基于改良跳跃基因遗传算法的浮点数编码神经网络超参数优化

洪铭乐[1)]

[1)](华南师范大学人工智能学院　广东 佛山 528000)

摘　要　针对神经网络参数优化时部分参数为整数值以及各超参取值范围不同以致无法使用许多交叉与变异算子的问题，提出了一种基于浮点数编码的改良跳跃基因遗传算法(FCJGGA)来进行神经网络的超参数优——FCJGGA-DNN(FCJGGA optimizes Deep Neural Network, FCJGGA-DNN)。算法使用的浮点数编码使得算法的搜索空间更自由，同时也解决了各超参取值范围不一致的问题，算法的交叉算子有了更多的搜索自由度。实验使用了真实数据进行测试，验证算法在预测鲍鱼年龄问题上的超参选择效率。实验结果表明，与 RCGA 算法相比，本文算法平均方差 MSE 减少了 0.040%，平均绝对误差 MAE 减少了 0.585%，且 MSE 与 MAE 的标准差分别减少了 16.393%与 22.226%。与 PSO 算法相比，本文算法平均方差 MSE 减少了 0.417%，平均绝对误差 MAE 减少了 11.185%，且 MSE 与 MAE 的标准差分别减少了 57.493%与 98.334%。

关键词　跳跃基因；遗传算法；浮点数编码；超参数优化；神经网络

中图法分类号　TP18

Hyperparameter optimization of floating-point coding neural network based on modified jumping gene genetic algorithm

Hong MingLe[1)]

[1)](Artificial Intelligence Institute of South China Normal University, FoShan GuangDong 528000, China)

Abstract　Aiming at the problem that some parameters are integer values and the range of values of each hyperparameter is so different that many crossover and mutation operators cannot be used when optimizing the parameters of neural networks, a modified jumping gene genetic algorithm (FCJGGA) based on floating point encoding is proposed to carry out the hyperparameter optimization of neural networks—FCJGGA-DNN (FCJGGA optimizes Deep Neural Network, FCJGGA-DNN). Network, FCJGGA-DNN). The floating-point encoding used by the algorithm makes the search space of the algorithm more free, and also solves the problem of inconsistent range of values of the hyperparameters, and the crossover operator of the algorithm has more freedom of search. The experiments were tested using real data to verify the efficiency of the algorithm in selecting hyperparameters for the problem of predicting the age of abalone. The experimental results show that compared with the RCGA algorithm, the mean variance MSE of this paper's algorithm is reduced by 0.040%, the mean absolute error MAE is reduced by 0.585%, and the standard deviation of the MSE and MAE are reduced by 16.393% and 22.226%, respectively. Compared with the PSO algorithm, the algorithm in this paper reduces the mean variance MSE by 0.417%, the mean absolute error MAE by 11.185%, and the standard deviation of MSE and MAE by 57.493% and 98.334%, respectively.

Key words jumping gene; genetic algorithm; floating-point code; hyperparameter optimization; neural network

1 引言

近年来，神经网络作为一种强大的机器学习工具，在各个领域取得了重要的应用，其性能很大程度上取决于其参数的选择和调整，如隐层节点数量、激活函数和学习速率等[1]。优化神经网络参数是一个复杂而耗时的任务，需要与问题的特性和数据集与相匹配的合适参数配置。因此，高效的自动调整超参数方法对提高神经网络模型的性能和训练效率具有重要意义。为了解决这个问题，研究者们不断发展出各种优化算法，如网格搜索（Grid Search，GS）、随机搜索（Random Search，RS）、贝叶斯优化（Bayesian Optimization）、进化算法(Evolutionary Algorithms，EA)。

网格搜索是一种传统的参数优化方法，通过指定参数的候选值组成的网格，在给定的参数空间中进行穷举搜索。对于每个参数组合，都进行一次模型训练和评估，最后选择表现最好的组合作为最优参数。网格搜索的优点是简单易用，适用于较小的参数空间，但在参数空间较大时计算开销较大。随机搜索是另一种基本的参数优化方法，与网格搜索不同的是，随机搜索从参数空间中随机采样一组参数进行模型训练和评估。重复这一过程多次后，选择具有最佳性能的参数组合作为最优参数。随机搜索的优点是在参数空间中更加均匀地进行搜索，且计算开销相对较小。贝叶斯优化是一种基于概率模型的参数优化方法，它通过构建一个概率模型来估计参数的性能，并使用这个模型来指导下一次参数采样的选择。贝叶斯优化采用了自适应的、序列化的参数优化策略，能够在较少的迭代次数下找到较好的参数组合。贝叶斯优化在处理高维参数空间和计算开销较大的情况下表现良好。进化算法是一类基于生物进化思想的优化方法，它通过模拟生物种群的遗传、变异和选择等操作，逐代地演化出更优的参数组合。进化算法适用于全局优化和非凸优化问题，能够在复杂的参数空间中搜索潜在的最优解。在神经网络参数优化中，进化算法常用于优化权重和偏置等参数。

