Genetic programming for learning acquisition functions for BO

Baez-Nieto Luis A.[1][0000−0003−2388−2790]

Instituto Nacional de Astrofísica Óptica y Electrónica, Luis Enrique Erro 1, Sta María Tonanzintla, 72840 San Andrés Cholula, Pue. baez@inaoep.mx

Abstract. El minimizar el número de iteraciones para alcanzar el mejor resultado dadas una serie de entradas en una función de caja negra es un problema que se estudia arduamente por sus grandes beneficios. En este trabajo se busca el mejorar la Optimización Bayesiana, un método que se usa para encontrar el máximo o mínimo global de una función de caja negra, a través de la búsqueda de nuevas funciones de adquisición. La búsqueda de estas funciones de adquisición se realiza mediante programación génetica, una estrategía de programación evolutiva. Se muestra que con tal estrategia es posible solucionar problemas de caja negra con una precisión superior a la que se obtiene mediante la función de adquisición denominada como mejora esperada, esto, dentro de problemas de optimización de una dimensión.

Keywords: Optimización Bayesiana · Programación Genética · Funciones de Adquisición.

1 Introducción

Una función de caja negra consiste en un proceso en el que se desconoce su funcionamiento interno, por lo que encontrar una configuración óptima para obtener el mejor resultado depende de los parámetros de entrada. A esta complejidad se le agrega el hecho de que es posible que la función sea multimodal, que la salida del proceso tenga ruido o que incluso, el evaluar una sola configuración tenga costos muy altos en tiempo y/o recursos.

Estas funciones de caja negra se encuentran en muchos tipos de problemas. Uno de los ejemplos más conocidos dentro del aprendizaje máquina, son las redes neuronales profundas. Para encontrar la mejor solución y el evitar el sobre ajuste del modelo, se es necesario probar con diferentes configuraciones en los hyperparámetros de la red. Una solución propuesta para obtener esta configuración es mediante el uso de técnicas de aprendizaje máquina automatizado (autoML) [1], de las cuales, ya existen una gran variedad.

Otro ejemplo que no involucra solo un proceso virtual se encuentra en el diseño de experimentos. Un experimento también cuenta con un cierto número de variables de entrada controlables y no controlables. A la salida se puede obtener desde una medición hasta un producto. Como en el ejemplo anterior, el

mejor resultado del experimento depende de la configuración de estos parámetros de entrada.

Los ejemplos descritos son solo una pequeña muestra del cómo pueden ser estos problemas que se comportan como una funcion de caja negra. Estas funciones se encuentran en áreas como la robótica, la teoría control, planeación, probabilidades intratables, pruebas A/B y entre muchas otras. Siempre y cuando la solución al problema dependa de los parámetros de entrada, se considerará como un problema de caja negra.

Dados estos argumentos es fácil observar que el encontrar estas soluciones con el menor número de iteraciones y observaciones, es una proriedad. Actualmente, una forma efectiva de atacar esta problemática es mediante el uso de Optimización Bayesiana (BO), ya que este método no toma en cuentaa priori la forma que pueda tener esta función.

Dentro de este proyecto se ataca el problema que consiste en encontrar la forma de esta función dados una serie de lecturas con ruido. Para ello, se usará BO, donde la función de adquisición (AF) se generará a partir de programación genética.

2 Trabajo relacionado

En el estado del arte se encuentran diversos trabajos que se enfocan en el desarrollo de nuevas AFs. Uno de los que se puede resaltar se encuentra el trabajo de [2]. La función de adquisición que ellos proponen se genera a partir del momento en el que la función sustituta se genera. La función de adquisición resultante tiene un parámetro que controla la relación de exploración explotación en la búsqueda.

Otro trabajo que aporta al desarrollo de las AFs, se encuenctra en [3], quienes usan otro enfoque. En su propuesta, se busca maximizar estas funciones, ya sea por medio de la integración de Monte Carlo en optimización basada en el gradiente, o mediante maximización voraz del myiopic maximal. En su propuesta se mostró que estos enfoques mejoran cada grupo de AFs que se trataron.

