Transfer Learning

Sohan Kumar Parida (18IE10025)
Gaurang Kaushik Mohta (18IE10033)
Payel Kumari Agarwal (18IE10036)
Jagriti Agarwal (18IE10037)

INTRODUCTION

Human learners appear to have inherent ways to transfer knowledge between tasks. That is, we recognize and apply relevant knowledge from previous learning experiences when we encounter new tasks. The more related a new task is to our previous experience, the more easily we can master it.

Common machine learning algorithms, in contrast, traditionally address isolated tasks. *Transfer learning* attempts to change this by developing methods to transfer knowledge learned in one or more *source tasks* and use it to improve learning in a related *target task* (see Figure 1). Techniques that enable knowledge transfer represent progress towards making machine learning as efficient as human learning.

Transfer methods tend to be highly dependent on the machine learning algorithms being used to learn the tasks, and can often simply be considered extensions of those algorithms. Some work in transfer learning is in the context of inductive learning, and involves extending well-known classification and inference algorithms such as neural networks, Bayesian networks, and Markov Logic Networks.

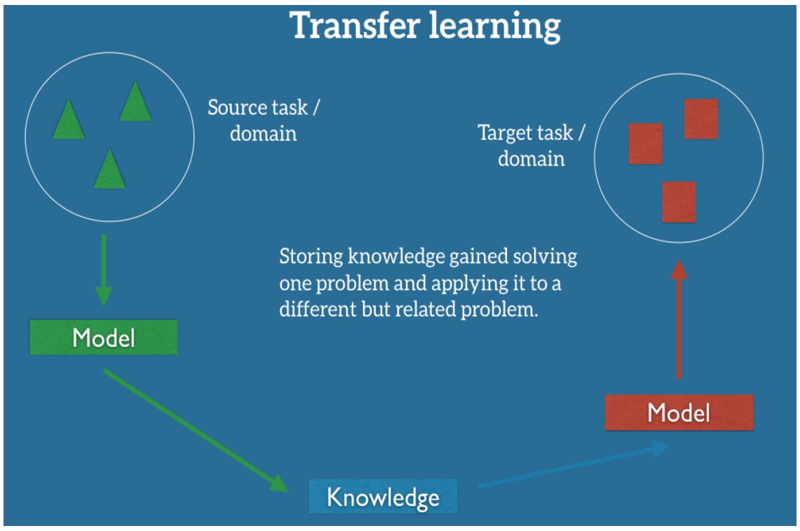
Transfer learning
Source task / domain
Target task / domain
Storing knowledge gained solving one problem and applying it to a different but related problem.
Model
Model
Knowledge

Fig.1. Transfer learning is a research problem in machine learning that focuses on storing knowledge gained while solving one problem and applying it to a different but related problem.

The goal of transfer learning is to improve learning in the target task by leveraging knowledge from the source task. There are three common measures by which transfer might improve learning. First is the initial performance achievable in the target task using only the transferred knowledge, before any further learning is done, compared to the initial performance of an ignorant agent. Second is the amount of time it takes to fully learn the target task given the transferred knowledge compared to the amount of time to learn it from scratch. Third is the final performance level achievable in the target task compared to the final level without transfer.

If a transfer method actually decreases performance, then *negative transfer* has occurred. One of the major challenges in developing transfer methods is to produce positive transfer between appropriately related tasks while avoiding negative transfer between tasks that are less related. A section of this chapter discusses approaches for avoiding negative transfer. Figure 2 illustrates the things discussed above..

When an agent applies knowledge from one task to another, it is often necessary to map the characteristics of one task onto those of the other to specify correspondences. In much of the work on transfer learning, a human provides this *mapping*, but some methods provide ways to perform the mapping automatically. Another section of the chapter discusses work in this area.

