An overview of Recurrent Neural Networks

Jeremy Watt

Features and supervised learning

- Some data types have **independent features**

Features and supervised learning

- Some data types have **independent features**
- **Order in which we feed them in doesn’t matter**

Race	Gender	Diabetes	Weight	Readmitted
caucasian	male	No	188	No

Features and supervised learning

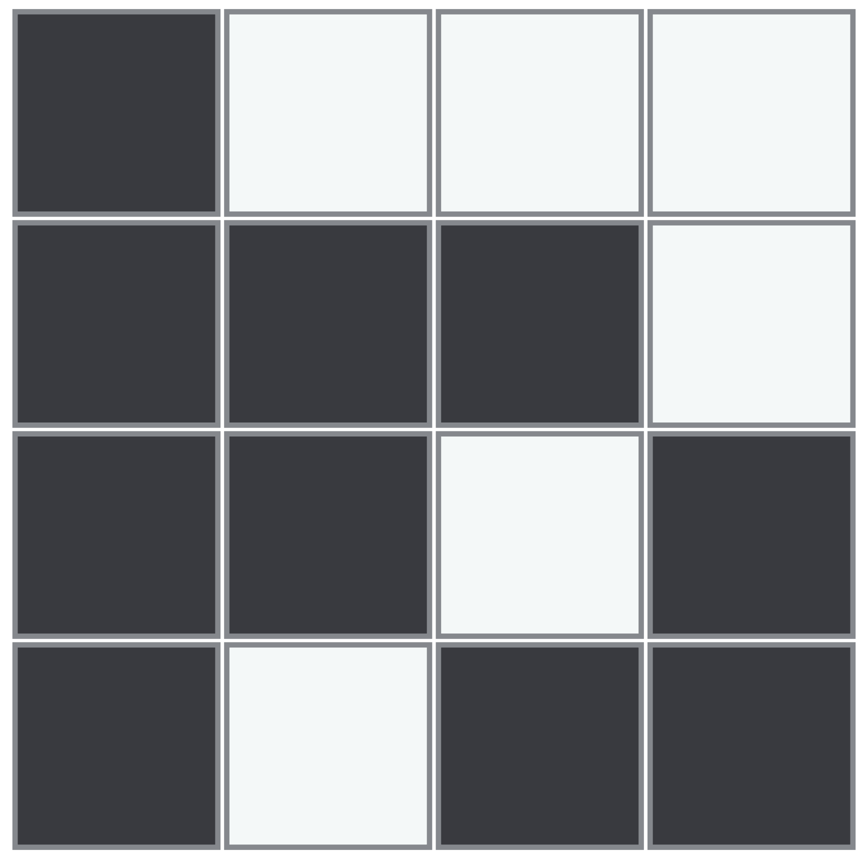
- Some data types have **spatially correlated features**

Features and supervised learning

- Some data types have **spatially correlated features**
- **Order in which we feed them in definitely matters**

Features and supervised learning

- Some data types have **spatially correlated features**
- **Order in which we feed them in definitely matters**
- Convolutional modeling can leverage this



Features and supervised learning

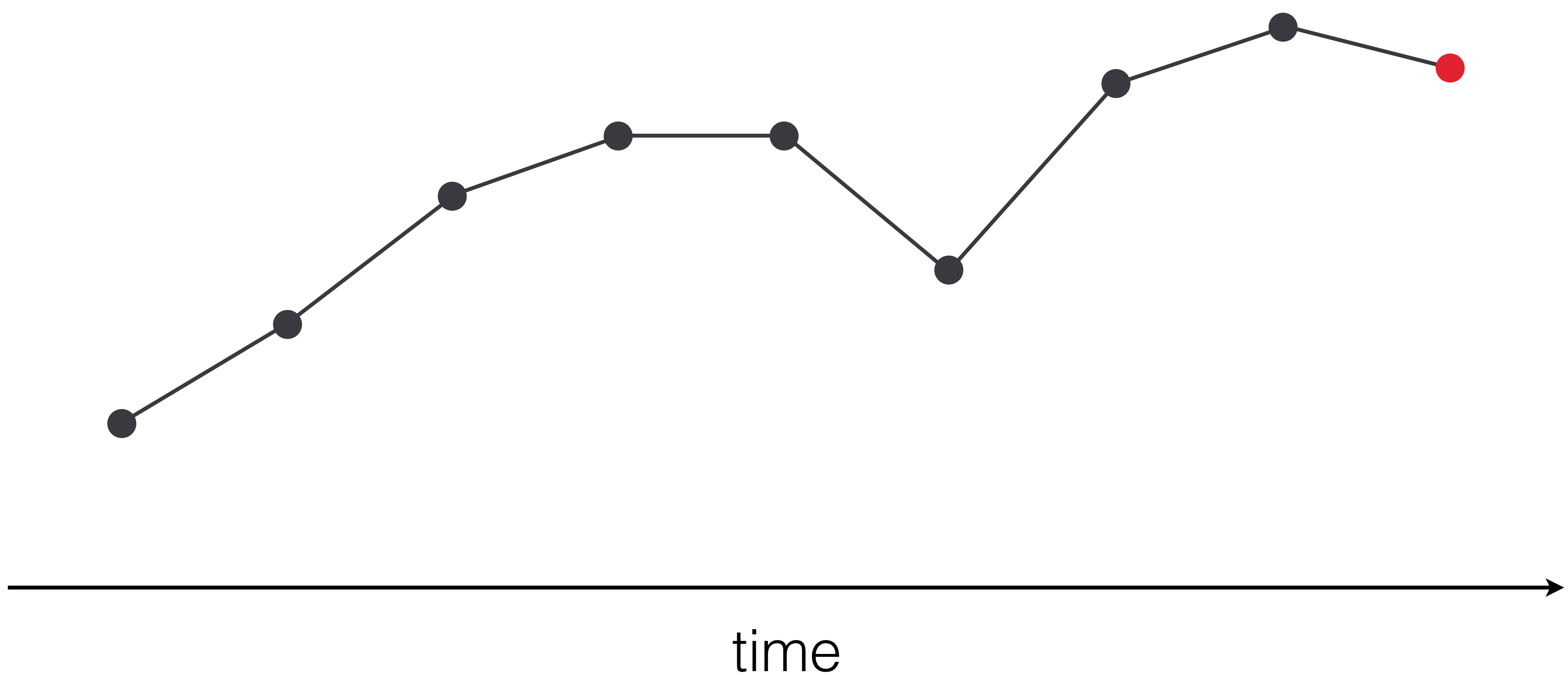
- Some data types have **temporarly ordered features**

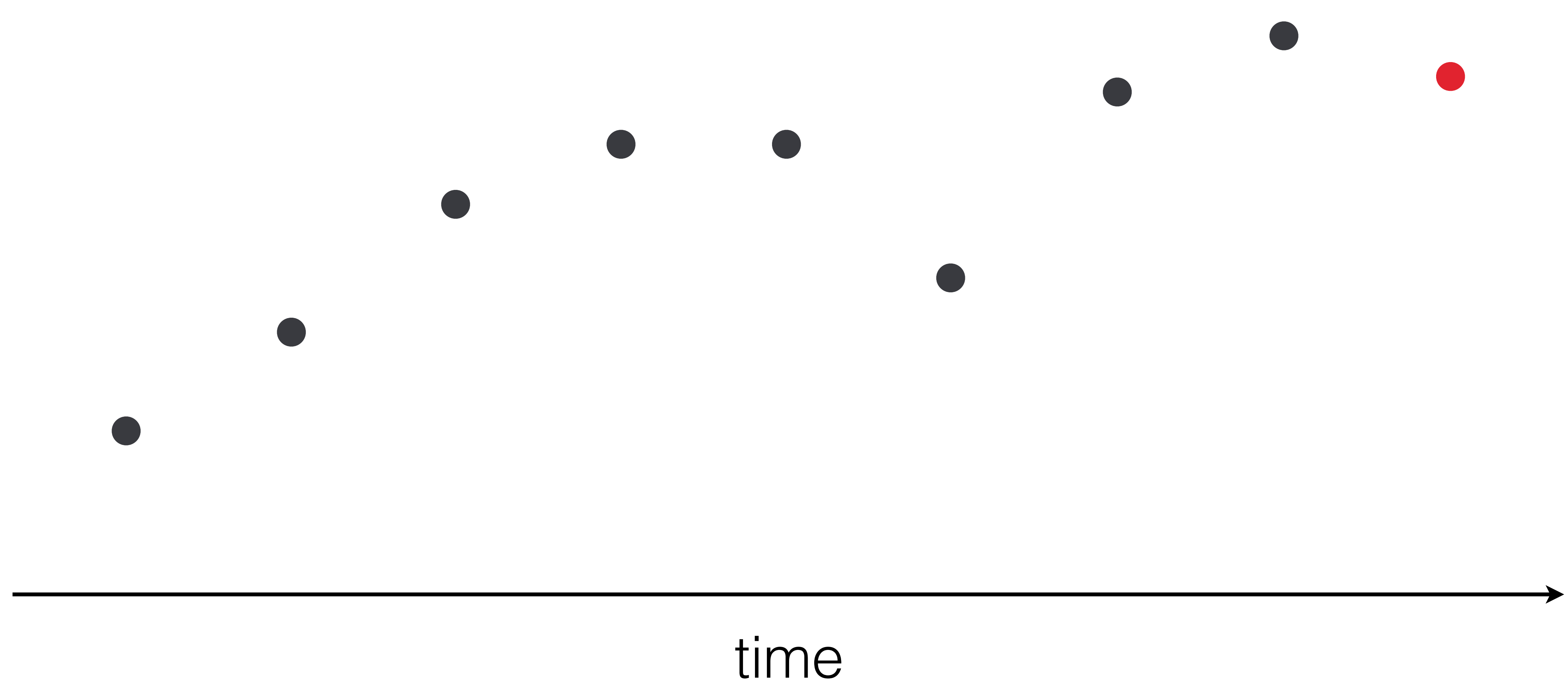
Features and supervised learning

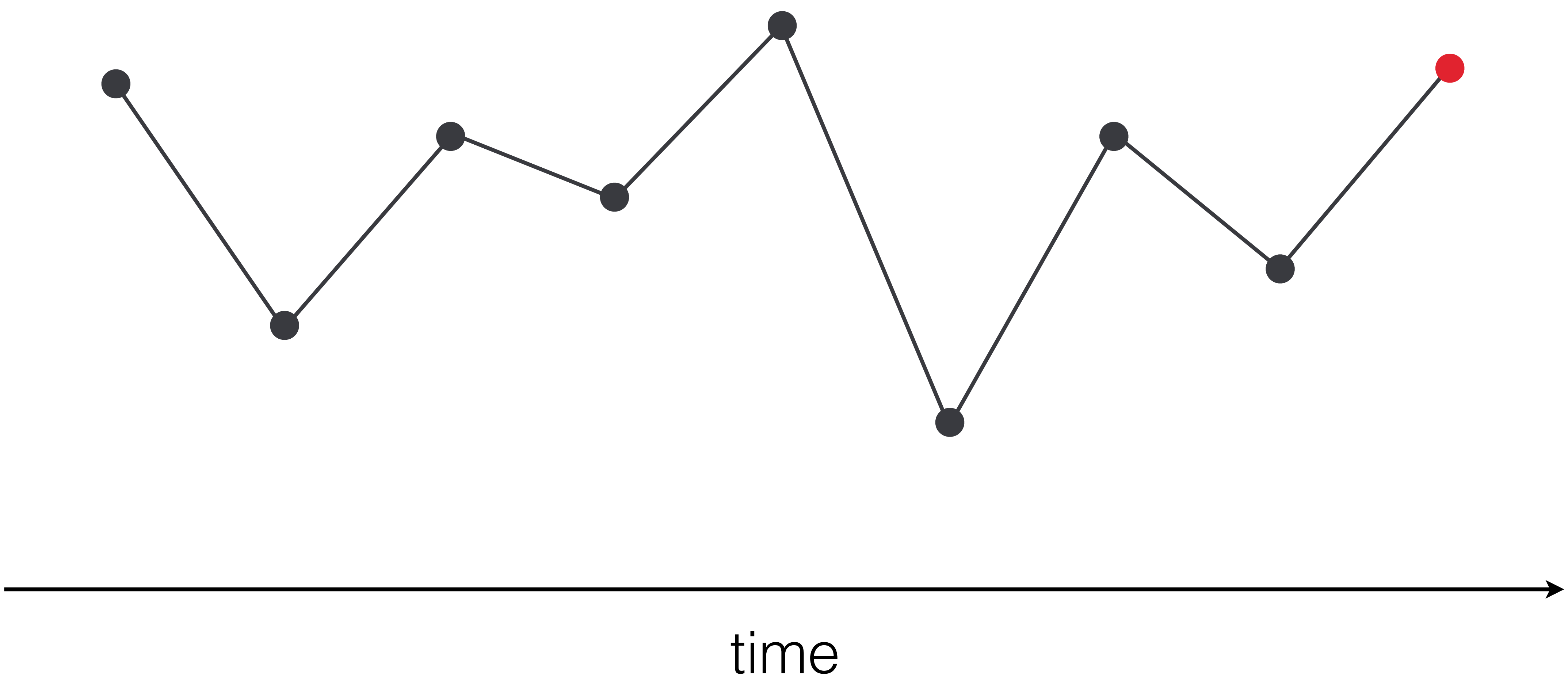
- Some data types have **temporarily ordered features**
- **Order in which we feed them in definitely matters**

Features and supervised learning

- Some data types have **temporarly ordered features**
- **Order in which we feed them in definitely matters**
- **Recursive modeling can leverage this** (e.g., RNNs)


time

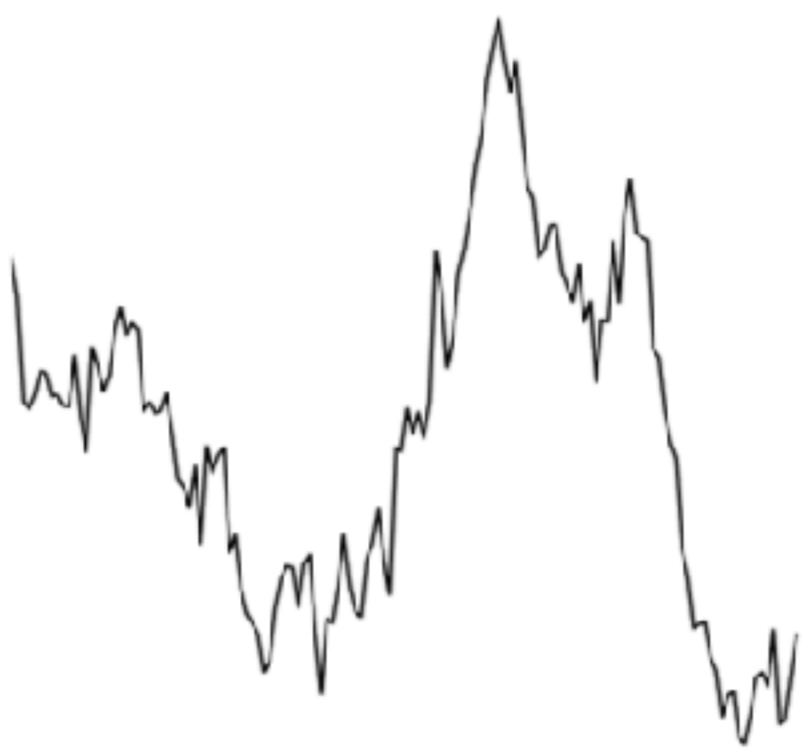

time


time

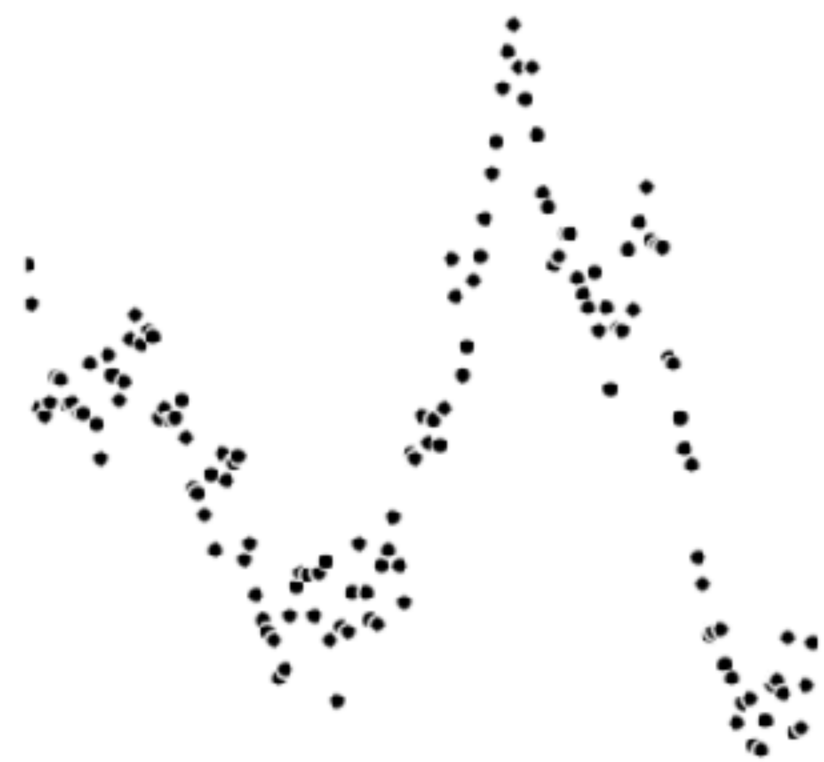
Popular problems with ordered sequential I/O

Time series prediction

- predict future values of a time series
- **input:** ordered sequence of past series values
- **output:** ordered sequence of future series values



series shown interpolated



series really looks like this

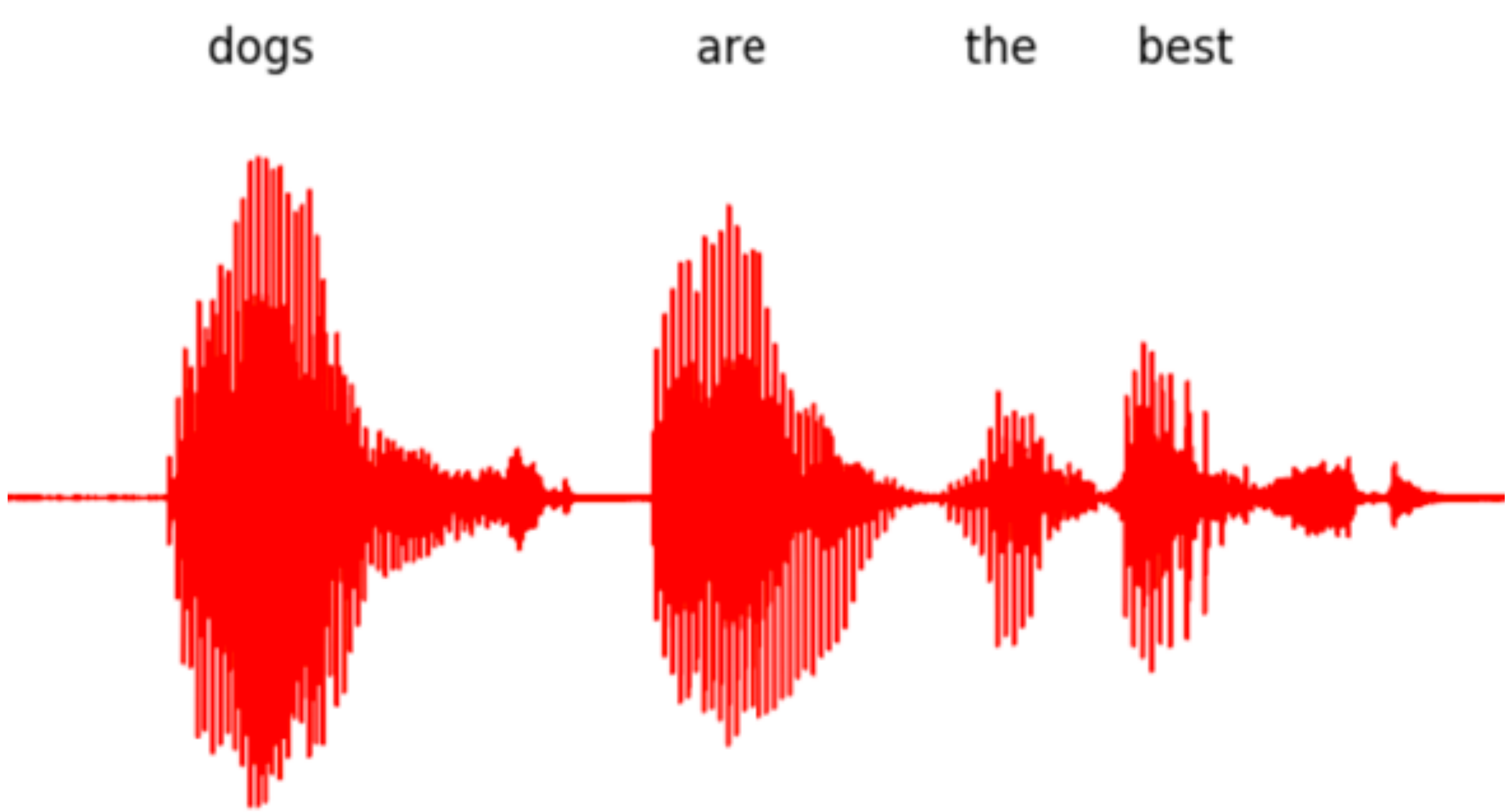
Machine translation

- Translate language X into language Y
- **input:** ordered sequence of words (in X)
- **output:** ordered sequence of words (in Y)

  - “I do not like cats” —> “Los gatos me caen mal”

Speech recognition

- Speech to text
- **input:** ordered sequence of raw audio
- **output:** ordered sequence of words



Text generation

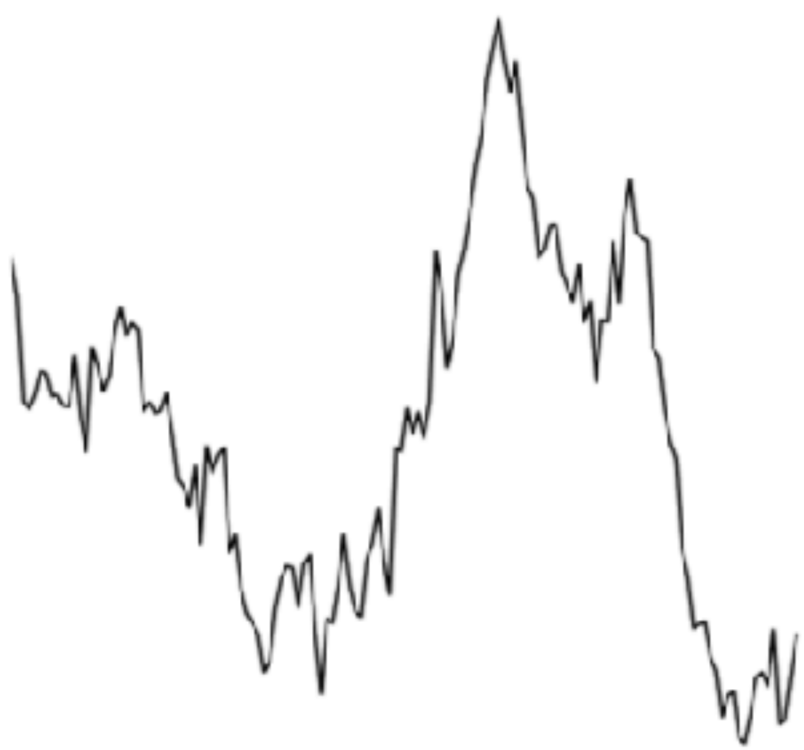
- generate valid but wacky text automatically
- **input:** ordered sequence of characters (training text corpus)
- **output:** ordered sequence of characters

- Generate wacky sentences with this academic RNN text generator
- Various twitter bots that tweet automatically generated text like this one.
- NanoGenMo annual contest to automatically produce a 50,000+ novel automatically
- Robot Shakespear a text generator that automatically produces Shakespear-esk sentences

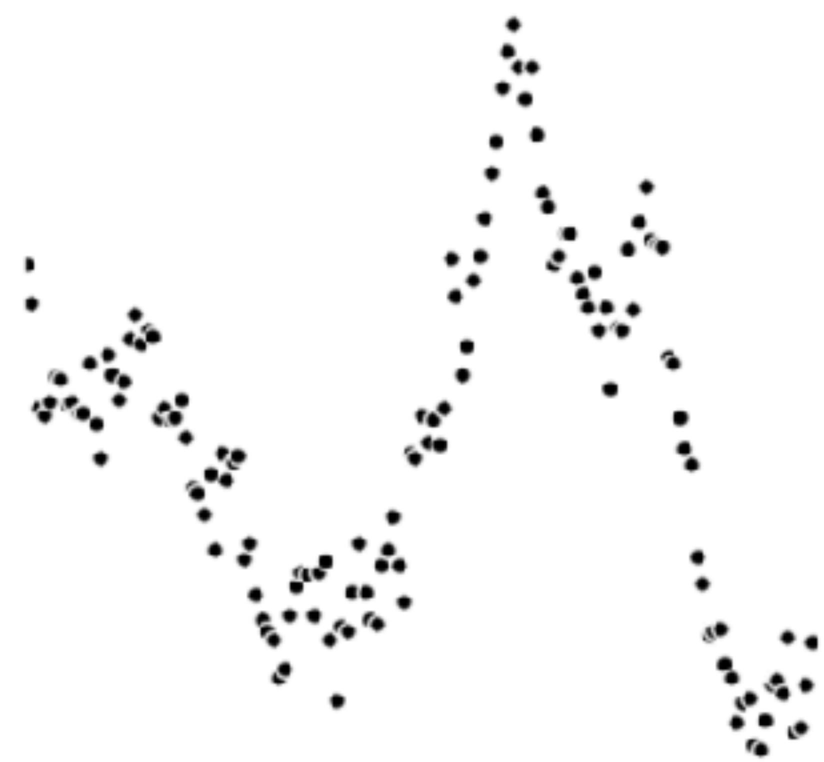
Ingesting sequential I/O data for supervised learning

Time series prediction

- Sequence of P (floating point) numbers: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$



