Inductive Transfer

Ricardo Vilalta
Department of Computer Science, University of Houston, USA

Christophe Giraud-Carrier
Department of Computer Science, Brigham Young University, USA

Pavel Brazdil, Carlos Soares
LIAAD-INESC Porto L.A./Faculdade de Economia, University of Porto, Portugal

Synonyms

Transfer learning, transfer of knowledge across domains, multitask learning

Definition

Inductive transfer refers to the ability of a learning mechanism to improve performance on the current task after having learned a different but related concept or skill on a previous task. Transfer may additionally occur between two or more learning tasks that are being undertaken concurrently. Transfer may include background knowledge or a particular form of search bias.

As an illustration, an application of inductive transfer shows in competitive games involving teams of robots (e.g., Robocup Soccer). In this scenario, transferring knowledge learned from one task into another task is crucial to acquire skills necessary to beat the opponent team. Specifically, imagine a situation where a team of robots has been taught to keep a soccer ball away from the opponent team. To achieve that goal, robots must learn to keep the ball, pass the ball to a close teammate, etc. always trying to remain at a safe distance from the opponents. Now let us assume that we wish to teach the same team of robots to play a different game where they must learn to score against a team of defending robots. Knowledge gained during the first activity can be transferred to the second one. Specifically, a robot can prefer to perform an action learned in the past over actions proposed during the current task because the past action has a significant higher merit value. For example, a robot under the second task may learn to recognize that it is preferable to shoot than to pass the ball because the goal is very close. This action can be learned from the first task by recognizing that the precision of a pass is contingent on the proximity of the teammate.

Structure of the System

The main idea behind a learning architecture using knowledge transfer is to produce a source model from which knowledge can be extracted and transferred to a target model. This allows for multiple scenarios (Brazdil, et. al. (2009) [2], Pratt and Thrun (1997) [6]). For example, the target and source models can be trained at different times such that the transfer takes place after the source model has been trained; in this case there is an explicit form of knowledge transfer, also called *representational transfer*. In contrast, we use the term *functional transfer* to denote

the case where two or more models are trained simultaneously; in this case the models share (part of) their internal structure during learning (see Neural Networks below). When the transfer of knowledge is explicit, we denote as *literal transfer* the case when the source model is left intact. In addition, we denote as *non-literal transfer* the case when the source model is modified before knowledge is transferred to the target model; in this case some processing step takes place on the model before it is used to initialize the target model.

Neural Networks. A learning paradigm amenable to test the feasibility of knowledge transfer is that of neural networks (Caruana (1993) [3]). A popular form of knowledge transfer is effected through multitask learning, where the output nodes in the multilayer network represent more than one task. In such a scenario, internal nodes are shared by different tasks dynamically during learning. As an illustration, consider the problem of learning to classify astronomical objects from images mapping the sky into multiple classes. One task may be in charge of classifying a star into several classes (e.g., main sequence, dwarf, red giant, neutron, pulsar, etc.). Another task can focus on galaxy classification (e.g., spiral, barred spiral, elliptical, irregular, etc.). Rather than separating the problem into different tasks where each task is in charge of identifying one type of luminous object, one can combine the tasks together into a single parallel multi-task problem where the hidden layer of a neural network shares patterns that are common to all classification tasks (see Fig. 1). The reasons explaining why learning often improves in accuracy and speed in this context is that training with many tasks in parallel on a single neural network induces information that accumulates in the training signals; if there exists properties common to several tasks, internal nodes can serve to represent common sub-concepts simultaneously.

Other Paradigms. Knowledge transfer can be performed using other learning and data-analysis paradigms such as kernel methods, probabilistic methods, clustering, etc (Raina, et al. (2006) [7], Evgeniou, et al. (2007) [11]). For example, inductive transfer can take place in learning methods that assume a probabilistic distribution of the data by guaranteeing a form of relatedness among the distributions adopted across tasks (Raina, et al. (2006) [7]). As an illustration, if learning to classify stars and galaxies both assume a mixture of normal densities to model the example-class distribution, one can force both distributions to have sets of parameters that are as similar as possible while preserving good generalization performance. In that case, shared knowledge can be interpreted as a set of assumptions about the data distribution for all tasks under analysis. The knowledge-transfer concept is also related to the problem of introducing new intermediate concepts in the process of bottom-up induction of rules. In the Inductive Logic Programming (ILP) setting this is referred to as *predicate invention* (Stahl (1995) [12]).

Meta-Searching for Problem Solvers. A different research direction in inductive transfer explores complex scenarios where the software architecture itself evolves with experience (Schmidhuber (1997) [9]). The main idea is to divide a program into different components that can be re-used during different stages of the learning process. As an illustration, one can work within the space of (self-delimiting binary) programs to propose an optimal ordered problem solver. The goal is to solve a sequence of problems, deriving one solution after the other, as optimally as possible; ideally the system should be capable of exploiting previous solutions and incorporate them into the solution to the current problem. This can be done by allocating computing time to the search for previous solutions that, if useful, become transformed into building blocks. We assume the current problem can be solved by copying or invoking previous pieces of code (i.e., building blocks or knowledge). In that case the mechanism will accept those solutions with substantial savings in computational time.

