Network training using gradient descent

Modify the weight values

to obtain better predictions

Network training using gradient descent

We need a way to measure the difference

between the predictions and the true species

(the cross-entropy error)

Network training using gradient descent

And our goal is to find the weights

that minimizes that error function

(using gradient descent)

Notation

The matrices $W^{(1)}$ and $W^{(2)}$

are combined as W

Network training using gradient descent

row	features				true species			predictions		
n	S.L.	S.W.	P.L.	P.W.	t_{n1}	t_{n2}	t_{n3}	y_{n1}	y_{n2}	y_{n3}
4	4.6	3.1	1.5	0.2	1	0	0	1	0	0
5	5.0	3.6	1.4	0.2	1	0	0	0.94	0.05	0.01

The cross-entropy error/loss function for the *n*th row:

$$E_n(W) = -\sum_{k=1}^{3} \overbrace{t_{nk}}^{true\ \ species} \cdot \log[\overbrace{y_{nk}(W)}^{prediction}]$$

Network training using gradient descent

row	features				true species			predictions		
n	S.L.	S.W.	P.L.	P.W.	t_{n1}	t_{n2}	t_{n3}	y_{n1}	y_{n2}	y_{n3}
4	4.6	3.1	1.5	0.2	1	0	0	1	0	0
5	5.0	3.6	1.4	0.2	1	0	0	0.94	0.05	0.01

The cross-entropy error/loss function for the *n*th row:

$$E_n(W) = -\sum_{k=1}^{3} \overbrace{t_{nk}}^{\textit{true species}} \cdot \log[\overbrace{y_{nk}(W)}^{\textit{prediction}}]$$

Example:

$$E_4(W) = -t_{n1} \cdot \log[y_{n1}(W)] = -\log(1) = 0$$

Network training using gradient descent

row	features				true species			predictions		
n	S.L.	S.W.	P.L.	P.W.	t_{n1}	t_{n2}	t_{n3}	y_{n1}	y_{n2}	y_{n3}
4	4.6	3.1	1.5	0.2	1	0	0	1	0	0
5	5.0	3.6	1.4	0.2	1	0	0	0.94	0.05	0.01

The cross-entropy error/loss function for the *n*th row:

$$E_n(W) = -\sum_{k=1}^{3} \overbrace{t_{nk}}^{\textit{true species}} \cdot \log[\overbrace{y_{nk}(W)}^{\textit{prediction}}]$$

Example:

$$E_5(W) = -t_{n1} \cdot \log[y_{n1}(W)] = -\log(0.94) = 0.06$$

Network training using gradient descent

Find the weights W that minimize

the total cross-entropy error:

$$E(W) = \sum_{n=1}^{N} E_n(W)$$

How to derive

the cross-entropy formula ?

The maximum likelihood method

Data:

$$X_1, \ldots, X_N \sim \mathcal{N}(\mu, \sigma^2)$$

Point estimate of μ:

$$\hat{\mu} = \text{argmax}_\mu \mathcal{L}(\mu|X) = \text{argmax}_\mu \prod_{i=1}^{N} \mathcal{N}_i(\mu, \sigma^2) = \frac{1}{N} \sum_{i=1}^{N} X_i$$

Equivalently:

$$\hat{\mu} = \text{argmin}_\mu \left[-\log \mathcal{L}(\mu|X) \right]$$

Network training using gradient descent

The total cross-entropy error is defined

as the negative log-likelihood

$$E(W) = -\log \mathcal{L}(W|T)\ ; \qquad \hat{W} = \mathrm{argmin}_W E(W)$$

where

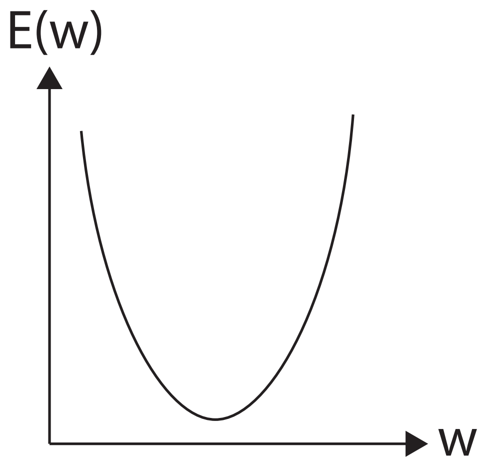
$$\mathcal{L}(W|T) = \prod_{n=1}^{N} \prod_{k=1}^{K} y_{nk}^{t_{nk}}(W)$$

In RED: the probability for the correct class

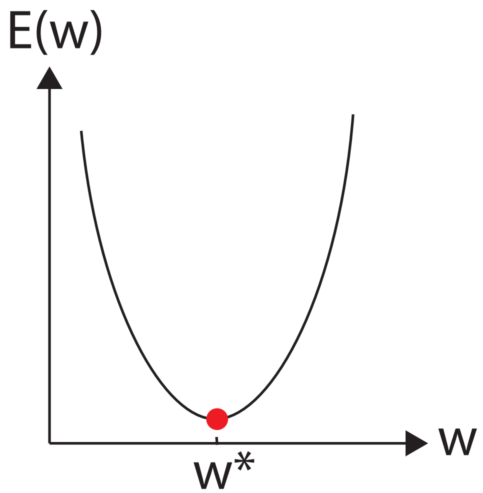
Network training using gradient descent

How does
gradient descent work ?

Network training using gradient descent

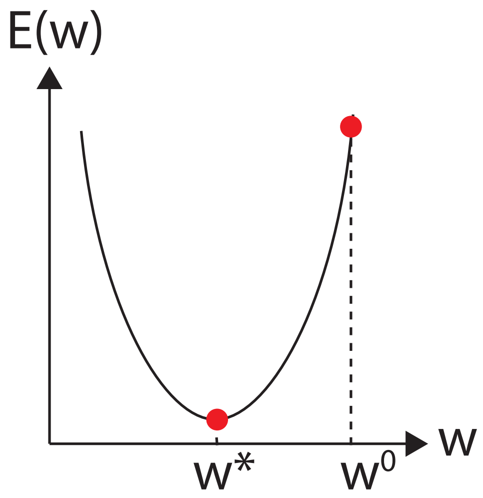


Network training using gradient descent



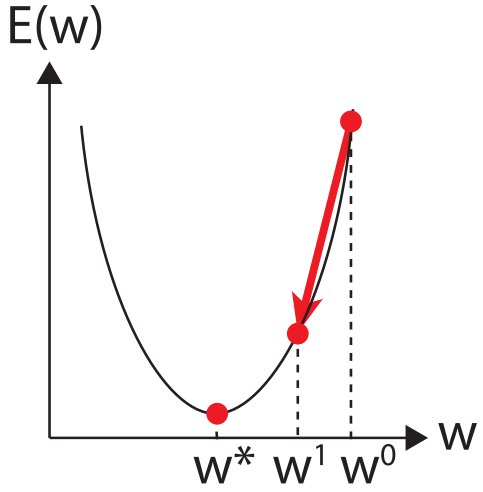
E(w*)=min(E)

Network training using gradient descent



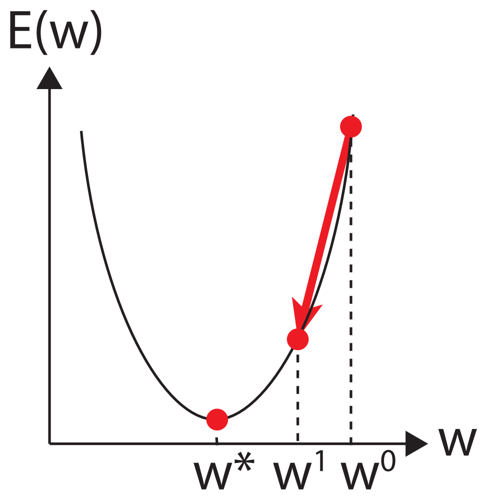
E(w*)=min(E)

Network training using gradient descent



E(w*)=min(E)

Network training using gradient descent

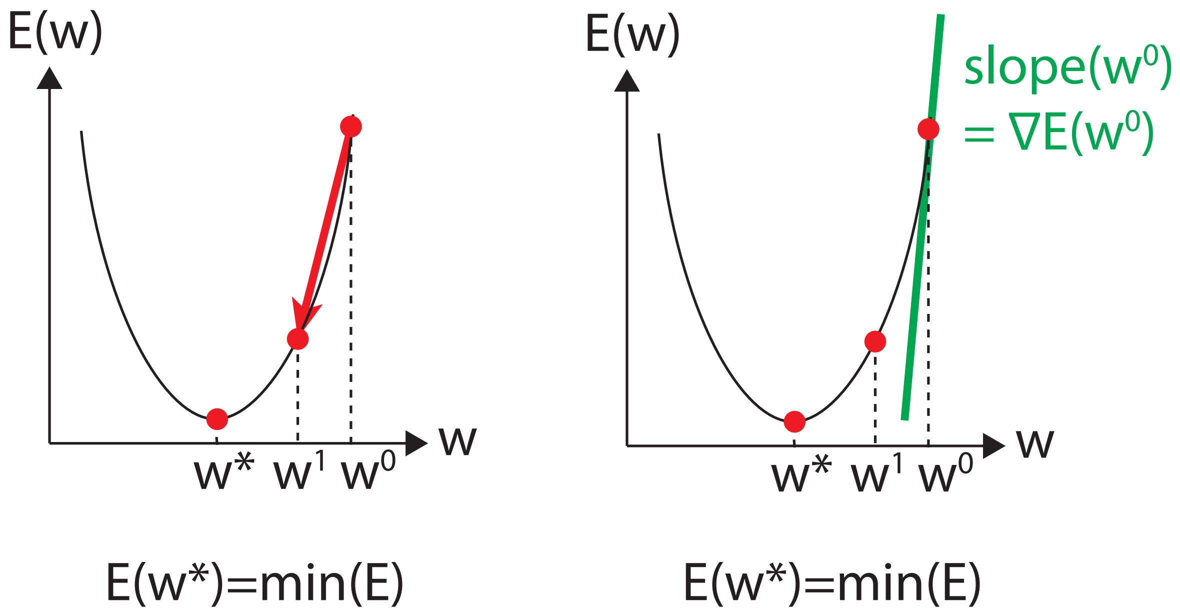


E(w*)=min(E)

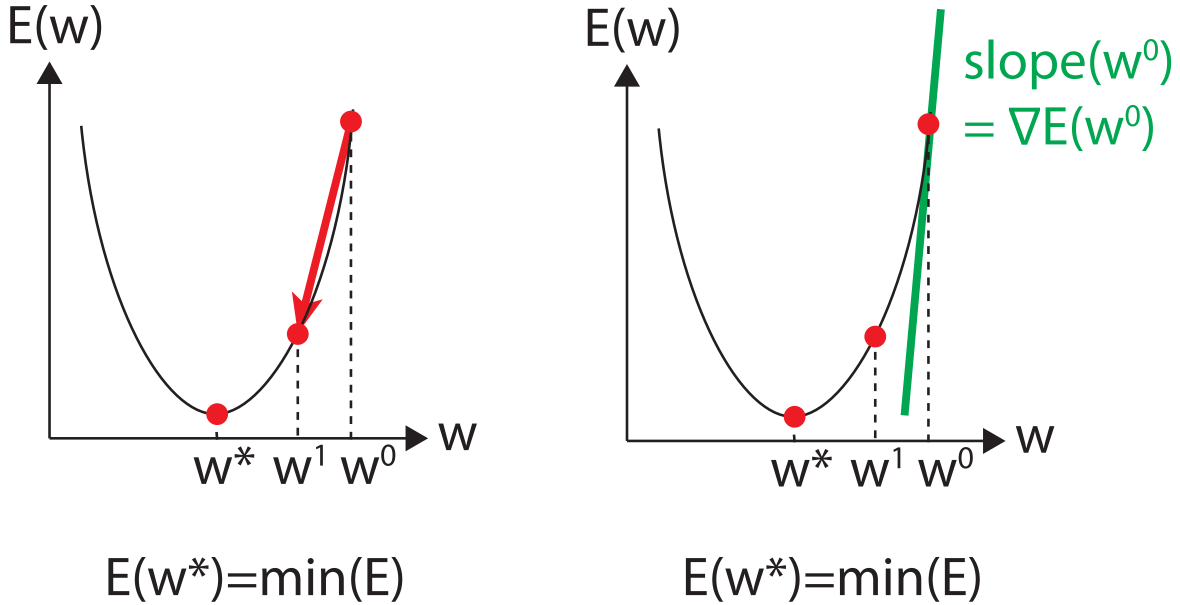
Several iterations:

w^0 -> w^1 -> w^2 -> w^3 -----> w^*

Network training using gradient descent



Network training using gradient descent



Gradient Descent:

