Self-Reinforced Meta Learning for Belief Generation

Alexandros Gkiokas[1], Alexandra I. Cristea[2], and Matthew Thorpe[3]

Abstract Contrary to common perception, learning does not stop once knowledge has been transferred to an agent. Intelligent behaviour observed in humans and animals strongly suggests that after learning, we self-organise our experiences and knowledge, so that they can be more efficiently reused; a process that is unsupervised and employs reasoning based on the acquired knowledge. Our proposed algorithm emulates meta-learning in-silico: creating beliefs from previously acquired knowledge representations, which in turn become subject to learning, and are further self-reinforced. The proposition of meta-learning, in the form of an algorithm that can learn how to create beliefs on its own accord, raises an interesting question: can artificial intelligence arrive to similar beliefs, rules or ideas, as the ones we humans come to? The described work briefly analyses existing theories and research, and formalises a practical implementation of a meta-learning algorithm.

Key words: Meta Learning; Reinforcement Learning; Inductive Learning; Conceptual Graphs; Cognitive Agents; Complex Systems.

1 Introduction

A question often asked, is what could be done with the knowledge of a domain, after it has been acquired. Raymond Cattell hypothesised that *Crystallised Intelligence* [1] is the key to acquiring knowledge in a supervised or unsupervised manner, and that there exists a connection between learning and what takes place after: reusing the acquired knowledge or skill.

[1],[2] Computer Science Department ·[3] Mathematics Institute
University of Warwick, Coventry, CV4 7AL, UK

The connection of learning and the processes taking place after are the focus of our paper, as it describes the transition from learning to meta-learning, the ability to manipulate and reshape existing knowledge in order to optimise its reuse. Meta learning in psychology was first described by Donald Maudsley [2] in 1979. In Computer Science, meta-learning has been considered a sub-field of machine learning used automatically on meta-data [3][4]. Psychology and Computer Science display different approaches to the same idea: *some process* takes place after learning, probably *consciously* [2].

Existing research on meta-learning in artificial intelligence is scarce; Vanschoren and Blockeel [3] examine the computational aspects of meta-learning in data mining, yet they emphasise the usage of synthetic knowledge representation. In an analogue manner, we propose the employment of conceptual graphs [5][6], as the knowledge representation structure. We do so because the knowledge acquisition agent from our previous work [7], already implements conceptual graphs, but also because they are based on *common logic*, and parsimoniously capture the structural and relational information of the generated knowledge, in the form of a belief without any excess of other meta-data.

State of the art research done for IBM by Vilalta and Drissi [4], best describes the key components of a meta-learning agent or system. They emphasise classification as being an important part, but most importantly describe meta-learning as a fusion of inductive learning, classification and knowledge representation. Their work is of a more theoretical nature, as they focus on structured and unstructured data, as well as cross-domain performance. Yet, their most important insight is that meta-learning is a self-adaptive processes relying on experience and knowledge.

2 Objectives

Our foremost objective is the generalisation of knowledge, or in this case, conceptual graphs. The knowledge generated is not *discovered* in the traditional sense; it is not a search or exploration, as is often the case with artificial intelligence architectures, reasoners or problem solvers, such as UIMA [8]. The first *goal is to compress the accumulated knowledge* for a specific cluster of highly correlated knowledge, in more generalised or abstract forms, that can represent a group, or parts of it. Our expectation of doing so, is achieving better learning performance while reducing knowledge instances. The algorithm must self-assess and evaluate its own beliefs, and self-reward its learning progress, without the intervention of an external party. The positively reinforced beliefs, should be apt to changes and non-monotonic; if one contradicts another they should not cancel each-other out, whereas belief generation should become better as learning converges.

The input is the set of conceptual graphs G, stored in the agent's memory; where g_n is any conceptual graph, positively rewarded by a user; that set is first classified into a set of clusters K, by using vector space model [9] and K-Means [10][11]. Each *cluster* in set K contains a sub-set of G that is attributionally similar (e.g., a high frequency of similar words). Following classification, every graph g_n in cluster k_n, is examined semantically and compared to other graphs in its cluster k_n, so as to induce a new set of generalised graphs (called beliefs) g'_n.

$$Size(g'_n) < Size(k_n). \qquad \text{and} \qquad g'_n \not\subset k_n. \tag{1}$$

The conditions of the output (1) are that the result must be smaller than the initial number of conceptual graphs, and the result must **not** contain any identical conceptual graph (but only *similar*). Any new derived conceptual graph *g'*, *may or may not* depend, on a single graph or more, from k_n. The most important objective of the system, is to learn what semantic distance is acceptable for the generation of a new belief through induction. We employ reinforcement learning for this purpose [12], in an episodic manner, with the aim to converge on reasoning: *what semantic similarity is acceptable for the generation of a belief.* Rewarding of the generated beliefs is done posteriorly, as the belief needs be tested against new incoming conceptual graphs; in that sense the agent is self-reinforcing its decision to create that belief, express by a series of policies Q as shown in (2), for the episode that constructed the belief g'_n. Constants α and γ are the learning and discount rates, whilst R is the reward of the next state.