Una propuesta aún más interesante es la que se expone en [4]. En este artículo se propone cambiar la función de adquisición por una red neuronal profunda, con el propósito de usar este modelo en diferentes problemas.

Cada una de estas propuestas ofrece el mejorar la función de adquisición, sin embargo, en el caso del primer trabajo, la AF resultante solo ha sido probada en la función Ackley con resultados muy similares a la función tradicional de mejora esperada (EI). En el segundo trabajo se limita en mejorar las funciones existentes, además, solo en las que es posible obtener el gradiente y en aquellas que están dentro del grupo de myiopic maximal. Solo el tercer artículo propone una AF que puede ser usada para solventar varios tipos de problemas.

3 Marco teórico

Ya que el proyecto propone el encontrar una AF que mejore los resultados de la Optimización Bayesiana mediante Programación Genética, en esta sección se dará una breve descripción de estos procesos.

3.1 Optimización Bayesiana

La optimización Bayesiana es una estrategia de diseño secuencial que no necesita la derivada de la función para encontrar el resultado óptimo global. Dicho de otra forma, la tarea se concentra en aproximarse al valor óptimo (mínimo o máximo) de una función objetivo de dimención d usando una secuencia de variables.

Para aproximarse a la función se usa una función sustituta (generalmente un proceso Gaussiano), el cual modela la incertidumbre de la función como una probabilidad de distribución. Usando la base de datos (los puntos en los que ya se ha evaluado la función) el modelo actualiza el modelo previo a uno posterior (Figura 1)

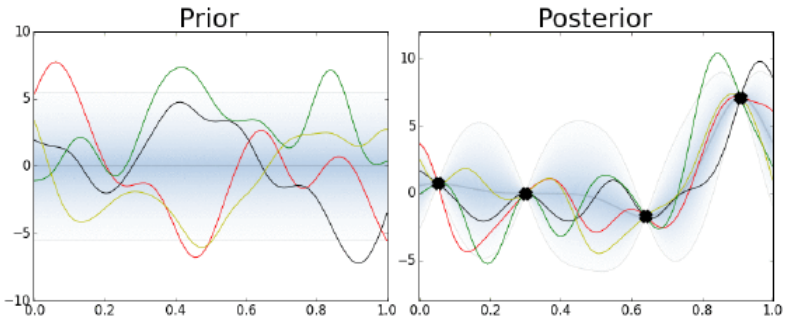

Fig. 1. Actualización del modelo sustituto

Una vez obtenido la actualización del modelo sustituto, es posible establecer una ganancia de la función para soluciones desconocidas. Esta ganancia se le conoce como la función de adquisición, la cual establece el balance entre la exploración de nuevos puntos dentro de la distribución, con la explotación de puntos cercanos óptimo global actual.

Como ya se planteó en la introducción, existen varias funciones de adquisición. Entre las más comunes se encuentra la mejora esperada (EI), la cual trata de balancear entre el grado de exploración y explotación; la AF denominada como probabilidad de mejora (PI) es una alternativa a EI, la cual tiende a explorar más para mejorar la mejor solución; El límite de confianza bajo (LCB) se generó para resolver el problema aleatorio bandit balanceando la explotación con la exploración. En la literatura se encuentran aún más funciones de adquisición que, dependiendo del problema al que se tenga que abordar, se usará una u otra.

3.2 Programación genética

La programación genética es una estrategia de programación evolutiva en la cual se tiene una población de individuos que están representados, generalmente, en forma de árboles. Cada hoja del árbol representa a las terminales y los nodos intermedios, las primitivas.