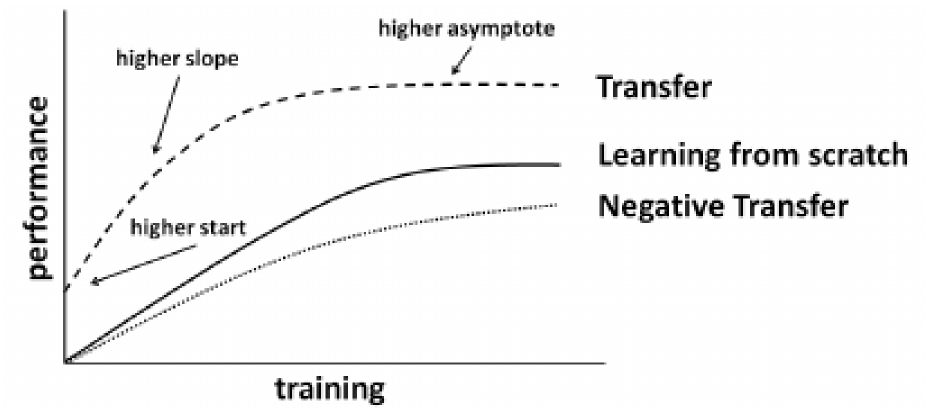

Fig.2. Ways in which transfer might affect learning.

TRANSFER IN INDUCTIVE LEARNING

In an inductive learning task, the objective is to induce a predictive model from a set of training examples.. Often the goal is classification, i.e. assigning class labels to examples. Examples of classification systems are artificial neural networks and symbolic rule-learners. Another type of inductive learning involves modeling probability distributions over interrelated variables, usually with graphical models. Examples of these systems are Bayesian networks and Markov Logic Networks..

The predictive model learned by an inductive learning algorithm should make accurate predictions not just on the training examples, but also on future examples that come from the same distribution. In order to produce a model with this generalization capability, a learning algorithm must have an *inductive bias* – a set of assumptions about the true distribution of the training data.

The bias of an algorithm is often based on the *hypothesis space* of possible models that it considers. For example, the hypothesis space of the Naive Bayes model is limited by the assumption that example characteristics are conditionally independent given the class of an example. The bias of an algorithm can also be determined by its search process through the hypothesis space, which determines the order in which hypotheses are considered. For example, rule-learning algorithms typically construct rules one predicate at a time, which reflects the assumption that predicates contribute significantly to example coverage by themselves rather than in pairs or more.

Transfer in inductive learning works by allowing source-task knowledge to affect the target task's inductive bias. It is usually concerned with improving the speed with which a model is learned, or with improving its generalization capability. The next subsection discusses inductive transfer, and the following ones elaborate on three specific settings for inductive transfer.

Inductive Transfer

In *inductive transfer* methods, the target-task inductive bias is chosen or adjusted based on the source-task knowledge (see Figure 4). The way this is done varies depending on which inductive learning algorithm is used to learn the source and target tasks. Some transfer methods narrow the hypothesis space, limiting the possible models, or remove search steps from consideration. Other methods broaden the space, allowing the search to discover more complex models, or add new search steps.

Baxter frames the transfer problem as that of choosing one hypothesis space from a family of spaces. By solving a set of related source tasks in each hypothesis space of the family and determining which one produces the best overall generalization error, he selects the most promising space in the family for a target task. He derives bounds on the number of source tasks and examples needed to learn an inductive bias, and on the generalization capability of a target-task solution given the number of source tasks and examples in each task.