近年来，许多研究工作已经将遗传算法[2]应用于神经网络参数优化，并取得了显著的成果。这些研究表明，遗传算法能够在大规模参数空间中高效地搜索，帮助神经网络找到更好的参数配置。

此外，一些研究工作还探索了遗传算法与其他优化方法的融合，如粒子群优化算法、模拟退火算法等。这种混合方法的目的是通过结合不同算法的优点，从而进一步提高神经网络参数优化的性能和效果。

尽管遗传算法在神经网络参数优化中已经取得了一定的成功，但仍然存在一些挑战和待解决的问题：

(1) 参数选择、种群大小、变异率和交叉率等优化策略的设计仍然是研究的焦点。

(2) 遗传算法应用于神经网络参数优化的效率和可扩展性也需要进一步研究和改进，以满足大规模网络和复杂任务的需求。

2 相关知识

2.1 遗传算法

遗传算法是一种启发式优化算法，受到生物遗传和进化思想的启发而发展起来。它模拟了自然界中物种进化的过程，通过遗传操作（交叉、变异）和适应度评估（选择）来搜索问题的解空间以找到最优解或近似最优解。

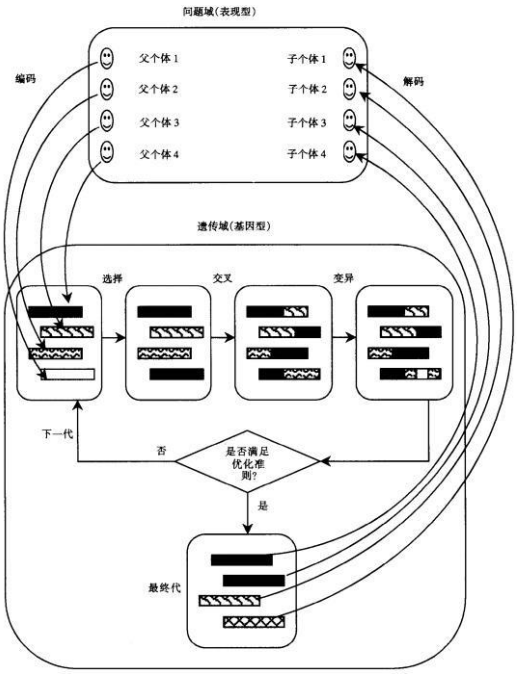


图 1 遗传算法操作示意图

Figure 1 Schematic diagram of genetic algorithm

算法主要参数有：待求解变量 x 的维度 n,种群规模，迭代次数，变异概率，交叉概率。

2.2 神经网络参数优化问题

神经网络模型的超参数优化问题可定义为：对于模型中的 n 个超参数 $\theta \in \mathbb{R}^n$，找到一组最优的超参数配置 θ'，基于超参数 θ'的网络模型具有最佳的性能评价指标 λ 。一般认为超参数 θ 与性能评价指标 λ 之间存在未知函数关系：

$$\lambda = f(\theta)$$

在神经网络超参数优化问题中，函数 $f(\theta)$为未知函数，且神经网络训练需要时间较长，而想要知道评价指标 λ 则必须通过网络模型的训练得到。故该问题为一个全局高开销的黑盒函数问题[1]。

3 基于浮点数编码的改良跳跃基因遗传算法

针对神经网络参数优化问题，本文使用 FCJGGA-DNN 进行优化。算法实现了浮点数编码，使用改良交叉算子与突变算子[3]，同时引入了跳跃基因步骤与基因密度加权计算[4]的基因适应度。

3.1 浮点数编码

由于待优化的超参数部分取值为整数[5]，如每一层神经元个数，每一层的激活函数序号（算法提供多个选择函数，从零开始编号）等，同时这些参数的取值范围各不相同，导致许多连续交叉算子与突变算子无法运用于该优化问题上。