$$Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha[R_{(t+1)} + \gamma Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)]. \tag{2}$$

To obtain meta-learning, we follow the phases as defined by Vivalta et al [4]. The first phase of meta-learning they propose is characterised by a clustering operation. We fuse K-Means [10], and vector space model [9] (VSM) and use the methodology described in [11] for estimating the number of clusters K. The VSM constructs a sparse matrix M of token-pattern frequencies, using the frequency of appearance of a token within a graph, and each graph is associated with a positively rewarded conceptual graph. The vector space model, given as input the proportionally vectorised $V(g_n)$ of the queried graph, will return as output a vector of degrees of similarity $S(g_n)$, with respect to the (graph index) columns of matrix M, as shown in (3).

$$S(g_n) = \frac{M * V(g_n)}{\|M\| \, \|V(g_n)\|}. \tag{3}$$

The output vector $S(g_n)$ represents the similarity of the input graph g_n to all other graphs in memory. For each VSM output $S(g)$, we *min-max* normalise it between *0* and *1*, and then append it as a row into a matrix A. We use K-Means, by minimizing (4) with respect to 'cluster centers' μ_i, on data set

$\{x_j\}_{j=1}^n$ where x_i are the similarity values of graph g_n to all other graphs.

$$\sum_{j=1}^{n} \min_{i=1,\ldots,K} \|x_j - \mu_i\|^2 . \tag{4}$$

The partition is then defined by associating each x_j with the cluster center closest to it, i.e., $k_i = \{x_j : \|x_j - \mu_i\| \leq \|x_j - \mu_m\| \text{ for all } m = 1, 2, \ldots, K\}$. The number of clusters are estimated by the number of dominant eigenvalues of the matrix A where $A(i,j) = \|x_i - x_j\|^2$.

Semantic approximation takes the form of induction or generalisation, by iterating or traversing a semantic graph, starting from a query node, and moving towards super-ordinates. WordNet [13] implements its "senses" in a *Hasse* graph, a mono-directional and ordered semantic graph. In inductive logic, a traversal from *n* to *n'* (often denoted as [n,n']), where *n* is in a different layer from *n'*, the originating node *n* will *most likely* inherit the properties and attributes of *n'*. Induction is essential to meta-learning, when producing generalisations, as expressed by Vivalta et al [4]. Using WordNet, we acquire the super-ordinate graph of node *n* and *n'* (known as hypernyms). Establishing a semantic path from one another, is done by iterating super-ordinates and their layers. Finding a direct path from *[n,n']* might not always be possible, in which case, the algorithm uses a breadth-first search, by finding if nodes *n* and *n'* share a common super-ordinate. For the first common super-ordinate obtained, the semantic distance, quantifiable by value $\delta_{[n,n']}$ is calculated, as shown in (5).

$$\delta_{[n,n']} = w_{s_i}(\sum d_{[n,n']}). \qquad \text{and} \qquad w_{s_i} = \frac{s_i - s_{min}}{s_{max} - s_{min}}. \tag{5}$$

The WordNet sense s_i is min-max normalised to w_{s_i}, as shown in (5), and $\sum d_{[n,n']}$ (5) is the sum of traversed semantic graph layers. The normalised sense weight w_{s_i}, biases towards most frequently appearing WordNet senses, since the topmost frequent is presented first, and least frequent is returned last [13]. Semantic distance is employed directly by the meta-learning algorithm, as a means to estimate the substitutability of a node n, by its super-ordinate n'.

3 Meta-Learning

Generating beliefs in the form of conceptual graphs, is the second step. After having acquired clusters of *similar* conceptual graphs, the algorithm iterates conceptual graphs in a cluster k_n. Each conceptual graph g_i in cluster k_n, is compared to another graph g_k also from k_n. Calculating the similarity of g_i and g_k, involves summing all their vertex similarities expressed by the value

$\sum \delta[g_i^n, g_k^{n'}]$. A second condition is that g_i and g_k are *isomorphically equivalent*, in addition to being *semantically related*. Isomorphism, a key component in graphs [5], requires that g_i and g_k are *structurally identical* with regards to their edges (graph similarity). Yet, graph g_i may have an edge to a vertex to which g_k doesn't. If, however, there exists a semantic path between the vertices, meaning that the connected vertex n by edge e in g_i, is *similar* to the connected vertex *n'* by edge *e'* in graph g_k, induction is possible. Therefore, in addition to $\delta_{[n,n']}$, we examine *isomorphic similarity* by iterating all edges e of n and all edges *e'* of *n'*. Edge similarity is accumulative and expressed by $\epsilon_{[n,n']}$, as shown in (6).

