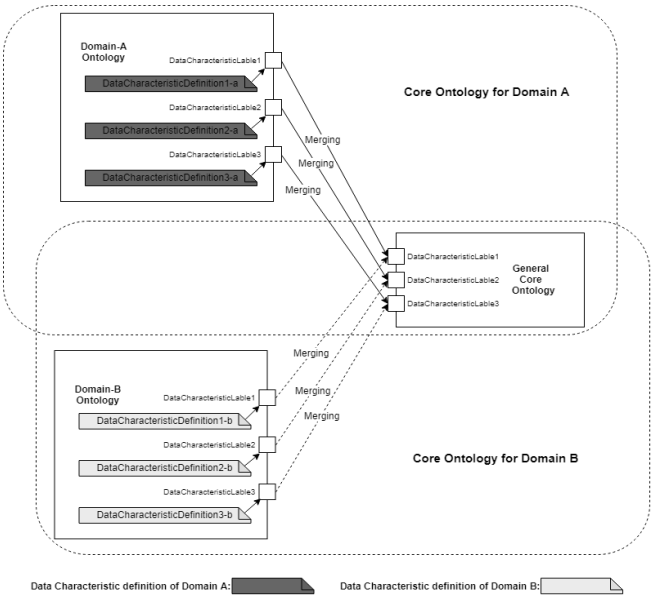

Fig. 4. Data representation through merging different domain ontologies

A. Ontology notations

An ontology is made up of a set of concepts *(C)*, properties *(P)*, property mappings *(T),* and relationships between the concepts *(R)* [6, 22].

- Let *O* define an ontology.
- Let *C* define the set of concepts in the ontology.
- Let *P* define the set of properties of the concepts.
- Let *T* define the set of property mappings, mapping properties to concepts.
- Let *R* define the set of relationships that relate one concept to another.

which is $O = \{C, P, T, R\}$.

Concepts are the nodes or objects that identify something that exists. Relationships are used to indicate a similarity between two concepts within an ontology. They can either link two concepts together or loop back and link to the same concept. Properties provide extra features used to identify the concept. The property mapping element is similar to a relationship element, but it links a property to a concept rather than one concept to another.

The merge process occurs in general core ontology O_g and domain ontology O_d. In general, core ontology, concepts C_{dm} and relationships P_{dm}, T_{dm}, R_{dm} in the field of data mining are described. The concepts C_g of data characteristics are also included as part of the algorithm performance description.

Which is $O_g = \{C_{dm}, C_g, P_{dm}, T_{dm}, R_{dm}\}$.

In the domain ontology O_d, domain experts define the concepts C_d ($C_d \subseteq C_g$) and specific descriptions (internal connections) P_d, T_d, R_d of the domain data characteristics according to the particular situation of the domain dataset.

Which is $O_d = \{C_d, P_d, T_d, R_d\}$.

B. Ontology merging

For the ontology merging technology, the problem of finding common points for merging is crucial [19, 20, 21]. Knowledge workers must ensure that as many merge points as possible are included in the original ontology to ensure a strong merge. And the ontologies to be merged are complete and valid at the beginning of the merge process.

In our ontology construction, general core ontology is a complete and valid ontology that has been created. The concepts in the domain ontology have been preset. The domain experts only need to specify the range and values of the data characteristics definitions and ensure that these values do not conflict.

The merging steps are as follows:

1) Check for consistency completeness of the initial ontologies O_g and O_d.

2) Check that there is at least one valid merging point C_d in both sets.

3) Merge O_g and O_d at each of the merge points C_d.

a) Replace the domain data characteristics name C_d in O_g with C_d in O_d.

b) Add the domain data characteristics definitions $\{P_{dm}, T_{dm}, R_{dm}\}$.

4) Generate the domain-oriented ontology $O_{gd} = \{ C_{dm}, C_g, P_{dm}, P_d, T_{dm}, T_d, R_{dm}, R_d\}$.

5) Check for the validity of the new merged ontology O_{gd}.

6) Check for semantic completeness of the merged ontology O_{gd}.

