ISBN: 978-950-42-0200-4
DOI: https://doi.org/10.33414/ajea.5.744.2020

Un Enfoque Jerárquico Bi-capa de Optimización Bayesiana de Hiper-parámetros en Aprendizaje por Refuerzos

A Hierarchical Two-tier Approach to Hyper-parameter Optimization in Reinforcement Learning

Presentación: 6-7/10/2020

Doctorando:
Juan Cruz Barsce
Grupo de investigación en simulación para Energía Química, Facultad Regional Villa María, Universidad Tecnológica Nacional - Argentina
jbarsce@gmail.com

Director:
Ernesto Martínez

Co-director:
Jorge Palombarini

Resumen

La optimización de hiper-parámetros en algoritmos de aprendizaje por refuerzos (RL) es una tarea clave, porque los mismos determinan cómo el agente aprenderá su política interactuando con su ambiente, y por lo tanto cómo los datos son recolectados. Se aquí un enfoque que aplica optimización Bayesiana para realizar una optimización de dos pasos: en primer lugar, los hiper-parámetros categóricos de RL son tomados como variables binarias y optimizados con una función de adquisición acorde. Luego, a un menor nivel de abstracción, se optimizan los hiper-parámetros directamente relacionados con la solución del problema con la función de adquisición de mejora esperada, usando los mejores hiper-parámetros categóricos encontrados en el nivel de abstracción superior. Este enfoque bi-capa es validado en tareas de control clásicas, arrojando resultados promisorios que abren camino a aplicaciones de RL independientes de usuario.

Palabras clave: aprendizaje por refuerzos, optimización de hiper-parámetros, optimización Bayesiana, optimización Bayesiana de estructuras combinatoriales

Abstract

Optimization of hyper-parameters in reinforcement learning (RL) algorithms is a key task, because they determine how the agent will learn its policy by interacting with its environment, and thus what data is gathered. An approach that uses Bayesian optimization to perform a two-step optimization is proposed: first, categorical RL structure hyper-parameters are taken as binary variables and optimized with an acquisition function tailored for such variables. Then, at a lower level of abstraction, solution-level hyper-parameters are optimized by resorting to the expected improvement acquisition function, while using the

best categorical hyper-parameters found in the optimization at the upper-level of abstraction. This two-tier approach is validated in a simulated control task. Results obtained are promising and open the way for more user-independent applications of reinforcement learning.

Keywords: reinforcement learning, hyper-parameter optimization, Bayesian optimization, Bayesian optimization of combinatorial structures (BOCS)

Introducción

La generalización a partir de los datos en algoritmos de aprendizaje automático involucra un proceso de entrenamiento, en donde el algoritmo aprende la estructura del modelo y los parámetros que mejor se ajustan a los datos disponibles. El entrenamiento, por su parte, depende de un proceso de diseño a priori que define los hiper-parámetros que establecen las condiciones del aprendizaje guiado por los datos. Establecer apropiadamente los hiper-parámetros es crucial en el proceso de aprendizaje, pudiendo ser la diferencia entre alcanzar un entrenamiento de estado del arte y uno mediocre (Hutter et al., 2015). Particularmente, la optimización de hiper-parámetros en algoritmos de aprendizaje por refuerzos (RL) (Sutton & Barto, 2018) es una tarea difícil, porque en la misma los datos no son provistos a priori sino que son generados incrementalmente mediante interacciones con el ambiente. Los hiper-parámetros, por caso, determinan los datos generados, los cuales a su vez determinan los valores aprendidos de los parámetros, que a su vez influencian el nuevo conjunto de datos generados, y así sucesivamente.

