Accelerating Access to Life-Saving Treatments to Patients

pumas AI

Machine Learning and Neural Networks

DeepPumas Workshop

Prepared by: Andreu Vall and Mohamed Tarek

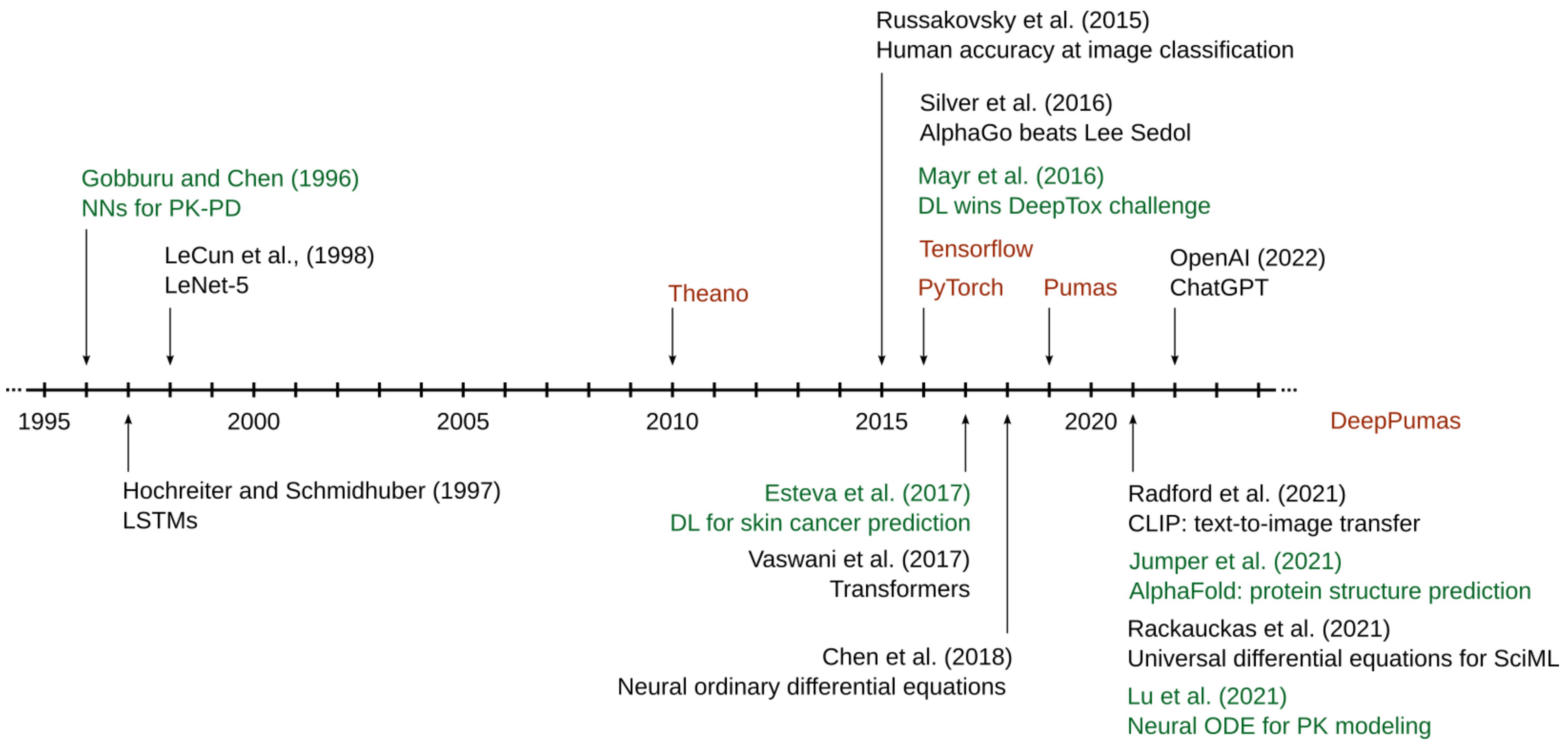
Pumas-AI Inc

Jun 24, 2024

DeepPumas

Machine Learning,
Deep Learning
Artificial Intelligence

Context and Evolution

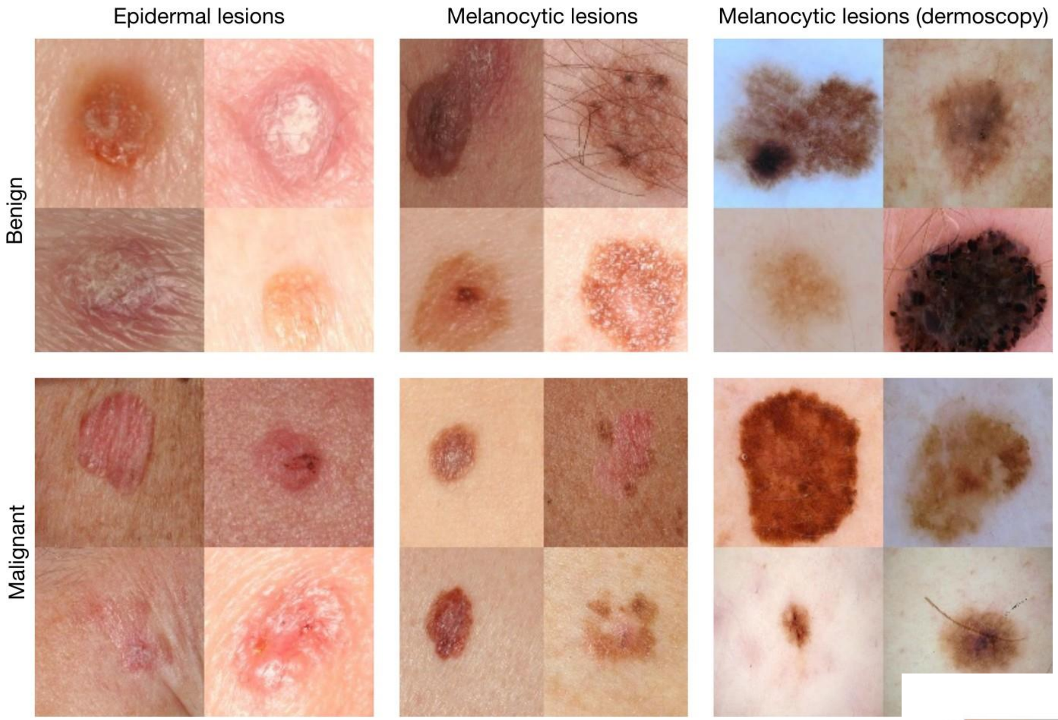


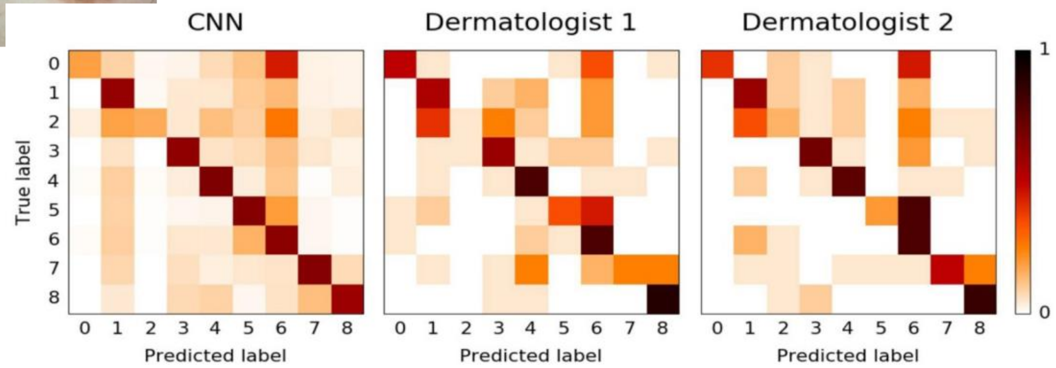
Toxicity Prediction with Deep Learning

	AVG	NR	SR	AhR	AR	AR-LBD	ARE	Aromatase	ATAD5	ER	ER-LBD	HSE	MMP	p53	PPAR.g
our method	**0.846**	**0.826**	**0.858**	**0.928**	0.807	**0.879**	**0.840**	0.834	0.793	**0.810**	0.814	**0.865**	0.942	0.862	**0.861**
AMAZIZ	0.838	0.816	0.854	0.913	0.770	0.846	0.805	0.819	**0.828**	0.806	0.806	0.842	**0.950**	0.843	0.830
dmlab	0.824	0.811	0.850	0.781	**0.828**	0.819	0.768	**0.838**	0.800	0.766	0.772	0.855	0.946	**0.880**	0.831
T	0.823	0.798	0.842	0.913	0.676	0.848	0.801	0.825	0.814	0.784	0.805	0.811	0.937	0.847	0.822
microsomes	0.810	0.785	0.814	0.901	–	–	0.804	–	0.812	0.785	**0.827**	–	–	0.826	0.717
filipsPL	0.798	0.765	0.817	0.893	0.736	0.743	0.758	0.776	–	0.771	–	0.766	0.928	0.815	–
Charite	0.785	0.750	0.811	0.896	0.688	0.789	0.739	0.781	0.751	0.707	0.798	0.852	0.880	0.834	0.700
RCC	0.772	0.751	0.781	0.872	0.763	0.747	0.761	0.792	0.673	0.781	0.762	0.755	0.920	0.795	0.637
frozenarm	0.771	0.759	0.768	0.865	0.744	0.722	0.700	0.740	0.726	0.745	0.790	0.752	0.859	0.803	0.803
ToxFit	0.763	0.753	0.756	0.862	0.744	0.757	0.697	0.738	0.729	0.729	0.752	0.689	0.862	0.803	0.791