It is worth noting that domain ontology and general core ontology describe distinct domains: data characteristics and algorithmic knowledge. Their only intersection is the conceptual names of the data characteristics, i.e., C_d, which are identified as the merging points.

Because O_g and O_d are highly independent, problems usually don't appear in completeness and validity checks.

V. Domain-oriented ontology content

In the initialization phase, core ontology is a general ontology, including an "INPUT" ontology and some other existing DM ontologies (DMOP, OntoDT, OntoDM, and DMWF).

Domain ontology is built through defining the existing entities of data characteristics in general core ontology.

Then experts import domain knowledge in the form of domain ontology, and we merge the domain ontology and the general core ontology to obtain a core ontology for a specific domain, i.e., domain-oriented ontology (see Fig. 5.).

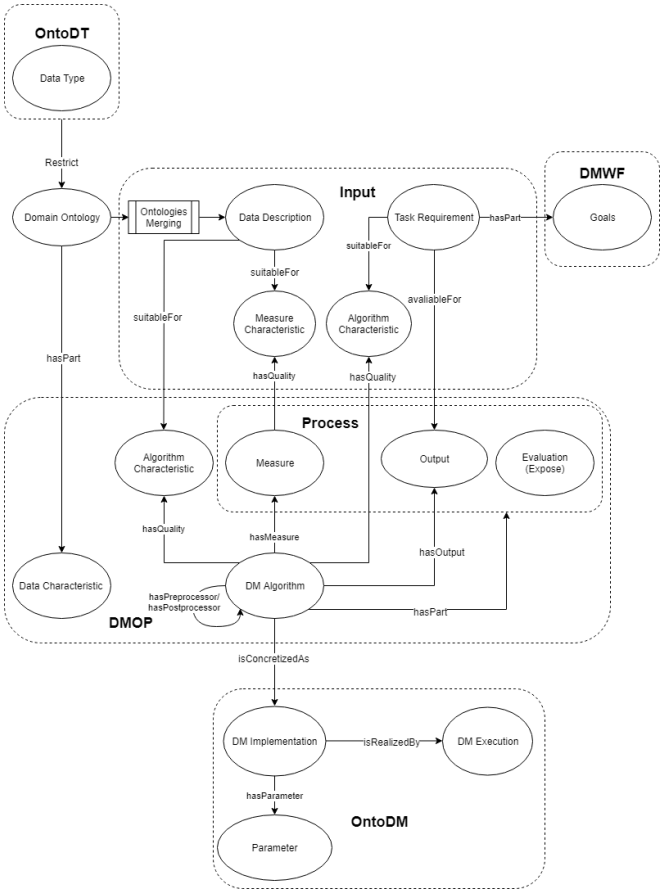

Fig. 5. The general structure of the domain-oriented ontology

A. *INPUT ontology for data understanding and business understanding*

We create “INPUT” ontology as the input interface for the user query. Its primary contents are:

- Define data characteristic entities corresponding to algorithmic characteristics.
- Describe the requirements of the DM task, that is, the output of the DM algorithm.
- Supplement the missing algorithm characteristics and measure characteristics in the existing DM ontologies.

INPUT ontology is the part directly associated with the user's queries. It makes the use of ontology more explicit. Users do not need to understand other internal structures of the ontology.

B. *Data characteristic description in INPUT ontology*

For building domain ontology, the critical point is to provide restrictions for the description of the domain ontology at the upper level. In the previous work, there is no suitable method to describe the data set in the form of ontology entities. In the general data type ontology OntoDT, the basic properties of the data set are defined. However, these properties cannot directly influence the DM generation process. The selection of the DM algorithm is based on the data set characteristics and task requirements. However, the definitions of these characteristics are different in different fields.

