First Annual Conference on Advances in Cognitive Systems Poster Collection (2012) 53-68

Conceptual Semantics of Domain Knowledge in Learning by Correcting Mistakes

Joshua K. Jones JKJ@CC.GATECH.EDU
Ashok K. Goel GOEL@CC.GATECH.EDU
College of Computing, Georgia Institute of Technology, Atlanta, GA

Abstract

Meta-reasoning is an integral part of high-level cognition. Among other tasks, an intelligent agent may use meta-reasoning for learning by correcting mistakes in its domain knowledge. For example, given examples where its classification knowledge fails, a classification agent may use meta-reasoning to self-diagnose and self-repair the erroneous domain knowledge. We know from prior work that this kind of self-diagnosis and self-repair is facilitated if a hierarchical structure can be imposed on the agent's classification knowledge, based upon available background knowledge. In this paper, we examine the effect of imposing fixed semantics on the intermediate nodes of the hierarchy on classification learning, as distinct from the advantage offered by the structuring of knowledge. We present empirical results from ablation experiments that demonstrate the generalization power provided by verification procedures at the intermediate concepts in the hierarchy, which are demonstrated to go well beyond those provided by the structure of the hierarchy.

1. Introduction

Meta-reasoning is an integral part of high-level cognition (Brachman, 2002; Cox and Raja, 2011; Minsky et al., 2004). We know that metareasoning is useful for planning about reasoning (Wilensky, 1981; Cox and Ram, 1999), control of reasoning (Davis, 1980; Stefik, 1981; Hayes-Roth and Larsson, 1996; Punch et al., 1995; Hansen and Zilberstein, 2001; Raja and Lesser, 2007), bounding reasoning (Horvitz et al., 1989; Russell and Wefald, 1991; Horvitz, 2001), self-explanation (Goel and Murdock, 1996; Cox and Ram, 1999), revision of beliefs (Doyle, 1979), revision of reasoning processes (Birnbaum et al., 1990; Stroulia and Goel, 1995; Leake, 1996; Murdock and Goel, 2008), refinement of knowledge indices (Fox and Leake, 2001), and guidance of situated learning (Anderson et al., 2006; Ulam et al., 2008). Cox (2005) provides a useful review of some AI research on metareasoning; Cox and Raja (2011) provide a useful collection of papers on recent AI work on meta-reasoning.

As we describe in Goel and Jones (2011), our work on metareasoning since the early 1990s has focused mostly on self-adaptation in intelligent agents (e.g. Stroulia and Goel, 1996, 1997; Murdock and Goel, 2001a,b, 2003). The need for self-adaptation arises because intelligent agents typically operate in dynamic task environments. It is useful to make a few distinctions here. Firstly, adaptations to an agent can be retrospective (i.e. when the agent fails to achieve a goal in its given environment; Birnbaum et al., 1990; Stroulia and Goel, 1995, 1999; Leake, 1996), or proactive (i.e.

when the agent is asked to operate in a new task environment; Murdock and Goel, 2008). Secondly, adaptations can be either to the deliberative element in the agent architecture (Birnbaum et al., 1990; Stroulia and Goel, 1995; Leake, 1996; Murdock and Goel, 2008), or the reactive element (Stroulia and Goel, 1999), or both. Thirdly, adaptations to the deliberative element may be modifications to its reasoning process (i.e. to its task structure, selection of methods, or control of reasoning; e.g. Birnbaum et al., 1990; Stroulia and Goel, 1995; Murdock and Goel, 2008), or to its domain knowledge (i.e. the content, representation and organization of its knowledge; e.g. Leake, 1996; Jones and Goel, 2008), or both. Jones and Goel (2012) describe a technique for retrospective self-adaptation of domain knowledge in a deliberative agent.

Learning by correcting mistakes in domain knowledge is an important issue in cognitive systems research. Winston (1992), for example, describes a technique in which a recognition agent autonomously diagnoses and repairs its domain knowledge. Given a domain model for identifying an instance as belonging to a class, and a failure of the agent to correctly identify an instance, Winston's recognition agent uses positive and negatives examples of identification to localize and repair the fault in the domain model that led to the failure. It is important to note that Winston's technique maintains a well-defined semantics for the identification class at all times.

Our technique for learning by correcting mistakes significantly expands Winston's work in three dimensions (Jones and Goel, 2012): it addresses the much more complex task of hierarchical classification, it combines knowledge-based and statistical techniques for self-diagnosis of the fault in the domain model, and it grounds the self-diagnosis in perception. We consider the classification agent as situated in the world such that at classification time only data at the leaf nodes of the classification tree is available and the agent produces a label at the root node. However, at the time of self-diagnosis, the agent can use procedures to verify the semantics of the intermediate nodes in the hierarchy. We found that the use of such empirical verification procedures (or EVPs) that can define and fix the semantics of the intermediate nodes significantly facilitates self-diagnosis and self-repair.