CGL	0.759	0.720	0.791	0.866	0.742	0.566	0.747	0.749	0.737	0.759	0.727	0.775	0.880	0.817	0.738
SuperTox	0.743	0.682	0.768	0.854	–	0.560	0.711	0.742	–	–	–	–	0.862	0.732	–
kibutz	0.741	0.731	0.731	0.865	0.750	0.694	0.708	0.729	0.737	0.757	0.779	0.587	0.838	0.787	0.666
MML	0.734	0.700	0.753	0.871	0.693	0.660	0.701	0.709	0.749	0.750	0.710	0.647	0.854	0.815	0.645
NCI	0.717	0.651	0.791	0.812	0.628	0.592	0.783	0.698	0.714	0.483	0.703	0.858	0.851	0.747	0.736
VIF	0.708	0.702	0.692	0.827	0.797	0.610	0.636	0.671	0.656	0.732	0.735	0.723	0.796	0.648	0.666
Toxic Avg	0.644	0.659	0.607	0.715	0.721	0.611	0.633	0.671	0.593	0.646	0.640	0.465	0.732	0.614	0.682
Swamidass	0.576	0.596	0.593	0.353	0.571	0.748	0.372	0.274	0.391	0.680	0.738	0.711	0.828	0.661	0.585

Mayr et al. "DeepTox: Toxicity Prediction Using Deep Learning." (Front. in Env. Sci., 2016)

Image-based Diagnostic with Deep Learning





Esteva et al. “Dermatologist-Level Classification of Skin Cancer with Deep Neural Networks.” (Nature, 2017)

Broad types of machine learning

- Supervised machine learning
  - Prediction
  - Classification
- Unsupervised machine learning
  - Generative models
  - Feature extraction
  - Clustering
- Semi-supervised machine learning
  - Has components of both supervised and unsupervised learning
  - Examples:
    - Semi-supervised conditional generative models
    - Feature extraction followed by supervised ML
- Reinforcement learning
  - Machine learning for sequential decision making in a stochastic environment
  - Not in scope for this workshop