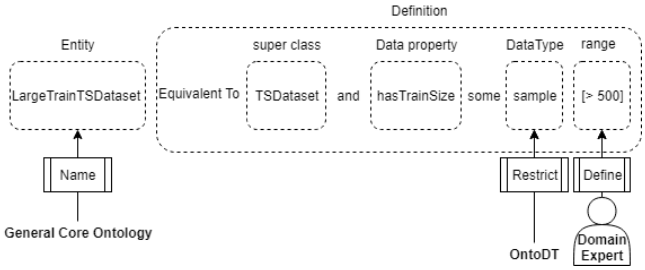

Fig. 6. The definition of data characteristic “LargeTrainTSDataset”

To make the ontology adaptively present data sets in various domains, we use the OntoDT classes as parameters to specify the definition (value or range) of data characteristics in general core ontology. Domain experts describe domain knowledge or existing domain ontology in general core ontology, making it suitable for data analysis tasks in this domain. An example of the definition in the domain of time series classification (TSC) is shown in Fig. 6.

The suitable DM processes are obtained by querying the generated core ontology for a specific domain.

C. *The integration of existing DM ontologies for other DM phases*

INPUT ontology is also the core part of integrating existing DM ontologies. The integration operation is based on the purpose of generating suitable solutions and processes.

In the process of integration, to reduce the complexity of the ontology, we discarded contents that were useless for this purpose and restructured the structures. The main classes in the domain-oriented ontology are shown in Table I.

The reconstruction contents are as follows:

- OntoDT is fully retained as an upper-level restriction that defines the characteristics of the data.
- The class "Goals" in DMWF and class "DM-Task" in OntoDM are extracted for the description of task requirements.
- Although DMOP provides more than a hundred DM algorithms and their characteristics, we have reconstructed its structure. As components of the DM algorithms, the classes “Measure,” "Output," "Evaluation," and "DM Algorithm" itself are included in a new class "Process" so that it is more understandable for the users.
- OntoDM describes the last CRISP-DM phase, “Development.” The classes "DM Implementation" and "Parameter" in OntoDM are integrated for the possible parameters setting. And "DM Execution" presents where and how to execute the selected algorithms.

TABLE I. THE MAIN CLASSES IN THE DOMAIN-ORIENTED ONTOLOGY

Class	source	Annotation
Data Description	INPUT	Describes the dataset characteristics in the form of ontology entities. Domain experts define their value and ranges.
Task Requirement	INPUT	Describes the task requirements in the form of ontology entities.
Measure Characteristic	INPUT	Existing DM ontologies do not describe the performance of measures (i.e., distance functions). In the INPUT ontology, we describe and name it "MeasureCharacteristic."
Algorithm Characteristic	INPUT /DMOP	Describes the performance of DM algorithms, including tolerating some data set defects (such as Missing value, Noise value), suitable for some task requirements (such as two-class, multi-class).
Data Type	OntoDT	Provide basic data types that describe the characteristics of the data set (such as sample, label)
Goals	DMWF	Provide a description of the task requirements. It mainly focuses on the generalization of the types of output results.
DM-Tasks	OntoDM	Provide a description of the task requirements. It mainly focuses on the description of specific details of the task.
Data Characteristic	DMOP	Provided by DMOP, the names of the characteristic of the dataset.
DM Algorithm	DMOP	Describe all DM algorithms that have been designed to perform any of the DM tasks, such as feature selection, missing value imputation, or modeling (or induction).
Measure	DMOP	Describes the distance functions and similarity functions, which usually directly affect the performance of DM algorithms.
Output	DMOP	Describe the output models of the DM algorithms (such as decision tree structure, probability distribution structure).
Evaluation	DMOP	Describe the evaluation functions of the DM algorithms (such as external validity model function for clustering algorithms).
DM Implementation	OntoDM	Provide a DM algorithm implementation scheme and parameter settings
DM Execution	OntoDM	Provide executable solutions for DM algorithms (such as R, python package, Weka)
Parameter	OntoDM	Provide parameters for DM algorithms (such as distance threshold, number of clusters and variance threshold for K-means algorithm)

In order to build the logical structure of core ontology, the relevant properties are defined in Table II.