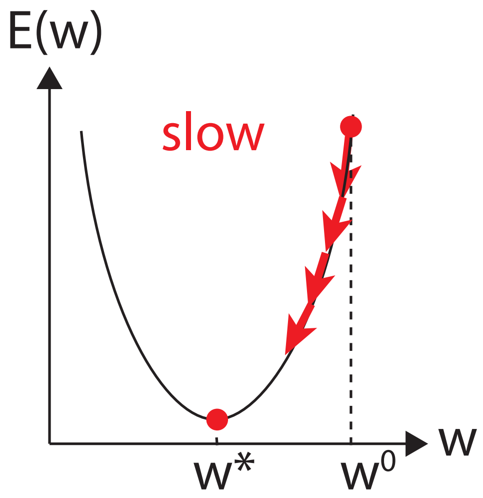
$w^1 = w^0 - \eta * \nabla E(w^0)$

$\nabla E(w^*) = 0; \;\; \eta > 0$

η = learning parameter

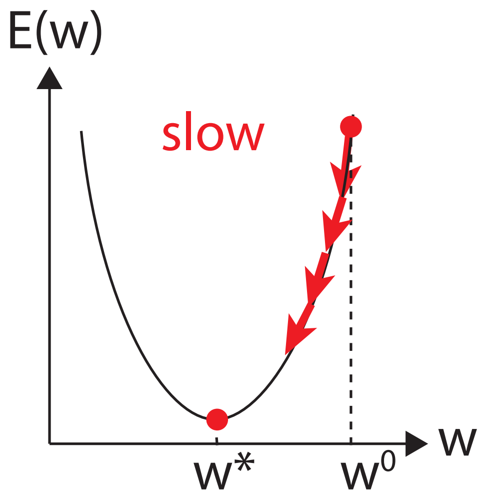
Possible problems

Network training using gradient descent

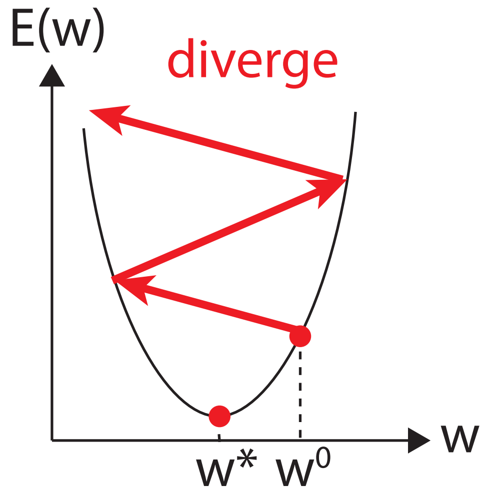


η too small

Network training using gradient descent

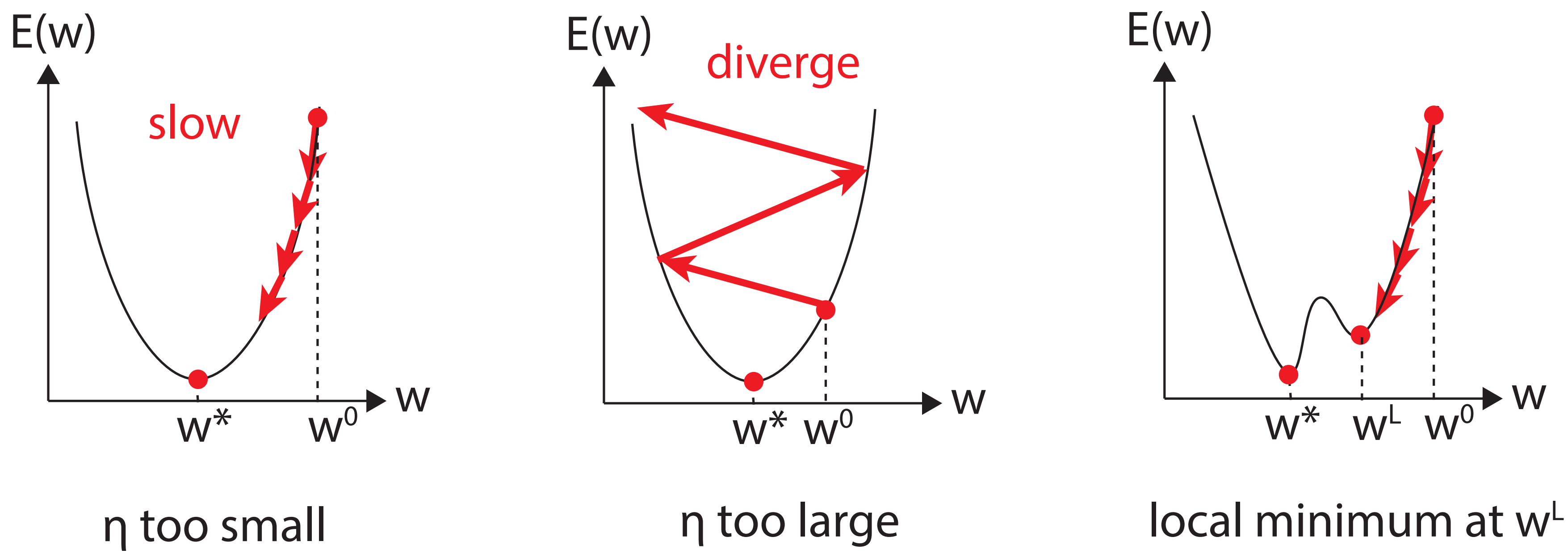


η too small



η too large

Network training using gradient descent



Network training using gradient descent

Initialization $\tau = 0$: Choose the initial weights W^0 with $\mathcal{N}(0, \sigma^2)$

Network training using gradient descent

Initialization $\tau = 0$: Choose the initial weights W^0 with $\mathcal{N}(0, \sigma^2)$

$$W^{(1)} = \begin{pmatrix} 0.5 & 0.1 & -0.2 & -0.4 \\ -0.4 & 1.0 & 0.5 & 1.0 \\ -0.2 & -0.2 & -0.5 & -0.1 \\ 0.2 & 0.7 & 0.3 & 0.2 \\ 0.6 & 0.6 & 0.1 & -0.4 \end{pmatrix}$$

$$W^{(2)} = \begin{pmatrix} 0.6 & 0.1 & 0.9 & -0.2 & -0.5 \\ 0.3 & -0.3 & 0.3 & -0.9 & -0.9 \\ 0.3 & 0.2 & 0.4 & -1.0 & 0.6 \end{pmatrix}$$

Network training using gradient descent

3 possibilities for the next steps

3 possibilities for the next steps

1. Batch Gradient Descent
2. Mini-batch Gradient Descent
3. Stochastic Gradient Descent

Batch Gradient Descent

1. Apply the NN to all the train set
2. Record all the errors
3. Update the weights:

$$W^{\tau} = W^{\tau-1} - \eta \cdot \nabla E(W^{\tau-1})$$

Mini-batch Gradient Descent

1. Apply the NN to a batch of the train set
2. Record the corresponding errors
3. Update the weights:

$$W^{\tau} = W^{\tau-1} - \eta \cdot \nabla \sum_{n \in batch} E_n(W^{\tau-1})$$

Stochastic Gradient Descent

1. Apply the NN to one sample of the train set
2. Record the one sample error
3. Update the weights:

$$W^{\tau} = W^{\tau-1} - \eta \cdot \nabla E_n(W^{\tau-1})$$

Network training using gradient descent

Iteration

1 iteration (or pass) is one weight update

Epoch

1 epoch is reached

when the NN has passed through

all the training data

EXAMPLE

If you have 100 training samples,

and your batch size is 50,

then it will take 2 iterations to complete 1 epoch

Gradient Descent

1. Batch Gradient Descent:
   1 epoch = 1 iteration

2. Mini-batch Gradient Descent:
   1 epoch = (N/batch) iterations

3. Stochastic Gradient Descent:
   1 epoch = N iterations

Performance metrics

What are

the performance metrics ?

Performance metrics

They may be used on

the training, validation and test sets

Cross-entropy error/loss function

$$E = -\sum_{n=1}^{N}\sum_{k=1}^{3} t_{nk} \cdot \log y_{nk}$$

Confusion matrix

```
            predictions
actuals          setosa versicolor virginica
  setosa             14          0         0
  versicolor          0          9         0
  virginica           0          2        10
```

Performance metrics

Accuracy rate = 1 - Error rate

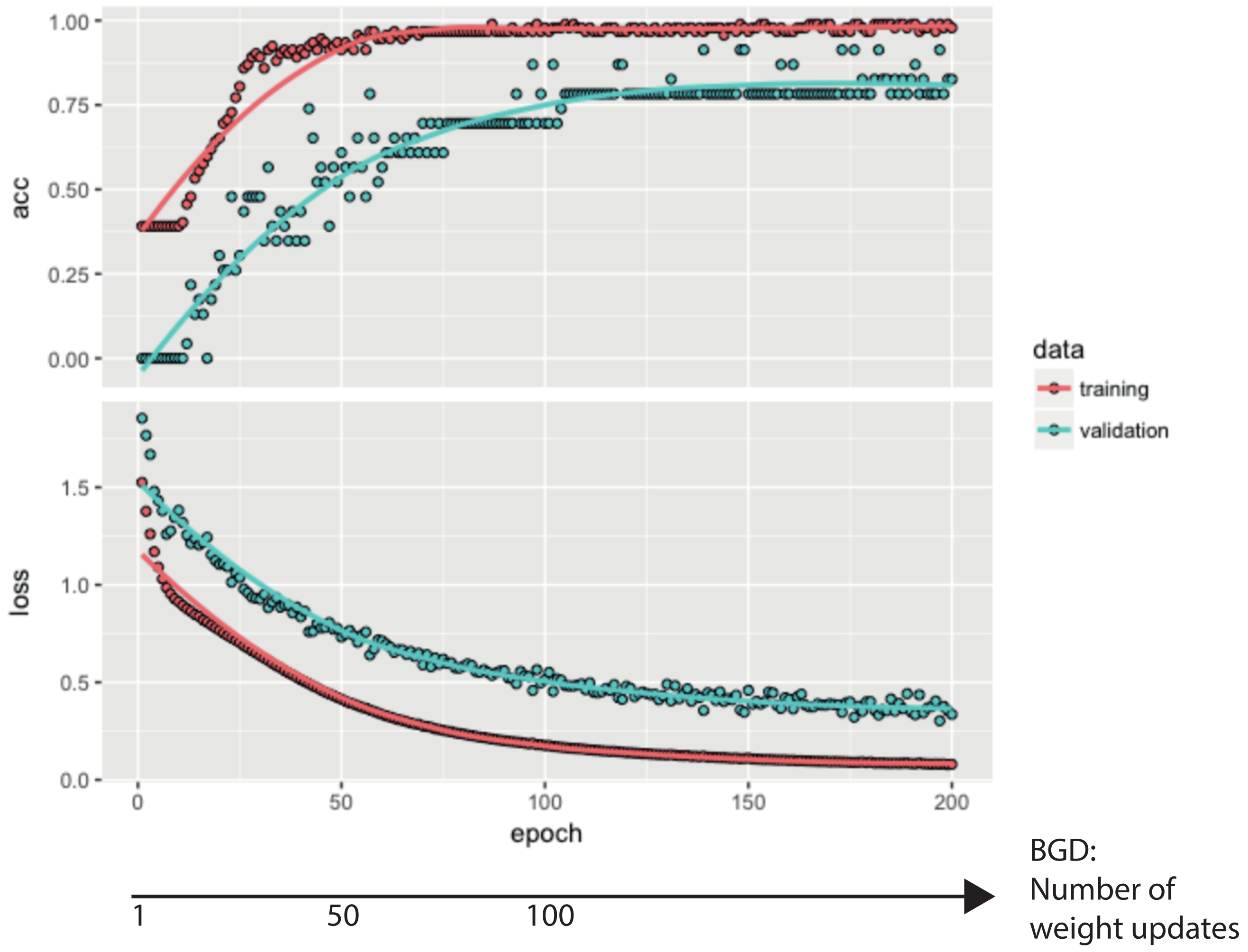
$$\text{Accuracy rate} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} = \frac{33}{35} = 94\%$$

$$\text{Error rate} = \frac{\text{Number of wrong predictions}}{\text{Total number of predictions}} = \frac{2}{35} = 6\%$$

