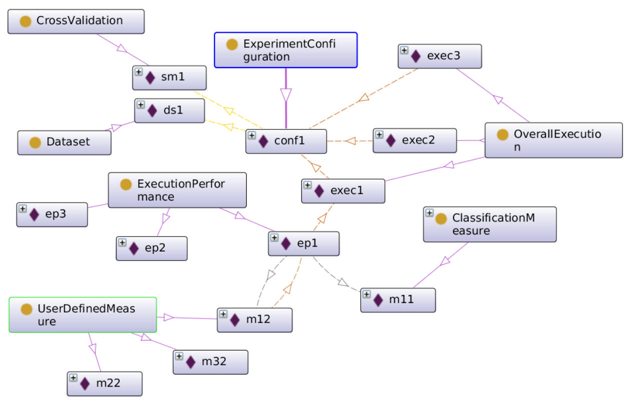


Figure 9: An excerpt of the RDF graph from the ACL POS Tagging use-case.

Other real-world examples for machine learning implementations can be found into the MEX repository on GitHub[7]. These examples refer to the work presented in [3, 5, 17]. All resources are available on the MEX Project[8] website.

5. CONCLUSIONS AND FUTURE WORK

In this paper, we introduced the MEX Vocabulary to explain provenance information for machine learning experiments, built on the PROV-O ontology and following Linked Data best-practices. We argue that by using the MEX vocabulary, researchers will be able to interchange the machine learning experiments by a metadata common, interoperable and lightweight format. Despite the high level of interoperability achieved with MEX vocabulary, the development of *APIs* is planned in order to support the usage for research groups which have no semantic web knowledge as background, once the manipulation of the vocabulary in different programming-languages entails an extra effort (e.g.: a machine learning problem implementation using *Java* or *Python* as programming-language). Also, a notorious issue is related with the update management process for algorithms and software implementations, since it is impractical to map all existing entities for those classes. In order to minimize this gap, besides focus on state-of-the-art software for machine learning (e.g.: Weka), we plan to provide an user interface for suggesting missing entities.

As future work, we plan the integration with *OpenML* platform as well as examine more machine learning representations. Besides, we plan to integrate MEX into existing machine learning tools, such as Weka [7] and DL-Learner [12], for instance, allowing a fully transparent process of the *mex* file generation to the end user.

Acknowledgments This research has been partially supported by grants from the CAPES foundation, Ministry of Education of Brazil, Brasilia - DF 70040-020, Brazil (Bolsista da CAPES - Proc. n: BEX 10179/13-5) and the H2020 ALIGNED Project (GA No. 644055). Sincere thanks to Maria Claudia Reis Cavalcanti and Julio Cesar Duarte for assisted the research and Sebastian Hellmann for his expertise.

[7]https://github.com/AKSW/mexproject/tree/master/proof
[8]http://aksw.org/Projects/MEX.html

6. ADDITIONAL AUTHORS

7. REFERENCES

[1] W3C PROV-O ontology. http://www.w3.org/TR/prov-o/.

[2] H. Blockeel and J. Vanschoren. Experiment databases: Towards an improved experimental methodology in machine learning. pages 6–17. Springer Berlin Heidelberg, 2007.

[3] L. Bühmann, D. Fleischhacker, J. Lehmann, A. Melo, and J. Völker. Inductive lexical learning of class expressions. In *Knowledge Engineering and Knowledge Management*, pages 42–53. 2014.

[4] E. Dumbill. Doap: Description of a project, 2012.

[5] D. Esteves and J. C. Duarte. Prediction of assets behavior in financial series using machine learning algorithms. *International Journal of Advanced Research in Artificial Intelligence*, 2(11), 2013.

[6] Y. Gil, V. Ratnakar, J. Kim, P. A. González-Calero, P. T. Groth, J. Moody, and E. Deelman. Wings: Intelligent workflow-based design of computational experiments. *IEEE Intelligent Systems*, pages 62–72, 2011.

[7] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I. H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, Nov. 2009.

[8] S. Hellmann, J. Lehmann, S. Auer, and M. Nitzschke. Nif combinator: Combining nlp tool output. In *Knowledge Engineering and Knowledge Management*, volume 7603 of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg, 2012.

[9] C. M. Keet, A. Ławrynowicz, C. d'Amato, A. Kalousis, P. Nguyen, R. Palma, R. Stevens, and M. Hilario. The Data Mining OPtimization Ontology. *Web Semantics: Science, Services and Agents on the World Wide Web*, pages 43 – 53, 2015.

[10] J. Kim, E. Deelman, Y. Gil, G. Mehta, and V. Ratnakar. Provenance trails in the wings-pegasus system. *Journal of Concurrency And Computation: Practice And Experience*, pages 587–597, Apr. 2008.

[11] T. Lebo, S. Sahoo, D. McGuinness, K. Belhajjame, J. Cheney, D. Corsar, D. Garijo, S. Soiland-Reyes, S. Zednik, and J. Zhao. Prov-o the prov ontology. *W3C Recommendation, 30th April*, 2013.

[12] J. Lehmann. Dl-learner: Learning concepts in description logics. *J. Mach. Learn. Res.*, 10:2639–2642, Dec. 2009.

[13] L. Moreau and P. T. Groth. *Provenance: An Introduction to PROV*. Synthesis Lectures on the Semantic Web: Theory and Technology. Morgan & Claypool Publishers, 2013.

[14] N. F. Noy and D. L. Mcguinness. Ontology development 101: A guide to creating your first ontology. Technical report, 2001.

[15] P. Panov, L. Soldatova, and S. Džeroski. Ontodm-kdd: Ontology for representing the knowledge discovery process. In *Discovery Science*, pages 126–140. 2013.

[16] D. D. Roure, C. Goble, and R. Stevens. The design and realisation of the myexperiment virtual research environment for social sharing of workflows. *Future Generation Computer Systems*, 25:561–567, February 2009.

[17] T. Soru and A.-C. N. Ngomo. A comparison of supervised learning classifiers for link discovery. SEM '14, pages 41–44, 2014.

[18] N. I. e. a. Tcheremenskaia O, Benigni R. Opentox predictive toxicology framework: toxicological ontology and semantic media wiki-based opentoxipedia. *Journal of Biomedical Semantics*, 2012.

[19] J. Vanschoren and L. Soldatova. Exposé: An ontology for data mining experiments. *International workshop on third generation data mining: Towards service-oriented knowledge discovery (SoKD-2010)*, pages 31–46, 2010.

[20] J. Vanschoren, J. N. van Rijn, B. Bischl, and L. Torgo. Openml: Networked science in machine learning. *SIGKDD Explor. Newsl.*, pages 49–60, June 2014.

[21] S. Weibel, J. Kunze, C. Lagoze, and M. Wolf. Dublin core metadata for resource discovery. *Internet Engineering Task Force RFC*, page 132, 1998.