An open question in prior work is the extent to which the use of empirical verification of the semantics of the intermediate nodes in the classification hierarchy improves the efficiency of classification learning beyond the advantage provided by the accompanying *structure* imposed upon the agent's knowledge representation. In this paper, we are specifically concerned with compositional classification (e.g. (Jones and Goel, 2008; Bylander et al., 1991; Goel and Bylander, 1989)), where prior knowledge in the form of a tree structure of classification subproblems is given, and where both the top level class label and the output values of the subproblems can be obtained during learning (but the functions computing the subproblems are not known to the knowledge engineer, and thus cannot be hard-coded). This task is so common in cognition (Chandrasekaran and Goel, 1988) that Chandrasekaran (1988) called it a Generic Task. Past research has shown that when this kind of tree-structured background knowledge is available, it can be exploited to increase the efficiency of learning, e.g., (Fu and Buchanan, 1985; Wogulis and Langley, 1989; Tadepalli and Russell, 1998). Here we are specifically concerned with empirically distinguishing the additional benefit to learning efficiency afforded by the use of verification procedures at internal nodes in a classification hierarchy from the benefit afforded by the imposition of a hierarchical structure on the knowledge.

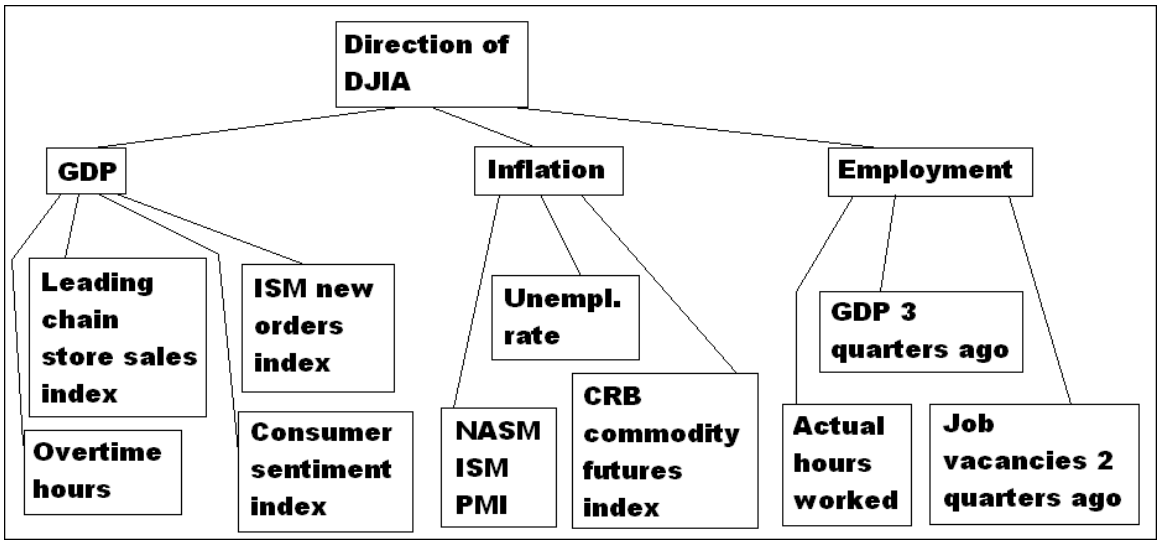

Figure 1. DJIA Classification Hierarchy

A number of learning techniques have been proposed that make use of background knowledge about problem structure to bias classification learning. Of these techniques, some enforce the semantics of intermediate concepts in the classification hierarchy during learning (e.g. Fu and Buchanan, 1985; Wogulis and Langley, 1989; TSB, Tadepalli and Russell, 1998; Layered Learning Whiteson et al., 2005; Structured Induction Shapiro, 1987; and BNs Pearl, 1988), while others do not (e.g. KBANN Towell and Shavlik, 1994; and EBNN Mitchell and Thrun, 1993). For example, consider the knowledge structure depicted in Figure 1 , which could be used to learn to predict the direction of the Dow Jones Industrial Average next month based on currently available financial data. The leaves in this hierarchy represent values obtainable at the time a prediction is produced (current and past values of various indicators). Non-leaf nodes in the hierarchy represent future values that are not known at the time that the prediction is produced – for example, next month's unemployment rate or the direction of the DJIA next month. Methods that enforce fixed semantics at internal nodes will ensure that the concepts learned at intermediate nodes match the semantics indicated by their annotations. This is done by constructing training examples for each node using target values that reflect the intended semantic. This means that, to enforce semantics at intermediate nodes in such a classification hierarchy, each internal concept learner must be able to somehow obtain or infer the correct (according to the prescribed semantics of the concept) local value for each global training example. In the case of the DJIA hierarchy depicted in Figure 1, this can be achived by waiting a month and then obtaining the true values of concept nodes for training purposes. Techniques that do not enforce fixed semantics at internal nodes may still make use of background knowledge to structure and initialize the knowledge representation, but make use of credit assignment techniques that do not modify knowledge at intermediate nodes based on local examples generated according to fixed semantics. For this reason, the concepts learned at internal nodes may, over time, cease to have identifiable semantics.

While it is clear based on this past research that background knowledge about classification problem structure can enhance the generalization power of a classification learner, the power provided by fixing the semantics of intermediate concepts in a classification hierarchy is less well understood. Intuitively, it seems as if enforcing fixed semantics for these concepts will reduce the expressive capability of the knowledge structure – and thus, some additional generalization power should be provided. However, because there is a measurable cost to using a technique that enforces fixed semantics at nodes within a classification hierarchy during learning (knowledge engineering cost, and more stringent demands on the form of the learning environment), it is important to go beyond intuition to tangible evidence regarding how much benefit is derived from the enforcement of fixed concept semantics *specifically*, as distinct from the associated use of structural background knowledge. We are aware of no existing studies that have individually examined the merits of these two related sources of bias. Instead, past research has made use of either some incarnation of structural+semantic knowledge or structural knowledge alone and analyzed the performance of the technique as a whole. Due to the preponderance of other variations between techniques, the specific impact of the enforcement of fixed semantics has remained unclear.

