# Practical Methodology

Deep Learning Reading Group

July 4, 2017

Sivarajan Karunanithi

UNIVERSITÄT DES SAARLANDES

CLUSTER OF EXCELLENCE

# Model capacity

- Representational capacity
- Learning algorithm
- Performance metric

# Hyperparameters

**Architecture**

- Hidden layers
- Hidden units (HU)
- Cost function
- Activation function
- Regularization parameters

**Training**

- Instances (N)
- Iterations (n)
- Batch size (B)
- Epochs (E)
- Learning rate
- Early stopping

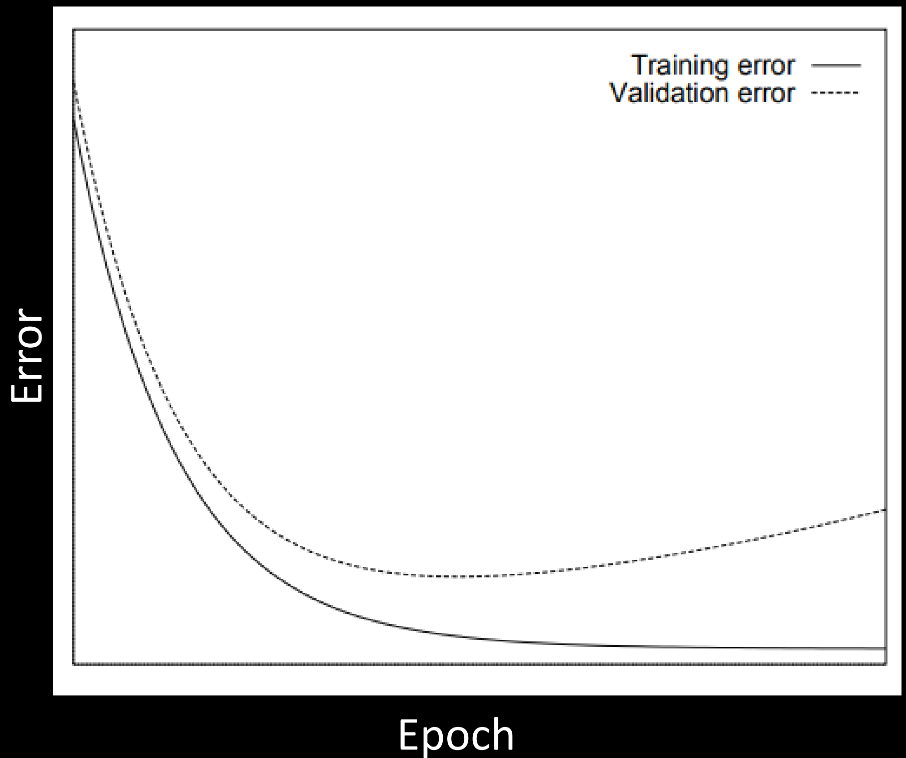
# Hidden Units – Rule of thumb

- HU = # Input (I) + # output(o)/2
- 1≤HU ≤ 2I

- Training data availability?
  - HU = N/30
- No regularization?
  - Avoid overfitting using more (?) training data
- Other design choices ?