浮点数编码对每个超参数所对应的基因 c_i 编码一个[0,1)区间内的浮点数(i = 0,1,2…n, n 为待优化超参个数),假设第 i 个超参的取值范围为$[a_i,b_i]$，则其参数值为：

$$x_i = \lfloor a_i + (b_i + 1 - a_i) * c_i \rfloor$$

由于 g_i 无法取值为 1，故参数取值区间为$[a_i,b_i+1)$。使用了浮点数编码的改进算法不仅能够支持连续交叉函数与突变函数，同时也能够支持对不同位置上的基因进行变化步骤（如跳跃基因步骤）。大大增强了算法的寻优自由度与寻优空间，使得算法能够迅速找到较为良好的超参数搭配。

3.2 改进交叉算子与突变算子

使用了浮点数编码使得算法支持连续交叉与突变，改进的交叉算子与突变算子不同于单点交叉与突变算子对整个基因值进行变化，而是改变基因部分值，这使得优良基因结构不会因为剧烈的变化而被破坏，增强了算法的局部收敛能力：

（1）交叉算子采用线性交叉，其操作过程为：假设种群中的 v1 和 v2 被选为交叉运算的父代，则由父代产生的子代如下：

$$\begin{cases} v_1' = \alpha v_1 + (1-\alpha)v_2 \\ v_2' = \alpha v_2 + (1-\alpha)v_1 \end{cases}$$

其中，α 为区间 [0 , 1]内的随机数，这种交叉是对父代需要交叉的基因进行部分值交换，而不是单点交叉那种全部值进行交换，保证了基因优良因子的传播，也避免了过于剧烈的变动对个体基因结构的破坏（即避免较优的解因某些维度的值被完全替换而消失）。

（2）突变算子采用非均匀突变。假设 S＝(v1 ,v2 ,……, vn)为待突变个体的基因，选择维度 vk 进行突变，而在该维度上，变量的定义域为[ak ， bk],则首先随机获得一个布尔值 sign，根据其结果进行如下突变操作：

$$v_k' = \begin{cases} v_k + \Delta v_k \text{ ,若 sign 为假} \\ v_k + \Delta v_k \text{ ,若 sign 为真} \end{cases}$$

其中，$\Delta v_k = y[r(1-t)]b$

其中，若 sign 为真，则 $y = b_k - v_k$,否则 $y = v_k - a_k$ ；t 为演变标签，设置为当前进化代数 g_c 与最大进化代数 g_m 的比值；r 为区间[0,1]内的随机数，b 为条件因子，可起到调节函数曲线非均匀变化的作用，一般设为 3。

3.3 跳跃基因遗传算法

跳跃基因遗传算法在经典遗传算法的基础上引入了跳跃基因思想并做出了一定改良，其在遗传算法中的选择操作之后加入基因跳跃操作，其基本步骤为：（1） 随机产生规模为 N 的初始染色体群 $\{X_1, X_2, \cdots, X_N\}$，其中，$X_i$ 由实数编码的待优化参数组成，染色体的长

度即为待优化参数个数 L_0;（2）计算每条染色体的适应度函数值，以评价参数解的目标函数值；（3）根据适应度大小进行选择操作；（4） 基因跳跃操作；（5）交叉和变异操作；（6）若满足终止条件，则停止进化，否则返回步骤（2）。

假设每条染色体上的转位子基因个数为 T，长度为 L，跳跃概率为 P∈［0，1］，以及跳跃方式的选择概率为 p ∈［0，1］，则转位子基因进行跳跃操作的主要步骤为：

（1） 随机产生整数序列 ai∈［0，L_0］，bi∈［0，N］和 c_i∈［0，L_0］（i＝1，2，…，T） ，分别表示转位子基因位置、跳跃的目标染色体序号和目标染色体内的基因插入位置；

