Sistemas de Inteligencia Artificial
Algoritmos Genéticos
Trabajo Práctico Especial Nº 5

Juan Pablo Civile Alvaro Crespo Esteban Ordano

Resumen

Se implementó un motor de algoritmos genéticos con el objetivo de obtener los pesos de una red neuronal multicapa que aprenda la siguiente función: $z(x,y) = c\sin(ax-by)$.

1 Descripción del Problema

En el trabajo anterior, la mejor arquitectura de una red neuronal para el problema en cuestión tenía 9 y 6 neuronas en sus capas ocultas, 2 entradas, y una capa de salida de una neurona. Esto implica que los pesos entre las capas pueden representarse como matrices de dimensiones 3x9, 10x6 y 7x1 respectivamente, lo que da un total de 94 valores. Estos 94 valores son los alelos de los cromosomas, los individuos de la población.

Anteriormente, se utilizó el error cuadrático medio respecto de un conjunto de datos de prueba como medida de que tan buena era una red neuronal. Por lo tanto, la función de *fitness* se definió como el inverso multiplicativo del error cuadrático medio, para que mayor *fitness* se corresponda con una red neuronal con menor error.

2 Implementación

2.1 Método de Reemplazo

Se implementaron los métodos de reemplazo presentados por la cátedra. El primero de ellos, reemplaza toda la generación por nuevos individuos, provenientes de la recombinación de individuos de la generación anterior. El segundo, un método con recambio generacional que selecciona tanto individuos de la generación anterior como individuos obtenidos a partir de la recombinación de individuos de la generación anterior. El tercero, similar al segundo, genera nuevos individuos por recombinación y a través de un criterio de selección, elige entre la generación anterior y estos nuevos individuos aquellos que formarán parte de la nueva generación.

2.2 Operador Genético de Cruce

Se implementaron los siguientes operadores genéticos de recombinación o cruce:

- **Cruce de un punto**: Se elige un locus al azar y se intercambian los alelos de los dos individuos anteriores a partir de ese locus.
- **Cruce de dos puntos**: Se eligen dos locus al azar y se intercambian los alelos de los dos individuos anteriores entre estas dos posiciones.
- **Anular**: Se elige un locus al azar y se intercambian L alelos a partir de ese locus. Se utilizó $L = 10$ y $L = 30$.
- **Cruce Uniforme Parametrizado**: Para cada locus, con una probabilidad determinada, se produce el cruce. Se utilizó una probabilidad de 0.6.

2.3 Operador Genético de Mutación

Se implementaron dos ligeras variaciones del algoritmo de mutación. Ambas en sus versiones con mutación simple y no uniforme.

- **Mutación por individuo**: Se genera una mutación en el individuo con una probabilidad determinada, en un único locus seleccionado aleatoriamente.
- **Mutación simple**: Por cada locus, con una probabilidad determinada, se produce una mutación en ese alelo. Se utilizó una probabilidad de 0.01.

Para las variaciones no uniformes, se utilizó $p_i = 0.995p_{i-1}$ y $p_i = 0.999p_{i-1}$ (una variación de 0.5% y 0.1% por generación).

2.4 Criterio de Selección

Se implementaron los siguientes criterios de selección:

- **Elitismo**: Se seleccionan los individuos de mayor *fitness*.
- **Ranking**: La probabilidad de seleccionar un individuo es proporcional a su posición en una lista de los individuos ordenados por *fitness*, de menor a mayor.
- **Ruleta**: La probabilidad de seleccionar un individuo es proporcional a su *fitness*, y para cada individuo a seleccionar se utiliza un número aleatorio.
- **Estocástico**: La probabilidad de seleccionar un individuo es proporcional a su *fitness*, pero se selecciona un solo número al azar.
- **Torneo**: Se seleccionan individuos en parejas al azar y se comparan sus valores de *fitness*, y con una probabilidad determinada, se selecciona el de mayor o el de menor valor de *fitness*. Se utilizó una probabilidad de 0.75%.
- **Boltzmann**: La probabilidad de seleccionar un individuo es función de su *fitness* y de la cantidad de generaciones. Se utilizaron temperaturas $T = g$ y $T = 0.5g$, donde g es la cantidad de generaciones desde la primera.
- **Mixto**: Seleccionar los individuos según distintos criterios. Se utilizaron las variantes:
  - **Mix Estocástico y Elitismo**: Se selecciona un 30% de individuos según el criterio de elitismo, y el procentaje restante de acuerdo al criterio estocástico. También se utilizó 70% según el criterio estocástico y 30% según el criterio de elitismo.
  - **Mix Ruleta y Elitismo**: De manera similar al punto anterior, pero utilizando el criterio de ruleta en vez del estocástico.