Usualmente los hiper-parámetros son optimizados a mano, lo cual puede ser muy ineficiente (Hutter et al., 2015), o a través de métodos como búsqueda aleatoria (Bergstra & Bengio, 2012), u optimización Bayesiana (BO) (Močkus et al., 1978) (Shahriari et al., 2016). Esta última realiza una optimización de caja negra de una función f, y recurre tanto a una distribución a priori $f \sim P(y)$, como a muestreos previos (X, y) para computar la media y varianza de puntos de entrada no observados. Esto suele calcularse mediante regresión por procesos Gaussianos (GP) (Rasmussen & Williams, 2008), los cuales utilizan una *función de adquisición* que es barata de optimizar globalmente para predecir el mejor próximo punto a muestrear. La función de adquisición más común es la de *mejora esperada*, que predice el próximo punto a muestrear considerando la probabilidad estimada de que el mismo sea el próximo máximo, tomando en cuenta también la magnitud de la varianza predicha. En esto se diferencia de métodos no informados como la búsqueda aleatoria, que usan información muy limitada sobre las consultas previas a f. Mientras que la optimización Bayesiana utiliza información sobre consultas pasadas, posee dos limitaciones similares a búsqueda aleatoria: 1) no involucra asunciones sobre la influencia de los hiper-parámetros en el contenido informativo y 2) no es eficiente para optimizar hiper-parámetros categóricos tales como el algoritmo de aprendizaje seleccionado. Se propone para ello un algoritmo, *RLOpt two-tier*, que asume una relación jerárquica entre hiper-parámetros de RL, en la cual un conjunto de los mismos, los hiper-parámetros estructurales, es más influyente que otro de menor nivel de abstracción, y en base a eso optimizar los mismos iterativamente en dos capas. Los hiper-parámetros que definen la estructura a menudo son discretos (como el número de capas ocultas de una red neuronal), categóricos o binarios (como un posible hiper-parámetro que determina si la tasa de exploración debe ser decrementada con el paso de los episodios). En otras palabras, representan una estructura combinatoria de caja negra que es costosa de muestrear.

Tomando en cuenta estas consideraciones, en el enfoque propuesto se aborda primero la optimización de los hiper-parámetros estructurales usando una variante de optimización Bayesiana para manipular hiper-parámetros categóricos. Tras ello, se usa optimización Bayesiana tradicional para ajustar los hiper-parámetros de números reales del algoritmo de aprendizaje (que dependen directamente de los primeros) comenzando con la mejor estructura y parámetros previamente encontrados. Comparado con la optimización Bayesiana tradicional, el algoritmo propuesto presenta dos ventajas principales:

1. Se enfoca en la optimización de un subconjunto de hiper-parámetros a la vez, permitiendo el manejo de diferentes niveles de complejidad en cada paso de optimización.

2. Al dividir la optimización en dos capas, los mejores modelos o hiper-parámetros de una capa pueden ser reusados al optimizar la otra. Por ejemplo, la optimización de los hiper-parámetros de menor jerarquía puede comenzar con los parámetros θ pre-entrenados a partir del mejor modelo de mayor jerarquía.

El algoritmo propuesto es validado frente a búsqueda aleatoria y optimización Bayesiana estándar en entornos de control clásicos.

Desarrollo

En el algoritmo propuesto, se realiza una optimización bi-capa tanto de los hiper-parámetros estructurales Θ_a como de los hiper-parámetros de solución Θ_b del agente RL. En primer lugar, se realiza la optimización de los hiper-parámetros estructurales de alto nivel, para luego dar lugar a la optimización de los hiper-parámetros de menor nivel de abstracción. Tras la primera ronda de optimización de los hiper-parámetros estructurales y de algoritmo, estos últimos son congelados y se usa la mejor combinación encontrada como punto de partida de las próximas optimizaciones de hiper-parámetros de estructura. El proceso de optimización bi-capa se ilustra en la Fig. 1, donde se muestra cómo ambas capas se encuentran interrelacionadas por bucles interno y externo para realizar una mejora iterativa de dos conjuntos de hiper-parámetros.