（2） 置 i＝1；

（3）产生随机数 r_1∈［0，1］，若满足 P≥r_1，则发生跳跃行为，继续执行步骤（4），否则执行步骤（5）；

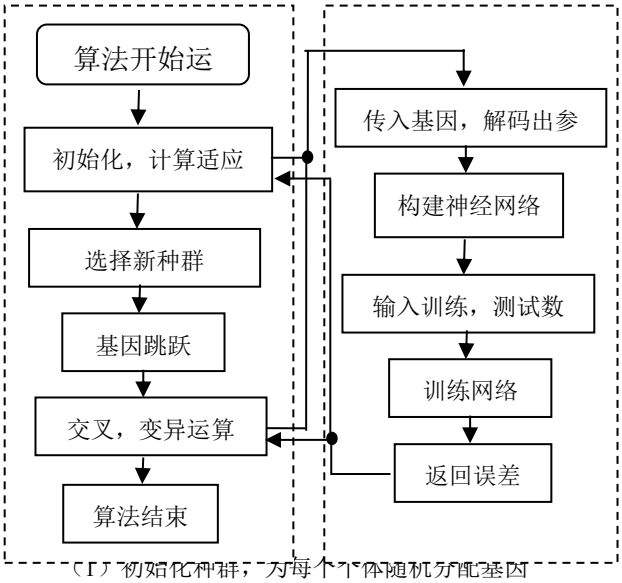
（4）产生随机数 $r_2 \in [0, 1]$，若满足 $p \geq r_2$，则按剪切和粘贴方式发生跳跃，否则按复制和粘贴方式发生跳跃；

（5）若 $i \geq T$ ，结束；否则 $i = i + 1$，返回步骤（3）。

此外，该算法还做了一些改进，引入了基因密度（根据该基因与其他基因的欧氏距离来参与计算）来对基因的适应度进行加权计算，使得基因密度越大的基因适应值有所减小。

4 FCJGGA-DNN

4.1 算法流程：



（1）初始化种群，为每个个体随机分配基因

（2）将基因解码出超参数，传入神经网络，构造模型并传入数据进行训练，返回损失值作为个体适应值

（3）根据个体适应度以及算法的选择算子进行选择，获得新种群。

（4）计算基因密度并对个体的适应度进行基因密度加权计算，随后进行基因跳跃操作

（5）根据交叉概率与交叉算子对新种群进行基因交叉运算。

（6）根据突变概率与突变算子对新种群进行基因突变运算。

（6）重新计算新种群适应度，将每个个体基因解码出超参数，传入神经网络，构造模型并传入数据进行训练，返回损失值作为个体适应值。

（7）更新种群历史最优基因。

（8）若算法未结束则返回（2），否则输出历史最优解，算法结束。

算法主要参数有：基因的维度（即待优化超参个数）n，种群规模，迭代次数，变异概率，交叉概率，基于密度权重以及基因跳跃概率权重。

4.2 网络模型评价指标：

本文选取均方误差（MSE）和平均绝对误差（Mean Absolute Error，MAE）两个指标联合评价神经网络模型。2 个评价指标的计算公式为：

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|\hat{y}_i - y_i|$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$$

其中：$\hat{y}_i$ 代表模型的预测值，y_i 代表期望值，n 代表测试集的数量。

模型训练时使用均方差 MSE 作为损失值，同时记录模型的平均绝对误差 MAE 参与后续评测。

5 实验与结果分析

本次实验使用 kaggle 上的鲍鱼数据集进行神经网络训练，通过九个特征预测鲍鱼年龄。数据按照 7：3 随机划分出训练集与测试集，最后通过几个算法的对比来验证 FCJGGA 算法的优化效果。

5.1 实验环境

为了方便多种算法的对比，本次实验多种算法在同一实验环境下运行多次。实验配置为：Intel® Core(TM) i7-11800H 处理器，NVIDIA GeForce RTX 3070 独立显卡，内存 16GB。软件配置为：window10, python3.8, torch1.12.0 + cu116, numpy1.24.3 库。

5.2 参数设置：

5.2.1 网络参数配置

神经网络隐藏层数为不参与优化的超参数，这里参考了算法在不同隐藏层数下运行的时长，将隐藏层数设置为 3，网络一共 5 层，即 1 层输入层，3 层隐藏层，以及 1 层输出层，每个隐藏层的神经元个数 $a_i(i = 1,2,3)$为待优化参数，由于目前神经元个数尚无理论指导，这里采用经验公式[6]确定：

$$r = \lfloor\sqrt{n + m}\ + d\rfloor$$

其中：r 为隐含层神经元个数，n 为输入层的神经元个数，m 为输出层的个数，d 通常取值为区间[0，10]内的整数，考虑到算法的运行时长，本次实验取 $d = 5$。

每个隐藏层的激活函数序号为 $b_i(i = 1,2,3)$，其取值为序号 0，1，2 中的一个整数，分别对应三个激活函数：Relu 函数，Sigmod 函数，Tanh 函数。

此外，本次实验还有两个超参需要优化，分别是每次梯度更新的样本数 batch 和学习率 learning-rate，分别记为 bh 和 lr，其取值都为整数序号 0,1,2,3,分别对应 batch 取值[256,512,1024,2048]与 learning-rate 取值[0.0001,0.001,0.01]。

