Neural Networks: knobs and dials

Joris Mollinga

SURF

Parameters vs. Hyperparameters

Parameters: weights (W_1, W_2, ...) and biases (b_1, b_2, ...)

Parameters vs. Hyperparameters

Parameters: weights (W_1, W_2, ...) and biases (b_1, b_2, ...)

Hyperparameters related to network design

- number of hidden layers L
- number of hidden units n_1, n_2, …
- dropout (regularization)
- weight initialization
- activation function – tanh, ReLU, …

Parameters vs. Hyperparameters

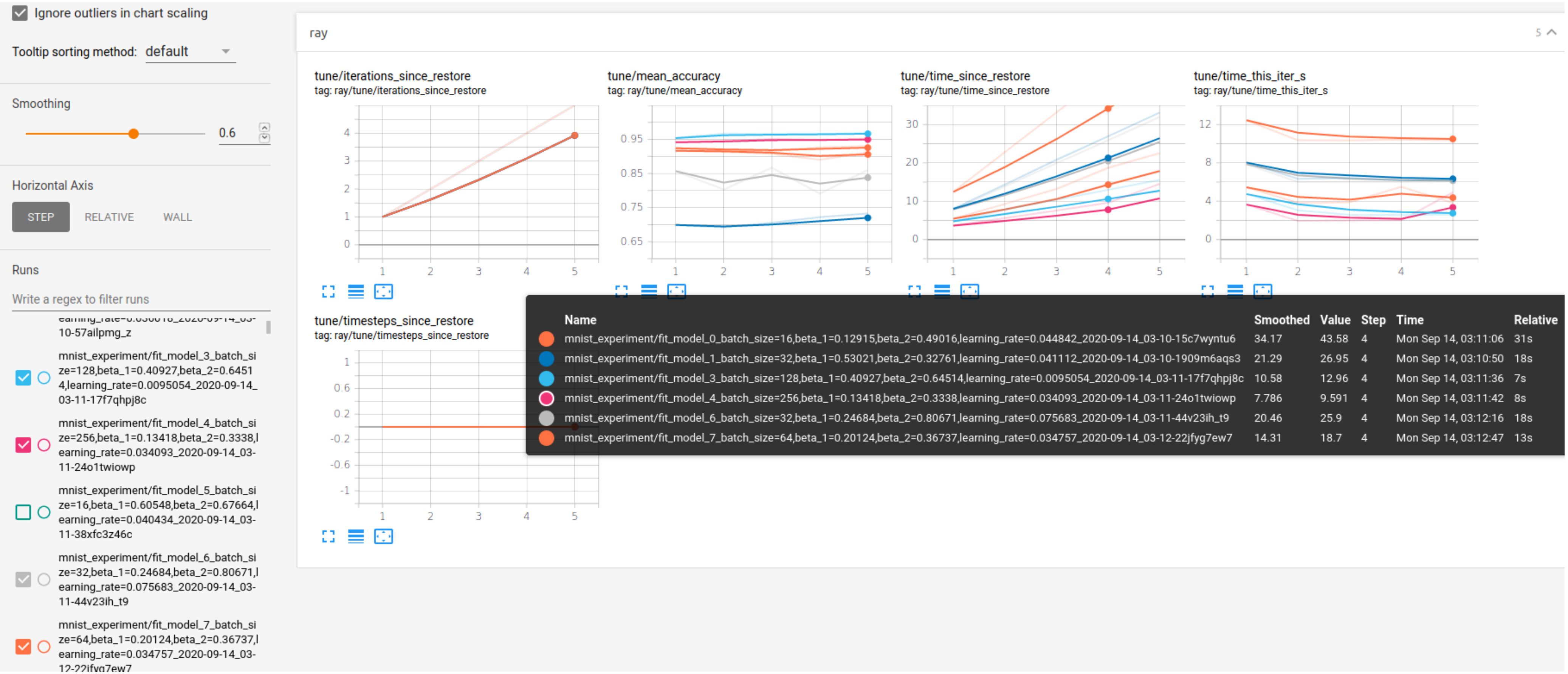
Parameters: weights (W_1, W_2, ...) and biases (b_1, b_2, ...)

