神经网络超参数优化的删除垃圾神经元策略*

黄颖 顾长贵 杨会杰†

(上海理工大学管理学院, 上海 200093)

(2022 年 3 月 10 日收到; 2022 年 4 月 21 日收到修改稿)

随着深度学习处理问题的日益复杂, 神经网络的层数、神经元个数、和神经元之间的连接逐渐增加, 参数规模急剧膨胀, 优化超参数来提高神经网络的预测性能成为一个重要的任务. 文献中寻找最优参数的方法如灵敏度剪枝、网格搜索等, 算法复杂而且计算量庞大. 本文提出一种超参数优化的"删除垃圾神经元策略". 权重矩阵中权重均值小的神经元, 在预测中的贡献可以忽略, 称为垃圾神经元. 该策略就是通过删除这些垃圾神经元得到精简的网络结构, 来有效缩短计算时间, 同时提高预测准确率和模型泛化能力. 采用这一策略, 长短期记忆网络模型对几种典型混沌动力系统的预测性能得到显著改善.

关键词: LSTM, 混沌时间序列预测, 超参数优化, 删除垃圾神经元策略

PACS: 05.45.–a, 07.05.Mh, 05.45.Tp **DOI**: 10.7498/aps.71.20220436

1 引 言

深度学习被广泛应用于多学科领域, 极大地提高了人们对复杂系统的认识. 随着应用场景日益复杂, 数据量和系统特征量增多, 网络规模 (层数、神经元个数和神经元之间的连接) 日益膨胀, 算法复杂度和计算量也因此在指数增长, 通常用月或年来计算. 而大规模的参数, 也意味着过拟合问题, 从而降低习得的模型的泛化能力. 超参数优化, 也就是通过优化网络结构, 得到一个精简的网络结构, 在计算时间可接受的条件下, 显著提高预测水平, 达到性能最优, 成为当前人工智能领域一个基本而迫切需要解决的任务.

目前网络结构优化大致分为两种类型. 一是基于相关参数和评价指标的变化情况直观调整网络结构, 如拟合精度等指标的网格搜索法. 这类方法理论上要遍历所有参数, 耗时巨多, 远超当前计算能力. 实际中往往按照一定比例枚举参数取值, 这又极易跳过最佳参数. 二是基于一些高效的优化算法, 如贝叶斯优化[1]、灵敏度和相关性剪枝相结[2, 3]、学习率优化[4]、径向基函数优化[5]、多核极端学习机[6]、注意力机制引进[7]、扩展储量计算分化神经元[8]以及一些自适应算法[9]等, 来提升训练效率. 在广泛采用的灵敏度剪枝中, 轮流删除节点操作意味着庞大的计算量. 为避免大计算量而采用的工程近似方法, 易导致节点误删除[10].

本文提出神经网络超参数优化的"删除垃圾神经元策略". 这一策略依据的一个简单事实是, 神经元在时间序列预测中贡献是不一样的. 给定一个初始的网络结构, 并对它进行训练, 在权重矩阵中权重平均值小的神经元, 在预测中的贡献可忽略不计, 称为垃圾神经元. 从这一原始的神经网络中删除这些垃圾神经元, 简化网络结构, 来达到减小计算量、提高预测能力、增强泛化的目的. 采用这一策略对长短期记忆网络 (LSTM) 模型的网络结构进行了优化. LSTM 模型[11]作为循环神经网络 (RNN) 的改进, 既能记住短期信息, 又能记住长期信息,

* 国家自然科学基金 (批准号: 11875042, 11505114) 资助的课题.

† 通信作者. E-mail: hjyang@usst.edu.cn

克服了 RNN 模型在时间跨度过长时容易存在梯度爆炸或梯度消失的问题, 被广泛应用于时间序列分析, 如自然语言处理 (NLP)、语音识别、金融数据预测等. 对 Logistic, Henon, Rossler 三种典型混沌系统的预测表明, 这一策略可以有效改善 LSTM 的预测性能.

2 长短期记忆模型

如图 1(a) 所示 [12], LSTM 网络包含输入层、隐藏层和输出层, 每一层由多个单元组成. 在隐藏层的每个单元加入记忆细胞, 并通过输入门、遗忘门和输出门来控制状态.

记忆细胞与隐藏单元共同完成对历史信息的传输, 其内部结构如图 1(b) 所示. 其中, c 表示记忆细胞, h 表示隐藏单元, x 为输入信息, f 为遗忘门, i 为输入门, $\tilde{c}$ 为候选记忆细胞, o 为输出门, $\tan h$ 为激活函数, σ 为 sigmoid 激活函数, t 为时间步. 遗忘门 f_t、输入门 i_t 和输出门 o_t 均以当前时间步输入 x_t 与上一时间步隐藏单元信息 h_{t-1} 为输入, 以 σ 激活函数计算结果为输出, 阈值为 $[0,1]$; 候选记忆细胞 $\tilde{c}_t$ 的输入与前者相同, 但以 $\tan h$ 激活函数计算结果为输出, 阈值为 $[-1,1]$. 计算公式为

$$f_t = \sigma(x_t w_{xf} + h_{t-1} w_{hf} + b_f), \tag{1a}$$

$$i_t = \sigma(x_t w_{xi} + h_{t-1} w_{hi} + b_i), \tag{1b}$$

$$o_t = \sigma(x_t w_{xo} + h_{t-1} w_{ho} + b_o), \tag{1c}$$

$$\tilde{c}_t = \tanh(x_t w_{xc} + h_{t-1} w_{hc} + b_c), \tag{1d}$$

其中, $w_{xf}, w_{xi}, w_{xo}, w_{xc}$ 分别为 x_t 到 $f_t, i_t, o_t, \tilde{c}_t$ 的权重, $w_{hf}, w_{hi}, w_{ho}, w_{hc}$ 分别为 h_{t-1} 到 $f_t, i_t, o_t, \tilde{c}_t$ 的权重, b_f, b_i, b_o, b_c 分别为 $f_t, i_t, o_t, \tilde{c}_t$ 的偏置.