以上八个超参数的取值范围如下：

表 1　超参数取值范围

Tab. 1 Value range of hyperparameters

超参数	取值范围
a_1	[2,2r]
a_2	[2,2r]
a_3	[2,2r]
b_1	[0,2]
b_2	[0,2]
b_3	[0,2]
bh	[0,3]
lr	[0,3]

实验中神经网络训练轮次固定为 1000，使用 MSE 作为损失函数，同时记录 MAE 来参与对比评测。

5.2.2　优化算法参数配置

本次实验使用三种算法参与对比分析：实数编码遗传算法优化神经网络参数（real coding genetic algorithm optimizes Deep Neural Network，RCGA-DNN），粒子群算法优化神经网络参数[7]（PSO optimizes Deep Neural Network, PSO-DNN）以及本文算法 FCJGGA-DNN。

为尽可能保证算法对比效果，实验中尽量给各算法配置相似参数，各算法参数配置如下：

RCGA-DNN:种群 20，迭代次数 30，交叉概率 0.8，变异概率 0.1。

PSO-DNN:种群 20，迭代次数 30，惯性权重 0.8~0.4，加速系数 1~0.5。

FCJGGA-DNN：种群 20，迭代次数 30，交叉概率 0.8，变异概率 0.1，基因跳跃概率权重 0.2，基因密度权重 10000。

5.3 实验结果分析

各个算法通过预设参数在数据集上运行，为了避免偶然因素的影响，各算法分别重复运行 3 次，根据测试结果进行分析。

5.3.1　算法预测结果分析

针对各算法在训练数据上的寻优结果，取寻优参数所得模型损失值 MSE 作对比，结果如表 2 所示，同时记录所得模型对应的 MAE，作对比表格如表 3 所示（表 2，3 所有取值精度均为 10^{-5}）：

表 2　各算法优化结果 MSE 对比

Tab. 2 MSE Comparison of Optimization Results by Algorithm

优化算法	最差值	最优值	平均值	标准差
RCGA	3.909209	3.868271	3.891017	0.013505
PSO	3.935953	3.859794	3.905740	0.026563
FCJGGA	3.916345	3.867171	3.889448	0.011291

表 3　各算法优化结果 MAE 对比

Tab. 3 Comparison of MAE of optimization results of each algorithm

优化算法	最差值	最优值	平均值	标准差
RCGA	1.463671	1.447666	1.456950	0.005336
PSO	1.468517	1.436072	1.630837	0.249170
FCJGGA	1.453753	1.438552	1.448421	0.004150

从表中便可以看出，与 RCGA 算法相比，本文算法平均方差 MSE 减少了 0.040%，平均绝对误差 MAE 减少了 0.585%，且 MSE 与 MAE 的标准差分别减少了 16.393%与 22.226%。与 PSO 算法相比，本文算法平均方差 MSE 减少了 0.417%，平均绝对误差 MAE 减少了 11.185%，且 MSE 与 MAE 的标准差分别减少了 57.493%与 98.334%。

从以上结果可以推知：整实数编码的 RCGA 算法优于采用了单点交叉与单点变异的算子，其对某个位置上的基因值整个进行突变，这种过于剧烈的变动可能会造成优良的基因结构被破坏，使得算法较难收敛，其结果精度较低，同时算法的波动性较大标准差也较大；PSO 算法通过惯性权重参与计算的速度来更新粒子位置，可能导致算法解变化较快，波动较大，同时收敛精度也不够高。

而与这两个算法相比，采用了浮点数编码的 FCJGGA 算法搜索可以使用改良的连续交叉与变异算子，这使得算法的收敛精度更高，同时算法收敛过程中不会因为剧烈的基因变动而导致波动过大，故而本文算法的标准差更小。算法在各方面的表现较为出色。

5.3.2　算法收敛过程分析

各算法收敛代数如下表所示：

表 4　各算法收敛代数对比

Tab. 4 Algebraic comparison of convergence of algorithms

优化算法	收敛代数	均方误差（MSE）/10^{-5}
RCGA-DNN	7	3.875750
PSO-DNN	26	3.890805
FCJGGA-DNN	27	3.871283

