Q

Q-Learning

Peter Stone
The University of Texas at Austin, Austin, TX, USA

Definition

Q-learning is a form of ►temporal difference learning. As such, it is a model-free ►reinforcement learning method combining elements of ►dynamic programming with Monte Carlo estimation. Due in part to Watkins' (1989) proof that it converges to the optimal value function, Q-learning is among the most commonly used and well-known reinforcement learning algorithms.

Cross References

►Reinforcement Learning
►Temporal Difference Learning

Recommended Reading

Watkins, C. J. C. H. (1989). *Learning from delayed rewards*. PhD thesis. Cambridge: King's College.

Quadratic Loss

►Mean Squared Error

Qualitative Attribute

►Categorical Attribute

Quality Threshold Clustering

Xin Jin, Jiawei Han
University of Illinois at Urbana-Champaign
Urbana, IL, USA

Synonyms

QT Clustering

Quality threshold (QT) clustering (Heyer, Kruglyak, & Yooseph 1999) is a partitioning clustering algorithm originally proposed for gene clustering. The focus of the algorithm is to find clusters with guaranteed quality. Instead of specifying K, the number of clusters, QT uses the maximum cluster diameter as the parameter.

The basic idea of QT is as follows: Form a candidate cluster by starting with a random point and iteratively add other points, with each iteration adding the point that minimizes the increase in cluster diameter. The process continues until no point can be added without surpassing the diameter threshold. If surpassing the threshold, a second candidate cluster is formed by starting with a point and repeating the procedure. In order to achieve reasonable clustering quality, already assigned points are available for forming another candidate cluster.

For data partition, QT selects the largest candidate cluster, removes the points which belong to the cluster from consideration, and repeats the procedure on the remaining set of data.

Recommended Reading

Heyer, L., Kruglyak, S., & Yooseph, S. (1999). Exploring expression data: Identification and analysis of coexpressed genes. *Genome Research*, 9, 1106–1115.

Claude Sammut & Geoffrey I. Webb (eds.), *Encyclopedia of Machine Learning*, DOI 10.1007/978-0-387-30164-8,

Quantitative Attribute

► Numeric Attribute

Query-Based Learning

Sanjay Jain, Frank Stephan
National University of Singapore, Republic of Singapore

Definition

Most learning scenarios consider learning as a relatively passive process where the learner observes a set of data and eventually formulates a hypothesis that explains the data observed. Query-based learning is an ►active learning process where the learner has a dialogue with a teacher, which provides on request useful information about the concept to be learnt.

Detail

This article will mainly focus on query-based learning of finite classes and of parameterized families of finite classes. In some cases, an infinite class has to be learnt where then the behaviour of the learner is measured in terms of a parameter belonging to the concept. For example, when learning the class of all singletons $\{x\}$ with $x \in \{0,1\}^*$, the parameter would be the length n of x and an algorithm based on membership queries would need up to $2^n - 1$ queries of the form "Is y in L?" to learn an unknown set $L = \{x\}$ with $x \in \{0,1\}^n$. Query-based learning studies questions like the following: Which classes can be learnt using queries of this or that type? If queries of a given type are used to learn a parameterized class $\bigcup C_n$, is it possible to make a learner which (with or without knowledge of n) succeeds to learn every $L \in C_n$ with a number of queries that is polynomial in n? What is the exact bound on queries needed to learn a finite class C in dependence of the topology of C and the cardinality of C? If a query-based learner using polynomially many queries exists for a parameterized class $\bigcup C_n$, can this learner also be implemented such that it is computable in polynomial time?

In the following, let C be the class of concepts to be learnt and the concepts $L \in C$ are subsets of some basic set X. Now the learning process is a dialogue between a learner and a teacher in order to identify a language $L \in C$, which is known to the teacher but not to the learner. The dialogue goes in turns and follows a specific protocol that goes over a finite number of rounds. Each round consists of a query placed by the learner to the teacher and the answer of the teacher to this query. The query and the answer have to follow a specific format and there are the following common types, where $a \in X$ and $H \in C$ are data items and concepts chosen by the learner and $b \in X$ is a counterexample chosen by the teacher:

Query-Name	Precise Query	Answer if true	Answer if false
Membership-Query	Is $a \in L$?	"Yes"	"No"
Equivalence-Query	Is $H = L$?	"Yes"	"No" plus b (where $b \in H{-}L \cup L{-}H$)
Subset-Query	Is $H \subseteq L$?	"Yes"	"No" plus b (where $b \in H - L$)
Superset-Query	Is $H \supseteq L$?	"Yes"	"No" plus b (where $b \in L - H$)
Disjointness-Query	Is $H \cap L = \emptyset$?	"Yes"	"No" plus b (where $b \in H \cap L$)

While, for subset queries and superset queries, it is not required by all authors that the teacher provides a counterexample in the case that the answer is "no," this requirement is quite standard for the case of equivalence queries. Without counterexamples, a learner would not have any real benefit from these queries in settings where faster convergence is required, than by just checking "Is $H_0 = L$?," "Is $H_1 = L$?," "Is $H_2 = L$?," ..., which would be some trivial kind of algorithm.