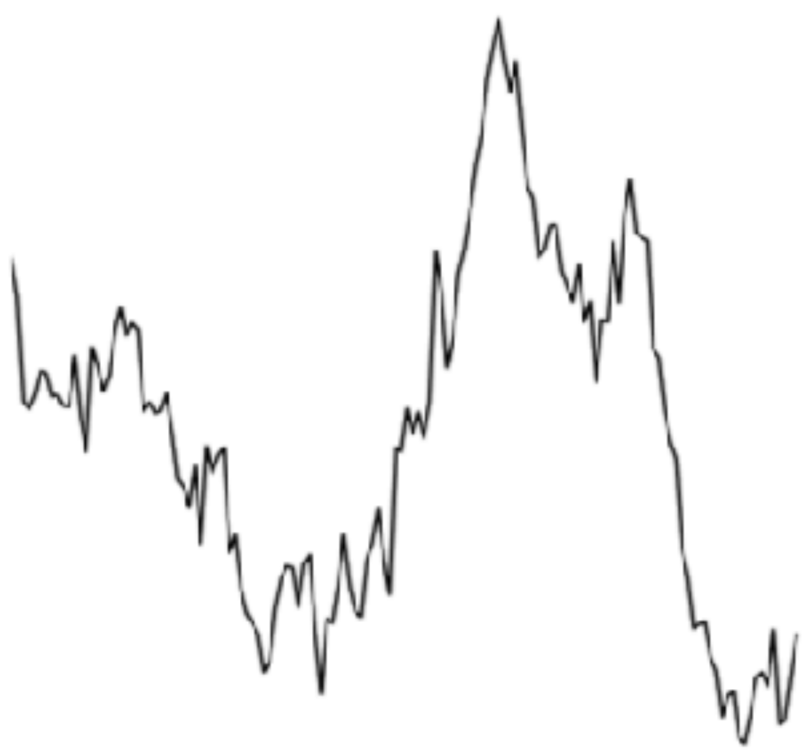
series shown interpolated



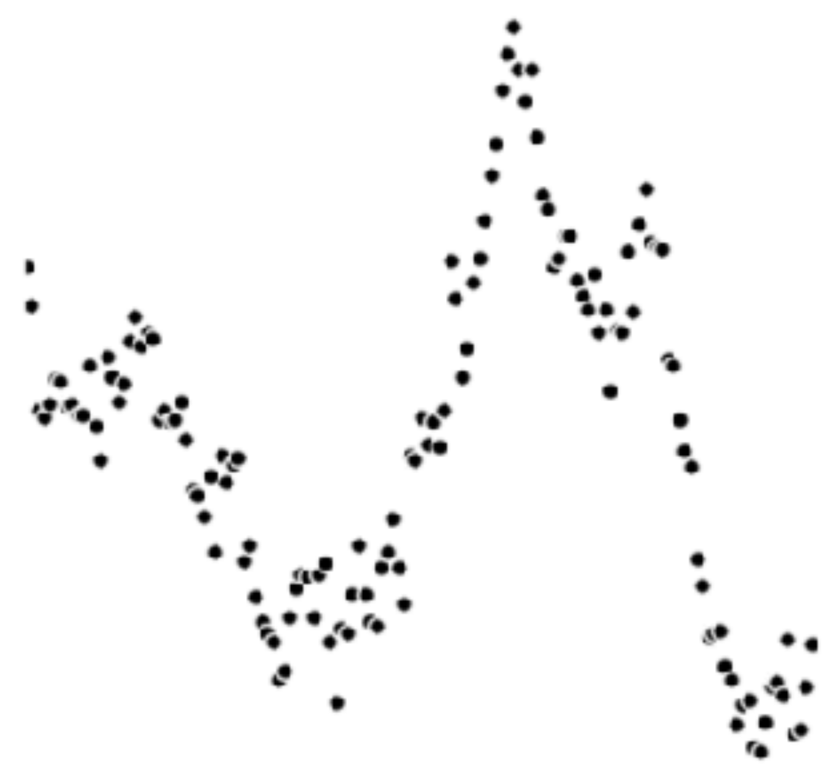
series really looks like this

Time series prediction

- Sequence of P (floating point) numbers: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p: the pth value



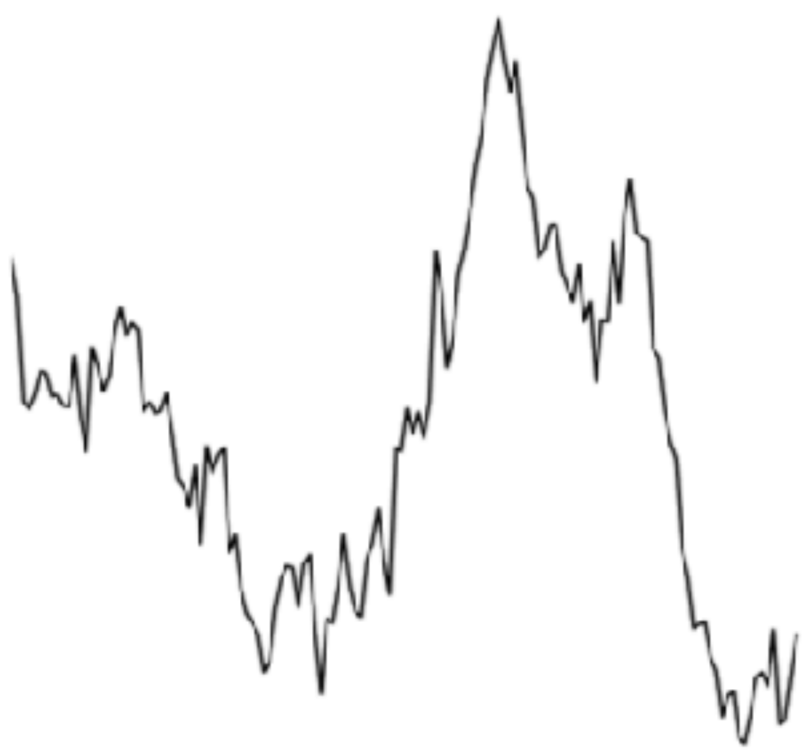
series shown interpolated



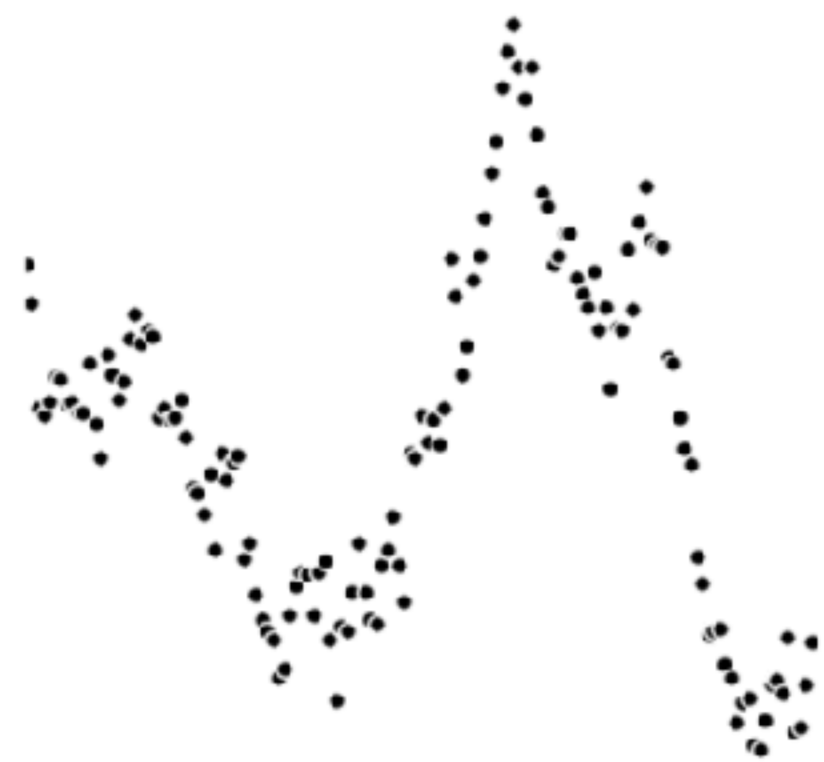
series really looks like this

Time series prediction

- Sequence of P (floating point) numbers: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p : the pth value
- **regression problem**- I/O are windowed subsequences



series shown interpolated



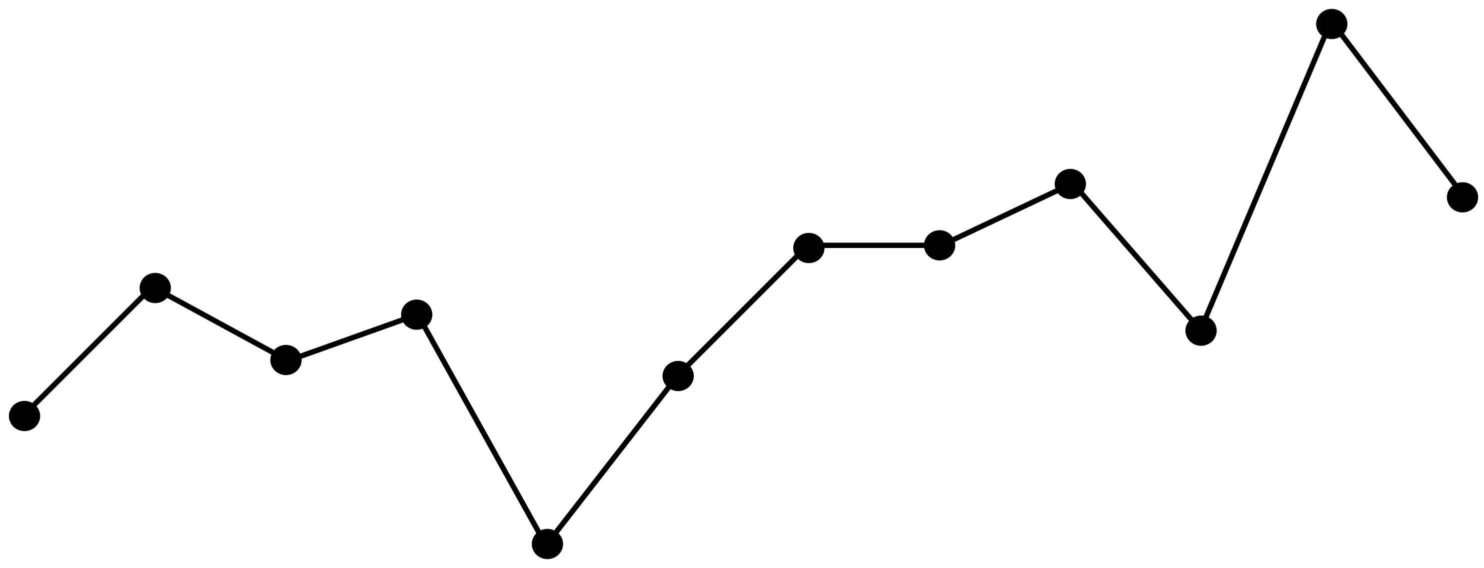
series really looks like this

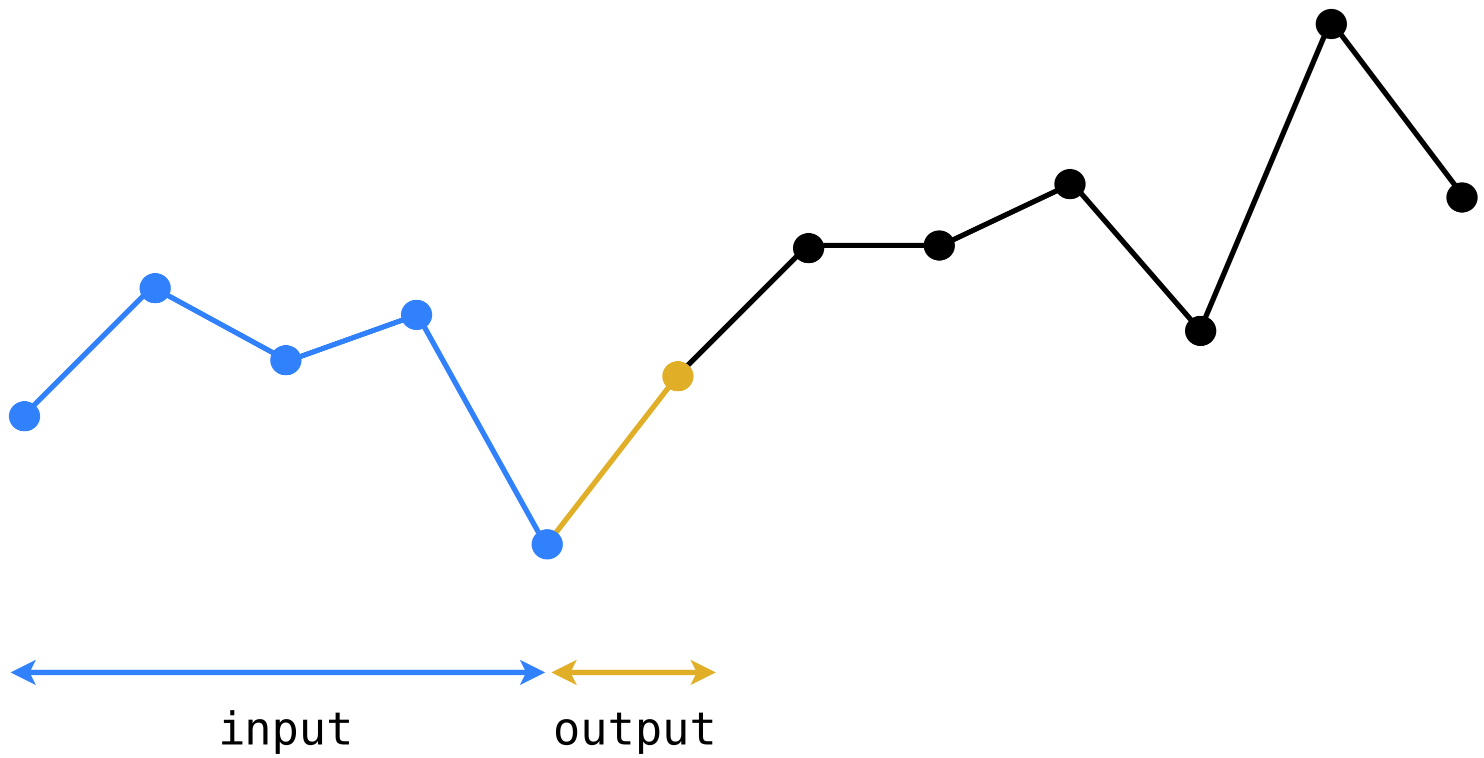
Time series prediction

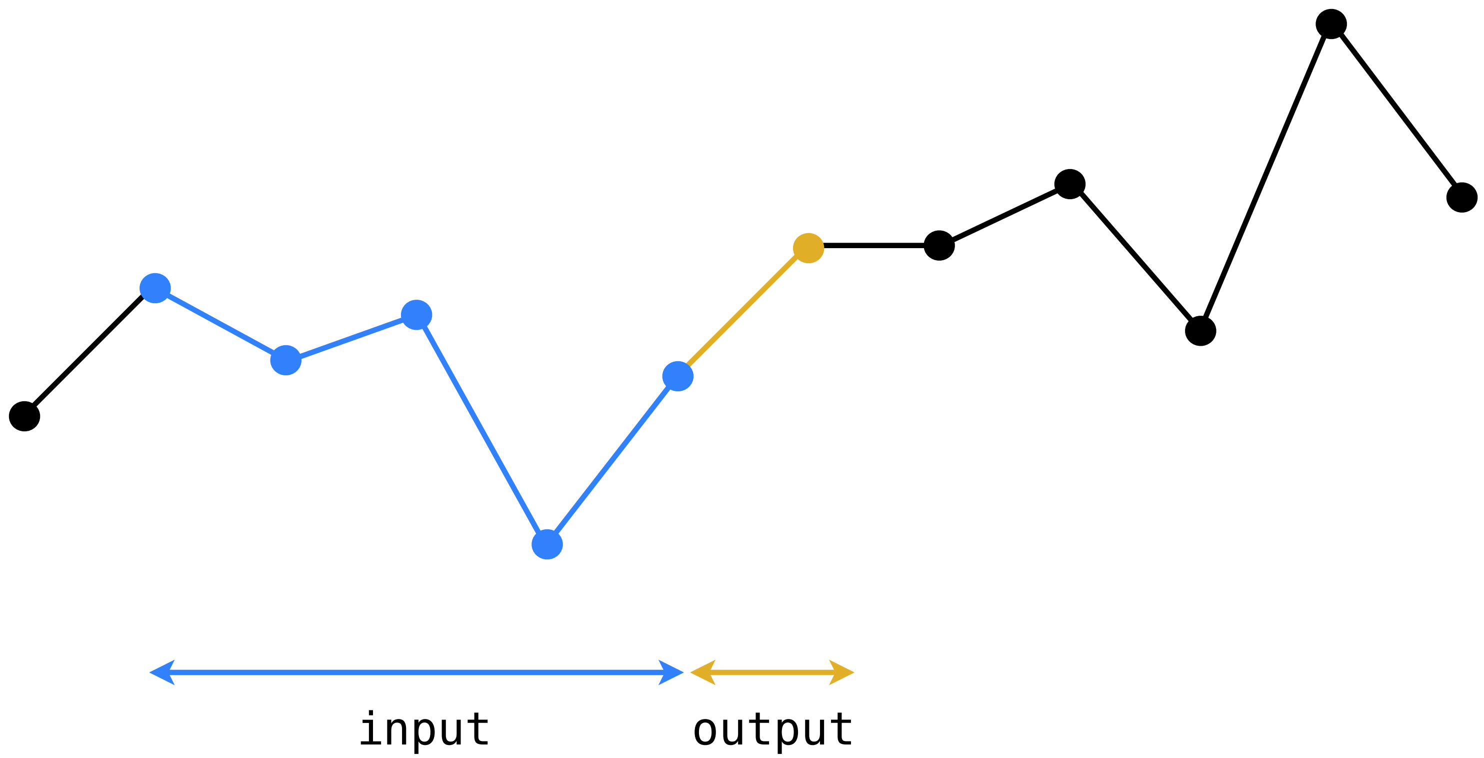
- Sequence of P (floating point) numbers: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p : the pth value
- - **regression problem**- I/O are windowed subsequences

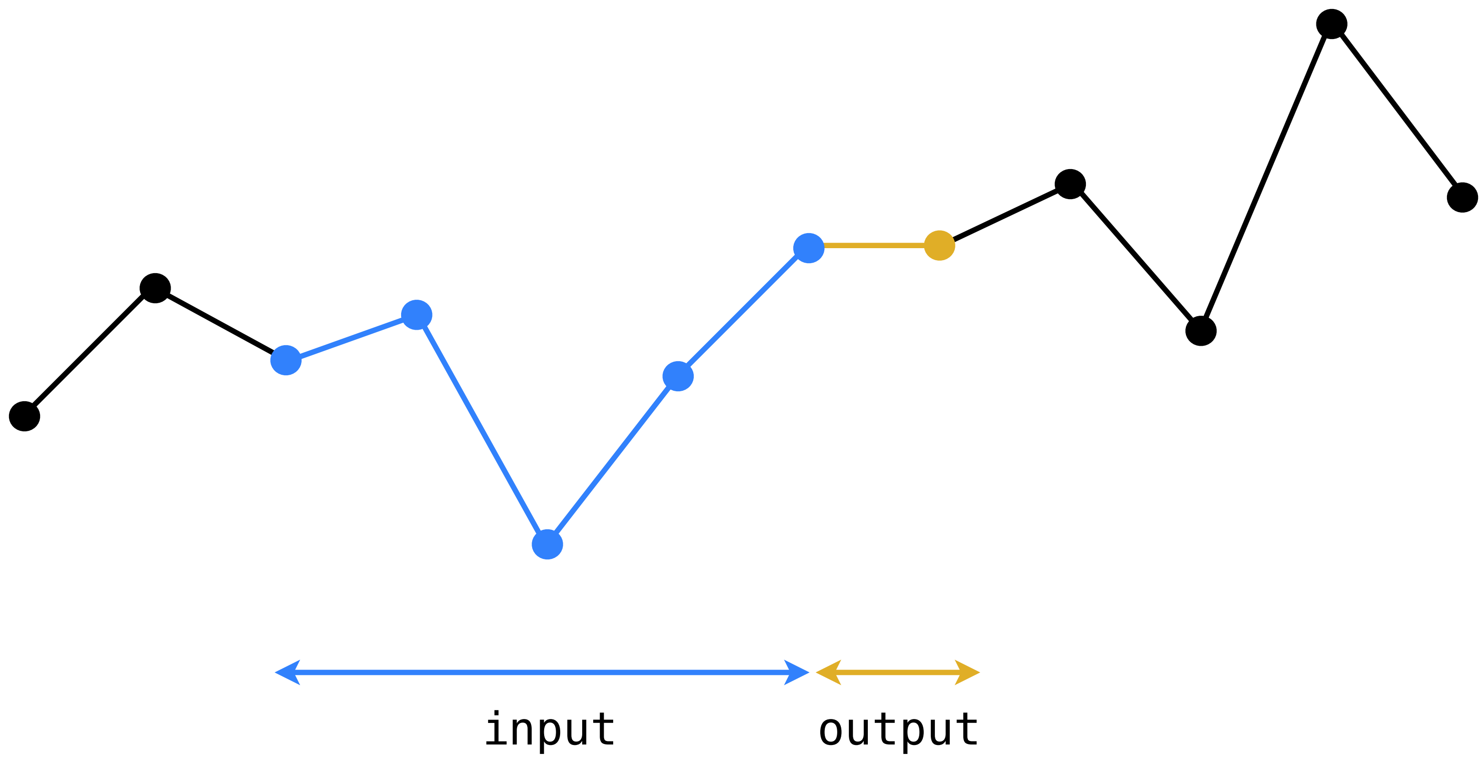
  - so need training and testing sets

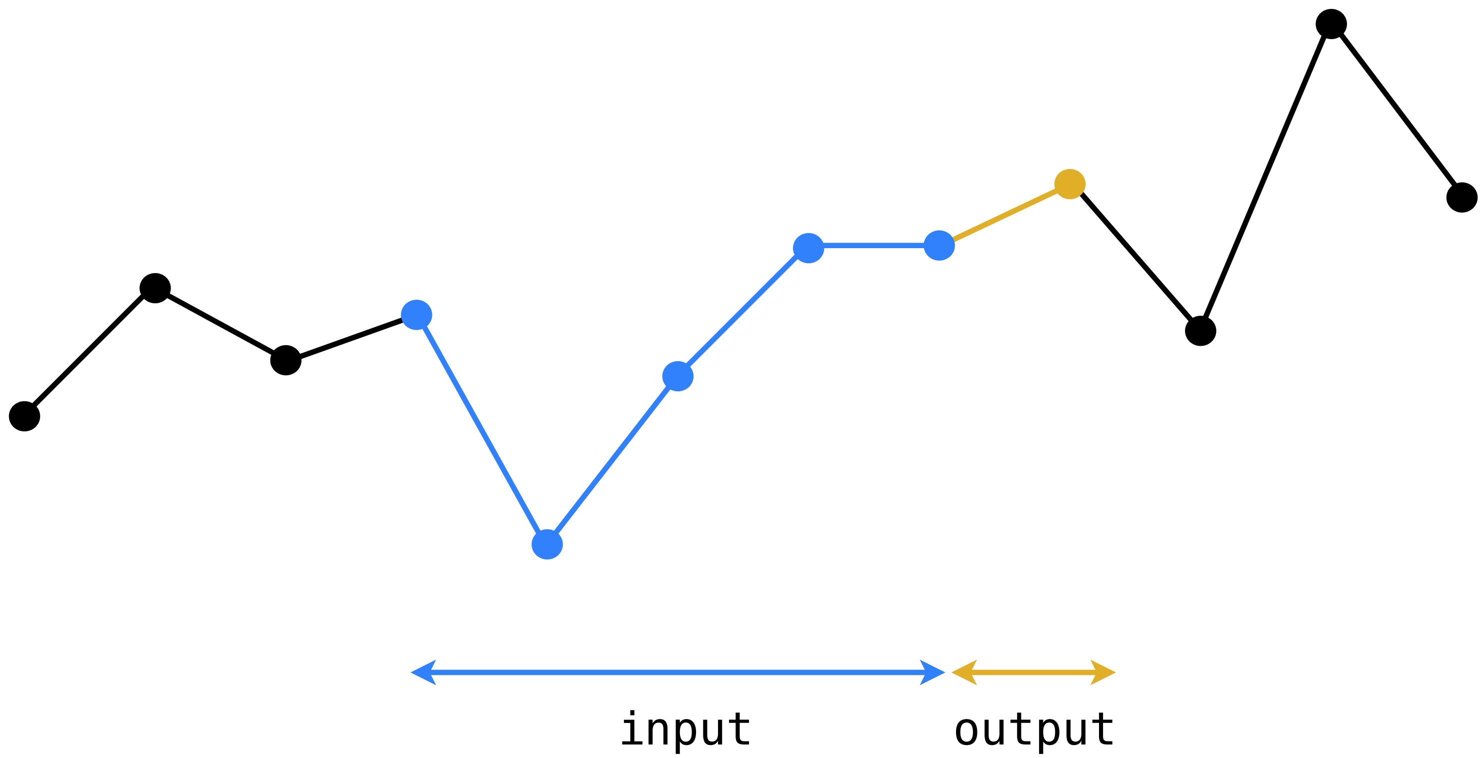
  - lets see how both are formed, start with training