EXAMPLES

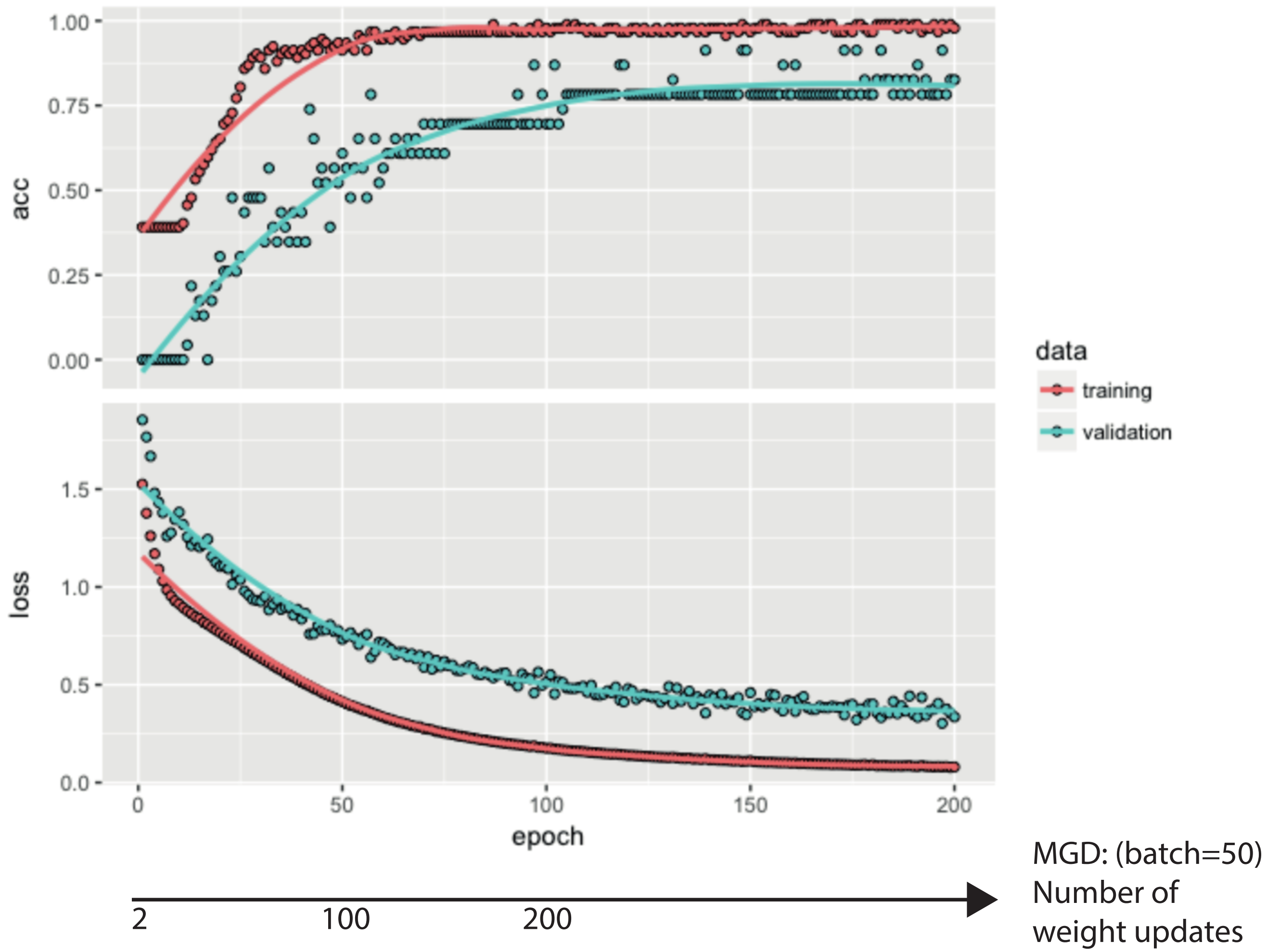
Performance metrics

Training and validation datasets:



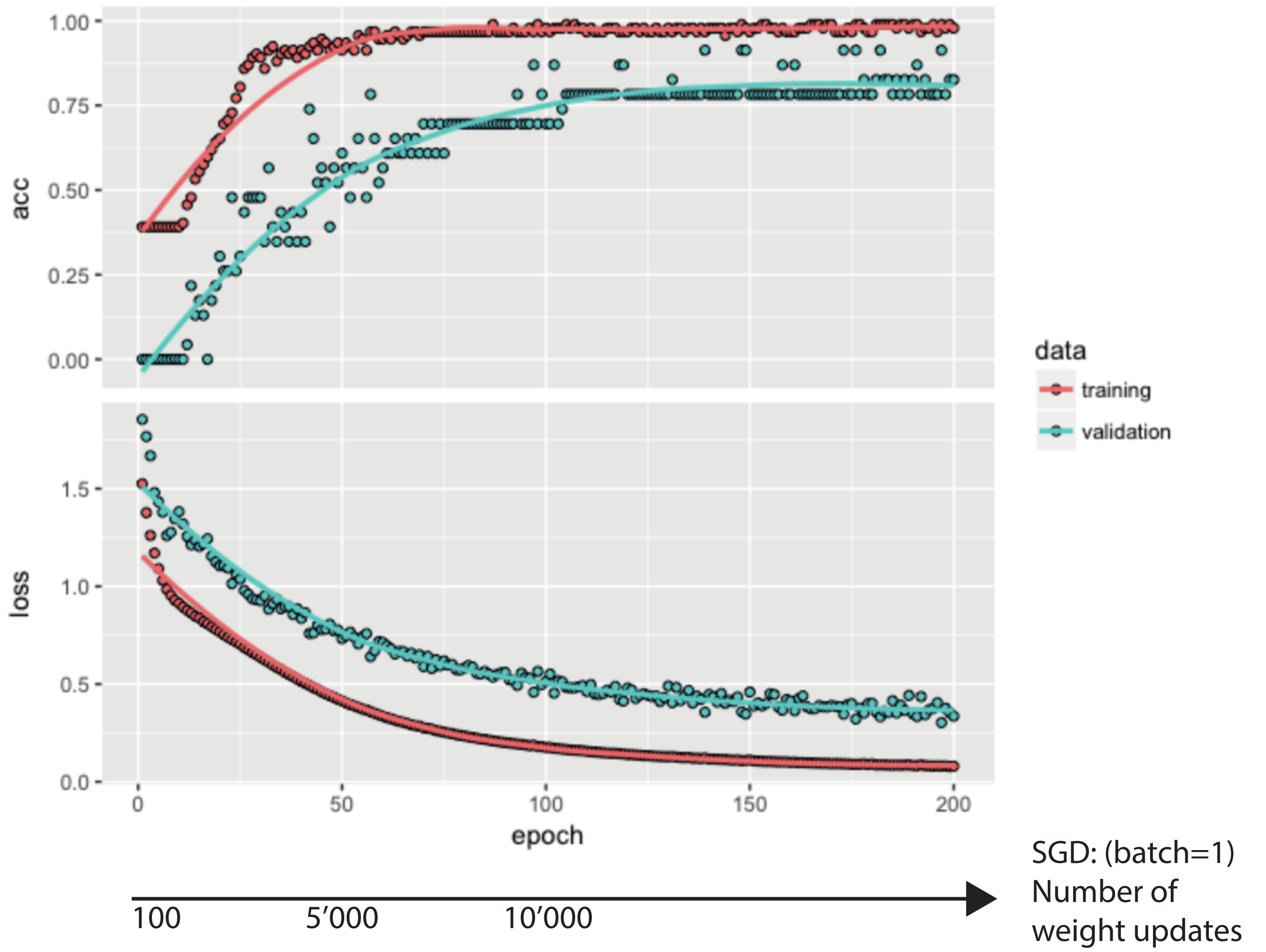
Performance metrics

Training and validation datasets:



Performance metrics

Training and validation datasets:



Test set

Test set

Accuracy=0.91 and cross-entropy loss=0.22

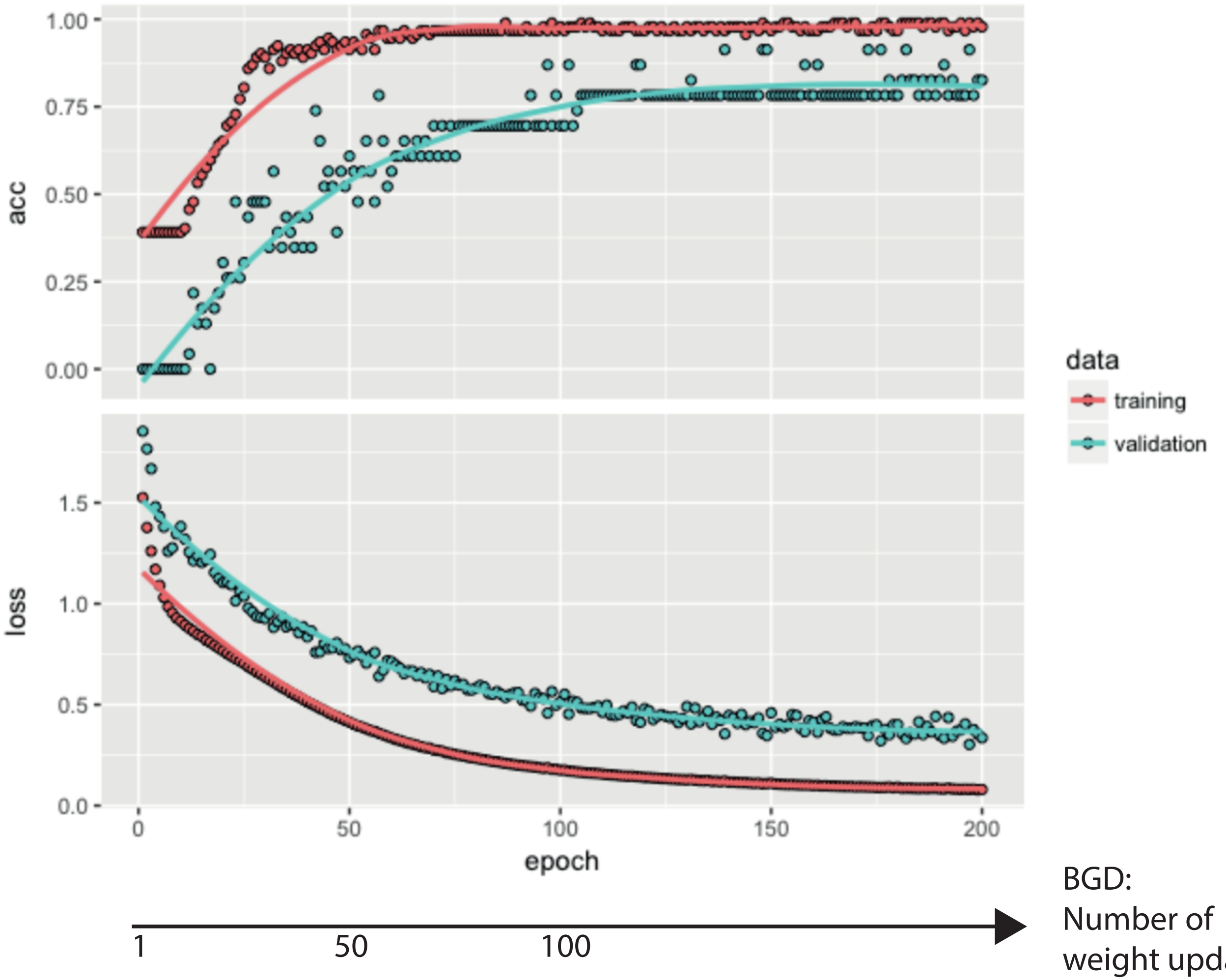
Performance metrics

Test set

Accuracy=0.91 and cross-entropy loss=0.22

THE END

Optimum number of epochs



BGD:
Number of
weight updates

Optimum number of epochs

What is the optimum

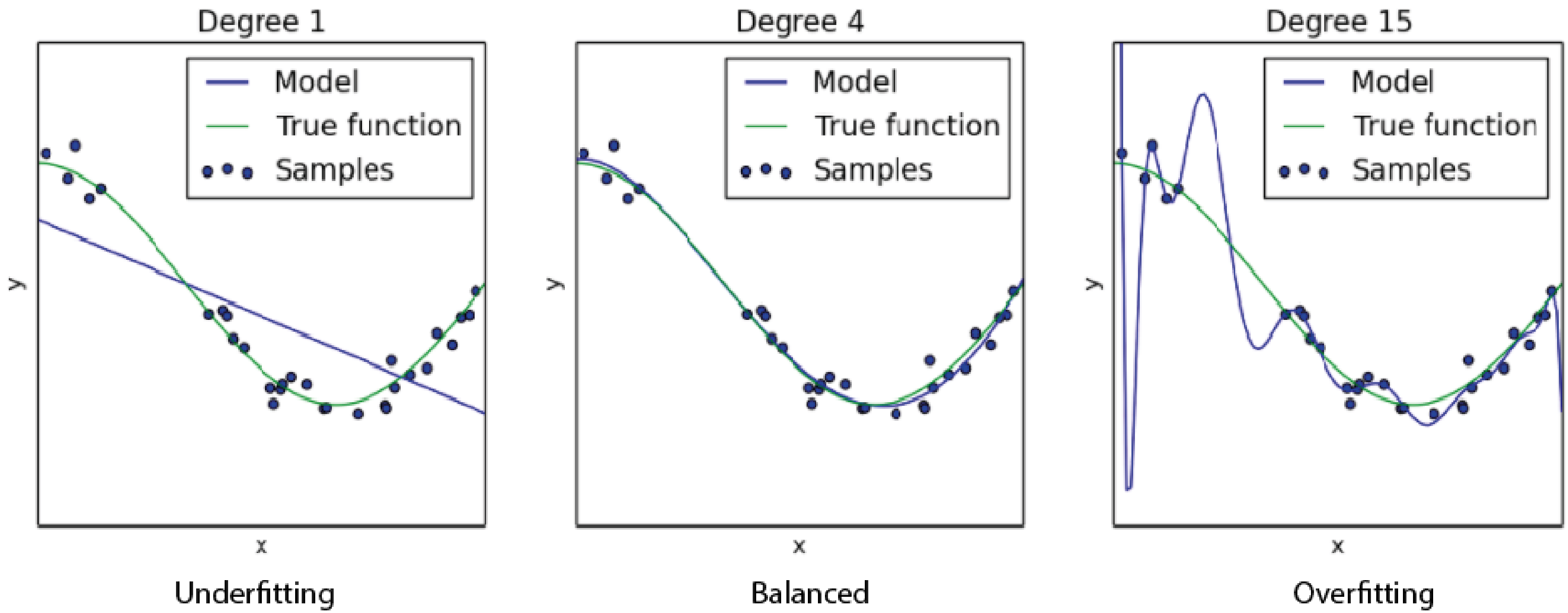
number of epochs ?