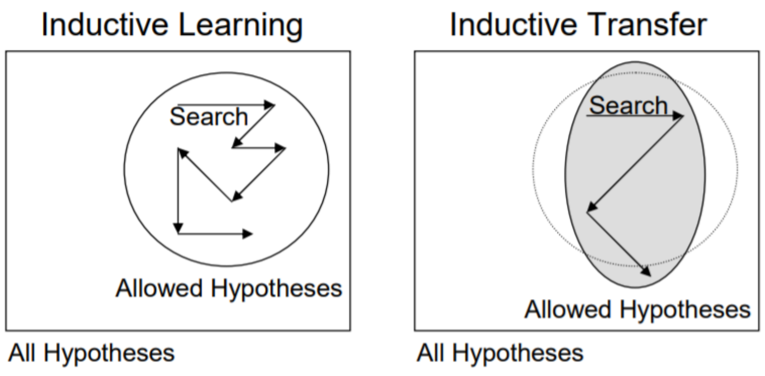

Fig.4. Inductive learning is a process where the learner discovers rules by observing examples. Inductive transfer refers to the ability of a learning mechanism to improve performance on the current task after having learned a different but related concept or skill on a previous task.

Thrun and Mitchell look at solving Boolean classification tasks in a lifelong-learning framework, where an agent encounters a collection of related problems over its lifetime. They learn each new task with a neural network, but they enhance the standard gradient-descent algorithm with slope information acquired from previous tasks. This speeds up the search for network parameters in a target task and biases it towards the parameters for previous tasks.

Mihalkova and Mooney perform transfer between Markov Logic Networks. Given a learned MLN for a source task, they learn an MLN for a related target task by starting with the source-task one and diagnosing each formula, adjusting ones that are too general or too specific in the target domain. The hypothesis space for the target task is therefore defined in relation to the sourcetask MLN by the operators that generalize or specify formulas.

Hlynsson phrases transfer learning in classification as a minimum description length problem given source-task hypotheses and target-task data. That is, the chosen hypothesis for a new task can use hypotheses for old tasks but stipulate exceptions for some data points in the new task. This method aims for a tradeoff between accuracy and compactness in the new hypothesis.

Ben-David and Schuller propose a transformation framework to determine how related two Boolean classification tasks are. They define two tasks as related with respect to a class of transformations if they are equivalent under that class; that is, if a series of transformations can make one task look exactly like the other. They provide conditions under which learning related tasks concurrently requires fewer examples than single-task learning.

Bayesian Transfer

One area of inductive transfer applies specifically to Bayesian learning methods. Bayesian learning involves modeling probability distributions and taking advantage of conditional independence among variables to simplify the model. An additional aspect that Bayesian models often have is a *prior distribution*, which describes the assumptions one can make about a domain before seeing any training data. Given the data, a Bayesian model makes predictions by combining it ___with the prior distribution to produce a *posterior distribution*. A strong prior can significantly affect these results (see Figure 5). This serves as a natural way for Bayesian learning methods to incorporate prior knowledge – in the case of transfer learning, source-task knowledge.

Marx et al. use a Bayesian transfer method for tasks solved by a logistic regression classifier. The usual prior for this classifier is a Gaussian distribution with a mean and variance set through cross-validation. To perform transfer, they instead estimate the mean and variance by averaging over several source tasks. Raina et al. use a similar approach for multi-class classification by learning a multivariate Gaussian prior from several source tasks.

Dai et al. apply a Bayesian transfer method to a Naive Bayes classifier. They set the initial probability parameters based on a single source task, and revise them using target-task data. They also provide some theoretical bounds on the prediction error and convergence rate of their algorithm.

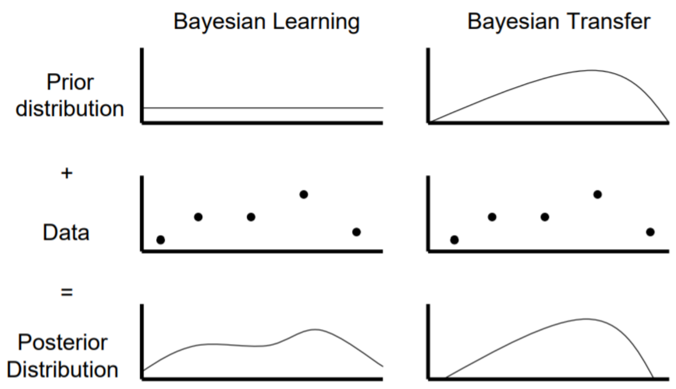

Fig.5. Bayesian learning uses a prior distribution to smooth the estimates from training data. Bayesian transfer may provide a more informative prior from source-task knowledge.