In this paper, we present empirical results using an existing, representative technique for compositional classification where we vary *only* the set of intermediate concepts for which fixed semantics are enforced. These results demonstrate the additional generalization power gained by enforcing fixed semantics of intermediate concepts within a classification hierarchy, beyond that provided by the structuring of knowledge. Specifically, we present experiments comparing the generalization performance of hierarchical classifiers that have fixed semantics at all internal nodes with other classifiers that have identical structure, but lack the enforcement of fixed semantics at one or more internal nodes. We refer to these learners with fixed semantics at only some internal nodes as *hybrid* learners, as they use a combination of diagnostic techniques to assign credit over nodes that do and do not have fixed semantics.

Below we first present an overview of the existing technique for compositional classification learning which was used to perform the experiments described in this paper. Then we turn to a description and discussion of the experiments themselves. The contribution of this paper is the individual analysis of the benefits of structural and semantic background knowledge in compositional classification, with the goal of providing guidance to machine learning practitioners on the value of investing effort in imbuing their hierarchical classification learning systems with fixed concept semantics.

2. Compositional Classification

Compositional classification is a common inference pattern within AI, and a variety of techniques have been proposed to learn classification knowledge that is hierarchically structured, as discussed above. Informally, a compositional classification problem is one in which a hierarchy of intermediate abstractions mediating between raw, observable state (the input feature vector) and the target classification, like that depicted in Figure 1, can reasonably be elaborated by a domain expert. The structure of these intermediate abstractions forms the background knowledge (provided by a knowledge engineer) that will be provided to the learner. In the experiments described in this paper,

we make use of *Abstraction Networks* (ANs), a specific technique for compositional classification learning. We provide an overview of ANs below. Formal definitions of both the compositional classification problem and ANs can be found in Jones and Goel (2008).

2.1 Representation in Abstraction Networks

The Abstraction Network knowledge structure contains a node for each intermediate concept as well as the top level target concept (the root of the hierarchy). These nodes are connected in a hierarchy that reflects direct dependence relationships according to background knowledge. Each node handles the subproblem of predicting the value of the concept with which it is associated, given the values of its children. For this purpose, each node will contain a classification learner. In principle, any supervised classification learner can be used within AN nodes. In this work, Artificial Neural Networks Rumelhart and Mcclelland (1986) (ANNs) are used. At the leaf nodes, raw inputs come directly from the environment in the form of an input feature vector, while at higher levels, values produced by child nodes become the inputs of the parent.

Normally, each node within an AN hierarchy has associated with it an *Empirical Verification Procedure* (EVP). For the purposes of the experiments described here, an EVP is an oracle that is able to obtain the correct value of the associated node for a given set of inputs to the node. When a node is trained using examples constructed by querying an associated EVP, the semantics of the node are fixed by the EVP – that is, the node will learn to match the concept encoded by the EVP. In these experiments, we will sometimes train ANs that have EVPs at some, but not all nodes. More details about the training mechanisms used are given below under "Learning".

2.2 Reasoning in Abstraction Networks

Performing inference using an AN is straightforward. The input feature vector is partitioned appropriately to produce the sub-vectors that form the inputs for each leaf of the AN concept hierarchy. Inference is performed at each leaf node, using the supervised classification learner within the node (here, the ANN within the node). The value produced is passed up to the parent to form part of its input vector. This process is repeated at each level in the hierarchy until the root node produces the overall output classification value.

2.3 Learning in Abstraction Networks

In the AN paradigm, learning is failure driven. This means that the learner is first presented with an input vector and asked to produce a classification, which it will do according to the procedure outlined under "Reasoning" above. Then, the true, desired classification is compared with that produced by the learner. If the value produced by the hierarchy was correct, no further action is taken. If the global classification produced by the root of the hierarchy is found to be erroneous, error is propagated backwards through the hierarchy according to the following diagnostic procedure, starting with the root node as the "current node":

1. If the current node is a leaf node, use the value returned from the current node's EVP to create a training example for the current node.

2. If the current node has one or more children with EVPs, execute each of those EVPs. If the value that was produced by one or more children was incorrect, recursively apply this procedure with each incorrect child as the "current node".
3. If all of the current node's children have EVPs, and none were found to be incorrect, use the value returned from the current node's EVP to create a training example for the current node.
4. Otherwise, the current node has one or more children without EVPs, and was found to have produced a value in error (only nodes with errors detected by EVP execution become the "current node"). Use standard error backpropagation through the current node into those children to effect reweighting of their ANN learners and of the local ANN learner.

This procedure is a kind of *mixed* backpropagation, where fixed, known semantics are used to identify the source of error where possible – that is, where nodes have associated EVP oracles – and statistical backpropagation is used in regions lacking EVPs. One will notice that, while this procedure is able to move from propagating error back through a region that has EVPs to one that lacks EVPs, the opposite move is not accounted for. That is, the procedure above gives no means to convert statistical backpropagation back into "semantic" backpropagation if a node with an EVP is a child of one without. In these experiments, we remove EVPs only from complete subtrees that include the leaves, so this situation does not occur. In general, this technique would need to be extended to cover such cases. Additionally, note that this procedure essentially treats regions of the AN that lack EVPs as deep ANNs, training those regions together via standard backpropagation if they are entered during diagnosis. In the work described here, we always used ANNs within all AN nodes. While ANN outputs are typically quantized by each node, in these experiments we eliminate the quantization of ANN outputs at nodes that do not have EVPs. This is necessary to cause those nodes to produce differentiable functions such that backpropagation can be used to push error back from ancestors into those nodes.