Optimum number of epochs

The answer is related to the problem of

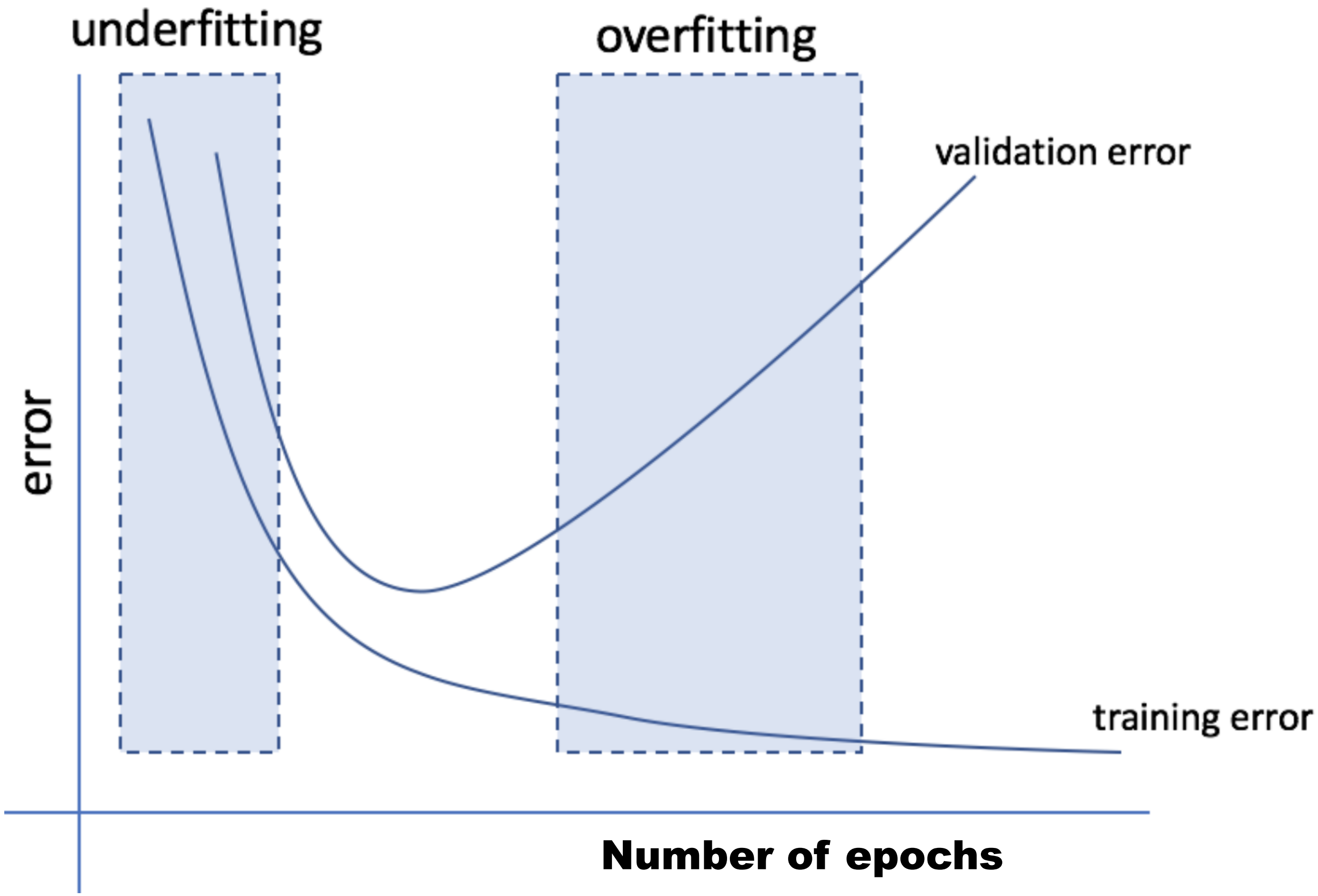
under-fitting and over-fitting

Optimum number of epochs



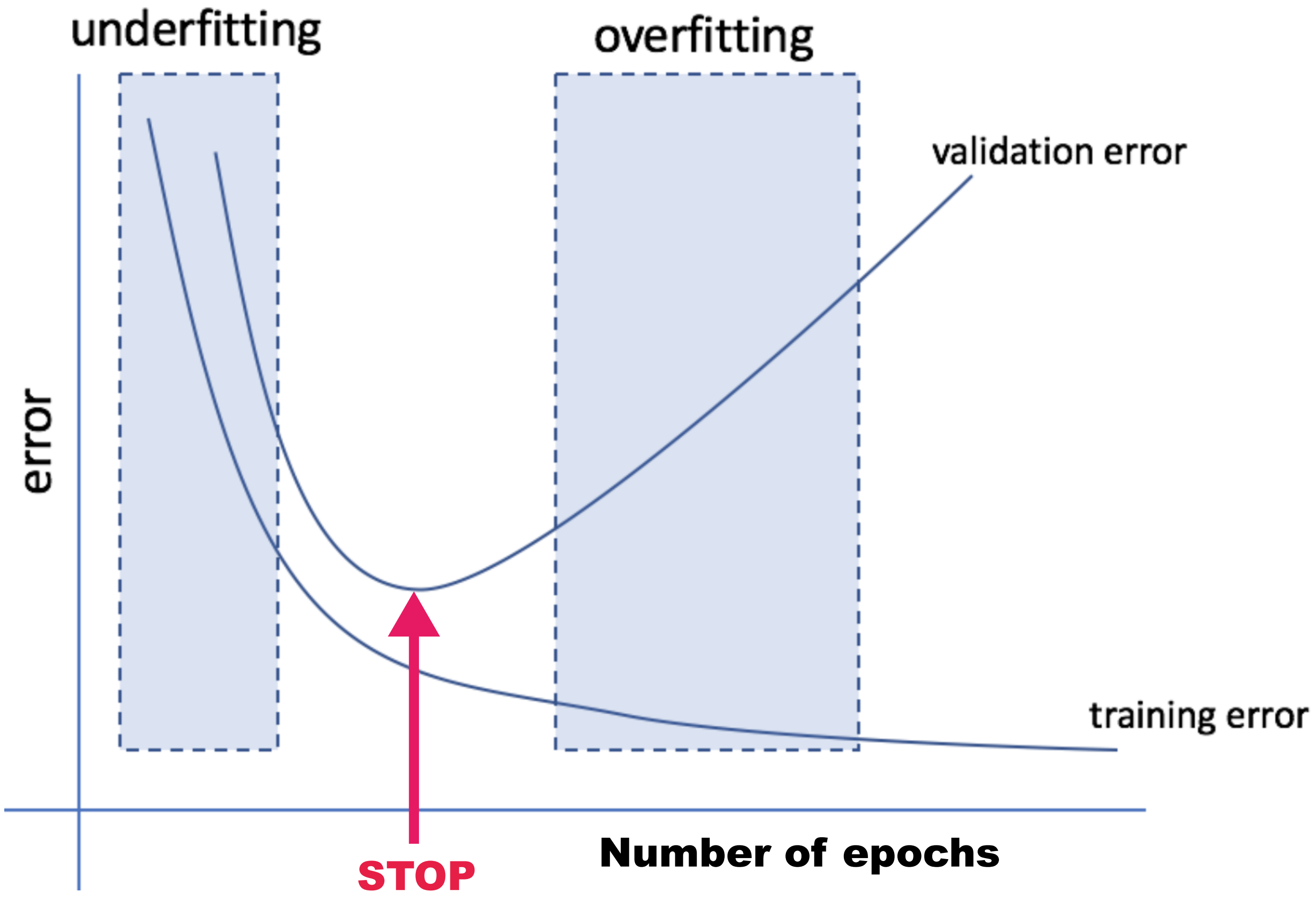
Ref: scikit-learn 0.18 documentation

Optimum number of epochs



Ref: www.jeremyjordan.me

Optimum number of epochs



Ref: www.jeremyjordan.me

Conclusion on neural network

Can one reach 100% accuracy ?

Conclusion on neural network

Short answer

It is possible only if

there is enough information in the input X

to predict Y uniquely

EXAMPLE

If two plants have the same four attributes

$$(X_1 = X_2)$$

but belong to two different species

$$(Y_1 \neq Y_2),$$

then we need additional features

to characterize uniquely the three iris species

Conclusion on neural network

If two people have the same gender and age

$(X_1 = X_2)$

but only one has a specific disease

$(Y_1 \neq Y_2)$,

then we need additional features

(physical activity, smoking, genetics)

to characterize uniquely the risk of this disease

IN GENERAL

$$Y = f(X) + \text{error}$$

Hyperparameters

Number of layers, number of nodes,

initial weight values, activation function,

error/loss function, number of epochs,

learning rate, batch size, bias node

Conclusion on neural network

How to choose them ?

Conclusion on neural network

Trial and Error

Select the combination that performs best

(highest validation accuracy)

Conclusion on neural network

Trial and Error

The goal is to predict

(and not really to explain)

Conclusion on neural network

Main advantage:

- Works well on a whole range of problems including image and signal recognitions.

Conclusion on neural network

Main advantage:

- Works well on a whole range of problems including image and signal recognitions.

Main disadvantage:

- Black box: difficult to understand what are the main features that the neural network uses to make prediction. Decision trees are better suited for interpretation.

QUESTIONS ?