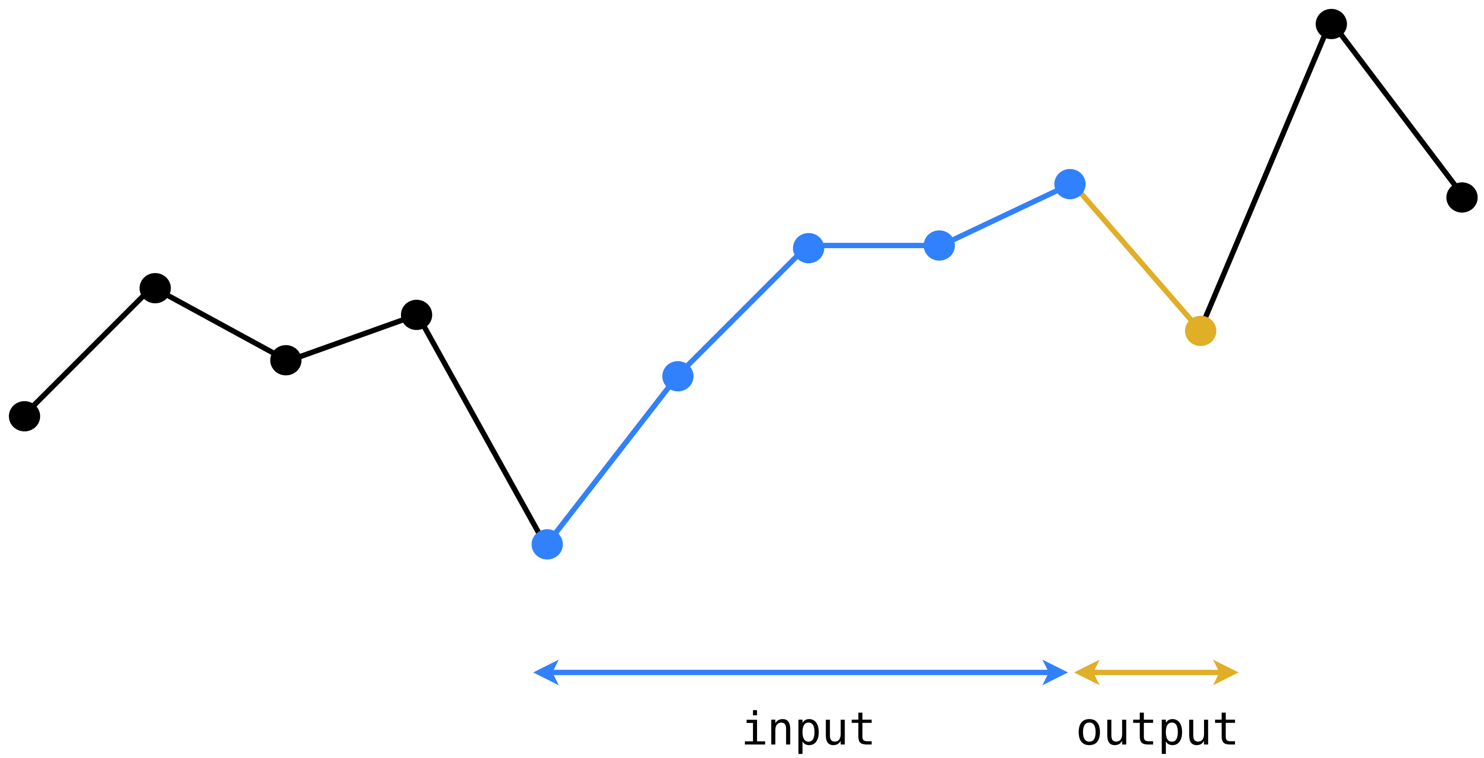


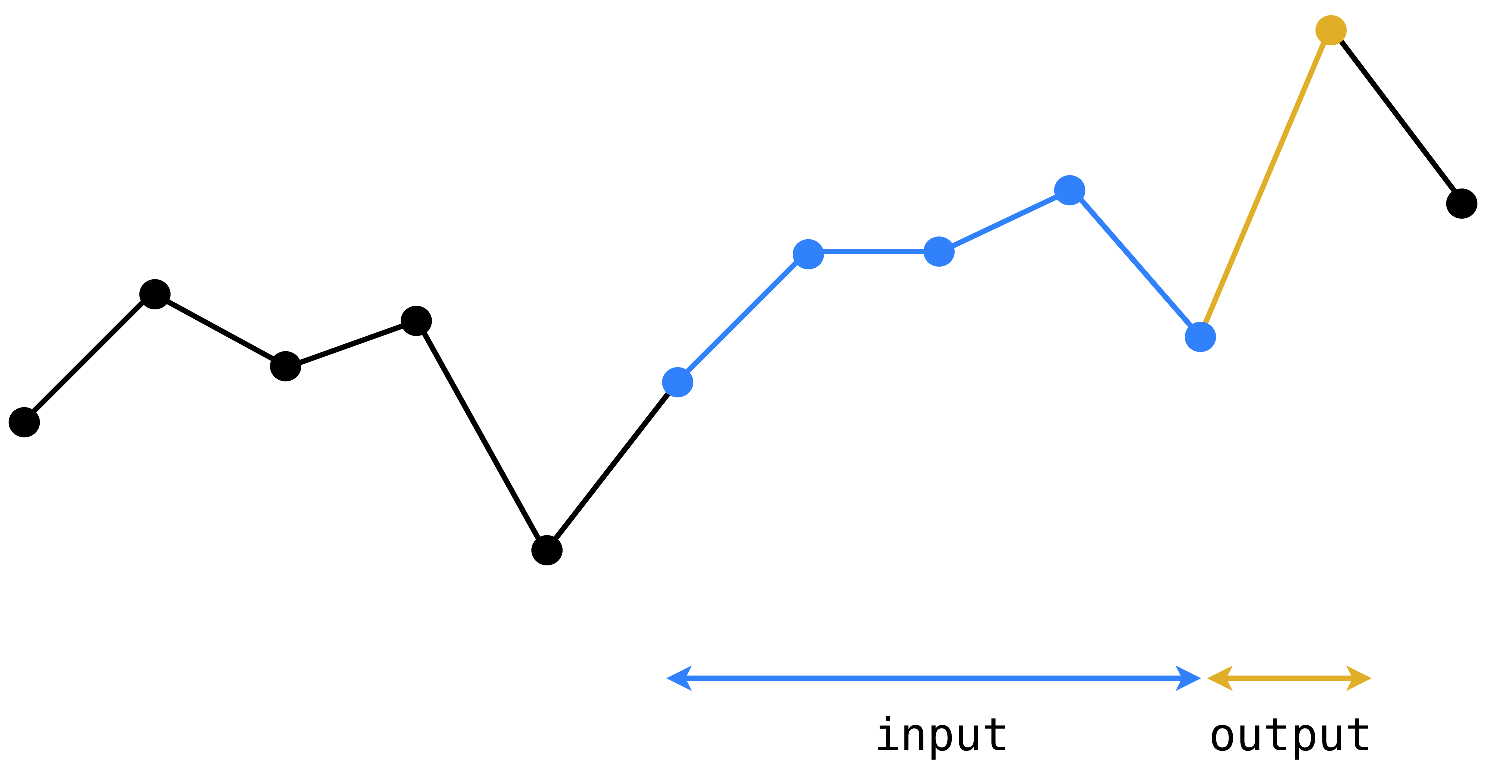

input
output

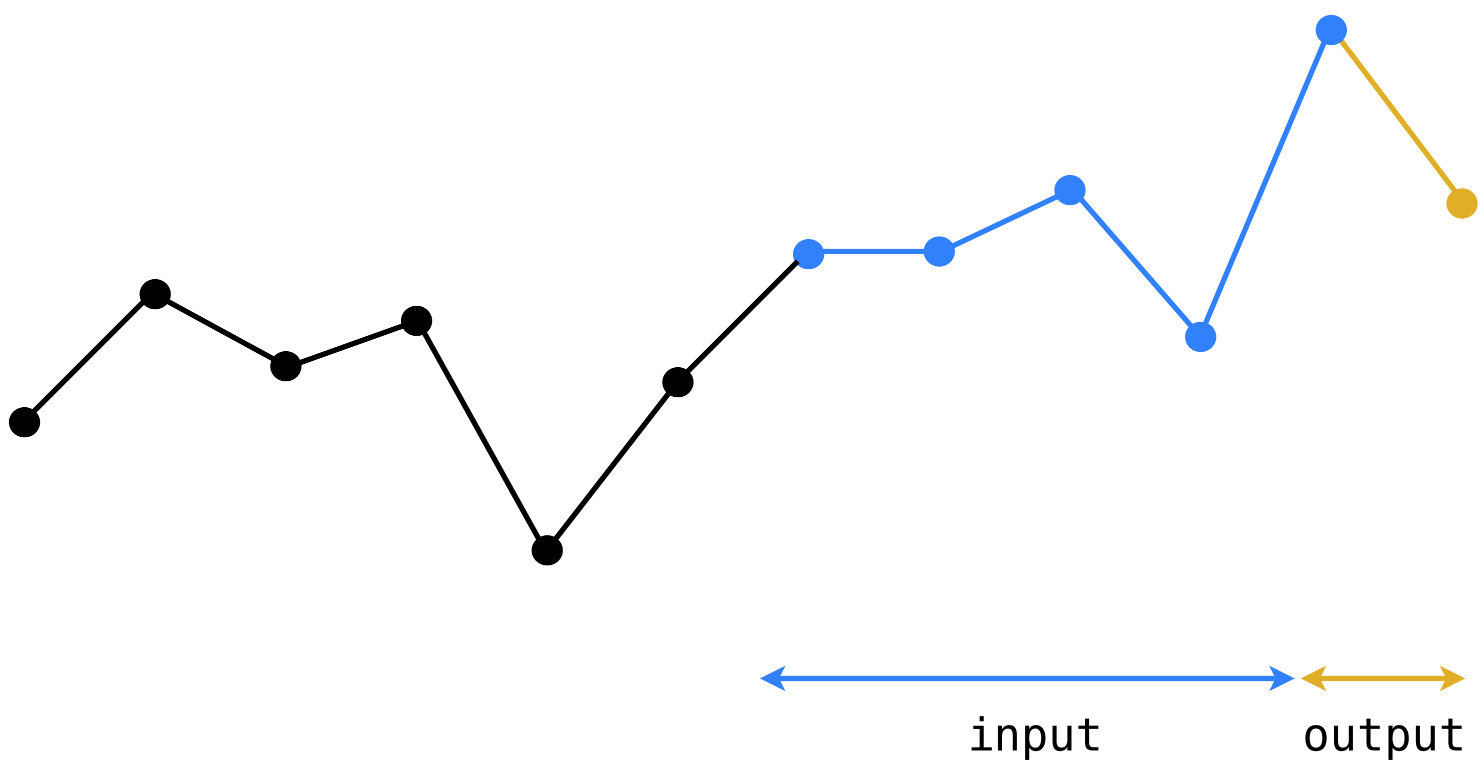

input
output

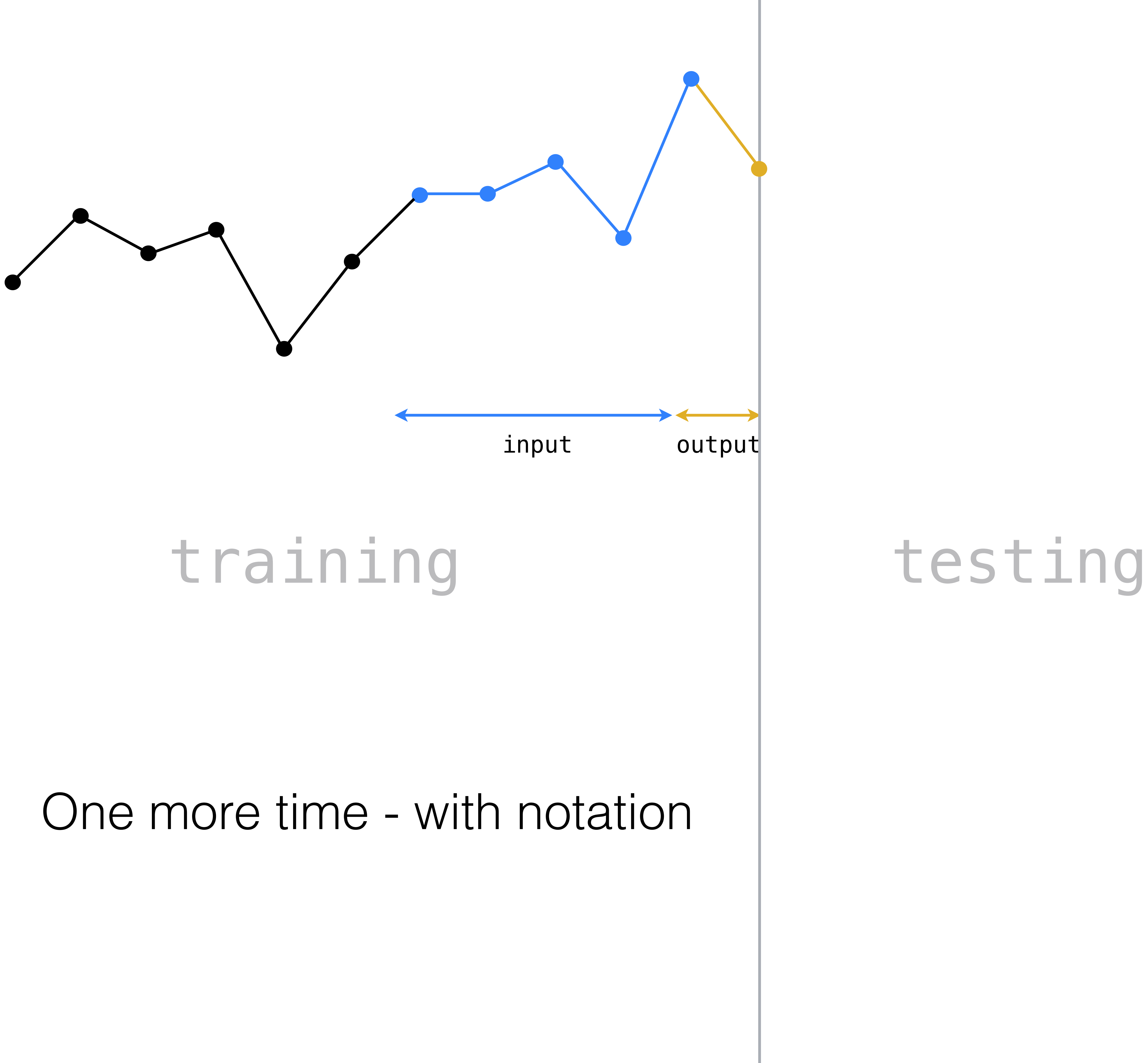

input
output


input
output

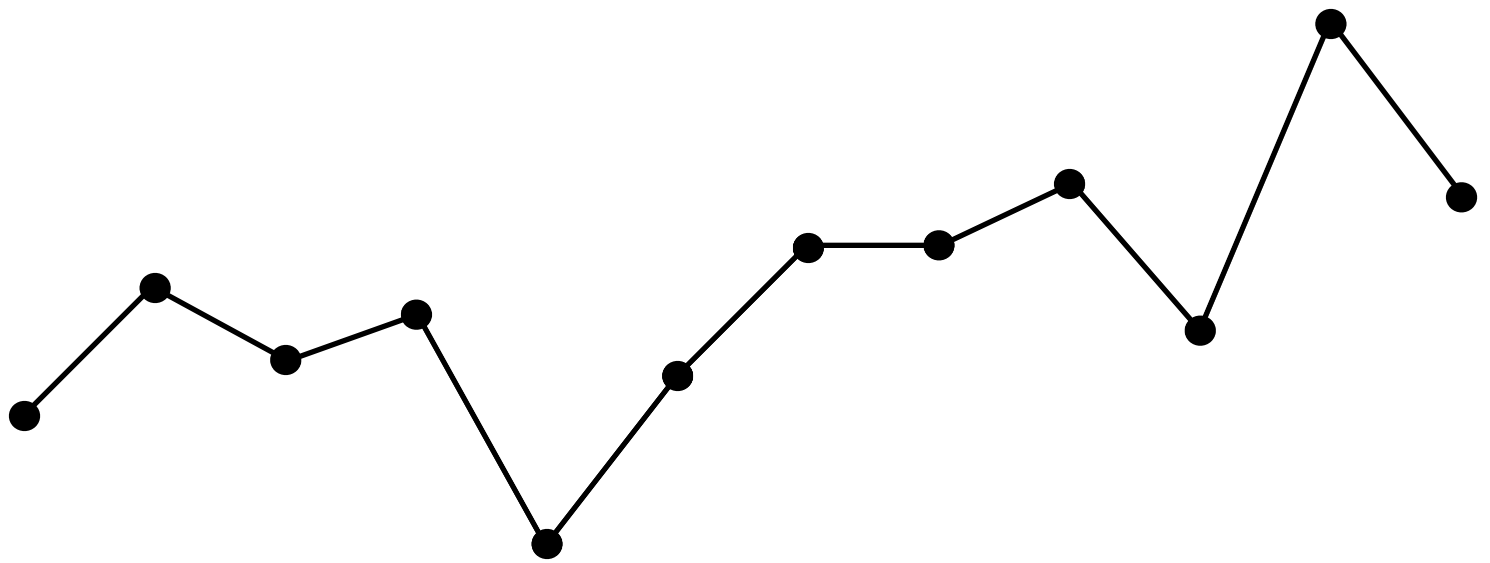

input
output

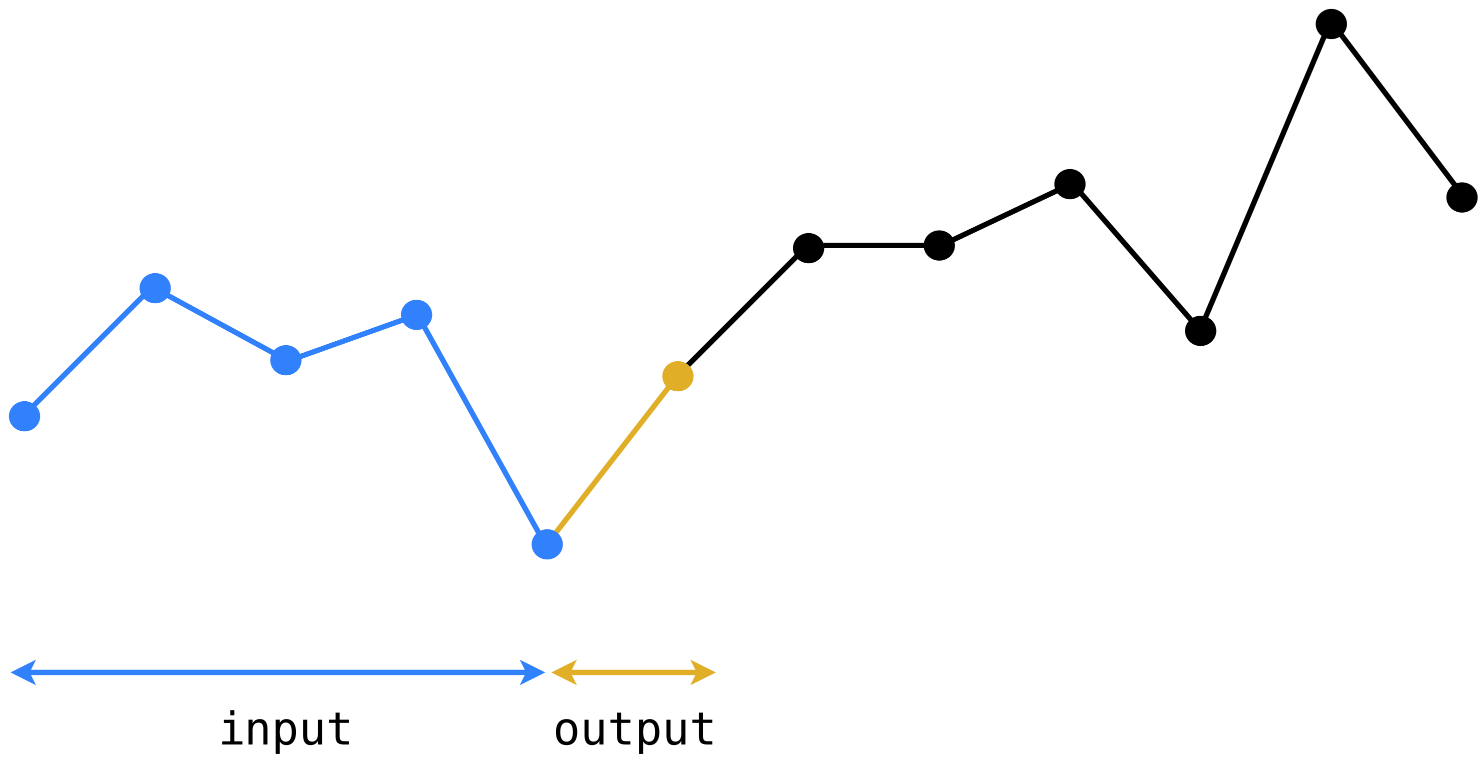

input
output


input
output

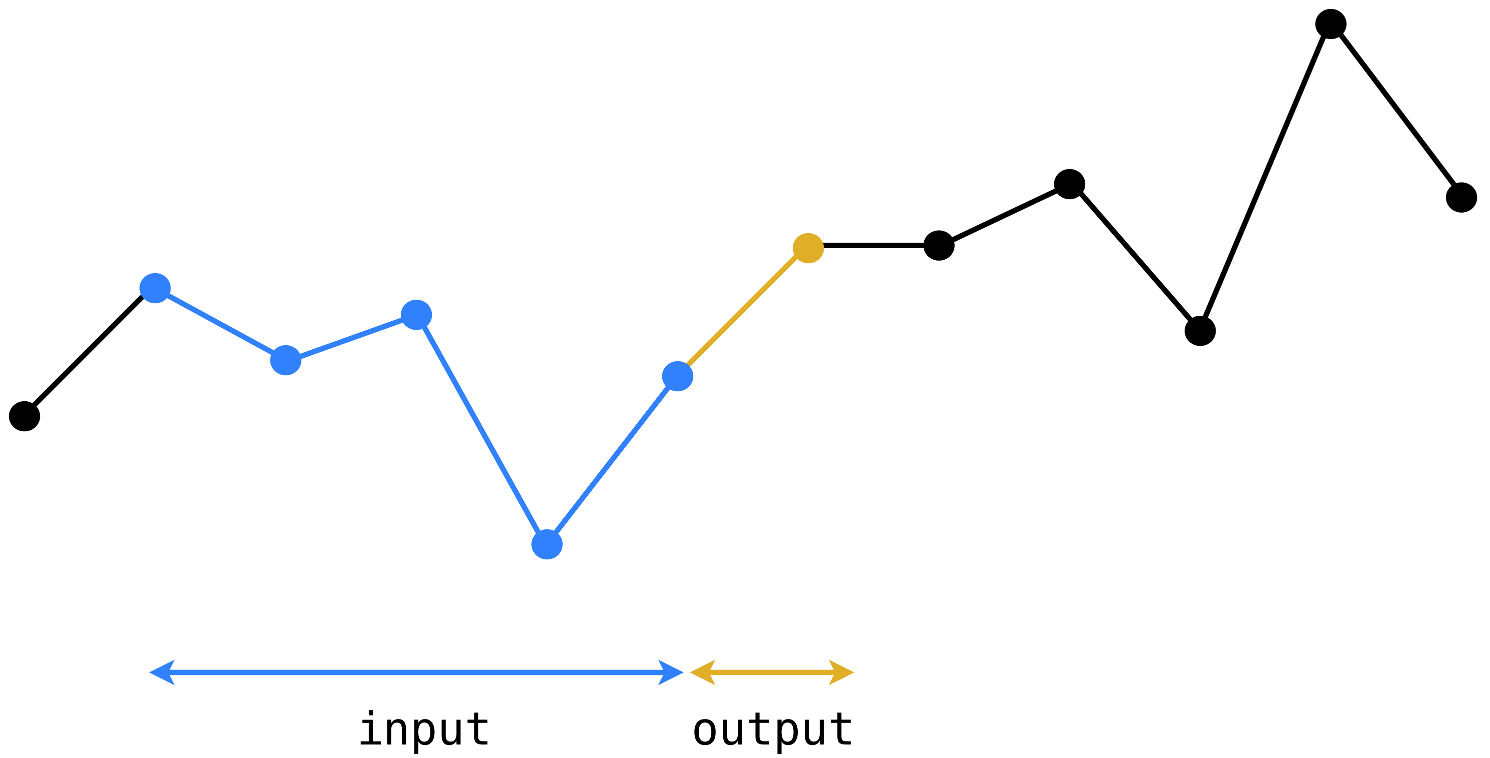


One more time - with notation

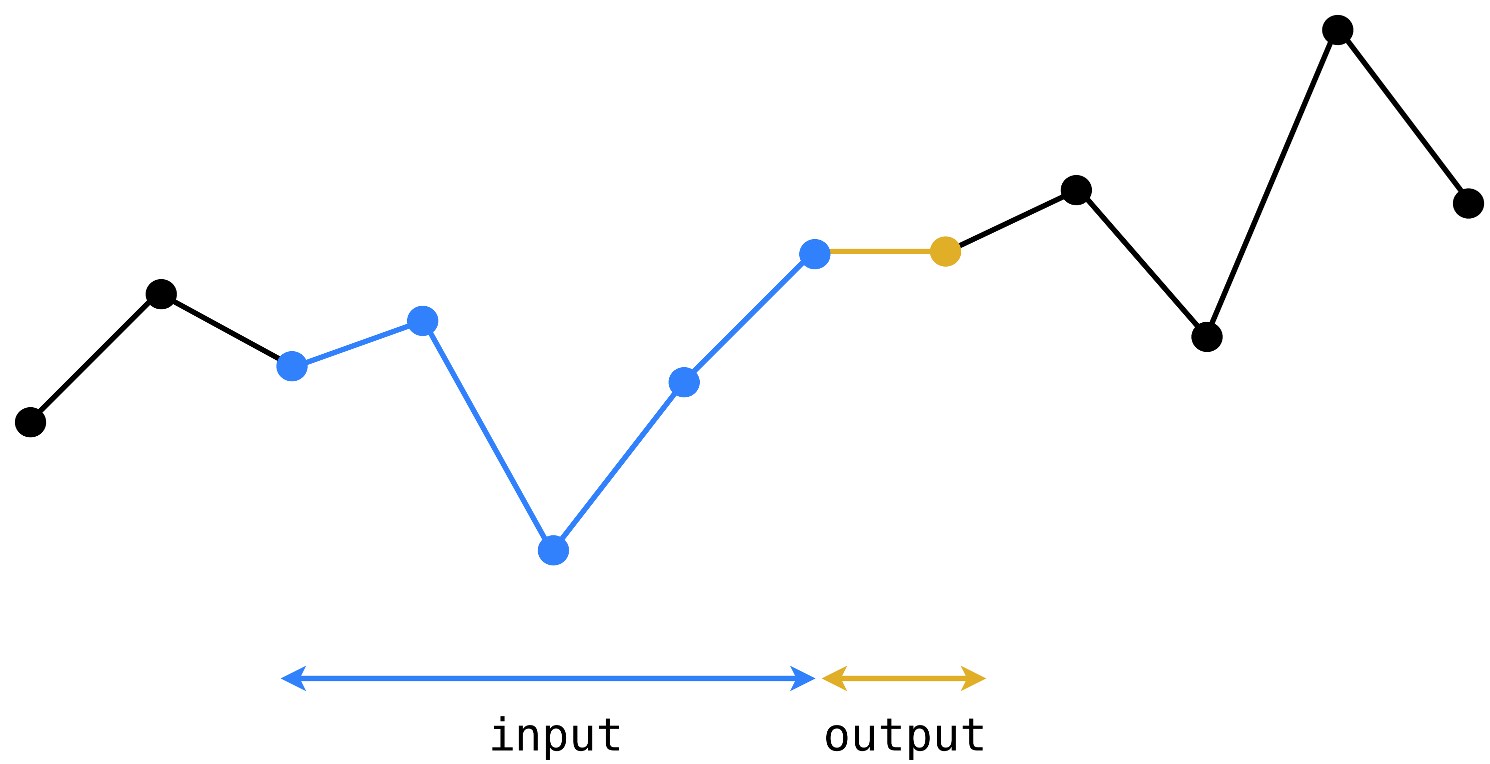




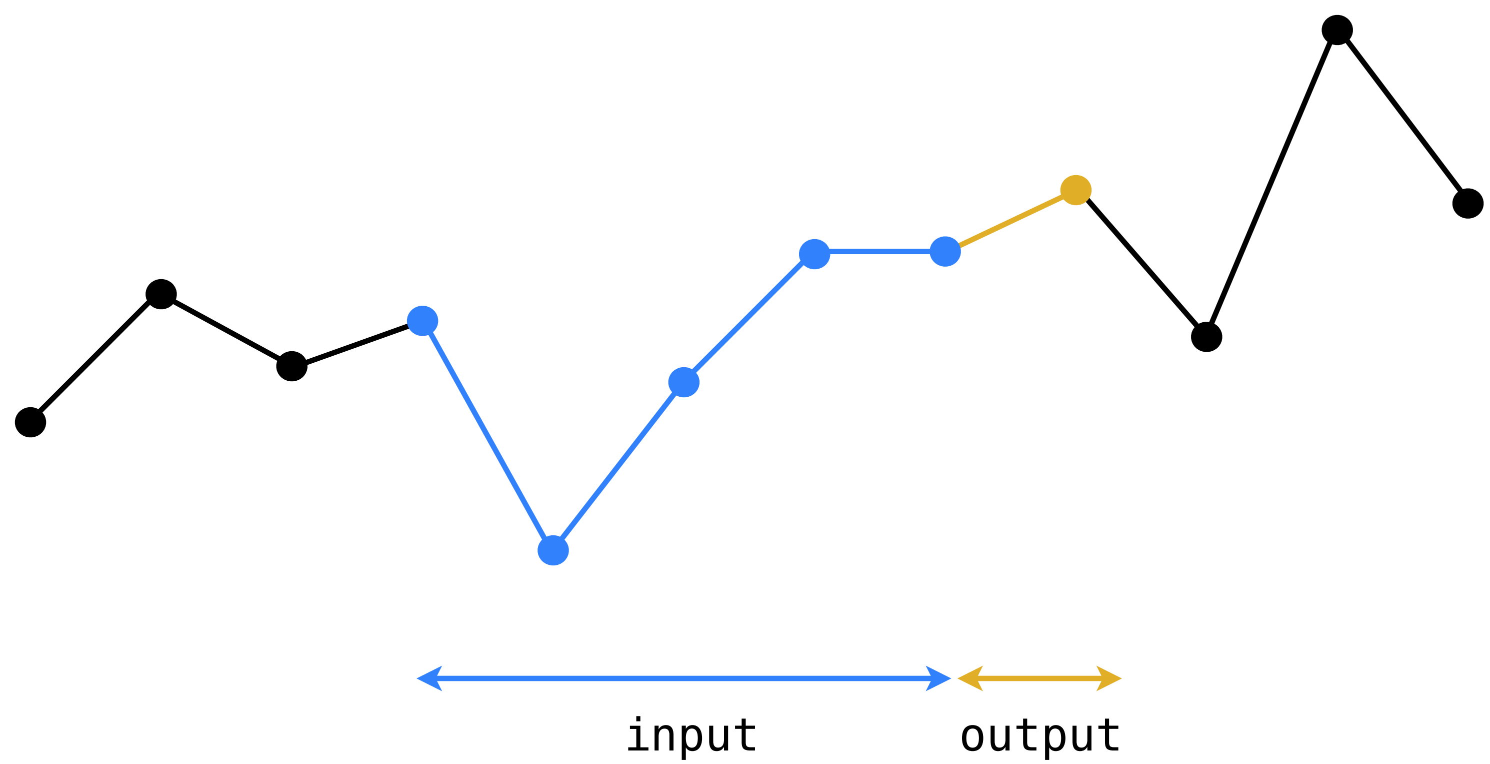
input
output



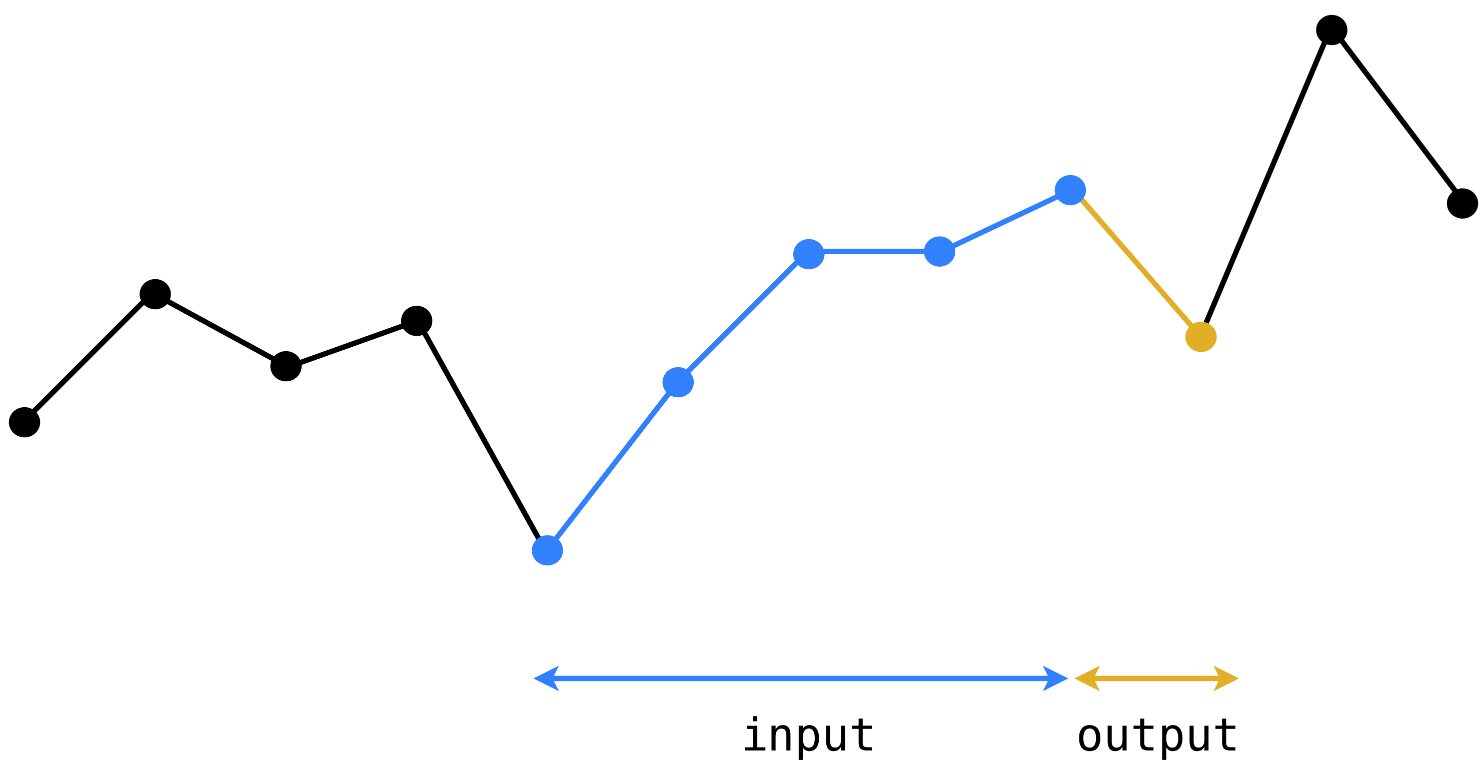
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4