3. Experiments

We have performed experiments in a synthetic domain that investigate the generalization power provided to compositional classification learners by the use of fixed semantics for intermediate concepts, beyond that provided by the knowledge structure. That is, the experiments aim to isolate the inductive bias provided by the *structure* of the intermediate concepts in a classification hierarchy from that provided by the fixed semantics of those intermediate concepts. In the AN framework, fixed semantics are enforced by the EVPs associated with concept nodes. In these experiments, we will remove the EVPs from some nodes within an AN while retaining the structure, and observe the impact on learning.

The target knowledge in the synthetic domain is represented using a fixed abstraction network, over which no learning will occur, that is defined as representing the correct, target content (and structure) for the problem. Given this fixed AN, we then create a separate *learner* AN that will be initialized with incorrect knowledge content and expected to learn to produce the same top-level classifications as the fixed AN. This is implemented by initializing the knowledge content of both the fixed and learner AN nodes separately with pseudo-random values. The randomly-generated content

of the fixed AN forms the target knowledge for the learner AN. Training proceeds by repeating the following steps:

1. Generate a pseudo-random sequence of integers within a defined range to serve as the input vector.
2. Perform inference with the fixed AN, saving the values produced by all nodes (intermediate and root).
3. Perform inference with the learner AN.
4. Perform diagnosis and learning over the learner AN according to the procedure described in Section 2.3.

EVPs are not needed in the fixed AN, since no learning occurs in the fixed AN. EVPs in the learning AN are set up to examine the saved output value from the corresponding node in the fixed AN. Thus, the fixed semantics of the concept nodes in the learner AN are enforced (grounded) by reference to the knowledge encoded in the fixed AN. This is reasonable in a synthetic domain because the generator provides the environment for learning. In a non-synthetic domain, the EVPs would instead need to obtain values from a "real" environment.

In these experiments, we sometimes remove the EVPs from some of the learner AN's internal nodes, and train by executing standard error backpropagation Rumelhart and Mcclelland (1986) over those sections of the structure from which EVPs have been removed, as described above. We will refer to these ANs with some EVPs removed as *hybrid* learners, as they combine standard backpropagation with EVP-based diagnosis. An ANN learner is used within each AN node. In each experiment, we used fixed and learner ANs of four levels with binary tree structures (level sizes 8-4-2-1). In each case, we allow 3 values to be produced by each node in the hierarchy, including the root. We also trained a standard ANN learner (operating outside the AN framework) on the same problem, for the sake of providing a baseline that uses neither knowledge-based structure nor fixed semantics. We refer to this learner as the "flat" learner, as it does not make use of knowledge-based structure. We have run experiments with EVPs at all nodes; one EVP removed at the layer immediately above the leaves; two EVPs removed at peer nodes within the layer immediately above the leaves; and three EVPs removed in the form of a connected subtree rooted in the layer immediately beneath the root, thus also involving two peer nodes in the layer immediately above the leaves. For each of these learner setups, we experimented with various training set sizes – 60, 125, 250, 500 and 1000 examples. We always used a test set consisting of 1000 examples. In each case, the goal was to train until the error rate ceased to decrease, or until there was evidence of overfitting. Results reported here are the average of five independent trials, where inital learner knowledge and the training/test sets were re-randomized in each trial.

4. Results and Discussion

The results of the runs in a representative subset of these experimental setups are depicted in Figures 2-7, and the results of all runs are summarized in Figure 8. The difference between the red lines in Figures 2-7 and the green/blue lines illustrate the generalization power of structured knowledge representations – the flat learners lack the benefit of a structured knowledge representation. The difference between the blue and green lines in these figures demonstrates the generalization power

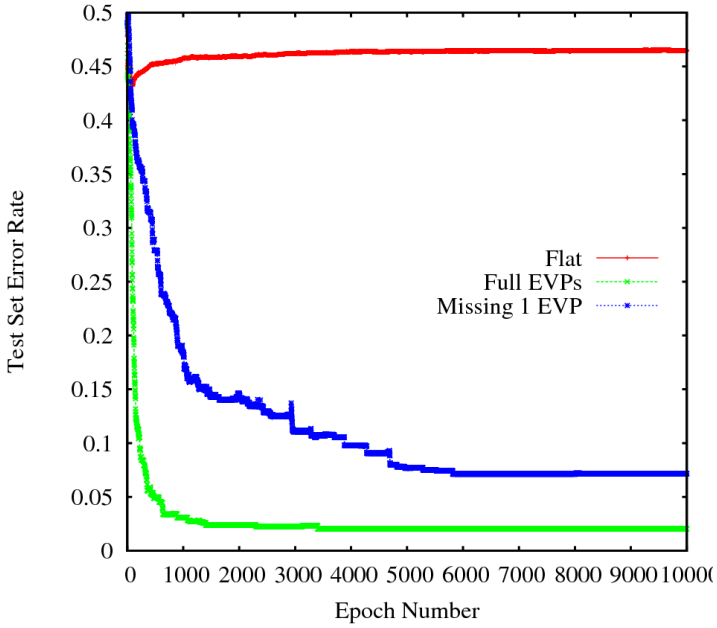

Figure 2. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 60 examples, where the hybrid learner is missing one EVP.

realized by enforcing fixed semantics at all internal nodes within a structured representation, as the hybrid learners fail to achieve error rates as low as those of the learners with EVPs at all concept nodes, for fixed training sets of various sizes.