Applications

Application 1

Predict the 1-year mortality rate

of elderly patients

with intertrochanteric fractures

Ref: Artificial neural network models for predicting 1-year mortality in elderly patients with intertrochanteric fractures in China, L. Shi, X.C. Wang and Y.S. Wang, Brazilian Journal of Medical and Biological Research (2013)

Application 1

Some older people fall

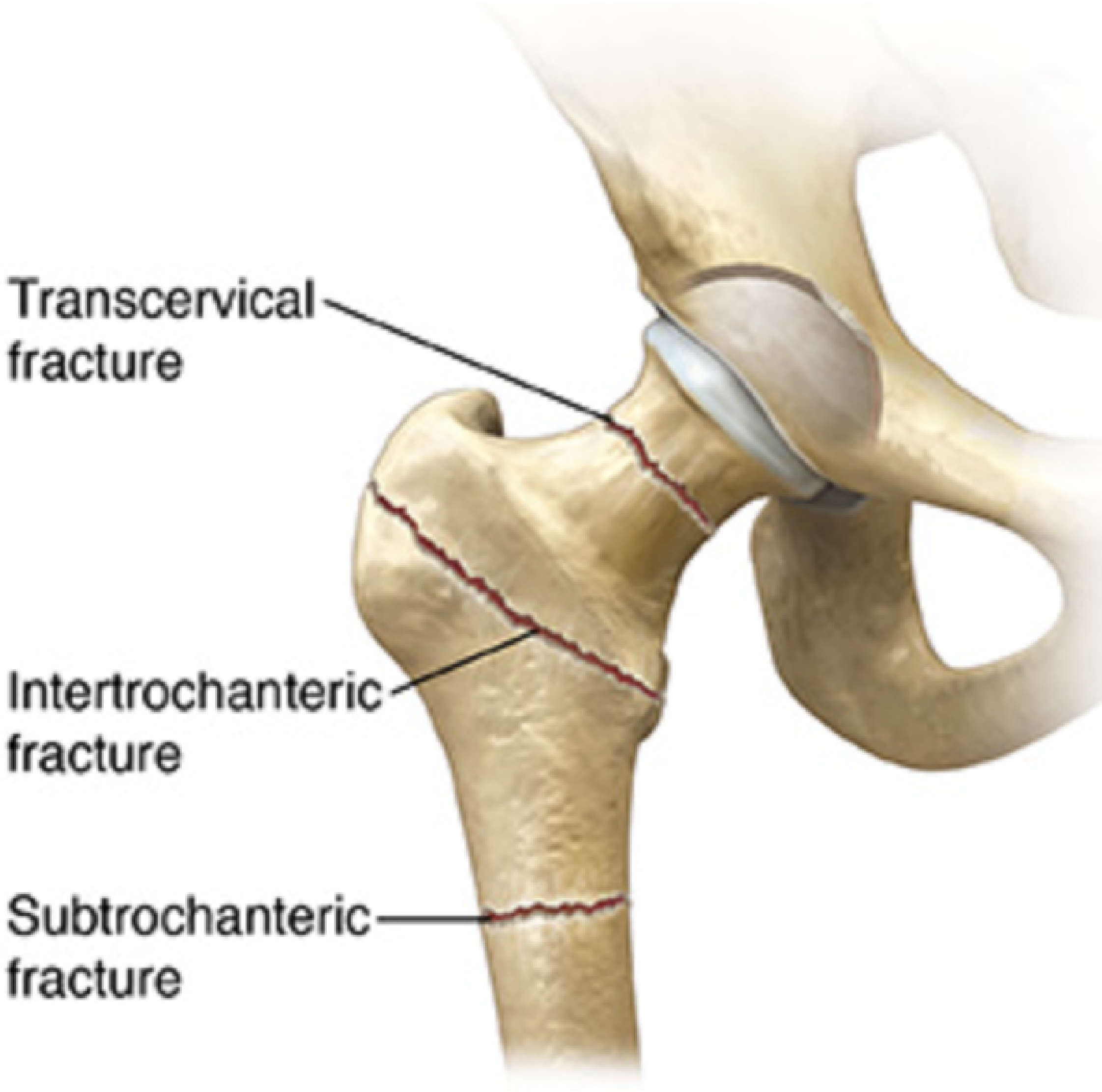
and break one of their hips

Application 1

50% of hip fractures

are intertrochanteric fractures

Application 1



Application 1

There is an increase of death

after intertrochanteric fractures

(because of reduced mobility)

Application 1

$$\text{1-year mortality rate} = D/N$$

D = number of deaths occurring within 1 year

N = the size of the population

(all patients with intertrochanteric fractures)

Data

2150 patients with intertrochanteric fractures:

70% in the training group

30% patients in the testing group

Application 1

After some trial and error

with different hyperparameters

(number of layers and nodes)

they end up with the following neural network

Application 1

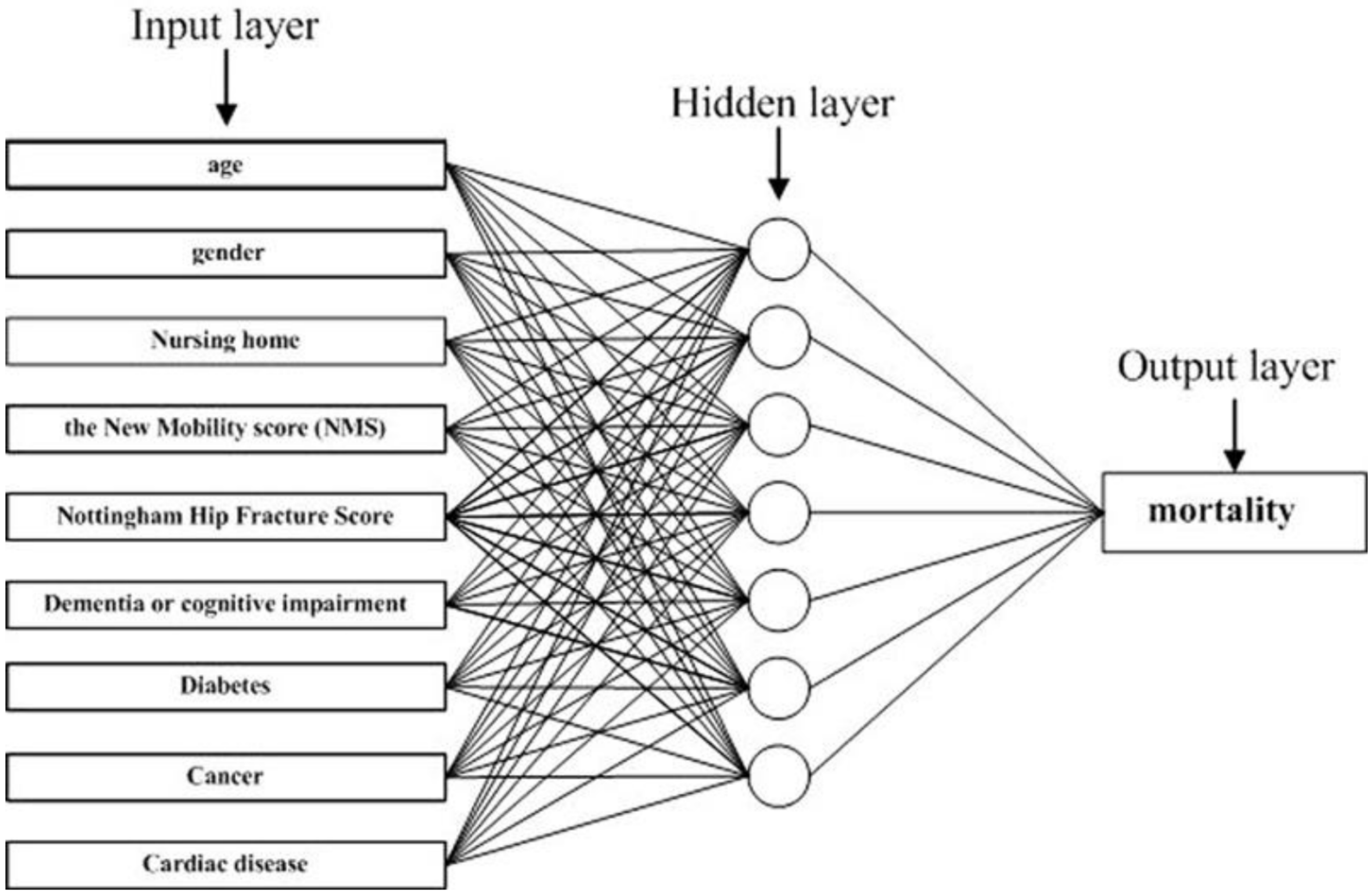


Figure 2 Schematic representation showing the structure of the artificial neural network models, which have 8 input nodes, 6 nodes in hidden layer, and 1 output node, which represents 1-year mortality in elderly patients with intertrochanteric fracture.

Application 1

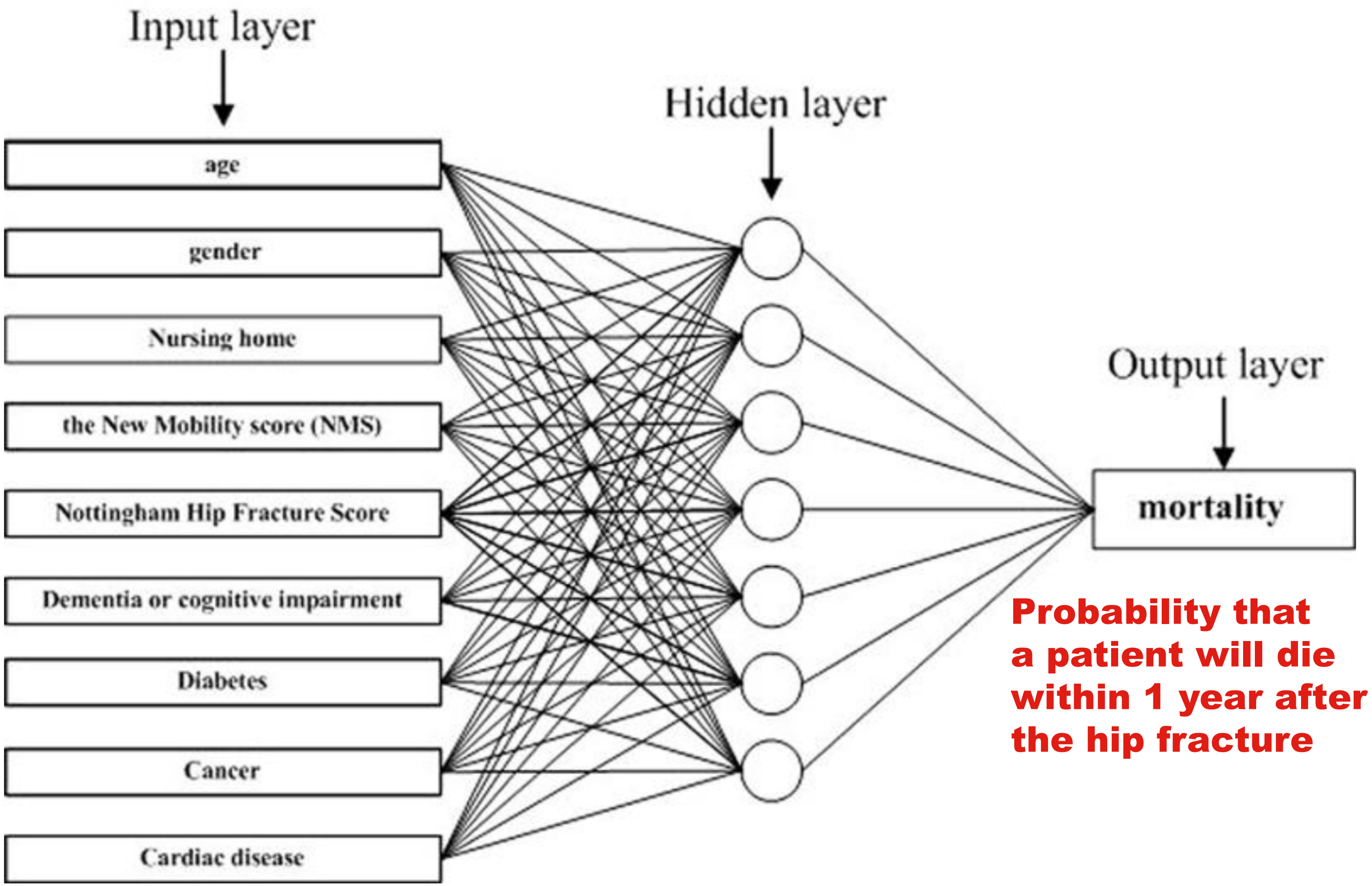


Figure 2 Schematic representation showing the structure of the artificial neural network models, which have 8 input nodes, 6 nodes in hidden layer, and 1 output node, which represents 1-year mortality in elderly patients with intertrochanteric fracture.

