Sistemas de Inteligencia Artificial

Trabajo Práctico Especial IV

Grupo 4

Docentes

Parpaglione, María Cristina

Luciani, Ignacio José

Pierri, Alan

Alumnos

Badi, Leonel

Farré, Lucas Andrés

Gómez, Jorge Exequiel

1. Descripción

1.1. Objetivo

Implementar un motor de algoritmos genéticos el cual tenga la capacidad de obtener los pesos de una red neuronal multicapa. En particular se buscará obtener un sub óptimo de pesos para el problema presentado en el TPE 2.

1.2. Implementación

1.2.1. Representación de los individuos

Los individuos se representan utilizando una estructura de *Octave*. Cada elemento tiene la propiedad *w1*,*w2*, *fitness* y η , donde:

- *w1* es una matriz de números que representan los pesos que van de la entrada hacia la capa oculta,
- *w2* es una matriz de números que representan los pesos que van desde la capa oculta hacia la salida,
- *fitness* es un número que representa la aptitud de ese individuo. En nuestro caso se tomó el fitness como *1/error cuadrático medio*, y
- η , el valor de η utilizado en el *backpropagation*.

Se decidió representar los pesos como una matriz para hacer más fácil la integración con las funciones necesarias para el back propagation del TP E2.

La población total se representa como un vector de las estructuras definidas anteriormente. De esta manera se puede iterar y modificar subconjuntos de la población de manera fácil (utilizando el operador ":" de octave para la selección de los mismos).

Esta representación de los individuos es fiel ya que mantiene exactamente la misma representación que tiene la red neuronal. Es decir, representamos a los pesos como matrices de igual manera que lo hicimos con la red.

1.2.2. Cruce

Se implementaron los métodos de cruce, "cruce de un punto", "cruce de dos puntos", "cruce anular" y "cruce uniforme". Cada uno de ellos recibe como parámetro dos elementos de la población (reciben dos estructuras explicadas en el punto 1.2.1). Como el método de de cruza uniforme debe recibir además la probabilidad de realizar la cruza en cada alelo, se le pasa una estructura llamada co que tiene la propiedad p (co.p).

1.2.3. Selección

Se implementaron diversos métodos de selección, *Elite*, *Ruleta*, *Boltzmann* y *Torneos* entre otros. Cada uno de ellos recibe como parámetros la población y la cantidad de individuos a seleccionar *k*. Como algunos necesitan además otros parámetros, se decidió implementar una interfaz genérica, la cual además de recibir los parámetros comunes, recibe también uno llamado *c*, que consiste en una estructura de *Octave* que contiene todos los otros parámetros especiales de cada método.

Cabe destacar que para los métodos mixtos implementados, ya sea el de *Elite + Ruleta* como el de *Elite + Universal*, asumimos que ambas selecciones se realizan sobre el total de la

población, por lo cual puede ocurrir que se repitan individuos. De no ser así, cómo la interfaz definida retorna la subpoblación, y no los índices de los individuos seleccionados de la población, se vuelve muy compleja la refactorización del código.

1.2.3. Mutación

Para las mutaciones se decidió implementar tres métodos de mutación: mutación lineal (*mutateLineal*), mutación linealmente decreciente (*mutateLinealM*) y mutación no lineal (*mutateNoLineal*).

Para la mutación lineal se decidió que cada uno de los pesos del individuo a mutar cambie con una probabilidad p_m de la siguiente manera:

$$w_{new} = w_{old} + 2 \cdot (r - 0.5) \cdot s \,,$$

donde w_{new} es el nuevo peso, w_{old} es el viejo peso, r es un número aleatorio entre 0 y 1 y s es un factor de ajuste de cambio que llamamos *strength*.

Para la mutación linealmente decreciente se reduce la fuerza de mutación de forma lineal de la siguiente manera:

$$w_{new} = w_{old} + 2 \cdot (r - 0.5) \cdot s \cdot (1 - m \cdot i) \,,$$

Donde m es el coeficiente de ajuste tomando su valor de modo que al final del paso de las generaciones valga 0.

En cambio, para la no lineal, el cambio se realiza de la siguiente manera:

$$w_{new} = w_{old} + 2 \cdot (r - 0.5) \cdot s \cdot (1 - tanh(b \cdot i)) \,,$$

donde b es un coeficiente de ajuste que depende de la cantidad total de iteraciones e i es el número de la generación.