The key result illustrated in Figures 2 through 7 is that in all cases, ANs that are missing EVPs from one or more nodes fail to reach a final error as low as the complete ANs. In general, this final error is higher (worse) when more EVPs are removed, and when the training set size is smaller. A progressive degeneration in learning ability as more EVPs are removed is apparent as a general trend, as demonstrated by Figure 8, which summarizes the results of all experiments run with hybrid ANs, showing the average final error rate for each experimental setup. It can also be seen from Figure 8 that larger training sets generally lead to lower final errors for each learner type, as one would expect. This main result clearly demonstrates that the enforcement of fixed semantics at concept nodes within a classification hierarchy introduces substantial inductive bias beyond that provided by the structure itself.

Somewhat surprisingly, in Figure 7, we see that in this case, the AN with three EVPs removed has failed to achieve a lower final error rate than the flat learner! This is likely due to the susceptibility of large EVP-less regions (which are effectively deep ANNs) to local minima. There are techniques that specifically address this problem, such as KBANN Towell and Shavlik (1994), which initializes a deep network based upon prior knowledge, or the work of LeCun et al. (1998) in which components in a structure are pretrained before end-to-end training is applied. However, in both cases the semantics of intermediate portions of the knowledge structure are enforced *before*

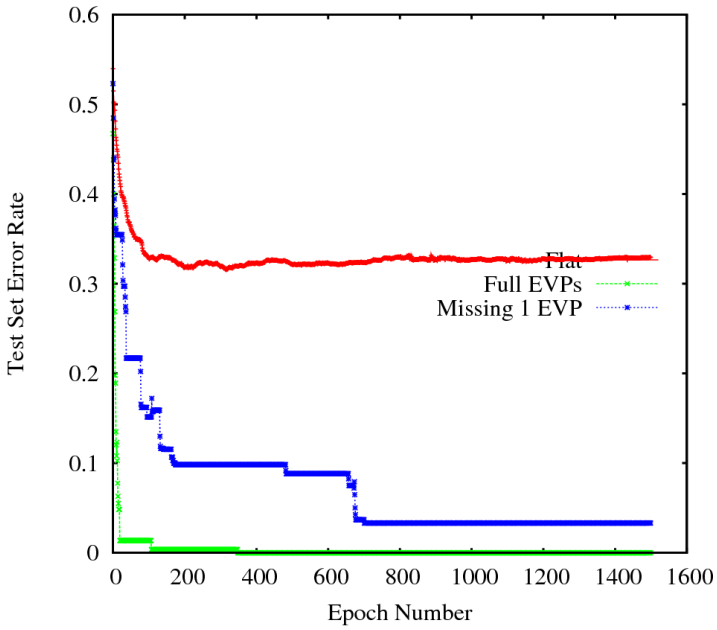

Figure 3. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 1000 examples, where the hybrid learner is missing one EVP.

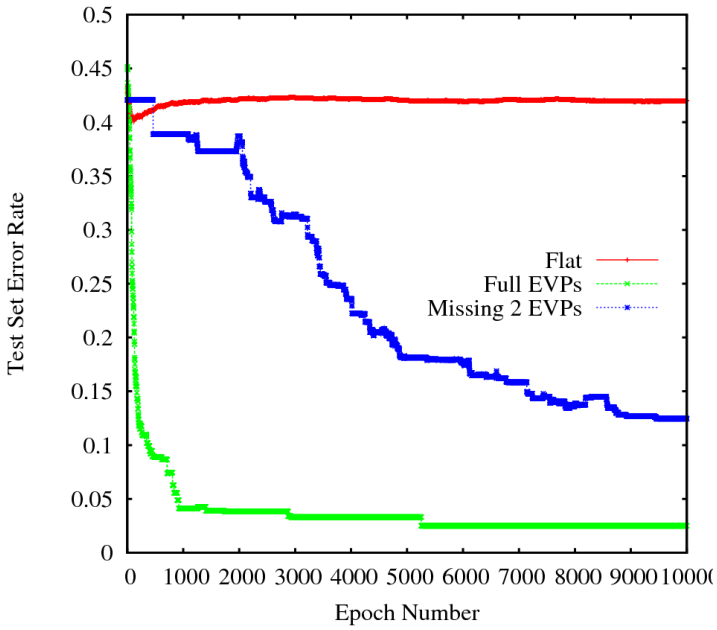

Figure 4. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 60 examples, where the hybrid learner is missing two EVPs.