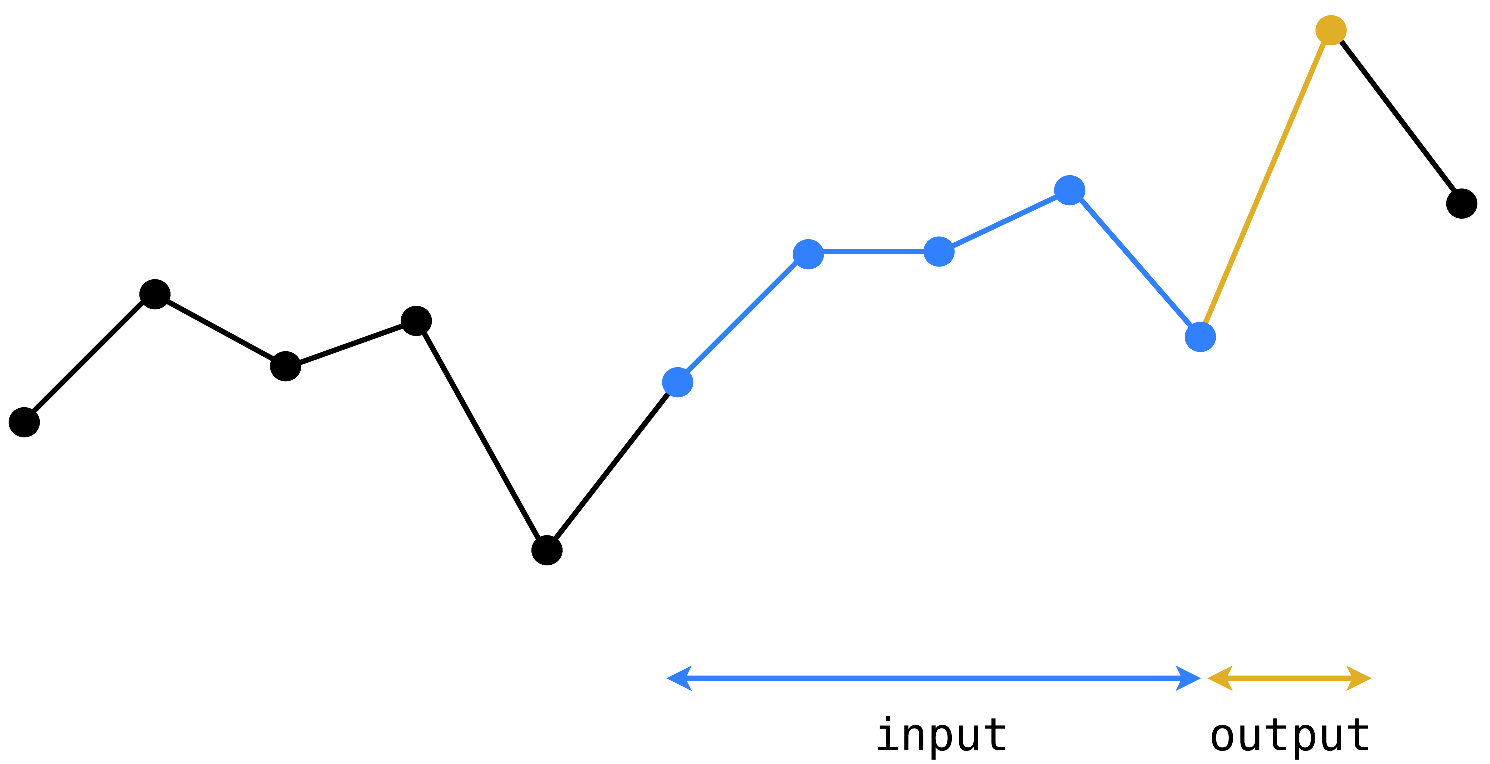
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



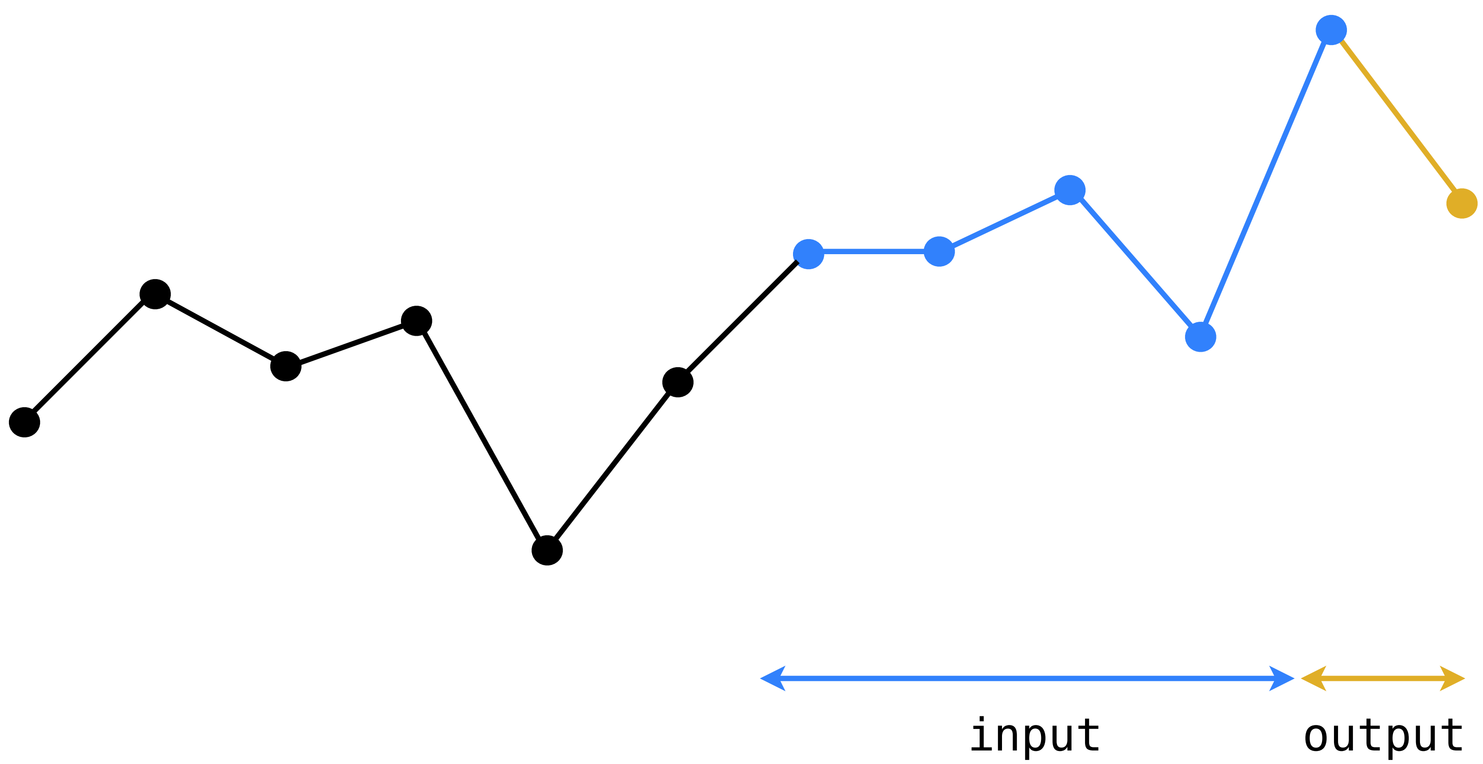
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



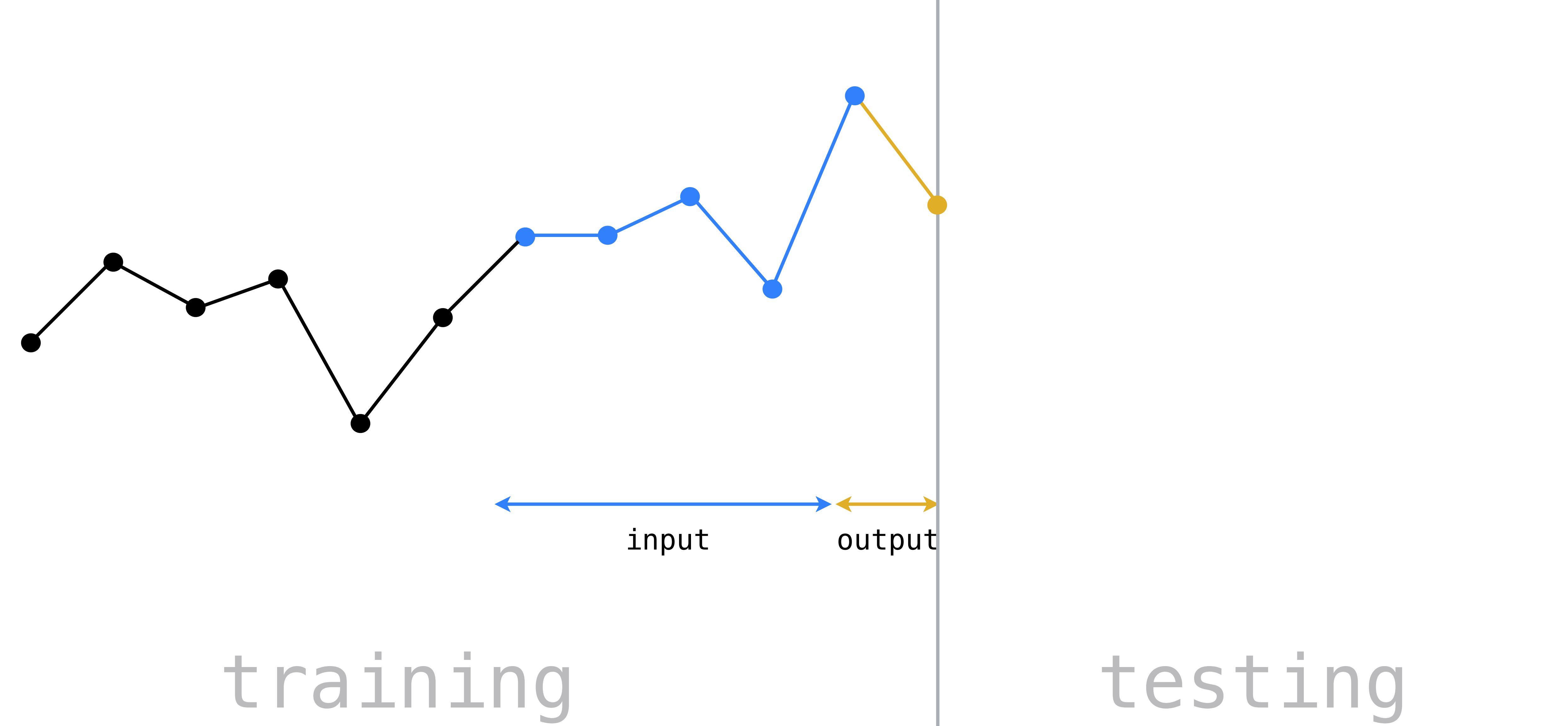
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



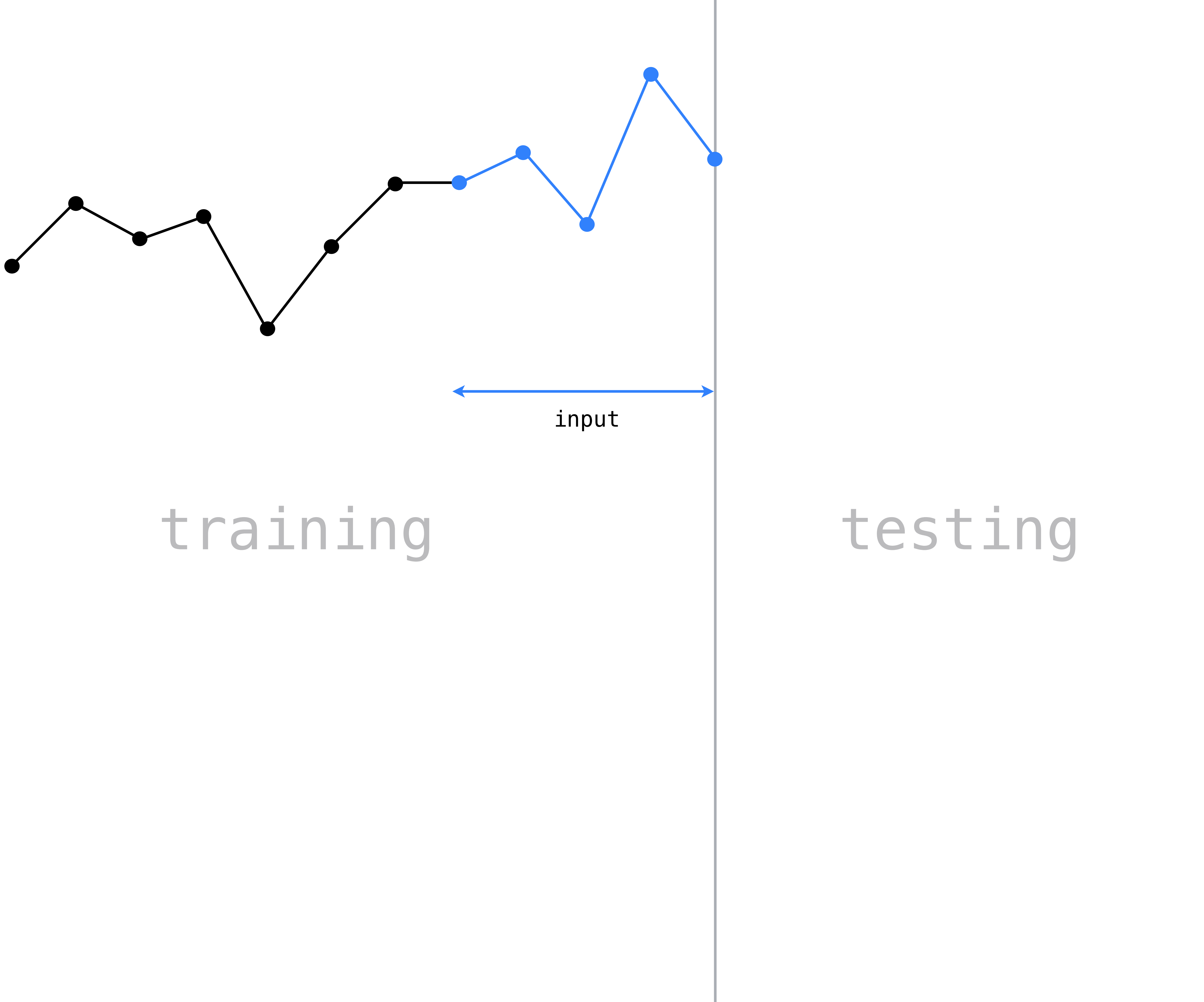
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$

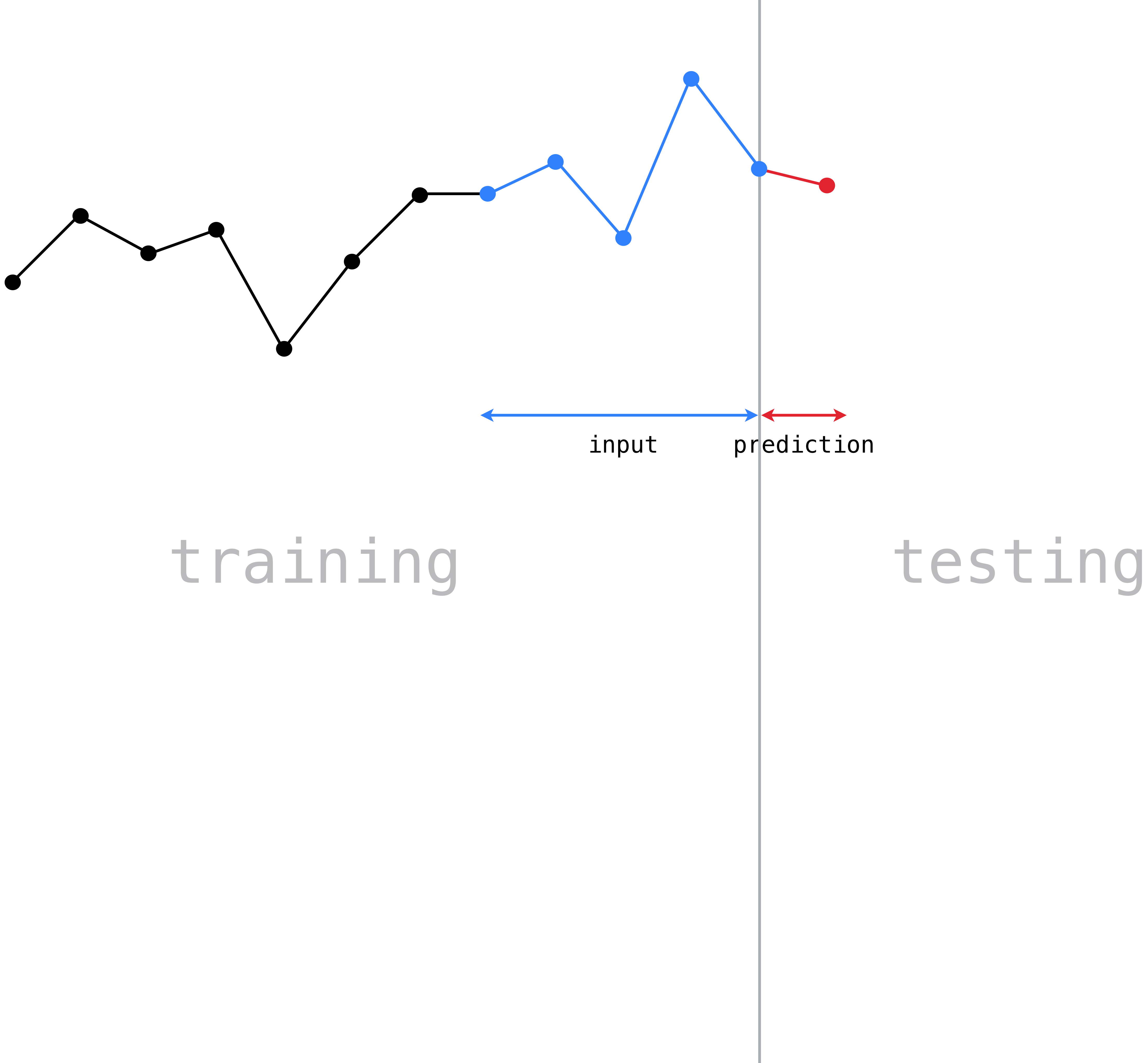


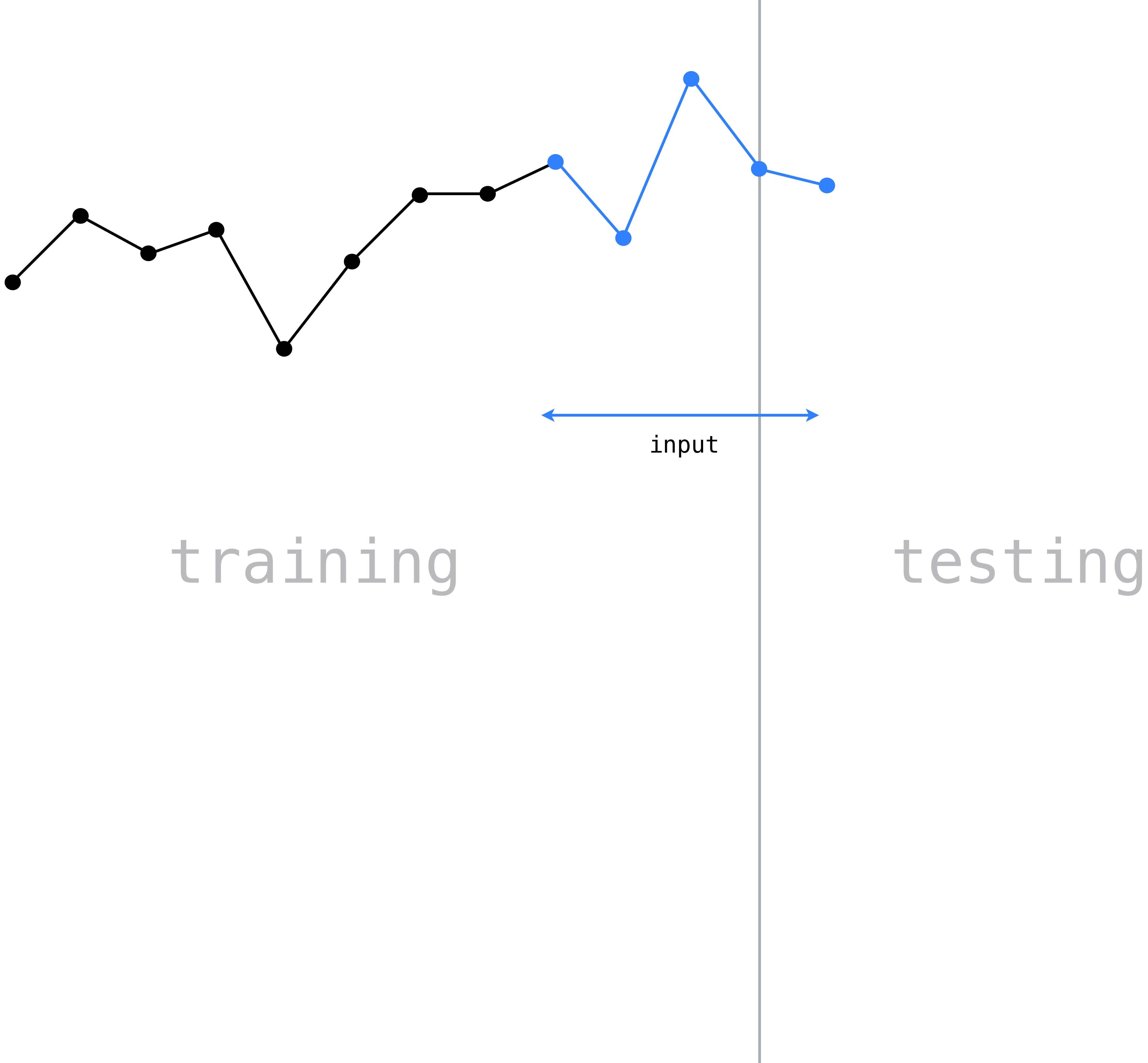
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$
$\langle s_{P-4}, s_{P-3}, s_{P-2}, s_{P-1} \rangle$	s_P

