greatlearning
Learning for Life

Deep Neural Networks

Introduction

Deep Neural Network TOC

1. Introduction to Deep Neural Network
    a. What is it?
    b. Artificial Neuron
        I. It's inspiration from Neurology
        II. Boolean gates
        III. First attempt at Artificial Neuron - McCulloch Pitts Neuron / Rosenblatt Neuron (Perceptron)
        IV. Weakness of Perceptron
    c. Evolution of Artificial Neural Network
    d. Fully connected Artificial Neural Network
    e. Mathematical foundations of Artificial Neural Networks

2. Building blocks of ANN
    a. Layer
        I. Input layer
        II. Hidden Layer
        III. Output Layer
    b. Neuron structure
        I. Connections and weights
        II. Linear transformations
        III. Activation functions
        IV. Bias / bias layers
    c. Forward propagation
    d. Backward propagation
    e. Loss function
    f. Gradient Descent

3. Code walk-thru

Introduction to Deep Neural Network

What is Deep Neural Network

Deep Neural Network / Artificial Neural Network

1. Deep Neural Network (DNN) is a supervised learning method based on Artificial Neural Network computing methods

2. Artificial Neural Networks (ANN) is inspired by biological neural networks and employs a collection of interconnected artificial neurons to extract the patterns from given data

3. Artificial Neurons are software objects that mimic behaviour of biological neurons

4. The early research artificial neuron borrowed ideas such as thresholds, linear summation, neuron firing etc. from neurology

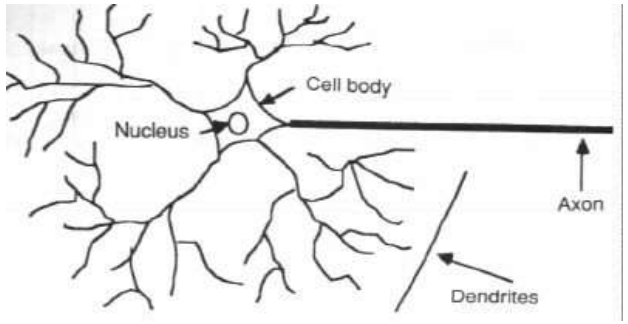
5. The contemporary research and development in DNN is not as tightly coupled with neurology as it was in the early days

Artificial Neurons

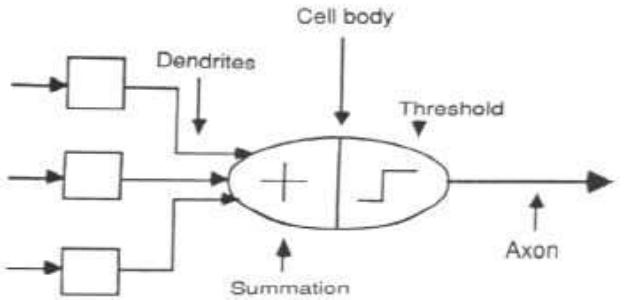
A brief history

Inspiration from Neurology

Biological Neuron



Artificial Neuron



http://www.andreykurenkov.com/writing/ai/a-brief-history-of-neural-nets-and-deep-learning/

Boolean Gates and Artificial Neurons

1. The objective of research in artificial neuron was to develop a computing system that could learn to do tasks on it's own without instruction how to do it

2. Most of the tasks that we do in our day to day life are classification. Hence, research was on to develop an artificial neuron that could classify

3. Since computing systems are based on Boolean gates which generated two classes , it was natural to check whether the artificial neurons can learn to mimic these gates such as the OR, AND gate