Here is an example: Given the class C of all finite subsets of $\{0,1\}^*$, a learner using superset queries could just work as follows to learn each set of the form $L = \{x_1, x_2, \ldots, x_n\}$ with $n+1$ queries:

Round	Query	Answer	Counter example
1	Is $L \subseteq \varnothing$?	"No"	x_1
2	Is $L \subseteq \{x_1\}$?	"No"	x_2
3	Is $L \subseteq \{x_1, x_2\}$?	"No"	x_3
⋮	⋮	⋮	⋮
n	Is $L \subseteq \{x_1, x_2, \ldots, x_{n-1}\}$?	"No"	x_n
$n+1$	Is $L \subseteq \{x_1, x_2, \ldots, x_{n-1}, x_n\}$?	"Yes"	—

Here, of course, the order on how the counterexamples come up does not matter; the given order was just preserved for the reader's convenience. Note that the same algorithm works also with equivalence queries in place of superset queries. In both cases, the algorithm stops with outputting "$L = \{x_1, x_2, \ldots, x_n\}$" after the last query. However, the given class is not learnable using membership and subset queries which can be seen as follows: Assume that such a learner learns $\varnothing$ using the subset queries "Is $H_0 \subseteq L$?," "Is $H_1 \subseteq L$?," "Is $H_2 \subseteq L$?," ..., "Is $H_m \subseteq L$?" and the membership queries "Is $y_0 \in L$?," "Is $y_1 \in L$?," "Is $y_2 \in L$?," ..., "Is $y_k \in L$?" Furthermore, let D be the set of all counterexamples provided by the learner to subset queries. Now let $E = D \cup H_0 \cup H_1 \cup \ldots \cup H_m \cup \{y_0, y_1, \ldots, y_k\}$. Note that E is a finite set and let x be an element of $\{0,1\}^* - E$. If $L = \{x\}$ then the answers to these queries are the same to the case that $L = \varnothing$. Hence, the learner cannot distinguish between the sets $\varnothing$ and $\{x\}$; therefore, the learner is incorrect on at least one of these sets.

In the case that C is finite, one could just ask what is the number of queries needed to determine the target L in the worst case. This depends on the types of queries permitted and also on the topology of the class C. For example, if C is the power set of $\{x_1, x_2, \ldots, x_n\}$, then n membership queries are enough; but if C is the set of all singleton sets $\{x\}$ with $x \in \{0,1\}^n$, then $2^n - 1$ membership queries are needed to learn the concept, although in both cases the cardinality of C is 2^n.

Angluin (2004) provides a survey of the prior results on questions like how many queries are needed to learn a given finite class. Maass and Turán (1992) showed that usage of membership queries in addition to equivalence queries does not speed up learning too much compared to the case of using equivalence queries alone. If EQ is the number of queries needed to learn C from equivalence queries alone (with counterexamples) and EMQ is the number of queries needed to learn C with equivalence queries and membership queries then

$$\frac{EQ}{\log(EQ+1)} \leq EMQ \leq EQ;$$

here the logarithm is base 2. This result is based on a result of Littlestone (1988) who characterized the number of queries needed to learn from equivalence queries alone and provided a "standard optimal algorithm" for this task.

Angluin (1987) showed that the class of all regular languages can be learnt in polynomial time using queries and counterexamples. Here the learning time is measured in terms of two parameters: the number n of states that the smallest determinisitic finite automaton generating the language has and the number m of symbols in the longest counterexample provided by the teacher. Ibarra and Jiang (1988) showed that the algorithm can be improved to need at most dn^3 equivalence queries when the teacher always returns the shortest counterexample; Birkendorf, Böker, and Simon (2000) improved the bound to dn^2. In these bounds, d is the size of the alphabet used for defining the regular languages to be learnt.

Much attention has been paid to the following question: Which classes of Boolean formulas over n variables can be learnt with polynomially many queries, uniformly in n (see, for example, Aizenstein et al. (1995); Aizenstein (1995); Angluin, Hellerstein, and Karpinski (1993); Hellerstein, Pillaipakkamnatt, Raghavan, and Wilkins (1996))? Angluin, Hellerstein, and Karpinski (1993) showed that read-once formulas, in which every variable occurs only once, are learnable in polynomial time using membership and equivalence queries. On the other hand, read-thrice DNF (disjunctive normal form) formulas cannot be learnt in polynomial time using the same queries (Aizenstein et al., 1992) unless

P = NP. In other words, such a learner would not succeed because of the limited computational power of a polymomial time learner; hence, equipping the learner with an additional oracle that can provide this power would permit to build such a learner. Here an oracle - in contrast to a teacher - does not know the task to be learnt but gives information which is difficult or impossible to compute. Such an oracle could, for example, be the set SAT of all satisfiable formulas and thus the learner could gain additional power by asking the oracle whether certain formulas are satisfiable. A special class of Boolean formulas is that of Horn clauses and the study in this field is still active (see, for example, Angluin, Frazier and Pitt (1992), Arias (2004), Arias & Balcazar (2009) and Arias & Khardon (2002)).