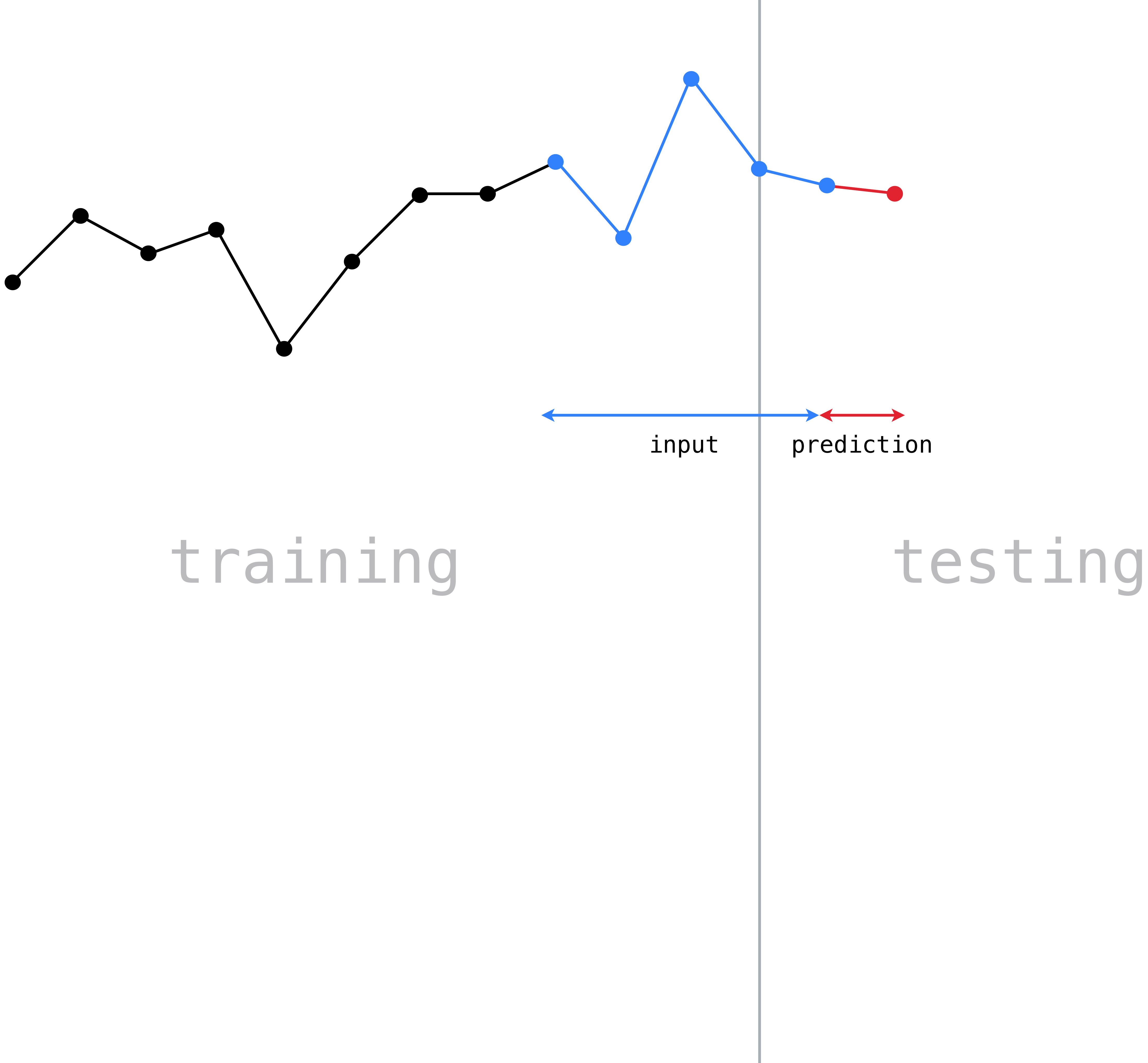


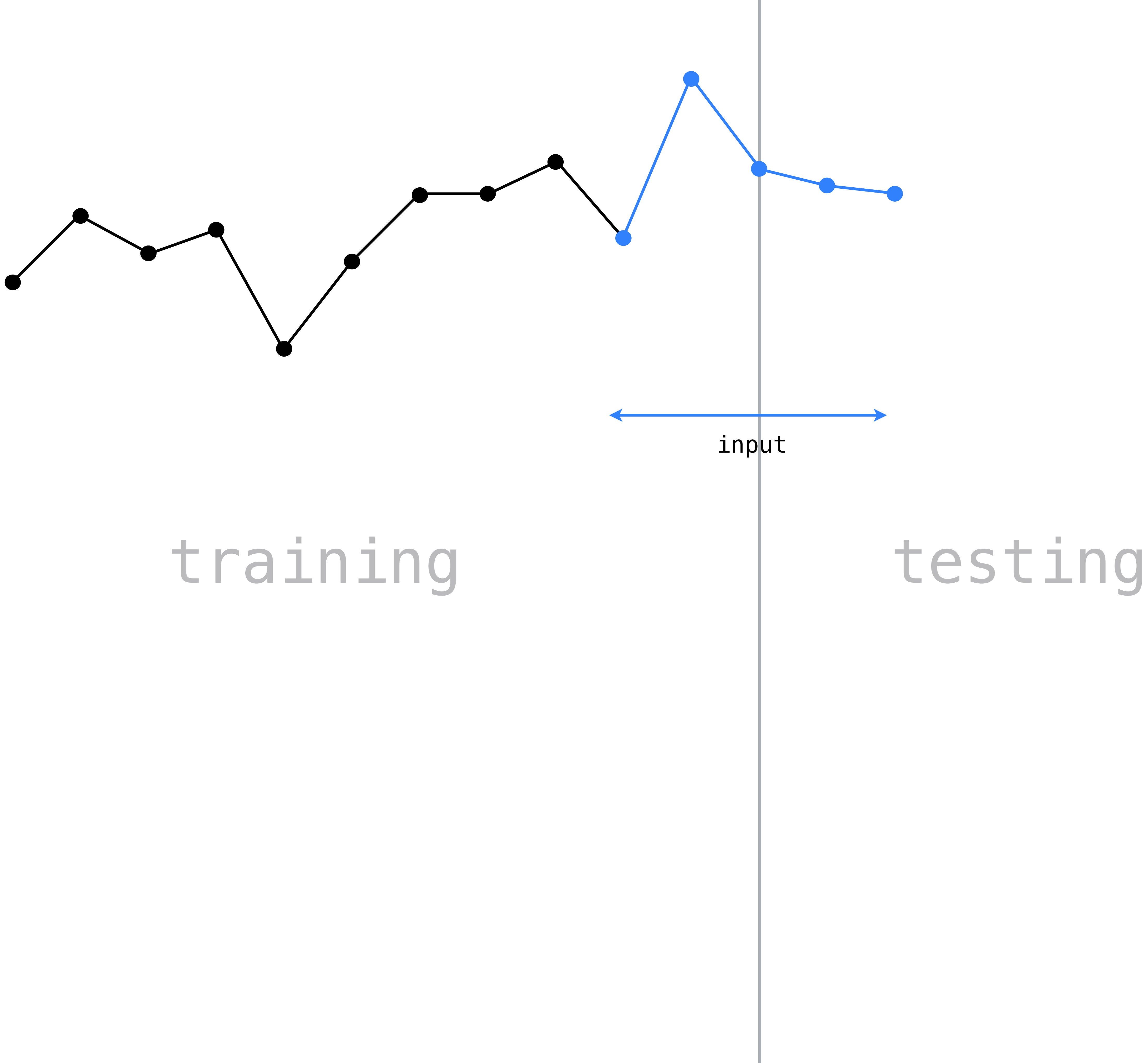
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$
$\langle s_{P-4}, s_{P-3}, s_{P-2}, s_{P-1} \rangle$	s_P