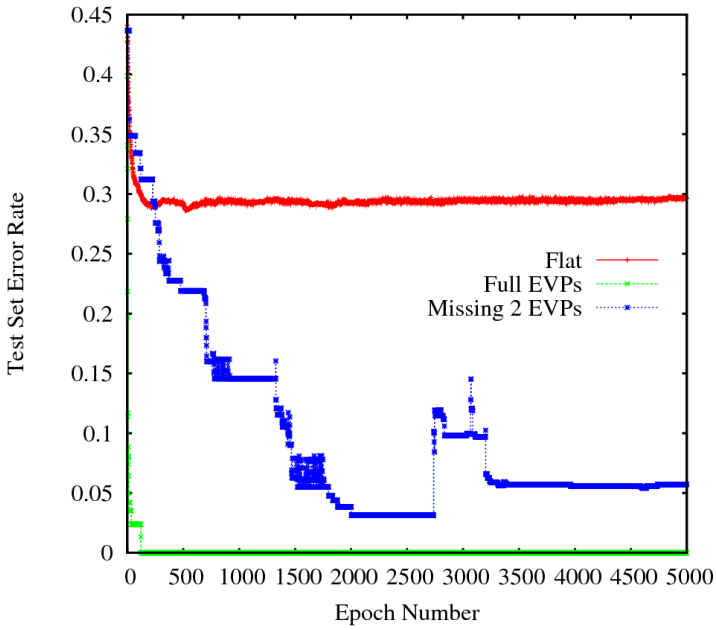

Figure 5. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 1000 examples, where the hybrid learner is missing two EVPs.

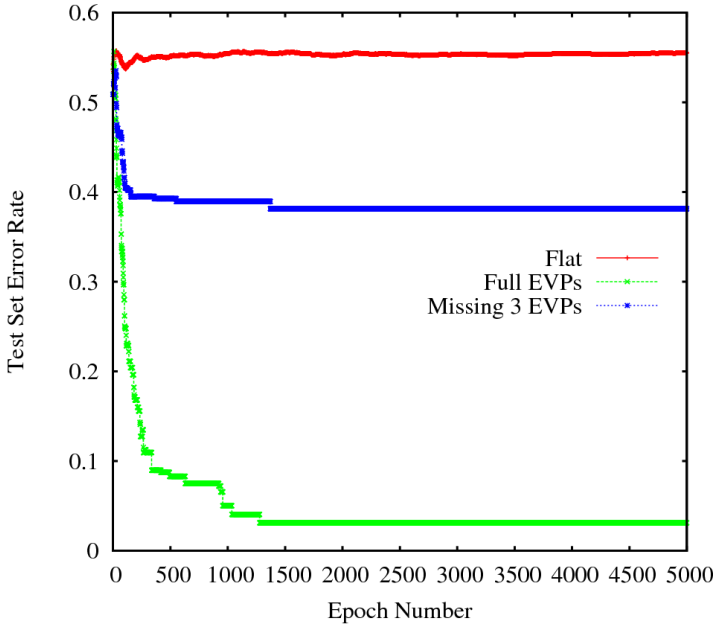

Figure 6. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 60 examples, where the hybrid learner is missing three EVPs.

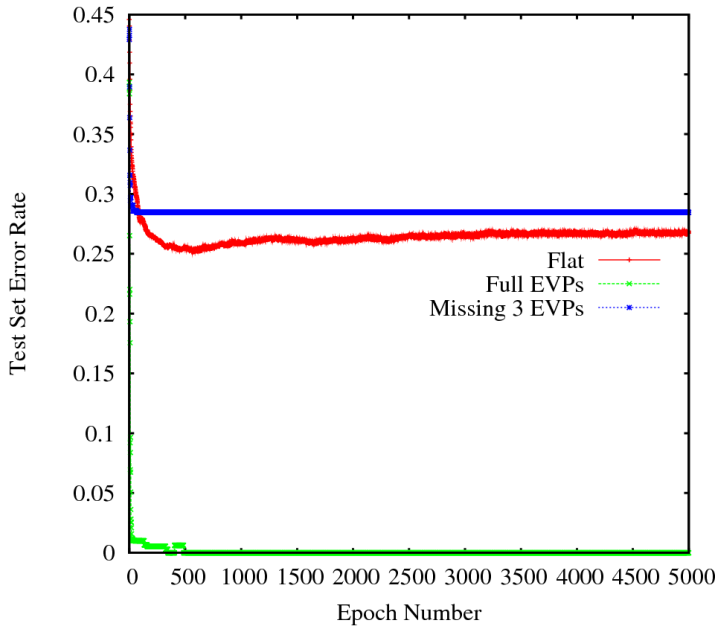

Figure 7. Training epoch vs. error rate for flat, AN and hybrid-AN learners with a training set size of 1000 examples, where the hybrid learner is missing three EVPs.

end-to-end training, even if this enforcement is relaxed after initialization/pretraining. Thus, even though the requirement for enforced semantics at internal nodes is weaker in these other techniques than when using a classification hierarchy with strict enforcement of concept semantics, there is still some reliance upon internal semantics to produce an initialization of the structure to be trained. The benefit of such techniques is that the problem of local minima in deep structure can be ameliorated to some extent by the initialization. ANs make even stronger use of these explicit internal node semantics, and, as evidenced by the very low final errors achieved by “complete" ANs in each of the experiments in this section, show a strong resistance to local minima.

An interesting secondary point to be noted is that, in each case, more epochs are required for the ANs with EVPs removed to reach their lowest error rate. This computational savings is another benefit of using fixed concept semantics wherever possible within a hierarchical classification learner, and is further evidence that the semantic “pinning" of nodes via EVPs is providing a restriction bias beyond that offered by the hierarchical structuring of knowledge alone.

Examining Figure 8, it is clear that the degradation in final error rate is much more pronounced in the AN that has had three EVPs ablated. It is likely that this is because a *connected subtree* of three nodes had their EVPs removed, creating a more substantial region within the AN that lacked fixed semantics. A large, connected region that lacks EVPs is likely to be very susceptible to local minima. In addition, these deep networks have more inputs than each member of the set of AN nodes they are replacing. Because the input dimension is higher, these deep networks will require more training examples to learn the target concept than the AN nodes they replace.