Se definió una interfaz única para las funciones de mutación, que recibe como parámetros el individuo a mutar y, al igual que para la selección, otra estructura *c* que contiene los parámetros especiales para cada uno de ellos.

1.2.3. Reemplazo

Se definieron los métodos de reemplazo 1,2 y 3. Como cada uno de estos métodos son los encargados de llamar a las funciones de selección, cruza y mutación, debe recibir las estructuras y los parámetros que estas necesitan. La brecha generacional no se pasa explícitamente sino que se pasa un k que representa la cantidad de individuos a seleccionar.

1.2.3. Criterios de corte

El algoritmo finaliza una vez que se cumpla cualquiera de los siguientes métodos de corte:

- Máxima cantidad de generaciones alcanzadas.
- Se encontró un individuo que cumple con el fitness deseado.
- Contenido. El mejor fitness no progresa en una cierta cantidad de generaciones.
- Estructura. Gran parte de la población no mejora luego de una cierta cantidad de generaciones. Para ello se tomó como parte relevante de la población el 80%.

Para el desarrollo de los tests para elegir los mejores valores de los parámetros, se utilizó como único método de corte la máxima cantidad de generaciones alcanzadas, ya que se tomó como principal criterio de comparación el mejor *fitness* obtenido.

2. Resultados

2.1. Realización de pruebas

Debido a la gran cantidad de parámetros y métodos de selección, cruza , reemplazo y mutación posible, se decidió tomar unos parámetros y métodos bases para realizar la mayoría de las pruebas. De esta manera al querer realizar una comparación sobre un tipo de métodos, por ejemplo de cruza, se dejan todos los métodos constantes y solo se cambia lo que queremos analizar.

Si bien cada método puede presentar comportamientos diferentes al ser usado con otros métodos o parámetros, las pruebas exhaustivas hubiesen sido imposible de realizar.

Los métodos base que se utilizaron son:

- Selección : torneos
- Cruza: uniforme
- Mutación: mutación lineal.
- Reemplazo: método 3
- Selección para reemplazo: elite+torneos.

Para ver con más detalle los parámetros para cada método ver la tabla A.1.

Para la realización de todas las pruebas se utilizó la misma población inicial para que no haya diferencias al inicializarla.

2.2 Métodos

2.2.1 Selección

Luego de varias pruebas, con distintos parámetros para cada método y observando los resultados obtenidos en la tabla A.2 del anexo, podemos destacar lo siguiente:

- El clásico método Elite no alcanzó nuestro valor deseado de fitness superior a 100.
- El método Boltzmann presenta variaciones de fitness desde 88,751 hasta 127,98 dependiendo de la función de decremento de la temperatura utilizada. La que presentó el mejor fitness fue una función seno montada sobre una lineal:

$$T = T - m \cdot t - sen(0,5 \cdot t) + 1$$

La misma fue probada con T y m tales que la temperatura inicial fuera 100 y la final -100.

- El método de Torneos con m igual a 2 resultó mejor que con m igual a 3, alcanzando un fitness de 120,56 en lugar de 104,69.
- Los algoritmos mixtos presentaron valores variados de acuerdo al valor de N1 elegido, el mejor fue el del 20% de k, alcanzando para ambas combinaciones un valor de fitness alrededor de 120. Se pudo observar que los mismos alcanzaban el valor de fitness de 100 en menos de 300 generaciones, pero luego se estancaban por muchas generaciones. Por ejemplo para Elite+Ruleta, una vez alcanzado el valor de 122,29, el mismo se mantuvo en ese valor por más de 800 generaciones.

2.2.2 Cruza

Analizando los resultados obtenidos luego de realizar las pruebas, que se encuentran en la tabla A.3 del anexo, se pudo observar que el método de cruza que encontró un mejor valor para 1500 generaciones fué el *onePointCross*, obteniendo un *fitness* de 114,72 con los parámetros de testeo. Si bien este fué el mejor resultado, el *uniCross* y el *anularCross* también arrojaron muy buenos resultados, obteniendo un *fitness* de 111,34 (Para un valor de uniP de 0,6) y 110,61 respectivamente. Cómo los resultados pueden variar respecto de otros si los pesos se inicializan al azar, cualquiera de estos métodos es una buena elección para la ejecución del algoritmo.