Hierarchical Transfer

Another setting for transfer in inductive learning is *hierarchical transfer*. In this setting, solutions to simple tasks are combined or provided as tools to produce a solution to a more complex task (see Figure 6). This can involve many tasks of varying complexity, rather than just a single source and target. The target task might use entire source-task solutions as parts of its own, or it might use them in a more subtle way to improve learning.

Sutton and McCallum begin with a sequential approach where the prediction for each task is used as a feature when learning the next task. They then proceed to turn the problem into a multi-task learning problem by combining all the models and applying them jointly, which brings their method outside our definition of transfer learning, but the initial sequential approach is an example of hierarchical transfer.

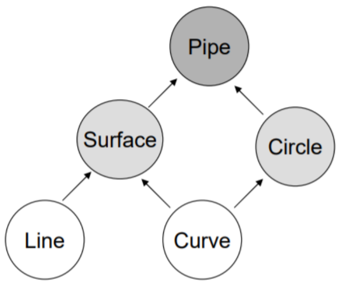

Fig.6. In hierarchical transfer, simple tasks are used to help learn a more complex task. Example, in this figure, simple tasks include recognizing lines and curves in images. The more complex tasks include recognizing surfaces, circles, and finally pipe shapes.

Stracuzzi looks at the problem of choosing relevant source-task Boolean concepts from a knowledge base to use while learning more complex concepts. He learns rules to express concepts from a stream of examples, allowing existing concepts to be used if they help to classify the examples, and adds and removes dependencies between concepts in the knowledge base.

Taylor et al. propose a transfer hierarchy that orders tasks by difficulty, so that an agent can learn them in sequence via inductive transfer. By putting tasks in order of increasing difficulty, they aim to make transfer more effective. This approach may be more applicable to the multi-task learning scenario, since by our definition of transfer learning the agent may not be able to choose the order in which it learns tasks, but it could be applied to help choose from an existing set of source tasks.

Transfer with Missing Data or Class Labels

Inductive transfer can be viewed not only as a way to improve learning in a standard supervised-learning task, but also as a way to offset the difficulties posed by tasks that involve unsupervised learning, semi-supervised learning, or small datasets. That is, if there are small amounts of data or class labels for a task, treating it as a target task and performing inductive transfer from a related source task can lead to more accurate models. These approaches therefore use source-task data to enhance target-task data, despite the fact that the two datasets are assumed to come from different probability distributions.

The Bayesian transfer methods of Dai et al. and Raina et al. are intended to compensate for small amounts of target-task data. One of the benefits of Bayesian learning is the stability that a prior distribution can provide in the absence of large datasets. By estimating a prior from related source tasks, these approaches prevent the overfitting that would tend to occur with limited data.

Dai et al. address transfer learning in a boosting algorithm using large amounts of data from a previous task to supplement small amounts of new data. Boosting is a technique for learning several weak classifiers and combining them to form a stronger classifier. After each classifier is learned, the examples are reweighted so that later classifiers focus more on examples the previous ones misclassified. Dai et al. extend this principle by also weighting source-task examples according to their similarity to target-task examples. This allows the algorithm to leverage source-task data that is applicable to the target task while paying less attention to data that appears less useful.

Shi et al. look at transfer learning in unsupervised and semi-supervised settings. They assume that a reasonably sized dataset exists in the target task, but it is largely unlabeled due to the expense of having an expert assign labels. To address this problem they propose an active learning approach, in which the target-task learner requests labels for examples only when necessary. They construct a classifier with labeled examples, including mostly source-task ones, and estimate the confidence with which this classifier can label the unknown examples. When the confidence is too low, they request an expert label.