http://faqs.cs.uu.nl/na-dir/ai-faq/neural-nets/part3.html

# Performance



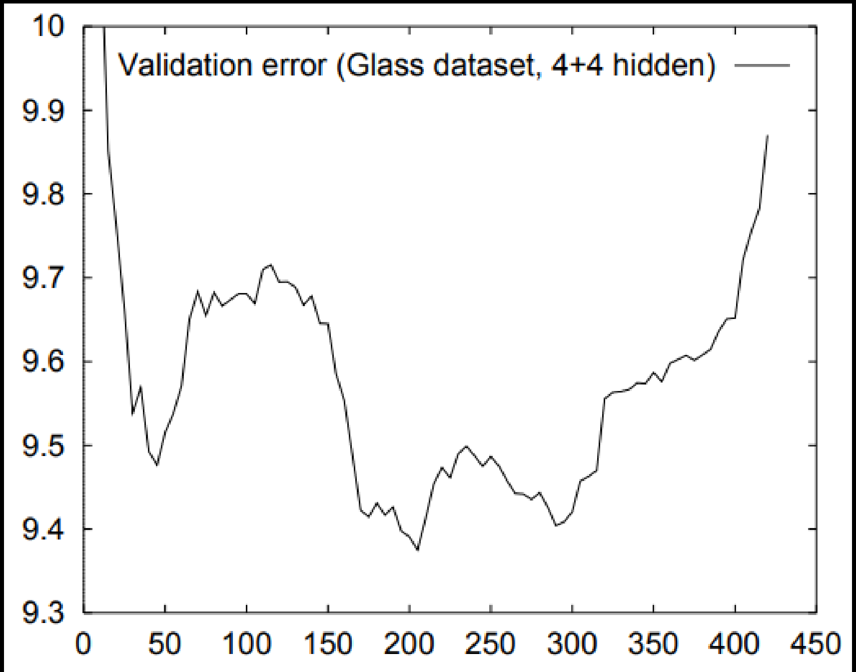


http://page.mi.fu-berlin.de/prechelt/Biblio/stop_tricks1997.pdf

https://pbs.twimg.com/media/B5NSIqZCMAAgIcw.png

# Reality is ugly

Validation Error




Epoch


http://page.mi.fu-berlin.de/prechelt/Biblio/stop_tricks1997.pdf

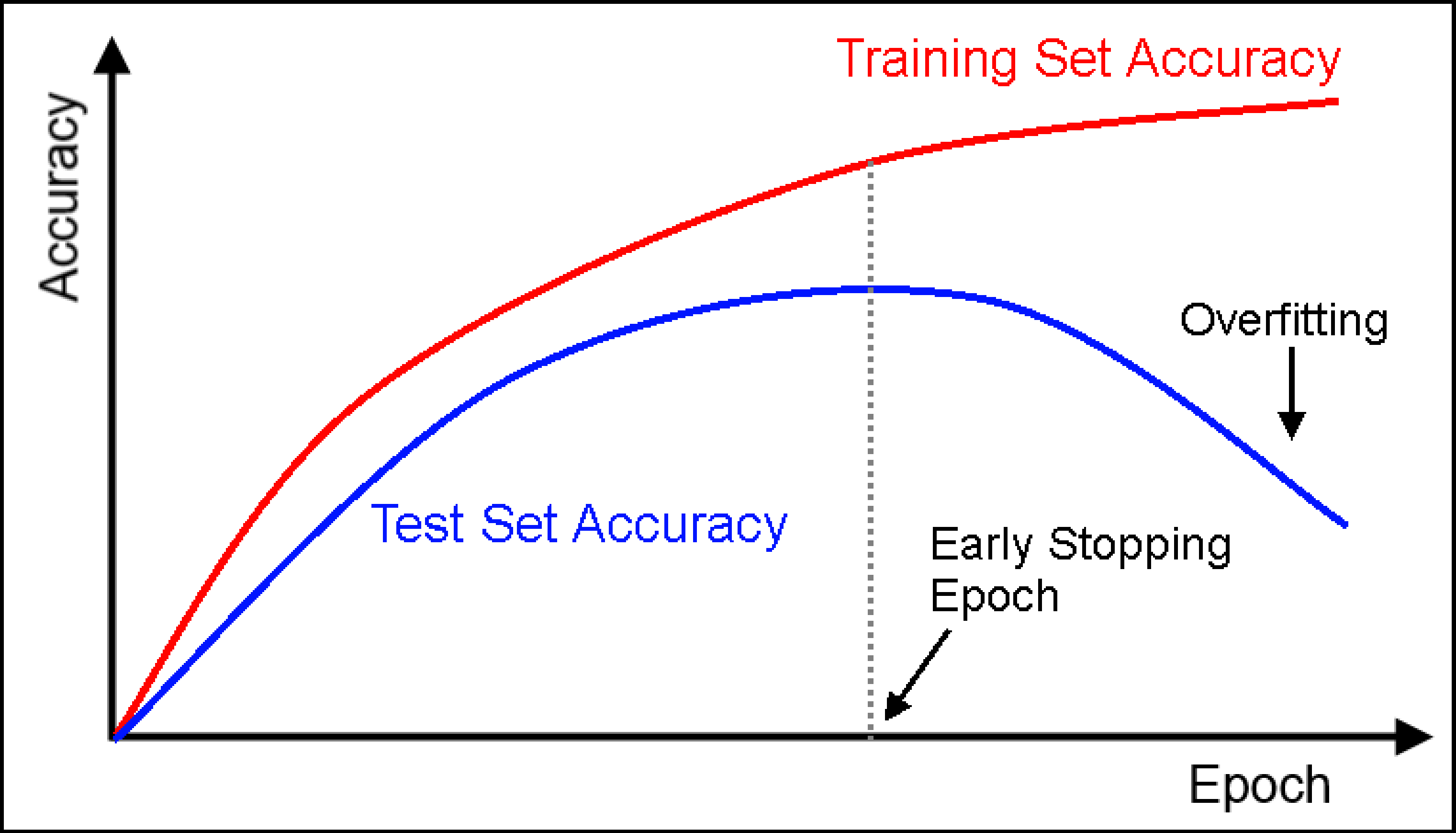
https://pbs.twimg.com/media/B5NSIqZCMAAgIcw.png