2.2.3 Reemplazo

Fueron probados los distintos métodos de reemplazo (1, 2 y 3) con distintos parámetros, para ver cómo se comportaban. La mejor configuración resultó ser la que utilizaba la selección Elite+Ruleta y un valor de k igual a 10. La misma alcanzó un valor de fitness igual a 143,76. Se pueden observar los resultados con más detalle en la tabla A.4 del anexo.

2.2.4 Mutación

Para las pruebas de mutación se desarrolló un *script scriptTestMutation* que realizaba pruebas con las posibles combinaciones posibles para encontrar los mejores parámetros dado una función de mutación. Los resultados del script se encuentran en la tabla A.5 y A.6 del anexo. Se pudo observar como la mutación lineal obtuvo los mejores resultados, un *fitness* de 207,78 para los valores de 0,1 probabilidad de mutación y 0,3 fuerza de mutación. En cambio se obtuvieron buenos resultados con la mutación no lineal, esto se debe al rápido decrecimiento de la tangente hiperbólica para primeros valores de las iteraciones. Por este motivo se decidió implementar la mutación linealmente decreciente para poder comparar resultados, y logró obtener un *fitness* de 167,58.

2.3 Parámetros

2.3.1 Eta Adaptativo

Al momento de realizar las primeras pruebas del algoritmo, se notó que no se lograba llegar al un fitness cercano al esperado en una cantidad razonable de generaciones. Por más de que se utilizaba el *Back Propagation*, el fitness de la población no mejoraba considerablemente. Se necesitaba una gran cantidad de generaciones (más de 10000) para lograr un *fitness* aproximado de 60.

Debido a estos resultados se decidió modificar los valores de incremento y decremento de eta, ya que como es lógico, al correr el algoritmo de *Back Propagation* una cantidad mucho menor de épocas se necesitaban otros valores de incremento y decremento de eta para obtener un mejor resultado. A esto se obtuvo los nuevos valores de incremento, 0.06 y decremento, 0.02, utilizando un script de generación de valores posibles que guarda los datos en un archivo.

Aún así no se encontraron los resultados esperados y se notó lo siguiente: El eta inicial utilizado cada vez que se llama al *perceptron* era siempre el mismo. Esto nos llevó a notar que en cada inicio de entrenamiento de cualquier individuo, el eta inicial tomaba los valores 0.5 o 1.0 dependiendo la configuración y eran valores muy altos una vez que los individuos llegaban a un gran fitness. Esto se producía ya que los pesos estaban cerca de una buena solución pero el eta alto los hacía variar mucho en el momento de entrenamiento.

Como solución, se decidió que cada individuo contenga el último eta utilizado, utilizando la mejora del eta adaptativo del *perceptron*. De esta forma luego de realizar el entrenamiento cada individuo contiene su propio eta utilizado, de manera que el mismo vaya disminuyendo conforme al aumento de generaciones. Para el método de *crossover* se decidió que el eta de los hijos es el promedio del eta de los padres. Luego de realizar esta implementación, se notó una gran mejora en los resultados, se pudo obtener un *fitness* mucho mayor al esperado en una buena cantidad de generaciones. Aún así se notó que por un largo tiempo el *fitness* de los individuos permanecía casi constante en valores cercanos a 1, hasta luego llegar a un gran salto en una cierta cantidad de iteraciones.

Se analizó detalladamente las iteraciones y el problema encontrado fue el esperado: El eta inicial de 0.05 era muy chico para los inicios de la población, ya que su fitness empezaba a aumentar drásticamente luego de que pasaban los valores de 0.2. Para encontrar el mejor valor de eta, se desarrolló un segundo *script*, que, en unas pocas generaciones (100 para ser exactos) calculaba el fitness de la población y guardaba la evolución del fitness cada 10 iteraciones. Como resultado se obtuvo que uno de los mejores valores iniciales de eta es 0.2 para nuestra configuración y lograba alcanzar un *fitness* de 90 en 100 generaciones.