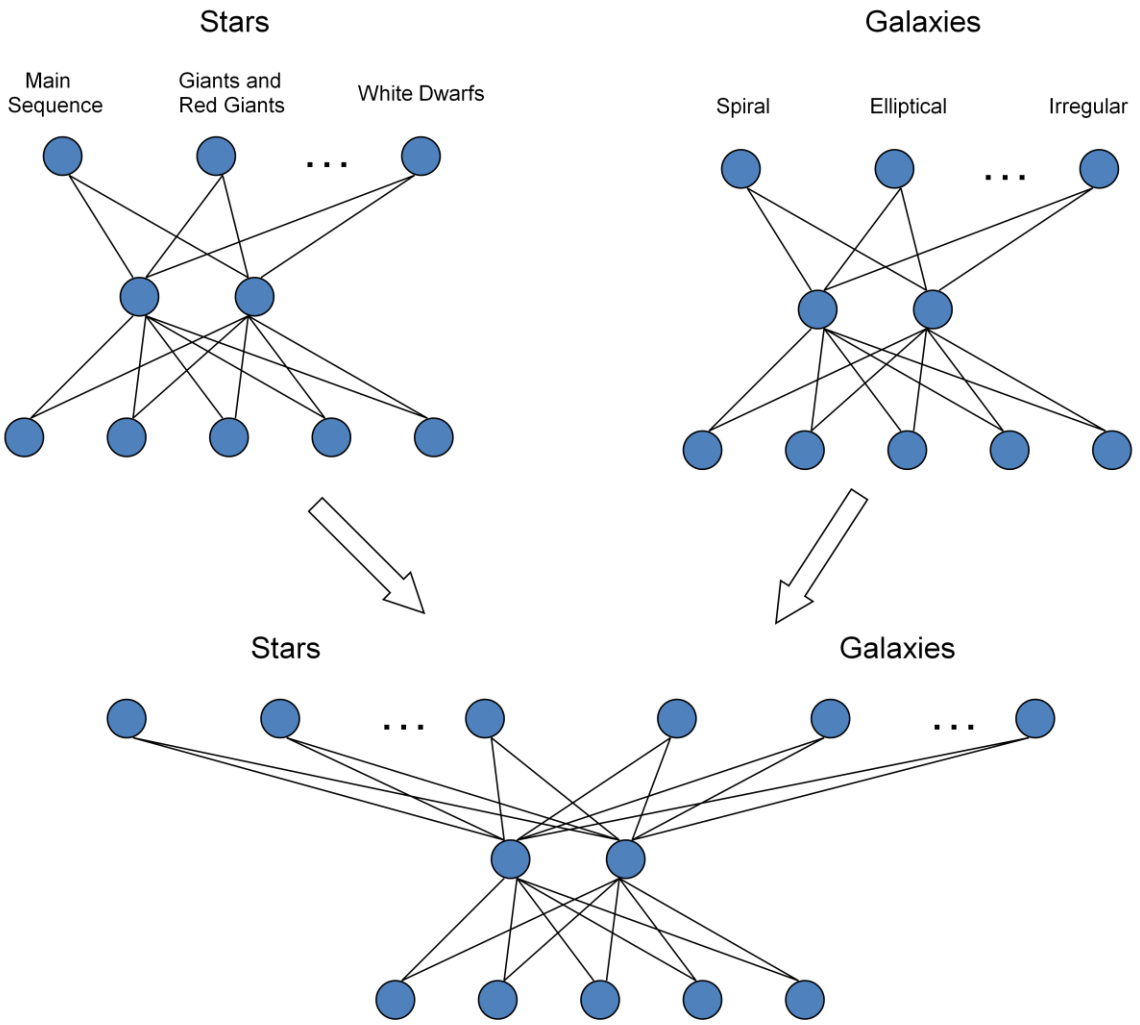

Fig.1. Example of multitask learning on astronomical images.

Theoretical Work

Several studies have provided a theoretical analysis of the case where a learner uses experience from previous tasks to learn a new task. This process is often referred to as metalearning. The aim is to understand the conditions under which a learning algorithm can provide good generalizations when embedded in an environment made of related tasks. Although the idea of knowledge transfer is normally made implicit in the analysis, it is clear that the metalearner extracts and exploits knowledge on every task to perform well on future tasks. Theoretical studies fall within a Bayesian model, and within a Probably Approximately Correct (PAC) model. The idea is to find not only the right hypothesis in a hypothesis space (base learning), but in addition to find the right hypothesis space in a family of hypothesis spaces (metalearning).