General notes

- Not all machine learning algorithms use neural networks
- Not all machine learning algorithms use probabilistic models, e.g.
  - K-nearest neighbor algorithm
  - Decision trees
  - Hierarchical clustering
  - DBSCAN
- Probabilistic machine learning borrows many concepts from computational (Bayesian) statistics
- Machine learning builds on many fields including
  - Statistics
  - Optimization
  - Database management
  - Image and signal processing
  - Software engineering
- Related terms: data mining, data science, artificial intelligence, and more

Probabilistic supervised machine learning

- Data type
  - Labelled data $(\boldsymbol{x}_i, \boldsymbol{y}_i)$ for $i \in 1 \ldots N$
- Tasks
  - Prediction
  - Classification
- Model type
  - Conditional model of $\boldsymbol{y} \mid \boldsymbol{x}$
  - Also known as discriminative models

Probabilistic supervised machine learning examples

- Prediction model examples
  - $\boldsymbol{y} \sim \text{Normal}(\boldsymbol{f}(\boldsymbol{x}), \sigma^2 \cdot \boldsymbol{I})$
  - $\boldsymbol{y} \sim \text{LogNormal}(\boldsymbol{f}(\boldsymbol{x}), \sigma^2 \cdot \boldsymbol{I})$
- Classification model examples
  - $y \sim \text{Bernoulli}\left(\text{logistic}(f(\boldsymbol{x}))\right)$
  - $y \sim \text{Categorical}\left(\textbf{softmax}(\boldsymbol{f}(\boldsymbol{x}))\right)$

$$\text{logistic}(x) = \frac{1}{1 + e^{-x}}$$

$$\textbf{softmax}(\boldsymbol{x}) = \left[\frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \text{ for } i \; in \; 1 \ldots K\right]$$