c_t 和 h_t 均由上一时间步记忆细胞 c_{t-1} 和当前时间步候选记忆细胞 $\tilde{c}_t$ 二者组合而成, 计算公式为

$$c_t = f_t \odot c_{t-1} + i_t \odot \tilde{c}_t, \tag{2a}$$

$$h_t = o_t \odot \tan h(c_t), \tag{2b}$$

其中, $\odot$ 为向量元素相乘.

f_t 控制 c_{t-1} 到当前时间步的信息流动, i_t 控制 $\tilde{c}_t$ 到当前时间步的信息流动, o_t 控制当前时间步的 c_t 到 h_t 的信息流动. 如当 f_t 接近 1 且 i_t 接近 0 时, 过去的记忆细胞信息将会一直保留, 可更好地捕捉时间序列中时间步较大的样本间的依赖关系. 当 o_t 接近 1 时, c_t 的信息将传递至 h_t 供输出层使用; 当 o_t 接近 0 时, c_t 的信息将自己保留.

3 删除垃圾神经元策略

作为实例, 我们考察了 LSTM 对混沌系统预测的能力. 采用删除垃圾神经元策略, 简化了隐藏层结构, 显著提高了预测能力. 删除垃圾神经元策略的具体操作步骤如下.

1) 搭建含有两层 LSTM 和两层全连接层的模型, 用网格搜索法寻找最佳参数的大致取值范围. 由于模型中参数众多, 若以等差为 1 的序列遍历, 模型训练耗时将长达数年, 因此以等比序列 2^n 或其他步长遍历参数.

2) 由于初始权重随机生成, 单次训练结果没有统计意义, 本文滑动读取样本分别进行多次训练, 并计算各组参数的预测准确率 (预测值的涨跌趋势和真实值涨跌趋势相同的样本量占总预测样本量的比值)、R 方值、MSE 等评价指标的平均值

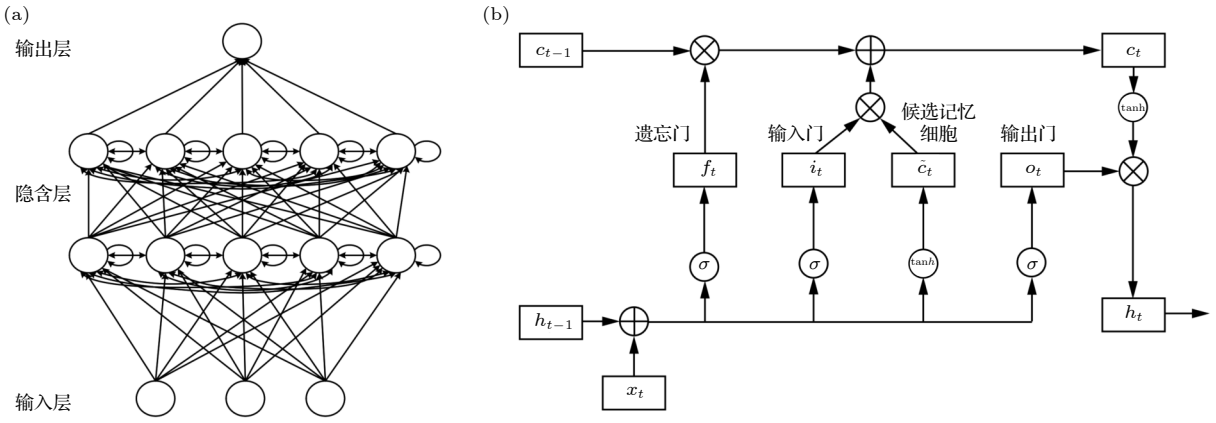

图 1 LSTM 神经网络 (a) LSTM 模型网络结构; (b) 单元内部运行逻辑

Fig. 1. LSTM neural network: (a) network structure of LSTM; (b) run logic inside the cell.

并输出. 根据准确率和 R 方值最高、MSE 最低、神经元数最少的原则, 初步选取最佳参数组合, 包括各层神经元数、迭代次数、batch、dropout 等.

3) 用初步选取的最佳参数组合训练模型并输出网络权重, 分析各组权重代表的意义, 明确垃圾神经元. 删除垃圾神经元得到简洁的网络结构, 比较模型效果.

4) 以不同的权重阈值为界, 定义垃圾神经元, 尽量使每组阈值删除的神经元数分布均匀; 比较精简网络后的预测效果, 找到能最大程度提升模型性能的阈值.

4 计算实验

具体考察 Logistic[13], Henon[14] 和 Rossler[15] 三个典型混沌系统的 LSTM 预测. Takens 嵌入定理指出, 混沌系统的每一维度变量都包含整个系统的长期演化信息[16]. 因此, 我们从每个系统的动力学轨迹中, 只抽取一维数据作为样本, 以使得各系统实验结果之间具有可比性.

4.1 数据处理

Logistic 模型也称虫口模型, 其差分方程表示为

$$x_{n+1}=\mu x_n(1-x_n). \tag{3}$$

随着参数 μ 的增加, 系统发生倍周期分叉, 当 $\mu\in[3.569,4]$, 系统出现混沌现象. 由于 Logistic 系统随着参数取值不同, 混沌程度也不同, 因此 μ 分别取 3.6, 3.7, 3.8, 3.9, 3.99 生成样本量为 50000 的一维时间序列数据. 图 2(a) 给出的是 μ 取 3.9 时的轨迹.

Henon 映射的迭代表达式为

$$x_{n+1}=1-ax_n^2+y_n, \tag{4a}$$

$$y_{n+1}=bx_n, \tag{4b}$$

其中 $a=1.4$, $b=0.3$. x 和 y 初始值取为 0.01 生成样本量为 50000 的时间序列数据. 图 2(b) 中横轴表示系统的迭代次数, 图像给出迭代前 100 次得到的 x, y 值构成的轨迹.

