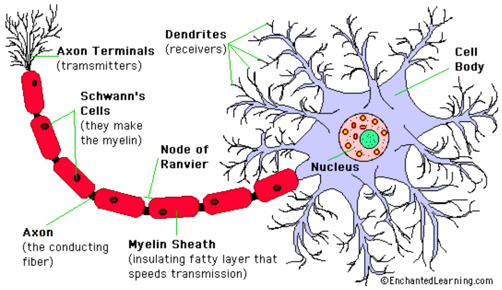
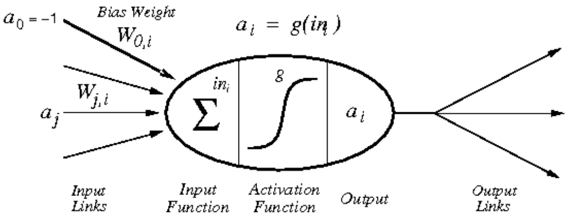
Machine Learning
Artificial Neural Networks

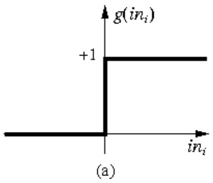
Neurons



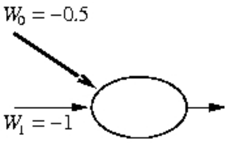
A computational model of a neuron



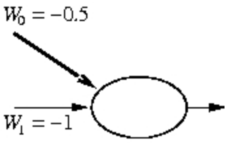
A simple threshold activation function



An example neuron

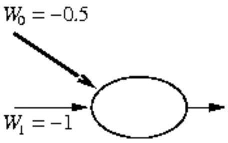


An example neuron



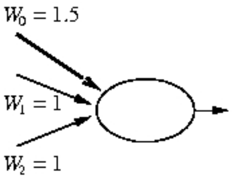
- to calculate the output, take the weighted sum of the inputs and apply the activation function

An example neuron

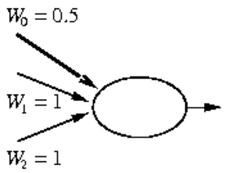


- to calculate the output, take the weighted sum of the inputs and apply the activation function
  - input: 1 -> output: -1
  - input: -1 -> output: 1

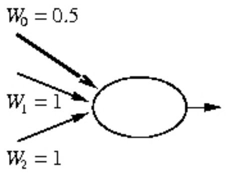
Another example



Yet another example



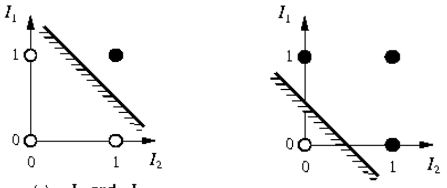
Yet another example



Exercises

1) Design a neuron that takes two inputs (plus the bias) and outputs 1 if both inputs are 0 and outputs 0 otherwise.
2) Design a neuron that takes two inputs (plus the bias) and outputs 0 if both inputs are 1. In all other cases it outputs 1.

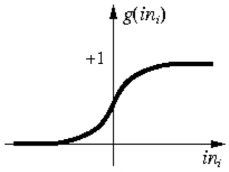
Linear separability



(a) I_1 and I_2

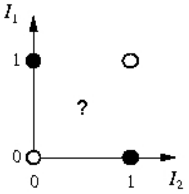
(b) I_1 or I_2

A more common activation function



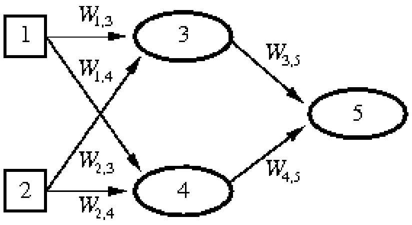
(b)

