Generating Workflow Graphs Using Typed Genetic Programming

Tomáš Křen[1], Martin Pilát[1], Klára Pešková[1], and Roman Neruda[2]

[1] Charles University in Prague, Faculty of Mathematics and Physics,
Malostranské nám. 25, Prague, Czech Republic

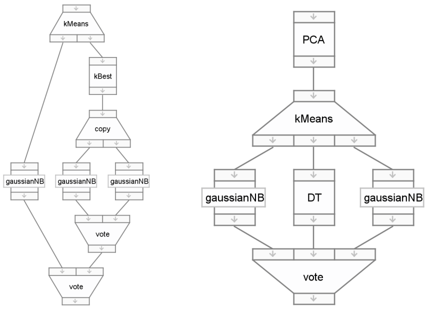
[2] Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod Vodárenskou věží 2, 18207 Prague, Czech Republic

In this paper we further develop our research line of chaining several preprocessing methods with classifiers [1], by generating the complete workflow schemes. These schemes, represented as directed acyclic graphs (DAGs), contain computational intelligence methods together with preprocessing algorithms and various methods of combining them into ensembles.

We systematically generate trees representing workflow DAGs using typed genetic programing initialization designed for polymorphic and parametric types. Terminal nodes of a tree correspond to the nodes of the DAG, where each node contains a computational intelligence method. Function nodes of a tree represent higher-order functions combining several DAGs in a serial or parallel manner. We use types to distinguish between input data (D) and predictions (P) so the generated trees represent meaningful workflows. In order to make the method general enough to handle methods like k-means (where k affects the topology of the DAG) correctly, we had to use the polymorphic type "*list of αs of size n*" ($[\alpha]_n$) with a natural number parameter n and an element type parameter α. The generating method systematically produces workflow DAGs from simple ones to more complex and larger ones, working efficiently with symmetries.

To demonstrate our first results we have chosen the winequality-white [2] and wilt [3] datasets from the UCI repository. They both represent medium size classification problems. The nodes of the workflow DAG contain three types of nodes; they can be *preprocessing* nodes (type $D \to D$) – k-Best (it selects k features most correlated with the target) or principal component analysis (PCA), or *classifier* nodes ($D \to P$) – gaussian naïve Bayes (gaussianNB), support vector classification (SVC), logistic regression (LR) or decision trees (DT). The last type of nodes implements *ensemble* methods – there is a copy node and a k-means node, which divides the data into clusters by the k-means algorithm (both $D \to [D]_n$), and two aggregating nodes – simple voting to combine the outputs of several methods, and merging for k-means node ($[P]_n \to P$).

To provide a baseline, we tested each of the four classifiers separately on the two selected datasets. The parameters of the classifiers were set using an extensive grid search with 5-fold cross-validation; the classifiers were compared using the quadratic weighted kappa metric. Next, we generated more than 65,000 different workflows using the proposed approach, and evaluated all of them. All computational intelligence methods used the default settings, or the tuned settings of the individual methods (denoted as 'default' or 'tuned' in Fig. 1c).



dataset	winequality		wilt	
params	default	tuned	default	tuned
SVC	0.1783	0.3359	0.0143	0.8427
LR	0.3526	0.3812	0.3158	0.6341
GNB	0.4202	0.4202	0.2916	0.2917
DT	0.3465	0.4283	0.7740	0.8229
workflow	**0.4731**	**0.4756**	**0.8471**	**0.8668**

(a) (b) (c)

Fig. 1: Best workflows for the winequality (a) and wilt (b) datasets, and comparison of κ metric from the cross-validation of the classifiers and the workflows (c).

The best workflows for the two datasets are presented in Figs. 1a and 1b, and their numerical results are presented in Fig.1c.

We have demonstrated how the valid workflow DAGs can be easily generated by a typed genetic programming initialization method. The generated workflows beat the baseline obtained by the hyper-parameter tuning of single classifier by a grid search, which is not surprising as the single method is also among the generated DAGs. On the other hand the workflows do not use any hyper-parameter tuning. In our future work, we will extend this approach to a full genetic programming solution, which will also optimize the hyper-parameters of the workflows and we intend to include the method in our multi-agent system for meta-learning – Pikater [4].

Acknowledgment

This work was supported by SVV project no. 260 224, GAUK project no. 187 115, and Czech Science Foundation project no. P103-15-19877S.

References

1. Kazík, O., Neruda, R.: Data mining process optimization in computational multi-agent systems. In: Agents and Data Mining Interaction. Volume 9145 of Lecture Notes in Computer Science. Springer (2015) 93–103
2. Cortez, P., Cerdeira, A., Almeida, F., Matos, T., Reis, J.: Modeling wine preferences by data mining from physicochemical properties. In Decision Support Systems, Elsevier **47**(4) (2009) 547–553
3. Johnson, B.A., Tateishi, R., Hoan, N.T.: A hybrid pansharpening approach and multiscale object-based image analysis for mapping diseased pine and oak trees. Int. J. Remote Sens. **34**(20) (October 2013) 6969–6982
4. Pešková, K., Šmíd, J., Pilát, M., Kazík, O., Neruda, R.: Hybrid multi-agent system for metalearning in data mining. In Vanschoren, J., Brazdil, P., Soares, C., Kotthoff, L., eds.: Proc. of the MetaSel@ECAI 2014. Volume 1201 of CEUR Workshop Proceedings., CEUR-WS.org (2014) 53–54