Rossler 系统是一连续混沌动力系统, 其微分方程组为

$$\mathrm{d}x/\mathrm{d}t=-(x+y), \tag{5a}$$

$$\mathrm{d}y/\mathrm{d}t=x+ay, \tag{5b}$$

$$\mathrm{d}z/\mathrm{d}t=b+z(x-c), \tag{5c}$$

其中参数 a 取 0.2, b 取 0.4, c 取 5.7. 以 $x=y=z=0$ 为初始值, 采用四阶 Runge-Kutta 方法[17], 以 0.001 为步长模拟出 $t\in[0,500]$ 的运动轨迹, 图 2(c) 给出前 10000 个时间点的轨迹. 为使样本量同另外两个系统一致从而结果具有可比性, 本文再以 10 为抽样步长得到 50000 条数据作为样本.

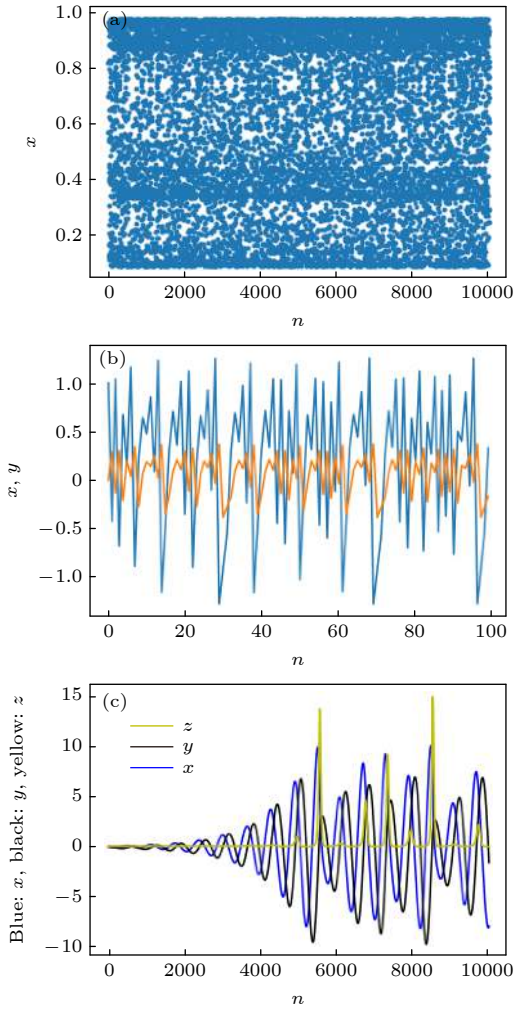

图 2 混沌时间序列 (a) Logistic 系统, $\mu=3.9$; (b) Henon 系统; (c) Rossler 系统

Fig. 2. Chaotic time series: (a) Logistic system, $\mu=3.9$; (b) Henon system; (c) Rossler system.

对数据进行归一化处理. 三个系统的训练集样本量均取为 5000. 由于 LSTM 模型预测较长时间后的数据意义不大, 因此选取测试集样本量为 15. 为使实验结果具有统计意义, 以一定步长滑动选取样本进行多次实验. 本文分别以 100, 200, 300, ···, 4000 作为滑动窗口训练 10 次模型, 取 10 次预测准确率的均值为最终结果, 发现不同滑动窗口对应的平均预测准确率在 70% 附近上下波动, 可见滑动窗口大小对预测结果没有显著影响. 为充分利用

并均匀覆盖已有样本, 选取 4000 为滑动窗口进行预测, 如第一批样本以 1—5000 条数据作为训练集, 5001—5015 条数据作为测试集; 第二批样本以 4000—9000 条数据作为训练集, 9001—9015 条数据作为测试集; 以此类推, 共训练 10 批样本 (最后一批样本以 40000—45000 条数据作为训练集, 45001—45015 条数据作为测试集). 最终以 10 次预测评价指标的均值作为模型最终的评价指标.

4.2 LSTM 模型建立

三个系统初步选取的最佳参数组合如表 1 所列. 其中, train 为训练集样本量; test 为测试集样本量; win 为滑动窗口数, 表示每次观测到的样本数, 如 win 为 3 表示第一批输入模型的样本为 x_1, x_2, x_3, 下一批为 x_2, x_3, x_4, 以此类推; L_1 为第一层 LSTM 输出神经元数; L_2 为第二层 LSTM 输出神经元数; D_1 为第一层全连接层输出神经元数; D_2 为第二层全连接层输出神经元数, 即最终输出.

表 1 模型参数及结果

Table 1. Parameters and results of the models.

模型	train	test	win	L_1	L_2	D_1	D_2	准确率
Logistic ($\mu = 3.6$)	5000	15	22	16	4	4	1	82.90%
Logistic ($\mu = 3.7$)	5000	15	22	16	4	4	1	70.70%
Logistic ($\mu = 3.8$)	5000	15	2	20	4	4	1	68.60%
Logistic ($\mu = 3.9$)	5000	15	2	16	4	2	1	60.00%
Logistic ($\mu = 3.99$)	5000	15	2	16	4	4	1	57.10%
Henon	5000	15	2	22	2	2	1	67.10%
Rossler	5000	15	2	16	4	4	1	77.10%

在多数模型中, 以较为常见的步长 2^n 进行网格搜索得到的预测准确率是相对较高的, 第一层网络的最优参数基本稳定在 16 个神经元, 对于训练集样本数为 5000 的数量级来说是足够的, 更复杂的网络容易造成过拟合; 当然也有部分模型以 2^n 为步长网格搜索时未能得到不错的预测效果, 文中也会视情况选择其他步长, 如$\mu = 3.8$的 Logistic 模型, 以 10 为步长进行网格搜索可得到更好的预测效果, 此时便在网格搜索最优参数为 20 的基础上进一步优化超参数.

