Transfer Learning for Algorithm Recommendation

Anonymous Author(s)
Affiliation
Address
email

1 Introduction

Meta-Learning is a subarea of Machine Learning that aims to take advantage of prior knowledge to learn faster and with fewer data [1]. There are different scenarios where meta-learning can be applied, and one of the most common is algorithm recommendation, where previous experience on applying machine learning algorithms for several datasets can be used to learn which algorithm, from a set of options, would be more suitable for a new dataset [2]. Perhaps the most popular form of meta-learning is transfer learning, which consists of transferring knowledge acquired by a machine learning algorithm in a previous learning task to increase its performance faster in another and similar task [3]. Transfer Learning has been widely applied in a variety of complex tasks such as image classification, machine translation and, speech recognition, achieving remarkable results [4, 5, 6, 7, 8]. Although transfer learning is very used in traditional or base-learning, it is still unknown if it is useful in a meta-learning setup. For that purpose, in this paper, we investigate the effects of transferring knowledge in the meta-level instead of base-level. Thus, we train a neural network on meta-datasets related to algorithm recommendation, and then using transfer learning, we reuse the knowledge learned by the neural network in other similar datasets from the same domain, to verify how transferable is the acquired meta-knowledge.

2 Algorithm recommendation problem and transfer learning

Meta-learning for algorithm recommendation: Meta-learning, or learning to learn, uses prior knowledge from a range of tasks, algorithms and model evaluations in order to perform better, faster and more efficient when applied to previously unseen data [9]. It is different from the traditional learning or base-learning, where the process of learning to induce a model is more focused on a specific task or dataset [10]. In the traditional machine learning setup, algorithms are trained using features already present in the datasets, whilst in meta-learning the algorithms or meta-learners use meta-features extracted from these original datasets [11]. A common meta-learning problem is the algorithm recommendation, where given a set of problem instances P from a distribution D, a set A of algorithms and a performance measure m: P x $A \rightarrow \mathbb{R}$, the algorithm recommendation problem consists of finding a mapping f: $P \rightarrow A$ that optimizes the expected performance measure m for instances P with a distribution D [3]. Even though algorithm recommendation is a well known problem, using meta-learning to explore prior knowledge and accelerate inference is a recent and potential trend which still requires exploration [2].

Transfer Learning: A formal definition of transfer Learning is given in terms of domains and learning tasks. Given a source domain D_s and target domain D_t, learning tasks T_s and T_t, transfer learning aims to improve the performance of T_t with knowledge obtained from a different but related domain D_s and learning task T_s, where $D_s \neq D_t$, or $T_s \neq T_t$ [12, 13].

Submitted to 33rd Conference on Neural Information Processing Systems (NeurIPS 2019). Do not distribute.

3 Experimental results

Setup: The datasets used in this paper were collected from Aslib [14], a repository associated with optimization problems such as the Traveling Salesman Problem (TSP), Quantified Boolean Formula (QBF) and Propositional Satisfiability Problem (SAT). We used 4 meta-datasets from Aslib related to the algorithm recommendation problem, which were preprocessed before training. Numerical features were normalized between 0 and 1, and targets were transformed in discrete numerical values. Since transfer learning requires certain standardization, the number of predictive features on the datasets were selected. Therefore, the Select K Best method [15] was used to select the K features with the highest score for a specific metric. The metric used was anova f-test, and K was the number of features from the dataset with fewer features. To handle class imbalance, we used a Stratified Hold-out validation method partitioning 80% of the original data for train and 20% for test. Our meta-learner was a MultiLayer Perceptron with two hidden layers optimized with Adam, Relu activation, Cross-entropy loss and He-et-al initialization. The neural network was trained 30 times on each configuration and mean values for accuracy and loss were extracted, as well as standard deviations. For the transfer learning setup, we used three configurations: (1) "freeze" two hidden layers, thus setting them to be not trainable and just using their weights; (2) freeze only the first hidden layer; (3) freeze no hidden layer, thus using weights to warm-start training on other datasets.

Results: In the headers of Table 1 are the datasets where transfer learning was applied and, above, the datasets that generated the pre-trained models. For example, the results above CSP-2010 (first row) refers to the final test accuracy and test loss obtained after the Normal training on CSP-2010, training using pre-trained models on CSP-MZN, CSP-Minizinc-Obj and CSP-Minizinc-Time, with the three transfer learning configurations (0HL, 1HL and 2HL). As seen for CSP-2010, using weights from the pre-trained model on CSP-MZN as a warm-start showed slightly better accuracy in comparison with Normal training, but resulted in considerably less loss. For CSP-MZN, the best results were obtained using transfer learning with two hidden layers from the pre-trained model of CSP-Minizinc-Obj, as well as for the pre-trained model of CSP-Minizinc-Time with one hidden layer frozen. Regarding CSP-Minizinc-Obj, although some executions showed competitive results, transfer learning configurations did not outperformed the Normal training. In the case of CSP-Minizinc-Time, the pre-trained model from CSP-MZN with two hidden layers frozen showed the best results in accuracy, quite surpassing the original/normal training. Our preliminary results suggests that: (1) transfer learning on the meta-level appears to be feasible and useful, achieving results comparable or superior to those obtained from training on original data; (2) transfer learning can be a potential tool for evaluating how general is the meta-knowledge leveraged by meta-learning approaches.

Table 1: Summary of the results. Columns 2HL stands for two hidden layers, which means that these layers were frozen. Columns 1HL stands for one hidden layer, meaning that the first hidden layer was not trained. Columns 0HL stands for no hidden layer, which means that no layer was frozen.