Probabilistic unsupervised machine learning

- Data type
  - Un-labelled data $\boldsymbol{y_i}$ for $i \in 1 \dots N$
- Tasks
  - Learning complex data distributions, aka generative modelling
  - Dimension reduction
  - Feature extraction / embedding
  - Clustering
- Model type
  - Generative (simulation) model for $\boldsymbol{y}$
  - Has latent random variables $\boldsymbol{z}$, e.g. $\boldsymbol{z} \sim \text{Normal}(0, \boldsymbol{I})$
  - Models the distribution of $\boldsymbol{y} \mid \boldsymbol{z}$ explicitly, e.g. $\boldsymbol{y} \sim \text{Normal}(\boldsymbol{f}(\boldsymbol{z}), \sigma^2 \cdot \boldsymbol{I})$
  - (Approximately) models the inverse distribution $\boldsymbol{z} \mid \boldsymbol{y}$ either explicitly or implicitly

Probabilistic unsupervised machine learning examples

- Probabilistic principal component analysis

$$\boldsymbol{y} \sim \text{Normal}(\boldsymbol{A} \cdot \boldsymbol{z} + \boldsymbol{b}, \sigma^2 \cdot \boldsymbol{I})$$

- Variational autoencoders (VAE)

$$\boldsymbol{y} \sim \text{Normal}(\mathbf{NN}(\boldsymbol{z}), \sigma^2 \cdot \boldsymbol{I})$$

- Generative adversarial networks (GANs)

$$\boldsymbol{y} = \mathbf{NN}(\boldsymbol{z})$$

- Clustering (ϕ is a K dimensional vector that sums up to 1 for K clusters)

$$z \sim \text{Categorical}(\boldsymbol{\phi})$$
$$\boldsymbol{y} \sim \text{Normal}(\boldsymbol{\mu}_z, \boldsymbol{\Sigma}_z)$$

DeepPumas

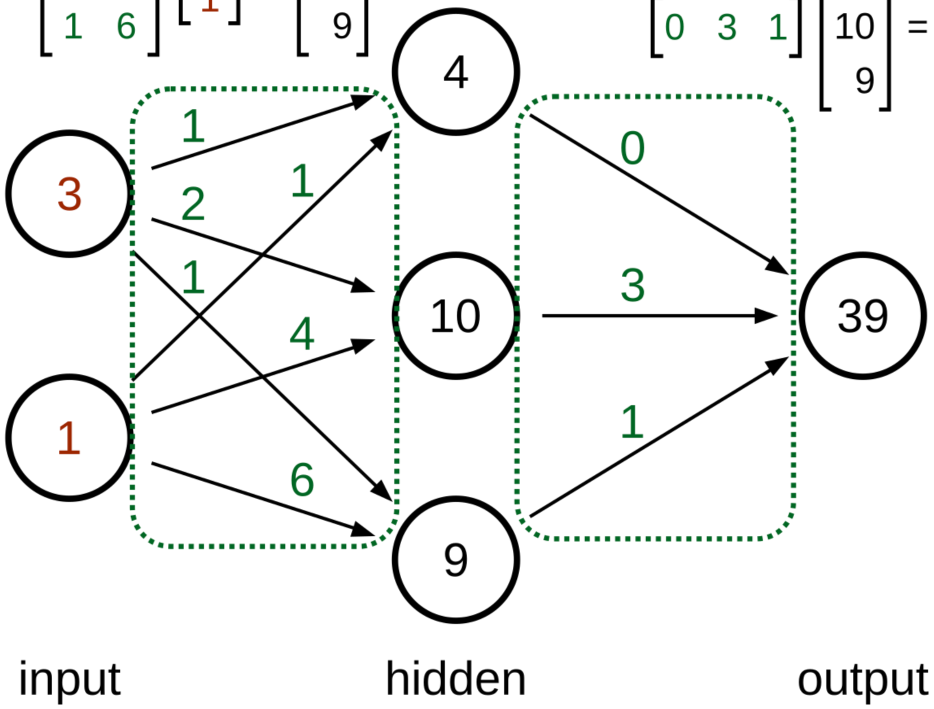
Fundamental Concepts of Machine Learning and Neural Networks

Neural Networks

f = g = identity

$$\begin{bmatrix} 1 & 1 \\ 2 & 4 \\ 1 & 6 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \end{bmatrix} = \begin{bmatrix} 4 \\ 10 \\ 9 \end{bmatrix}$$

$$\begin{bmatrix} 0 & 3 & 1 \end{bmatrix} \begin{bmatrix} 4 \\ 10 \\ 9 \end{bmatrix} = \begin{bmatrix} 39 \end{bmatrix}$$