4.3 权重分析

全连接层的权重较为简单, 在此不做分析. LSTM 层的权重包含三个张量: kernel, recurrent_kernel 和 bias, 每个张量的维数为 4×神经元数, 依次为input_gate, forget_gate, cell 和output_gate, 权重拆分如表 2 所列.

表 2 权重结构拆分

Table 2. Weight structure resolution.

结构	起始点	终点
input_gate	0	units
forget_gate	units	2×units
cell	2×units	3×units
output_gate	3×units	4×units

根据 (1a) 式—(1d) 式可知, output_gate 权重直接关系到神经元的最终输出结果, 因此对该权重做热度图以便于分析. 以 Logistic 系统中 $\mu = 3.99$ 为例, LSTM 输出层神经元数为 16, 输出门对应的权重矩阵维数为 16×16. 如表 3 所列, 第一行权重表示隐藏层输入的 16 个神经元对输出的第一个神经元影响大小, 第一列则表示隐藏层输入的第一个神经元对输出的 16 个神经元的影响大小, 均值行为每个输入神经元对所有输出神经元的权重绝对值的平均值, 均值越小的神经元可以理解为对整个输出层的影响越小.

在 python 中用 imshow 函数绘制出该权重矩阵对应的热图. 首先对表 3 中每一行的权重数据分别进行归一化, 便于分析隐藏层输入神经元对各个输出神经元的影响大小; 接着绘制热图, 颜色越黄权重越高, 颜色越蓝权重越低. 为了更直观地体现各输入神经元的重要性, 进一步绘制如图 3 和图 4 所示的热图. 同样地, 颜色越黄权重均值越高, 表

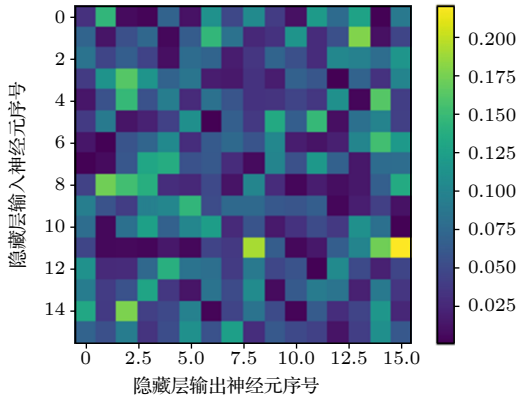

图 3 隐藏层输入神经元对输出神经元的权重热图

Fig. 3. Heat map of weight of input neuron to output neuron in hidden layer.

表 3 输出门权重矩阵图

Table 3. Heat diagram of output door's weights.

	w1	w2	w3	w4	w5	w6	w7	w8	w9	w10	w11	w12	w13	w14	w15	w16
x1	0.032129	0.144328	0.007293	0.003557	0.070465	0.011652	0.11142	0.048287	0.105727	0.039649	0.010162	0.119949	0.0771	0.126937	0.001947	0.089396
x2	0.070867	0.013232	0.054503	0.073496	0.005734	0.064071	0.145835	0.083217	0.025553	0.033065	0.113584	0.028327	0.054085	0.177761	0.010123	0.046546
x3	0.083647	0.046366	0.066226	0.044534	0.010054	0.077919	0.071001	0.018548	0.035294	0.071642	0.053888	0.028448	0.102273	0.096744	0.078535	0.114879
x4	0.039123	0.113002	0.161581	0.113866	0.070651	0.085571	0.01836	0.015408	0.032978	0.018375	0.068342	0.059137	0.00701	0.070451	0.032602	0.099853
x5	0.013762	0.055823	0.147481	0.055493	0.093444	0.027264	0.082384	0.058674	0.032903	0.033342	0.045937	0.035937	0.110378	0.004487	0.161653	0.041037
x6	0.038079	0.089649	0.013102	0.021696	0.042833	0.109787	0.001024	0.0673	0.036916	0.134038	0.066291	0.146953	0.009803	0.081372	0.098701	0.042456
x7	0.013865	0.001702	0.057869	0.072264	0.104456	0.029761	0.062669	0.07539	0.05715	0.102282	0.017876	0.012626	0.020022	0.100243	0.153076	0.11875
x8	0.002042	0.006925	0.060008	0.13031	0.136406	0.056203	0.061606	0.028064	0.006926	0.099129	0.055122	0.117276	0.06846	0.014505	0.078184	0.078834
x9	0.044205	0.171724	0.153162	0.13818	0.029189	0.025947	0.049391	0.012338	0.100584	0.028133	0.004946	0.017914	0.008463	0.014741	0.066944	0.134139
x10	0.094619	0.0563	0.040223	0.096283	0.10152	0.145036	0.051991	0.075623	0.075216	0.061209	0.057986	0.066076	0.004787	0.01945	0.042341	0.011339
x11	0.078909	0.005603	0.08149	0.125202	0.069081	0.10143	0.122451	0.027058	0.057647	0.016226	0.03275	0.050667	0.036795	0.11072	0.081767	0.002204
x12	0.046159	0.005688	0.006237	0.004618	0.014815	0.005272	0.04598	0.037005	0.190933	0.065535	0.005131	0.015155	0.065812	0.099804	0.172294	0.21956
x13	0.111226	0.027026	0.027497	0.074868	0.139154	0.084413	0.080342	0.038769	0.088824	0.047083	0.056548	0.002081	0.10549	0.049929	0.020529	0.04622
x14	0.092772	0.03999	0.055938	0.128114	0.036386	0.013061	0.083943	0.051033	0.106374	0.007257	0.063049	0.091929	0.084821	0.020458	0.089496	0.035379
x15	0.131586	0.038161	0.176303	0.041758	0.049173	0.096633	0.0033	0.045529	0.084262	0.050839	0.003322	0.063406	0.029601	0.045323	0.116047	0.024757
x16	0.070289	0.05378	0.092472	0.03372	0.052087	0.110236	0.055639	0.124221	0.029371	0.06142	0.04904	0.043376	0.012261	0.041226	0.109564	0.061299
均值	0.060205	0.054331	0.075087	0.072372	0.064091	0.065266	0.065459	0.050404	0.066666	0.054326	0.043998	0.056204	0.049428	0.067134	0.082113	0.072915

示该输入神经元给整个输出层提供的信息越多; 颜色越蓝权重均值越低, 表示该输入神经元给整个输出层提供的信息越少, 均值足够低的输入神经元即可作为垃圾神经元删除.