CSP-2010

	Normal	**CSP-MZN**			**CSP-Minizinc-Obj**			**CSP-Minizinc-Time**		
	-	0HL	1HL	2HL	0HL	1HL	2HL	0HL	1HL	2HL
Acc	0.87 ± 0.01	$\mathbf{0.88 \pm 0.01}$	0.87 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.86 ± 0.01	0.87 ± 0.01	0.87 ± 0.01	0.86 ± 0.01	0.85 ± 0.01
Loss	0.64 ± 0.15	$\mathbf{0.56 \pm 0.12}$	0.84 ± 0.08	1.01 ± 0.06	0.65 ± 0.15	0.90 ± 0.04	1.01 ± 0.09	0.65 ± 0.17	1.00 ± 0.06	1.07 ± 0.09

CSP-MZN

	Normal	**CSP-2010**			**CSP-Minizinc-Obj**			**CSP-Minizinc-Time**		
	-	0HL	1HL	2HL	0HL	1HL	2HL	0HL	1HL	2HL
Acc	0.71 ± 0.01	0.71 ± 0.01	0.70 ± 0.01	0.71 ± 0.01	0.71 ± 0.01	0.71 ± 0.01	$\mathbf{0.72 \pm 0.01}$	0.71 ± 0.01	$\mathbf{0.72 \pm 0.01}$	0.71 ± 0.01
Loss	1.23 ± 0.19	1.27 ± 0.22	1.71 ± 0.07	1.95 ± 0.07	1.25 ± 0.21	1.71 ± 0.08	1.98 ± 0.08	$\mathbf{1.18 \pm 0.18}$	1.63 ± 0.11	1.92 ± 0.06

CSP-Minizinc-Obj

	Normal	**CSP-2010**			**CSP-MZN**			**CSP-Minizinc-Time**		
	-	0HL	1HL	2HL	0HL	1HL	2HL	0HL	1HL	2HL
Acc	$\mathbf{0.91 \pm 0.02}$	0.87 ± 0.04	0.90 ± 0.00	0.90 ± 0.00	0.66 ± 0.02	0.70 ± 0.00	0.70 ± 0.00	0.90 ± 0.00	0.90 ± 0.00	0.90 ± 0.00
Loss	$\mathbf{0.77 \pm 0.06}$	1.01 ± 0.16	1.37 ± 0.11	1.55 ± 0.06	3.26 ± 0.06	3.30 ± 0.01	3.27 ± 0.01	0.79 ± 0.11	1.01 ± 0.10	1.37 ± 0.15

CSP-Minizinc-Time

	Normal	**CSP-2010**			**CSP-MZN**			**CSP-Minizinc-Obj**		
	-	0HL	1HL	2HL	0HL	1HL	2HL	0HL	1HL	2HL
Acc	0.65 ± 0.00	0.65 ± 0.01	0.65 ± 0.00	0.65 ± 0.00	0.67 ± 0.03	0.70 ± 0.00	$\mathbf{0.73 \pm 0.02}$	0.70 ± 0.00	0.70 ± 0.00	0.70 ± 0.00
Loss	4.11 ± 0.60	3.78 ± 0.92	5.24 ± 0.18	5.59 ± 0.04	$\mathbf{3.27 \pm 0.30}$	3.81 ± 0.10	4.04 ± 0.03	3.87 ± 0.74	4.54 ± 0.07	4.60 ± 0.02

References

[1] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Automated machine learning-methods, systems, challenges, 2019.

[2] Edesio Alcobaça, Rafael G Mantovani, André LD Rossi, and André CPLF de Carvalho. Dimensionality reduction for the algorithm recommendation problem. In *2018 7th Brazilian Conference on Intelligent Systems (BRACIS)*, pages 318–323. IEEE, 2018.

[3] John R Rice. The algorithm selection problem. In *Advances in computers*, volume 15, pages 65–118. Elsevier, 1976.

[4] Catherine Wong, Neil Houlsby, Yifeng Lu, and Andrea Gesmundo. Transfer learning with neural automl. In *Advances in Neural Information Processing Systems*, pages 8356–8365, 2018.

[5] Valerio Perrone, Rodolphe Jenatton, Matthias W Seeger, and Cédric Archambeau. Scalable hyperparameter transfer learning. In *Advances in Neural Information Processing Systems*, pages 6845–6855, 2018.

[6] Tyler Scott, Karl Ridgeway, and Michael C Mozer. Adapted deep embeddings: A synthesis of methods for k-shot inductive transfer learning. In *Advances in Neural Information Processing Systems*, pages 76–85, 2018.

[7] Simon S Du, Jayanth Koushik, Aarti Singh, and Barnabás Póczos. Hypothesis transfer learning via transformation functions. In *Advances in Neural Information Processing Systems*, pages 574–584, 2017.

[8] Wataru Kumagai. Learning bound for parameter transfer learning. In *Advances in neural information processing systems*, pages 2721–2729, 2016.

[9] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1126–1135. JMLR. org, 2017.

[10] Pavel Brazdil, Christophe Giraud Carrier, Carlos Soares, and Ricardo Vilalta. *Metalearning: Applications to data mining*. Springer Science & Business Media, 2008.

[11] Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Automatic machine learning: methods, systems, challenges. *Challenges in Machine Learning*, 2019.

[12] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[13] Lorien Pratt and Barbara Jennings. A survey of transfer between connectionist networks. *Connection Science*, 8(2):163–184, 1996.

[14] Bernd Bischl, Pascal Kerschke, Lars Kotthoff, Marius Lindauer, Yuri Malitsky, Alexandre Fréchette, Holger Hoos, Frank Hutter, Kevin Leyton-Brown, Kevin Tierney, et al. Aslib: A benchmark library for algorithm selection. *Artificial Intelligence*, 237:41–58, 2016.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.