Hyperparameters related to network design
- number of hidden layers L
- number of hidden units n_1, n_2, …
- dropout (regularization)
- weight initialization
- activation function – tanh, ReLU, …

Hyperparameters for the optimization process
- learning rate α
- number of iterations for gradient descent
- momentum
- minibatch size
- number of epochs

Empirical tuning of parameters – Building intuition

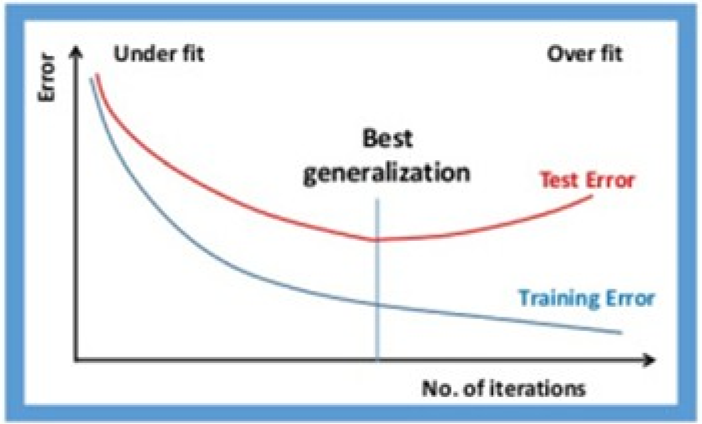
Idea → Experiment → Code and repeat



Understanding hyperparameters - Generalization

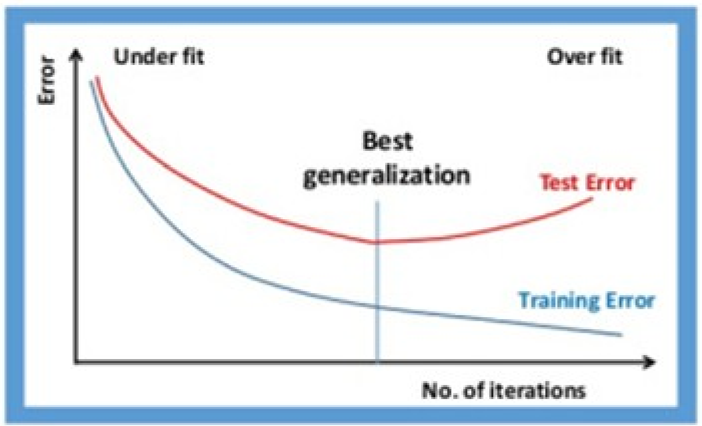
- ANNs generalize well despite having many more parameters than samples.

Generalization



Understanding hyperparameters - Generalization

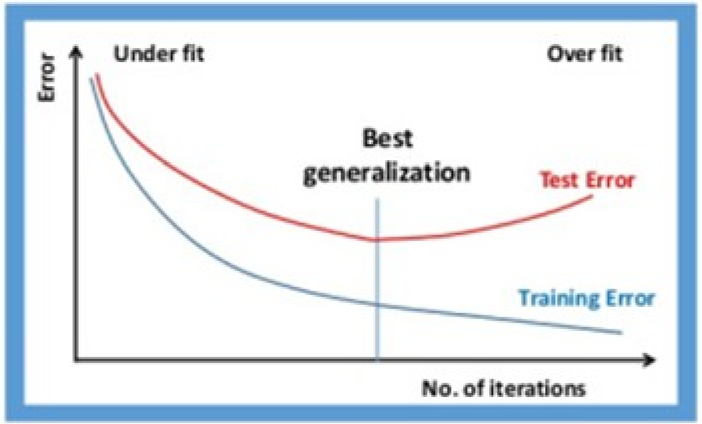
Generalization



- ANNs generalize well despite having many more parameters than samples.
- Test error vs Hidden units