Accuracy

92% for the training group

86% for the testing group

Application 2

Predict if there is a residual tumor

after bladder cancer treatment

Ref: Bladder Cancer Treatment Response Assessment in CT using Radiomics with Deep-Learning, Kenny H. Cha, Lubomir Hadjiiski, Heang-Ping Chan, Alon Z. Weizer, Ajjai Alva, Richard H. Cohan, Elaine M. Caoili, Chintana Paramagul and Ravi K. Samala, Scientific Reports volume 7, Article number: 8738 (2017)

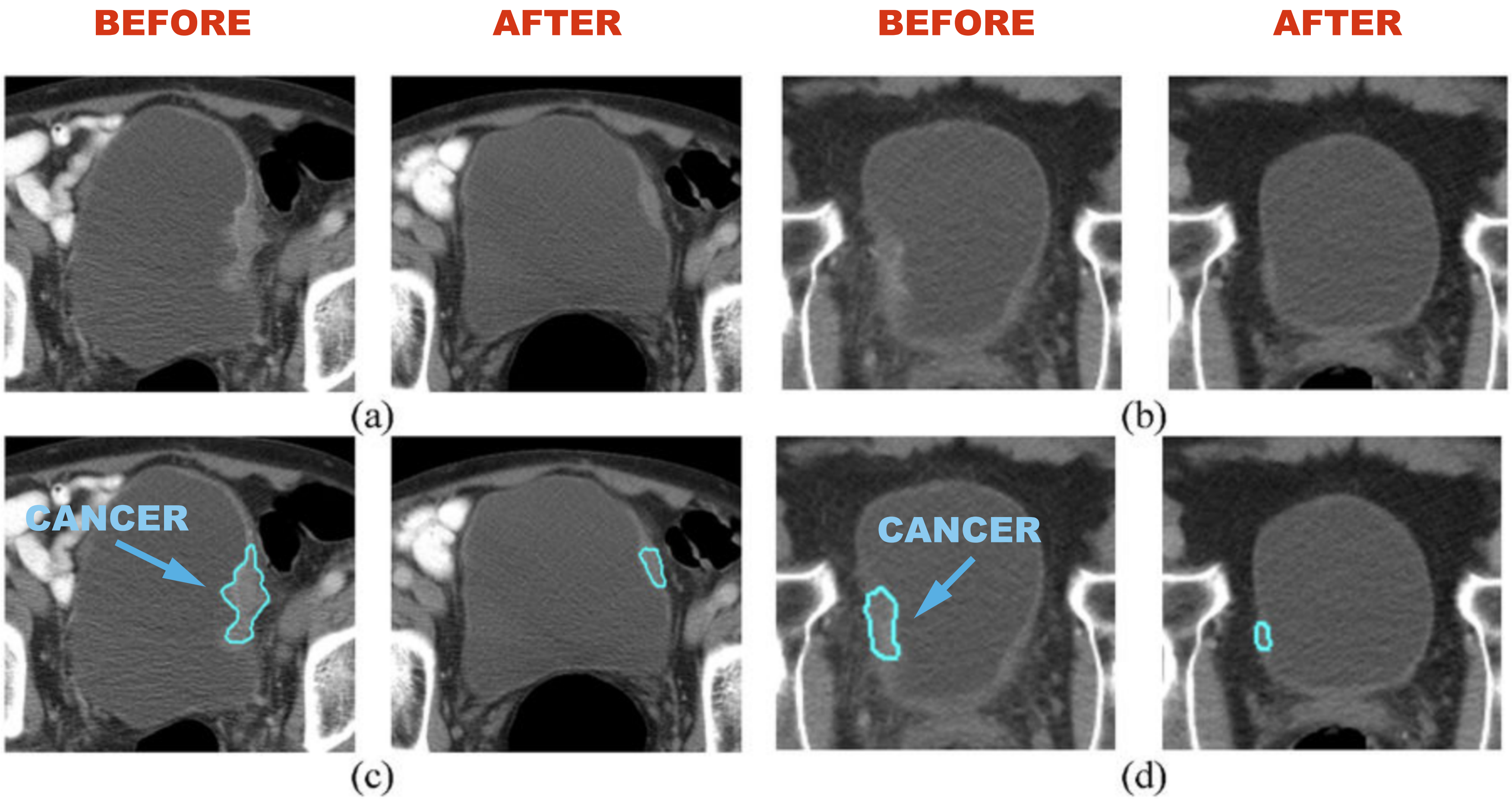
Application 2

They take X-ray images of the bladder and

use an algorithm to localise the cancer region

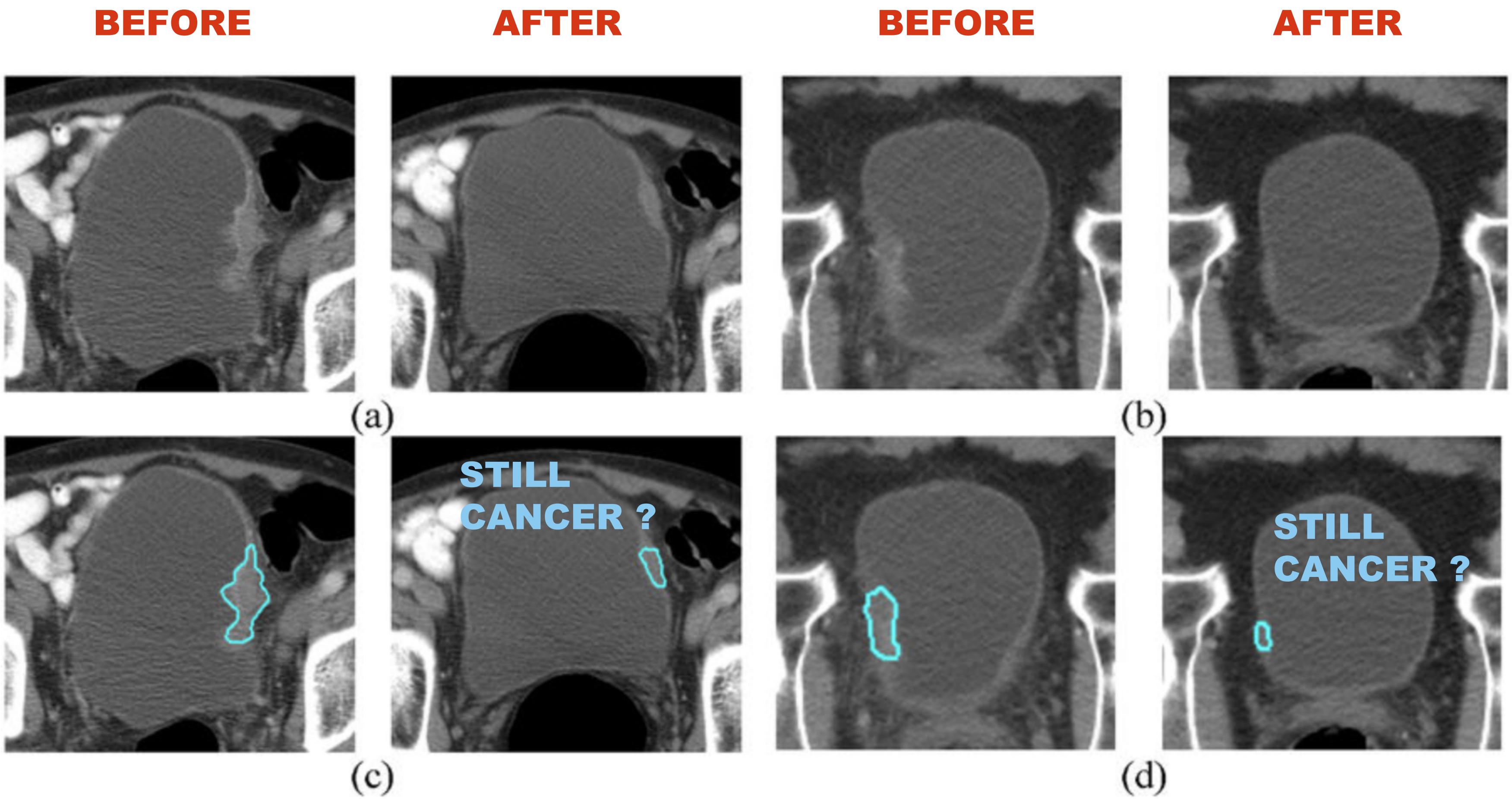
before and after treatment

Application 2



Bladder lesion segmentations. Two segmented bladder cancers are illustrated. The lesions in the pre- and post-treatment scan pairs shown in (**a**,**b**) are segmented using AI-CALS, as shown in (**c**,**d**), respectively. The pre-treatment scan is on the left and the post-treatment scan is located on the right of each pair.

Application 2



Bladder lesion segmentations. Two segmented bladder cancers are illustrated. The lesions in the pre- and post-treatment scan pairs shown in (**a**,**b**) are segmented using AI-CALS, as shown in (**c**,**d**), respectively. The pre-treatment scan is on the left and the post-treatment scan is located on the right of each pair.