TABLE II. THE RELEVANT PROPERTIES IN THE DOMAIN-ORIENTED ONTOLOGY

Property	Domains	Ranges	Answering the competency questions
availableFor	INPUT	Characteristics	Given data characteristics or task requirements, which characteristics should the DM algorithms have so that they are suitable for?
suitableFor	INPUT	Characteristics	Given data characteristics or task requirements, which characteristics should the DM algorithms have so that they are available?
hasQuality	Process	Characteristics	Which characteristics does the given process have?
hasPreprocessor hasPostprocessor hasOutput hasMeasure hasEvaluation	DM Algorithm	Process	Which processes do the DM algorithm have?
isConcretizedAs	DM Algorithm	DM Implementation	How can we implement the DM algorithm?
hasParameter	DM Implementation	Parameter	Which parameters should we set when we implement the DM algorithm?
isRealizedBy	DM Implementation	DM Execution	Where and how can we execute the DM algorithm?

VI. USAGE

As long as the structure of the ontologies is reasonable, they can be operated on the corresponding editing software, for instance, Protégé. Based on the relations presented in Table 2, users can query for suitable solutions with the following workflow.

A. General Workflow

The workflow of domain-oriented ontology for data analysis in a specific domain is as follow:

1) Based on the restrictions of OntoDT, domain experts define the characteristics of domain data in the form of ontology.

2) Merge the domain ontology and the general core ontology to obtain the core ontology for the specific domain.

3) Manually obtain task requirements and data sets and describe them in the form of ontology entities as the inputs.

4) Execute the selection process on this core ontology for a specific domain.

a) Input the entities of input-data description and task requirements. Based on the relation "suitableFor", obtain the characteristics which the solutions should have.

b) According to the relation "hasQuality," obtain the algorithms or measures which have suitable characteristics.

If the results are measures, obtain the algorithms according to the relation "hasMeasure."

c) Choose the most suitable algorithms which meet the characteristics as many as possible. They are the selected solutions.

d) According to the relation "hasPre/Postprocessor," obtain the entire DM process.

e) According to the relation "hasPart," obtain the process of the selected solutions.

f) According to the relation "isConcretizedAs," obtain the implementations and parameter variants.

g) According to the relation "isRealizedBy," obtain the available executions

B. The application for time series classification

Domain-oriented ontology can be flexibly applied to the data analysis process in different fields. As an application example, we constructed an ontology oriented on solving the time series classification (TSC) tasks. The entities of TSC data characteristics have been named in "INPUT" ontology. For describing the TS datasets in the form of these entities, explicit definitions are needed.

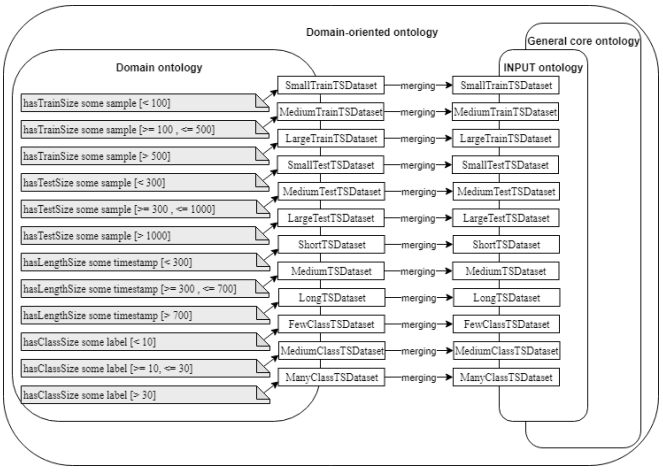

Fig. 7. Merging TSC domain ontology with general core ontology

Expert knowledge of the definition of characteristics of TSC data comes from [17]. We define them in domain ontology, then merge them with the labels in "INPUT ontology" as the Fig. 7. shows. Then users can represent the TS datasets in the domain-oriented ontology.

The interaction between the users and domain-oriented ontology takes place on "INPUT" ontology. Users can describe the dataset and query the corresponding entities of data characteristics in the following form:

"TSDataset and hasTrainSize exactly 40 sample"

Then users can receive the corresponding entity "SmallTrainDataset".