$$\epsilon_{[n,n']} = \sum \delta_{[e_i(n), e_i(n')]}. \tag{6}$$

If both semantic and isomorphic similarity are satisfied for g_i and g_k, then a new graph g'_{ik} will be created, representing a *belief.* That *belief* is created, by replacing the *common super-ordinate* found for semantic path *[n,n']* when calculating $\delta_{[n,n']}$. Those criteria (node delta value and isomorphism) are two of the Markov properties used by reinforcement learning, in the *decision making* (third) phase. The Markov decision process (MDP), controlled by reinforcement learning, creates an *episode*: a sequence of *states* s_t, chained together by *actions* a_t. A state s_t is described and *partially observable* by three components: conceptual graph distributional similarity (with respect to another graph) normalised from 0 to 1, node semantic distance expressed by $\delta_{[n,n']}$, and isomorphic variance expressed by $\epsilon_{[n,n']}$. An action a_t may be one of following decisions: abort the process, or substitute nodes n and *n'* with their common super-ordinate *k.* The reward $R_{(t+1)}$ is the reward of the next state, and is given to the algorithm at a later time, by cross-evaluating the belief g'_{ik}. As per the reinforcement learning literature [12], the reward is given only to terminal states. When creating a belief g'_{ik}, the policy $Q(s_t, a_t)$ updates its value, whilst trying to converge on the optimal decision. Evaluating actions is done by rewarding positively the episode that created the belief g'_{ik} when a new conceptual graph g_z (newly acquired knowledge) verifies it. Verification is performed by finding that graph g_z could substitute any or all of its nodes with the ones in belief g'_{ik}, and by establishing that they are isomorphically equivalent. The graph g_z must belong to the same cluster k_n, from which g'_{ik} was generated, in order to be eligible for evaluation. Self-reinforcement essentially strengthens the algorithm's policy and decision making process, making it learn how to best create beliefs.

4 Conclusions

In this paper we have presented a novel meta-learning algorithm, created via a fusion of existing artificial intelligence models and technologies.

We've formulated the theoretical and algorithmic foundations of the meta-learning model, and systematically explained how it would work under training and testing. The proposed algorithm constructs meta-knowledge representations, by generalising and abstracting existing knowledge into beliefs, based on induction. The learning mechanism is based on a robust reinforcement learning algorithm, whereas the knowledge representation scheme used is Sowa's conceptual graphs. Finally, classification is done via a fusion of K-means and vector space model, whereas Semantics are obtained via WordNet. By designing and describing this algorithm, we've taken every precaution to adhere to previous theoretical and practical work: adaptive behaviour, learning via induction, semantics, classification, knowledge representation, all emphasised key characteristics from previous researchers involved in state-of-the-art frameworks.

References

1. R. B. Cattell, "Abilities: Their structure, growth, and action", New York, Houghton Mifflin, 1971, ISBN-10: 0395042755
2. D. B. Maudsley, "A Theory of Meta-Learning and Principles of Facilitation: An Organismic Perspective", University of Toronto, 1979
3. J. Vanschoren and H. Blockeel, "Towards understanding learning behavior", Proceedings of the Annual Machine Learning Conference of Belgium and the Netherlands, Benelearn 2006, pp. 89-96, 2006
4. R. Vilalta and Y. Drissi, "A Perspective View And Survey Of Meta-Learning", Artificial Intelligence Review, volume 18, pp. 77–95, 2002
5. J. F. Sowa, "Knowledge Representation: Logical, Philosophical and Computational Foundations", Brooks Press, 1999, ISBN-10: 0534949657
6. M. Chein and M. Mugnier, "Graph-based Knowledge Representation", Springer, 2009, ISBN: 978-1-84800-285-2
7. A. Gkiokas and A. I. Cristea, "Training a Cognitive Agent to Acquire and Represent Knowledge from RSS feeds onto Conceptual Graphs", IARIA COGNITIVE 2014, Venice, Italy, pp 184 - 194, 25 May 2014
8. D. Ferrucci and A. Lally, "UIMA: an architectural approach to unstructured information processing in the corporate research environment", Journal Natural Language, volume 10, issue 3 - 4, pp. 327 - 348, September 2004
9. P. Turney and P. Pantel, "From Frequency to Meaning: Vector Space Models of Semantics", Journal of Artificial Intelligence Research, vol. 37, 2010, pp. 141-188.
10. J. B. MacQueen, "Some Methods for classification and Analysis of Multivariate Observations", Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, Berkeley, University of California Press, volume 1, pp. 281-297, 1967
11. M. Girolami, "Mercer kernel-based clustering in feature space", Neural Networks, IEEE Transcations, vol. 13, issue 3, May 2002, pp. 780 - 784.
12. R. S. Sutton and A. G. Barto, "Reinforcement Learning: An Introduction", MIT press, 1998, ISBN-10: 9780262193986
13. C. Fellbaum, "WordNet: An Electronic Lexical Database (Language, Speech, and Communication)", MIT press, 1998, ISBN-10: 026206197X