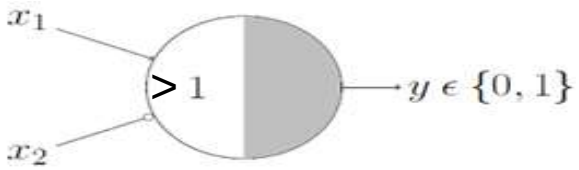
AND

INPUT		OUTPUT
A	B	Q
0	0	0
0	1	0
1	0	0
1	1	1

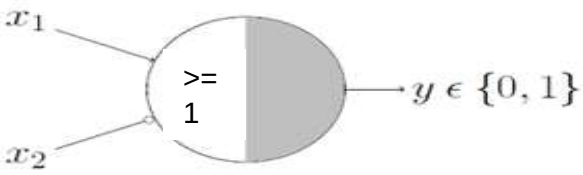
OR

INPUT		OUTPUT
A	B	Q
0	0	0
0	1	1
1	0	1
1	1	1

Warren McCulloch & Walter Pitts Neuron



AND Gate



OR Gate

1. In McCulloch Pitt neurons inputs and outputs are binary. The output is only one but inputs can be many

2. All inputs have same positive weights and hence not shown in the picture

3. The inputs multiplied with corresponding weights are summed up and the result sent to a step function.

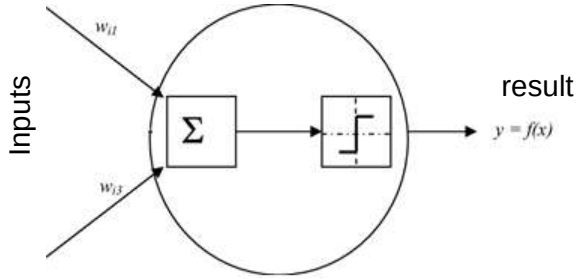
4. The threshold of the step function is fixed (for e.g. 1 for “AND” gate with two inputs each with weight 1)

5. The threshold has to be modified if number of inputs is greater than two. What would the threshold be if we had three inputs?

McCulloch Pitts Neuron

AND Gate

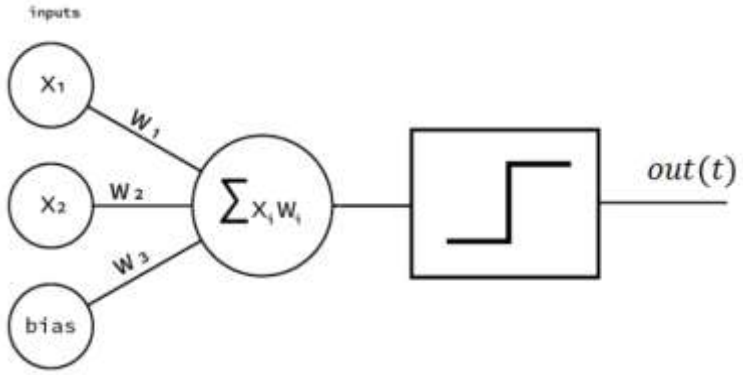
```
w[1] = 1
w[0] = 1

training_data = [
    (array([0,0]), 0),
    (array([0,1]), 0),
    (array([1,0]), 0),
    (array([1,1]), 1),
]

# Step function with threshold of 1 Anything equal to or below is 0
step_function = lambda sum: 0 if sum < 2 else 1

w[1] = 1
w[0] = 1

for x, _  in training_data:
    result = dot(x, w)
    print("{}: {} -> {}".format(x[:2], result, step_function(result)))
```



McCulloch Pitts Neuron

1. The McCulloch Pitts neuron is too simplistic.
2. The threshold have to be encoded i.e. there is no learning!

Rosenblatt Neuron / Perceptron



1. Uses weights for the inputs. A concept borrowed from William Hebb's rule "neurons that wire together, fire together"
2. Other than the inputs they also get a special 'bias' input, which just has a value of 1
3. Incorrect outputs for a given input are captured as errors
4. A learning rule modifies the weight to correct errors in output
5. The neuron could behave like "AND" gate or "OR" gate with same threshold by adjusting weights
6. This neuron learns the patterns from the data. If the data is for "AND" gate, it learns to mimic "AND" gate
7. This concept of learning from data was a major improvement over McCulloch-Pitts neuron.

