University of Glasgow | School of Computing Science

THE AWARDS 2020 | UNIVERSITY OF THE YEAR

Optimisation of a Multi-Layer Perceptron (Part1)

Dr. Fani Deligianni,

fani.deligianni@glasgow.ac.uk

Lecturer (Assistant Professor)

Lead of the Computing Technologies for Healthcare Theme

https://www.gla.ac.uk/schools/computing/staff/fanideligianni

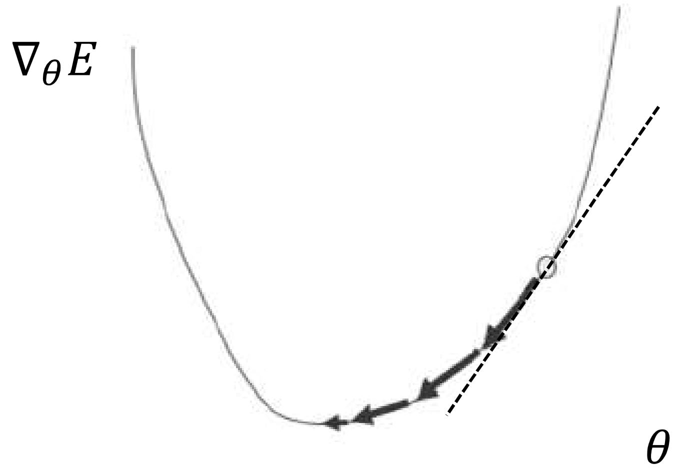
WORLD CHANGING GLASGOW

Optimisation Process

- **Optimisation algorithm and learning rate**
- Loss function
- Regularisation

Optimisation Process

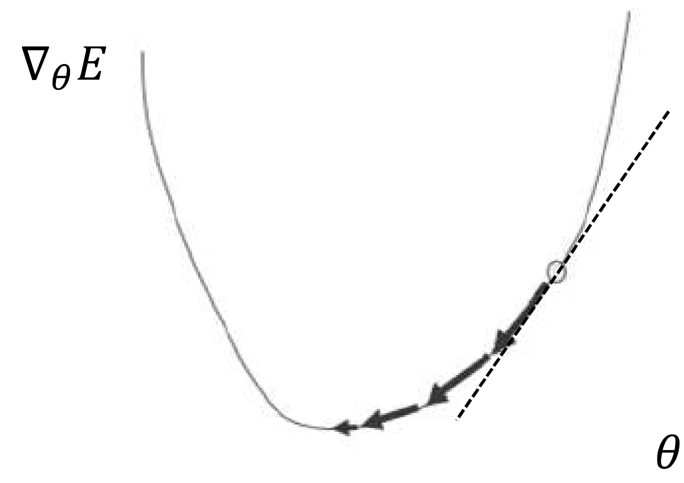
$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta E.$$



- Gradient descent minimizes the loss function iteratively:
  - Computes the slope (gradient): first-order derivative of the function at current point
  - Move in the opposite direction of the slope increase from the current point

Optimisation Process

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta E.$$



- Gradient descent minimizes the loss function iteratively:
    - Computes the slope (gradient): first-order derivative of the function at current point
    - Move in the opposite direction of the slope increase from the current point
- Batch Gradient Descent

Optimisation Process

$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta E.$$

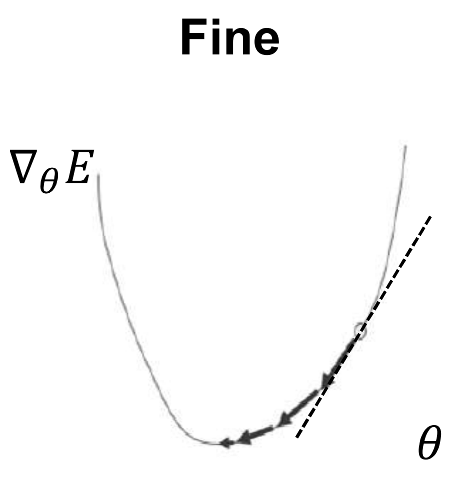
- Gradient descent minimizes the loss function iteratively:
- Batch Gradient Descent
- **Stochastic Gradient Descent**
  - Take each sample
  - Feed it to Neural Network
  - Calculate it's gradient
  - Use the gradient we calculated in step 3 to update the weights