Training

- Find network parameters $\mathbf{W}$ that minimize a suitable cost function

$$R_{\text{emp}}\left(\mathbf{W},\mathbf{X},\mathbf{Y}\right)=\frac{1}{N}\sum_{n=1}^{N}L\left(y^{n},\hat{y}^{n}(\mathbf{x}^{n};\mathbf{W})\right)$$

- Iteratively refine the parameters by stochastic gradient descent

$$\mathbf{W}^{\text{new}}=\mathbf{W}^{\text{old}}-\eta\nabla_{\mathbf{W}}R_{\text{emp}}\left(\mathbf{W},\mathbf{X},\mathbf{Y}\right)$$

- Gradient computed by backpropagation (Rumelhart et al., 1986)

Control on Withheld Data

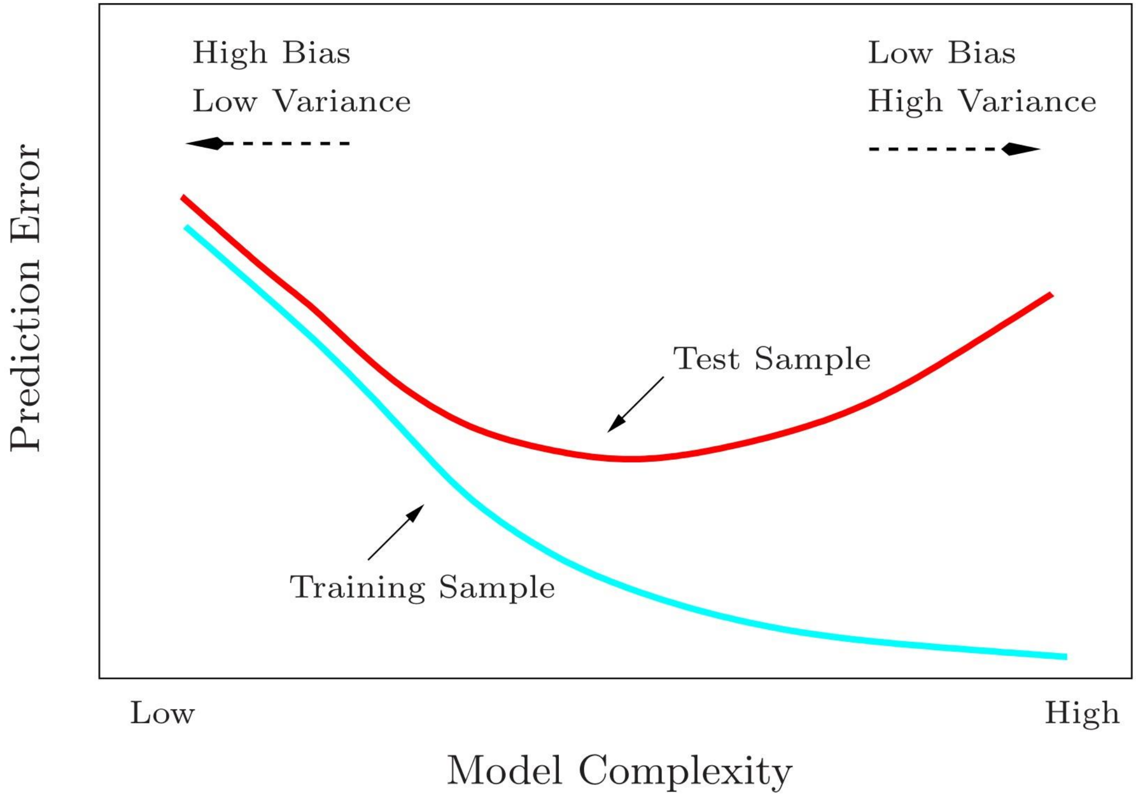


FIGURE 2.11. *Test and training error as a function of model complexity.*

Figure by Hastie et al., 2008

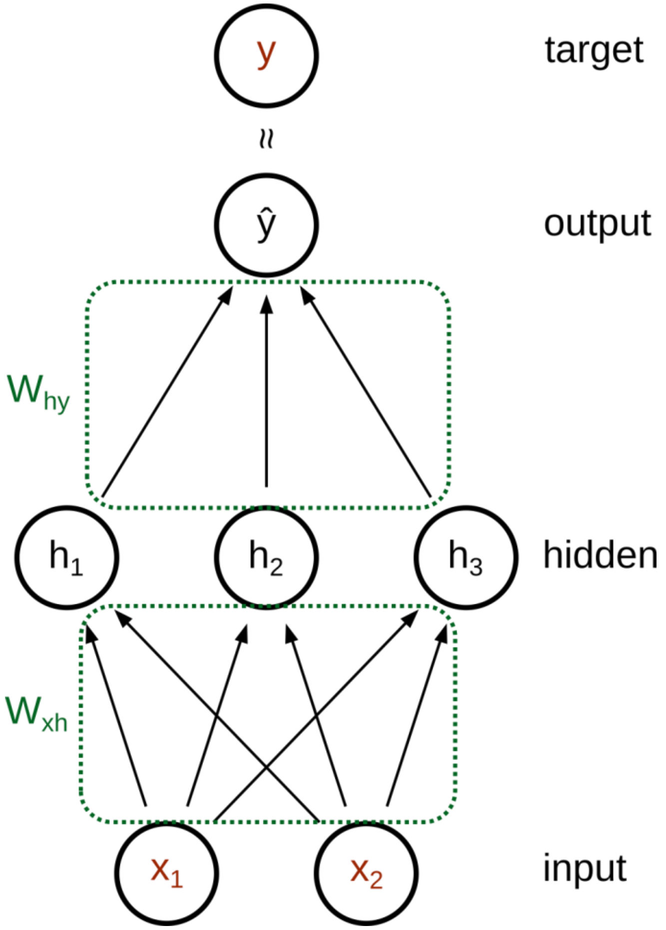
DeepPumas

Practical Guide to Training Neural Networks

Feed-Forward Networks

- The basic interface is quite given by the task
  - Number of input units
  - Number of output units
  - Activation function for the output layer

- The *architecture* must be selected carefully
  - Number of hidden layers
  - Number of units per hidden layer