2.5 Criterio de Corte

Se implementaron los siguientes criterios de corte:

- **Cantidad de Generaciones**: El algoritmo se ejecuta hasta llegar a un límite máximo de generaciones creadas.
- **Estructura similar**: El algoritmo se ejecuta hasta que se detecta que generaciones sucesivas están compuestas de individuos muy similares. Se comparan las diferencias absolutas entre los alelos de todos los individuos. Una variante de este criterio es seleccionar un muestreo al azar de individuos de ambas generaciones.
- **Contenido similar**: El mejor *fitness* de la población no varía con el tiempo.
- ***Fitness* buscado**: El algoritmo se ejecuta hasta que el mejor *fitness* de una generación supera una cota determinada.
- **Criterio mixto**: El algoritmo corta cuando se dá una de dos condiciones: el mejor *fitness* supera un valor buscado o la cantidad de generaciones supera una cantidad límite.

El criterio utilizado en todas las pruebas realizadas fue un criterio mixto, donde el algoritmo detenía su ejecución después de 200 generaciones o al alcanzar un *fitness* de 1000.

2.6 Explosión combinatoria

La numerosa cantidad de parámetros variables del motor de algoritmos genéticos presenta la dificultad de que conseguir una configuración óptima sea muy complejo.

En particular, para los métodos de reemplazo 2 y 3, se deben elegir dos criterios de selección (que pueden ser distintos), con lo cual teniendo 10 criterios de selección, son 100 distintas combinaciones. Otro parámetro que entra en juego aquí es el coeficiente de reemplazo generacional.

En todo método de reemplazo, también se debe elegir entre 5 operadores genéticos de recombinación, 4 operadores genéticos de mutación.

Todas estas distintas opciones se traducen en una cantidad de ejecuciones del algoritmo genético en el orden de 10^5, si el objetivo es comparar todas las posibles configuraciones del motor. Dado que esta cantidad de ejecuciones escapa al poder de cómputo disponible se decidió hacer un analisis reducido.

Se estudió:

- **Cantidad de épocas de *backpropagation***: Se seleccionó una cantidad de épocas de backpropagation en una solución de compromiso entre resultados obtenidos y velocidad de ejecución.
- **Cantidad de individuos por generación**: Se buscó una cantidad de individuos que genere resultados aceptables sin requerir mucho poder de procesamiento.
- **Criterio de selección**: Se buscó, utilizando el método de reemplazo 1, sin mutaciones, y con cruce de un punto, cuál es el método de selección que dió mejores resultados para valores de $N = 80$.

- **Operadores genéticos de Cruce**: Utilizando el método de reemplazo 1 y el criterio de selección mixto (estocástico y 30% elitismo), se buscó el mejor de los operadores genéticos de cruce para $N = 80$.

- **Criterio de selección para reemplazo**: Se buscó el mejor método de selección para ejecutar el reemplazo generacional en el método de reemplazo 2. Se utilizó un valor de $N = 80$ y se probó con un coeficiente de reemplazo generacional de 60%.

- **Comparación entre métodos de reemplazo**: Se compararon los métodos de reemplazo 2 y 3 con los mejores algoritmos de selección encontrados.

3 Resultados

3.1 Backpropagation

Se ejecutó el algoritmo genético con una cantidad de ejecuciones del algoritmo de *backpropagation* muy chica (entrenando la red unas 2000 veces a lo largo de distintas generaciones). Los resultados son comparables con aquellos obtenenidos de una red neuronal con valores seleccionados aleatoriamente, como se muestra en el anexo en 1.

Se utilizó una probabilidad de 0.05 de ejecutar el algoritmo de *backpropagation* con un conjunto de entrenamiento de 400 elementos y 40 épocas del algoritmo. Estos valores no son óptimos, si no que fueron elegidos para poder comparar los distintos parámetros de configuración utilizando poco poder de procesamiento.