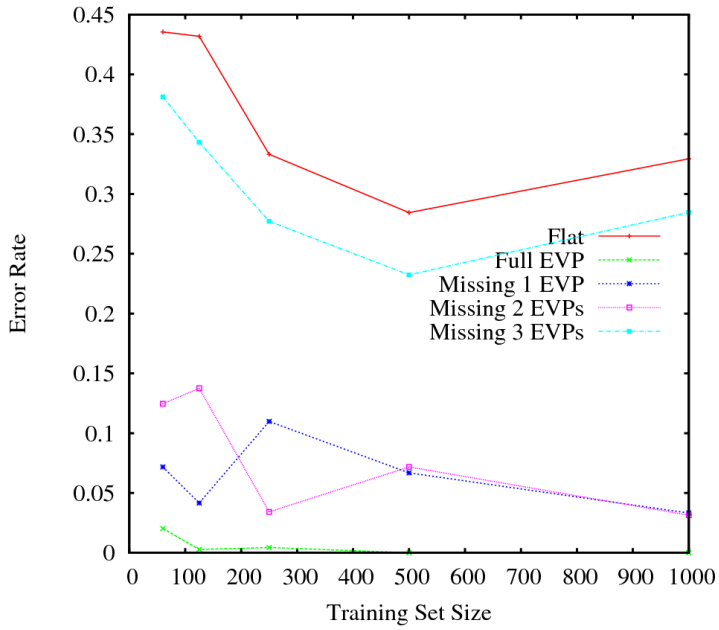

Figure 8. Training set size vs. best achieved error rate for flat, AN and hybrid-AN learners.

5. Conclusion

In this paper, we have presented empirical results that demonstrate the generalization power gained by enforcing fixed semantics of intermediate concepts in a classification hierarchy, beyond the power known to be provided by the knowledge structure itself. These results provide empirical support to the intuition that fixed concept semantics provide a useful form of inductive bias, and justify spending time and energy imbuing a learner with this kind of background knowledge when possible. Benefits are apparent even if fixed semantics cannot be enforced at all nodes within a classification hierarchy. Further, it appears that minimizing the size of connected subtrees within the knowledge structure that lack fixed semantics is particularly important in terms of maximizing the quality of generalizations made by the learner.

This result is useful for further developing techniques for learning by correcting mistakes. In general, a knowledge engineer not only develops a classification tree but also labels the intermediate nodes in the tree. It turns out that as in Winston's (1992) original technique for learning by correcting mistakes, this labeling of the intermediate nodes provides much needed learning bias.

References

Anderson, M. L., Oates, T., Chong, W., and Perlis, D. (2006). The metacognitive loop I: Enhancing reinforcement learning with metacognitive monitoring and control for improved perturbation tolerance. *Journal of Experimental and Theoretical AI*, 18(3):387–411.

Birnbaum, L., Collins, G., Freed, M., and Krulwich, B. (1990). Model-based diagnosis of planning failures. *In Proceedings of the Eighth National Conference on Artificial Intelligence*, pages 318–323.

Brachman, R. (2002). Systems that know what they are doing. *IEEE Intelligent Systems*, pages 67–71.

Bylander, T., Johnson, T., and Goel, A. (1991). Structured matching: A task-specific technique for making decisions. *Knowledge Acquisition*, 3:1–20.

Chandrasekaran, B. (1988). Generic tasks as building blocks for knowledge-based systems: The diagnosis and routine design example. *KNOWLEDGE ENG. REV.*, 3(3):183–210.

Chandrasekaran, B. and Goel, A. (1988). From numbers to symbols to knowledge structures: Artificial Intelligence perspectives on the classification task. *IEEE Trans. Systems, Man & Cybernetics*, 18(3):415–424.

Cox, M. and Raja, A., editors (2011). *Metareasoning: Thinking About Thinking*. MIT Press, Boston.

Cox, M. T. (2005). Metacognition in computation: A selected research review. *Artificial Intelligence*, 169(2):104–141.

Cox, M. T. and Ram, A. (1999). Introspective multistrategy learning: on the construction of learning strategies. *Artificial Intelligence*, 112(1-2):1–55.

Davis, R. (1980). Meta-rules: Reasoning about control. *Artificial Intelligence*, 15:179–222.

Doyle, J. (1979). A truth maintenance system. *Artificial Intelligence*, 12:231–272.

Fox, S. and Leake, D. (2001). Introspective reasoning for index refinement in case-based reasoning. *Journal of Experimental and Theoretical Artificial Intelligence*, 13:63–88.

Fu, L. and Buchanan, B. (1985). Learning intermediate concepts in constructing a hierarchical knowledge base. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, volume 1, pages 659–666.

Goel, A. and Bylander, T. (1989). Computational feasibility of structured matching. *IEEE Trans. Pattern Anal. Mach. Intell.*, 11(12):1312–1316.

Goel, A. and Jones, J. (2011). Metareasoning for self-adaptation in intelligent agents. In Cox, M. and Raja, A., editors, *Metareasoning: Thinking about Thinking*, chapter 10, pages 151–166. MIT Press, Boston.

Goel, A. K. and Murdock, J. W. (1996). Meta-cases: Explaining case-based reasoning. *European Workshop on Case-Based Reasoning (EWCBR)*, pages 150–163.