  - Activation function for the hidden layers
  - Parameters of the optimization algorithm



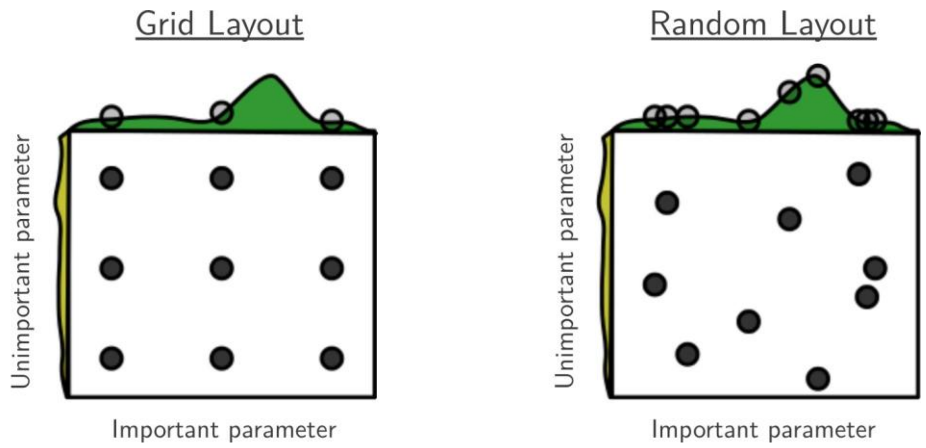
Activation Functions

- At the output layer we typically use
  - the identity function for regression tasks,
  - the sigmoid function for binary classification tasks,
  - the softmax function for multiclass classification tasks, and
  - the softplus function to ensure non-negativity.

- At the hidden layers,
  - traditional activations are the sigmoid and tanh functions,
  - but deeper networks will suffer from vanishing gradients.
  - In this case, experiment with the ReLU and SELU functions.

Architecture Selection

- Typically based on empirical investigation and assessment on withheld data
- Initial exploration to get a feel of reasonable network dimensions
- Hyperparameter search
  - Design a grid of hyperparameters based on the previous exploration
  - In case of a tight budget, pay attention to the learning rate
  - Exhaustive, random, Bayesian search



Bergstra and Bengio "Random search for hyper-parameter optimization." (JMLR, 2012)