Data

6700 pre-post-treatment paired images

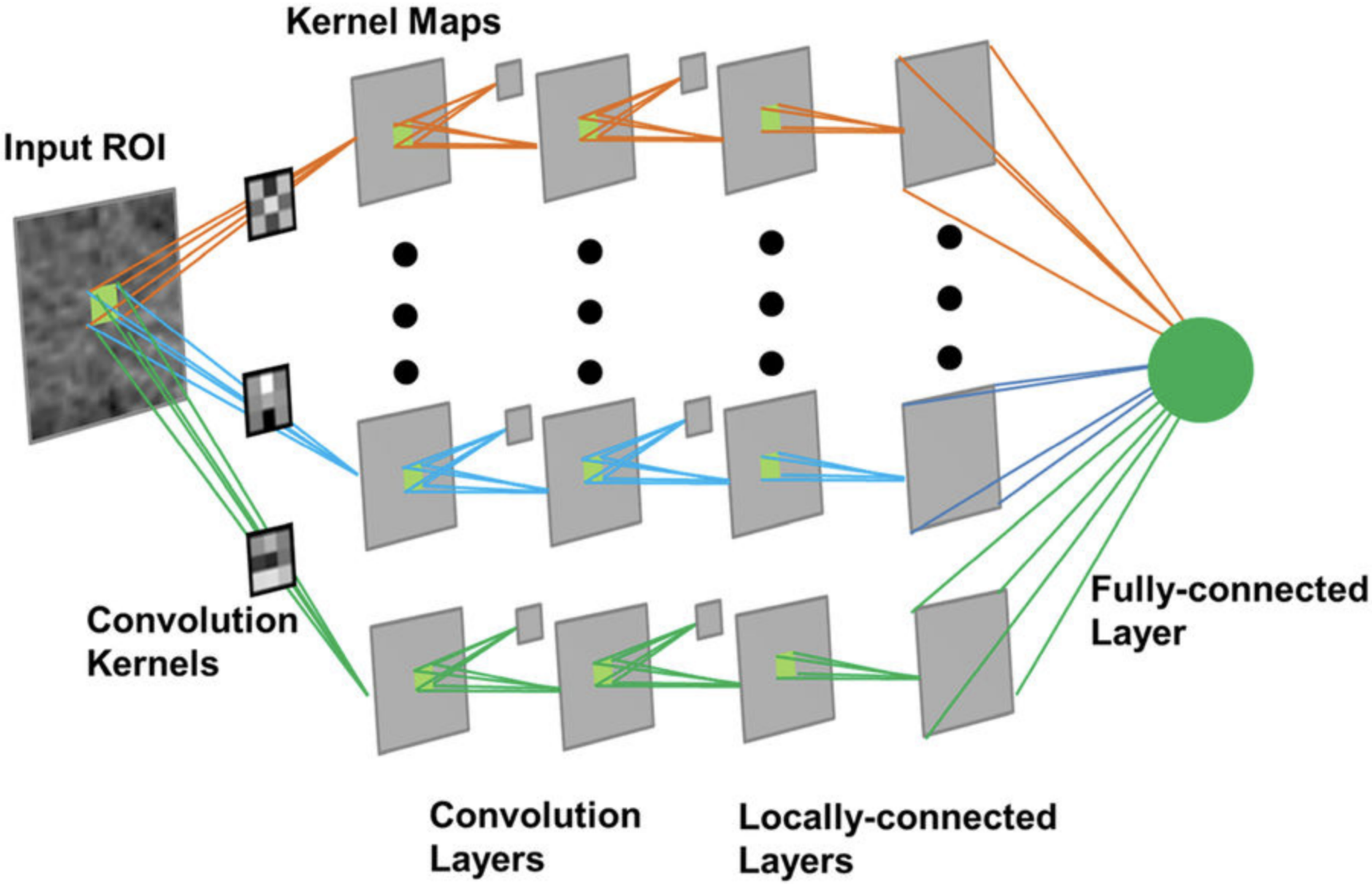
with located cancer region

Data

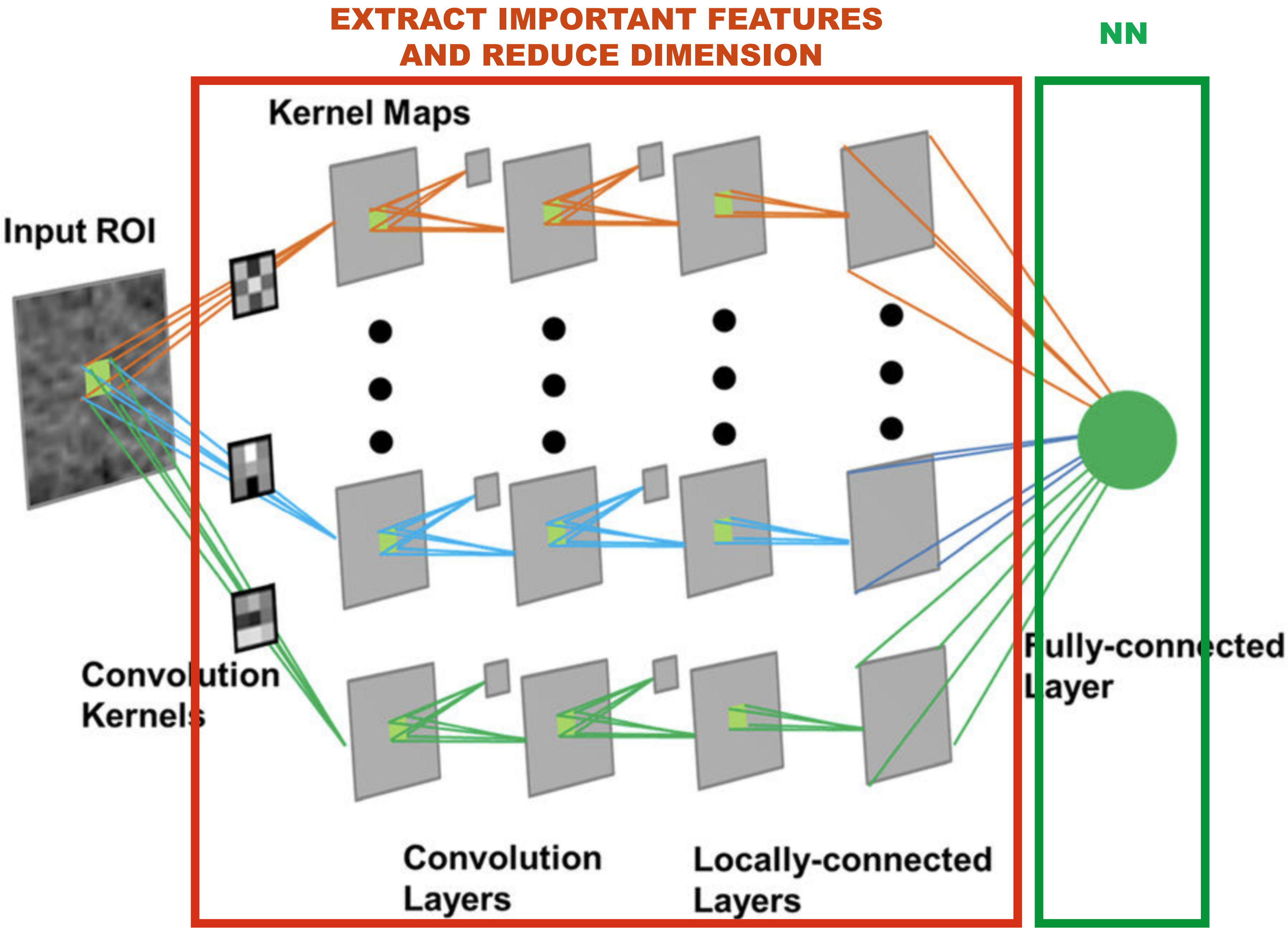
They combined the paired images

into Region Of Interest (ROI) images

Application 2



Application 2



Application 2

Table 2 Number of correctly predicted bladder cancer treatment response assessment of the test set at an operating point determined using the training set.

From: Bladder Cancer Treatment Response Assessment in CT using Radiomics with Deep-Learning

	DL-CNN	RF-SL	RF-ROI	Radiologist 1	Radiologist 2
Complete Response (Sensitivity)	6/12 (50%)	6/12 (50%)	8/12 (66.7%)	11/12 (91.7%)	11/12 (91.7%)
Non-complete Response (Specificity)	34/42 (81.0%)	33/42 (78.6%)	23/42 (54.8%)	18/42 (42.9%)	16/42 (38.1%)

DL-CNN: Deep-learning convolution neural network. RF-SL: Radiomics features extracted from segmented lesions. RF-ROI: Radiomics features extracted from pre- and post-treatment paired ROIs.

Complete response = No residual cancer

Non-complete response = Residual cancer

Application 2

Table 2 Number of correctly predicted bladder cancer treatment response assessment of the test set at an operating point determined using the training set.

From: Bladder Cancer Treatment Response Assessment in CT using Radiomics with Deep-Learning

	DL-CNN	RF-SL	RF-ROI	Radiologist 1	Radiologist 2
Complete Response (Sensitivity)	6/12 (50%)	6/12 (50%)	8/12 (66.7%)	11/12 (91.7%)	11/12 (91.7%)
Non-complete Response (Specificity)	34/42 (81.0%)	33/42 (78.6%)	23/42 (54.8%)	18/42 (42.9%)	16/42 (38.1%)

DL-CNN: Deep-learning convolution neural network. RF-SL: Radiomics features extracted from segmented lesions. RF-ROI: Radiomics features extracted from pre- and post-treatment paired ROIs.

Complete response = No residual cancer

Non-complete response = Residual cancer

Application 2

Table 2 Number of correctly predicted bladder cancer treatment response assessment of the test set at an operating point determined using the training set.

From: Bladder Cancer Treatment Response Assessment in CT using Radiomics with Deep-Learning

	DL-CNN	RF-SL	RF-ROI	Radiologist 1	Radiologist 2
Complete Response (Sensitivity)	6/12 (50%)	6/12 (50%)	8/12 (66.7%)	11/12 (91.7%)	11/12 (91.7%)
Non-complete Response (Specificity)	34/42 (81.0%)	33/42 (78.6%)	23/42 (54.8%)	18/42 (42.9%)	16/42 (38.1%)

DL-CNN: Deep-learning convolution neural network. RF-SL: Radiomics features extracted from segmented lesions. RF-ROI: Radiomics features extracted from pre- and post-treatment paired ROIs.

Complete response = No residual cancer

Non-complete response = Residual cancer

Application 3

Diagnose irregular heart rhythms (arrhythmias)

from single-lead electrocardiography signals

Ref: Cardiologist-Level Arrhythmia Detection With Convolutional Neural Networks, Pranav Rajpurkar, Awni Hannun,

Masoumeh Haghpanahi, Codie Bourn, and Andrew Ng, arXiv:1707.01836

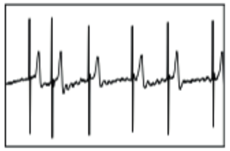
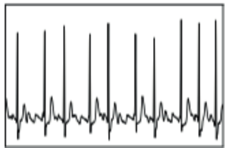
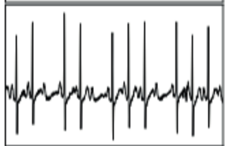
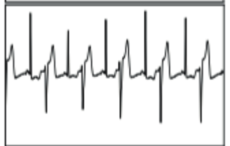
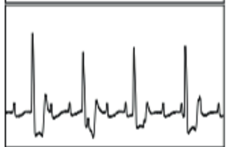
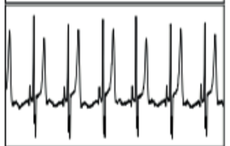
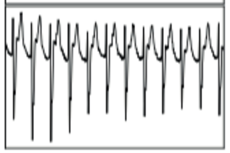
Data

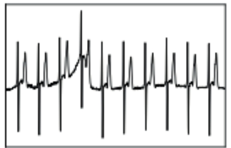
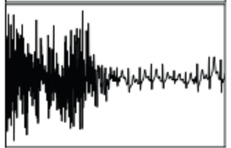
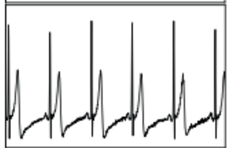
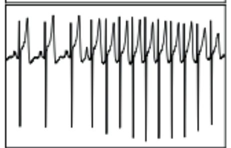
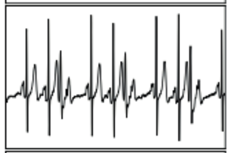
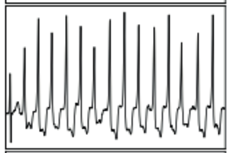
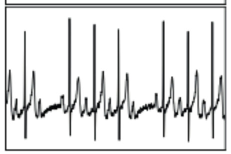
60’000 electrocardiography records

(annotated by experts with 14 classes)

from 30’000 patients

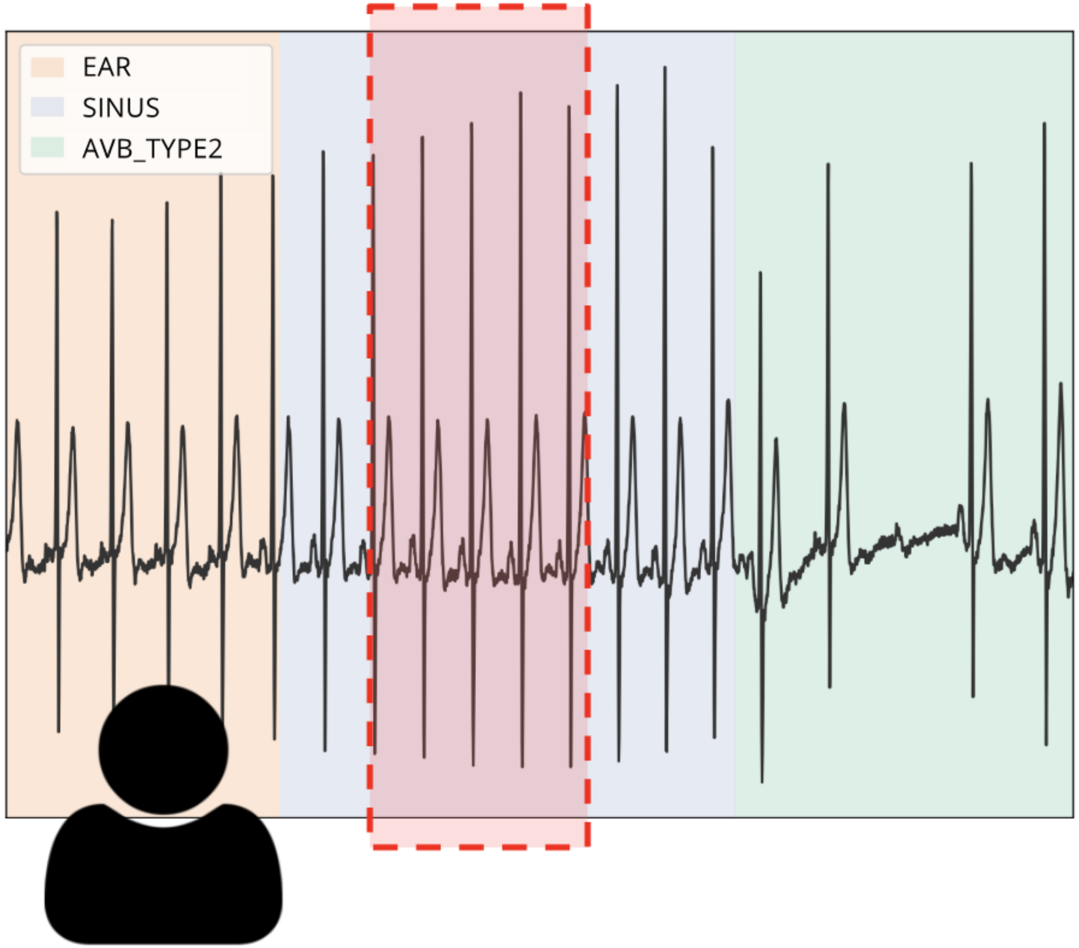
Application 3

Class	Description	Example	Train + Val Patients	Test Patients
AFIB	Atrial Fibrillation		4638	44
AFL	Atrial Flutter		3805	20
AVB_TYPE2	Second degree AV Block Type 2 (Mobitz II)		1905	28
BIGEMINY	Ventricular Bigeminy		2855	22
CHB	Complete Heart Block		843	26
EAR	Ectopic Atrial Rhythm		2623	22
IVR	Idioventricular Rhythm		1962	34

Class	Description	Example	Train + Val Patients	Test Patients
JUNCTIONAL	Junctional Rhythm		2030	36
NOISE	Noise		9940	41
SINUS	Sinus Rhythm		22156	215
SVT	Supraventricular Tachycardia		6301	34
TRIGEMINY	Ventricular Trigeminy		2864	21
VT	Ventricular Tachycardia		4827	17
WENCKEBACH	Wenckebach (Mobitz I)		2051	29