Cada uno de estos individuos es evaluado según una función de aptitud y, dependiendo del modo de selección, aquellos individuos que hayan obtenido un mejor resultado, serán más probables a ser seleccionados para que se combinen con otros individuos (Figura 2). Estos hijos entonces, pueden reemplazar a los padres. El objetivo de esta estrategia es encontrar el máximo o mínimo global del problema a resolver.

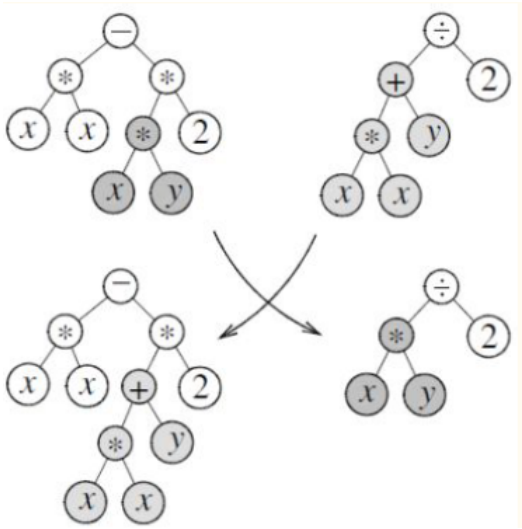

Fig. 2. Cruza entre dos individuos

Dentro de esta estrategia existen varias formas para generar a la población inicial. En este trabajo se considera el uso del full grown, que consiste en tener a todos los padres con la misma longitud. También existen varias formas de cruzar a los individuos. La forma que se usará en el trabajo actual, es la que se representa en la figura 2.

A diferencia de los algoritmos evolutivos, la mutación de individuos dentro de la población se da con una probabilidad muy inferior a la de la cruza. Dentro del proceso de este trabjo se elimina la mutación dentro del algoritmo.

4 Método

Gracias a la población y las combinaciones de los individuos dentro de la programación genética, el encontrar la mejor función de adquisición usando optimización bayesiana como medio de evaluación se presenta como el aporte de este trabajo.

Para llevar a cabo esta tarea se eligieron como terminales la media y la desviación estándar que genera el modelo sustituto, así como también tres constantes que actuarán como posibles parámetros de regularización para buscar la mejor relación entre explotación y exploración. Las primitivas constan de operaciones binarias, tales como la suma, resta, multiplicación y división. Estas operaciones están presentes en las AFs descritas anteriormente.

Aunque el método propuesto tiene un gran poder en encontrar la solución óptima dada una población y un número de generaciones, atadas a la evaluación de la función de aptitud. El procesar cada uno de estos individuos dentro de la optimización bayesiana puede aumentar el tiempo de cómputo si no se eligen los parámetros adecuados para cada problema. Además, el problema se agraba si se desea encontrar una AF que sea capaz de funcionar ante varios tipos de problemas.

5 Experimentos y resultados

Para tener el mayor número de experimentos, en lugar de usar un umbral en la función de aptitud, se establecieron cuatro generaciones para todos los experimentos. También se disminuyeron el número de puntos aleatorios para calcular con cada AF y se balancearon con el número de iteraciones del proceso de optimización. De los 10,000 puntos aleatorios a evaluar, se redujeron a 2,000. Tampoco se le proporcionó el mejor resultado generado por el modelo sustituto, esto, para determinar la efectividad de la función de adquisición genética (GAF).

La población inicial de todos los experimentos es de 1000 individuos, con un tamaño inicial de 7 a 14 caracteres. El ruido que se agrega a la AF, se trata de ruido gaussiano con centro en el cero.

5.1 Especificaciones del equipo de pruebas

Los experimentos se llevaron a cabo en una computadora portatil HP, con un procesador Intel Pentium Silver N5000 @ 1.10 GHz, con 4 GB de memoria RAM y un sistema operativo windows 10 de 64 bits.

5.2 Software y librerías

El lenguaje en el que está implementado el algoritmo es python, con la versión 3.8.2. Las librerías que se usaron son: math, numpy, scipy, sklearn, matplotlib y random.