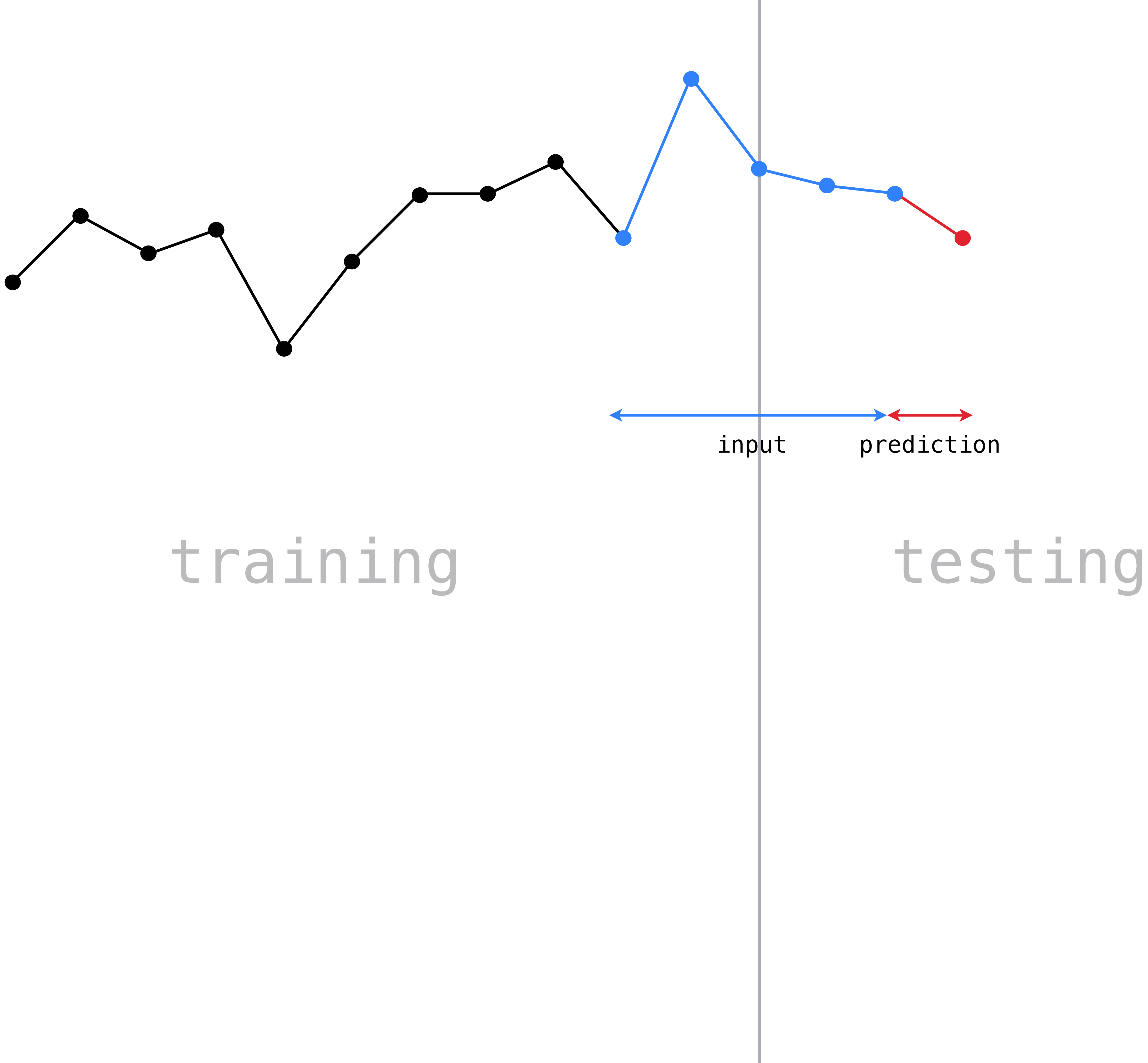

input
training
testing

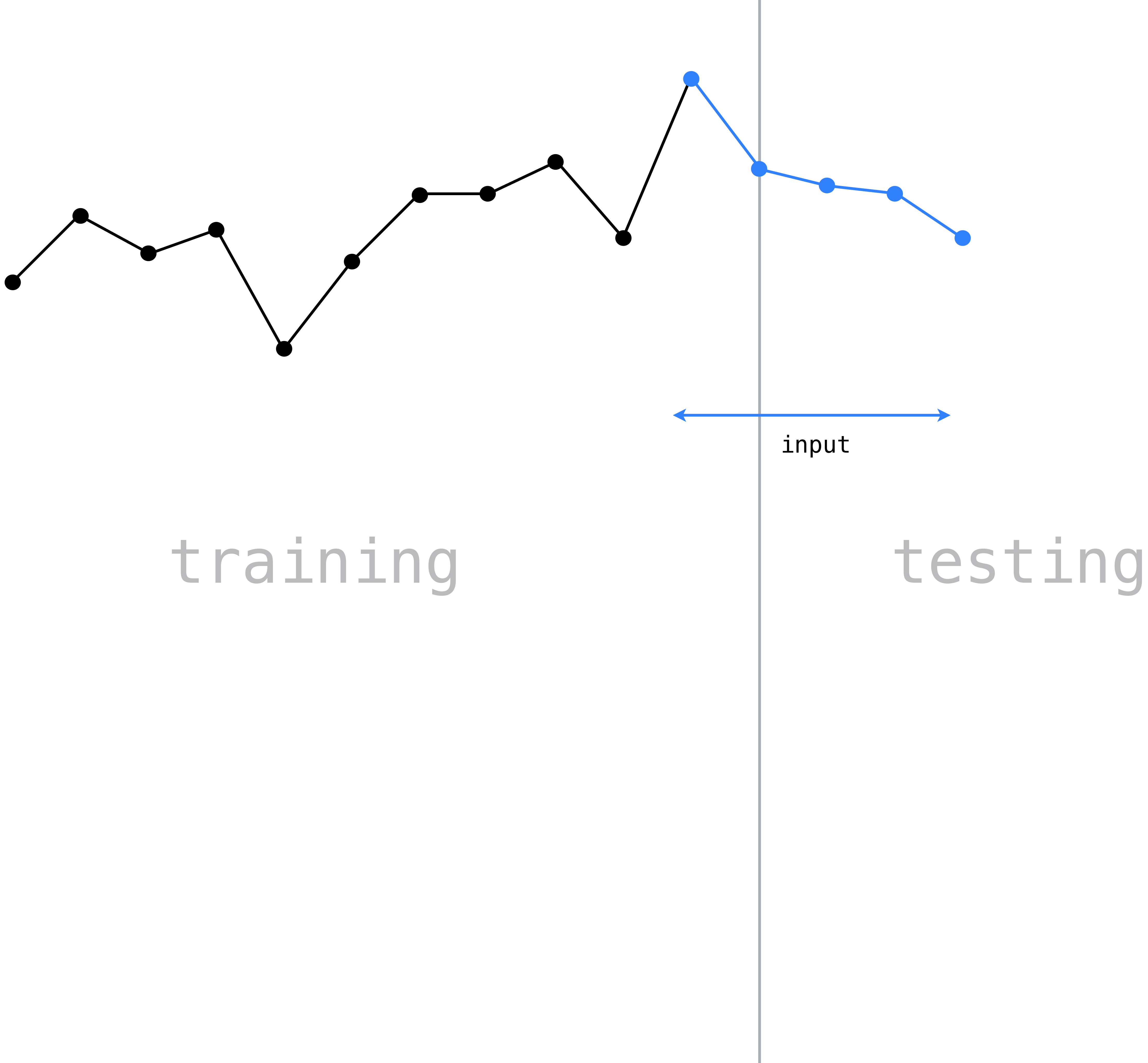

input
prediction
training
testing

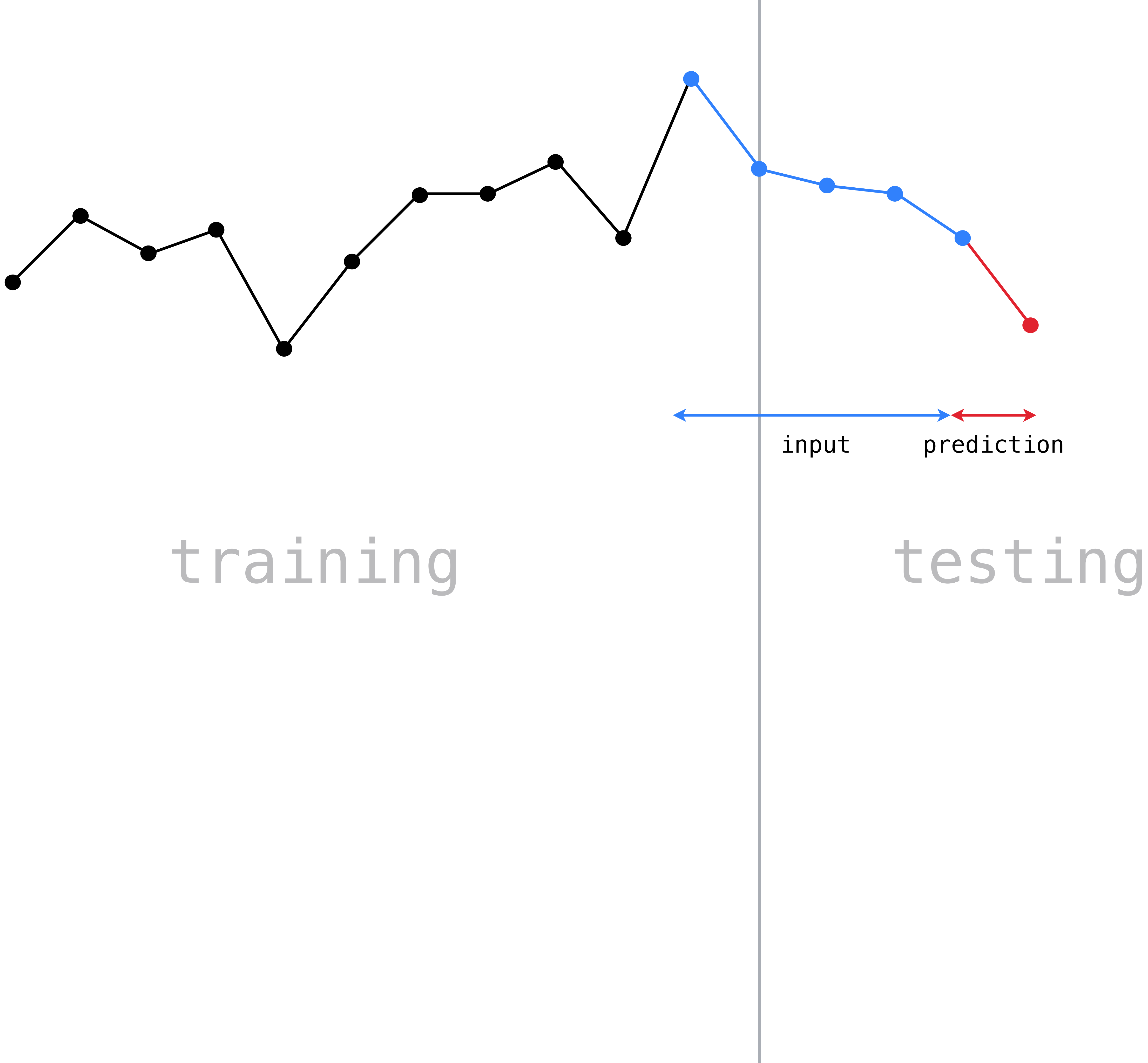

input
training
testing

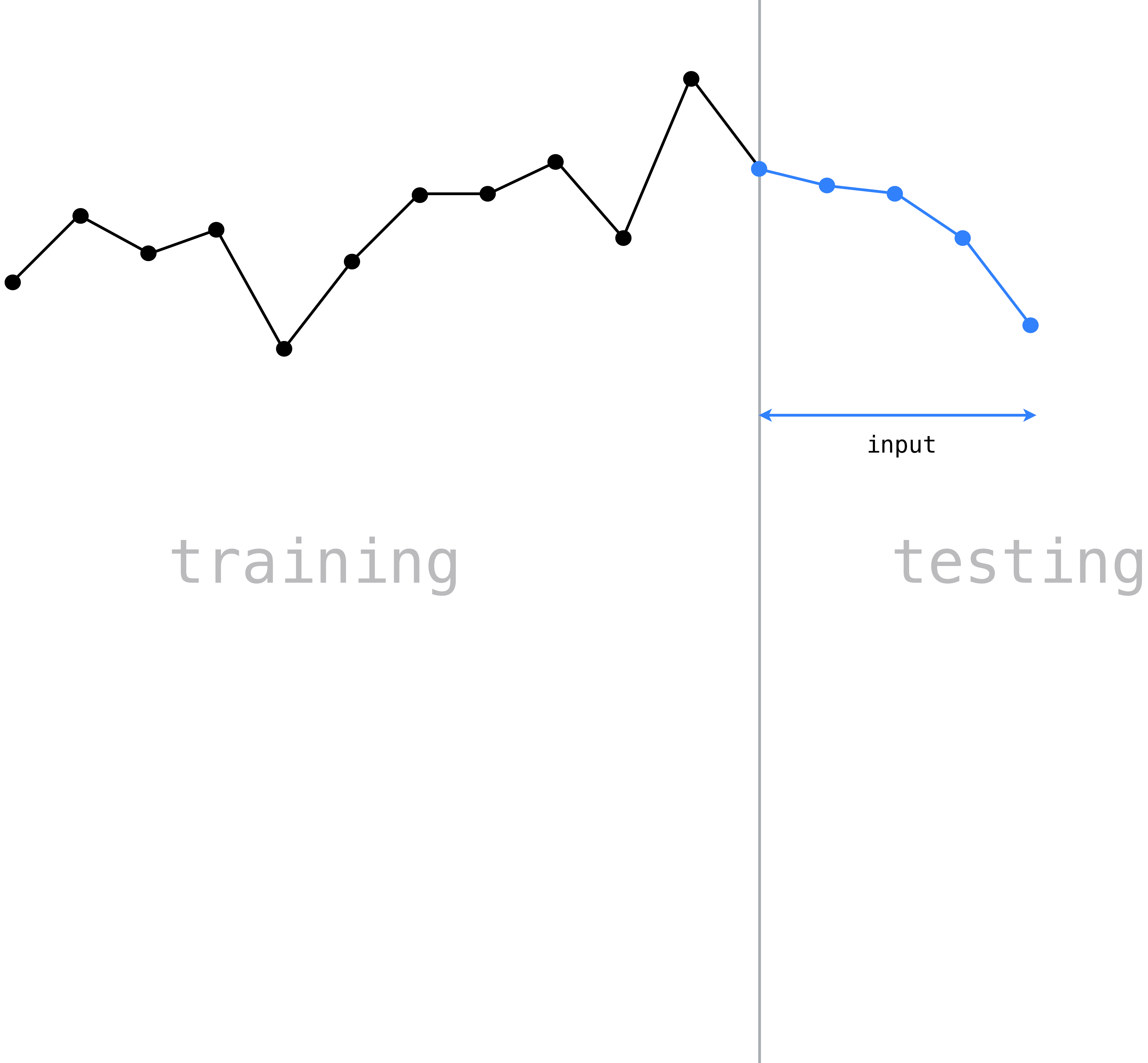

input
prediction
training
testing

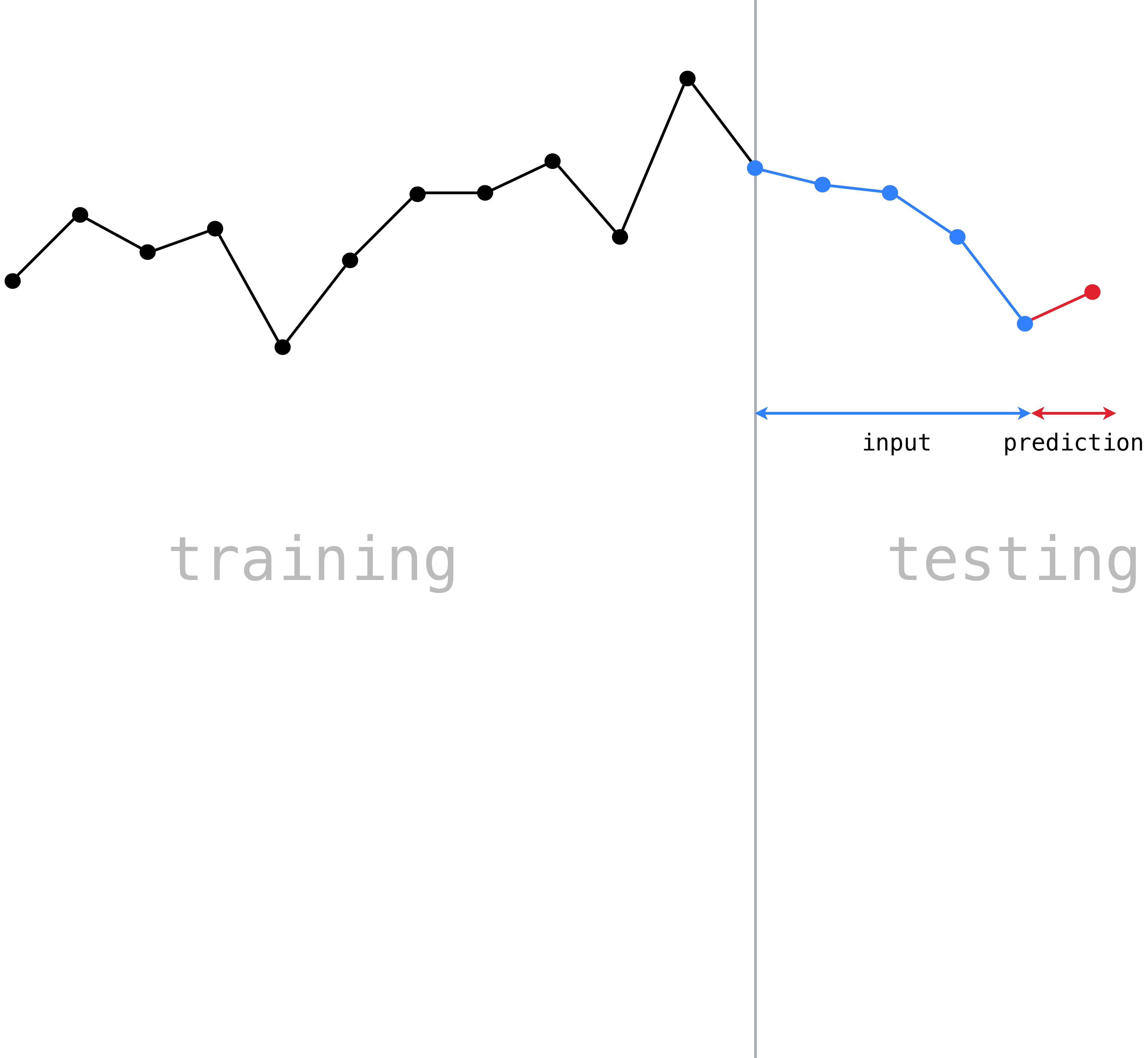

input
training
testing

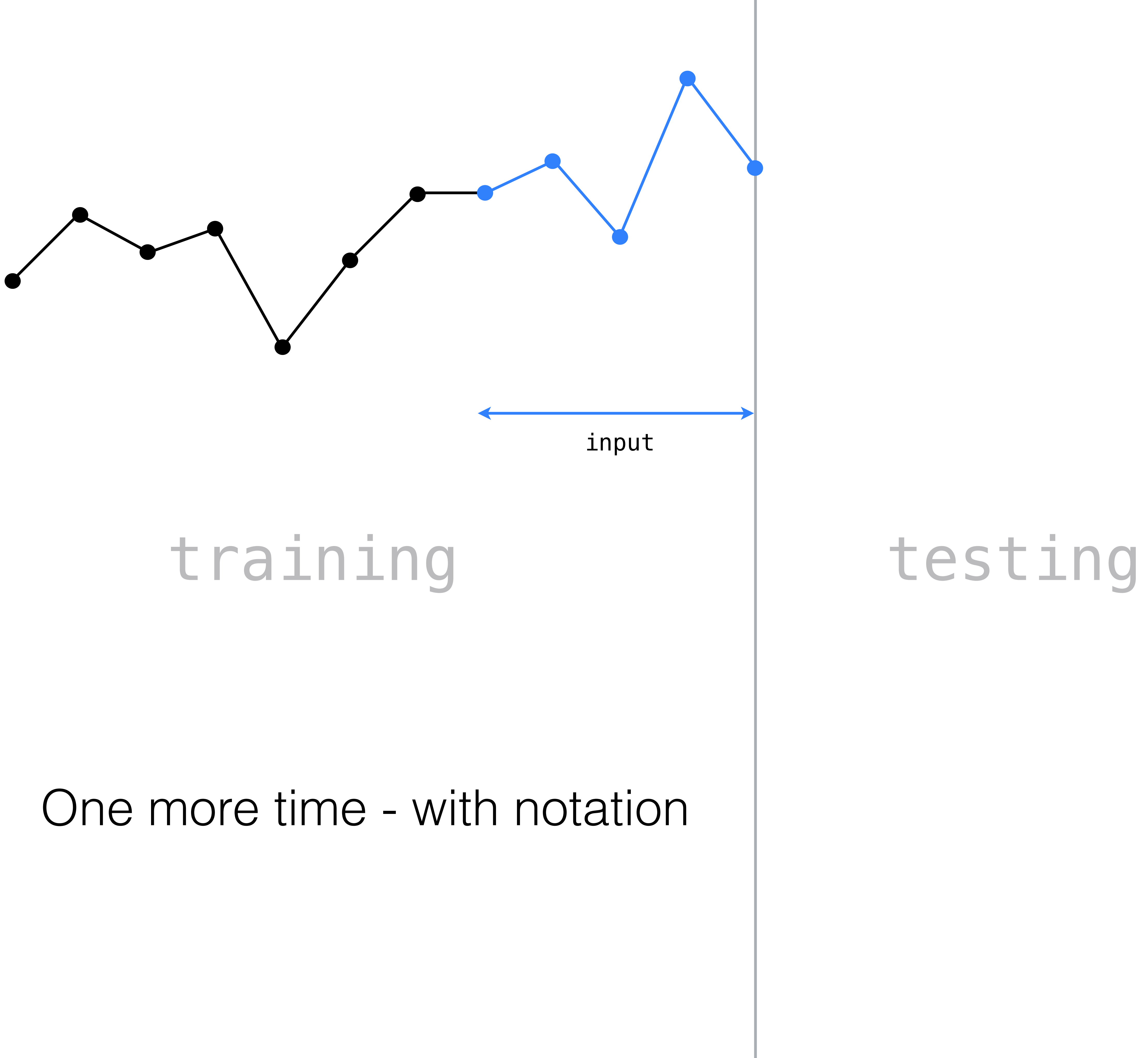

input
prediction
training
testing


input
training
testing

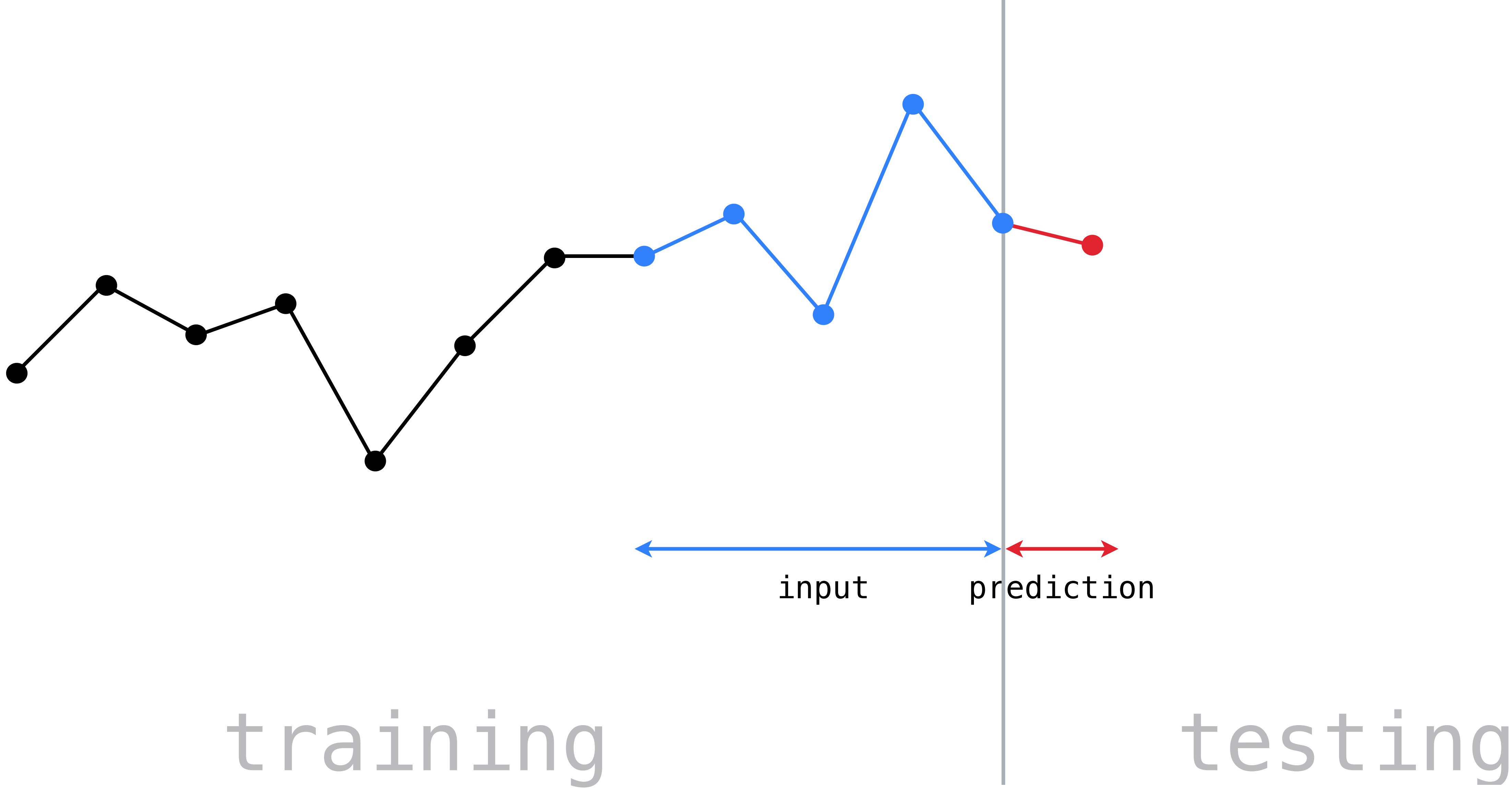

input
prediction
training
testing


input
training
testing

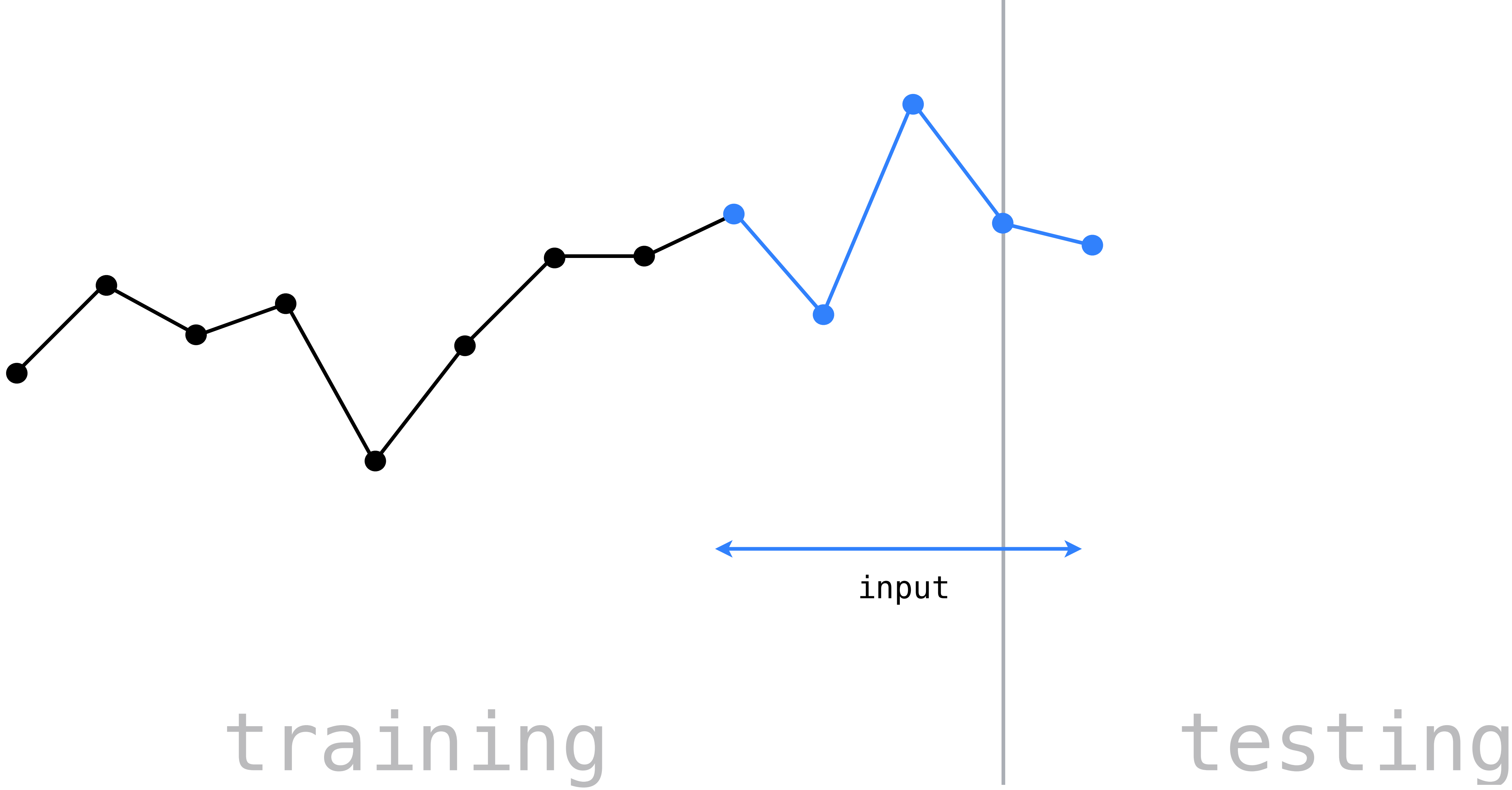

input
prediction
training
testing



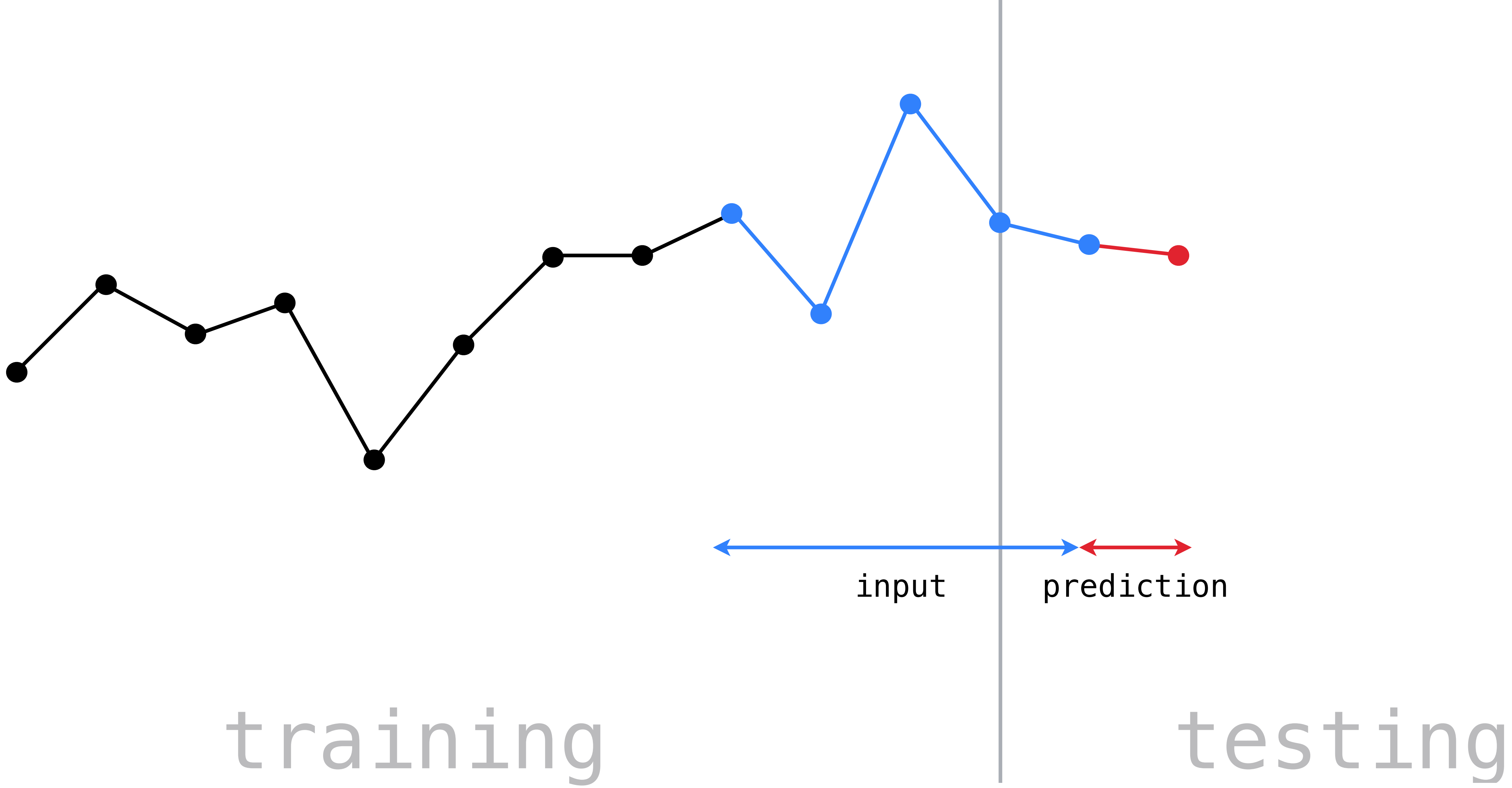
One more time - with notation



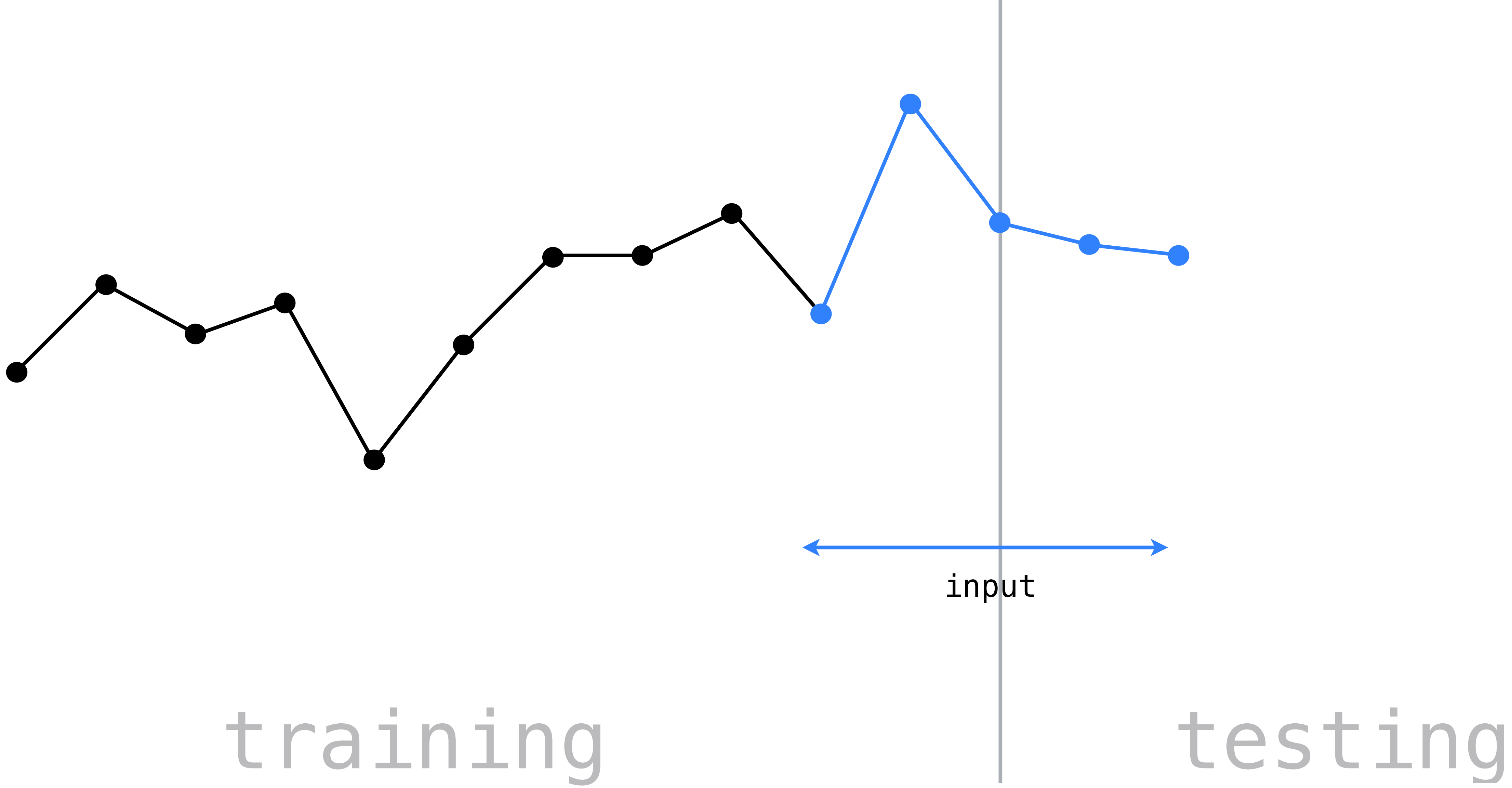
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$



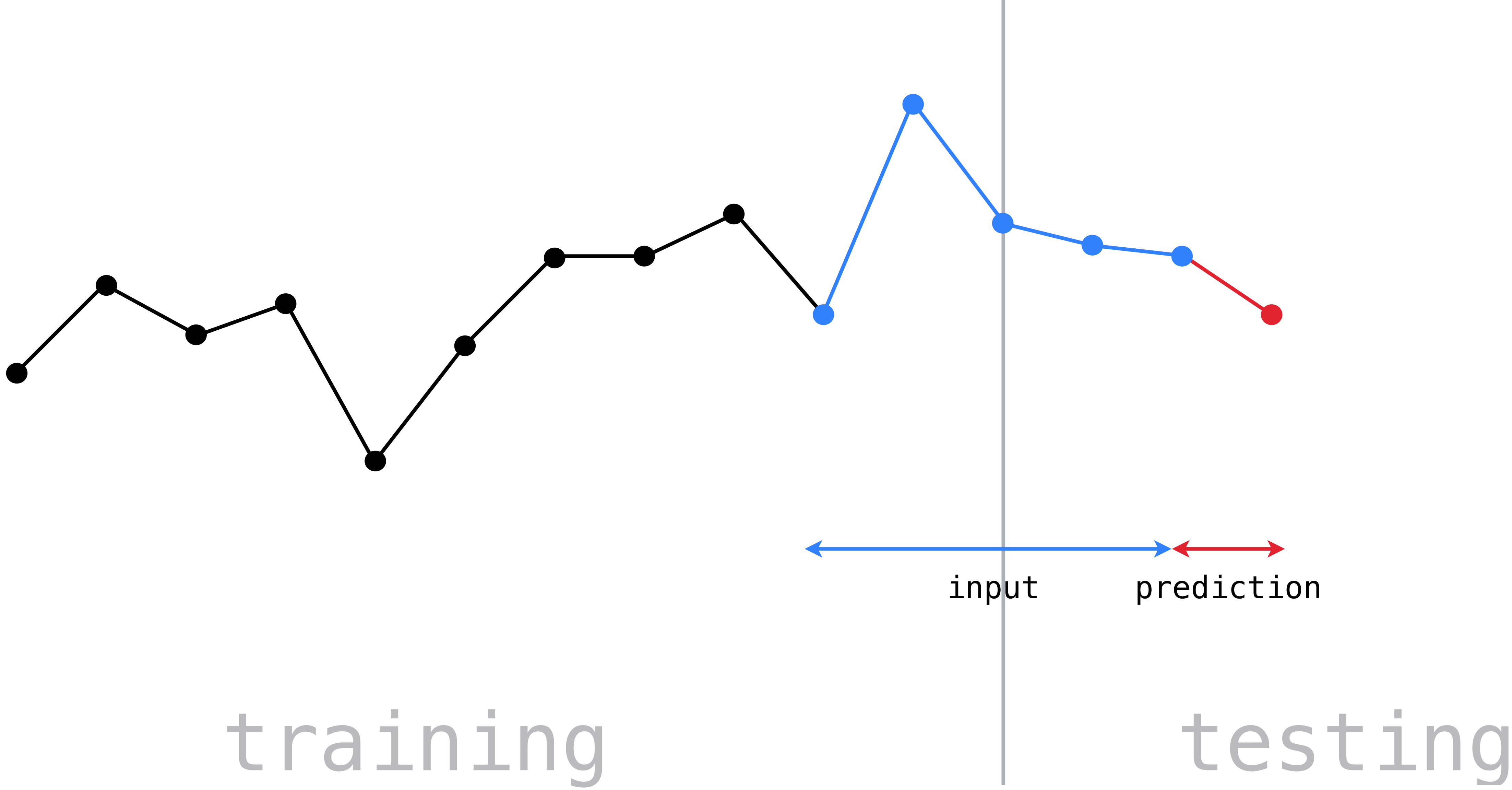
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$



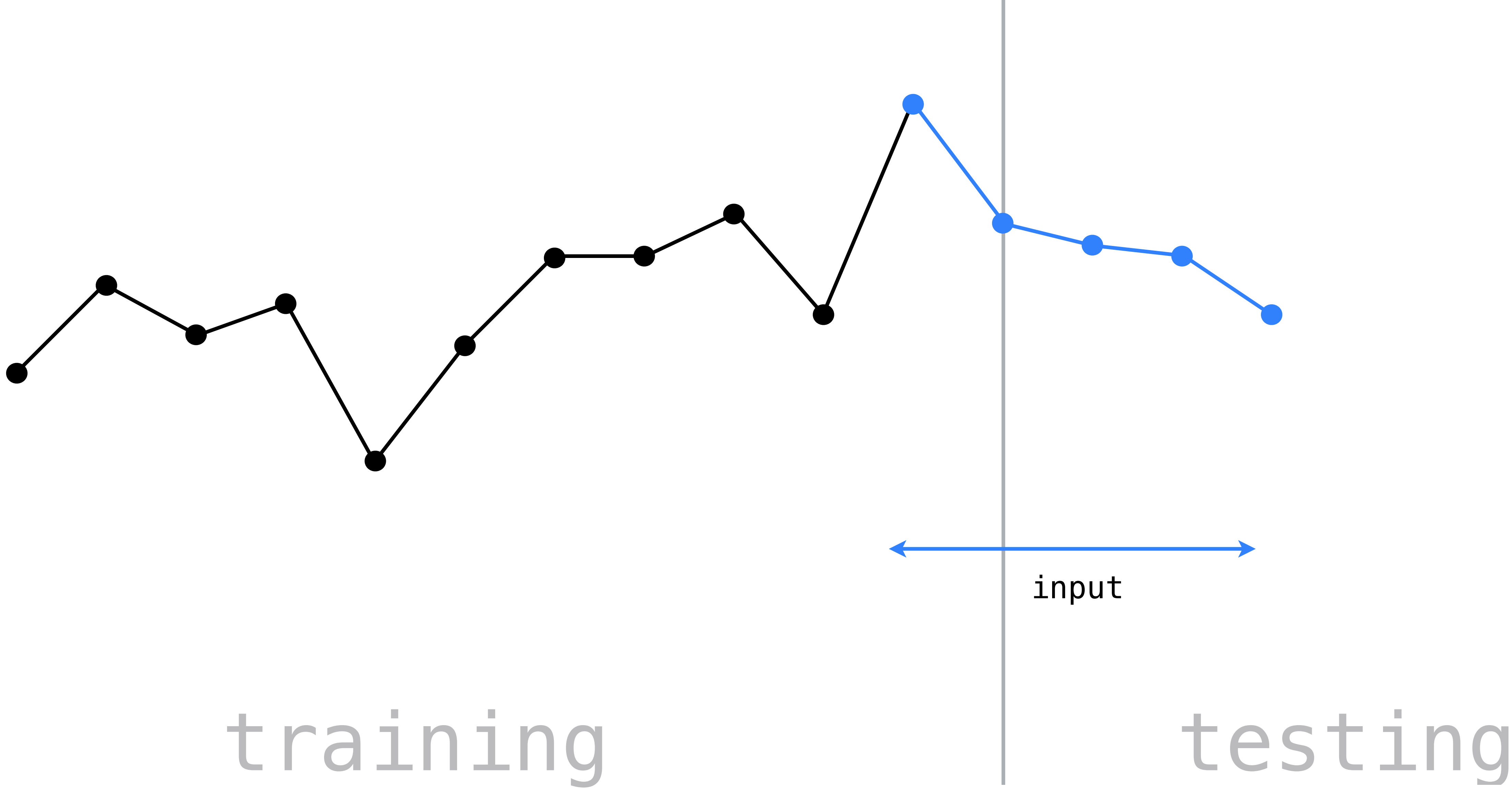
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$



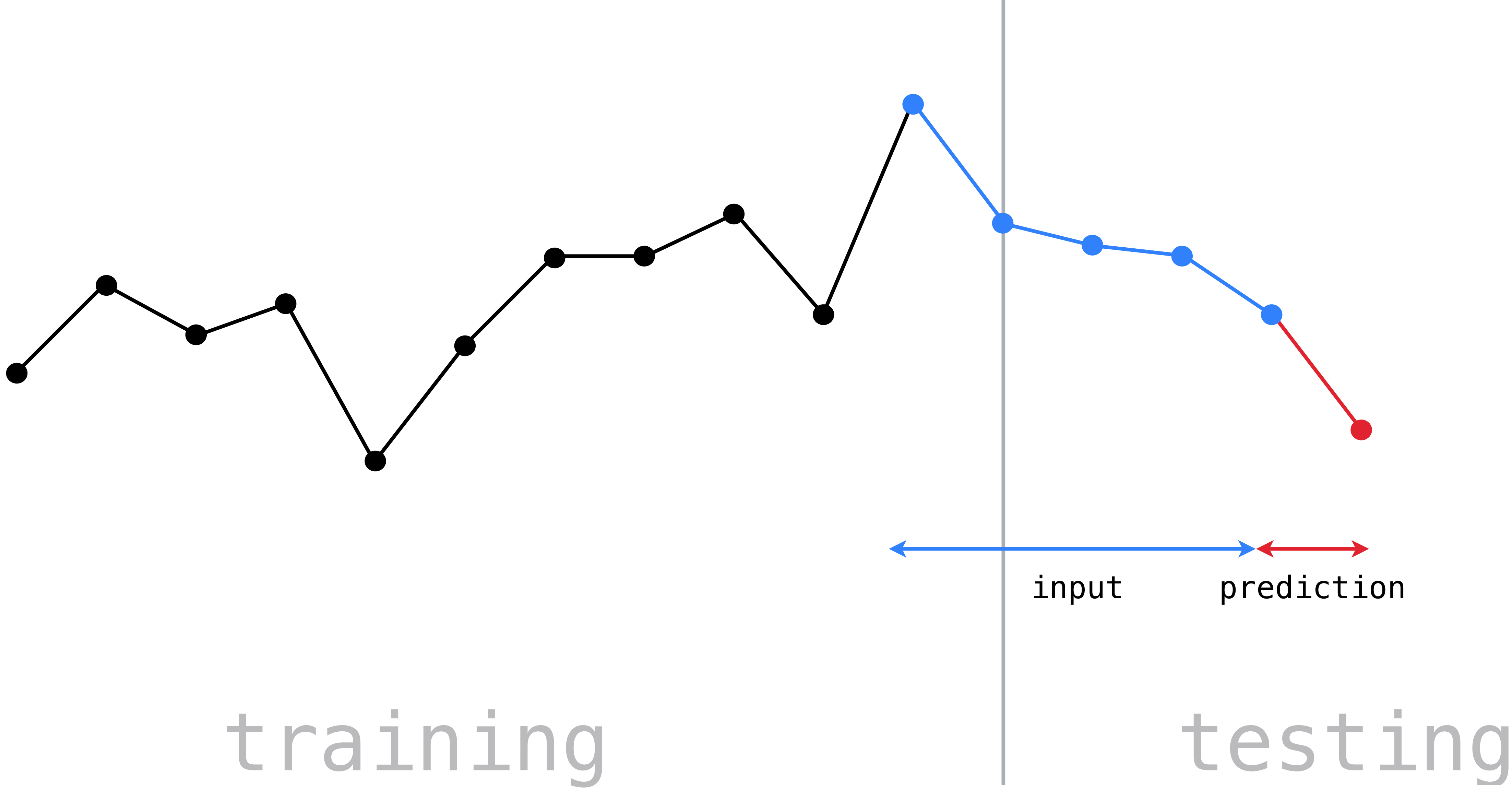
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$