There are links to other fields. Angluin (1988, 1990) investigated the relation between query learning and ►PAC Learning. She found that every class which is learnable using membership queries and equivalence queries is also PAC learnable (Angluin, 1988); the PAC learner also works in polynomial time and needs at most polynomially many examples. More recent research on learning Boolean formulas also combines queries with probabilistic aspects (Jackson, 1997). Furthermore, query learning has also been applied to ►Inductive Inference (see, for example, Gasarch (2008, 1992); Jain et al (2007); Lange (2005)). Here the power of the learner depends not only on the type of queries permitted but also on whether queries of the corresponding type can be asked finitely often or infinitely often; the latter applies of course only to learning models where the learner converges in the limit and may revise the hypothesis from time to time. Furthermore, queries to oracles have been studied widely, see the entry on ►Complexity of Inductive Inference.

Recommended Reading

Aizenstein, H., Hellerstein, L., & Pitt, L. (1995). Read-thrice DNF is hard to learn with membership and equivalence queries. In *Thirty-third annual symposium on foundations of computer science*, Pittsburgh, 24–27 October 1992 (pp. 523–532). Washington, DC: IEEE Computer Society.

Aizenstein, H., & Pitt, L. (1995). On the learnability of disjunctive normal form formulas. *Machine Learning, 19*(3), 183–208.

Angluin, D. (1987). Learning regular sets from queries and counterexamples. *Information and Computation, 75*(2), 87–106.

Angluin, D. (1988). Queries and concept learning. *Machine Learning, 2*(4), 319–342.

Angluin, D. (1990). Negative results for equivalence queries. *Machine Learning, 5*, 121–150.

Angluin, D. (2004). Queries revisited. *Theoretical Computer Science, 313*, 175–194.

Angluin, D., Frazier, M., & Pitt, L. (1992). Learning conjunctions of Horn clauses. *Machine Learning, 9*, 147–164.

Angluin, D., Hellerstein, L., & Karpinski, M. (1993). Learning read-once formulas with queries. *Journal of the Association for Computing Machinery, 40*, 185–210.

Arias, M. (2004). Exact learning of first-order Horn expressions from queries. Ph.D. Thesis, Tufts University.

Arias, M., & Balcázar, J. L. (2009). Canonical Horn representations and query learning. *Algorithmic learning theory: Twentieth international conference ALT 2009*, LNAI (Vol. 5809, pp. 156–170). Berlin: Springer.

Arias, M., & Khardon, R. (2002). Learning closed Horn expressions. *Information and Computation, 178*(1), 214–240.

Birkendorf, A., Böker, A., & Simon, H. U. (2000). Learning deterministic finite automata from smallest counterexamples. *SIAM Journal on Discrete Mathematics, 13*(4), 465–491.

Hellerstein, L., Pillaipakkamnatt, K., Raghavan, V. V., & Wilkins, D. (1996). How many queries are needed to learn? *Journal of the Association for Computing Machinery, 43*, 840–862.

Gasarch, W., & Lee, A. C. Y. (2008). Inferring answers to queries. *Journal of Computer and System Sciences, 74*(4), 490–512.

Gasarch, W., & Smith, C. H. (1992). Learning via queries. *Journal of the Association for Computing Machinery, 39*(3), 649–674.

Ibarra, O. H., & Jiang, T. (1988). Learning regular languages from counterexamples. In *Proceedings of the first annual workshop on computational learning theory*, MIT, Cambridge (pp. 371–385). San Francisco: Morgan Kaufmann.

Jackson, J. (1997). An efficient membership-query algorithm for learning DNF with respect to the uniform distribution. *Journal of Computer and System Sciences, 55*(3), 414–440.

Jain, S., Lange, S., & Zilles, S. (2007). A general comparison of language learning from examples and from queries. *Theoretical Computer Science, 387*(1), 51–66.

Lange, S., & Zilles, S. (2005). Relations between Gold-style learning and query learning. *Information and Computation, 203*, 211–237.

Littlestone, N. (1988). Learning quickly when irrelevant attributes abound: A new linear threshold algorithm. *Machine Learning, 2*, 285–318.

Maass, W., & Turán, G. (1992). Lower bound methods and separation results for on-line learning models. *Machine Learning, 9*, 107–145.