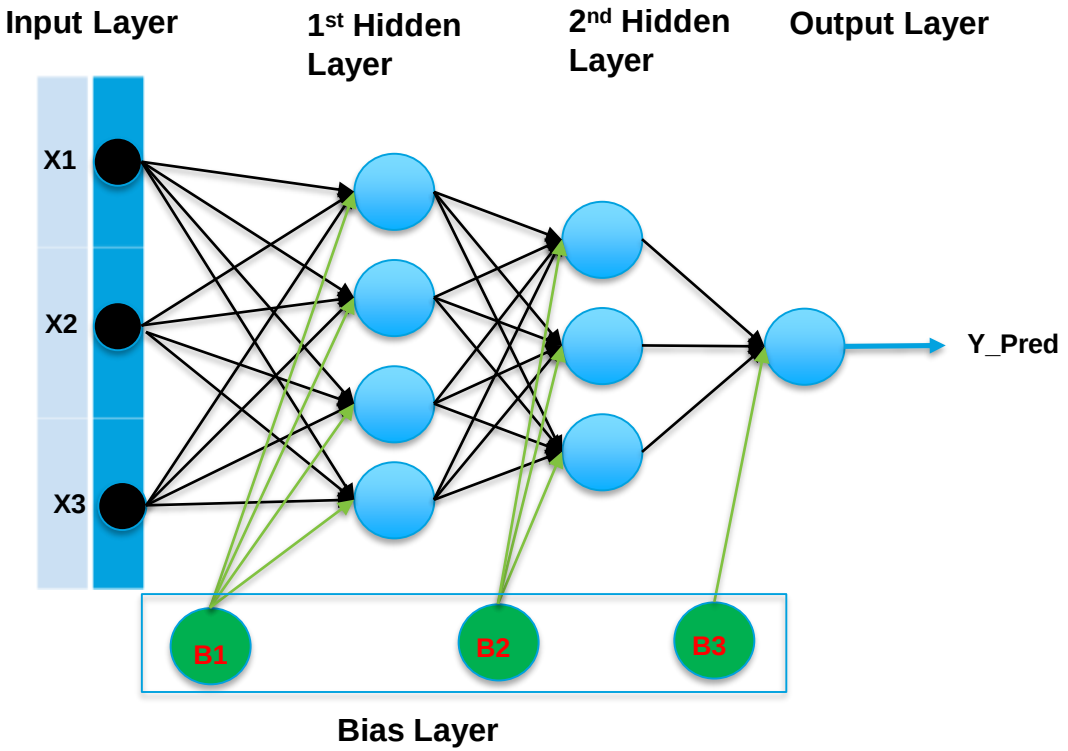
Weakness of Perceptron

1. Papert and Minsky demonstrated that the perceptron was incapable of handling some of the binary gates such as XOR**
2. Given that it cannot represent all possible binary gates, it could not have been used for all possible computations hence the objective of computer based AI was a pipe dream!
3. It was subsequently demonstrated that instead of making a neuron intelligent, a network of neurons can be used to do what a single neuron could not
4. This was the birth of a Artificial Neural Network