Underfitting and Overfitting

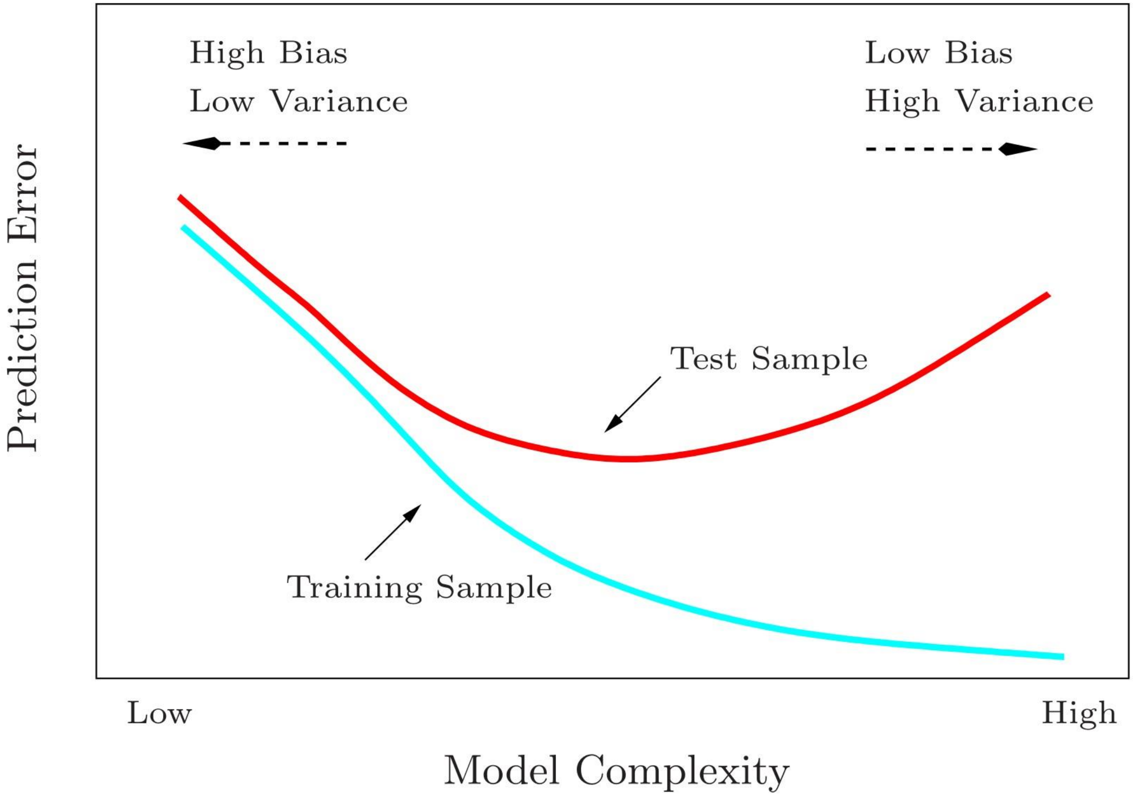


FIGURE 2.11. *Test and training error as a function of model complexity.*

Figure by Hastie et al., 2008

Strategies to Prevent Overfitting

- Early stopping based on withheld data
- Most careful data splitting to avoid data leakage
- Networks as large as possible, but not larger
- Larger and more diverse training datasets
- Regularization
    - Penalization of the cost function
    - Dropout (Srivastava et al., 2014)
    - Data *augmentation*

Penalization of the Cost Function

- Cost function

$$\arg\min_{\mathbf{W}} \frac{1}{N} \sum_{n=1}^{N} L\left(y^n, \hat{y}^n(\mathbf{x}^n; \mathbf{W})\right)$$

- Penalized cost function reduces expressiveness

$$\arg\min_{\mathbf{W}} \frac{1}{N} \sum_{n=1}^{N} L\left(y^n, \hat{y}^n(\mathbf{x}^n; \mathbf{W})\right) + \lambda \|\mathbf{W}\|_p$$

- Typically using $L1$ or $L2$ norm

Regularization as MAP estimation

$$\widehat{\mathbf{W}}_{\mathrm{ML}} = \arg\max_{\mathbf{W}} \mathcal{L}(\mathbf{W} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) = \arg\max_{\mathbf{W}} p(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \mathbf{W}) =$$
$$= \arg\max_{\mathbf{W}} \prod_{i=1}^{N} p(\mathbf{x}_i \mid \mathbf{W}) = \arg\max_{\mathbf{W}} \sum_{i=1}^{N} \log p(\mathbf{x}_i \mid \mathbf{W})$$

$$\widehat{\mathbf{W}}_{\mathrm{MAP}} = \arg\max_{\mathbf{W}} p(\mathbf{W} \mid \mathbf{x}_1, \ldots, \mathbf{x}_N) = \arg\max_{\mathbf{W}} p(\mathbf{x}_1, \ldots, \mathbf{x}_N \mid \mathbf{W}) p(\mathbf{W})$$
$$= \arg\max_{\mathbf{W}} \prod_{i=1}^{N} p(\mathbf{x}_i \mid \mathbf{W}) p(\mathbf{W}) = \arg\max_{\mathbf{W}} \sum_{i=1}^{N} \log p(\mathbf{x}_i \mid \mathbf{W}) + \log p(\mathbf{W})$$