收敛曲线对比图如下所示：

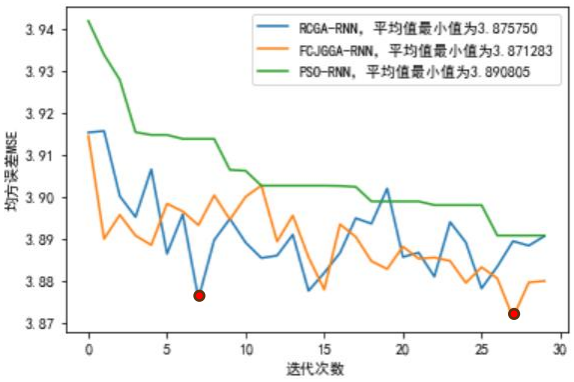


图 2 各算法迭代曲线

Figure 2 Iteration curves for each algorithm

从图中可以发现，两个遗传算法迭代过程中损失值波动较大，这时由于算法在执行过程中返回的是当前种群最优值，而不是历史最优（算法在执行结束后还是会返回历史最优解，因此算法实际上的收敛点为历史最优位置，图中已用红点标出），粒子群算法执行过程中返回的始终是全局最优，故算法呈现稳步下降的趋势。

从结果可以看出，FCJGGA 算法的收敛迭代次数与 RCGA 算法相比增加了 285.7%，损失值 MSE 减少了 0.115%，与 PSO 算法相比，收敛迭代次数增加了 3.704%,损失值 MSE 减少了 0.502%.

可见，本文算法收敛速度较慢，但收敛精度更高。此外，通过观察可以发现，RCGA 算法收敛迭代次数较早且波动性较大，考虑到本次各个算法实验只运行了 3 次，这里不排除偶然因素的影响。

6 结束语

本次实验针对神经网络参数优化问题提出了基于浮点数的改良跳跃基因优化算法，其使用的浮点数编码，使得遗传算法的交叉、变异算子不再受限于待优化参数是否为整数，可以使用改良的连续交叉与连续变异算子，而不是传统的单点交叉与变异算子，算法有了更大的搜索空间与搜索自由度，无论是全局寻优能力还是局部收敛能力都有了一定的提升。

此外，算法在浮点数编码时使用 0~1 比例来计算超参数的值，使得取值范围不同的超参数在算法计算时统一为 0~1 的取值区间，这使得在不同位置上的基因交流成为可能。算法使用的跳跃基因步骤加强了不同位置上基因的交流，同时还引入了基因密度对基因适应度进行加权计算，防止过多基因聚集在某一范围内，这都使得种群基因多样性大为提高，算法的全局寻优能力进一步增强，算法寻优结果优于其他寻优算法。从测试结果也可以看出，FCJGGA 算法无论是在误差方面还是稳定性，收敛速度方面均优于其他两种算法。

本次实验过程中，对于 FCJGGA 算法各个参数的调整为统一设定的默认值，优于算法的计算开销过大，并未详细进行参数调整，算法的寻优效率实际上还有较大的提升空间。同时，优于开销较大，算法运行次数并不是很大，因此无法充分排除偶然因素对实验结果的影响。此外，从收敛曲线可以看出，FCJGGA 算法的收敛速度较慢，这些问题都是今后实验的主要研究内容，也都将在今后的实验中去逐步解决。

参 考 文 献

[1] 孙永泽, 陆忠华. 基于超限学习机与随机响应面方法的深度学习超参数优化算法[J]. 高技术通讯 2019, 29(12): 1165-74

[2] Holland J H. Genetic algorithms. Scientific American, 1992: 44-50

[3] YANG Lu-gang, Research on Extreme Points Optimizing of Nonlinear Muti-peak Function Based on Genetic Algorithm, College of Electronics Engineering, Naval University of Engineering, Wuhan, China, Advanced Materials Research, vol 121-122 ,2010, pp.304-308.

[4] 浦黄忠, 甄子洋, 王道波, 刘媛媛, 用于多峰函数优化的改进跳跃基因遗传算法, 南京航空航天大学学报.

[5] 佘维，李阳，钟李红，孔德锋，田钊．基于改进实数编码遗传算法的神经网络超参数优化[J/OL]．计算机应用.
https://kns.cnki.net/kcms2/detail/51.1307.tp.20230605.1110.002.html

[6] 张宗腾, 张琳, 谢春燕, 等. 基于改进 GA-BP 神经网络的目标毁伤效果评估[J]. 火力与指挥控制 2021, 46(11): 43-8.

[7] J Kennedy, R C Eberhart. Particle swarm optimization. Om Proc. IEEE Int. Conf. Neural Networks, 1995(4): 1942-1948