Using Analogy Discovery to Create Abstractions

Marc Pickett

Cognition, Robotics, and Learning
University of Maryland, Baltimore County,
marc@coral.cs.umbc.edu

Concept formation is a form of abstraction that allows for knowledge transfer, generalization, and compact representation. Concepts are useful for the creation of a generally intelligent autonomous agent. If an autonomous agent is experiencing a changing world, then nearly every experience it has will be unique in that it will have at least slight differences from other experiences. Concepts allow an agent to generalize these experiences and other data. In some applications, the concepts that an agent uses are explicitly provided by a human programmer. A problem with this approach is that the agent encounters difficulties when it faces situations that the programmer had not anticipated. For this reason, it would be useful for the agent to automatically form concepts in an unsupervised setting. The agent should be able to depend as little as possible on representations tailored by humans, and therefore it should develop its own representations from raw uninterpreted data.

One purpose of concept formation (and abstraction in general) is to concisely characterize a set of data [7]. With this view, one can use *minimum description length* as a guiding principle for concept formation. My research uses this principle to form an ontology of concepts from a collection of data. This data is a set (or a stream) of statements, where each statement is an ordered tuple of symbols (representing relations). The symbols have no meaning for the program other than they're considered to be ground statements. For example, these symbols can be raw sensor data, or raw descriptions of chess games.

For SARA 2005, Tim Oates and I developed an ontology formation algorithm called The Cruncher [6] that works on attribute-value data. The Cruncher (an extension of PolicyBlocks [5], an algorithm for discovering useful macro-actions in Reinforcement Learning that I developed with Andy Barto) is a simple representation framework and algorithm based on minimum description length for automatically forming an ontology of concepts from attribute-value data sets. Although unsupervised, when The Cruncher is applied to the Zoo database from [1], it produces a nearly zoologically accurate categorization. The Cruncher can also be applied to find useful macro-actions in Reinforcement Learning, learn models from uninterpreted sensor data, or form an ontology of documents based on word-frequency.

It's useful to be able to develop *relational* concepts through *analogy*. Some suggest that analogy may even be the "core of cognition" [3]. Analogy allows us to focus on the relations among entities rather than superficial aspects of the entities. For example, we might notice that a red ant killing a black ant and stealing a piece of food it is analogous to a situation in Hamlet where Claudius murders Hamlet's father and usurps the throne of Denmark. In this situation, we must also be able to specify that the red ant corresponds to Claudius, the black ant to Hamlet's father, and the piece of food maps to the throne. Once found, relational concepts can be useful for knowledge transfer: conclusions about one domain can map to another domain.

Currently, I'm extending The Cruncher to work on relational data. The extended algorithm, The Übercruncher, discovers isomorphisms (or analogies) in relational data, and forms concepts from the analogies to compress the data. After finding a set of analogies, the best analogy is used to compress the Knowledge Base, resulting in a shorter description. This entire process (finding analogies and crunching with them) is repeated until no more useful analogies are found. In practice, useful analogies are often found as parts of other analogies, which produces a multi-tiered ontology.

The Übercruncher is related to the SUBDUE system [4] which compresses graphs by finding common substructures. Both SUBDUE and The Übercruncher work on data that's not presegmented, and both use minimum description length as the guiding principle by which substructures are evaluated. Like The Übercruncher, SUBDUE also does *induction* in the sense that frequently occurring substructures are replaced by a node that symbolizes the full substructure. However, SUBDUE uses a potentially slow beam search, upon which The Übercruncher improves by building a conceptual structure that can be used to accelerate learning and classification into a current ontology. Additionally, The Übercruncher represents both concepts and meta-concepts in the same framework so that the same algorithm can be used to find analogies in both data and meta-data. Ignoring differences in representation and search strategy, SUBDUE is essentially a strictly bottom-up version of The Übercruncher, which also uses top-down guidance for classification, similar to that described by [2].

Future work involves developing a full cognitive architecture, dubbed The Marchitecture, that *uses* the ontology developed by The Übercruncher for reasoning, planning, classification, and explanation, and integrates these processes with formation of the ontology.

References

1. C.L. Blake and C.J. Merz. UCI repository of machine learning databases, 1998.
2. Hawkins, J., and Blakeslee, S. 2004. *On Intelligence.* Times Books.
3. D. R. Hofstadter. Analogy as the core of cognition. *The Analogical Mind: Perspectives from Cognitive Science*, pages 499–538, 2001.
4. L. Holder, D. Cook, and S. Djoko. Substructure discovery in the subdue system. In *Proceedings of the Workshop on Knowledge Discovery in Databases*, 1994.
5. M. Pickett and A. Barto. Policyblocks: An algorithm for creating useful macro-actions in reinforcement learning. In *Proceedings of the International Conference on Machine Learning*, 2002.
6. M. Pickett and T. Oates. The cruncher: Automatic concept formation using minimum description length. In *proceedings of the 6th International Symposium on Abstraction, Reformulation and Approximation (SARA)*, 2005.
7. J. G. Wolff. Information compression by multiple alignment, unification and search as a unifying principle in computing and cognition. *Artif. Intell. Rev.*, 2003.