3. Conclusiones

Teniendo en cuenta las distintas pruebas que realizamos y los resultados obtenidos sacamos las siguientes conclusiones:

- No solo es importante variar los parámetros del algoritmo genético, sino también del backpropagation utilizado para entrenar a los individuos. Esto se pudo observar al variar el η, pues de no modificarlo la performance era menor, pues los fitness obtenidos no alcanzaban valores superiores a 40 en menos de 1000 generaciones.
- Los distintos métodos no son independientes, pues modificando solo uno de ellos y dejando fijo el resto para evaluar el fitness resultante y luego probar una nueva configuración con todos los métodos y parámetros que obtuvieron el mejor fitness por separado, no resultó la mejor, como hubiéramos esperado.
- La única manera de encontrar la mejor configuración es probando exhaustivamente cada una de todas las combinaciones de métodos y parámetros posibles.
- La combinación de distintos métodos puede ser favorable, pues por ejemplo los métodos de selección mixtos resultaron mejores que los mismos por separado. Además, por ejemplo un método puede ser bueno en las primeras iteraciones, pero luego hacer estancar al algoritmo en las siguientes generaciones. En este caso se podría utilizar ese método al principio y luego cambiarlo por otro una vez que alcanza una cierta cantidad de generaciones. Por ejemplo, hubiera sido interesante probar utilizar al principio los métodos mixtos, ya que los mismos alcanzaban valores de fitness superiores a 100 en menos de 300 iteraciones, y luego utilizar Torneos, que a diferencia de los anteriores se estancaba menos. Lamentablemente no pudimos probarlo, ya que el código no estaba preparado y hubiera sido costoso refactorizarlo para que lo soportara.
- Como se puede ver en la tabla A.8 se alcanzó un fitness de 167,53 superando el 91 que se alcanzó en el TPE2 con un back-propagation puro. Si se analizan todas las tablas, en la gran mayoría se superó el valor esperado de 100 de fitness
- El algoritmo de back-propagation aporta significativamente al fitness resultante.

A. Anexo

A.1 Tabla de parámetros base.

Genético	hidenN	5
	iterations	1500
	k	6
	n	40
	fitnessWish	300
	timeToMakeProgress	3000
	relevantpercent	0.8
	structuraliterationsallowed	20
Perceptrón	useBackPropagation:	1
	alpha	0.9
	eta	0.3
	etaAdaptation	1
	throwWeights	0
	etadec	0.2
	etainc	0.08
	iterations	4
Reemplazo	replaceMethod	replaceTwo
Cruza	crossOver	uniCross
	unip	0.5
Mutación	mutation	mutateLineal
	mutateProbability	0.02
	mutateStrength	0.2
Selección	selectMethod	torneos
	selectReplaceMethod	eliteruleta
	tm	3
	n1	0.2

A.2. Tabla de métodos de selección.

Método de selección	**Mejor fitness**	**Fitness promedio**
Elite		
	71,298	69,13
Ruleta		
	112,57	99,72
Torneos		
m = 3	104,69	95,709
m = 2	120,56	118,17
m = 2 (45400 iteraciones)	201,33	179,345
Boltzmann		
Lineal (Ti=373, Tf=273)	104,66	93,19
Lineal (Ti=1337.33, Tf=273)	88,751	80,585
Lineal sobre seno (Ti=100, Tf=-100)	**127,98**	119,82
f(t) = 0.2 * generaciones / t	99,955	92,69
f(t) = 1 / t + 0.5	98,138	88,56
f(t) = 1 - t / N + 0.5	96,172	78,52
f(t) = 373 - 373 * t / generaciones + sin(5 * t) + 1	104,59	101,455
Elite+Ruleta		
N1 = 20%	122,29	112,675
N1 = 10%	110,44	108,005
N1 = 25%	100,16	86,55
N1 = 15%	91,468	86,085
Elite+Univesal		
N1 = 20%	120,59	100,496

Nota: Todos los resultados son los obtenidos luego de 1500 iteraciones salvo que especifique lo contrario.

A.3. Tabla de métodos de cruza.

Método	Mejor fitness	Fitness promedio
Dos puntos		
	98,877	97,06
Clásico		
	114,72	109,445
Anular		
	110,61	105,58
Uniforme		
uniP = 0,3	105,092535	93,583354
uniP = 0,4	107,567320	100,5179391
uniP = 0,5	95,958323	94,66736665
uniP = 0,6	111,339129	109,5526556
uniP = 0,7	104,124820	101,3487364

Nota: Todos los resultados son los obtenidos luego de 1500 iteraciones.

