Error-Correction Learning Demo: Training a Perceptron on AND and OR Tasks

Abstract

Error-correction learning is one of the fundamental supervised learning mechanisms in Artificial Neural Networks. The Perceptron algorithm uses an error-driven update rule to iteratively adjust weights in order to minimize classification errors. This report demonstrates the implementation of the Perceptron learning rule on two classical logical problems: AND and OR gates.

The model is trained manually using step-by-step calculations to illustrate how weights evolve across iterations. The decrease in classification error is tracked over epochs to show convergence behavior. Additionally, the effect of different learning rates on speed of convergence and stability is analyzed. Graphical interpretation of error reduction and decision boundary formation is included conceptually.

The results confirm that the Perceptron successfully learns linearly separable functions and converges after a finite number of updates. The experiment also highlights the importance of choosing an appropriate learning rate for efficient training.

---

1. Introduction

Artificial Neural Networks (ANN) are computational systems inspired by biological neurons. The Perceptron, introduced by Frank Rosenblatt (1958), is the simplest form of a neural network capable of solving binary classification problems.

Error-correction learning is based on adjusting weights in proportion to the difference between predicted output and target output. This difference is known as the error signal. The Perceptron modifies its weights only when misclassification occurs.

In this report:

- A Perceptron is trained on AND and OR datasets.
- Error values are computed per epoch.
- Learning rate variations are analyzed.
- Convergence behavior is studied.

---

2. Objectives

1. To implement the Perceptron error-correction learning rule.
2. To train the model on AND and OR logical datasets.
3. To track classification error across iterations.
4. To study convergence behavior.
5. To analyze the effect of learning rate on training stability.

---

3. Theoretical background

3.1 Perceptron model

For inputs:

x1,x2x_1, x_2x1,x2

Weights:

w1,w2w_1, w_2w1,w2

Bias:

bbb

Net input:

z=w1x1+w2x2+bz = w_1x_1 + w_2x_2 + bz=w1x1+w2x2+b

Activation function (step function):

y={1z≥00z<0y = \begin{cases} 1 & z \ge 0 \\ 0 & z < 0 \end{cases}y={10z≥0z<0

3.2 Error-correction learning rule

Weight update:

wi(new)=wi(old)+η(t−y)xiw_i(new) = w_i(old) + \eta (t - y)x_iwi(new)=wi(old)+η(t−y)xi

Bias update:

b(new)=b(old)+η(t−y)b(new) = b(old) + \eta (t - y)b(new)=b(old)+η(t−y)

Where:

- η\etaη = learning rate
- ttt = target output
- yyy = predicted output

Error term:

e=t−ye = t - ye=t−y

Total error per epoch:

E=∑∣t−y∣E = \sum |t - y|E=∑∣t−y∣

Training continues until total error becomes zero.

---

4. Dataset description

4.1 AND gate

x1	x2	Target
0	0	0
0	1	0
1	0	0
1	1	1

4.2 OR gate

x1	x2	Target
0	0	0

0	1	1
1	0	1
1	1	1

Both datasets are linearly separable.

---

5. Training demonstration – AND gate

5.1 Initialization

Learning rate: η=1\eta = 1η=1
Initial weights: w1=0,w2=0w_1 = 0, w_2 = 0w1=0,w2=0
Bias: b=0b = 0b=0

Epoch 1

Input (0,0), Target=0
z=0 → y=1
Error = -1
Update: b = -1

Input (0,1), Target=0
z=-1 → y=0
Correct

Input (1,0), Target=0
z=-1 → y=0
Correct

Input (1,1), Target=1
z=-1 → y=0
Error=1
Update:
w1=1
w2=1
b=0

Total error (epoch 1) = 2

Epoch 2

Repeat calculations.
After several updates:

Final weights converge approximately to:

w1 = 1
w2 = 1
b = -1.5

Now all outputs are correct.

Total error = 0

Convergence achieved.

---

6. Training demonstration – OR gate

Using same initialization.

After applying updates:

Converged weights approximately:

w1 = 1
w2 = 1
b = -0.5

Total error reduces to zero within fewer iterations compared to AND.

Reason: OR boundary is easier (less strict condition).

---

7. Error decrease across epochs

Example error progression (AND case):

Epoch	Total Error
1	2
2	1

3	0

Graphically, error decreases stepwise until it reaches zero.

The error curve resembles a downward staircase, not a smooth curve, because the perceptron uses a discrete update rule.

---

8. Learning rate effects

Case 1: Small learning rate (η = 0.1)

- Small weight updates
- Slower convergence
- Stable learning
- More epochs required

Case 2: Moderate learning rate (η = 1)

- Faster convergence
- Stable updates
- Efficient training

Case 3: Large learning rate (η = 5)

- Large weight jumps
- Possible oscillations
- May overshoot solution
- Can delay convergence

Conclusion: Learning rate controls speed vs stability trade-off.

---

9. Decision boundary interpretation

For AND:

x1+x2−1.5=0x_1 + x_2 - 1.5 = 0x1+x2−1.5=0

This line separates class 1 (only when both inputs are 1) from class 0.

For OR:

x1+x2−0.5=0x_1 + x_2 - 0.5 = 0x1+x2−0.5=0

This boundary allows classification when at least one input is 1.

Both are straight lines → confirms linear separability.

---

10. Discussion

Observations:

1. Error decreases only when misclassification occurs.
2. Weights remain unchanged when prediction is correct.
3. Convergence is guaranteed for linearly separable data.
4. OR converges faster than AND.
5. Learning rate influences convergence speed but not final solution (for separable problems).

The experiment validates the Perceptron convergence theorem.

---

11. Limitations

1. Cannot solve XOR (non-linearly separable).
2. Only binary output.

3. Hard threshold activation.

4. No probability output.

---

12. Applications

1. Simple pattern recognition

2. Logical circuit modeling

3. Linear classification

4. Introductory machine learning systems

---

13. Conclusion

This experiment demonstrated the Perceptron error-correction learning mechanism using AND and OR logical tasks. Through manual computation, the weight update process was analyzed step-by-step. Error values were tracked per epoch, showing a monotonic decrease until convergence.

The study confirmed that the Perceptron successfully learns linearly separable problems and that learning rate significantly influences convergence speed. Although limited in complexity, the Perceptron forms the conceptual basis of modern neural networks and deep learning architectures.

---

14. References

1. Rosenblatt, F. (1958). The Perceptron: A probabilistic model for information storage and organization in the brain.

2. Haykin, S. Neural Networks and Learning Machines.

3. Bishop, C. Pattern Recognition and Machine Learning.

4. Goodfellow, Bengio, Courville. Deep Learning.

5. ANN Course Lecture Notes.