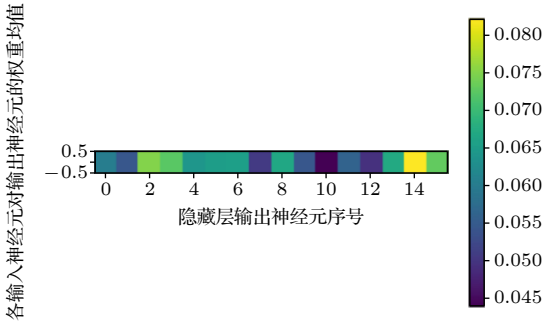

图 4 隐藏层各输入神经元对输出神经元的权重均值热图

Fig. 4. Heat map of weights' mean value of input neurons to output neurons in hidden layer.

输出神经元的计算公式

$$\begin{cases} x_1 = h_{1,1}w_{1,1} + h_{1,2}w_{1,2} + \cdots + h_{1,16}w_{1,16} \\ x_2 = h_{2,1}w_{2,1} + h_{2,2}w_{2,2} + \cdots + h_{2,16}w_{2,16} \\ \cdots \\ x_{16} = h_{16,1}w_{16,1} + h_{16,2}w_{16,2} + \cdots + h_{16,16}w_{16,16} \end{cases} \tag{6}$$

在以 2^n 为参数序列进行网格搜索 [18–20] 时, 与 16 个相邻的神经元参数为 8, 在该参数区间内, 以不同阈值删减神经元并观察模型预测结果变化.

图 5 为 16 个输入神经元的权重均值折线图, 如 $w11$ 列的权重均值在均值行中的颜色最浅, 则该神经元对输出值的影响最小, $w15$ 列的权重均值在均值行中的颜色最深, 则该神经元对输出值的影响最大.

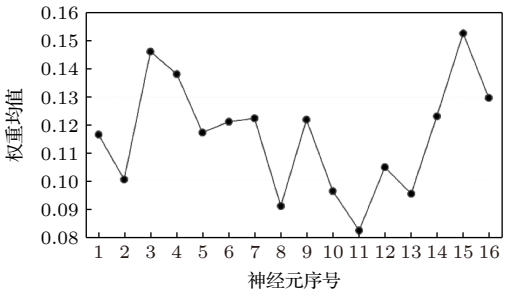

图 5 权重均值折线图

Fig. 5. Line graph of the weights' mean.

表 4 给出以权重均值低于 0.09, 0.1 和 0.11 为阈值删除垃圾神经元以及神经元数调整前后的预测准确率, 通过迷你趋势图观察以不同阈值删除垃圾神经元对模型预测性能的影响. 通过网格搜索得到神经元数为 16 时模型预测准确率为 57.1%; 权重均值低于 0.09 的神经元有 1 个, 删除后神经元数降为 15, 预测准确率为 59.3%, 比初始结果提升 2.2 个百分点; 权重均值低于 0.1 的神经元有 4 个, 删除后神经元数降为 12, 预测准确率为 56.4%,

比初始结果降低 0.7 个百分点; 权重均值低于 0.11 的神经元有 6 个, 删除后神经元数降为 10, 预测准确率为 51.4%, 与初始降低 5.7 个百分点. 从预测准确率的迷你趋势图来看, 删除 1 个权重均值低于 0.09 的垃圾神经元可提升预测效果, 减少了部分过拟合; 随着删减数目的增多, 模型欠拟合, 预测准确率逐渐低于原始水平. 因此, 基于权重分析适当删减垃圾神经元的方法能有效提升模型性能.

表 4 $\mu = 3.99$ 时不同参数的预测准确率

Table 4. The prediction accuracy of different parameters when $\mu = 3.99$.

指标	调整前	以各权重阈值调整后			变化趋势
		0.09	0.1	0.11	
L_1	16	15	12	10	
准确率	57.10%	59.30%	56.40%	51.40%	

5 结果分析

三个系统在不同参数取值下, 以相同策略选取删减垃圾神经元的权重阈值以提升最终的预测准确率; 删除垃圾神经元的数量及调整网络结构前后的预测准确率具体结果如表 5 所列.

表 5 神经元数及预测准确率变化表

Table 5. Table of neuron numbers and prediction accuracy.

模型	调整前 L_1	调整后 L_1	调整前准确率	调整后准确率	神经元数调整	准确率变化趋势
Logistic ($\mu = 3.6$)	16	15	82.90%	90.70%	–1	
Logistic ($\mu = 3.7$)	16	13	70.70%	71.40%	–3	
Logistic ($\mu = 3.8$)	20	16	68.60%	68.60%	–4	
Logistic ($\mu = 3.9$)	16	12	60.00%	60.00%	–4	
Logistic ($\mu = 3.99$)	16	15	57.10%	59.30%	–1	
Henon	22	21	67.10%	70.00%	–1	
Rossler	16	14	77.10%	83.60%	–2	

在 Logistic 系统中, $\mu = 3.6$时的结果如表 6 所列, 网格搜索得到神经元数为 16, 模型预测准确率为 82.9%. 为使神经元数位于区间 (8, 16) 中, 分别以权重均值低于 0.08, 0.09, 0.095 为阈值删减神经元并观察模型效果. 删除权重均值低于 0.08 的 1 个神经元, 预测准确率为 90.7%, 比初始预测结果提升 7.8%; 删除权重均值低于 0.09 的 4 个神经元, 预测准确率为 87.9%, 比初始结果提升 5%; 删除权重均值低于 0.095 的 6 个神经元, 预测准确率为 78.6%, 比初始结果降低 4.3%, 效果更差. 从迷你趋势图来看, 删除 1 个权重均值在 0.08 以下的垃圾神经元可最大程度提升预测效果; 随着删减数目增多, 运行成本持续降低, 预测准确率逐渐回落, 直至模型欠拟合使预测效果低于原始水平.