INPUT ontology allows formulating the tasks in the common form. For example, the query for suitable solutions is:

"*Algorithm*

and suitableFor some SmallTrainTSDataset

and suitableFor some LargeTestTSDataset

and suitableFor some LongTSDataset

and suitableFor some FewClassTSDataset

and suitableFor some ECGTSDataset"

Which the entities "SmallTrainTSDataset," "LargeTestTSDataset," "LongTSDataset," and "ECGTSDataset" are characteristics of the data set and the entities "FewClassTSDataset" means the task requirement is a few classes.

As Fig. 8. shows, BOSS (Bag of SFA Symbols), COTE (Collection of Transformation E), EE (Elastic Ensemble), MSM_1NN (Move-Split-Merge) and ST (Shapelet Transform) are selected as the answer to this query since these algorithms are suitable for all the conditions. For more concrete examples, please refer to [18].

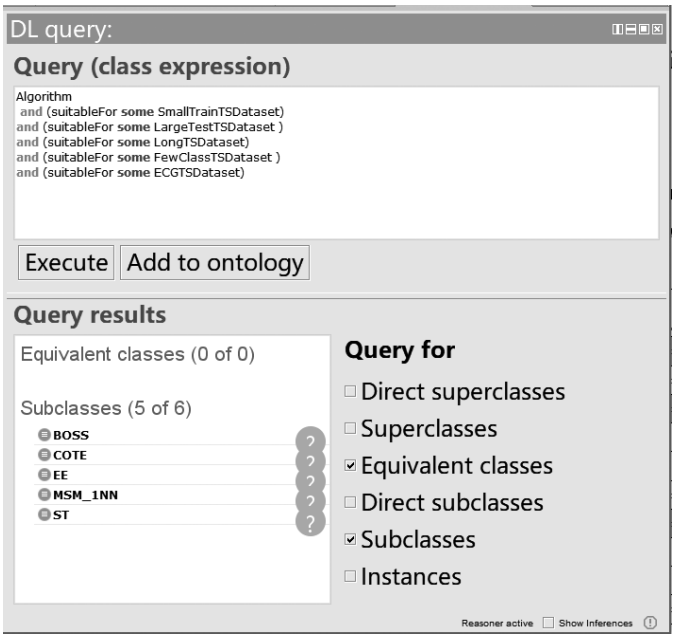

Fig. 8. An example of the query in the domain-oriented ontology

We used 45 available TSC algorithms to process the dataset, which has the example characteristics. A comparison of the accuracy of all algorithms is shown in Fig. 9. The selected algorithms have shown excellent performance. The average accuracy of selected algorithms (0.9364) is significantly better than the average accuracy of all algorithms (0.7660).

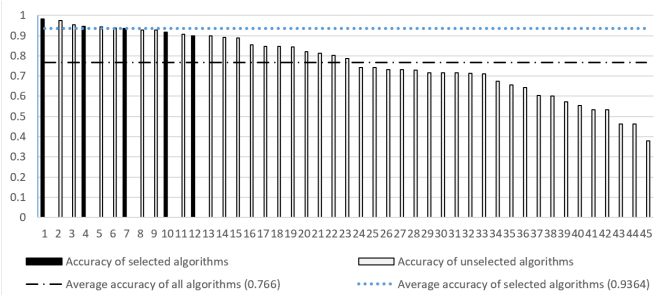

Fig. 9. A comparison of algorithm accuracy for the example dataset

VII. Conclusion

This paper proposes a meta mining ontology framework for domain data adaptive processing. It allows constructing the domain-oriented ontology through creating an "INPUT" ontology that describes the characteristics of the data and task requirements and reconstructing and integrating existing DM ontologies. The domain-oriented ontology can be used as an intelligent assistant for domain data mining. The basic usage has been presented in this paper.

We also propose an ontology merging method to solve the problem of describing domain-oriented data characteristics in the ontology. The data characteristics in the field of time series classification are described in the ontology by the proposed method.

Although, the ontology is focusing on building the foundation of data mining, it can be used by practitioners in real-world applications to optimize knowledge discovery processes by sequentially querying the suitable solutions based on specific task requirements and data characteristics. Meanwhile, domain-oriented ontology is intended to be extensible and will continue to be updated to reflect future advancements in using it for building high-quality data-analytical processes rapidly.

