Continuation of Perceptron

JOHN HANCOCK, Florida Atlantic University, USA

This article is on the weight updating algorithms that Zhu explains in Lecture 3 and lecture 4. It is based on the material Xinguan Zhu covers in [2]

Additional Key Words and Phrases: Deep Learning

1 INTRODUCTION

My main take-away from Zhu's lecture so far is on the two weight updating Algorithms that he presents in [2].

2 MAIN BODY

The main thing I want to reflect on is the weight updating rules.

Zhu calls the first weight updating algorithm he introduces the, "Perceptron Learning Rule."

For this rule, we update weights immediately, according to the rule:

$$\Delta w_i = \eta \left(d\left(n\right) - a\left(n\right)\right) x_i. \tag{1}$$

It is important to note that we add this change in weights to the weights after computing the output.

The second weight update algoritm Zhu covers he calls the, "Gradient Descent Learning Rule."

$$\Delta w_i = \eta \left(d\left(n\right) - o\left(n\right)\right) x_i. \tag{2}$$

This algorithm has a weight update calculation that looks much like 1, but the a is replaced with an o. Perhaps, for some reason, Zhu wants to stress the difference between output value and actual value. Otherwise, there is no difference in how we calculate the weight update. However, we use the value Δw_i differently.

In the Perceptron learning rule, we update weights immediately, however, in the Gradient Descent Learning Rule, we update weights after a round of training. We accumulate the sum of the Δw_i, and add that to the weights after the end of training.

So, even for one neuron, the weight updating is taking into account the Perceptron's output for all inputs in the training set before it makes an adjustment.

This is clear in the way he fills out the table in slide 21 versus slide 44, where we can see in slide 21 that Zhu changes the weight values immediately after calculating Δw_i, whereas in slide 44 he changes the weight values only after summing Δw_i for all of the output values.

3 CONCLUSIONS

Conclusion goes here

4 REFERENCES

ACKNOWLEDGMENTS

The author would like to thank the staff of the Computer Science Department of Florida Atlantic University for their priceless guidance over the years.

Author's address: John Hancock, Florida Atlantic University, 777 Glades Rd, Boca Raton, FL, 33431, USA, jhancoc4@fau.edu; john.t.hancock@aexp.com.

REFERENCES

[1] P. Bahl, R. Chancre, and J. Dungeon. 2004. SSCH: Slotted Seeded Channel Hopping for Capacity Improvement in IEEE 802.11 Ad-Hoc Wireless Networks. In *Proceeding of the 10th International Conference on Mobile Computing and Networking (MobiCom'04)*. ACM, New York, NY, 112–117.

[2] Xingquan Zhu. 2018. Perceptron Architecture and Learning. Retrieved September 3, 2018 from https://canvas.fau.edu/courses/50073/files/folder/Lectures?preview=14020569