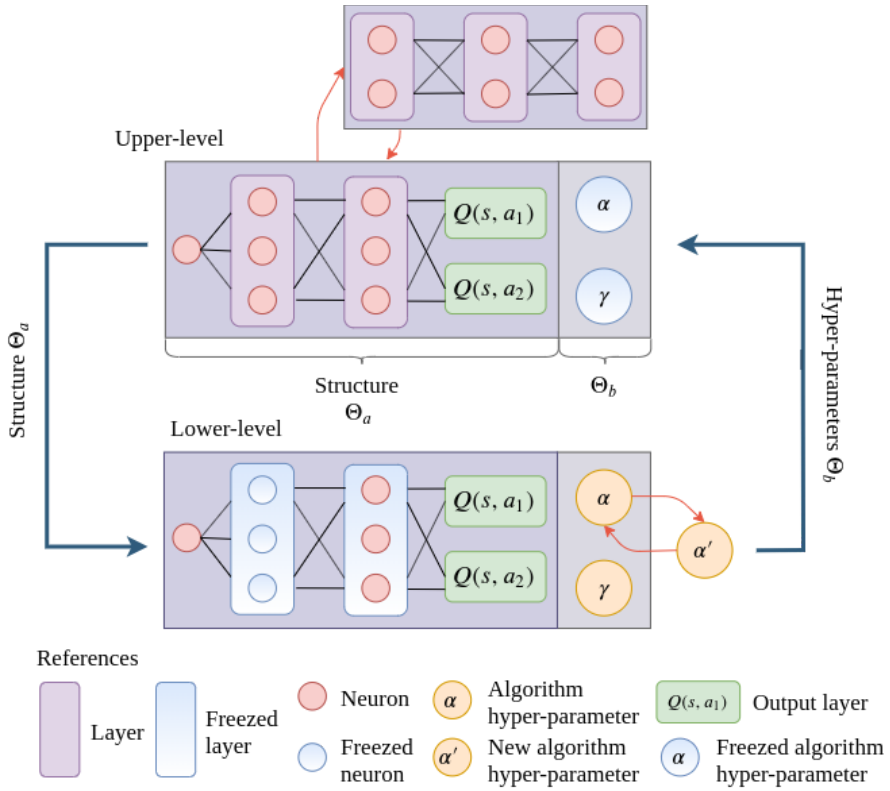

Figura 1. Optimización bi-capa de hiper-parámetros de RL. Al optimizar la capa superior, los hiper-parámetros de algoritmo se mantienen congelados. Tras finalizar cada optimización de capa superior, los hiper-parámetros estructurales tales como el número de redes neuronales y un conjunto predefinido de parámetros aprendidos se mantienen congelados, mientras que el resto de los hiper-parámetros es ajustado en el bucle inferior.

Para medir el rendimiento de la optimización, se propone la función objetivo $f: \Theta_a \cup \Theta_b \rightarrow \mathbb{R}$, que mapea una tupla $\Theta_a \cup \Theta_b$ de hiper-parámetros estructurales y de algoritmo a un número real que se corresponde en una medida del rendimiento del agente RL instanciado para simular una determinada cantidad de episodios en un ambiente predeterminado y con esa configuración, para que pueda aprender una política de comportamiento que maximice la recompensa promedio recibida por el entorno. Para reducir el sesgo de valores previos de hiper-parámetros al mínimo, cada vez que el agente prueba una nueva

configuración de hiper-parámetros, el mismo reinicia todo su conocimiento que tenía sobre la política, de modo de empezar aprendiendo la nueva política desde cero sin influencia de los hiper-parámetros que anteriormente se utilizaron. Por otra parte, cada nueva instancia del agente RL que corre por un cierto número de episodios bajo una misma configuración de hiper-parámetros es denominada aquí como *meta-episodio*.

Para optimizar los hiper-parámetros de estructura en el nivel superior de la jerarquía se utiliza el algoritmo de optimización Bayesiana para estructuras categóricas (BOCS) (Baptista & Poloczek, 2018), el cual toma los mismos como variables binarias. Cada hiper-parámetro individual $\Theta_{a_i} \in \Theta_a$ puede entonces representar la presencia o no de una determinada característica (tomando un valor de $\Theta_{a_i} = 1$ o en caso contrario de $\Theta_{a_i} = 0$, respectivamente). Esta característica puede ser categórica, por ejemplo para indicar si se usará una arquitectura de tipo *dueling deep Q-learning*, o bien puede corresponder a un valor entero, por ejemplo para representar el número de capas ocultas de la red neuronal. Al optimizar los hiper-parámetros categóricos, los hiper-parámetros categóricos que dependen de ellos son congelados (esto es, no son modificados por el algoritmo BOCS), partiendo desde un vector inicial Θ_b de hiper-parámetros preestablecidos. Tal vector inicial puede ser elegido usando conocimiento a priori, valores por defecto, o bien usando la última configuración encontrada al optimizar los hiper-parámetros de menor jerarquía.

Con respecto a la optimización de hiper-parámetros de menor jerarquía, se utiliza optimización Bayesiana estándar tras terminar la optimización en los hiper-parámetros de mayor nivel. Tal proceso involucra asunciones de normalidad y un modelo estadístico de procesos Gaussianos. Para empezar el entrenamiento desde una base sólida, y considerando que la optimización de alto nivel ya ha realizado alguna mejora de los hiper-parámetros de la arquitectura, los meta-episodios en el menor nivel son inicializados con el mejor vector de hiper-parámetros Θ_a de la capa superior. En otras palabras, esto significa que cada vez que el agente es reiniciado mientras se optimizan los hiper-parámetros de menor jerarquía, este reinicio establece los parámetros Θ_a de la última configuración del nivel superior. Esto es usado tanto para hacer un inicio rápido del agente con un modelo pre-entrenado promisorio con pesos θ, como para darle al modelo estadístico una referencia de f, de modo tal que la optimización Bayesiana comience con un punto de referencia del mejor valor de f obtenido hasta el momento. En ese sentido, la capa de bajo nivel actúa como un ajuste fino del modelo de la capa superior, debido a que empieza optimizando una porción de una red pre-entrenada. Teniendo esto en cuenta, si la arquitectura del modelo está basada en redes neuronales, la optimización de bajo nivel puede también hacerse con un subconjunto reducido de las últimas capas ocultas, dejando congelados los pesos de las primeras capas. Esto permite una optimización mucho más estable y precisa, reduciendo el costo de entrenamiento de toda la arquitectura, especialmente si se trata de una red neuronal profunda.