Hansen, E. and Zilberstein, S. (2001). Monitoring and control of anytime algorithms: A dynamic programming approach. *Artificial Intelligence*, 126(1-2).

Hayes-Roth, B. and Larsson, J. E. (1996). A domain-specific software architecture for a class of intelligent patient monitoring systems. *Journal of Experimental and Theoretical Artificial Intelligence*, 8(2):149–171.

Horvitz, E. (2001). Principles and applications of continual computation. *Artificial Intelligence*, 126(1-2).

Horvitz, E., Cooper, G., and Heckerman, D. (1989). Reflection and action under scarce resources: Theoretical principles and empirical study. *In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*.

Jones, J. and Goel, A. K. (2008). Retrospective self-adaptation of an agent's domain knowledge: Perceptually-grounded semantics for structural credit assignment. In *In Proceedings of the AAAI-08 Workshop on Metareasoning*, Chicago, IL.

Jones, J. K. and Goel, A. K. (2012). Perceptually grounded self-diagnosis and self-repair of domain knowledge. *Knowledge-Based Systems*, 27:281–301.

Leake, D. B. (1996). Experience, introspection and expertise: Learning to refine the case-based reasoning process. *Journal of Experimental and Theoretical Artificial Intelligence*, 8(3-4):319–339.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Minsky, M., Singh, P., and Sloman, A. (2004). The St. Thomas common sense symposium: Designing architectures for human-level intelligence. *AI Magazine*, 25(2):113–124.

Mitchell, T. M. and Thrun, S. B. (1993). Explanation-based neural network learning for robot control. In Giles, C. L., Hanson, S. J., and Cowan, J. D., editors, *Advances in Neural Information Processing Systems 5, Proceedings of the IEEE Conference in Denver*, San Mateo, CA. Morgan Kaufmann.

Murdock, J. and Goel, A. (2001a). Meta-case-based reasoning: Using functional models to adapt case-based agents. *Case-Based Reasoning Research and Development*, pages 407–421.

Murdock, J. W. and Goel, A. K. (2001b). Learning about constraints by reflection. In *AI '01: Proceedings of the 14th Biennial Conference of the Canadian Society on Computational Studies of Intelligence*, pages 131–140, London, UK. Springer-Verlag.

Murdock, J. W. and Goel, A. K. (2008). Meta-case-based reasoning: self-improvement through self-understanding. *Journal of Experimental and Theoretical Artificial Intelligence*, 20(1):1–36.

Murdock, W. and Goel, A. K. (2003). Localizing planning using functional process models. In *Proceedings of the Thirteenth International Conference on Automated Planning and Scheduling (ICAPS-03)*, pages 73–81.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: networks of plausible inference*. Morgan Kaufmann, San Mateo, CA.

Punch, W., Goel, A. K., and Brown, D. C. (1995). A knowledge-based selection mechanism for strategic control with application in design, assembly and planning. In *International Journal of Artificial Intelligence Tools*, volume 4, pages 323–348.

Raja, A. and Lesser, V. R. (2007). A framework for meta-level control in multi-agent systems. *Autonomous Agents and Multi-Agent Systems*, 15(2):147–196.

Rumelhart, D. E. and Mcclelland, J. L., editors (1986). *Parallel distributed processing: Explorations in the microstructure of cognition*, volume Volumes 1 & 2. MIT Press.

Russell, S. and Wefald, E. (1991). Principles of metareasoning. *Artificial Intelligence*, 49.

Shapiro, A. D. (1987). *Structured induction in expert systems*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Stefik, M. (1981). Planning and meta-planning (MOLGEN: Part 2). *Artificial Intelligence*, 16(2):141–170.

Stroulia, E. and Goel, A. (1995). Functional representation and reasoning in reflective systems. *Journal of Applied Intelligence, Special Issue on Functional Reasoning*, 9(1):101–124.

Stroulia, E. and Goel, A. K. (1996). A model-based approach to blame assignment: Revising the reasoning steps of problem solvers. *National Conference on Artificial Intelligence (AAAI-96)*, pages 959–965.

Stroulia, E. and Goel, A. K. (1997). Redesigning a problem-solver's operations to improve solution quality. *15th International Joint Conference on Artificial Intelligence (IJCAI-97)*, pages 562–567.

Stroulia, E. and Goel, A. K. (1999). Evaluating problem-solving methods in evolutionary design: the Autognostic experiments. *International Journal of Human-Computer Studies, Special Issue on Evaluation Methodologies*, 51:825–847.

Tadepalli, P. and Russell, S. J. (1998). Learning from examples and membership queries with structured determinations. In *Machine Learning*, volume 32, pages 245–295.

Towell, G. G. and Shavlik, J. W. (1994). Knowledge-based artificial neural networks. *Artificial Intelligence*, 70(1-2):119–165.

Ulam, P., Jones, J., and Goel, A. K. (2008). Using model-based reflection to guide reinforcement learning. In *Fourth AAAI Conference on AI in Interactive Digital Entertainment*.

Whiteson, S., Kohl, N., Miikkulainen, R., and Stone, P. (2005). Evolving keepaway soccer players through task decomposition. *Machine Learning*, 59(1):5–30.

Wilensky, R. (1981). Meta-planning: Representing and using knowledge about planning in problem solving and natural language understanding. *Cognitive science*, 5(3):197–233.

Winston, P. H. (1992). *Artificial intelligence (3rd ed.)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Wogulis, J. and Langley, P. (1989). Improving efficiency by learning intermediate concepts. In *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, pages 657–662.