Something our artificial neuron cannot express

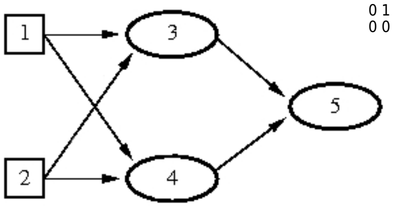


(c) I_1 xor I_2

Networks of Neurons

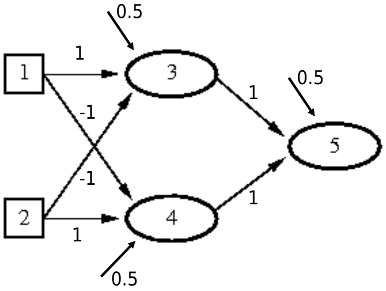


XOR neural net



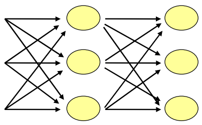
desired behavior:
1 1 → 0
1 0 → 1
0 1 → 1
0 0 → 0

XOR neural net

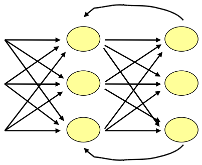


Network Architecture

feed-forward:



recurrent:

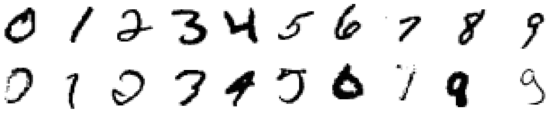


Most common: feed-forward networks with one hidden layer.

What can we use them for?

- pattern recognition/classification
- clustering (grouping of similar patterns)
- function approximation
- prediction/forecasting
- optimization
- associative memory (memory which can be accessed by its content rather than by address)
- control

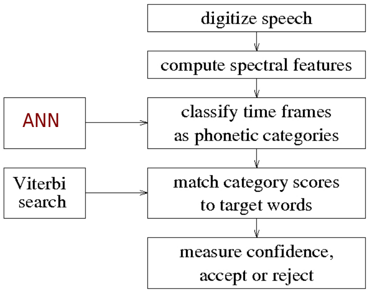
Character Recognition



a neural network for character recognition:

- what's the input?
- what's the output?

Speech Recognition



digitize speech

compute spectral features

ANN

classify time frames as phonetic categories

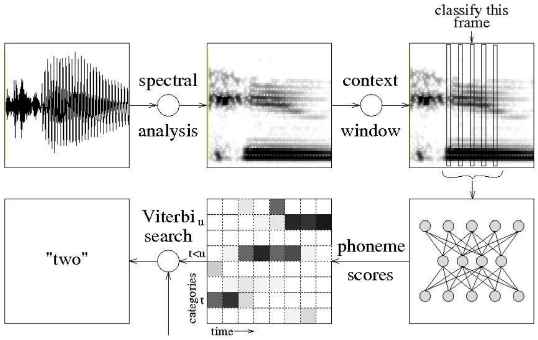
Viterbi search

match category scores to target words

measure confidence, accept or reject

source: Oregon Graduate Institute, Center for Spoken Language Understanding

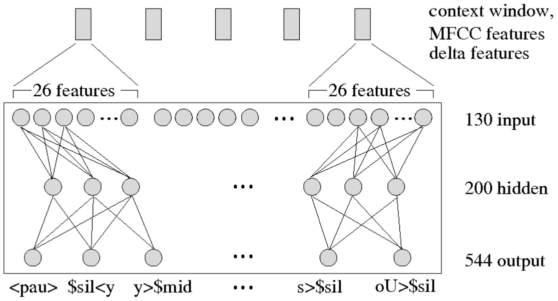
Speech Recognition



source: Oregon Graduate Institute, Center for Spoken Language Understanding

Speech Recognition

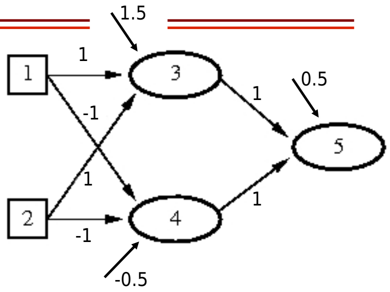
for each frame of input:



5-frame context window, MFCC features delta features

source: Oregon Graduate Institute, Center for Spoken Language Understanding

Exercises



1) What is the behavior of this neural network? What input patterns get mapped to what outputs? Would it be possible to achieve the same effect with just one neuron?