A.4. Tabla de métodos de reemplazo.

Método	Mejor fitness	Fitness promedio	Razón de corte	Generación
Método 1				
K = 6	117.12	106,7	Máximo de generaciones	1500
Método 2				
K=6	86.035	67,85	Tiempo sin mejoras	785
K=10	**143.76**	116,35	Tiempo sin mejoras	1314
Método 3				
K=6	102.56	99,05	Máximo de generaciones	1500
K=10	115.99	107,1	Máximo de generaciones	1500

A.5. Tabla de métodos de mutación nula y lineal.

Mutación			**Probabilidad (mutateProbability)**	**Magnitud (mutateStrength)**
Nula				
	76,127990	75,18512845	0	0
Lineal				
	102,557137	66,9997468	0,05	0,2
	113,456938	101,3762548	0,05	0,3
	131,103116	119,6315239	0,05	0,4
	139,872609	124,1659505	0,1	0,2
	207,780411	**151,6792985**	**0,1**	**0,3**
	144,449257	100,5160368	0,1	0,4
	142,451716	131,2151249	0,15	0,2
	145,897794	134,7678386	0,15	0,3
	133,394994	101,5143798	0,15	0,4
	157,390681	123,6179185	0,2	0,2
	159,841416	132,7878915	0,2	0,3
	127,419872	105,0775295	0,2	0,4

Nota: Todos los resultados son los obtenidos luego de 1500 iteraciones.

A.6. Tabla del método de mutación no lineal.

Mutación			**Probabilidad (mutateProbability)**	**Magnitud (mutateStrength)**
No lineal				
	86,420531	80,7439213	0,05	0,2
	104,012118	102,1855394	0,05	0,3
	98,271447	94,9379992	0,05	0,4
	114,078612	113,5724032	0,1	0,2
	105,802650	103,698854	0,1	0,3
	103,945338	100,588309	0,1	0,4
	100,677534	98,90503415	0,15	0,2
	114,303667	105,1058509	0,15	0,3
	112,170032	99,6768872	0,15	0,4
	58,870840	56,55837445	0,2	0,2
	108,893765	97,40119115	0,2	0,3
	120,264352	**110,8443615**	**0,2**	**0,4**
	108,634654	103,8237413	0,05	0,2
	118,108488	115,9049494	0,05	0,3
	119,976171	114,8520135	0,05	0,4
	120,226385	103,6076965	0,1	0,2
	118,874412	100,2088452	0,1	0,3
	112,343954	112,0128461	0,1	0,4
	113,494826	87,65087605	0,15	0,2
	114,055925	112,4845964	0,15	0,3
	109,298712	77,95606245	0,15	0,4
	167,588808	**125,1120355**	**0,2**	**0,2**
	134,804620	103,321765	0,2	0,3
	107,753088	82,26200355	0,2	0,4

Nota: Todos los resultados son los obtenidos luego de 1500 iteraciones.

A.7. Tabla fitness a través de 100 generaciones comparando diferentes valores de eta y utilizando los parámetros de testing.

Eta	10 G	20 G	30 G	40 G	50 G	60 G	70 G	80 G	90 G	100 G
0,15	0,90969	0,90969	0,90969	0,90969	0,90969	0,90969	0,90969	0,90969	0,90969	0,90969
0,20	**2,02713**	**70,2817**	**70,4998**	**72,3478**	**77,6744**	**83,3126**	**84,9575**	**86,1385**	**86,2230**	**90,7021**
0,25	20,7141	66,6432	70,7897	78,1075	80,5508	82,3531	82,4802	85,8833	86,1988	86,6599

A.8. Tabla de resultados utilizando la combinación de los mejores métodos y parámetros.

Épocas de Back Propagation	Mejor Fitness	Fitness Promedio
0	13,173	10,882
1	80,282	38,302
2	47,597	24,379
3	116,79	75,425
4	167,53	134,75

Nota: Todos los resultados son los obtenidos luego de 1500 iteraciones.

A.8 Gráfico de fitness del mejor individuo y fitness promedio de la población.