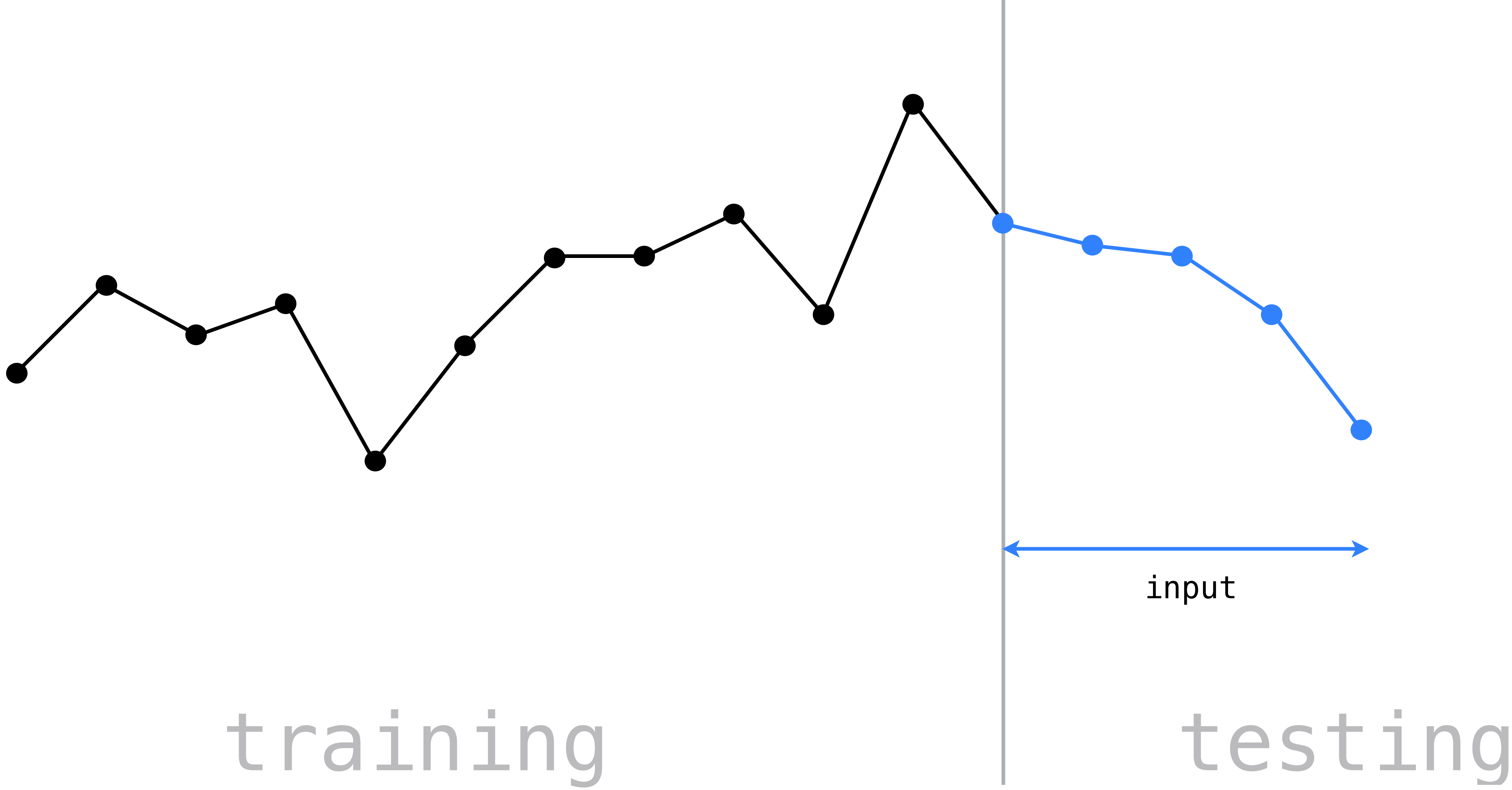
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$



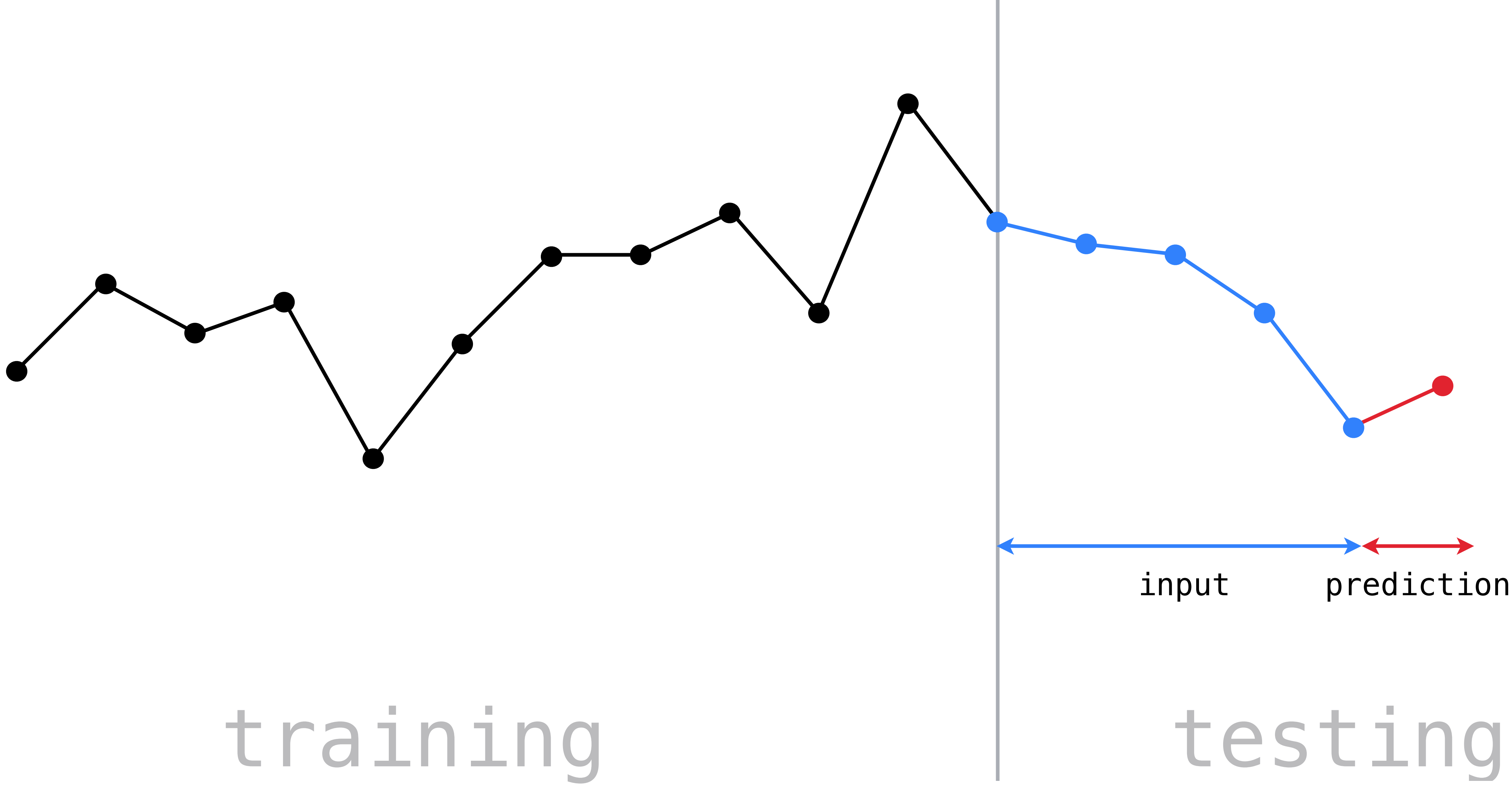
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$



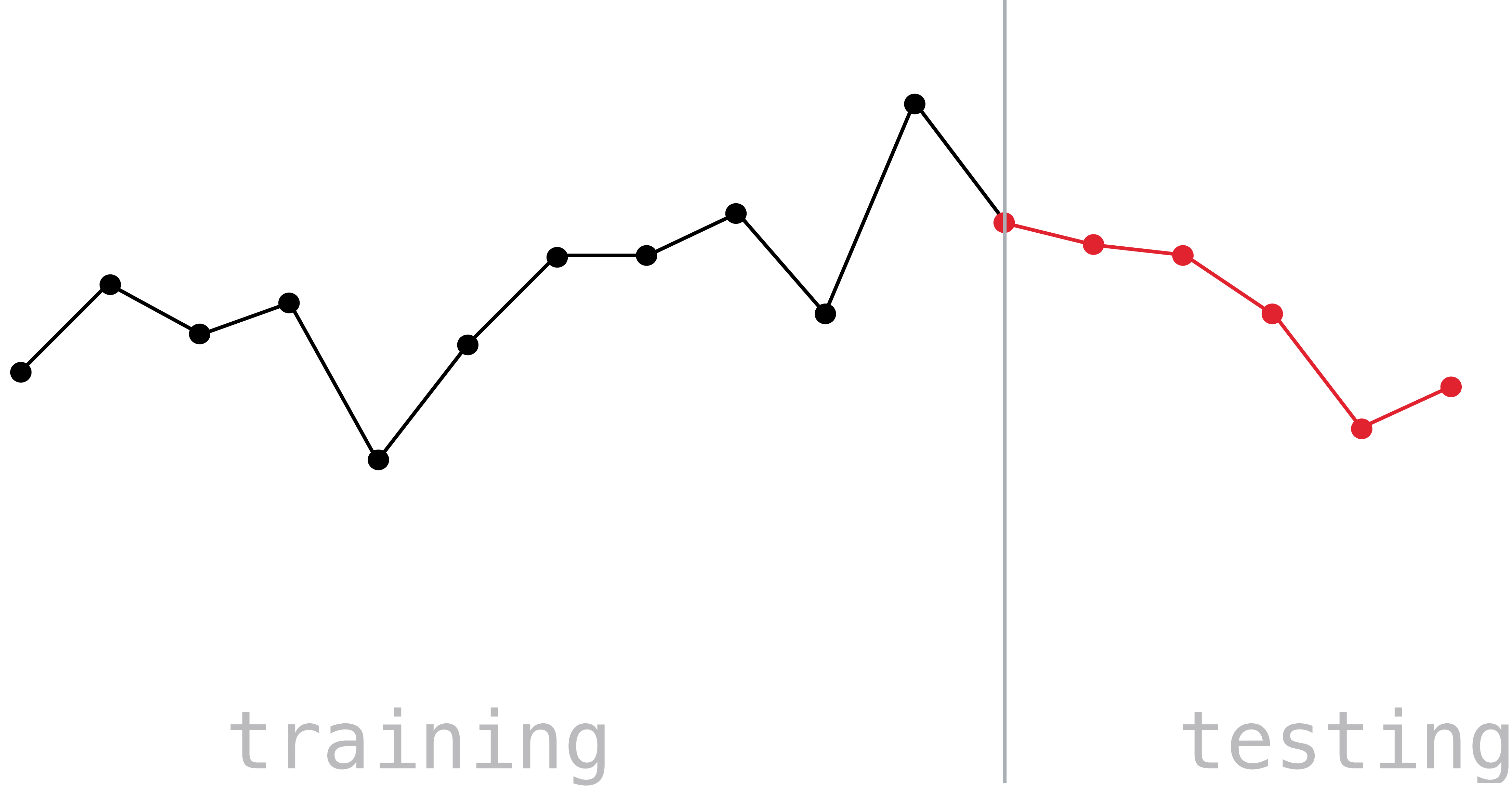
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$



Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$



Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$



Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$
$\vdots$	$\vdots$

here we illustratred with
window size T = 4

Ingesting sequential I/O data

Text generation

- Sequence of P characters: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p: the pth character

Text generation

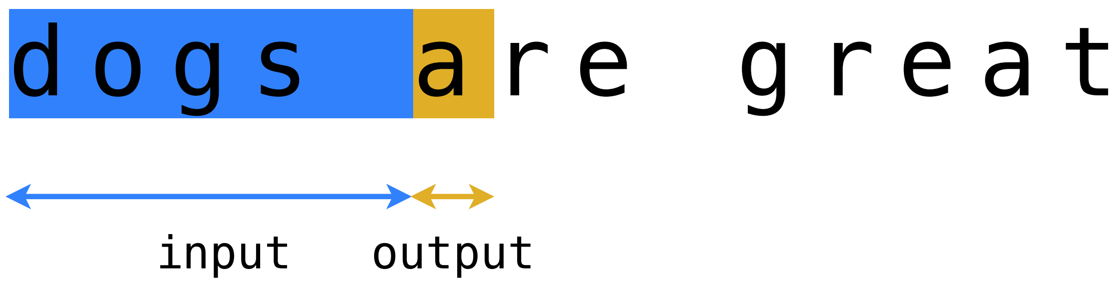
- Sequence of P characters: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p: the pth character
- - **classification problem**- I/O are windowed subsequences

Text generation

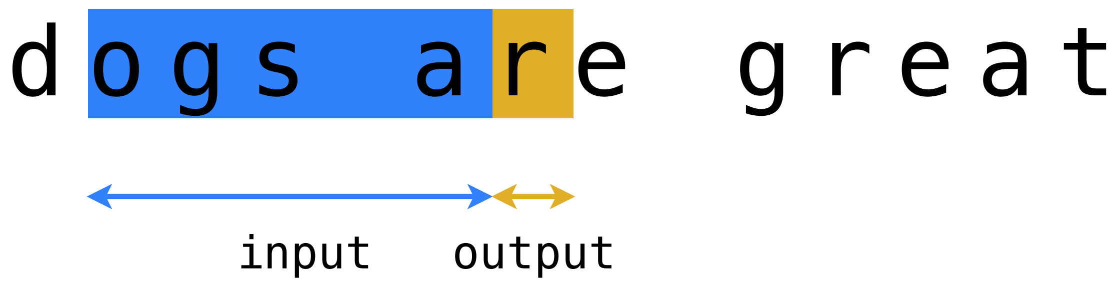
- Sequence of P characters: $\langle s_0, s_1, s_2, \ldots, s_P \rangle$
- s_p: the pth character
- **classification problem**- I/O are windowed subsequences

  - so need training and testing sets

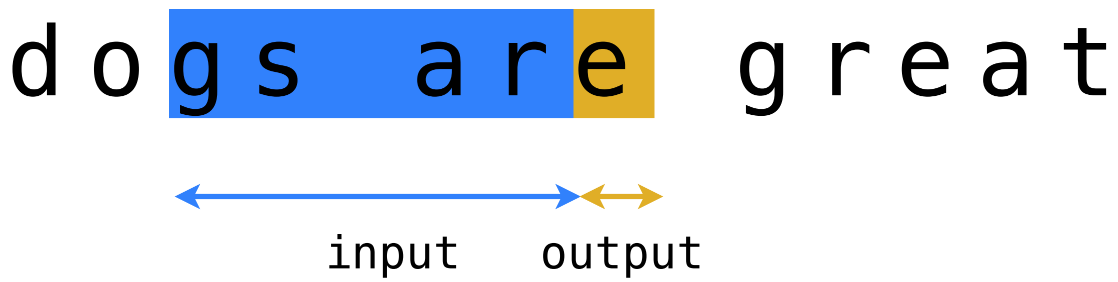
  - lets see how both are formed, start with training dogs are great

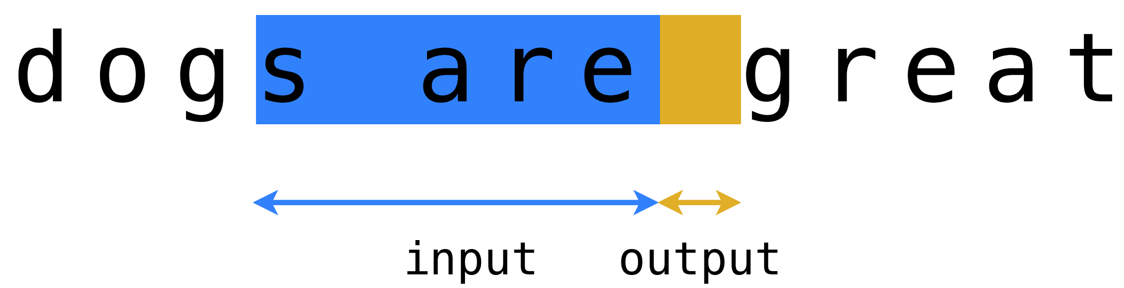


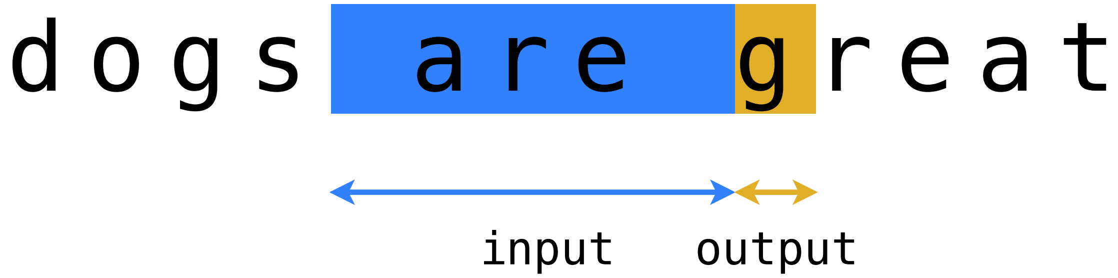
dogs are great
input
output



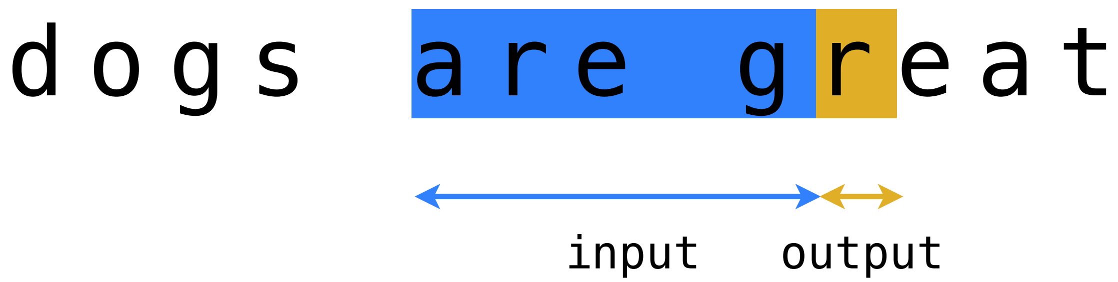
d o g s   a r e   g r e a t
input
output

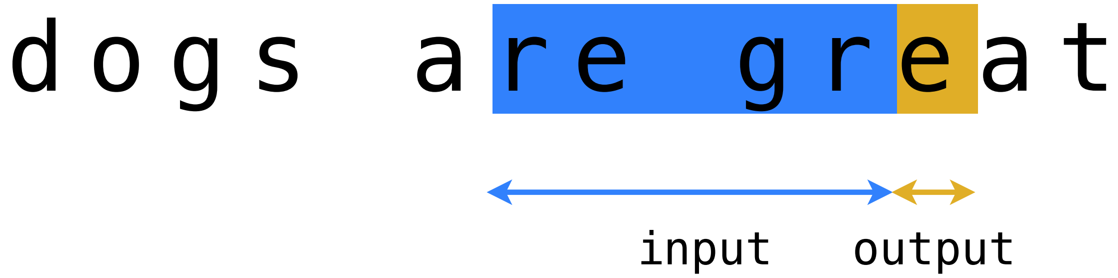

d o g s a r e g r e a t
input
output

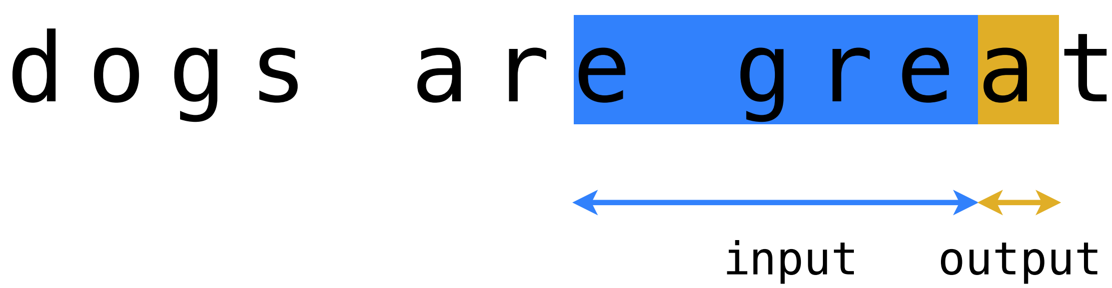

dogs are great
input
output

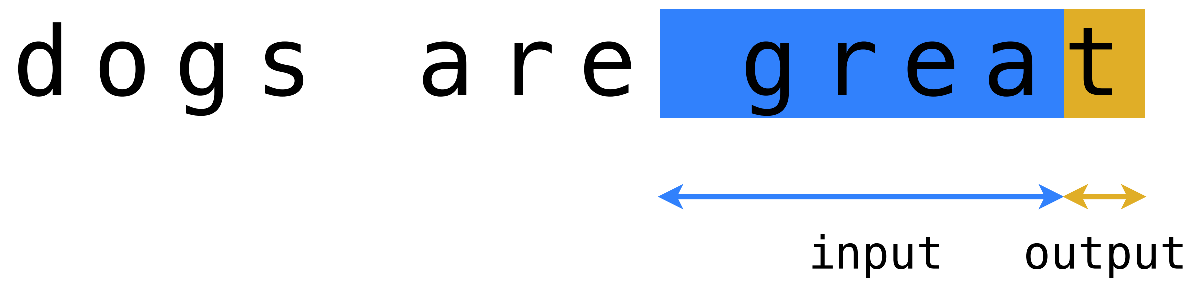


d o g s   a r e   g r e a t
input
output

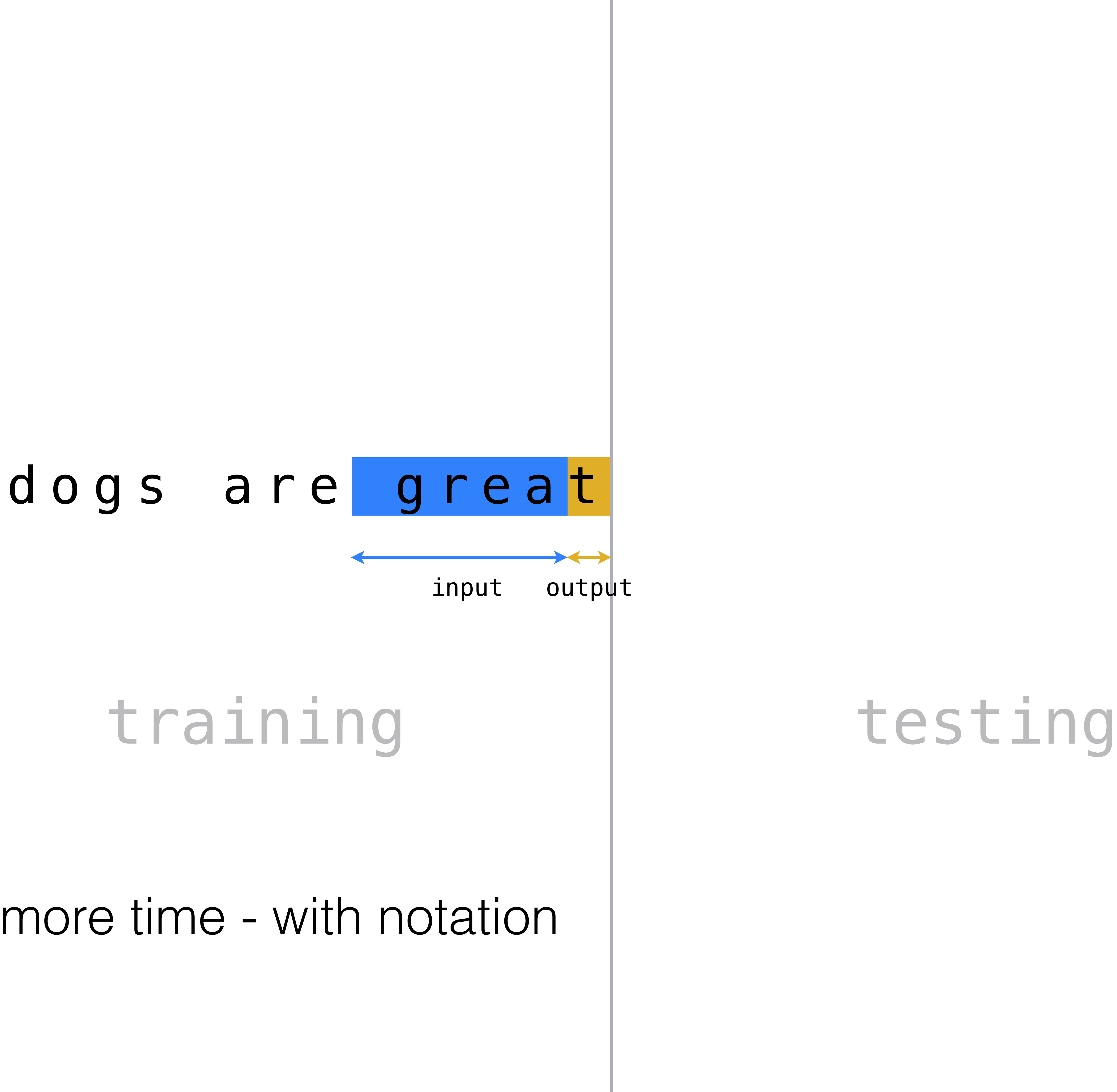

d o g s a r e g r e a t
input
output


dogs are great
input
output

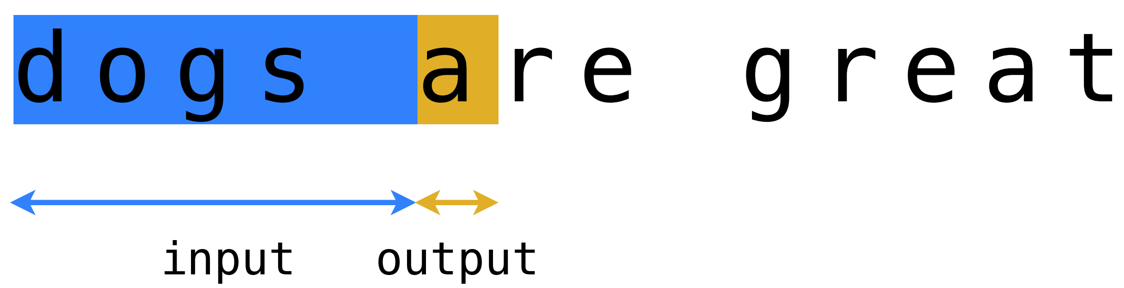

dogs are great
input
output



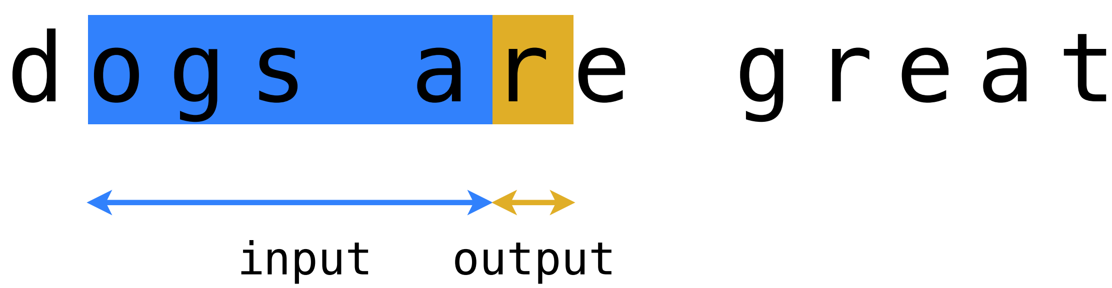
d o g s a r e g r e a t
input
output



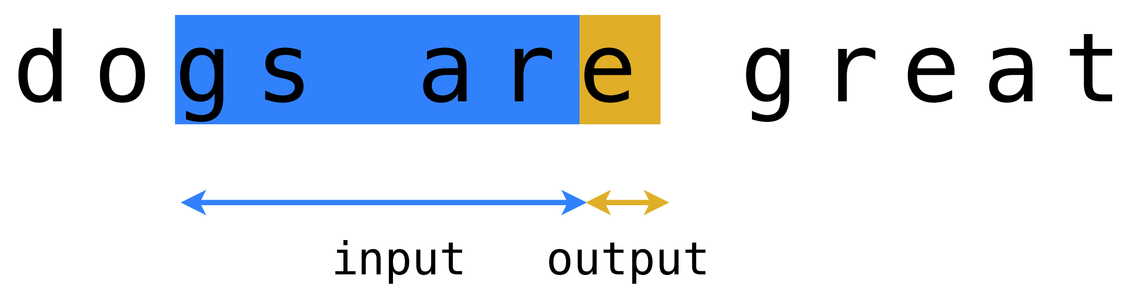
One more time - with notation



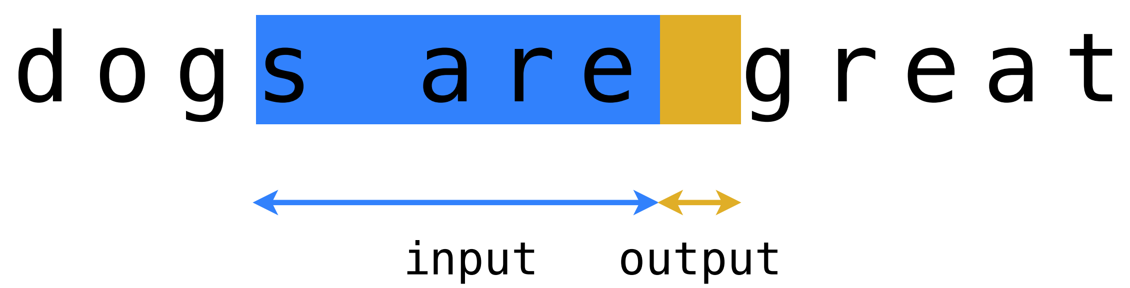
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4