# Hyperparameter Tuning

http://saimg-a.akamaihd.net/saatchi/20877/art/1275154/578052-PLXFJCMJ-7.jpg

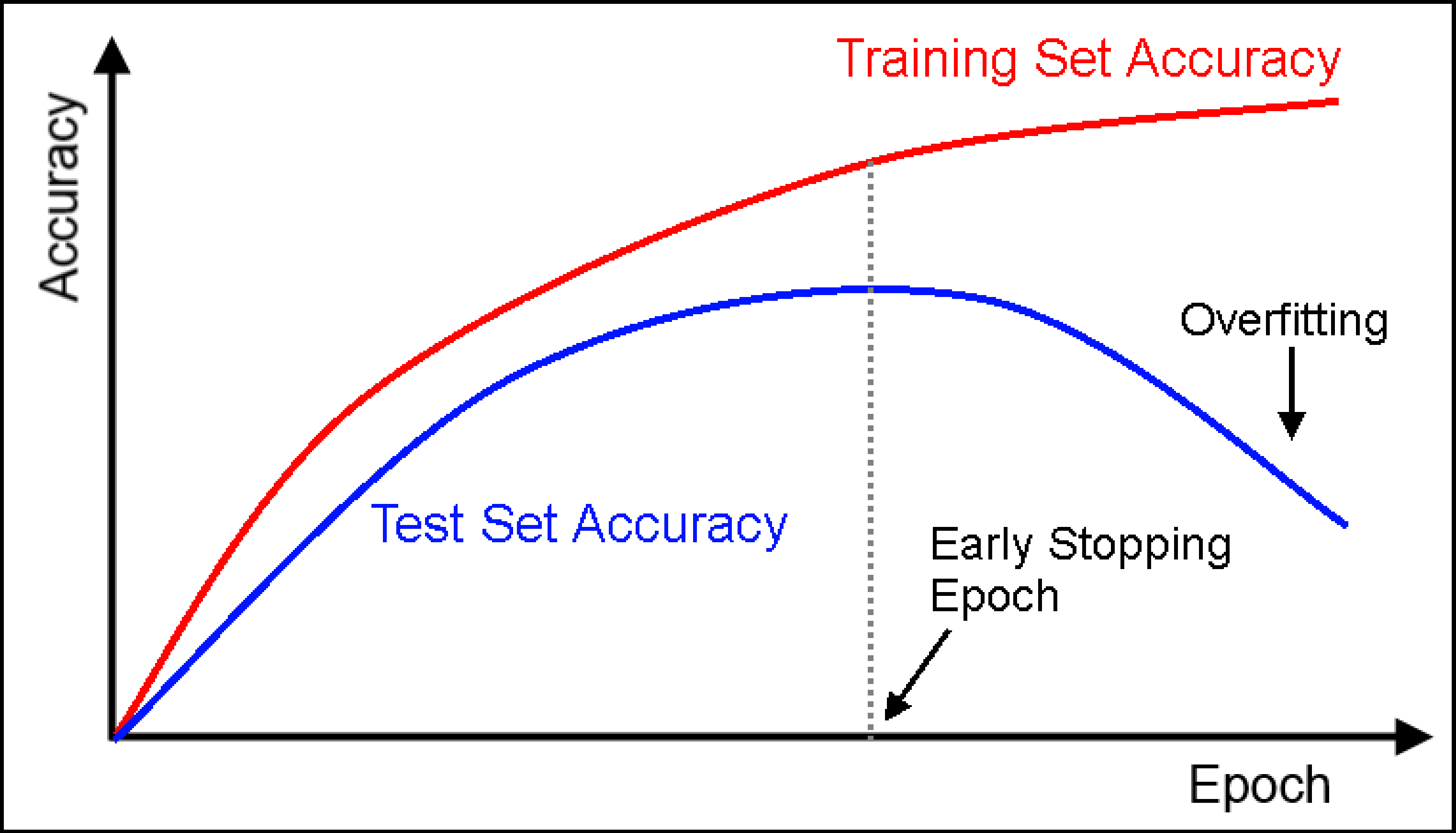
# So, what to tune?