5.3 Funciones de prueba

Las funciones de prueba que se usaron en los experimentos se muestran en las figuras 3 a 6. Se tomaron en cuenta estas funciones ya que tres de ellas cuentan con múltiples mínimos locales. La figura 6 tiene, aunque no tiene múltiples mínimos, la velocidad con la que llega a su mínimo global es de interés.

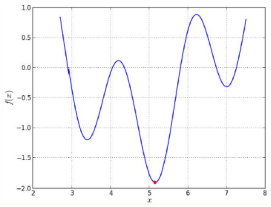

Fig. 3. Función de prueba 1

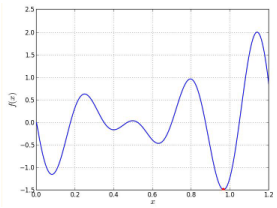

Fig. 4. Función de prueba 2

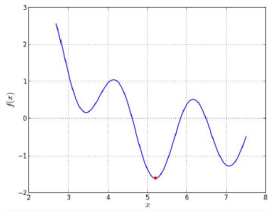

Fig. 5. Función de prueba 3

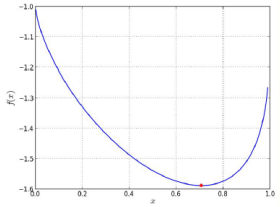

Fig. 6. Función de prueba 4

5.4 Experimento 1

En este primer experimento se usó la primera función de prueba (figura 3). Los parámetros que se usaron en el algoritmo, son los siguientes:

- Noise: 1.3
- Domain: 2.5 - 7.5
- Optimization iter.: 200
- Random Points 10

La función de adquisición genética resultante se generó con una aptitud de 0.38957594, y corresponde a:

$$GAF_1 = \frac{3 + u(1 + v)}{u} \tag{1}$$

Donde u equivale a la media y v a la desviación estándar. Estas variables corresponden a estos mimos valores para los siguientes experimentos.

Los resultados de este experimento se resumen en la tabla 1. Como podrá notar, el valor más bajo lo encontró la función de adquisición de la mejora esperada, sin embargo, este no corresponde a el mínimo global, el cual coincide con el que se obtuvo con la GAF. Note también, que el valor mínimo es el doble que el real. La razón de esos valores radica en el ruido que se agrega a el sistema.

	x	y
EI	0.916	**-4.954**
GAF	**5.138**	-4.305
Real	*5.1457*	*-1.8995*

Table 1. Resultados del experimento 1

5.5 Experimento 2

En el segundo experimento se usó la segunda función de prueba (figura 4). Los parámetros que se usaron en el algoritmo, son los siguientes:

- Noise: 1.3
- Domain: 0 - 1.2
- Optimization iter.: 200
- Random Points 10

La función de adquisición genética resultante se generó con una aptitud de 0.37968088, y corresponde a:

$$GAF_2 = \frac{7}{7 + u + 10v} \tag{2}$$

Los resultados de este experimento se resumen en la tabla 2. En este caso, solo se cambió el dominio, de esta forma se podría ver el efecto del ruido en el modelo y, como se esperaba, el ruido provocó que no se encontrara el máximo global, sin embargo, entre ambas AF, la génetica se acerca más al valor real.

	x	y
EI	0.071	**-3.755**
GAF	**0.558**	-3.626
Real	*0.9661*	*-1.3891*

Table 2. Resultados del experimento 2

5.6 Experimento 3

En el tercer experimento se usó la tercera función de prueba (figura 5.3). Los parámetros que se usaron en el algoritmo, son los siguientes:

- Noise: 1.3
- Domain: 2.5 - 7.5
- Optimization iter.: 200
- Random Points 10

La función de adquisición genética resultante se generó con una a ptitud de 1.26071199, y corresponde a:

$$GAF_3 = \frac{3}{u(1+v)} \tag{3}$$

Los resultados de este experimento se resumen en la tabla 3. En este experimento se observó que el número de generaciones no es el sufienciente para todos los casos, ya que en este caso, la función de adquisición no encontró el máximo global y se encuentra muy cerca de la EI.

