Designing Neural Networks

Asked 11 years, 6 months ago Modified 5 years, 1 month ago

Viewed 17k times

33

I am learning about Neural Networks and back-propagation. I think I understand how the network works, in terms of input, output, hidden layers, weights, bias etc However, I still don't fully understand how to design a network to fit a problem. Ie: Say I wanted a neural net to learn how to play Draughts, how would I translate the problem into a neural net design? Cheers :)

neural-network

Share Follow

asked Jun 12, 2013 at 19:23

MrD

5,086 ● 12 ● 53 ● 95

Modeling the problem using a neural network (or any other kind of model) is a challenging problem; there is no magic bullet for this. I would recommend reading about techniques others have developed and see if you can apply those to your problem. You could start with a reference like, en.wikipedia.org/wiki/Types_of_artificial_neural_networks And search through google scholar for more examples.
– hazydev Jun 12, 2013 at 20:00

3 Answers

Sorted by: Highest score (default)

25

✓

There are definitely a lot of decisions to be made in designing a neural net, and there is no one right answer. However, there are a few general questions that are often helpful to think about:

1. What are you trying to generate as an output? Draughts seems like a challenging game to play with a neural net, because there are a lot of potential moves and the ones available change from turn to turn, but presumably you would want the output to be the next move.
2. What are your inputs? This should include anything that you think would be useful in making the decision that you want the neural net to make. In the draughts example, you'd probably need to give the neural net the locations of all pieces on the board.
3. Recurrent or feed-forward? Generally, unless there's a really salient reason for giving it information about what it's done in the past, it's best to use feed-forward, because the enables you to train the net with back-propagation. For draughts, for instance, you'd probably want to use a feed-forward network.
4. Do you need a hidden layer? This is a harder question to know the answer to, and might require some experimentation, unless you know a lot about the high-dimensional space that your inputs occupy.

> Draughts is complex enough that it seems like it would require a hidden layer, but it's hard to be sure.

Obviously, there are a lot more decisions that can/have to be made about neural net set-up, but hopefully these will get you going.

Share Follow

answered Jun 13, 2013 at 19:26

seaotternerd

6,409 ● 2 ● 48 ● 59

6

Well, I think your problem is the problem of any other NN designer... One thing you have always to keep in mind is that NNs are heuristic models. Therefore, they learn from experience, similarly to us.. You cannot "insert" pure knowledge into the NN (that is possible in other machine learning algorithms) My approach to your problem, or any general problem I face, is to start with the questions: "How would I teach this to someone?", "what exercise will I propose so the person can learn it?"

You have to know/define the rules of the game, and which variables you can play with and what you want to achieve. Then, you have to train the network (get data) with the goal of wining the game, as if it'd be a child. After enough data and weight changes the NN should be able to answer reasonable plays to win the game... as more data you get, more accurate answer you might have, and therefore, a better player!

Although this may seem simple enough, there are many many aspects of training a neural network or any other machine learning algorithm that you have to take into account such as defining the type of model, a suitable loss function, proper architecture fine-tuning, training/testing data sampling, etc ad infinitum...

It's nothing conclusive nor linear, but is my perspective ;) Good luck!

Share Follow

edited Oct 29, 2019 at 12:50

answered May 20, 2014 at 9:57

diogoncalves
164 ● 1 ● 8

1 What's an "SC" algorithm? – HelloGoodbye Jan 11, 2016 at 21:04

1 Sorry, Soft Computing Algorithm. Soft Computing includes neural computing, evolutionary computing and fuzzy logic algorithms – diogoncalves Sep 15, 2016 at 14:43

0

Since you are developing AI agent for the game you should read Reinforcement Learning (Q networks) that is ideally applied for such tasks.

Share Follow

answered Oct 16, 2019 at 17:25

Maksim Shamihulau
1,658 ● 1 ● 21 ● 21