表 6 $\mu = 3.6$ 时不同参数的预测准确率

Table 6. The prediction accuracy of different parameters when $\mu = 3.6$.

指标	调整前	以各权重阈值调整后			变化趋势
		0.08	0.09	0.095	
L_1	16	15	12	10	
准确率	82.90%	90.70%	87.90%	78.60%	

$\mu = 3.7$ 时的结果如表 7 所列, 网格搜索得到神经元数为 16, 模型预测准确率为 70.7%. 为使神经元数位于区间 (8, 16) 中, 分别以权重均值低于 0.075, 0.09, 0.105 为阈值删减神经元并观察模型效果. 删除权重均值低于 0.08 的 3 个神经元, 预测准确率为 71.4%, 比初始预测结果提升 0.7%; 删除权重均值低于 0.095 的 5 个神经元, 准确率为 65%, 比初始结果降低 5.7%; 删除权重均值低于 0.105 的 7 个神经元, 预测准确率为 60.7%, 比初始结果降低 10%, 效果更差. 从迷你趋势图来看, 删除 3 个权重均值在 0.075 以下的垃圾神经元可最大程度提升预测效果; 随着删减数目的增多, 模型欠拟合导致预测效果愈发低于原始水平.

表 7 $\mu = 3.7$ 时不同参数的预测准确率

Table 7. The prediction accuracy of different parameters when $\mu = 3.7$.

指标	调整前	以各权重阈值调整后			变化趋势
		0.075	0.09	0.105	
L_1	16	13	11	9	
准确率	70.70%	71.40%	65.00%	60.70%	

$\mu = 3.8$时的结果如表 8 所列, 网格搜索得到神经元数为 20, 模型预测准确率为 68.6%. 为使神经元数位于区间 (10, 20) 中, 分别以权重均值低于 0.08, 0.09, 0.1 为阈值删减神经元并观察模型效果. 删除权重均值低于 0.08 的 2 个神经元和权重均值

低于 0.095 的 5 个神经元, 预测准确率均为 68.6%, 均与初始结果持平, 但删减数量越多, 运行成本会相对越低; 权重均值低于 0.105 的神经元有 8 个, 删除后神经元数降为 12, 预测准确率为 65%, 比初始结果降低 3.6%. 从迷你趋势图来看, 删除 4 个权重均值在 0.09 以下的垃圾神经元可在预测准确率不降低的前提下节省最多的运行成本; 继续删减便会导致模型欠拟合, 使得预测效果低于原始水平.

表 8 $\mu = 3.8$ 时不同参数的预测准确率

Table 8. The prediction accuracy of different parameters when $\mu = 3.8$.

指标	调整前	以各权重阈值调整后			变化趋势
		0.085	0.095	0.105	
L_1	20	18	16	12	
准确率	68.60%	68.60%	68.60%	65.00%	

$\mu = 3.9$时的结果如表 9 所列, 网格搜索得到神经元数为 16, 模型预测准确率为 60%. 为使神经元数位于区间 (8, 16) 中, 分别以权重均值低于 0.09, 0.095, 0.1 为阈值删减神经元并观察模型效果. 删除权重均值低于 0.09 的 2 个神经元和权重均值低于 0.095 的 4 个神经元, 预测准确率均为 60%, 均与初始结果持平, 但删减数量越多, 运行成本会相对越低; 权重均值低于 0.105 的神经元有 6 个, 删除后神经元数降为 10, 预测准确率为 55%, 比初始结果降低 5%. 从迷你趋势图来看, 删除 4 个权重均值在 0.095 以下的垃圾神经元可在预测准确率不降低的前提下节省最多的运行成本; 继续删减便会导致模型欠拟合, 使得预测效果低于原始水平.

表 9 $\mu = 3.9$ 时不同参数的预测准确率

Table 9. The prediction accuracy of different parameters when $\mu = 3.9$.

指标	调整前	以各权重阈值调整后			变化趋势
		0.09	0.095	0.1	
L_1	16	14	12	10	
准确率	60.00%	60.00%	60.00%	55.00%	

图 6 给出参数 μ 分别取值 3.6, 3.7, 3.8, 3.9, 3.99 时选择最优权重阈值的变化. μ 值越小混沌程度越弱, μ 值越大混沌程度越强, 可见随着混沌程度的提升, 最优权重阈值整体呈上升趋势; 系统越混沌, 需要保留的神经元权重越高. 图 7 给出参数 μ 分别取值 3.6, 3.7, 3.8, 3.9, 3.99 时优化超参数前后的模型预测准确率变化. 由于参数越大, 模型的混沌程度越高, 当 μ 接近 4 时, x 取值越近似于在 0—1 之间随机分布, 预测难度更大. 因此整体来看, 无论超参数调整前还是调整后的预测准确率都呈下降趋势; 从调整超参数前后的预测准确率变化幅度来看, 剔除垃圾神经元对模型性能的提升效果是逐步降低的. 但最差的效果也就是预测准确率与原先持平, 而运行成本却大大降低了, 说明该方法可准确定位到对模型预测没有贡献的垃圾神经元, 在不影响训练效果的情况下, 最大程度降低模型运行负担、提升模型性能, 这在参数众多、耗时较长的深度学习模型训练中是有重要意义的.