Let us review the main ideas behind these studies (Baxter (2000) [1]). We begin by assuming the learner is embedded in a set of related tasks that share certain commonalities. Going back to the problem where a learner is designed for recognition of astronomical objects, the idea is to

classify objects (e.g., stars, galaxies, nebulae, planets) extracted from images mapping certain region of the sky. One way to transfer learning experience from one astronomical center to another is by sharing a metalearner that carries a bias towards recognition of astronomical objects. In traditional learning, we assume a probability distribution **p** that indicates which examples are more likely to be seen in such task. Now we assume there is a more general distribution **P** over the space of all possible distributions. In essence, the metadistribution **P** indicates which tasks are more likely to be found within the sequence of tasks faced by the metalearner (just as an example distribution **p** indicates which examples are more likely to be seen in one task). In our example, the metadistribution **P** peaks over tasks corresponding to classification of astronomical objects. Given a family of hypothesis spaces **{H}**, the goal of the metalearner is to find a hypothesis space **H*** that minimizes a functional risk corresponding to the expected loss of the best possible hypothesis in each hypothesis space. In practice, since we ignore the form of **P** we need to draw samples $\mathbf{T_1, T_2, \ldots, T_n}$ to infer how tasks are distributed in our environment. To summarize, in the transfer-learning scenario our input is made of samples $\mathbf{T = \{T_i\}}$, where each sample $\mathbf{T_i}$ is composed of examples. The goal of the metalearner is to output a hypothesis space with a learning bias that generates accurate models for a new task.

Future Directions

The research community faces several challenges on how to efficiently transfer knowledge across tasks. One challenge involves devising learning architectures with an explicit representation of knowledge about models and algorithms, i.e., metaknowledge. Most systems that integrate knowledge transfer mechanisms make an implicit assumption about the transfer process by modifying the bias embedded by the hypothesis space. For example, we may change bias by selecting a learning algorithm that draws linear boundaries over the input space instead of one that draws quadratic boundaries; here no explicit knowledge is transferred specifying our preference for linear boundaries. Because of this limitation, transferring knowledge across domains becomes problematic.

Another challenge is to understand why a learning algorithm performs well or not on certain datasets, and to use that (meta)knowledge to improve its performance. Recent work in metalearning has explored the idea that high-quality dataset characteristics or metafeatures provide enough information to differentiate the performance of a set of given learning algorithms. From a practical perspective, a proper characterization of datasets leads to an interesting goal: the construction of metalearning assistants. The main role of these assistants is to recommend a good predictive model given a new dataset, or to attempt to modify the learning mechanism before it is invoked again in a dataset drawn from a similar distribution.

See Also: Metalearning

Recommended Readings

1. Baxter, J. (2000). A Model of Inductive Learning Bias. *Journal of Artificial Intelligence Research*, **12**, pp. 149-198.

2. Brazdil, P. and Giraud-Carrier, C. and Soares, C. and Vilalta, R. (2009). *Metalearning: Applications to Data Mining*. Springer.

3. Caruana, R. (1993). Multitask Learning: A Knowledge-based Source of Inductive Bias. In *Proceedings of the Tenth International Conference on Machine Learning*, pp. 41-48.

4. Mihalkova, L. and Huynh, T. and Mooney, R.J. (2007). Mapping and Revising Markov Logic Networks for Transfer Learning. In *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*, pp. 608-614.

5. Oblinger, D. and Reid, M. and Brodie, M. and de Salvo Braz, R. (2002). Cross-training and its Application to Skill-Mining. *IBM Systems Journal* **41**, No. 3, pp. 449-460.

6. Pratt, L. and Thrun, S. (1997). Second Special Issue on Inductive Transfer. *Machine Learning*, **28**.

7. Raina, R. and Ng, A. Y. and Koller, D. (2006). Constructing Informative Priors using Transfer Learning. In *Proceedings of the Twenty-third International Conference on Machine Learning*.

8. Reid, M. (2004). Improving Rule Evaluation Using Multitask Learning. In *Proceedings of the 14th International Conference on ILP*, pp. 252-269.

9. Schmidhuber, J. (1997). Shifting Inductive Bias with Success-Story Algorithm, Adaptive Levin Search, and Incremental Self-Improvement. *Machine Learning*, **28**, pp. 105-130.

10. Dai, W. and Yang, Q. and Xue, G. and Yu, Y.. (2007). Boosting for Transfer Learning. In *Proceedings of the 24th Annual International Conference on Machine Learning* pp. 193–200.

11. Evgeniou T. and Micchelli, C. A. and Pontil, M. (2005). Learning Multiple Tasks with Kernel Methods. *Journal of Machine Learning Research*, **6**, pp. 615-637.

12. Stahl. I. (1995). Predicate Invention in Inductive Logic Programming. In L. De Raedt (ed.) Advances in Inductive Logic Programming, pp. 34-47. IOS Press.