## Performance




https://deeplearning4j.org/earlystopping

# Early stopping




https://deeplearning4j.org/earlystopping

# Early stopping

$E_{tr}(t)$ - Average training set error, after epoch 't'

$E_{va}(t)$ - Average validation set error, after epoch 't'

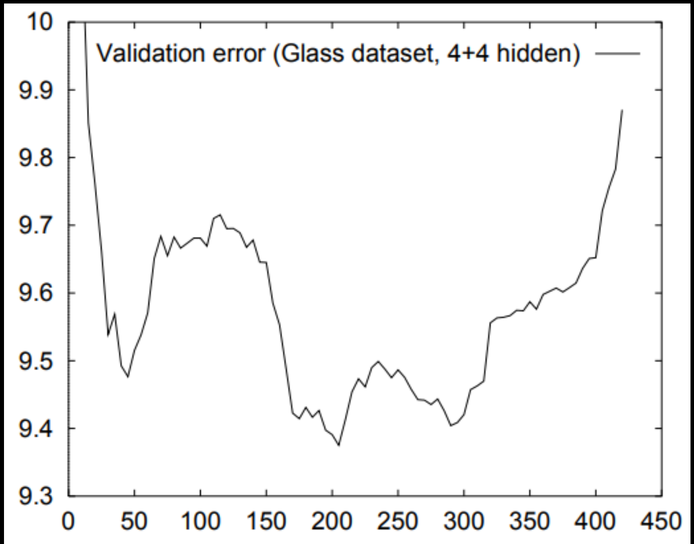
$$E_{opt}(t) := \min_{t' \le t} E_{va}(t')$$

Generalization loss

$$GL(t) = 100 \cdot \left( \frac{E_{va}(t)}{E_{opt}(t)} - 1 \right)$$

$GL_\alpha$ : stop after first epoch $t$ with $GL(t) > \alpha$

$$P_k(t) := 1000 \cdot \left( \frac{\sum_{t'=t-k+1}^{t} E_{tr}(t')}{k \cdot \min_{t'=t-k+1}^{t} E_{tr}(t')} - 1 \right)$$




Progress quotient

$PQ_\alpha$ : stop after first end-of-strip epoch $t$ with $\dfrac{GL(t)}{P_k(t)} > \alpha$

# Hyperparameter Tuning

- Manual
- Automatic optimization
  - Grid search
  - Random search
  - Model based optimization

# Manual tuning

- Learning rate, regularization parameter
- Approach
  - Explore: Bottom up
  - Step-size: Top Down
- Feedback
  - Validation error, cost function