Understanding hyperparameters - Generalization

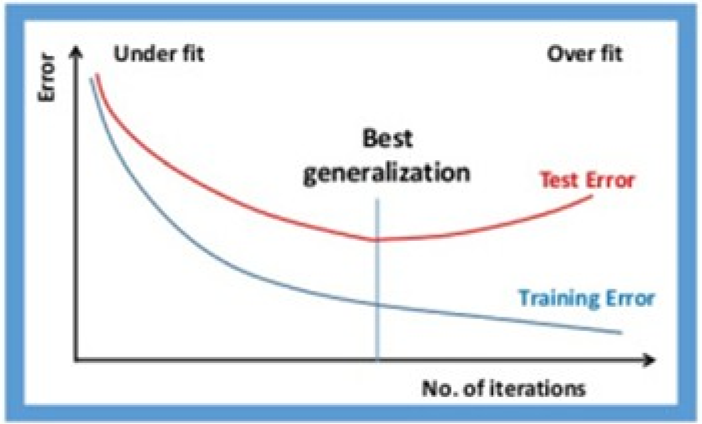
Generalization



- ANNs generalize well despite having many more parameters than samples.
- Test error vs Hidden units
- Especially interesting in distributed execution environments.

Understanding hyperparameters - Generalization

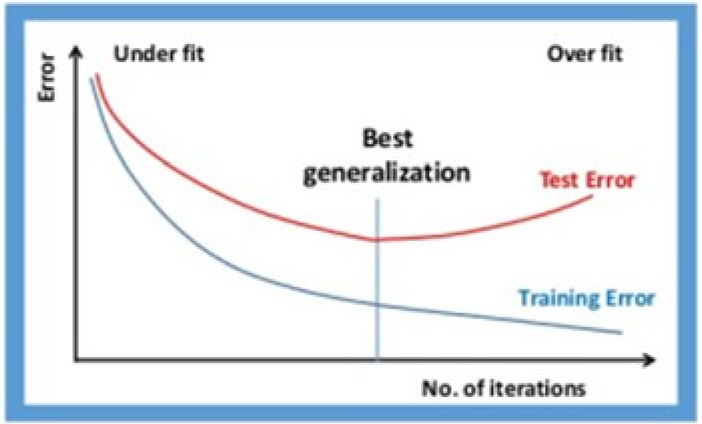
Generalization



- ANNs generalize well despite having many more parameters than samples.
- Test error vs Hidden units
- Especially interesting in distributed execution environments.
- Many hidden units within a layer with regularization techniques can increase accuracy.

Understanding hyperparameters - Generalization

Generalization



- ANNs generalize well despite having many more parameters than samples.
- Test error vs Hidden units
- Especially interesting in distributed execution environments.
- Many hidden units within a layer with regularization techniques can increase accuracy.
- Optimal stopping time depends on the signal to noise ratio → Low quality data requires longer training time

Understanding hyperparameters - Bias-Variance tradeoff

Understanding hyperparameters - Bias-Variance tradeoff

- **A bias** in a model appears when the algorithm is not flexible enough to generalize well from the data.

  Eg. : Linear parametric algorithms with low complexity like Naive Bayes tend to have a high bias.

Understanding hyperparameters - Bias-Variance tradeoff

- **A bias** in a model appears when the algorithm is not flexible enough to generalize well from the data.

  Eg. : Linear parametric algorithms with low complexity like Naive Bayes tend to have a high bias.

- **Variance** occurs in the model when the algorithm is sensitive and highly flexible to the training data.

  Eg.: Non-Linear non-parametric algorithms with high complexity such as Neural Networks tend to have high variance.

Understanding hyperparameters - Bias-Variance tradeoff

- **A bias** in a model appears when the algorithm is not flexible enough to generalize well from the data.

  Eg. : Linear parametric algorithms with low complexity like Naive Bayes tend to have a high bias.