	x	y
EI	7.133	-5.562
GAF	**7.050**	**-5.631**
Real	*5.19978*	*-1.6013*

Table 3. Resultados del experimento 3

5.7 Experimento 4

En el último experimento se usó la función de prueba (figura 6) restante. Los parámetros que se usaron en el algoritmo, son los siguientes:

- Noise: 0.23

– Domain: 0.0001 - 0.99

– Optimization iter.: 200

– Random Points 10

La función de adquisición genética resultante se generó con una aptitud de 0.14706475, y corresponde a:

$$GAF_4 = \frac{3}{-8 + 7u + v} \tag{4}$$

Los resultados de este experimento se resumen en la tabla 4. En este último experimento se observó que la EI no pudo localizar el mínimo global, si no que también la GAF se encuentra muy cerca del valor real.

	x	y
EI	0.296	-2.041
GAF	**0.799**	**-2.132**
Real	*0.7071*	*-1.5874*

Table 4. Resultados del experimento 4

5.8 Gráficas

Las funciones que la optimización Bayesiana reconstruye a partir del uso de las GAFs, se observan en las gráficas de la 7 a la 10.

6 Conclusiones

Se ha mostrado que dentro de la optimización en una dimensión, las GAFs dan mejores resultados que las AFs clásicas. Además,esto, con menos información, menos iteraciones y un número reducido de generaciones.

Se encontró que existe un umbral done el nivel de ruido hace más difícil encontral el valor óptimo, sin embargo, es probable que con la AF adecuada, se pueda solventar este problema.

El uso de programación genética tiene el potencial de encontrar una AF que se pueda usar en varios tipos de problemas, donde además, la información del gradiente no está disponible.

Además se observó que aquellas GAFs que obtuvieron mejores resultados, tienen como operador, al menos una división. Lo que da a entender que probablemente el valor resultante de estas funciones tiene que ser un número pequeño. Se requiere realizar más experimentos con generaciones condicionadas a un umbral y con una población más grande.

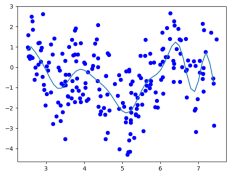

Fig. 7. GAF 1

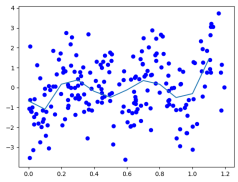

Fig. 8. GAF 2

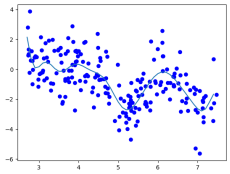

Fig. 9. GAF 3

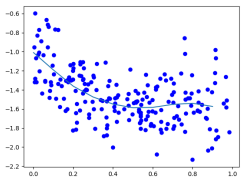

Fig. 10. GAF 4

References

1. Xin He, Kaiyong Zhao, Xiaowen Chu. AutoML: A Survey of the State-of-the-Art. Knowledge-Based Systems. (2020)
2. Hao Wang, Bas van Stein, Michael Emmerich, Thomas Back. A New Acquisition Function for Bayesian Optimization Based on the Moment-Generating Function. IEEE International Conference on Systems, Man, and Cybernetics (SMC). (2017)
3. James T. Wilson, Frank Hutter, Marc Peter Deisenroth. Maximizing acquisition functions for Bayesian optimization. 32nd Conference on Neural Information Processing Systems. (2018)
4. Michael Volpp1, Lukas P. Fröhlich, Kirsten Fischer, Andreas Doerr, Stefan Falkner, Frank Hutter, Christian Daniel1. Meta-learning acquisition functions for transfer learning in Bayesian Optimization. ICLR. (2020)