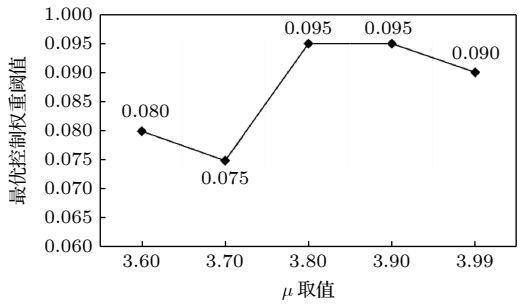

图 6 不同混沌状态对应的最优权重阈值变化

Fig. 6. The change of optimal weight threshold corresponding to different chaotic states.

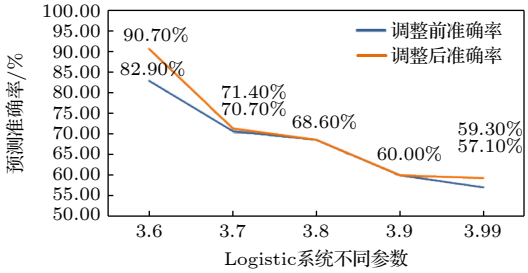

图 7 不同混沌状态对应的预测准确率变化

Fig. 7. The change of prediction accuracy of different chaotic states.

为了进一步说明该方法的可行性, 本文用 Henon 系统和 Rossler 系统的一维独立变量分别再次进行实证研究, 结果如表 10 和表 11 所列.

在 Henon 系统一维独立数据模型中, 网格搜索得到神经元数为 22, 模型预测准确率为 67.1%. 为使神经元数位于区间 (14, 22) 中, 分别以权重均值低于 0.1, 0.11, 0.12, 0.14 为阈值删减神经元并观察模型效果. 如表 10 所列, 删除权重均值低于 0.1 的 1 个神经元后, 预测准确率为 70%, 比初始

结果提升 2.9%; 删除权重均值低于 0.11 的 3 个神经元后, 预测准确率为 66.4%, 比初始结果降低 0.7%; 删除权重均值低于 0.12 的 6 个神经元后, 预测准确率为 65.7%, 比初始结果降低 1.4%; 删除权重均值低于 0.14 的 8 个神经元后, 预测准确率为 65%, 比初始结果降低 2.1%. 从迷你趋势图来看, 删除 1 个权重均值在 0.1 以下的垃圾神经元可最大程度提升预测效果; 更多的删减则导致模型欠拟合, 预测效果越来越差.

表 10 Henon 系统取不同参数的预测准确率

Table 10. Prediction accuracy of Henon system for different parameters.

指标	调整前	以各权重阈值调整后				变化趋势
		0.1	0.11	0.12	0.14	
L_1	22	21	19	16	14	
准确率	67.10%	70.00%	66.40%	65.70%	65.00%	

表 11 Rossler 系统取不同参数的预测准确率

Table 11. Prediction accuracy of Rossler system for different parameters.

指标	调整前	以各权重阈值调整后				变化趋势
		0.085	0.095	0.105	0.115	
L_1	16	14	12	11	8	
准确率	77.10%	83.60%	81.40%	80.70%	71.40%	

在 Rossler 系统一维独立数据模型中, 网格搜索得到神经元数为 16, 模型预测准确率为 77.1%. 为使神经元数位于区间 (8, 16) 中, 分别以权重均值低于 0.085, 0.095, 0.105, 0.115 为阈值删减神经元并观察模型效果. 如表 11 所列, 删除权重均值低于 0.085 的 2 个神经元后, 预测准确率为 83.6%, 比初始结果提升 6.5%; 删除权重均值低于 0.095 的 4 个神经元后, 预测准确率为 81.4%, 比初始结果提升 4.3%; 删除权重均值低于 0.105 的 5 个神经元后, 预测准确率为 80.7%, 比初始结果提升 3.6%; 删除权重均值低于 0.115 的 8 个神经元后, 预测准确率为 71.4%, 比初始结果降低 5.7%. 从迷你趋势图来看, 删除 1 个权重均值在 0.085 以下的垃圾神经元预测效果最好; 以低于 0.095 和 0.105 为阈值删减神经元带来的预测准确率提升相对前者越来越低, 但仍然高于初始水平, 且运行成本越来越低; 继续删减更多的神经元, 模型欠拟合导致预测效果越来越差, 低于初始水平.

以上所有模型均采用滑动选取样本的方法, 进行了 10 次以上的训练并对最终的评价指标取平均值, 故结果具有统计意义, 并非偶然现象. 观察这些系统在删减不同阈值下垃圾神经元时的预测效果变化, 发现大多情况下, 使预测效果提升最大的阈值不超过 0.1, 即通常将权重均值不超过 0.1(甚至更低) 的神经元视为垃圾神经元并删除可较大提升模型性能. 通过多次实验, 充分证明所提方法的可行性与有效性.

6 结论与展望

针对现有的超参数优化方法存在的过拟合、计算量庞大等问题, 本文提出了通过分析权重含义定位冗余神经元, 从而快速高效地精简网络结构、降低运行成本、提升训练效果的方法.

由于深度学习模型参数众多, 挨个遍历所有参数动辄耗时长达数年, 运行成本极高; 否则又会跃过最佳参数, 达不到最好的训练效果, 即使达到最好的训练效果, 也可能因为模型结构不够精简而导致过拟合, 冗余神经元也会大大降低运行效率. 通过 Logistic 模型、Henon 模型、Rossler 模型的实证分析, 证明此方法可大大提高寻找最优参数的效率, 减少模型中冗余的神经元, 从而避免过拟合、提高泛化能力. 在模型预测准确率不受影响的前提下, 有效缩短计算时间, 提高运行效率, 甚至可以通过减少过拟合使准确率得到提升.

混沌时间序列为看似无序的有序系统, 以此进行实证研究可证明方法本身的有效性. 但在实际数据中往往存在许多白噪声, 因此, 如何降噪并有效应用于实际数据成为我们下一步继续研究的问题.