AVOIDING NEGATIVE TRANSFER

Given a target task, the effectiveness of any transfer method depends on the source task and how it is related to the target. If the relationship is strong and the transfer method can take advantage of it, the performance in the target task can significantly improve through transfer. However, if the source task is not sufficiently related or if the relationship is not well leveraged by the transfer method, the performance with many approaches may not only fail to improve – it may actually decrease. This section examines work on preventing transfer from negatively affecting performance.

Ideally, a transfer method would produce positive transfer between appropriately related tasks while avoiding negative transfer when the tasks are not a good match. In practice, these goals are difficult to achieve simultaneously. Approaches that have safeguards to avoid negative transfer often produce a smaller

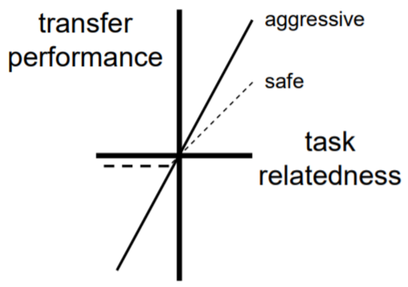

Fig.7. We see 2 cases here. For aggressive transfer (solid line), we have better performance for more task relatedness, but when task relatedness decreases, there is negative transfer learning. For safe transfer (dotted line), we have worse performance when compared to aggressive transfer for higher task relatedness, but for lower values of it, the negative transfer is limited.

effect from positive transfer due to their caution. Conversely, approaches that transfer aggressively and produce large positive-transfer effects often have no protection against negative transfer (see Figure 7).

Rejecting Bad Information

One way of approaching negative transfer is to attempt to recognize and reject harmful source-task knowledge while learning the target task. The goal in this approach is to minimize the impact of bad information, so that the transfer performance is at least no worse than learning the target task without transfer. At the extreme end, an agent might disregard the transferred knowledge completely, but some methods also allow it to selectively reject parts and keep other parts.

Option-based transfer in reinforcement learning is an example of an approach that naturally incorporates the ability to reject bad information. Since options are treated as additional actions, the agent can choose to use them or not to use them; in *Q*-learning, for example, agents learn *Q*-values for options just as for native actions. If an option regularly produces poor performance, its *Q*-values will degrade and the agent will choose it less frequently. However, if an option regularly leads to good results, its *Q*-values will grow and the agent will choose it more often. Option-based transfer can therefore provide a good balance between achieving positive transfer and avoiding negative transfer.

A specific approach that incorporates the ability to reject bad information is the KBKR advice-taking algorithm for transfer in reinforcement learning.. Recall that KBKR approximates the *Q*-function with a support-vector machine and includes advice from the source task as a soft constraint. Since the *Q*-function trades off between matching the agent's experience and matching the advice, the agent can learn to disregard advice that disagrees with its experience.

Choosing a Source Task

There are more possibilities for avoiding negative transfer if there exists not just one source task, but a set of candidate source tasks. In this case the problem becomes choosing the best source task (see Figure 8). Transfer methods without much protection against negative transfer may still be effective in this scenario, as long as the best source task is at least a decent match.

An example of this approach is the previously-mentioned transfer hierarchy], which orders tasks by difficulty. Appropriate source tasks are usually less difficult than the target task, but not so much simpler that they contain little information. Given a task ordering, it may be possible to locate the position of the target task in the hierarchy and select a source task that is only moderately less difficult.

Talvitie and Singh use a straightforward method of selecting a previous Markov decision process to transfer. They run each candidate MDP in the target task for a fixed length of time and order them by their performance. Then they select the best one and continue with it, only proceeding down the list if the current MDP begins performing significantly worse than it originally appeared. This trial-and-error approach, though it may be costly in the aggregate number of training episodes needed, is simple and widely applicable.

Kuhlmann and Stone look at finding similar tasks when each task is specified in a formal language. They construct a graph to represent the elements and rules of a task. This allows them to find identical tasks by checking for graph isomorphism, and by creating minor variants of a target-task graph, they can also search for similar tasks. If they find an isomorphic match, they conduct value-function transfer.