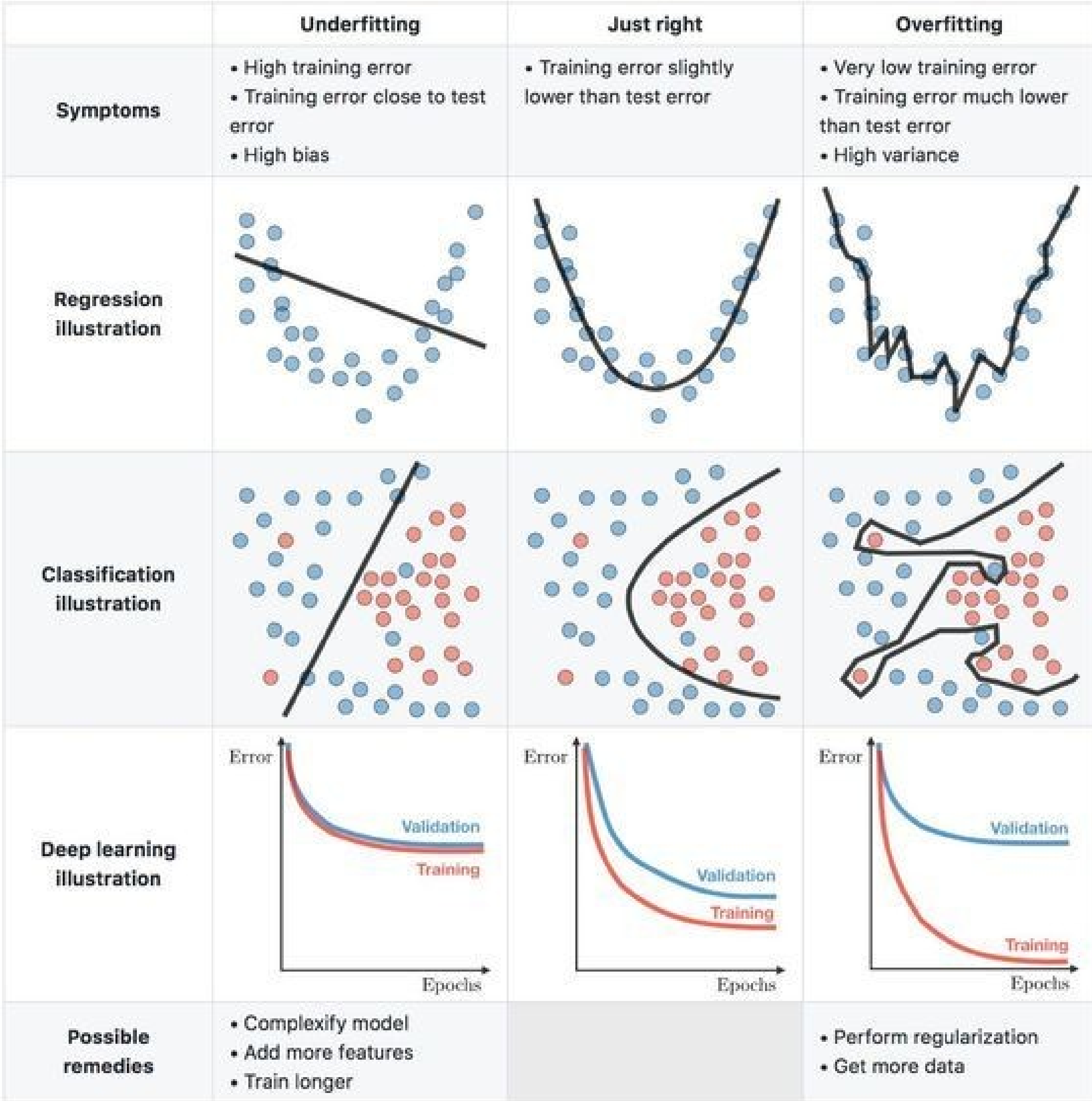
- **Variance** occurs in the model when the algorithm is sensitive and highly flexible to the training data.

  Eg.: Non-Linear non-parametric algorithms with high complexity such as Neural Networks tend to have high variance.

- There are various ways to find the balance of bias and variance for each algorithm family by using methods such as **regularization, pruning**, etc.