3.2 Número de individuos por Generación

En la figura 5 del anexo, se comparan los promedios de ejecución con distintos métodos de selección para distintos valores de N. Se utilizó $N = 80$ en todas las sucesivas pruebas.

3.3 Operador genético de Cruce

La tabla 1 muestra los valores obtenidos de ejecuciones con distintos operadores de cruce y los resultados obtenidos. A partir de estos resultados se decidio fijar el operador de Cruce a Anular con $L = 10$.

Criterio	Mejor	Promedio
Uniform Crossover 0.6	195.928	178.646
Single Crossover	389.887	356.701
Double Crossover	423.335	408.75
Anular, L = 30	196.316	280.827
Anular, L = 10	524.731	524.731

Table 1: Fitness obtenida para los distintos operadores de Crossover

3.4 Selección de individuos

Para el método de reemplazo 1, la tabla 2 muestra, luego de 200 generaciones, los valores de fitness obtenidos. El operador genético de cruce utilizado fue anular con $L = 10$. En el anexo (figuras 6, 7, 3, 4, 2) se muestra la evolución a lo largo de distintas generaciones de algunos de estos algoritmos. A partir de esto, se fijo el metodo de seleccion de cruce como Mix Roulette + 70% Elite.

Método	Mejor	Promedio
Tournament 0.75	449.918	447.498
Roulette	380.219	149.6
Rank	232.019	232.019
Elite	277.545	264.440
Boltzmann T = generations	320.483	306.531
Boltzmann T = 0.5 * generations	170.122	161.708
Mix Stochastic + 70% Elite	267.148	252.895
Mix Roulette + 70% Elite	505.591	446.057
Mix Roulette + 30% Elite	447.594	399.834
Mix Stochastic + 30% Elite	515.609	506.539

Table 2: Valores de fitness para distintos métodos de selección

3.5 Método de Reemplazo 2

Se buscó un valor para G, usando Mix Stochastic + Elite 30% y luego elitismo:

G	Mejor	Promedio
0.6	524.731	524.731
0.75	364.749	353.851
0.9	284.096	273.342

Table 3: Fitness obtenido para distintos valores de G

Selección de los individuos de la nueva generación:

Criterio	Mejor	Promedio
Tournament 0.75	345.249	289.807
Roulette	377.249	361.807
Rank	508.293	495.604
Elite	375.369	365.45
Boltzmann T = generations	284.096	273.342
Mix Stochastic + 30% Elite	364.749	353.851
Mix Roulette + 30% Elite	450.569	446.164

Table 4: Fitness para los distintos métodos de selección

3.6 Método de Reemplazo 3

La 3.6 muestra una comparación entre los promedios de los mejores individuos y los promedios de los valores de *fitness* obtenidos utilizando estos dos métodos, con distintos algoritmos de selección. Se muestra una evolución del mejor individuo en cada generación para las mejores ejecuciones de los tres métodos de reemplazo en la figura 8.

Método	Mejor	Promedio
Reemplazo 2	508.293	495.604
Reemplazo 3	386.398	342.212

Table 5: Promedio de los mejores valores y el promedio generacional luego de 200 generaciones para distintas ejecuciones del algoritmo

4 Conclusión

En los resultados observados utilizando backpropagation como operador en el algoritmo genético (sección 3), se encontró que buenos parámetros para el motor son: utilizar el método de reemplazo 2, con el método de selección para cruce "mix estocástico + 30% elite", método de selección para reemplazo generacional "ranking", operador de cruce "anular, L=10", y un coeficiente de reemplazo generacional $G = 0.6$.

En el trabajo práctico anterior, utilizando meramente el algoritmo de *backpropagation* se obtuvo una red con un error cuadrático medio de 0.00025. La mejor red obtenida con algoritmos genéticos tuvo un valor de *fitness* de 764.166, es decir, un error cuadrático medio de 0.00130.

El uso de *backpropagation* mejora sustancialmente los individuos de la población a través de las generaciones. Si se considera a *backpropagation* un operador genético, se puede decir que es conciente de cómo funciona una red neuronal, mientras que los algoritmos de crossover implementados no consideran a la estructura si no como un vector de números.

De lo anterior se concluye que mejores resultados se obtendrían si el operador de cruce estuviese diseñado especificamente para el problema de redes neuronales, con backpropagation en todos los individuos. También sería interesante analizar cómo este operador de cruce podría considerar redes neuronales de distintas estructuras, incluso eliminando conexiones entre distintas neuronas o agregando capas.