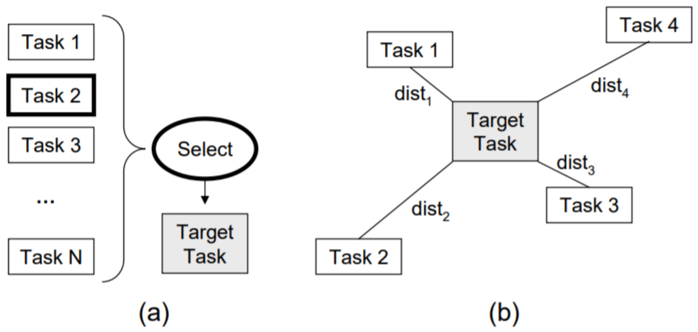

Fig. 8. (a) One way to avoid negative transfer is to choose a good source task from which to transfer. In this example, Task 2 is selected as being the most related. (b) Another way to avoid negative transfer is to model the way source tasks are related to the target task and combine knowledge from them with those relationships in mind.

Eaton and DesJardins propose choosing from among candidate solutions to a source task rather than from among candidate source tasks. Their setting is multi-resolution learning, where a classification task is solved by an ensemble of models that vary in complexity. Low-resolution models are simple and coarse, while higher-resolution models are more complex and detailed. They reason that high-resolution models are less transferable between tasks, and select a resolution below which to share models with a target task.

Modeling Task Similarity

Given multiple candidate source tasks, it may be beneficial to use several or all of them rather than to choose just one (see Figure 8). Some approaches discussed in this chapter do this naively, without evaluating how the source tasks are related to the target. However, there are some approaches that explicitly model relationships between tasks and include this information in the transfer method. This can lead to better use of source-task knowledge and decrease the risk of negative transfer.

Carroll and Seppi develop several similarity measures for reinforcement learning tasks, comparing policies, value functions, and rewards. These are only measurable while the target task is being learned, so their practical use in transfer scenarios is limited. However, they make the relevant point that task similarity is intimately linked with a particular transfer method, and cannot be evaluated independently.

Eaton et al. construct a graph in which nodes represent source tasks and distances represent a transferability metric. Given a new inductive learning task, they estimate parameters by fitting the task into the graph and learning a function that translates graph locations to task parameters. This method not only models the relationships between tasks explicitly, but also gives an algorithm for the informed use of several source tasks in transfer learning.

Ruckert and Kramer look at inductive transfer via kernel methods. They learn a meta-kernel that serves as a similarity function between tasks. Given this and a set of kernels that perform well in source tasks, they perform numerical optimization to construct a kernel for a target task. This approach determines the inductive bias in the target task (the kernel) by combining information from several source tasks whose relationships to the target are known.

THE FUTURE OF TRANSFER LEARNING

The challenges discussed in this chapter will remain relevant in future work on transfer learning, particularly the avoidance of negative transfer and the automation of task mapping. Humans appear to have mechanisms for deciding when to transfer information, selecting appropriate sources of knowledge, and determining the appropriate level of abstraction. It is not always clear how to make these decisions for a single machine learning algorithm, much less in general.

Another challenge for future work is to enable transfer between more diverse tasks. Davis and Domingos provide a potential direction for this in their work on MLN transfer. They perform pattern discovery in the source task to find second-order formulas, which represent universal abstract concepts like symmetry and transitivity. When learning an MLN for the target task, they allow the search to use the discovered formulas in addition to the original predicates in the domain. This approach is recognizable as inductive transfer, but the source-task knowledge is highly abstract, which allows the source and target tasks to differ significantly.

Transfer learning has become a sizable subfield in machine learning. It has ideological benefits, because it is seen as an important aspect of human learning, and also practical benefits, because it can make machine learning more efficient. As computing power increases and researchers apply machine learning to more complex problems, knowledge transfer can only become more desirable.