参考文献

[1] Deng S 2019 *Appl. Res. Comput.* **36** 1984 (in Chinese) [邓帅 2019 计算机应用研究 **36** 1984]

[2] Shao E Z, Wu Z Y, Wang C 2020 *Ind. Contrl. Comput.* **33** 11 (in Chinese) [邵恩泽, 吴正勇, 王灿 2020 工业控制计算机 **33** 11]

[3] Qiao J F, Fan R Y, Han H G, Ruan X G 2010 *Contl. Theor. Appl.* **27** 111 (in Chinese) [乔俊飞, 樊瑞元, 韩红桂, 阮晓钢 2010 控制理论与应用 **27** 111]

[4] Chen G M, Yu T T, Liu X W 2021 *J. Num. Method. Comp. Appl.* **42** 215 (in Chinese) [陈国茗, 于腾腾, 刘新为 2021 数值计算与计算机应用 **42** 215]

[5] Wei D Z, Chen F J, Zheng X X 2015 *Acta Phys. Sin.* **64** 110503 (in Chinese) [魏德志, 陈福集, 郑小雪 2015 物理学报 **64** 110503]

[6] Wang X Y, Han M 2015 *Acta Phys. Sin* **64** 070504 (in

Chinese) [王新迎, 韩敏 2015 物理学报 **64** 070504]
[7] Huang W J, Li Y T, Huang Y 2021 *Acta Phys. Sin.* **70** 010501 (in Chinese) [黄伟建, 李永涛, 黄远 2021 物理学报 **70** 010501]
[8] Yamaguti Y, Tsuda I 2021 *Chaos* **31** 013137
[9] Graves A 2013 arXiv: 1308.0850 [cs. NE]
[10] Johnston D E 1978 *Proc 8 th BHRA Int Conf Fluid Sealing* Durham, UK, 1978 pC1-1
[11] Sezer O B, Gudelek M U, Ozbayoglu A M 2020 *Appl. Soft Comput. J.* **90** 106181
[12] Gan W J, Chen Y H, Han J, Wang Y F 2020 *Comput. Syst. Appl.* **29** 212 (in Chinese) [甘文娟, 陈永红, 韩静, 王亚飞 2020 计算机系统应用 **29** 212]
[13] Farmelo G 2002 *It Must Be Beautiful: Great Equations of Modern Science* (London: Granta Publications) pp28–45
[14] Grassberger P, Procaccia I 1983 *Physica D* **9** 189
[15] Nauenberg M 1983 *Ann. N. Y. Acad. Sci.* **410** 317
[16] Zhang Z H, Ding H F 2009 *Comput. Technol. Dev.* **19** 185 (in Chinese) [张中华, 丁华福 2009 计算机技术与发展 **19** 185]
[17] Butcher J C 1967 *J. ACM* **14** 84
[18] Liu C, Yin S Q, Zhang M, Zeng Y, Liu J Y 2014 *Appl. Mech. Mater.* **644-650** 2216
[19] Bao Y K, Liu Z T 2006 *LNCS* **4224** 504
[20] Ou Y Y, Chen G H, Oyang Y J 2006 *LNCS* **4099** 1017

Junk-neuron-deletion strategy for hyperparameter optimization of neural networks*

Huang Ying Gu Chang-Gui Yang Hui-Jie †

(*Business School, University of Shanghai for Science and Technology, Shanghai* 200093, *China*)

(Received 10 March 2022; revised manuscript received 21 April 2022)

Abstract

With the complexity of problems in reality increasing, the sizes of deep learning neural networks, including the number of layers, neurons, and connections, are increasing in an explosive way. Optimizing hyperparameters to improve the prediction performance of neural networks has become an important task. In literatures, the methods of finding optimal parameters, such as sensitivity pruning and grid search, are complicated and cost a large amount of computation time. In this paper, a hyperparameter optimization strategy called junk neuron deletion is proposed. A neuron with small mean weight in the weight matrix can be ignored in the prediction, and is defined subsequently as a junk neuron. This strategy is to obtain a simplified network structure by deleting the junk neurons, to effectively shorten the computation time and improve the prediction accuracy and model the generalization capability. The LSTM model is used to train the time series data generated by Logistic, Henon and Rossler dynamical systems, and the relatively optimal parameter combination is obtained by grid search with a certain step length. The partial weight matrix that can influence the model output is extracted under this parameter combination, and the neurons with smaller mean weights are eliminated with different thresholds. It is found that using the weighted mean value of 0.1 as the threshold, the identification and deletion of junk neurons can significantly improve the prediction efficiency. Increasing the threshold accuracy will gradually fall back to the initial level, but with the same prediction effect, more operating costs will be saved. Further reduction will result in prediction ability lower than the initial level due to lack of fitting. Using this strategy, the prediction performance of LSTM model for several typical chaotic dynamical systems is improved significantly.

Keywords: LSTM, chaotic time series prediction, hyperparameter optimization, junk neuron deletion strategy

PACS: 05.45.–a, 07.05.Mh, 05.45.Tp **DOI:** 10.7498/aps.71.20220436

* Project supported by the National Natural Science Foundation of China (Grant Nos. 11875042, 11505114).

† Corresponding author. E-mail: hjyang@usst.edu.cn

物理学报 Acta Physica Sinica
Chinese Physical Society
Institute of Physics, CAS

神经网络超参数优化的删除垃圾神经元策略

黄颖　顾长贵　杨会杰

Junk-neuron-deletion strategy for hyperparameter optimization of neural networks

Huang Ying　Gu Chang-Gui　Yang Hui-Jie

引用信息 Citation: Acta Physica Sinica, 71, 160501 (2022)　DOI: 10.7498/aps.71.20220436
在线阅读 View online: https://doi.org/10.7498/aps.71.20220436
当期内容 View table of contents: http://wulixb.iphy.ac.cn