DeepPumas

Hands-on session: Fitting, Overfitting and Regularizing Neural Networks

Conditional generative models

Generative models

- Definitions
  - $\boldsymbol{z}$: latent variables of dimension d
  - $\boldsymbol{y}$: observed response/data
  - $\boldsymbol{y_g}$: generated/simulated/synthetic response/data
- Model

$$\boldsymbol{y_g} = \boldsymbol{f}(\boldsymbol{z}) + \boldsymbol{\epsilon}$$
$$\boldsymbol{z} \sim \text{Normal}(0, \boldsymbol{I}_{d \times d})$$
$$\epsilon_i \sim \text{Normal}(0, \sigma^2)$$

- Objective: choose $\boldsymbol{f}$ such that the distribution of $\boldsymbol{y_g}$ is close to the distribution of the observed data $\boldsymbol{y}$

Conditional generative models

- Definitions
  - $\boldsymbol{z}$: latent variables of dimension d
  - $\boldsymbol{x}$: observed covariates
  - $\boldsymbol{y}$: observed response
  - $\boldsymbol{y_g}$: generated/simulated/synthetic response
- Model

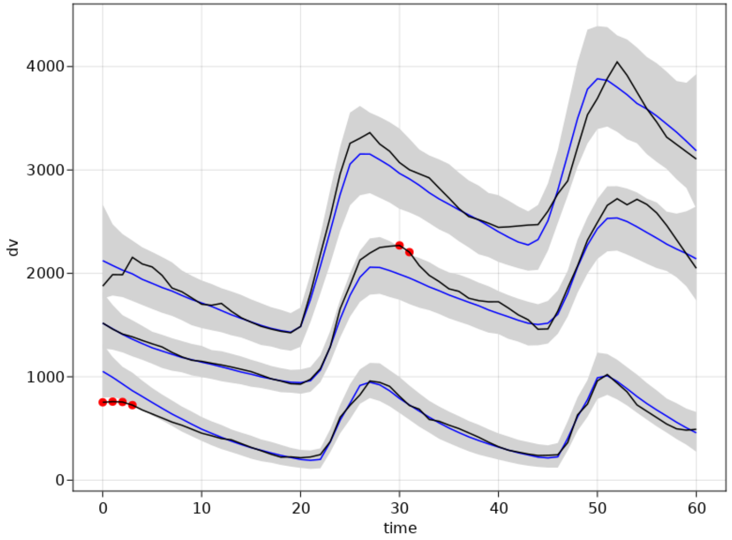
$$\boldsymbol{y_g} = \boldsymbol{f}(\boldsymbol{z}, \boldsymbol{x}) + \boldsymbol{\epsilon}$$
$$\boldsymbol{z} \sim \text{Normal}(0, \boldsymbol{I}_{d \times d})$$
$$\epsilon_i \sim \text{Normal}(0, \sigma^2)$$

- Objective: choose $\boldsymbol{f}$ such that the conditional distribution of $\boldsymbol{y_g} \mid \boldsymbol{x}$ is close to the conditional distribution of the observed data $\boldsymbol{y} \mid \boldsymbol{x}$

Population PK is machine learning!

- Definitions
  - $\boldsymbol{\eta}$: latent variables of dimension d and covariance matrix $\boldsymbol{\Omega}$
  - $\boldsymbol{x}$: observed covariates
  - $\mathbf{dv}$: observed response
  - $\mathbf{dv_g}$: generated/simulated/synthetic response
- Model

$$\mathbf{dv_g} = \boldsymbol{f_\theta}(\boldsymbol{\eta}, \boldsymbol{x}) + \boldsymbol{\epsilon}$$
$$\boldsymbol{\eta} \sim \text{Normal}(0, \boldsymbol{\Omega})$$
$$\epsilon_i \sim \text{Normal}(0, \sigma^2)$$



- Objective: choose $\boldsymbol{f_\theta}$ such that the conditional distribution of $\mathbf{dv_g} \mid \boldsymbol{x}$ is close to the conditional distribution of the observed data $\mathbf{dv} \mid \boldsymbol{x}$