http://neuralnetworksanddeeplearning.com/index.html

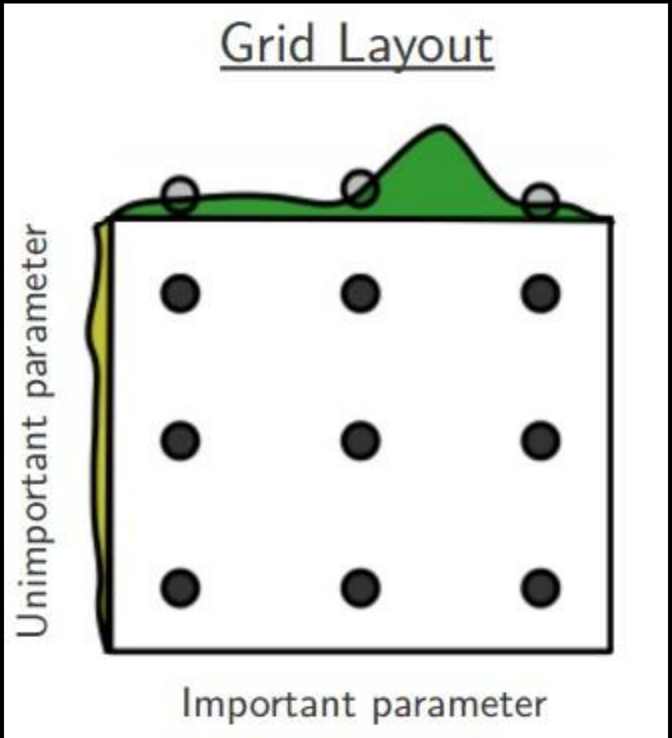
# Manual tuning

| Hyperparameter | Increases capacity when... | Reason | Caveats |
|---|---|---|---|
| Number of hidden units | increased | Increasing the number of hidden units increases the representational capacity of the model. | Increasing the number of hidden units increases both the time and memory cost of essentially every operation on the model. |
| Learning rate | tuned optimally | An improper learning rate, whether too high or too low, results in a model with low effective capacity due to optimization failure | |
| Convolution kernel width | increased | Increasing the kernel width increases the number of parameters in the model | A wider kernel results in a narrower output dimension, reducing model capacity unless you use implicit zero padding to reduce this effect. Wider kernels require more memory for parameter storage and increase runtime, but a narrower output reduces memory cost. |
| Implicit zero padding | increased | Adding implicit zeros before convolution keeps the representation size large | Increased time and memory cost of most operations. |
| Weight decay coefficient | decreased | Decreasing the weight decay coefficient frees the model parameters to become larger | |
| Dropout rate | decreased | Dropping units less often gives the units more opportunities to "conspire" with each other to fit the training set | |

http://timdettmers.com/2015/03/09/deep-learning-hardware-guide/

http://www.deeplearningbook.org/contents/guidelines.html

# Grid search

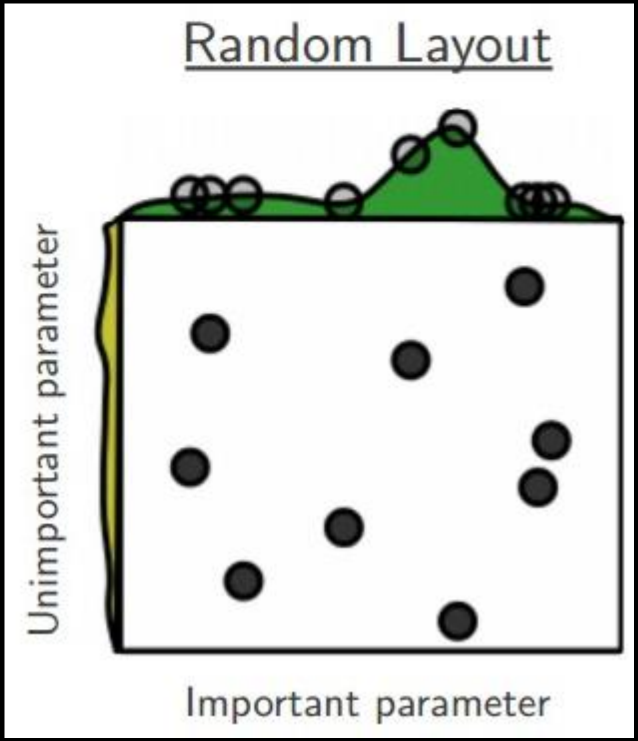



- Experience
- Discrete values
- Logarithmic scale
  - Learning rate
  - Hidden units
  - Regularization
- Scale top-down
- Slow convergence
- Multiple iterations
- Parallelization

http://cs231n.github.io/assets/nn3/gridsearchbad.jpeg

http://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

# Random search




- Non-discrete
- Marginal distribution for each parameter
- Eg. log_learning_rate ~ u(−1, −5)
- Fast covergence
- Time/cost efficient
- Ease of manual intervention to refine

http://cs231n.github.io/assets/nn3/gridsearchbad.jpeg

http://www.jmlr.org/papers/volume13/bergstra12a/bergstra12a.pdf

# Model-Based optimization

- Optimal hyperparameters
- Trade off between exploration and exploitation
- Open research field

# Bayesian optimization

- Prior over functions
  - Expressing assumptions of optimization function
- Acquisition function
  - Uses the model posterior to determine the next point of evaluation
- Cannot confidently recommend Bayesian optimization
- Not necessarily better than manual search

# Debugging

97% accuracy!!!
Time to check before you open a bottle of champagne!

https://media.tenor.com/images/44fbe7833375c0d0f7fce91f164dc76d/tenor.gif

# Debugging

- Check the outputs
  - Not just the performance measures
  - Images, audio, etc.
- Fit tiny datasets
  - Especially if you are building network from scratch
  - Make manual comparison of derivatives
- Check the reliability of software/packages

# Summary

- Manual:
  - Learning rate, regularization parameter
- Automatic optimization algorithms
- Random search is better than grid search
- Do not get over excited by DL!!!

# Thank you!

# References


- [1] Neural Networks and Deep Learning (Michael Nielsen)
- [2] Deep Learning (Ian Goodfellow, Yoshua Bengio and Aaron Courville)