Una vez que la optimización de la jerarquía de menor nivel termina, se finaliza un ciclo de optimización. Tras el mismo, comienza una nueva evaluación con un nuevo conjunto de N evaluaciones estructurales, en donde el mejor vector de hiper-parámetros Θ_b es usado como el vector inicial del nuevo ciclo. El algoritmo de optimización completo se puede ver en el Algoritmo 1.

Por otra parte, los resultados obtenidos para el entorno de control clásico *Pendulum* pueden verse en la Fig. 2. En la misma, se muestran ejecuciones de 30 meta-episodios por cada optimizador, donde cada meta-episodio consiste a su vez en 100 episodios. Los hiper-parámetros estructurales optimizados fueron el número de capas ocultas, el número de neuronas por capa y el tamaño del lote. Por otra parte, en el algoritmo se usó *Soft Actor-Critic* (Haarnoja et al., 2018) y se optimizaron el ratio de aprendizaje, el factor de descuento, la magnitud de actualización de los parámetros de la política y el coeficiente de entropía. En la figura puede apreciarse cómo un incremento en la cantidad de ciclos completos de optimización tiene aparejado un incremento significativo en la velocidad de convergencia, y cómo el compartir el modelo mejora significativamente frente a los demás optimizadores.

Input : Función de agente-RL f, conjunto de hiper-parámetros a priori Θ_b, número de evaluaciones (N, E, M), hiper-parámetros del proceso Gaussiano, función de adquisición α, distribución posterior P, λ

Establecer los hiper-parámetros de algoritmo al conjunto de hiper-parámetros a priori Θ_b para la primera ronda de N evaluaciones estructurales

for *evaluación completa* $= 1$ *hasta* E **do**

  for *evaluación estructural* $= 1$ *hasta* N **do**

    Obtener los hiper-parámetros estructurales Θ_a que optimizan $\alpha_{BOCS}(.) \rightarrow \mathbb{R}$

    Evaluar la función de adquisición f en el punto (Θ_a, Θ_b)

    Agregar la nueva tripleta (Θ_a, Θ_b, f) a D_1,

    Actualizar la distribución posterior de BOCS $P(\alpha \mid D_1)$

  end

  Inicializar D_2 con la tripleta (Θ_a, Θ_b, f) que arrojó el mayor f

  Establecer los hiper-parámetros estructurales como el mejor conjunto estructural Θ_a y establecer sus hiper-parámetros aprendidos bajo esa configuración como los hiper-parámetros iniciales θ por las próximas M evaluaciones de hiper-parámetros

  for *Evaluación de hiper-parámetros de algoritmo* $= 1$ *to* M **do**

    Obtener los hiper-parámetros de algoritmo Θ_b que optimizan $\alpha_{EI}(.) \rightarrow \mathbb{R}$

    Evaluar la función objetivo f en el punto (Θ_a, Θ_b)

    Agregar la nueva tripleta (Θ_a, Θ_b, f) a D_2

    Actualizar las funciones de predicción por procesos Gaussianos $\mu_{n+1}(X)$ y $\sigma^2_{n+1}(X)$

  end

  Establecer los hiper-parámetros de algoritmo a priori como los mejores Θ_b para las próximas N evaluaciones estructurales

end

Output: $(\arg\max_{(\Theta_a, \Theta_b)} f, \max f)$

Algoritmo 1. Optimización Bayesiana aplicada para optimizar hiper-parámetros estructurales, de arquitectura (ej: red neuronal) y de algoritmos.