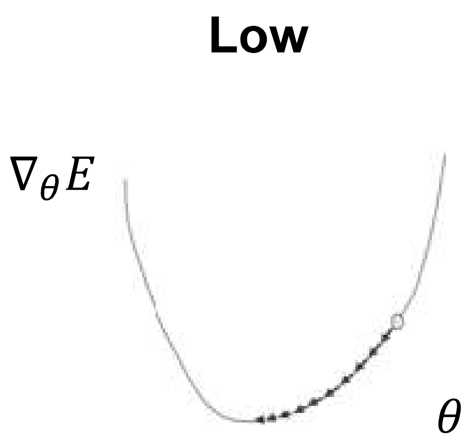
Optimisation Process

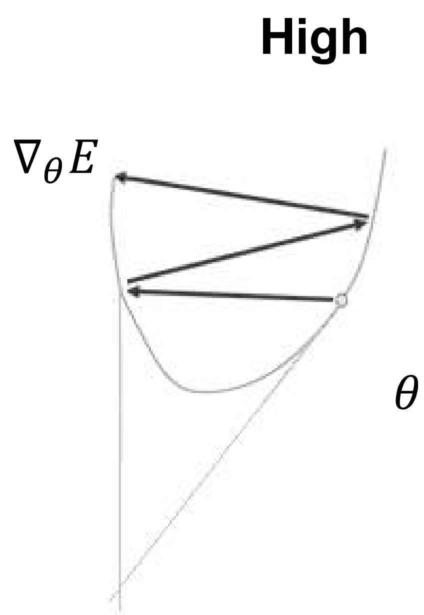
$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta E.$$

- Gradient descent minimizes the loss function iteratively:
- Batch Gradient Descent
- **Stochastic Gradient Descent**
  - Take each sample
  - Feed it to Neural Network
  - Calculate it’s gradient
  - Use the gradient we calculated in step 3 to update the weights
- Mini Batch Gradient Descent

Learning Rates







$$\theta_{t+1} = \theta_t - \alpha \nabla_\theta E.$$

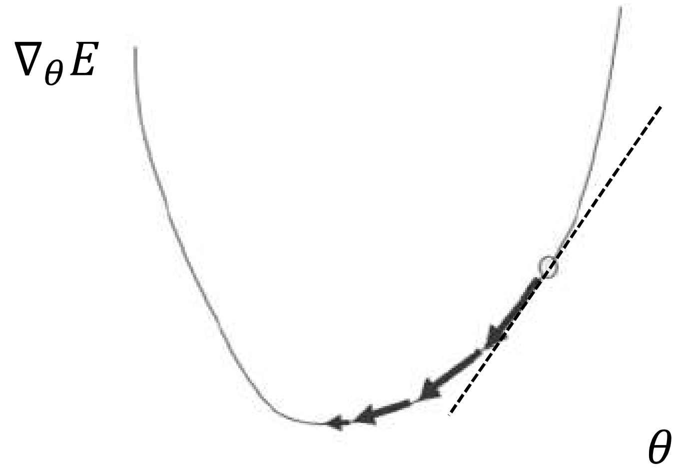
- A small learning rate slows down training and might be prohibitive in large models
- A large learning rate causes large parameters’ update that may cause divergence problems

Optimisation Algorithms

Momentum

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta E$$

$$\theta_{t+1} = \theta_t - v_t$$



Optimisation Algorithms

Momentum

$$v_t = \gamma v_{t-1} + \eta \nabla_\theta E$$

$$\theta_{t+1} = \theta_t - v_t$$

Adaptive moment (ADAM)

$$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t \quad \hat{m}_t = \frac{m_t}{1-\beta_1^t}$$

$$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2 \quad \hat{v}_t = \frac{v_t}{1-\beta_2^t}$$

$$\theta_{t+1} = \theta_t - \frac{\eta}{\sqrt{\hat{v}_t} + \varepsilon} \hat{m}_t$$

Summary

- Deep Neural Network optimization is not convex and it can result in local minimum
- Several optimization strategies have been developed that depend also on the size of the data
- Learning rate in an important hyperparameter of the training procedure

References

- Ravi et al. Deep Learning for Health Informatics, IEEE Journal of Biomedical and Health Informatics, 21(1), 2017
- Kamath, Deep Learning for NLP Applications, Springer, 2019
- Foster, Generative Deep Learning – Teaching Machines to Paint, Write, Compose and Play, O’Reilly, 2019