Evolution of Artificial Neural Networks

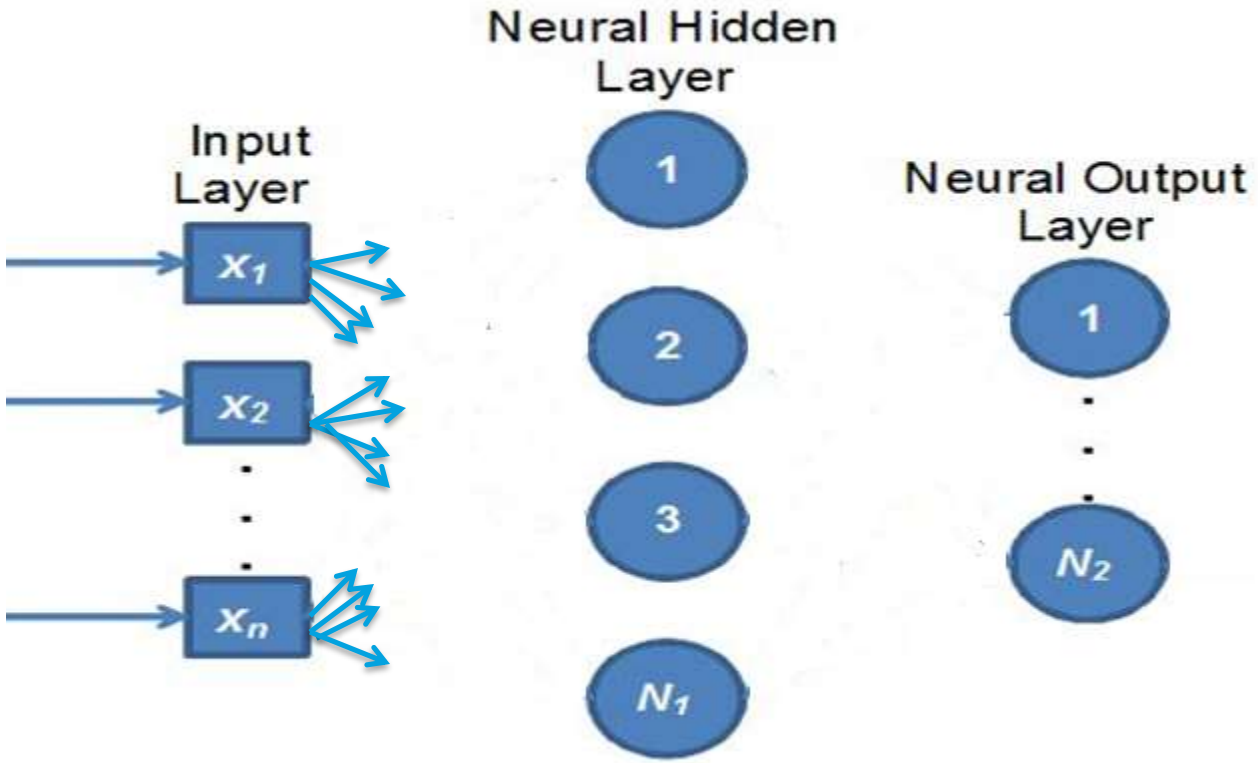
Artificial Neural Network

The processing elements of a ANN is called ***a node***, representing the artificial neuron. Each ANN is composed of a collection of nodes grouped in layers. A typical structure is shown The initial layer is the input layer and the last layer is the output layer. In between we have the hidden layers



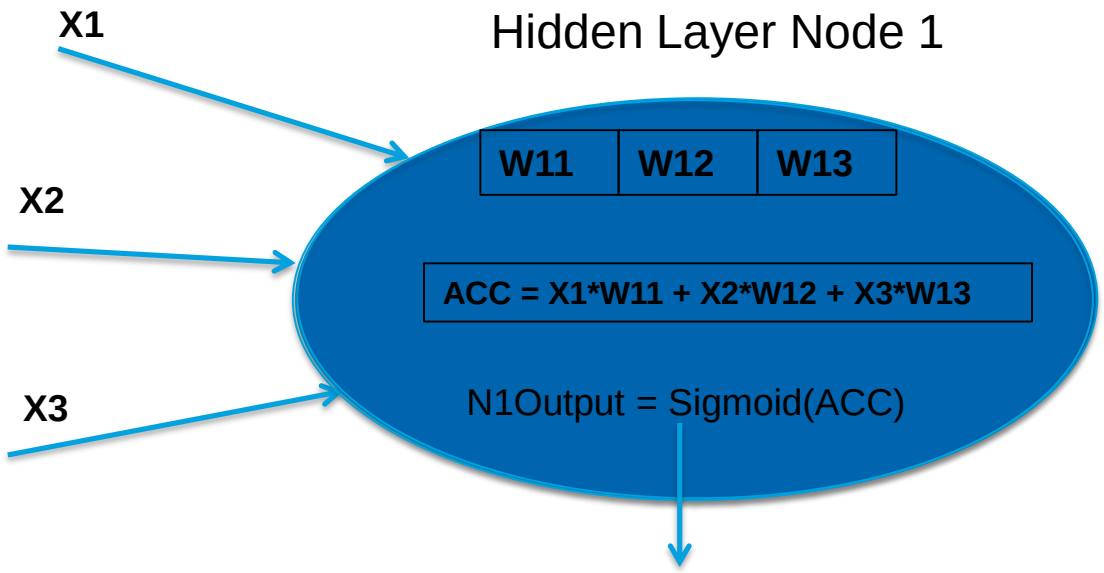
Fully connected Artificial Neural Network

1. The input layer is passive, does no processing, only holds the input data to supply it to the first hidden layer



Fully connected Artificial Neural Network

2. Each node in the first hidden layer, takes all input attributes, multiplies with the corresponding weights, adds bias and the output is transformed using non_linear function