Conclusiones

Se mostró un nuevo enfoque que permite optimizar hiper-parámetros tanto categóricos como que toman valores reales, asumiendo una relación jerárquica entre las capas de abstracción. La validación en problemas de control clásico muestra que el enfoque propuesto performa consistentemente mejor que dos enfoques comunes de optimización de hiper-parámetros. Se mostró también que la jerarquía bi-capa puede también ser usada como forma de ajuste fino de una red neuronal pre-entrenada, dada una estructura predefinida. Este fue el aspecto distintivo que hizo que el enfoque bi-capa obtenga un rendimiento mejor que el enfoque monolítico de optimización Bayesiana, porque así como cada meta-episodio consistió en buscar una porción del espacio total de hiper-parámetros, compartir el mejor modelo permitió arrancar la optimización desde mejores condiciones iniciales, algo que no es posible al optimizar todo el espacio de hiper-parámetros a la vez, como sucedía en la optimización Bayesiana monolítica.

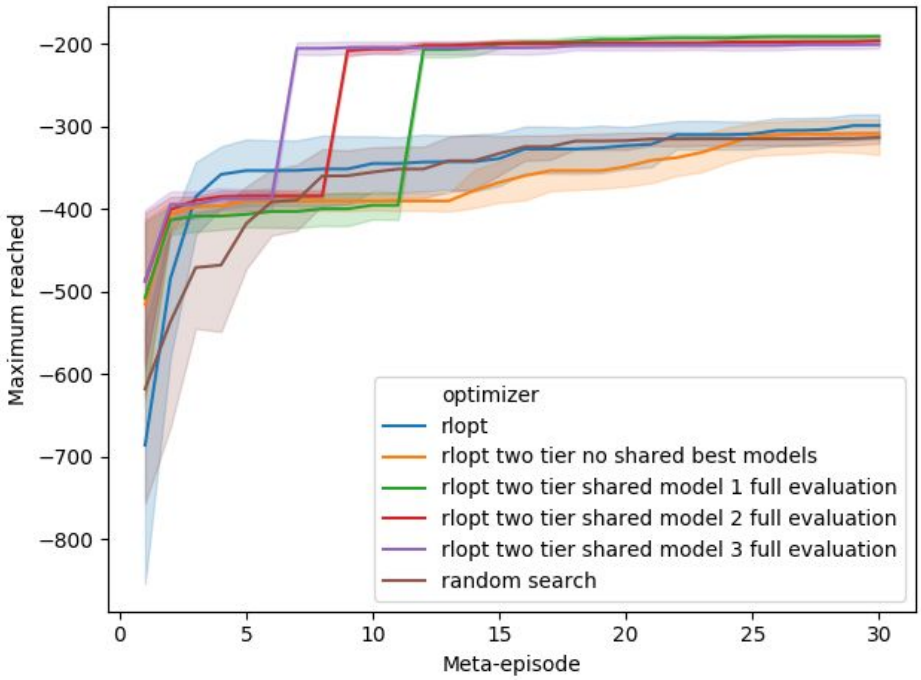

Figura 2. Comparación de los diferentes máximos obtenidos.

Referencias

Baptista, R., & Poloczek, M. (2018). Bayesian Optimization of Combinatorial Structures. ArXiv:1806.08838 [Cs, Math, Stat]. http://arxiv.org/abs/1806.08838

Bergstra, J., & Bengio, Y. (2012). Random Search for Hyper-Parameter Optimization. Journal of Machine Learning Research, 13(Feb), 281–305.

Haarnoja, T., Zhou, A., Abbeel, P., & Levine, S. (2018). Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor. ArXiv:1801.01290 [Cs, Stat]. http://arxiv.org/abs/1801.01290

Hutter, F., Lücke, J., & Schmidt-Thieme, L. (2015). Beyond Manual Tuning of Hyperparameters. KI - Künstliche Intelligenz, 29(4), 329–337. https://doi.org/10.1007/s13218-015-0381-0

Močkus, J., Tiesis, V., & Zilinskas, A. (1978). The application of Bayesian methods for seeking the extremum. In L. C. W. Dixon & G. P. Szego (Eds.), Towards Global Optimisation 2 (pp. 117–129). North-Holand.

Rasmussen, C. E., & Williams, C. K. I. (2008). Gaussian processes for machine learning (3. print). MIT Press.

Shahriari, B., Swersky, K., Wang, Z., Adams, R. P., & de Freitas, N. (2016). Taking the Human Out of the Loop: A Review of Bayesian Optimization. Proceedings of the IEEE, 104(1), 148–175. https://doi.org/10.1109/JPROC.2015.2494218

Sutton, R. S., & Barto, A. G. (2018). Reinforcement Learning: An Introduction (2nd ed.). MIT Press.