Understanding hyperparameters - Overfitting and Underfitting

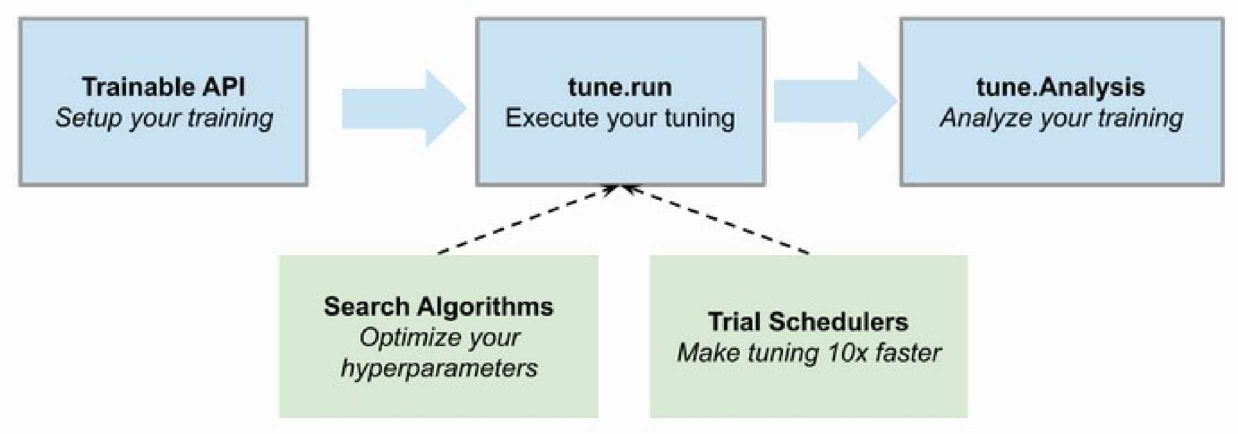
- Overfitting → bad generalization
- Smaller number of units may cause underfitting.
- Dropout is regularization technique to avoid overfitting (increase the validation accuracy) thus increasing the generalizing power. You are likely to get better performance when dropout is used on a larger network, giving the model more of an opportunity to learn independent representations.



	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration	Error / Validation / Training / Epochs	Error / Validation / Training / Epochs	Error / Validation / Training / Epochs
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

CS 229 - Machine Learning Tips and Tricks Cheatsheet

Tuning parameters with automated tools – Ray Tune



Advanced trial schedulers

- ASHA
- Median Stopping Rule
- HyperBand
- BOHB
- Population Based Training

Many optimization algorithms

- Bayesian Optimization
- Bayesian Opt/HyperBand
- NevergradSearch
- Gradient-free Optimization
- SigOptSearch
- ...

Learning rate

- The Learning Rate is a parameter of the Gradient Descent optimizer.
- Controls the change of weights for the network to the loss of gradient.

Learning rate

- The Learning Rate is a parameter of the Gradient Descent optimizer.
- Controls the change of weights for the network to the loss of gradient.
- Model training will progress very slowly for low learning rates as it would be making tiny adjustments to the weights in the network and it could also lead to overfitting.
- High learning rates tend to leaps bouncing around chaotically, not leading to the optimization of the objective function and missing the local optima making the training diverge. But having a large learning rate helps in regularizing the model well to capture the true signal.

Learning rate

- The Learning Rate is a parameter of the Gradient Descent optimizer.
- Controls the change of weights for the network to the loss of gradient.
- Model training will progress very slowly for low learning rates as it would be making tiny adjustments to the weights in the network and it could also lead to overfitting.
- High learning rates tend to leaps bouncing around chaotically, not leading to the optimization of the objective function and missing the local optima making the training diverge. But having a large learning rate helps in regularizing the model well to capture the true signal.
- There are many choices of schedulers.
- The interaction between batch size and learning rate is very important for certain distribution schemes.
- Adaptive learning rate algorithms are prevalent nowadays.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.
- Batch size affects the training time.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.
- Batch size affects the training time.
- In general we want to maximize efficiency and utilization of memory.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.
- Batch size affects the training time.
- In general we want to maximize efficiency and utilization of memory.
- Rather modifying the batch size rather than changing the learning rate.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.
- Batch size affects the training time.
- In general we want to maximize efficiency and utilization of memory.
- Rather modifying the batch size rather than changing the learning rate.
- Larger batches enable using a large learning rate.

Minibatch size

- Minibatch Size = number of training samples propagated through the network.
- Our goal to obtain the maximum performance by minimizing the computational time required.
- Batch size affects the training time.
- In general we want to maximize efficiency and utilization of memory.
- Rather modifying the batch size rather than changing the learning rate.
- Larger batches enable using a large learning rate.
- Larger batch sizes tend to have low early losses while training.
- Final loss values are low when the batch size is reduced.

THANK YOU FOR YOUR ATTENTION

www.prace-ri.eu