Application 3

GOAL



Application 3

The model outputs a new prediction once every second

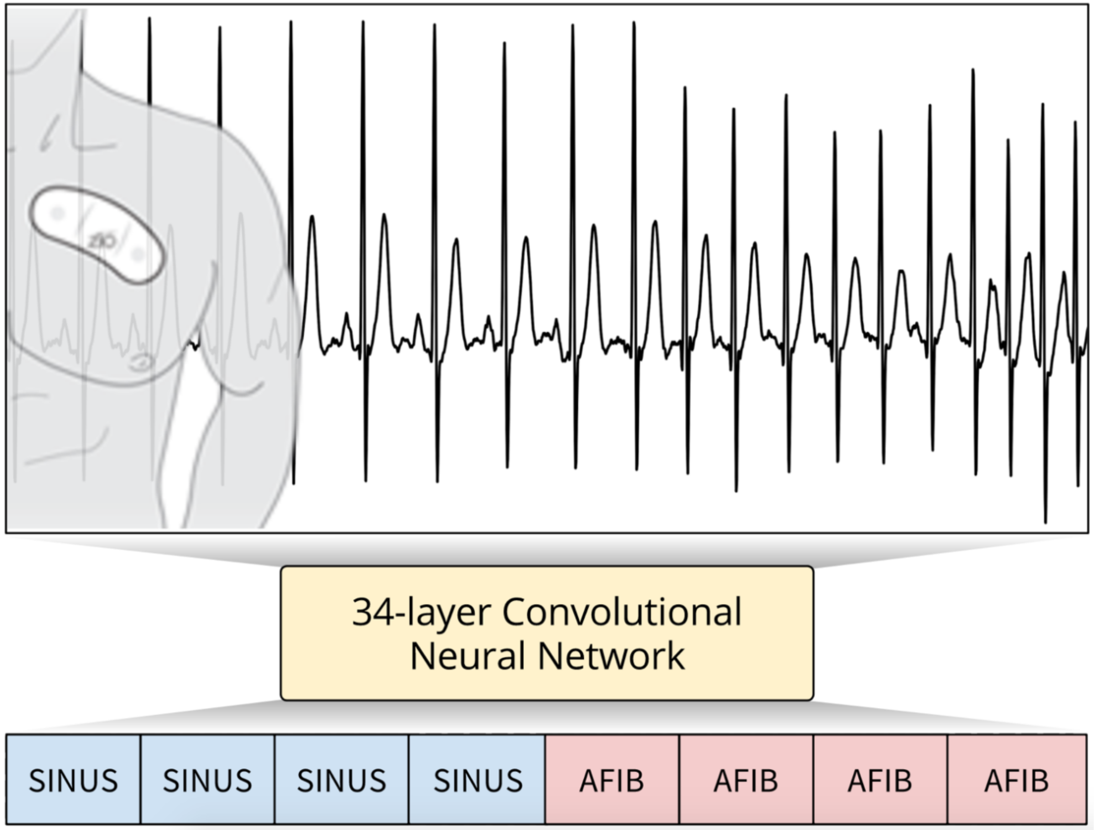
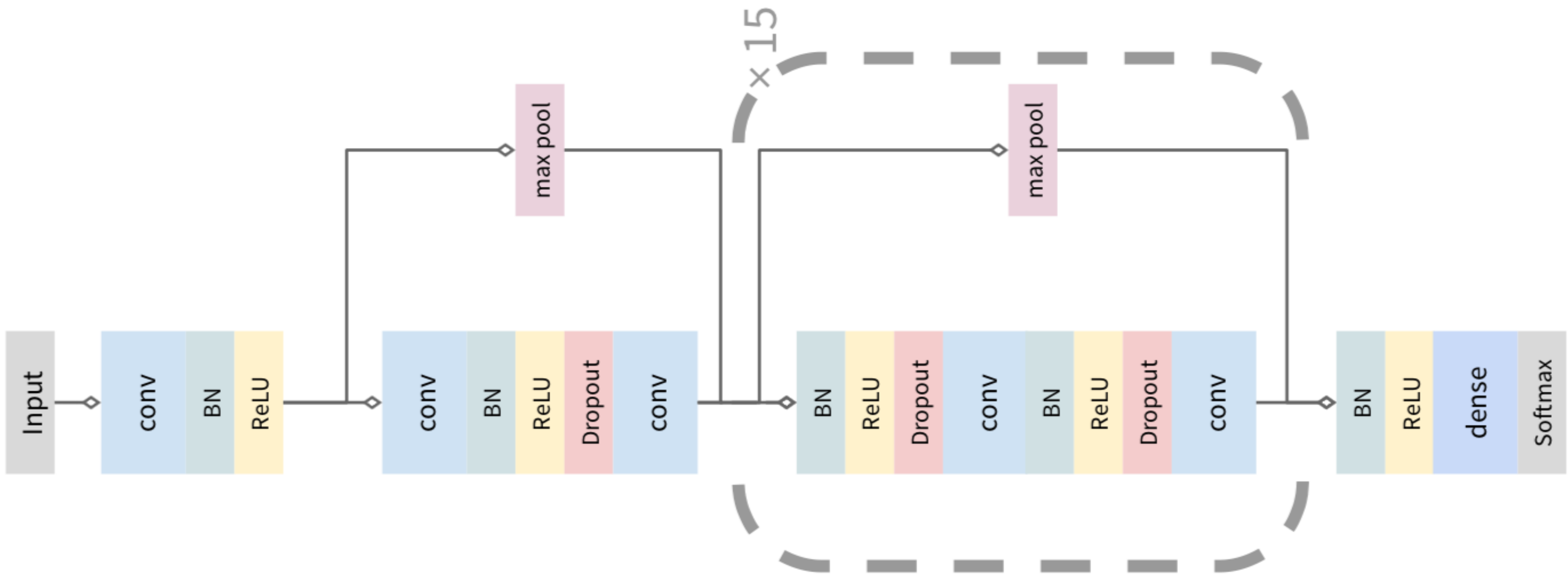


Figure 1. Our trained convolutional neural network correctly detecting the sinus rhythm (SINUS) and Atrial Fibrillation (AFIB) from this ECG recorded with a single-lead wearable heart monitor.

Application 3



33 layers of convolution followed by a fully connected layer

Application 3

The model outperforms the cardiologist

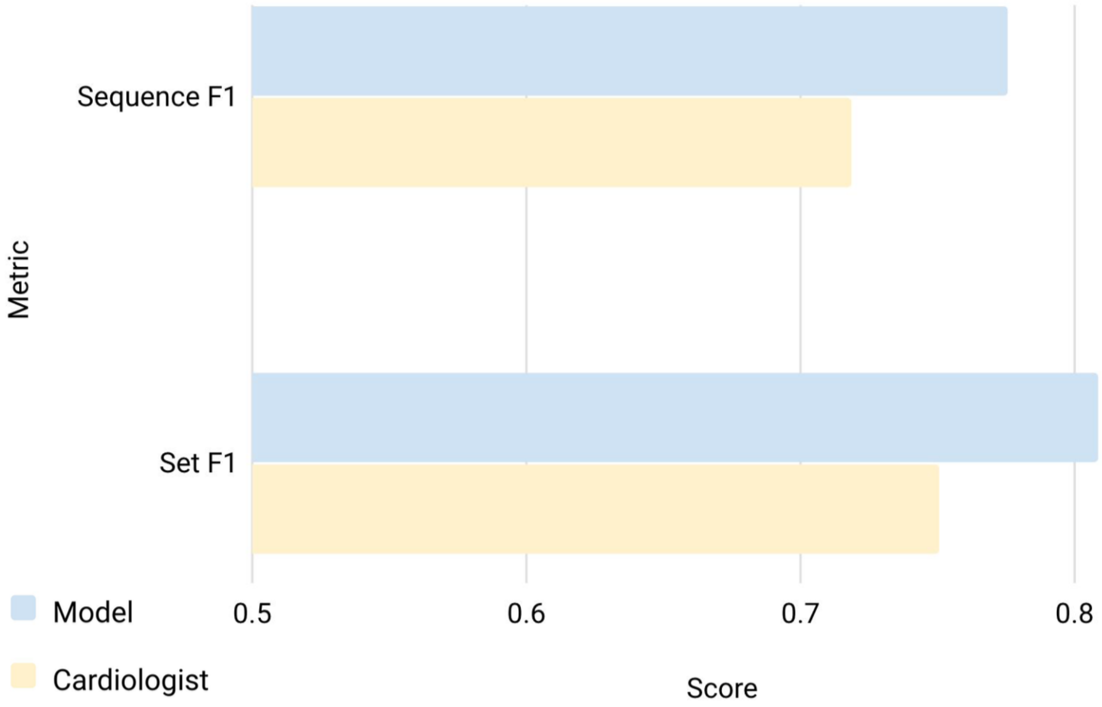
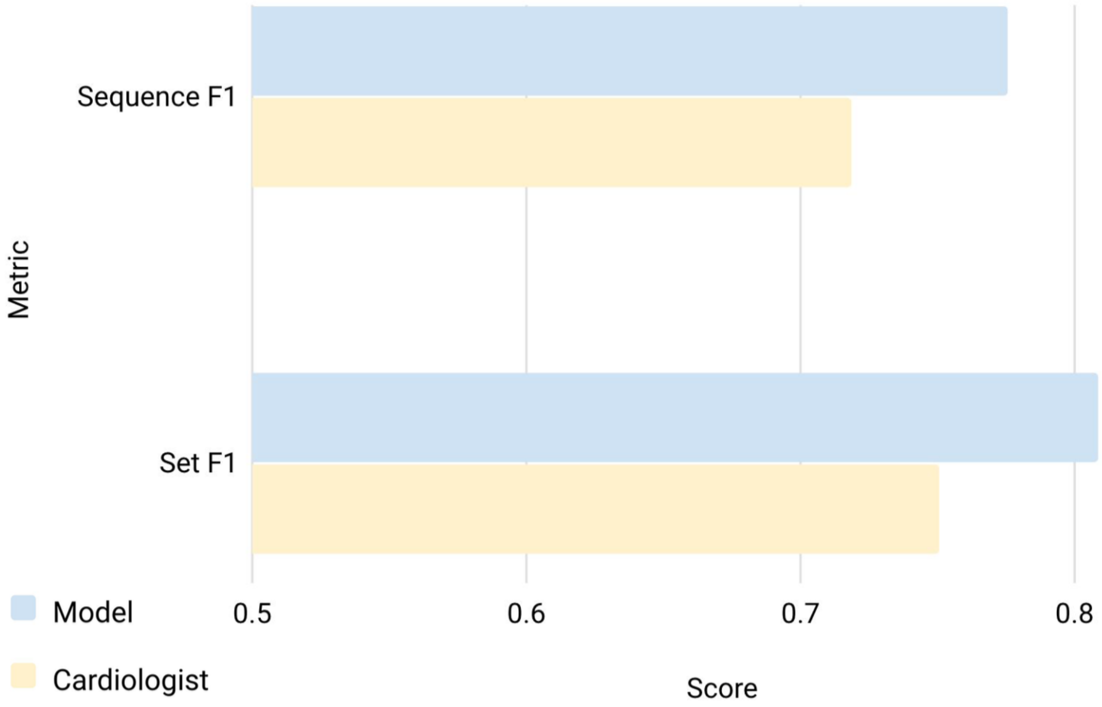


Figure 3. Evaluated on the test set, the model outperforms the average cardiologist score on both the Sequence and the Set F1 metrics.

Application 3

The model outperforms the cardiologist



Every second:
prediction vs truth
Sequence F1 = average

Look in 30 seconds:
prediction vs truth
Set F1 = average

Figure 3. Evaluated on the test set, the model outperforms the average cardiologist score on both the Sequence and the Set F1 metrics.

Application 4

14 Buzz

Algorithmes plus doués que les dermatologues

LOGICIEL Une machine a été capable de détecter 95% des mélanomes sur une série de photos, contre 89% pour l'humain.

Chaque année, 55 000 personnes décèdent d'un mélanome malin. -ISTOCK

Les dermatologues ont du souci à se faire. Un ordinateur a réussi à être meilleur qu'eux pour repérer les cancers de la peau sur des clichés, rapporte la revue «Annals of Oncology». Une équipe germano-franco-américaine a entraîné un système d'intelligence artificielle à distinguer des lésions de la peau et grains de beauté selon qu'ils étaient bénins ou alarmants, en lui montrant plus de 100 000 images. Les performances de la machine (un réseau neuronal convolutif) ont ensuite été comparées à celles de 58 médecins spécialistes de 17 pays. Résultat: «La plupart des dermatologues ont fait moins bien», écrivent les chercheurs.

Confrontés à 100 photos de cas jugés compliqués, les médecins ont correctement identifié 87% des mélanomes qui leur étaient présentés. Quand ils obtenaient des images en plus gros plan et des infos plus détaillées (âge, sexe du patient, position de la lésion cutanée, par exemple), ce taux montait à 89%. Mais la machine a fait mieux, avec 95% de mélanomes détectés dès la première série de photos.

Pour les chercheurs, la question n'est pas de se passer des médecins au profit de l'intelligence artificielle, mais de faire d'elle «un outil supplémentaire». «Aujourd'hui rien ne remplace un examen clinique approfondi», ont rappelé dans l'étude deux professeurs australiens en dermatologie. -ATS

QUESTIONS ?

BONUS

Bonus 1: Bias node

A simple linear regression model:

$$y_i = \alpha + \beta \cdot x_i + \varepsilon_i$$

where α is called the intercept parameter

Bonus 1: Bias node

In neural network,

the intercept parameter α

is introduced via the bias node