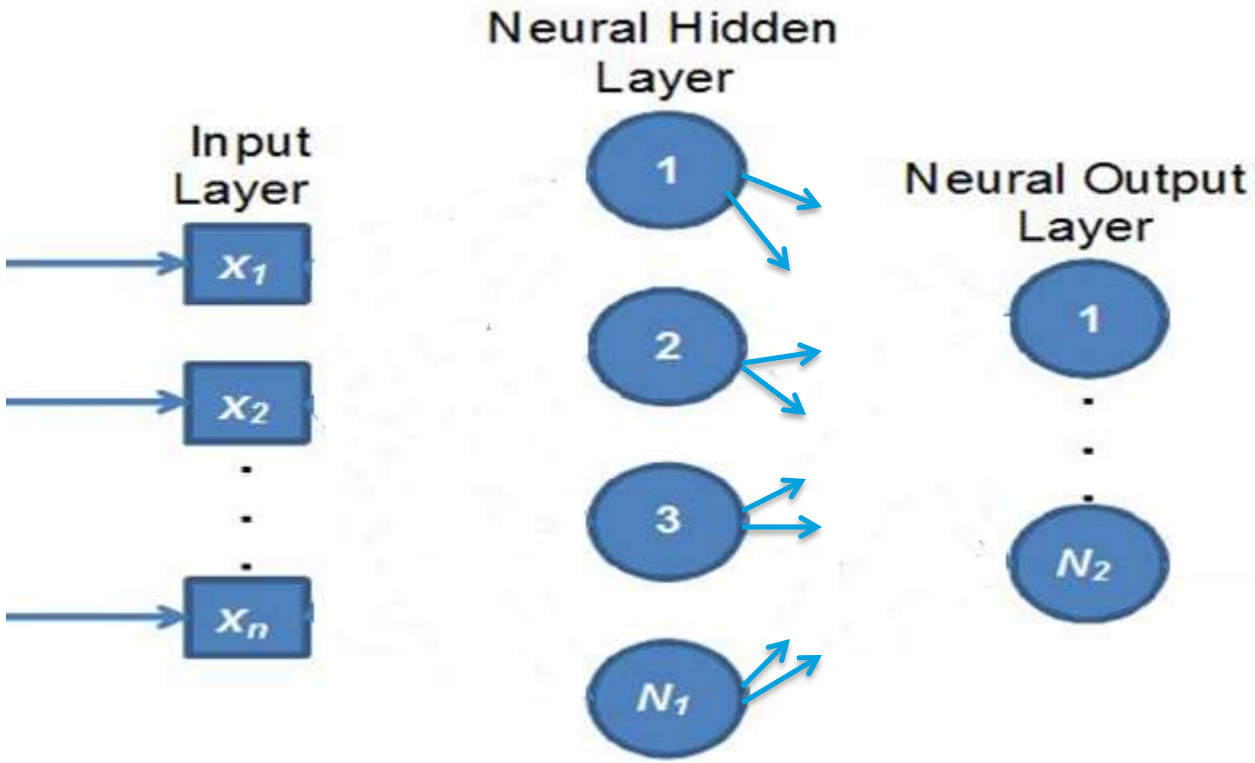


3. The weights for a given hidden node is pre-fixed and all the nodes in the hidden layer have their own weights

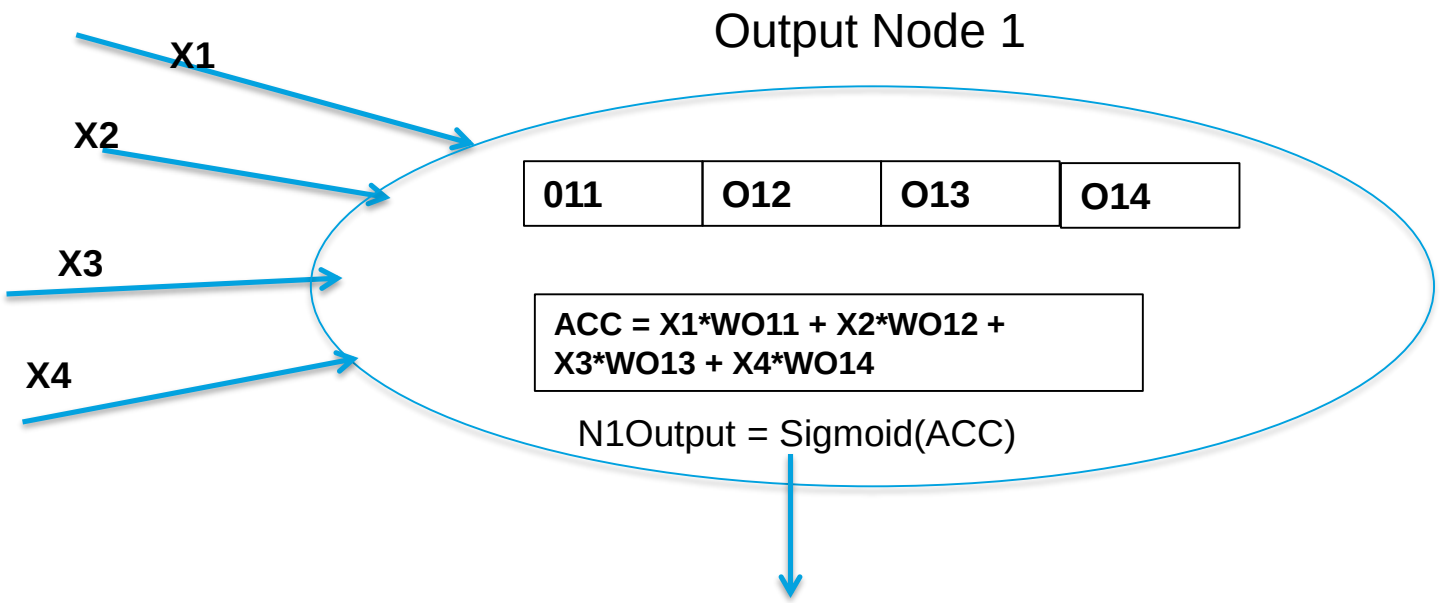
4. The output of each node is fed to output layer nodes or another set of hidden nodes in another hidden layer

Fully connected Artificial Neural Network

5. The output value of each hidden node is sent to each output node in the output layer

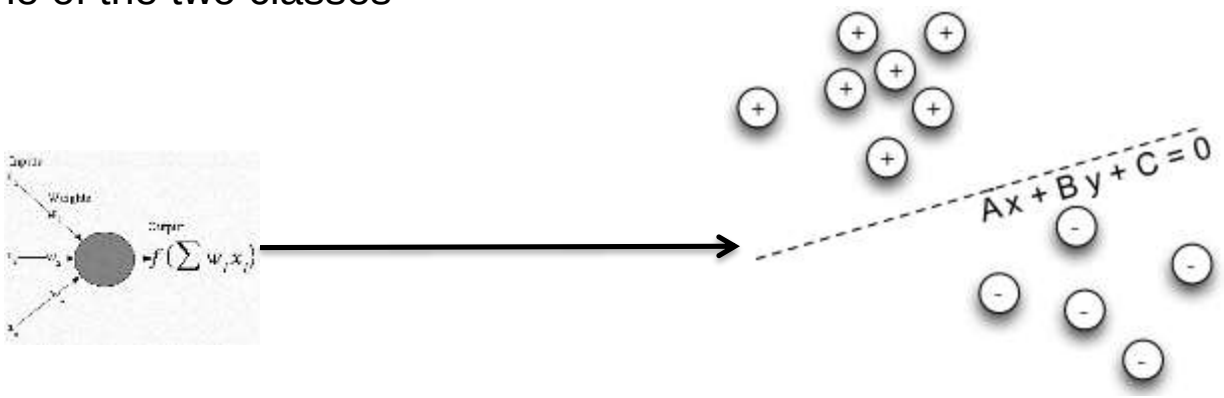


Fully connected Artificial Neural Network



Fully connected Artificial Neural Network

6. In a binary output ANN, the output node acts like a perceptron classifying the input into one of the two classes



7. Examples of such ANN applications would be to detect fraudulent transaction, whether a customer will buy a product given the attributes etc.

8. ANN can also be used for multi-class classification problems such as digit recognition. In fact, it is often used for multi-class classification

Mathematical foundations for Artificial Neural Networks

a. Kolmogorov theorem – any continuous function f defined on n-dimensional cube is representable by sums and superpositions of continuous functions of only one variable

$$f(x_1, x_2, \ldots, x_n) = \sum_{q=1}^{2n+1} g\left(\sum_{p=1}^{n} \lambda_p \phi_q(x_p)\right)$$

b. Cover's theorem - states that given a set of training data that is not linearly separable, one can with high probability transform it into a training set that is linearly separable by projecting it into a higher-dimensional space via some non-linear transformation.