Report On Coding Assignment 1

Welding Experiment with Artificial Neural Network

Patel Harsh BhavinBhai
214103227
Soft Computing (ME674)
Coding Assignment

DED melt pool dimensions ANN to Forecast LASER Weld power Requirement

• Introduction: -

Problem Definition: -

This Experiment Consist Relationship Between Different Features of LASER Welding Parameters as Well as Material Properties with Laser Power.

Here 20 Different input features are available for prediction of the LASER Power Requirement.

Background: -

Published: 23 November 2020|Version 4|DOI:10.17632/8nxk8f36j7.4
Contributors:
Darren Feenstra,
Andrey Molotnikov,
Nick Birbilis

Melt pool dimensions of cross-sections of single track DED deposits produced by a Trumpf Trusper Cell 7040. More details of the work herein are contained in the work titled "On the use of Artificial Neural Networks to Determine Processing Windows in Directed Energy Deposition Applications.

• Methodology: -

Optimal Number of Hidden Neurons: -
After Reviewing some Literatures, I got to know that optimal Number of Hidden Neurons are;

$N_h = (n + n_i)C\ n_i / n_i + 2$; OR N_h = Square Root($n_i * n_o$) ;
Here I have 20 input Features and 1 output Feature, so I had taken 7 hidden neurons For the ANN model.

Size Of Data-Sets: -
By looking at the Literature review we come to know;
Size Of Training Data-Set (N) =

$$\text{Order of}\left(\frac{\text{Total Number of free parameters}}{\textit{Fraction Of Error Permitted on test data}}\right)$$

I had chosen the accordingly the big dataset for that.

Activation Function: -
I had chosen Log-Sigmoid T.F. Through After Normalizing data Between 0.1 to 0.9. Because Outputs are in the range of 0 to 1 only for Log-Sigmoid.

Learning Rate: -
It is Chosen by Trial-and-Error Method. First, we take a Lower Learning Rate and see the Number of Iteration for Training of ANN. Then I slightly Increase the Learning Rate To the optimum values.

Momentum Term: -
To make the learning faster I use a Momentum Term to increase the Rate of Convergence, overcome the local minima etc.
Here I choose random Momentum Coefficient and then Increase It to some Optimum value.

- **Results And Discussion: -**

  Here I made a code with a While loop until the Mean Square Error become Less than 0.001 Approximately 0.1% Training Pattern Error. For That Process I had counted Number of Iterations And tried to minimise the Iteration By changing the above parameters.

  I had taken 20 input Features for 1 output so I get approximately 167 Iterations For Training The ANN to make up to 0.08 error on Test Set.

  Here I had 277 Patterns. Among which I choose 159 as Training Pattern and remaining 118 as a Test Pattern.

  Here I trained the model in such a way that it will train until the MSE becomes less than 0.001.

  I choose SEQUENTIAL MODE of Training for that I don't Required to Shuffle the Data-set. I can Directly go for training Of ANN.

  After Successfully Weight Updatation after each Iteration we get the Optimum weights.

  Now We move On the Test Pattern to check If our model Predict exact value of the Patterns then We see the MSE overall for Test Patterns and I get Average MSE 0.07(7%) error for all the test Pattern. I think it is a good Approximation for the given data set

Results: -

Test Pattern Number	Network output	Mean Square Error
1	891.550224	0.12423
2	891.719914	0.124197
3	891.34196	0.124269
4	892.723078	0.124007
5	889.232091	0.12467
6	888.434238	0.212568
7	890.218696	0.212125
8	887.293574	0.188722
9	889.211502	0.188273
10	889.915256	0.188109
11	890.343635	0.188009
12	891.331454	0.187778
13	891.784102	0.187672
14	891.621796	0.18771
15	892.109093	0.187596
16	891.629736	0.187708
17	891.703488	0.187691
18	903.450141	0.000777
19	898.906247	0.000742
20	895.265886	0.006738
21	891.460294	0.018766
22	889.384043	0.036642
23	887.715951	0.060391
24	886.888852	0.089906
25	886.418779	0.125206
26	885.899467	0.166349
27	885.502948	0.213297
28	889.745149	0.036605
29	890.233322	0.036555
30	892.6281	0.036308
31	893.709682	0.036197
32	899.024368	0.035655
33	893.78729	0.088795
34	895.579951	0.036006
35	896.311982	0.035931

36	894.318149	0.036135
37	893.461013	0.036223
38	892.125674	0.03636
39	892.460446	0.036326
40	894.205264	0.059535
41	898.237727	0.059006
42	898.83174	0.058928
43	900.337699	0.058731
44	902.391051	0.058464
45	903.454948	0.058325
46	900.355623	0.087744
47	897.147371	0.059149
48	898.28731	0.03573
49	906.0799	0.006269
50	899.899565	0.018148
51	896.918685	0.035869
52	892.164582	0.059803
53	889.457781	0.089491
54	888.276317	0.124852
55	887.083791	0.166089
56	886.872525	0.212956
57	897.49072	0.018323
58	899.505526	0.018176
59	897.716601	0.018307
60	896.005576	0.018432
61	896.655255	0.018384
62	897.354966	0.018333
63	893.58836	0.03621
64	891.060435	0.059949
65	889.058295	0.089556
66	888.574172	0.124795
67	893.875279	0.0068
68	900.167468	0.006523
69	903.552366	0.006377
70	906.61676	0.006246
71	906.481139	0.006251
72	903.100995	0.006396
73	903.195186	0.017909
74	900.257713	0.035529
75	897.955103	0.059043
76	895.847521	0.088465
77	890.531908	0.047546

78	890.474011	0.047552
79	891.809322	0.047396
80	892.715009	0.04729
81	897.140838	0.046772
82	898.155565	0.046654
83	901.496852	0.046266
84	896.795063	0.035882
85	899.157538	0.035641
86	901.793893	0.035374
87	904.148201	0.035135
88	896.040182	0.059294
89	893.0316	0.088916
90	889.609507	0.124599
91	891.427271	0.124253
92	894.202873	0.088728
93	899.484154	0.0465
94	889.894186	0.106257
95	893.833975	0.105566
96	894.894624	0.105381
97	894.783655	0.1054
98	894.572714	0.105437
99	894.837607	0.105391
100	893.675945	0.123826
101	892.003284	0.143852
102	892.338904	0.187543
103	886.437309	0.144991
104	886.712303	0.144935
105	886.797747	0.144917
106	888.249131	0.14462
107	887.949375	0.144681
108	888.86588	0.144493
109	890.111521	0.144238
110	892.440597	0.143762
111	912.257835	0.025066
112	897.61951	0.046717
113	895.410056	0.07323
114	892.855315	0.105738
115	890.429082	0.144173
116	888.615864	0.188412
117	887.997499	0.238248
118	887.364228	0.293929

- # Conclusion: -

It Was a Good Experience to Wright a Raw Code for the Training of Artificial Neural Network. Here I learn How the Gradient Decent Work for Optimization and apply it for supervise learning for Forecasting of Weld Power Requirement depending on all the possible parameters.

Weld variables are combined to make Twenty input variables for these techniques. Literature based data is successfully used to test and validate the architectures of ANN.