References

[1] Jankowski, Norbert, Włodzisław Duch, and Krzysztof Grąbczewski, eds. *Meta-learning in computational intelligence*. Vol. 358. Springer, 2011.

[2] Hilario, Melanie, et al. "A data mining ontology for algorithm selection and meta-mining." *Proceedings of the ECML/PKDD09 Workshop on 3rd generation Data Mining (SoKD-09)*. 2009.

[3] Brachman, Ronald J., and Tej Anand. "The process of knowledge discovery in databases." *Advances in knowledge discovery and data mining*. 1996. 37-57.

[4] Chapman, Pete, et al. "CRISP-DM 1.0: Step-by-step data mining guide." *SPSS inc* 9 (2000): 13.

[5] SAS Enterprise Miner – SEMMA. SAS Institute, 2014 [online] available: http://www.sas.com/technologies/analytics/datamining/miner/semma.html (September 2014.)

[6] Bhatt, Mehul, et al. "A distributed approach to sub-ontology extraction." *18th International Conference on Advanced Information Networking and Applications, 2004. AINA 2004.*. Vol. 1. IEEE, 2004.

[7] Serban, Floarea, et al. "A survey of intelligent assistants for data analysis." *ACM Computing Surveys (CSUR)* 45.3 (2013): 1-35.

[8] Ristoski, Petar, and Heiko Paulheim. "Semantic Web in data mining and knowledge discovery: A comprehensive survey." *Journal of Web Semantics* 36 (2016): 1-22.

[9] Panov, P., Džeroski, S., & Soldatova, L. (2008, December). OntoDM: An ontology of data mining. In *2008 IEEE International Conference on Data Mining Workshops* (pp. 752-760). IEEE.

[10] Panov, Panče, Sašo Džeroski, and Larisa N. Soldatova. "Representing entities in the OntoDM data mining ontology." *Inductive Databases and Constraint-Based Data Mining*. Springer, New York, NY, 2010. 27-58.

[11] Ristoski, P., & Paulheim, H. (2016). Semantic Web in data mining and knowledge discovery: A comprehensive survey. *Journal of Web Semantics*, *36*, 1-22.

[12] Panov, Panče, Larisa N. Soldatova, and Sašo Džeroski. "Generic ontology of datatypes." *Information Sciences* 329 (2016): 900-920.

[13] Hilario, Melanie, et al. "A data mining ontology for algorithm selection and meta-mining." *Proceedings of the ECML/PKDD09 Workshop on 3rd generation Data Mining (SoKD-09)*. 2009.

[14] Záková, Monika, et al. "Automating knowledge discovery workflow composition through ontology-based planning." *IEEE Transactions on Automation Science and Engineering* 8.2 (2010): 253-264.

[15] Diamantini, Claudia, Domenico Potena, and Emanuele Storti. "Kddonto: An ontology for discovery and composition of kdd algorithms." *Third Generation Data Mining: Towards Service-Oriented Knowledge Discovery (SoKD'09)* (2009): 13-24.

[16] Kietz, Jörg-Uwe, et al. "Towards cooperative planning of data mining workflows." (2009).

[17] Bagnall, Anthony, et al. "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances." *Data Mining and Knowledge Discovery* 31.3 (2017): 606-660.

[18] Tianxing, Man, Nataly Zhukova, and Nikolay Mustafin. "A Knowledge-based Recommendation System for Time Series Classification." *Proceedings of the 24th Conference of Open Innovations Association FRUCT*. FRUCT Oy, 2019.

[19] Corbett, Dan. "Interoperability of ontologies using conceptual graph theory." *International Conference on Conceptual Structures*. Springer, Berlin, Heidelberg, 2004.

[20] McGuinness, Deborah L., et al. "An environment for merging and testing large ontologies." *KR*. 2000.

[21] Bakhtouchi, Abdelghani, et al. "MIRSOFT: mediator for integrating and reconciling sources using ontological functional dependencies." *International Journal of Web and Grid Services* 8.1 (2012): 72-110.

[22] Wouters, Carlo. *A formalization and application of ontology extraction*. Diss. La Trobe University, 2005.