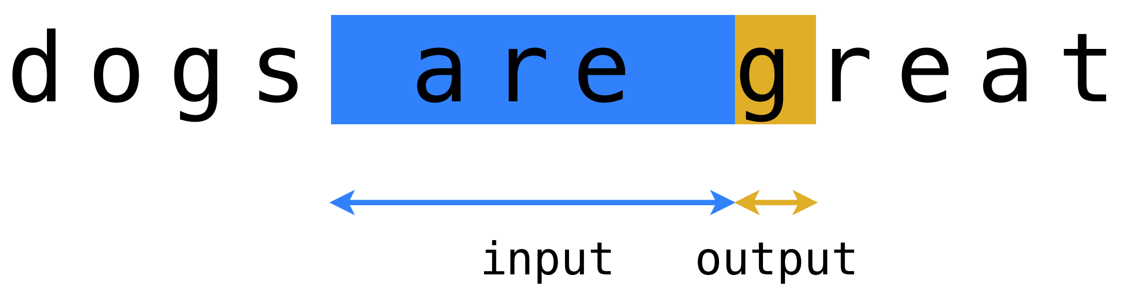
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5



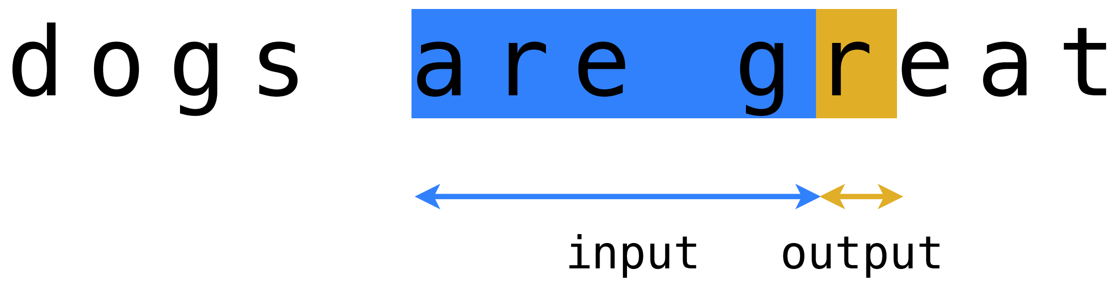
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



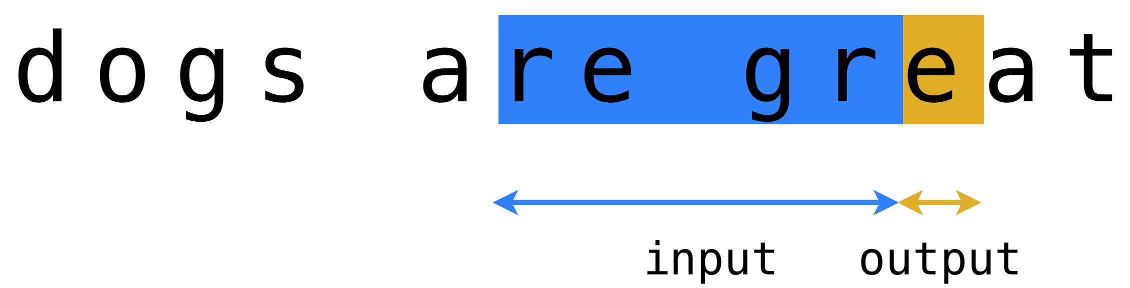
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



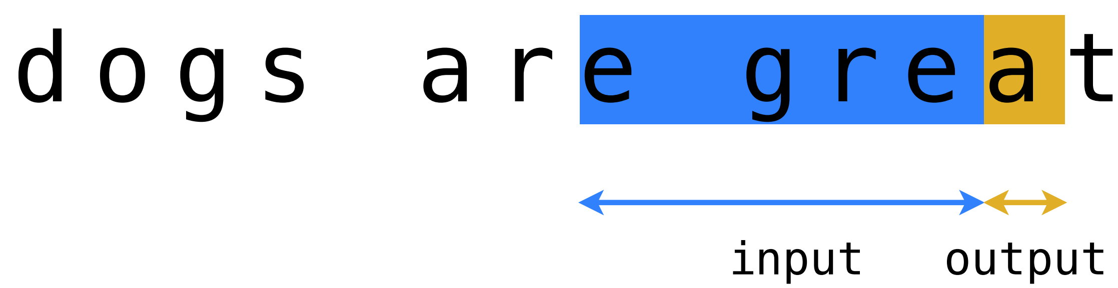
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



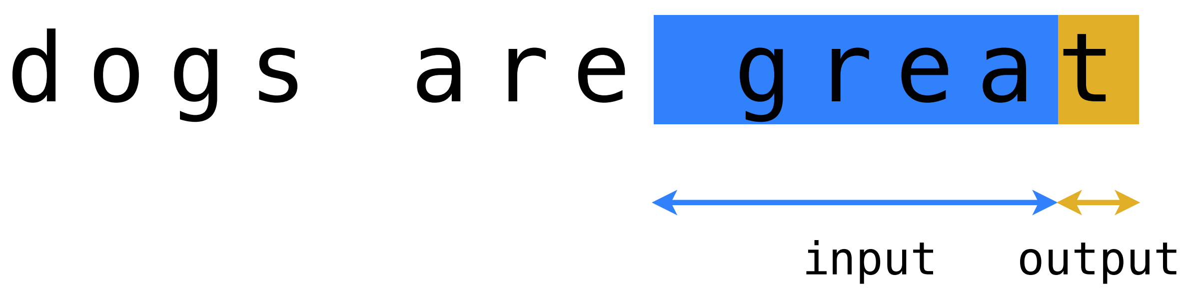
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



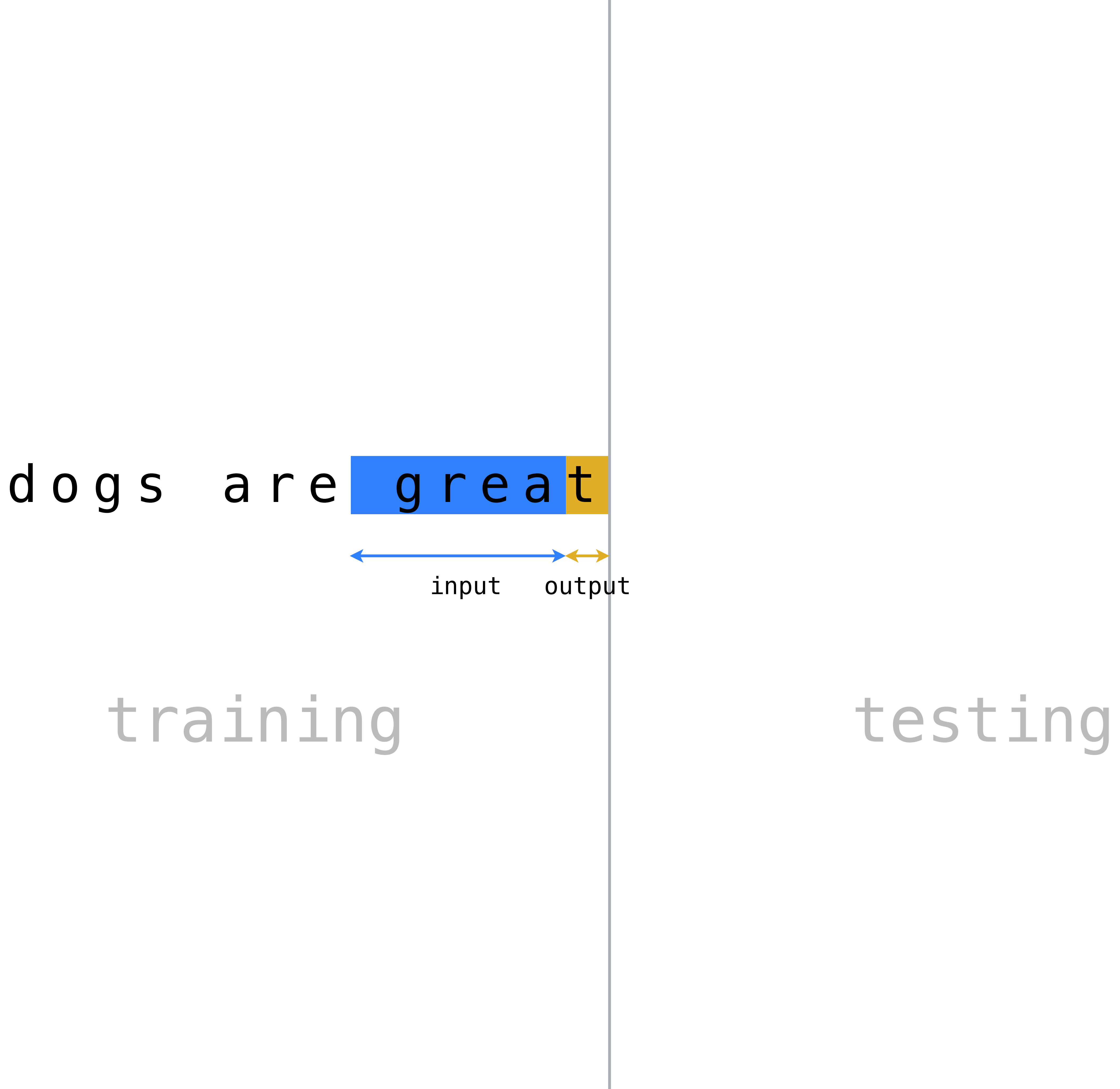
Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$



Input	Output
$\langle s_0, s_1, s_2, s_3 \rangle$	s_4
$\langle s_1, s_2, s_3, s_4 \rangle$	s_5
$\vdots$	$\vdots$
$\langle s_{P-4}, s_{P-3}, s_{P-2}, s_{P-1} \rangle$	s_P



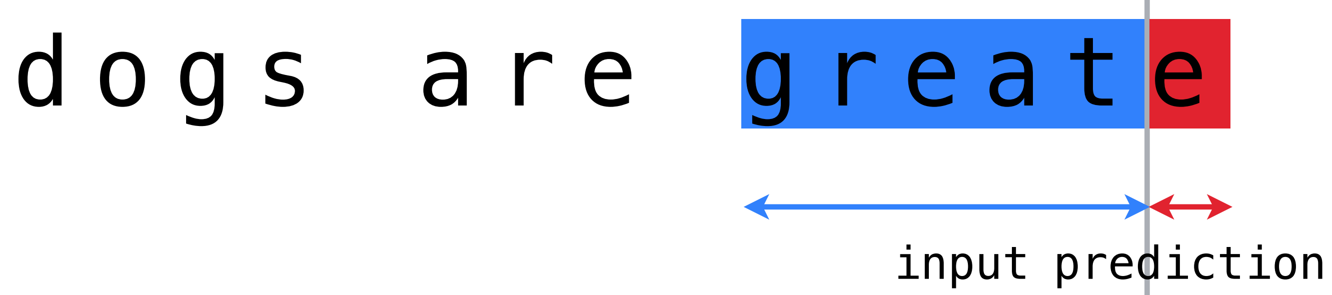
dogs are great
input
output
training
testing

d o g s   a r e   g r e a t

input

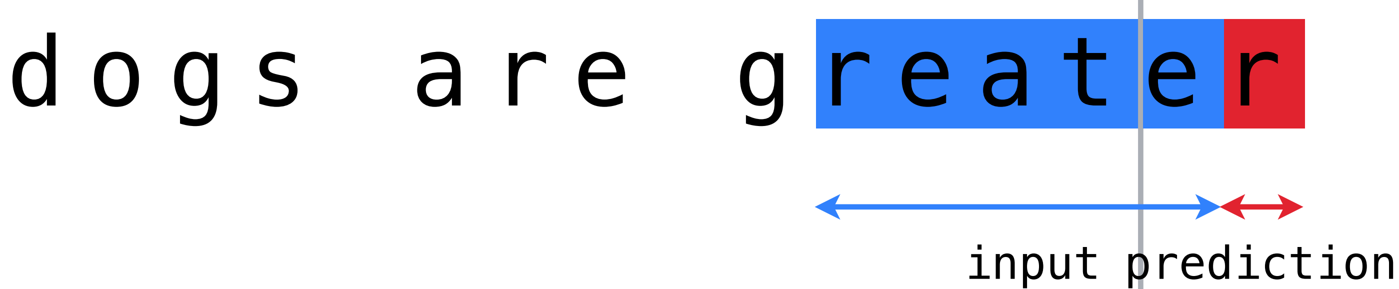
training

testing

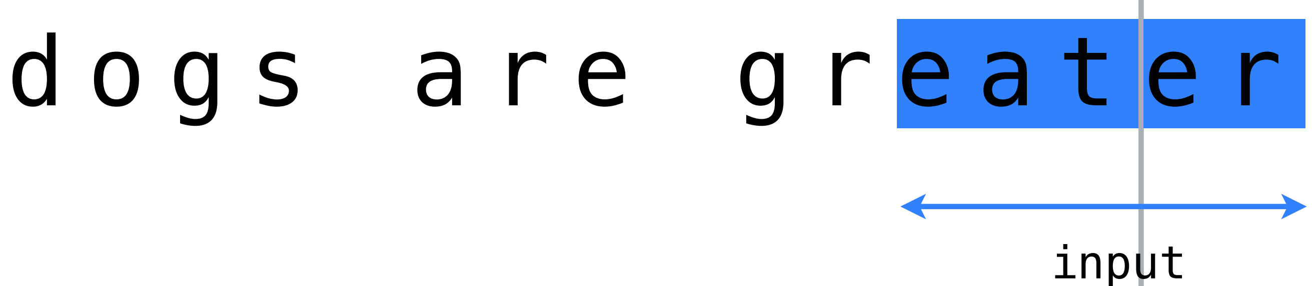


Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$

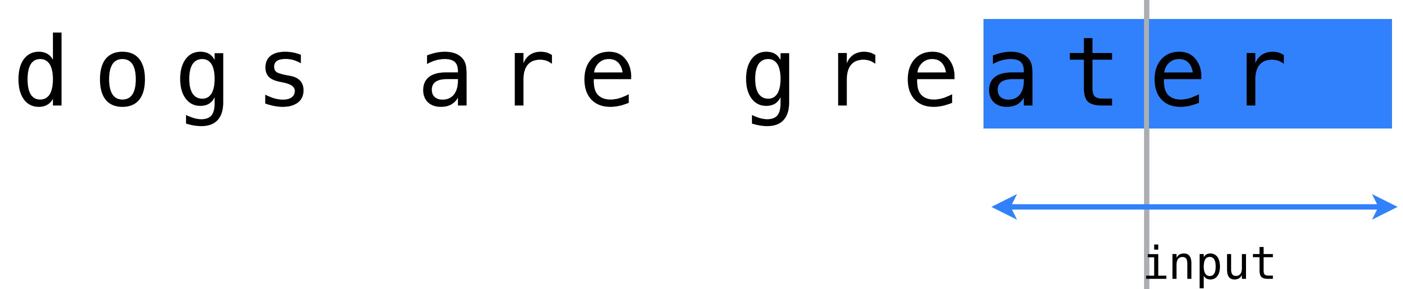


Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$



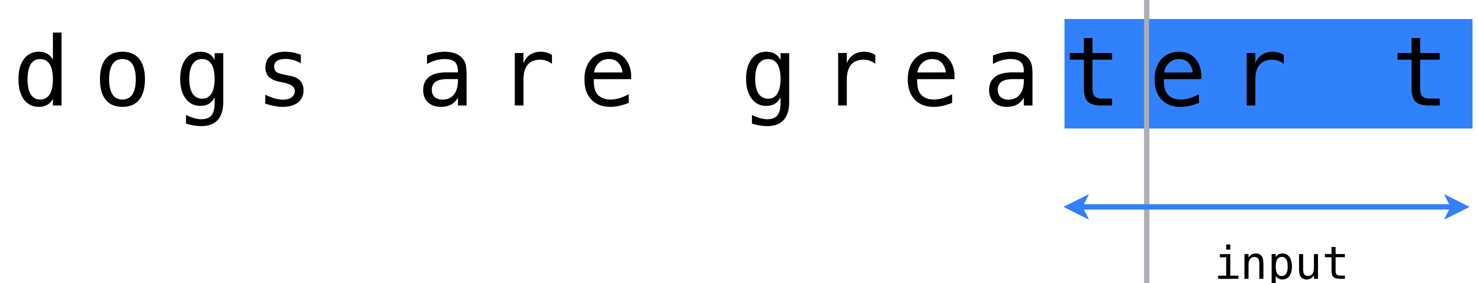
Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$



Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$



Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$
$\vdots$	$\vdots$

here we illustratred with
window size T = 4

Input	Output
$\langle s_{P-3}, s_{P-2}, s_{P-1}, s_P \rangle$	$\hat{s}_{P+1}$
$\langle s_{P-2}, s_{P-1}, s_P, \hat{s}_{P+1} \rangle$	$\hat{s}_{P+2}$
$\langle s_{P-1}, s_P, \hat{s}_{P+1}, \hat{s}_{P+2} \rangle$	$\hat{s}_{P+3}$
$\langle s_P, \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3} \rangle$	$\hat{s}_{P+4}$
$\langle \hat{s}_{P+1}, \hat{s}_{P+2}, \hat{s}_{P+3}, \hat{s}_{P+4} \rangle$	$\hat{s}_{P+5}$
$\vdots$	$\vdots$

dogs are greater than

(character pre-processing)

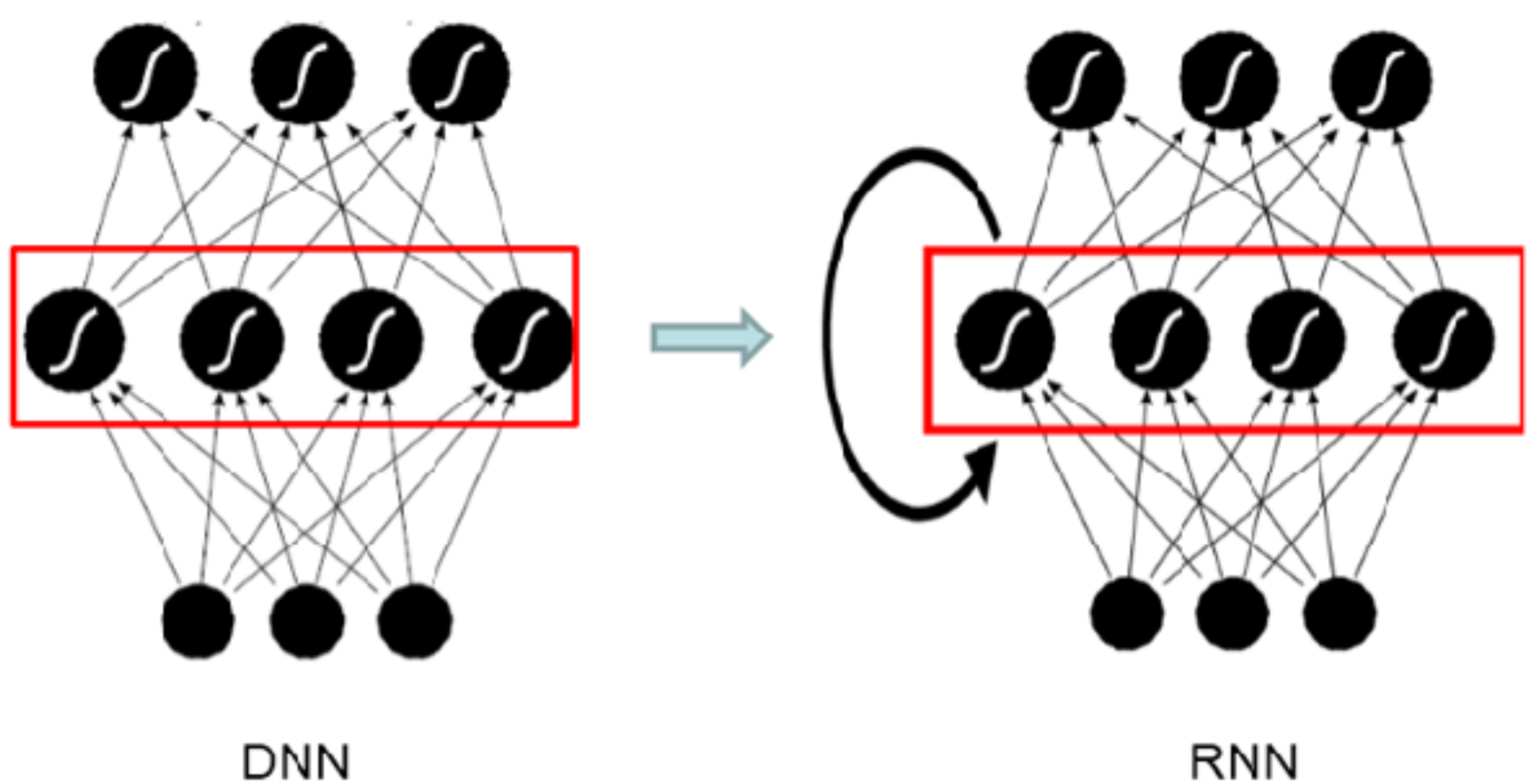
- characters —> numbers for supervised models
- use one-hot encoding scheme

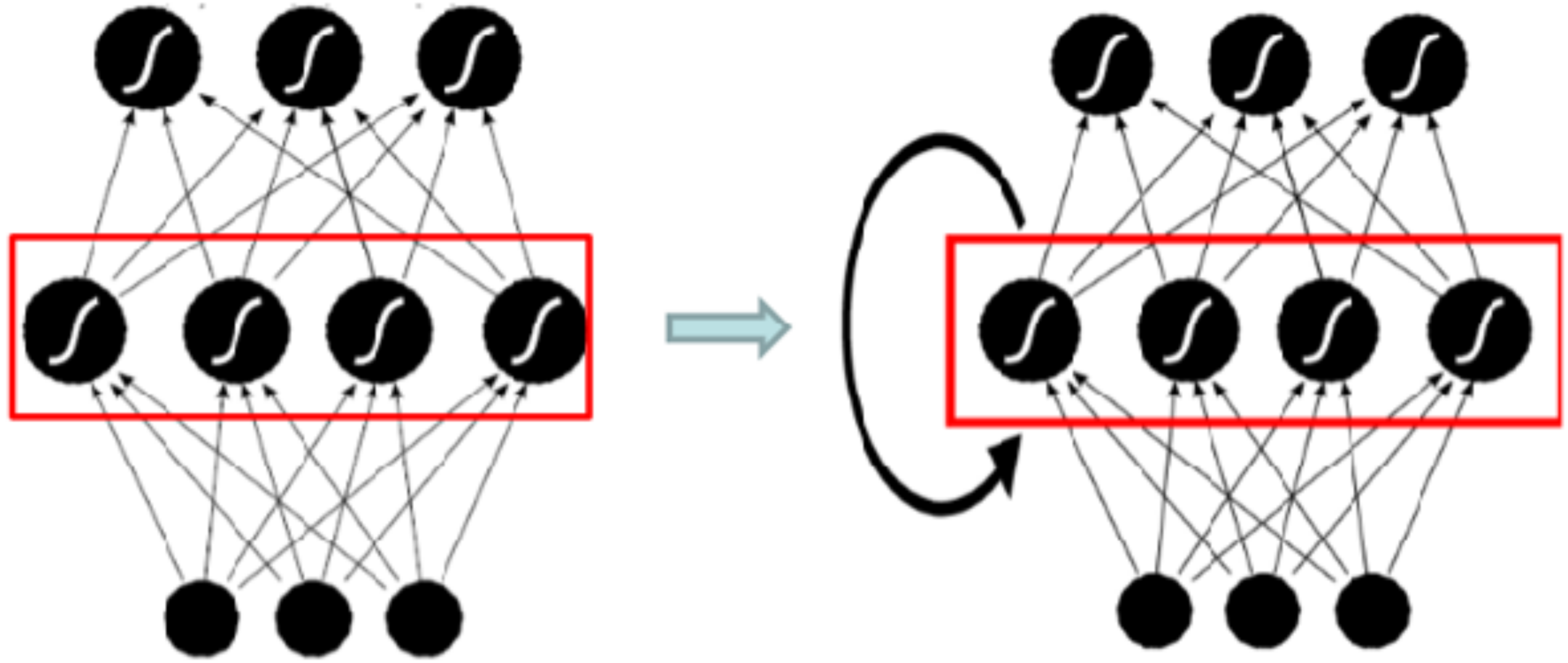
$$a \longleftarrow \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad b \longleftarrow \begin{bmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \quad c \longleftarrow \begin{bmatrix} 0 \\ 0 \\ 1 \\ \vdots \\ 0 \\ 0 \end{bmatrix} \cdots$$

RNN architecture

RNN architecture

- **Want:** parameterized supervised learning model that enforces ordered sequential-ness on I/O
- So: extend feedforward nets to ingest sequences





DNN RNN

NN

feed input in **as vector**

$$h = \tanh\left(v_0 + \sum_{t=1}^{T} v_t s_t\right)$$

$$\hat{y} = b + wh$$

$$(y - \hat{y})^2$$

$$h_0 = \tanh\left(v_0 + v s_0\right)$$

$$h_t = \tanh\left(v_0 + v s_t + u h_{t-1}\right)$$

$$\hat{y} = b + w h_T$$

$$(y - \hat{y})^2$$

RNN technical issues

- RNN still trained via gradient descent (a.k.a. backprop)
- Similar 'vanishing gradient' problem as with feedforward nets
- Using different activation (relu) helps, but additional unit architecture helps a lot (<u>Long Term Short Memory</u> module)

SUMMARY

Supervised learning

+

structured data

Beyond vanilla

- Vanilla models don’t exploit ordered sequential I/O
- Include I/O structure in learning framework —> better results
    - Engineer into fixed feature extraction
    - Iimbed parameterized feature extractor in model (e.g., convnets, RNNs)
- RNNs one parameterized way to exploit sequential I/O
- RNNs are natural extension of feedforward nets, and inherit similar technical issues