2) Design a neuron that takes two inputs (plus the bias) and outputs 1 if both inputs are 0 or the first input is 0 and the second 1. Otherwise, it outputs 0.

Where do the weights come from?

Where do the weights come from?

- Given :

  example inputs + the desired output for each of these inputs

- Learning strategy (backpropagation):

  1. randomly assign small weights

  2. for each pattern:

  – feed pattern into the NN

  – compare actual output to desired output and adjust weights to make actual output closer to desired

  6. repeat from step 2 until the weight changes get very small (or some other stopping criterion is met)

Adjusting the weights

actual output: [o1, o2, ..., on]

desired output: [t1, t2, ..., tn]

error: $\frac{1}{2}\sum_{i=1}^{n}(t_i - o_i)^2$

goal: minimize the error

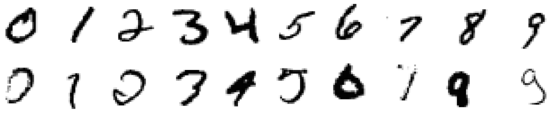
- Adjust the weights of on the output units and then of the hidden units:

  $w_{i,j} = w_{i,j} + \Delta w_{i,j}$

Exercises

- Download bpnn.py and bpnn_tester.py.
- bpnn_tester.py defines a function `test1` which shows you how to use the bpnn module to build, train and test neural networks.
- Add a function which builds, trains and tests a neural network that computes a boolean *and*.
- Add a function which builds, trains and tests a neural network that decides whether the inputs represent a binary number which is divisible by 3. Use four input units – this will allow you to represent the numbers 0,...,15.

Character Recognition



a neural network for character recognition:

- what's the input?
- what's the output?

A more interesting exercises

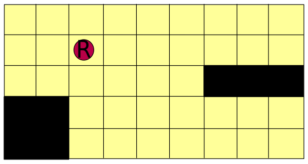
- Download digits.zip and decompress it. optdigits.names contains a description of the data, the other three files contain data for training and testing.
- Download handwritten_digits.py.
- Complete the program so that it builds, trains, and tests a neural network that can distinguish between the digit '8' and the digit '9'.

  A function to read in the training and testing data from the files is already provided. All you have to do is to build, train, and test the neural network.

Exercise: learning how to follow a wall

If we wanted to design a neural network to control our wall-following stimulus-response agent,

- what would be the inputs?
- what would be the outputs?
- how many input and output neurons would we need?



Exercise: learning how to follow a wall – step 1

Using the bpnn module, write code to build, train, and test a neural network that learns how to find and follow a wall. That is, given a sequence of inputs representing the sensory information the agent has, this neural net should produce an output representing the direction in which the agent should take its next step.

Use the training patterns that I provide in wall_data.txt for training and also testing purposes. Try out different numbers of hidden units - what works best?

Exercise: learning how to follow a wall – steps 2 and 3

Now integrate this code into our simulation of a wall following agent, such that a neural network is trained and then used to guide the agent.

2) First, download our old code for the wall following agent and plan what changes you need to make. What do you need to add? What do you need to change? Where does the network get built? Where do you use the network to predict the next step?

3) Then implement this. To do this you need the method `update` that neural networks built with bpnn have. The method update takes a list of inputs and returns the list of outputs for those inputs. That is after a network `n` has been created and trained you can use the statement `outputs = n.update(inputs)` to calculate the outputs the trained network produces give the list of inputs `inputs`.

Topics for the final

Everything we have done.

- algorithms, control flow, expressions and statements, variables and assignment, functions, conditional statements and loops, lists, dictionaries, tuples, searching and sorting lists, recursion, files.
- Turing test, Searle's Chinese room, different views on the purpose of AI (thinking/acting humanly/rationally)
- rational agents, stimulus response agents, artificial life, n-grams, neural nets