Por último, el poder de procesamiento requerido para ejecutar el algoritmo genético con estas condiciones es uno o dos órdenes de magnitud superior al algoritmo de *backpropagation*, pero esto es evidente por estar realizando operaciones con *N* redes neuronales.

5 Anexo

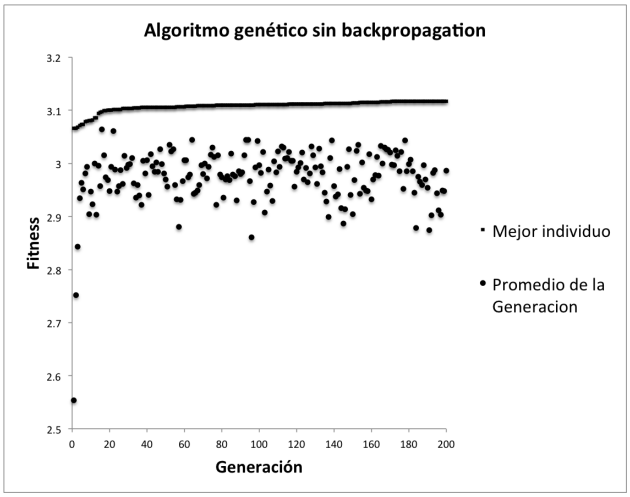

Figure 1:

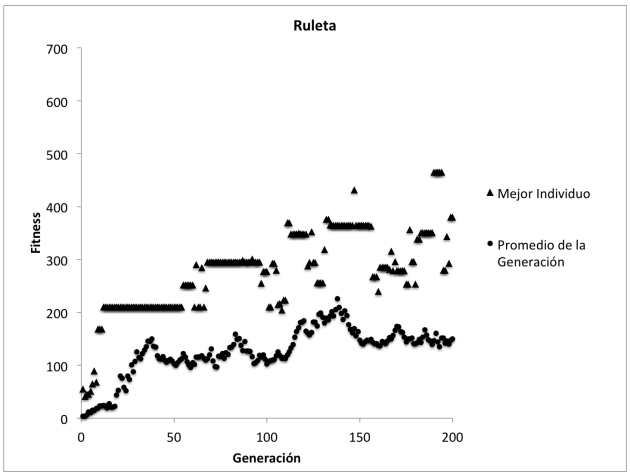

Figure 2:

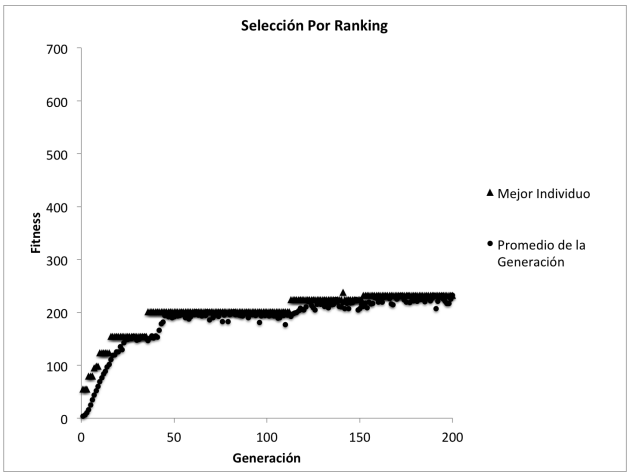

Figure 3:

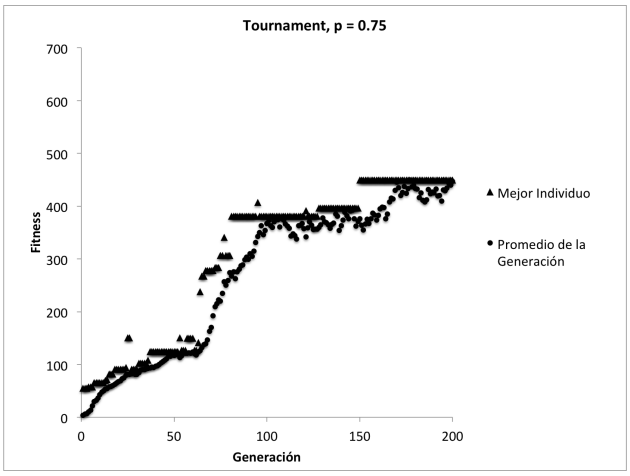

Figure 4:

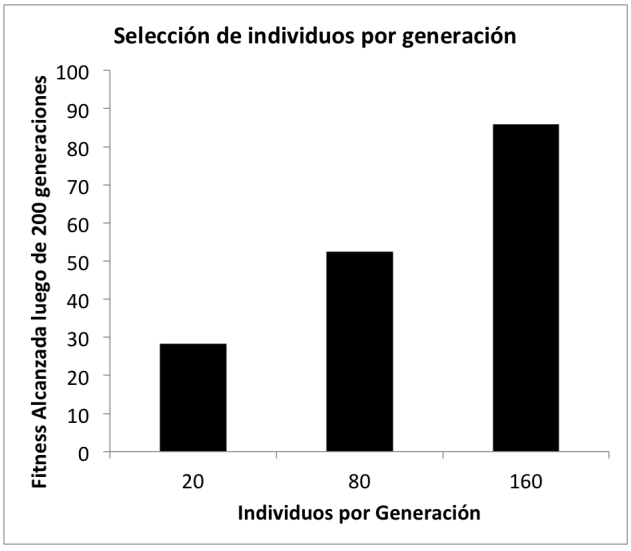

Figure 5:

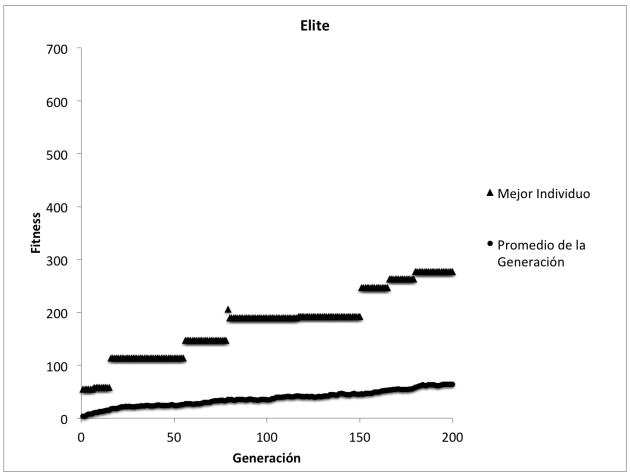

Figure 6:

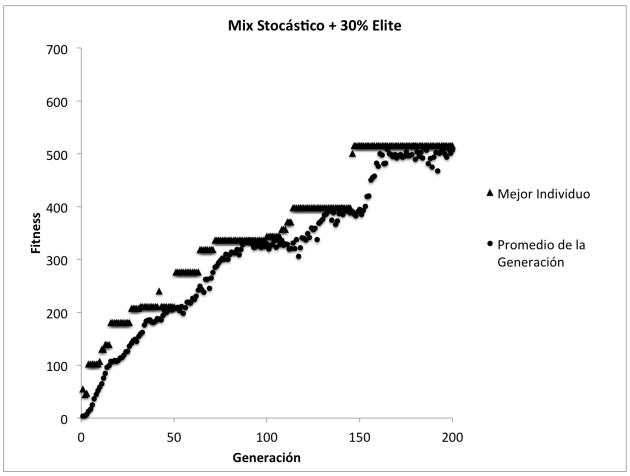

Figure 7:

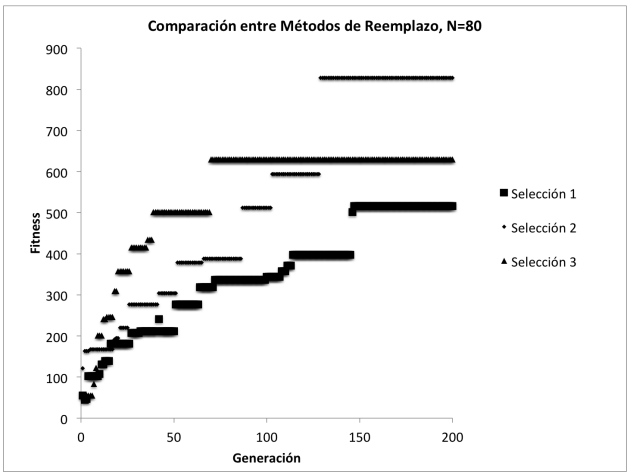

Figure 8: