Deep learning for sequential data

Artificial Intelligence for Digital Health (AID)
M.Sc. in Digital Health – University of Pisa
Davide Bacciu (davide.bacciu@unipi.it)

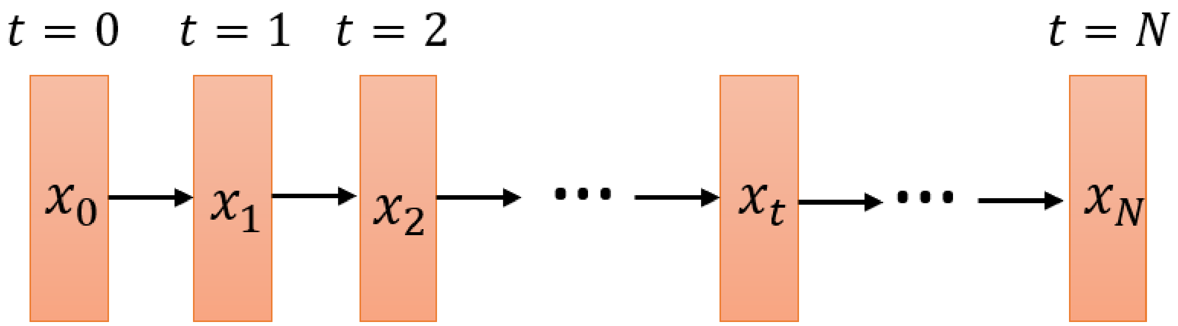
Lecture Outline

- Sequential data in healthcare
  - Dealing with sequential data and learning tasks definition
  - Physiological timeseries
  - Electronic health records
- Recurrent neural networks (RNNs)
  - Main intuition and learning issue in the vanilla model
  - Gated RNNs
  - Bidirectional models
  - Convolutional RNNs
- RNNs in healthcare applications (with a bonus track on models)

Sequential Data (in Healthcare)

Sequences

- Ordered series of observations of variable length
- Each element of the sequence is (possibly) a vector (multivariate)
- Sequence elements can be sampled at irregular times



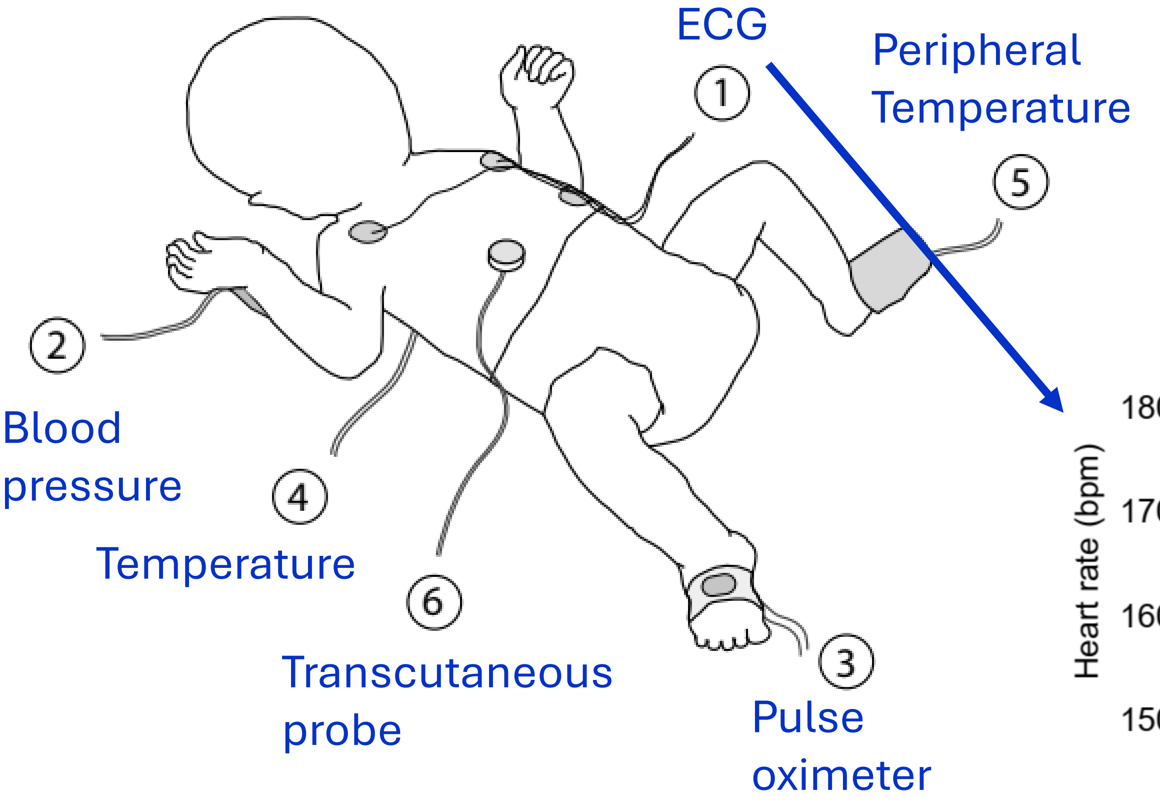
Inductive Bias
The element at time t in the sequence may depend only on its (more or less) recent past

Kinds of Sequential Data

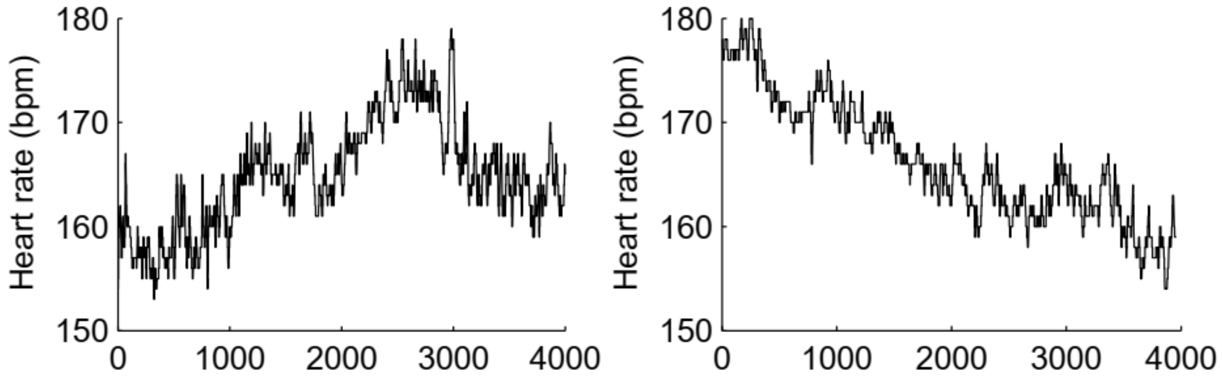
- When ordering is given by time, our sequence is also known as a timeseries
- Numerical sequences: each element is a scalar (e.g. heartrate)
- Vectorial sequences: each element is a vector (e.g. ECG/EEG)
- Matrix sequences: each element is a matrix (e.g. an fMRI)
- Textual sequences: each element is the encoding of a symbolic item (e.g. genomic sequences)

Physiological Timeseries

Probes used to collect vital signs data from an infant in ICU

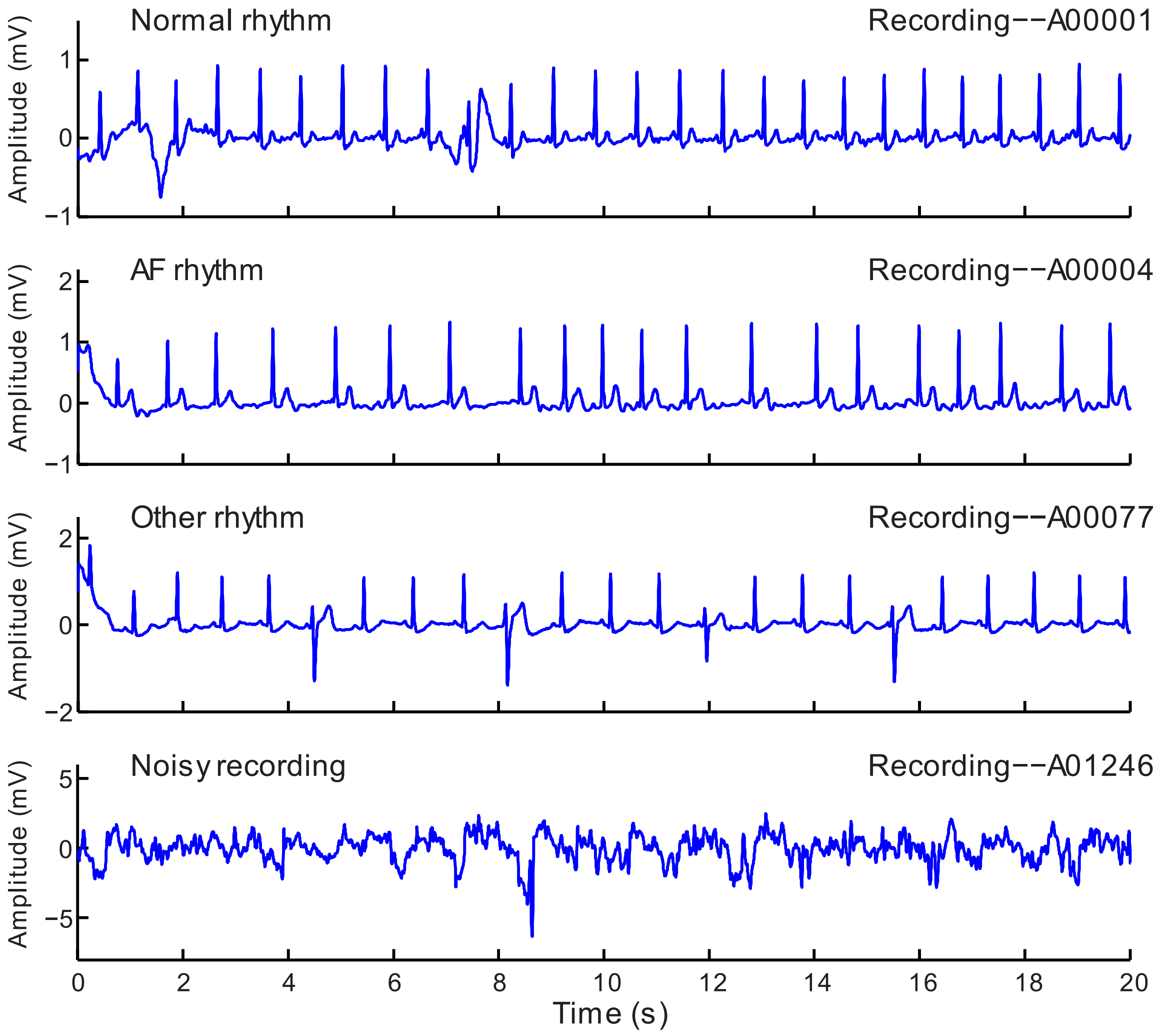


Rapidly varying HR timeseries (normal)



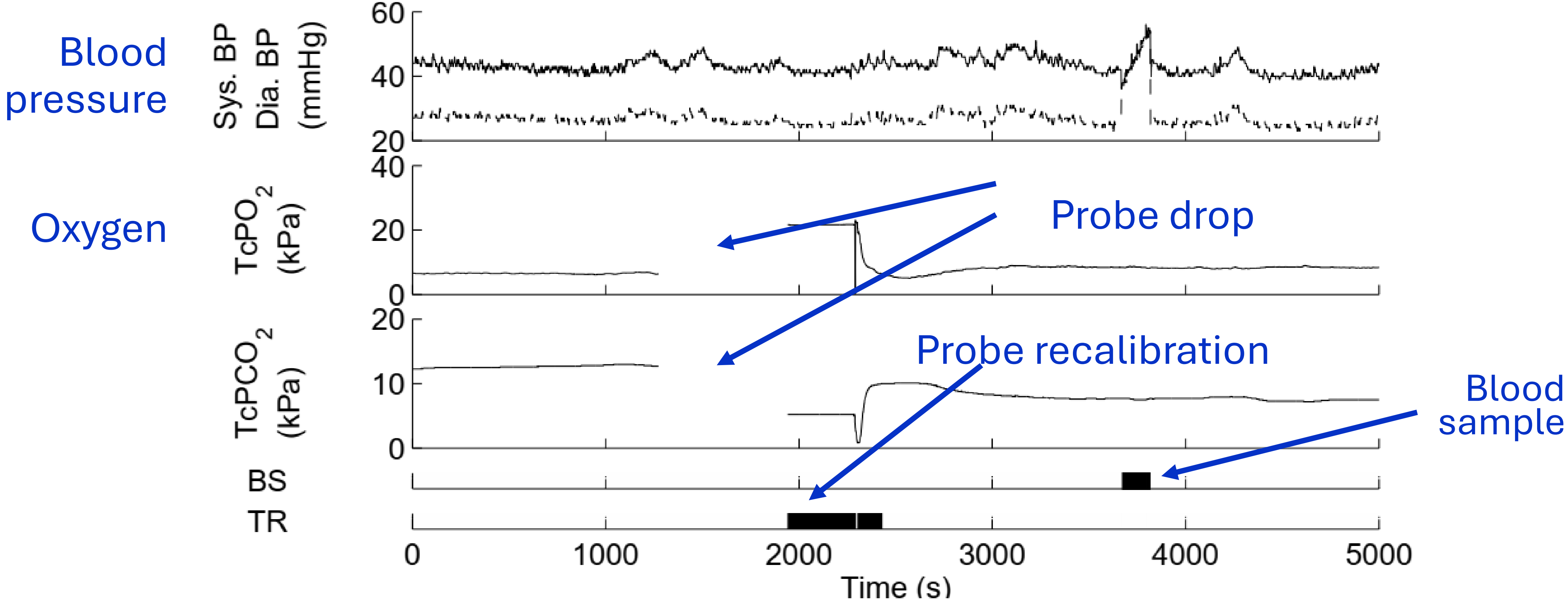
Source:Quinn et al., TPAMI 2008

The Challenging Nature of Physiological Timeseries (I)



Source: [Physionet]

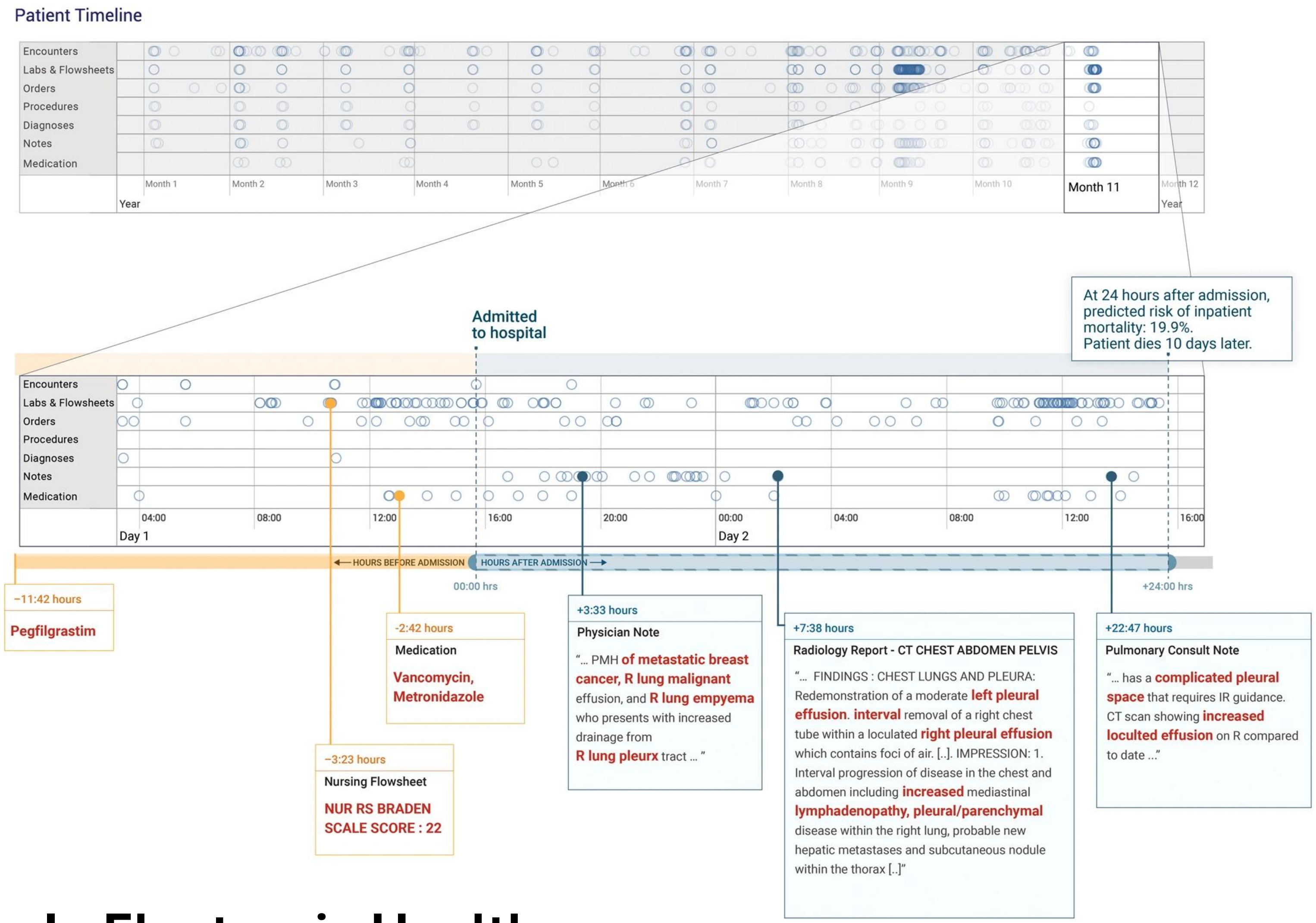
The Challenging Nature of Physiological Timeseries (II)



Source:Quinn et al., TPAMI 2008

Source: Rajkomar et al, Nature 2018

Fantastic timeseries and where to find them

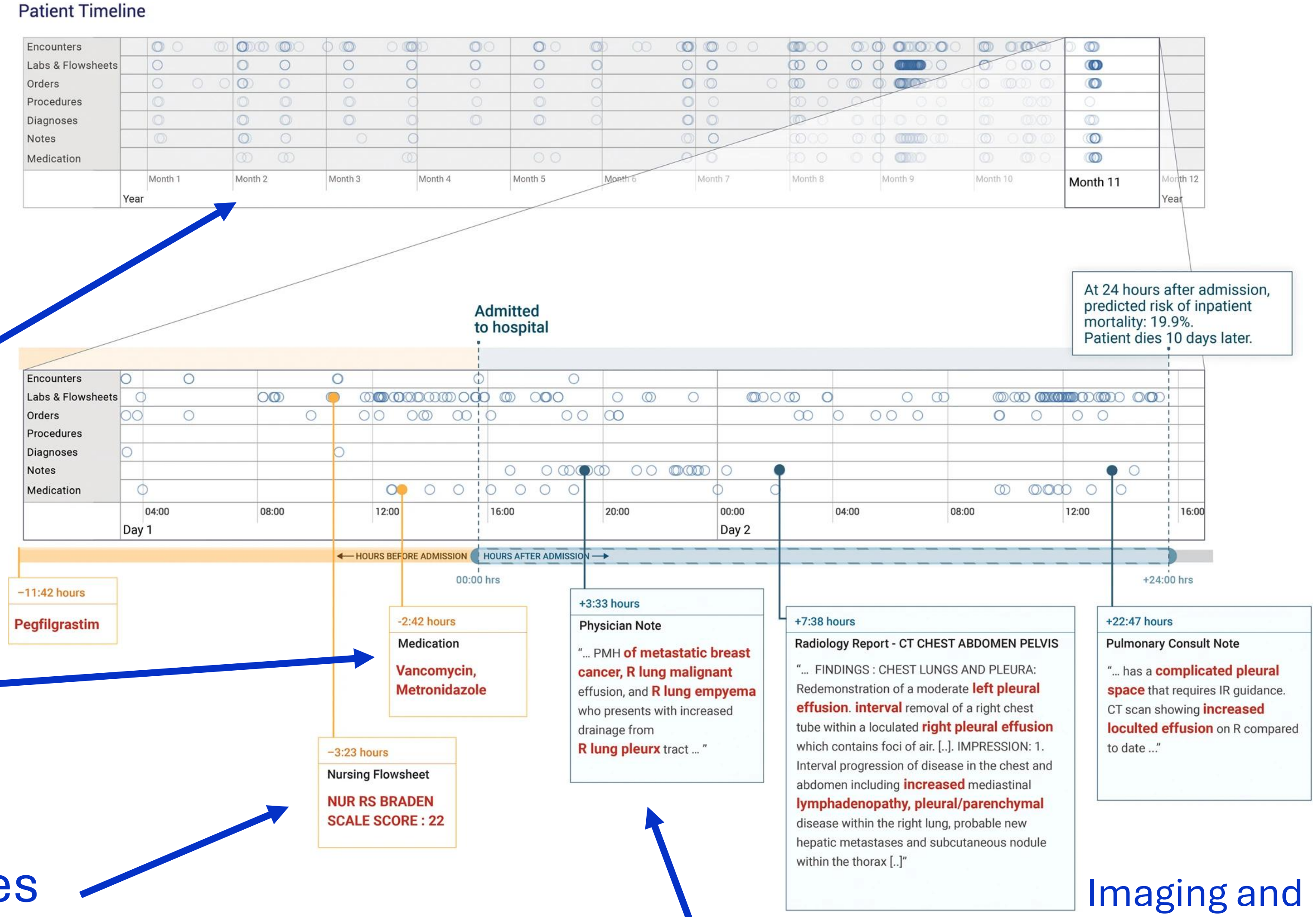


In Electronic Health Records (EHR)!

Electronic Health Record

Source: Rajkomar et al, Nature 2018

Patient chart in digital form, containing medical and treatment history



Patient information stored over time

Medications

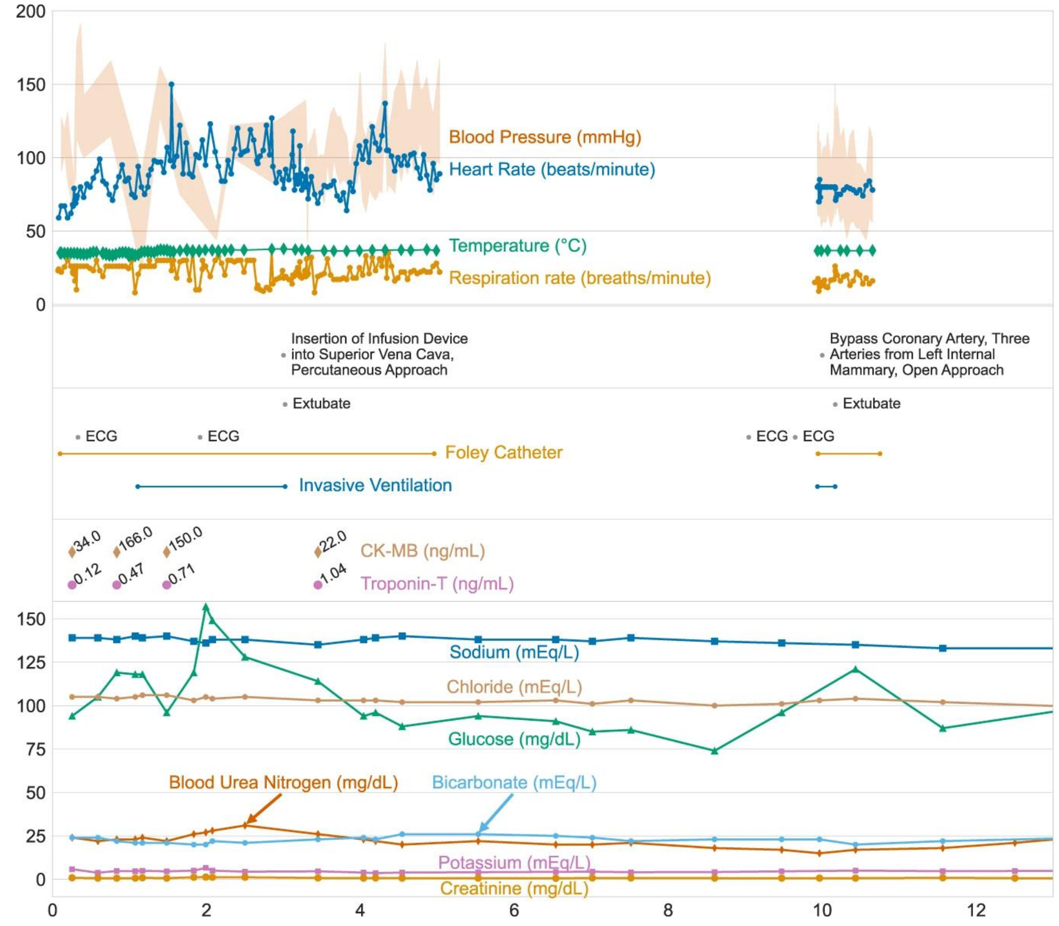
Nursing notes

Diagnoses and physician notes

Imaging and lab reports

EHR Example Dataset – MIMIC-III/IV

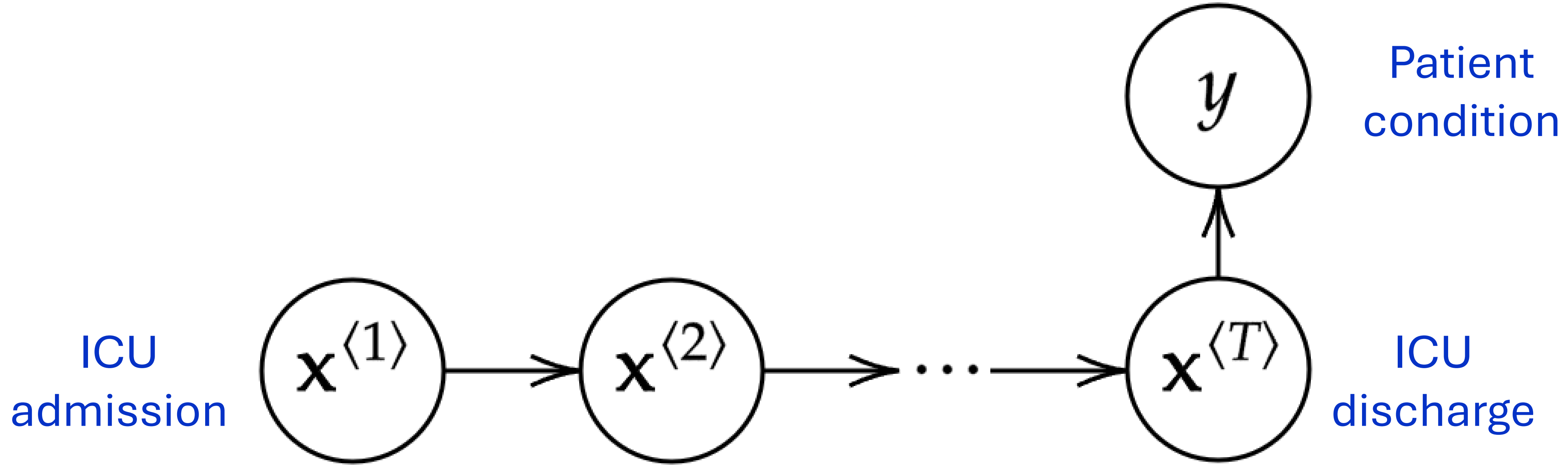
- Open-source database of de-identified data for 65,000 patients admitted to an ICU and over 200,000 patients admitted to the emergency department
- All patients admitted to critical care units at Beth Israel Deaconess Medical Center (Boston, MA) between 2008 - 2019



Johnson et al, Nature 2023

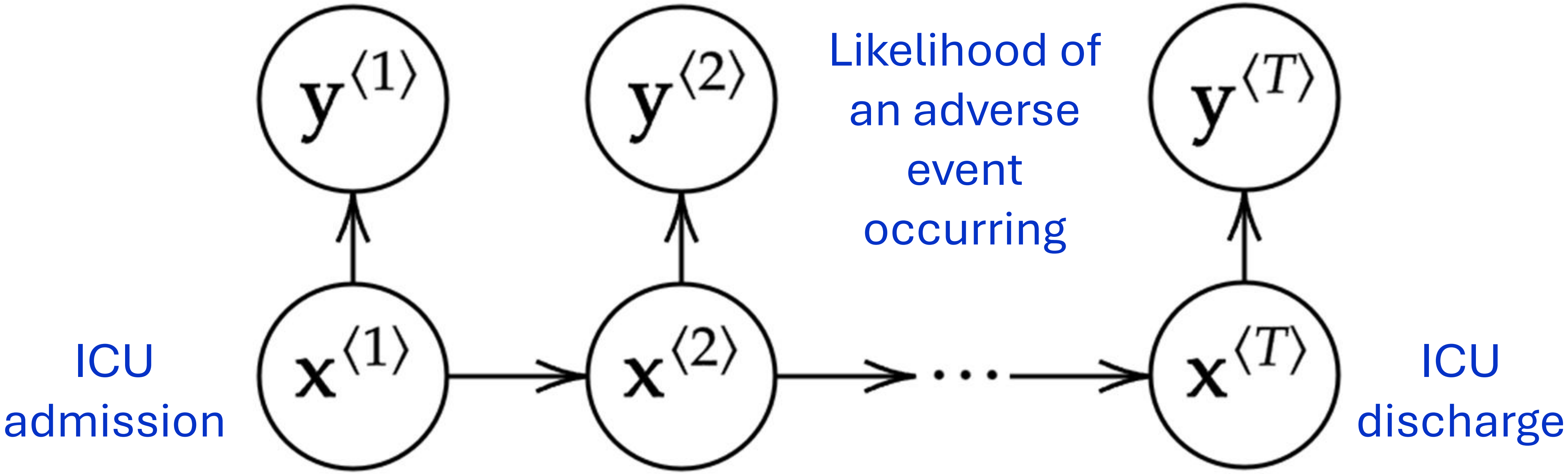
Type of sequential ML tasks: sequence prediction

The entire sequence **x** is associated with a single target **y**



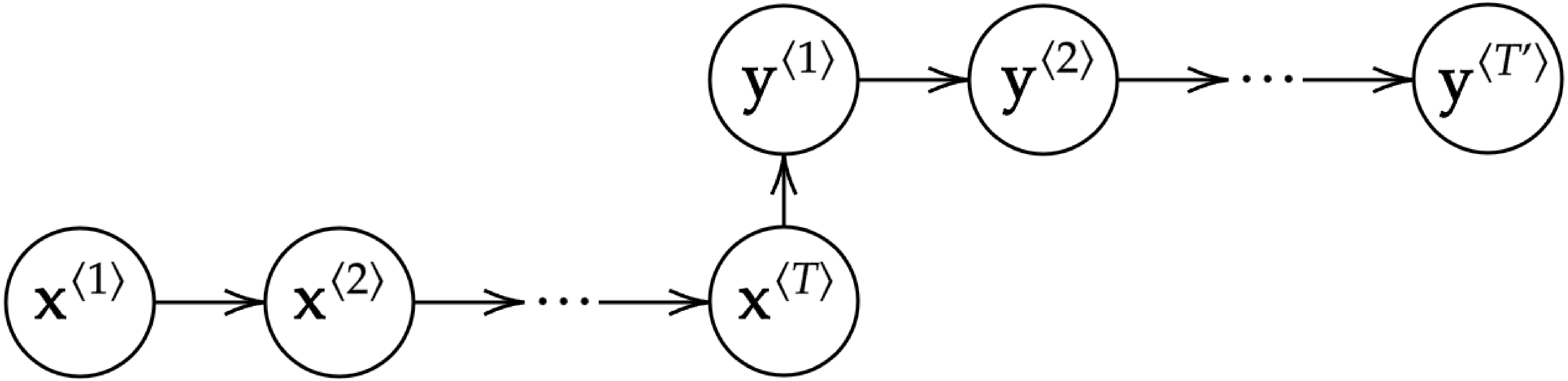
Type of sequential ML tasks: element-by-element prediction

Given a sequence **x** generate a prediction $y^{<t>}$ for each element

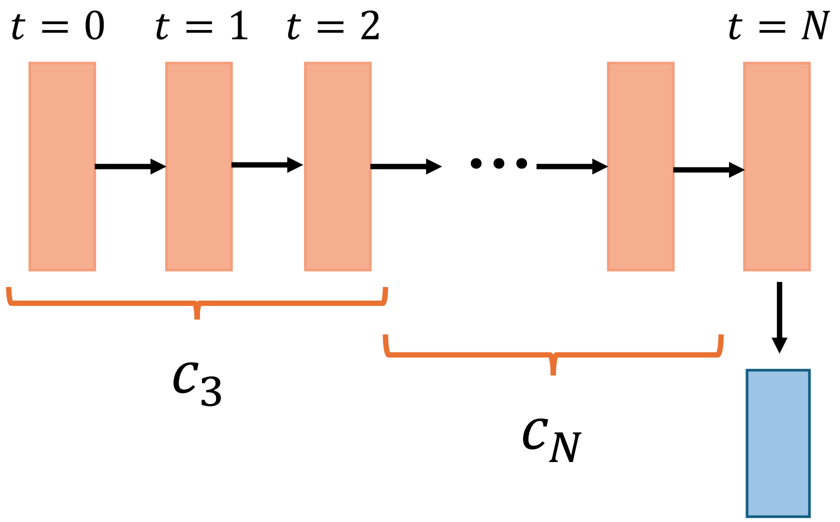


Type of sequential ML tasks: sequence-to-sequence

Given a sequence $\mathbf{x}$ generate an output sequence $\boldsymbol{y}$ (of different length and not synchronized)



Dealing with Sequences in Neural Networks



Variable size data describing sequentially dependent information

Neural models need to capture dynamic context c_t to perform predictions

Recurrent Neural Network

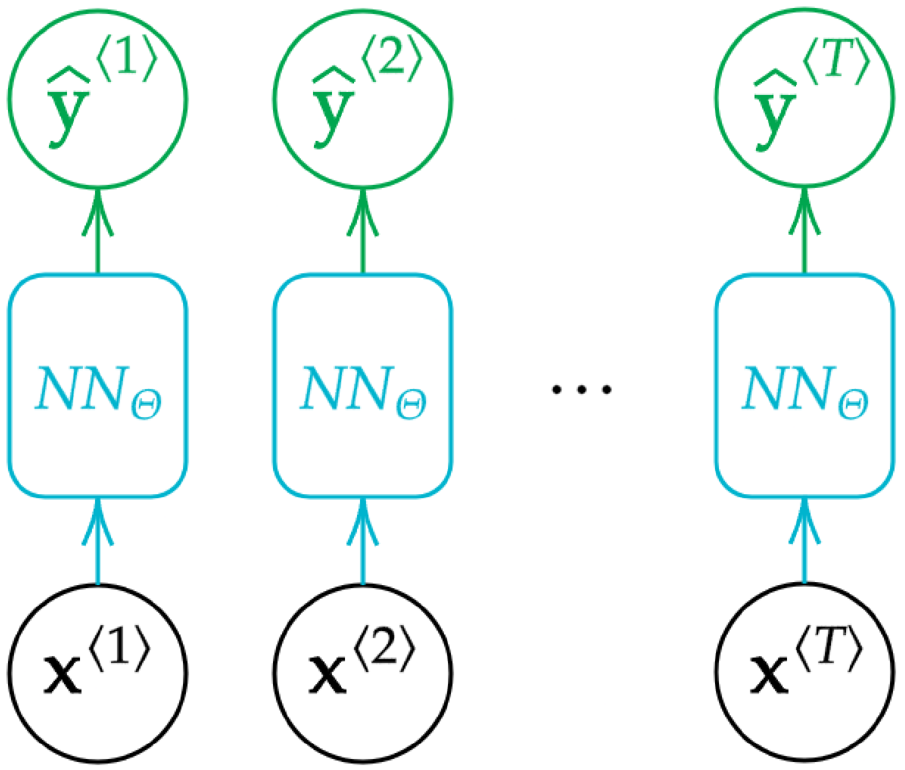
- Vanilla adaptive models (Elman, SRN, …)
- Randomized approaches (Reservoir Computing)
- Gated recurrent networks

Recurrent Neural Networks (RNN)

The intuition

We apply the same neural network to each element of the sequence (using weight sharing)

$$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_{in}\boldsymbol{x}_t)$$

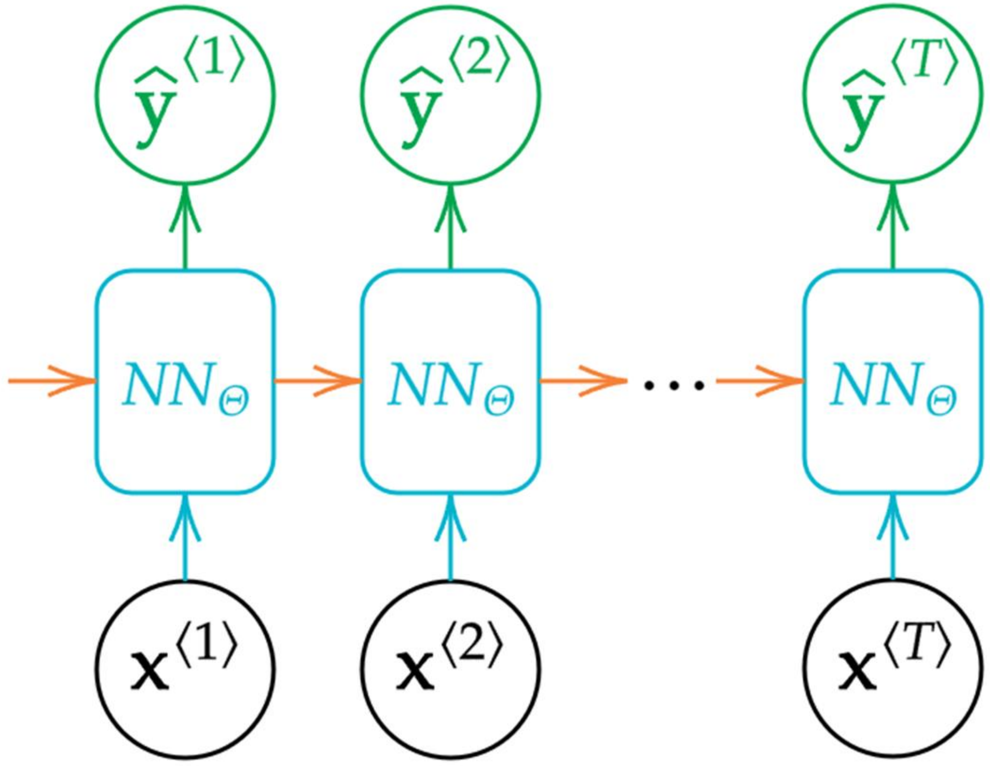


The intuition

We apply the same neural network to each element of the sequence (using weight sharing)

$$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_{in}\boldsymbol{x}_t + \boldsymbol{W}_h\boldsymbol{h}_{t-1})$$

We add a new input $\boldsymbol{h}_{t-1}$ which captures the information from the past inputs of the network

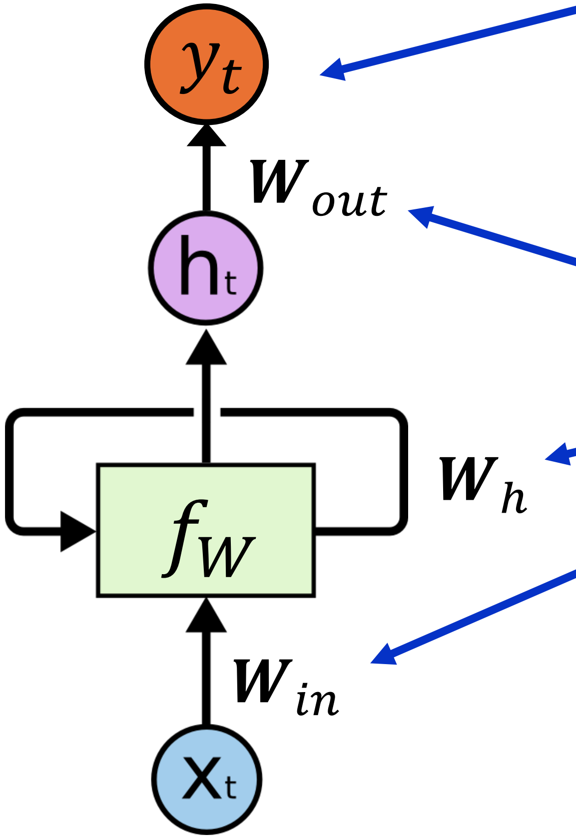


Interpreting the network state $\boldsymbol{h}_{t-1}$

- $\boldsymbol{h}_{t-1}$ encodes the information related to the elements of the sequence $\boldsymbol{x}_1 \dots \boldsymbol{x}_{t-1}$ processed before the current one ($\boldsymbol{x}_t$)
- It acts like a state/memory that summarizes the relevant information the network has processed up to that point
- The RNN flow in summary
  - We combine the current element of the sequence $\boldsymbol{x}_t$ with input weights
  - We combine the state $\boldsymbol{h}_{t-1}$ with recurrent weights
  - We sum the two results and apply an activation function
  - We pass the result to the next layer
  - For the first element $\boldsymbol{x}_1$, the state $\boldsymbol{h}_0$ is a vector of zeros

Recursive RNN model

Describes the network structure and parameterization prior to unfolding (i.e. applying the weight sharing copies) on the actual sequence



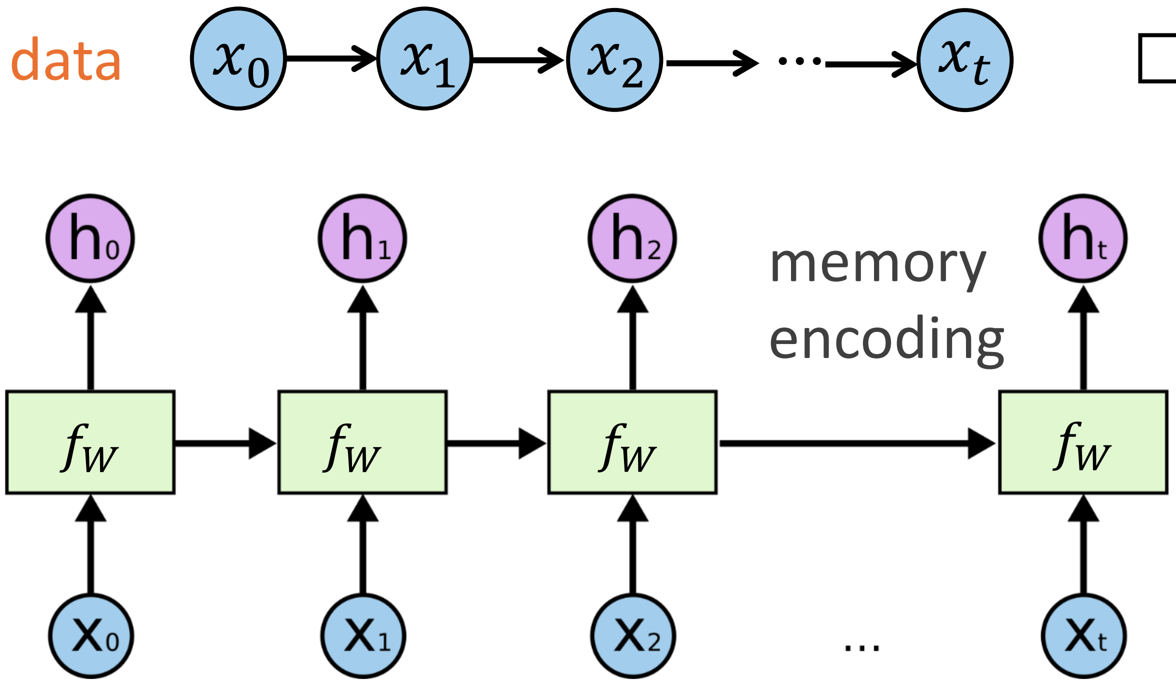
This is not necessarily computed for all t

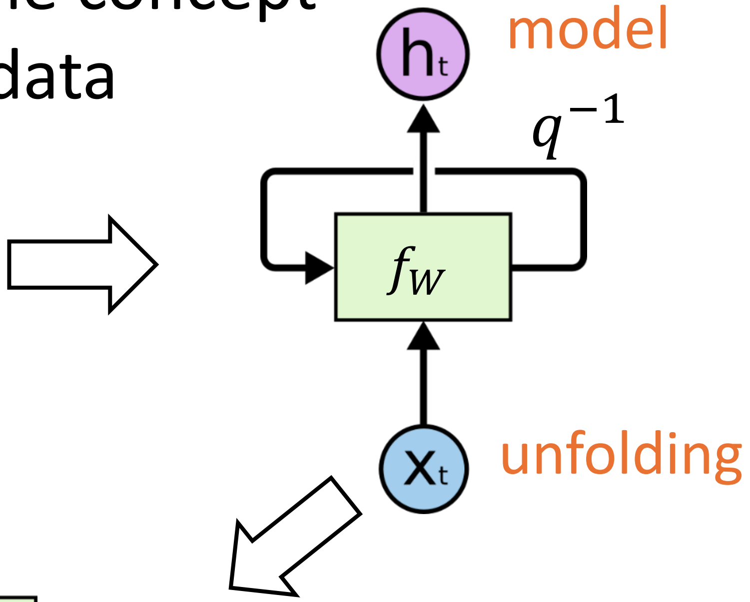
Typically all dense (fully-connected) layers

Graphics credit
@ colah.github.io

Unfolding RNN (Forward Pass)

By now you should be familiar with the concept of model unfolding/unrolling on the data

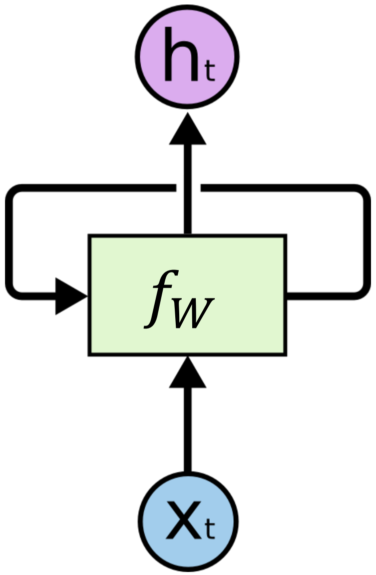




Map an arbitrary length sequence $x_0 .. x_t$ to fixed-length encoding $\boldsymbol{h}_t$

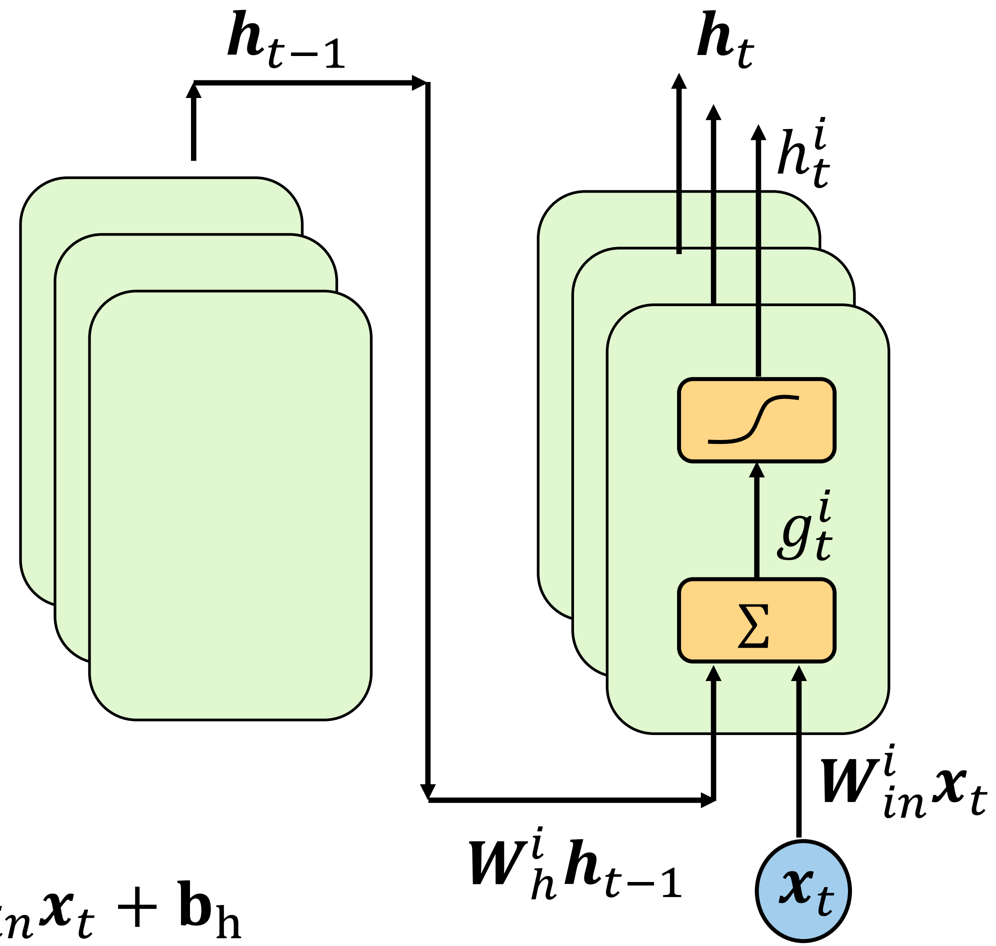
Vanilla RNN

$$\boldsymbol{y}_t = f(\boldsymbol{W}_{out}\boldsymbol{h}_t + \boldsymbol{b}_{out})$$

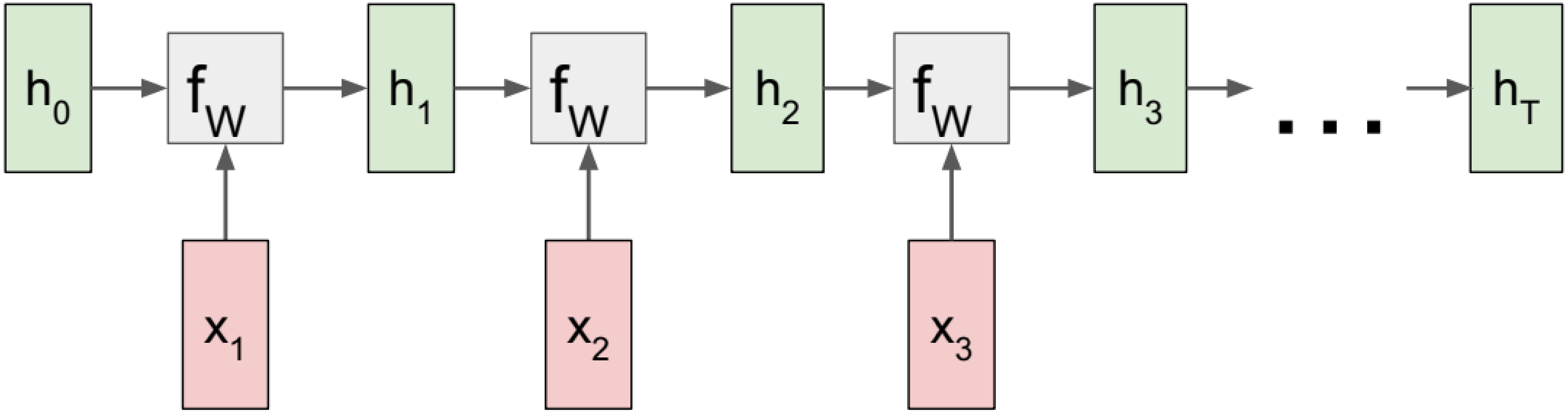


$$\boldsymbol{h}_t = tanh(\boldsymbol{g}_t)$$

$$\boldsymbol{g}_t(\boldsymbol{h}_{t-1}, \boldsymbol{x}_t) = \boldsymbol{W}_h\boldsymbol{h}_{t-1} + \boldsymbol{W}_{in}\boldsymbol{x}_t + \mathbf{b}_\mathrm{h}$$



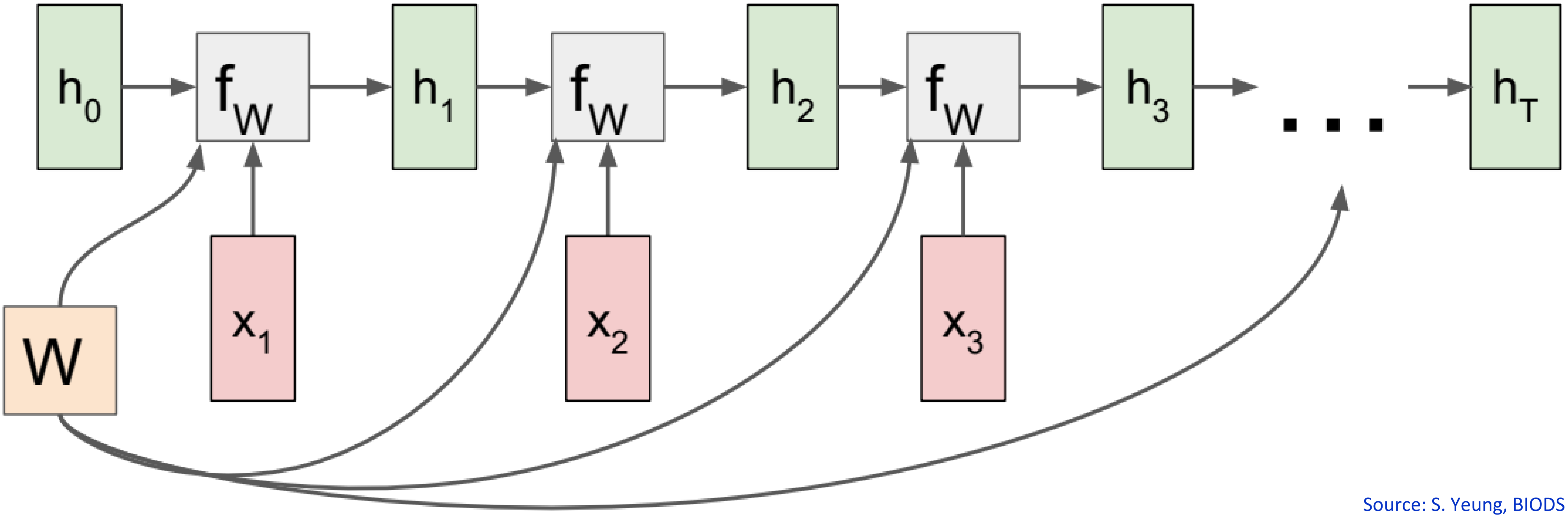
Training RNNs – Computational Graph



Source: S. Yeung, BIODS 220

Training RNNs – Computational Graph

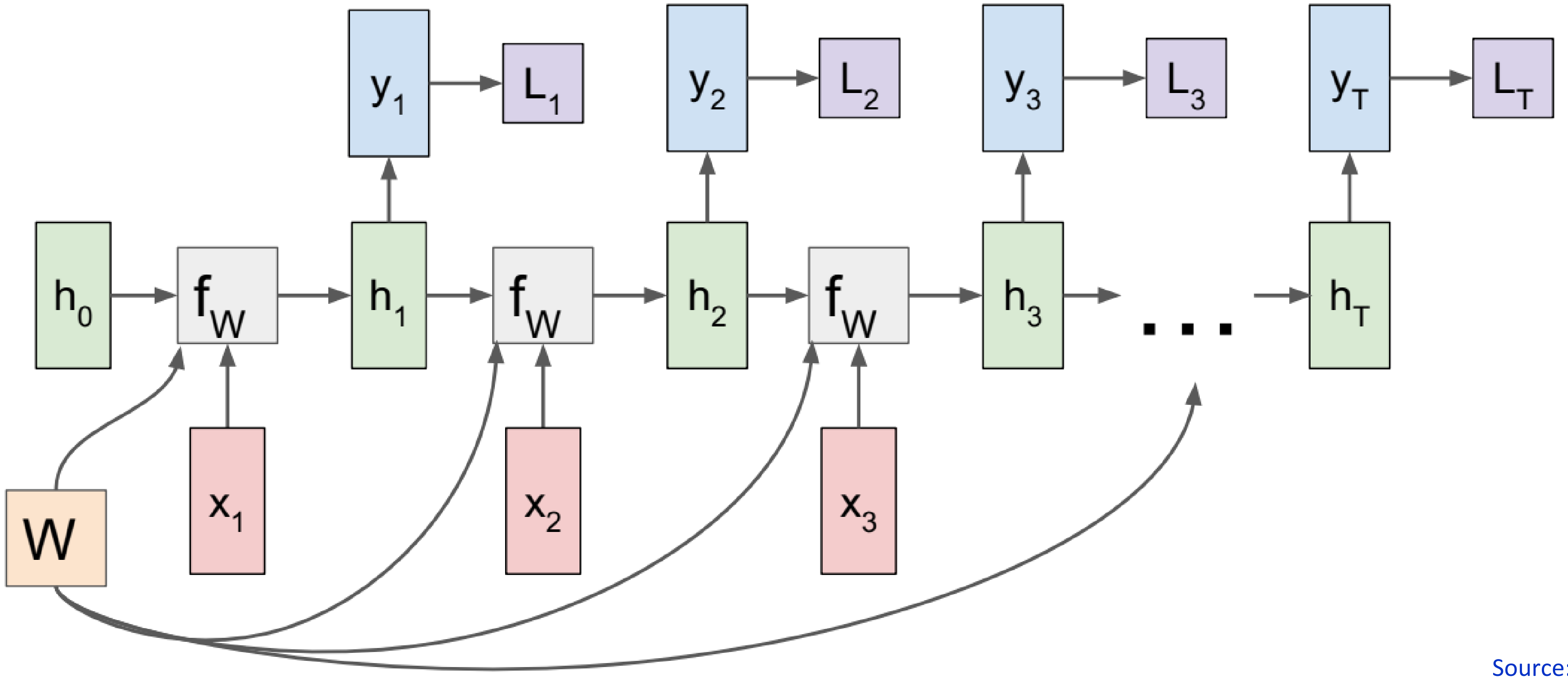
Same set of weights reused across time steps => gradient needs to be taken w.r.t. all weight copies



Source: S. Yeung, BIODS 220

Training RNNs – Computational Graph

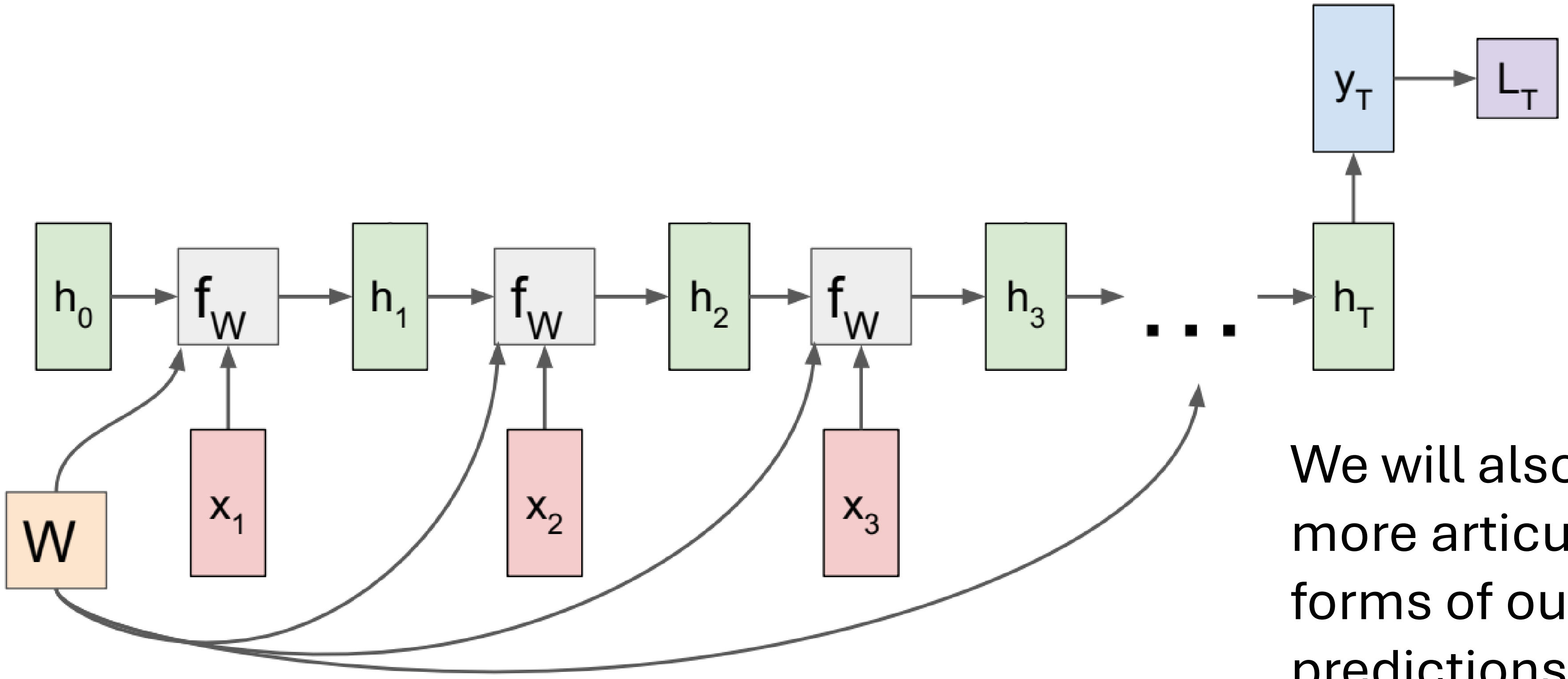
When predicting at each time step, adjust based on the error committed at each time step (i.e. sum the errors across time)



Source: S. Yeung, BIODS 220

Training RNNs – Computational Graph

For sequence-levels tasks, have one error (and one gradient) only at the end

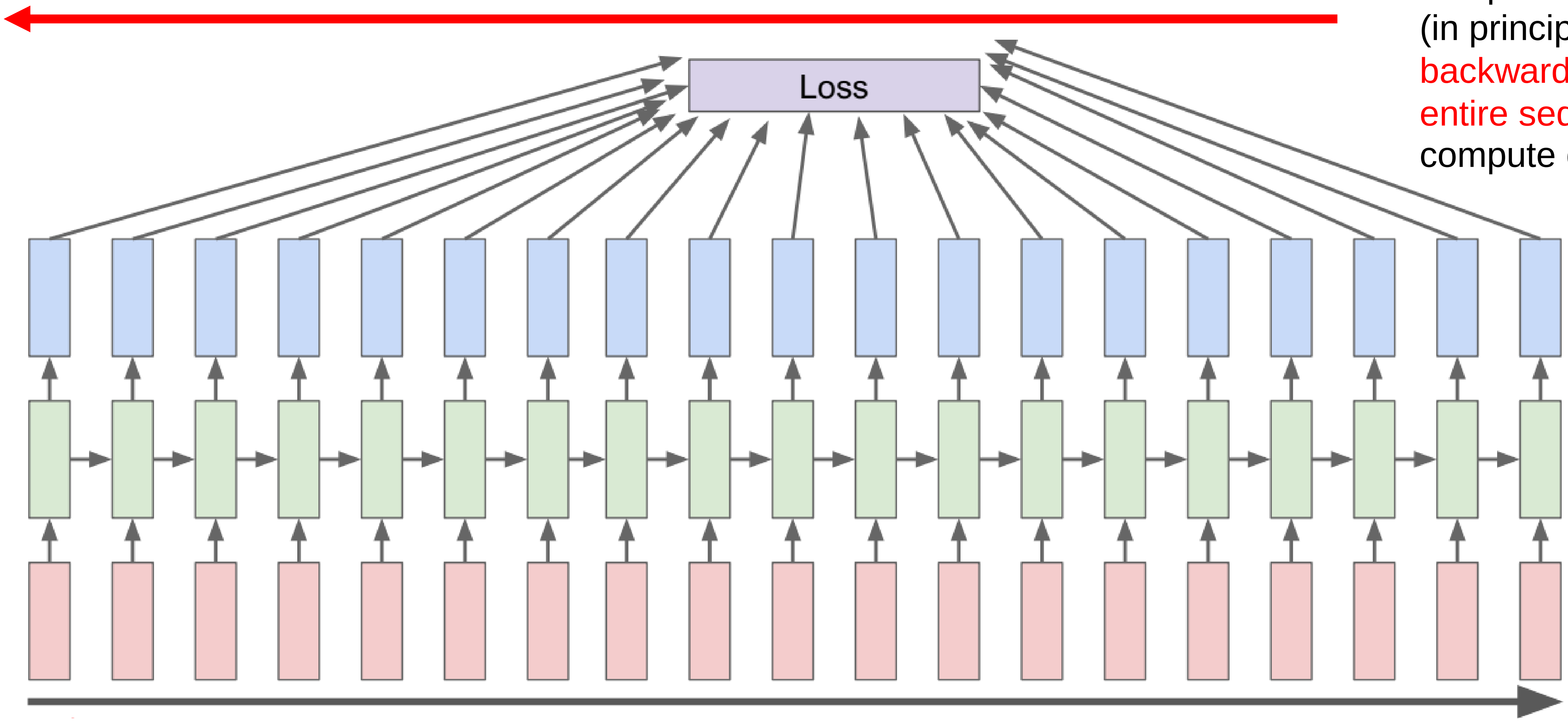


We will also see more articulated forms of output predictions later on

Source: S. Yeung, BIODS 220

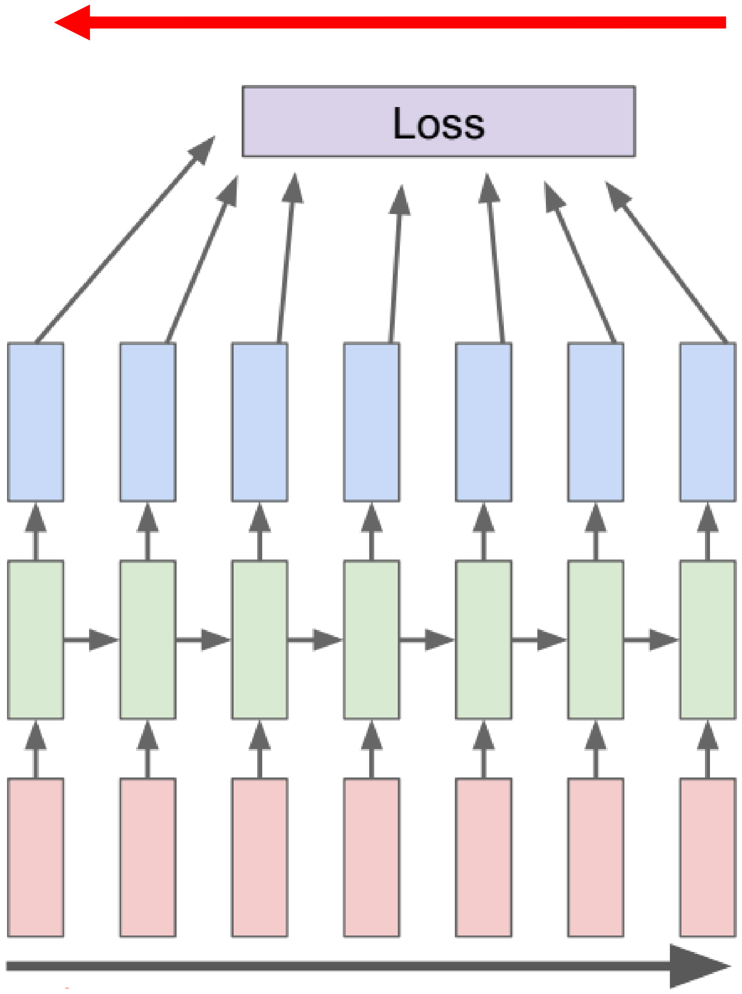
Backpropagation Through Time (BPTT)

Forward through entire sequence to compute loss, then (in principle) backward through entire sequence to compute gradient



Source: S. Yeung, BIODS 220

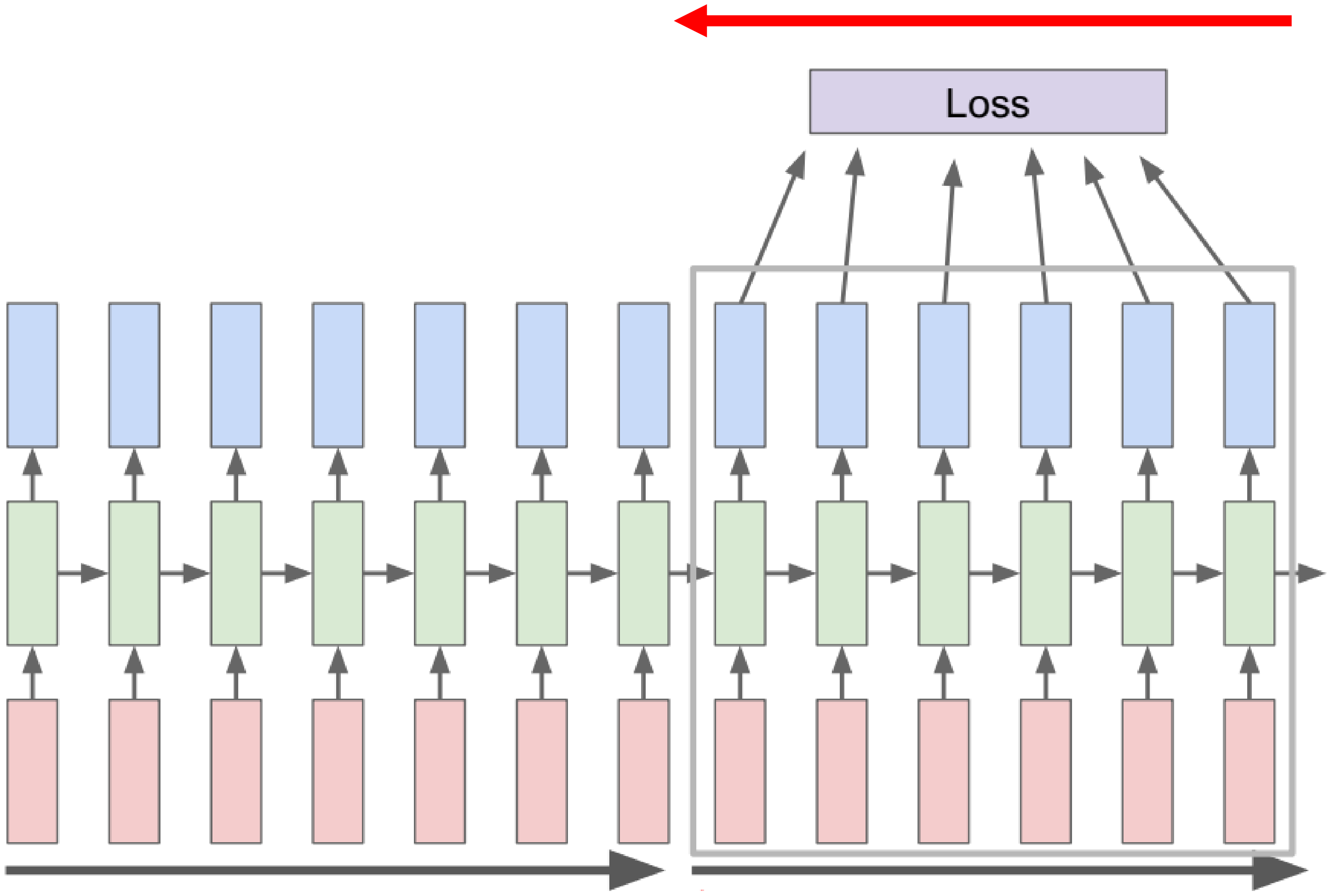
Truncated Backpropagation Through Time



Gradient tends to vanish (or explode) as you propagate it through many time steps backwards (for numerical reasons beyond the scope of this course)

Source: S. Yeung, BIODS 220

Truncated Backpropagation Through Time

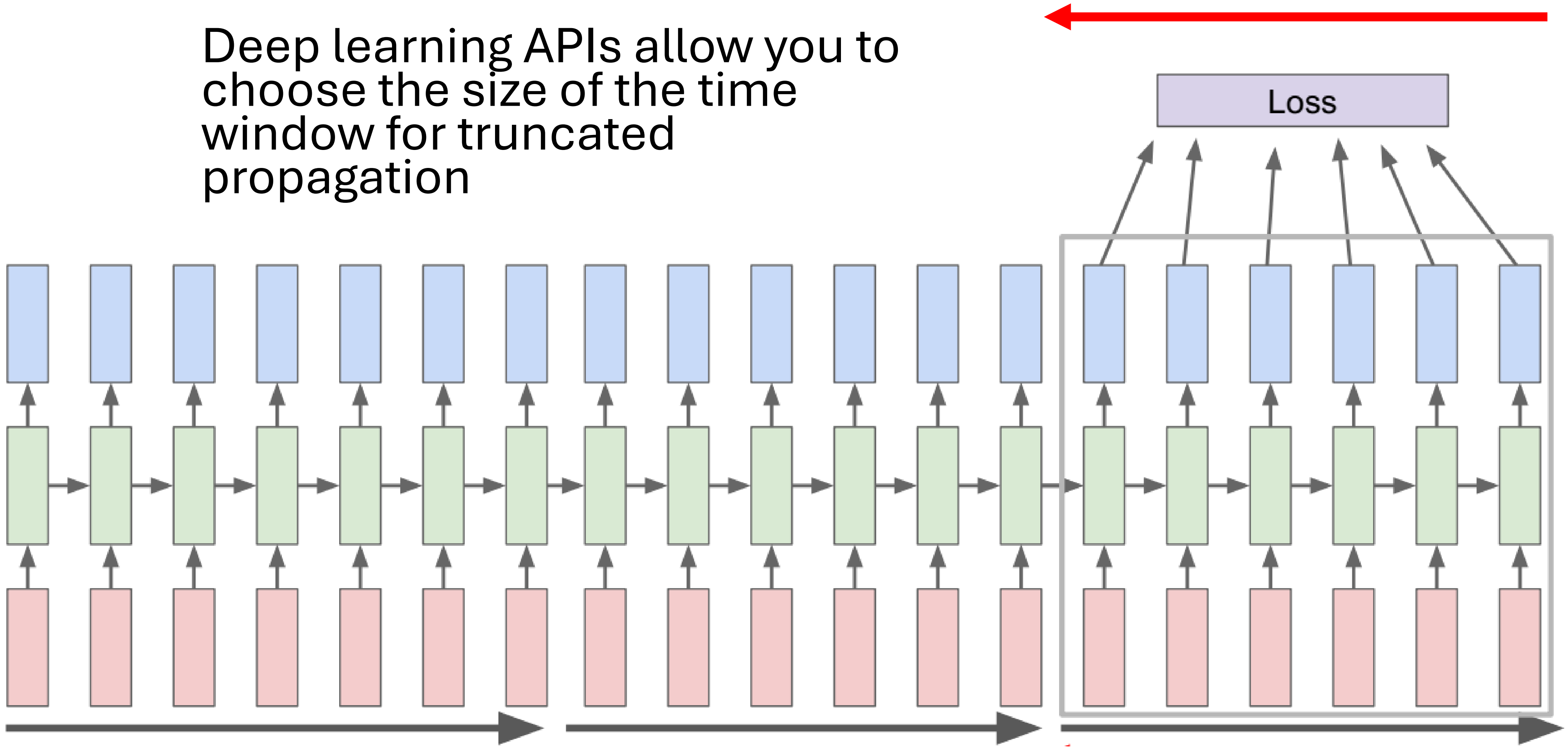


Run BPTT on chunks of the sequence rather than on the full sequence

Source: S. Yeung, BIODS 220

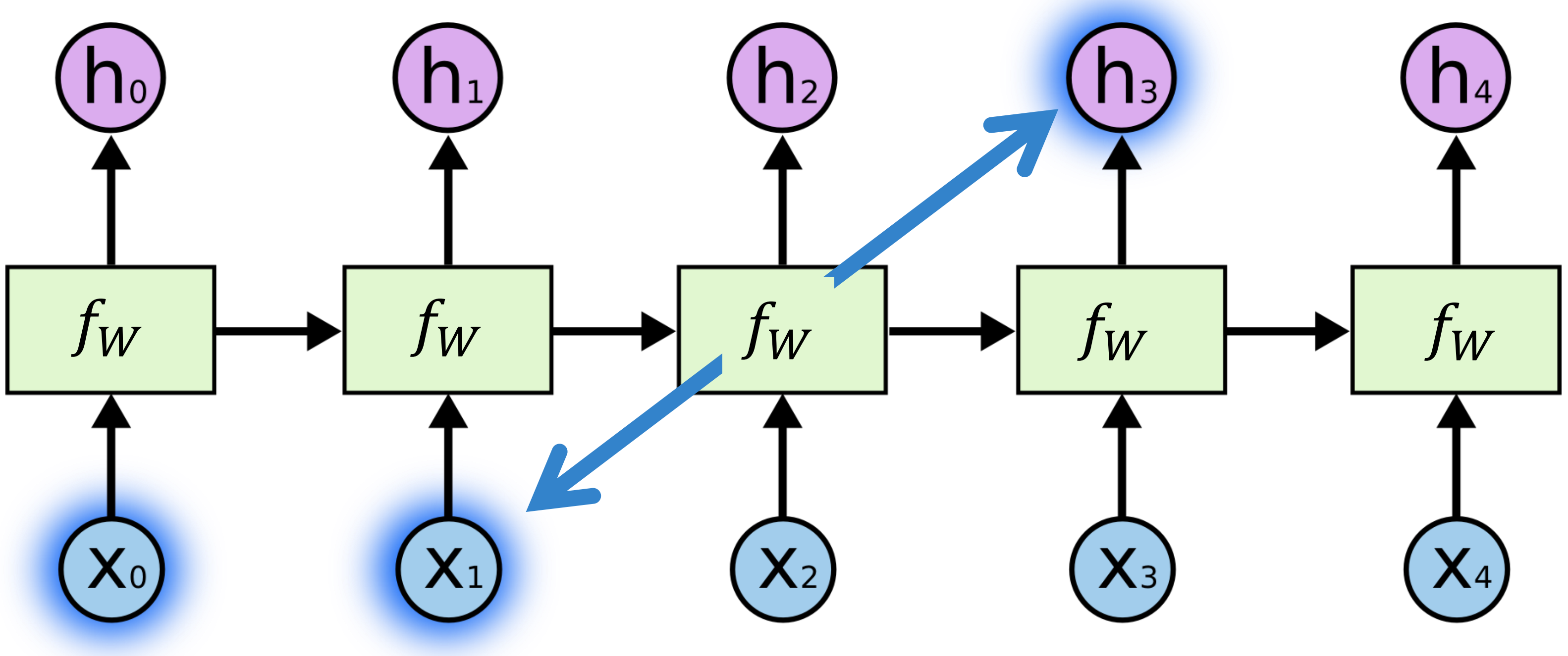
Truncated Backpropagation Through Time

Deep learning APIs allow you to choose the size of the time window for truncated propagation



Source: S. Yeung, BIODS 220

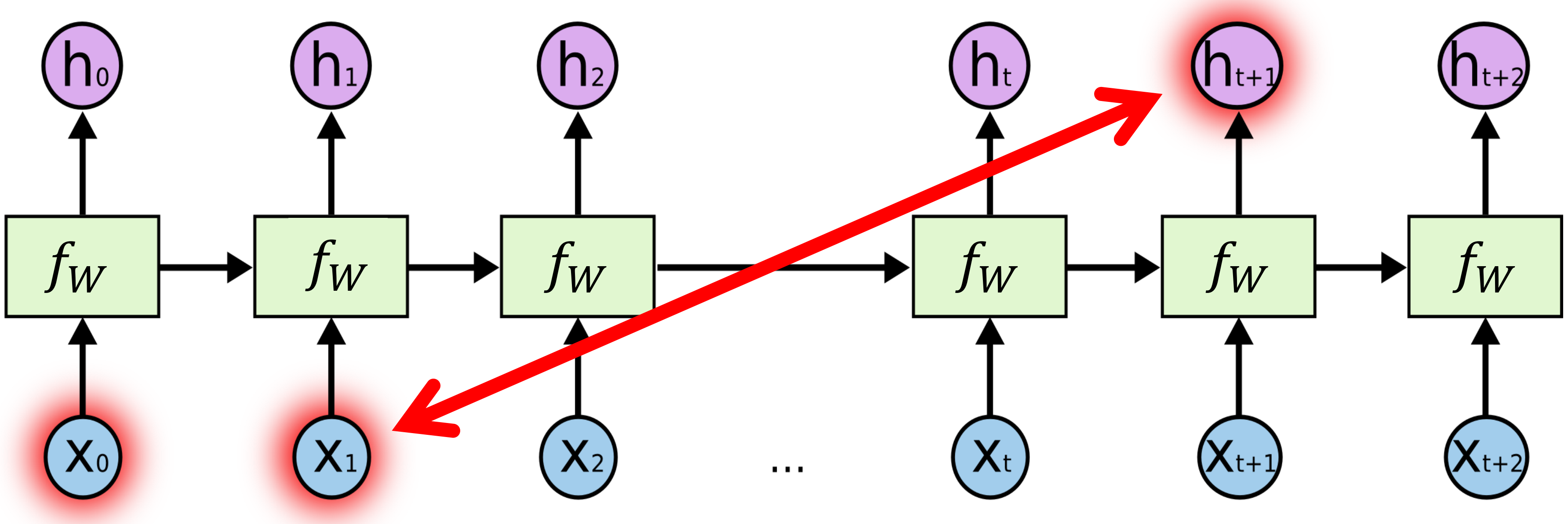
Learning to Encode Input History



Hidden state $\boldsymbol{h}_t$ summarizes information on the history of the input signal up to time t

Learning Long-Term Dependencies is Difficult

When the time gap between the observation and the state grows there is little residual information of the input inside of the memory



Gated Recurrent Networks

A motivating example

- Let's imagine we need to predict the next word in this sentence:

  I lived in England when I was little, until I was ten years old. Then I moved with my family. I speak fluently…

- It’s clear that if I want to predict the next word in this sentence (*English*), I need to remember having seen *England* earlier
- The problem with standard RNNs is that this dependency might be lost, so modifications to the standard RNN are needed to solve this issue

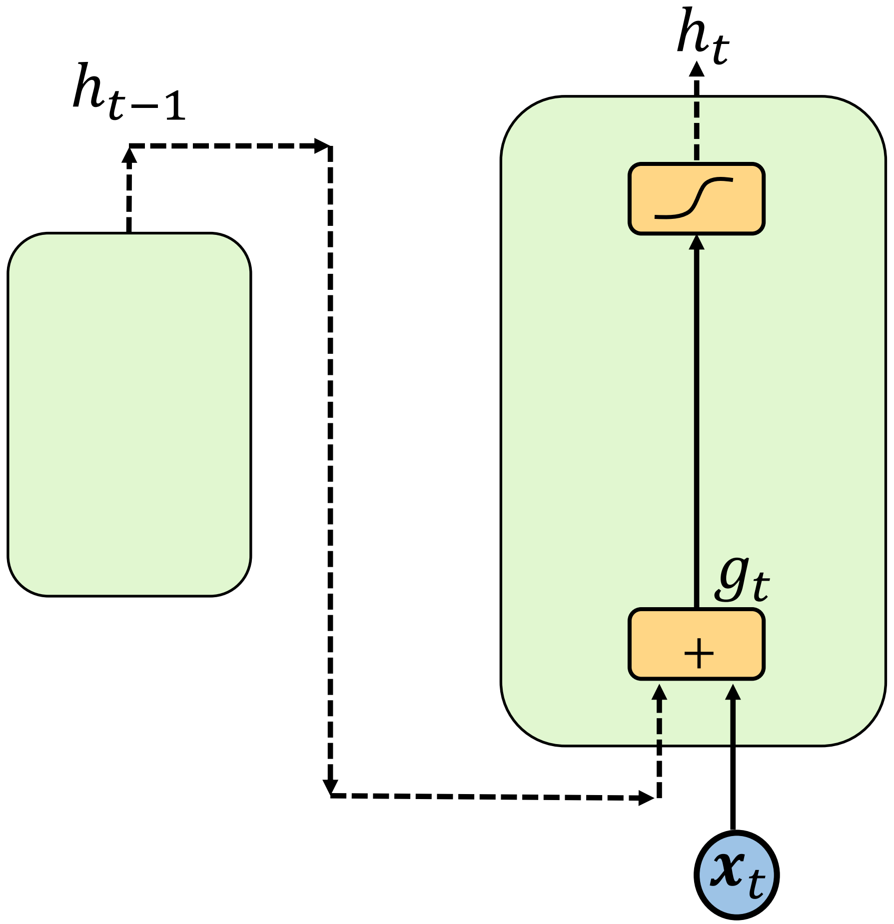
Long Short Term Memory (LSTM) – The first gated RNN

- The idea behind an LSTM is to introduce a memory **c**, a vector that holds a representation of elements (no matter how far back) that the current output/state might depend on
- In a simple RNN, the memory **c** coincides with the state **h** and it contains all past input elements
  - By trying to retain "everything", the network tends to "forget" the more distant elements (due to the vanishing gradient problem)
- The key idea in LSTMs is that, at each step, the network decides whether and how much to update the memory
- This update is managed by *gates* that learn how to combine the memory with the previous state to produce the current state and output.

LSTM Gates

- The forget gate tells us which parts of the memory to erase
- The update gate tells us which parts of the memory to update
- The output gate tells us which parts of the memory are used to compute the output and the current state
- The activation of a gate returns vectors with values between 0 and 1, where 0 = "throw away" and 1 = "keep"

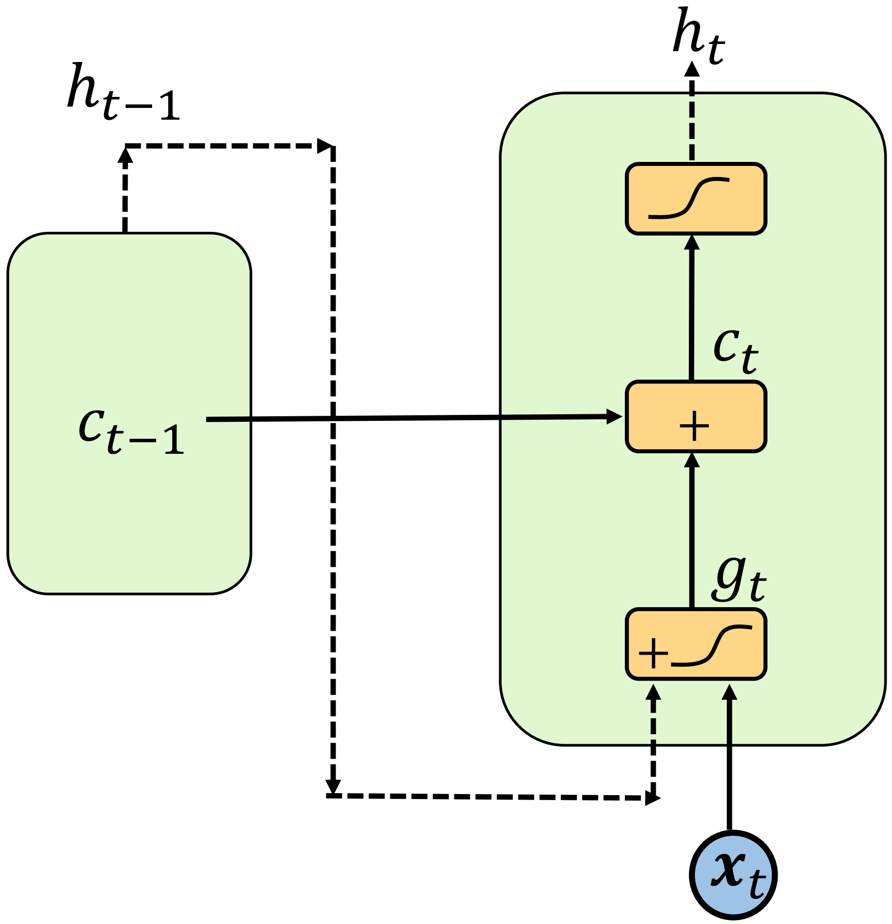
LSTM Design



Let's start from the vanilla RNN unit

S. Hochreiter, J. Schmidhuber, Long short-term memory". Neural Computation, Neural Comp. 1997

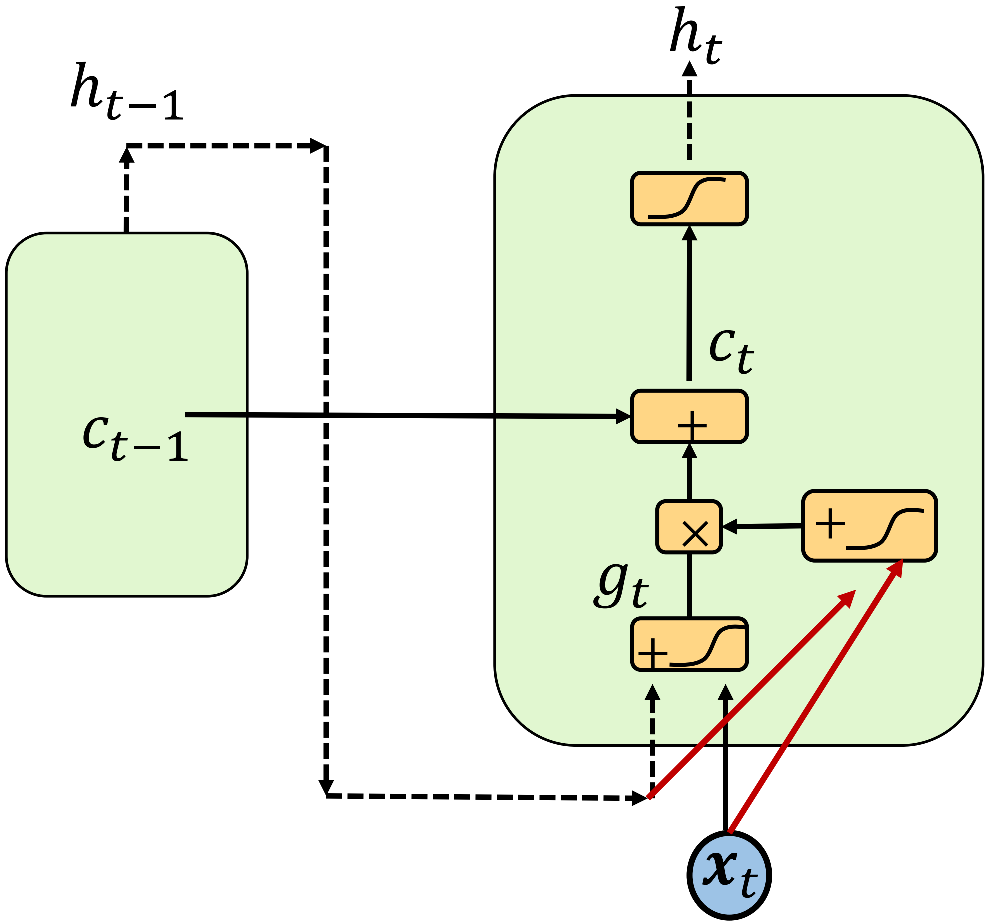
LSTM Design – Step 1



Introduce a memory c_t

Combines past internal state c_{t-1} with current input $\boldsymbol{x}_t$

LSTM Design – Step 2 (Gates)



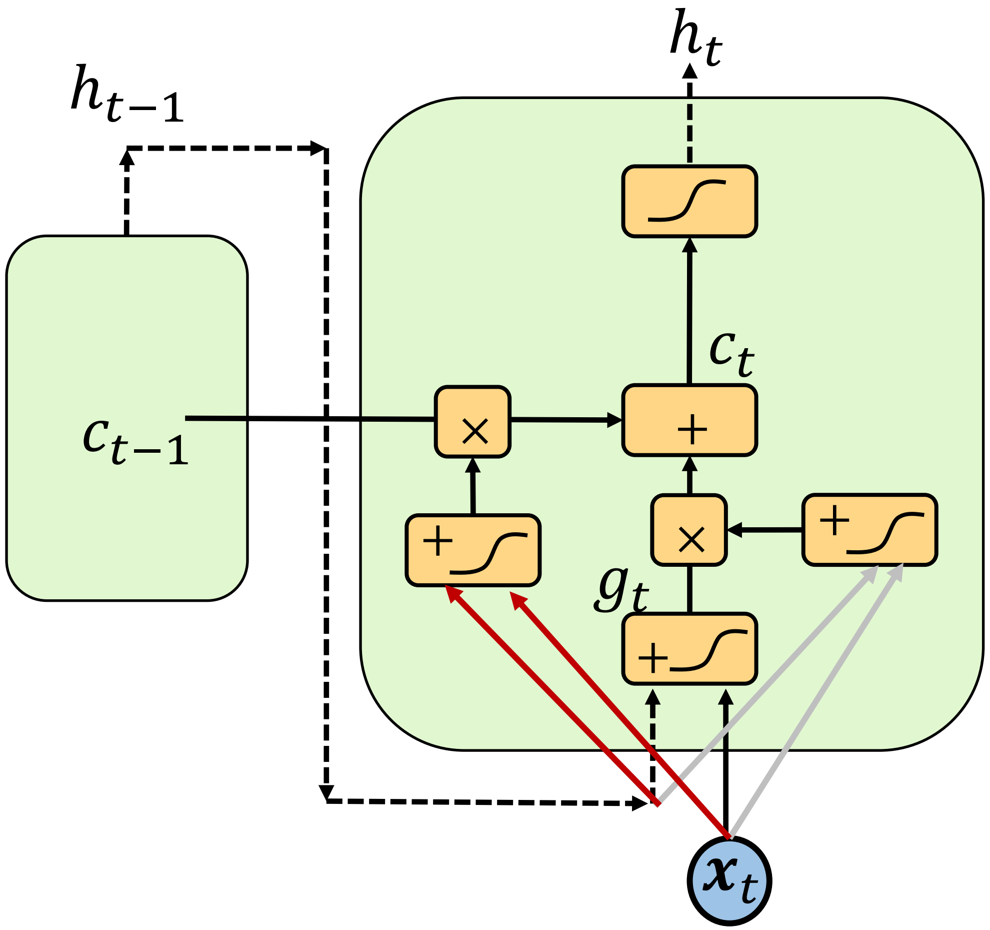
Input gate

Controls how inputs contribute to the internal state

$$I_t(x_t, h_{t-1})$$

Logistic sigmoid

LSTM Design – Step 2 (Gates)



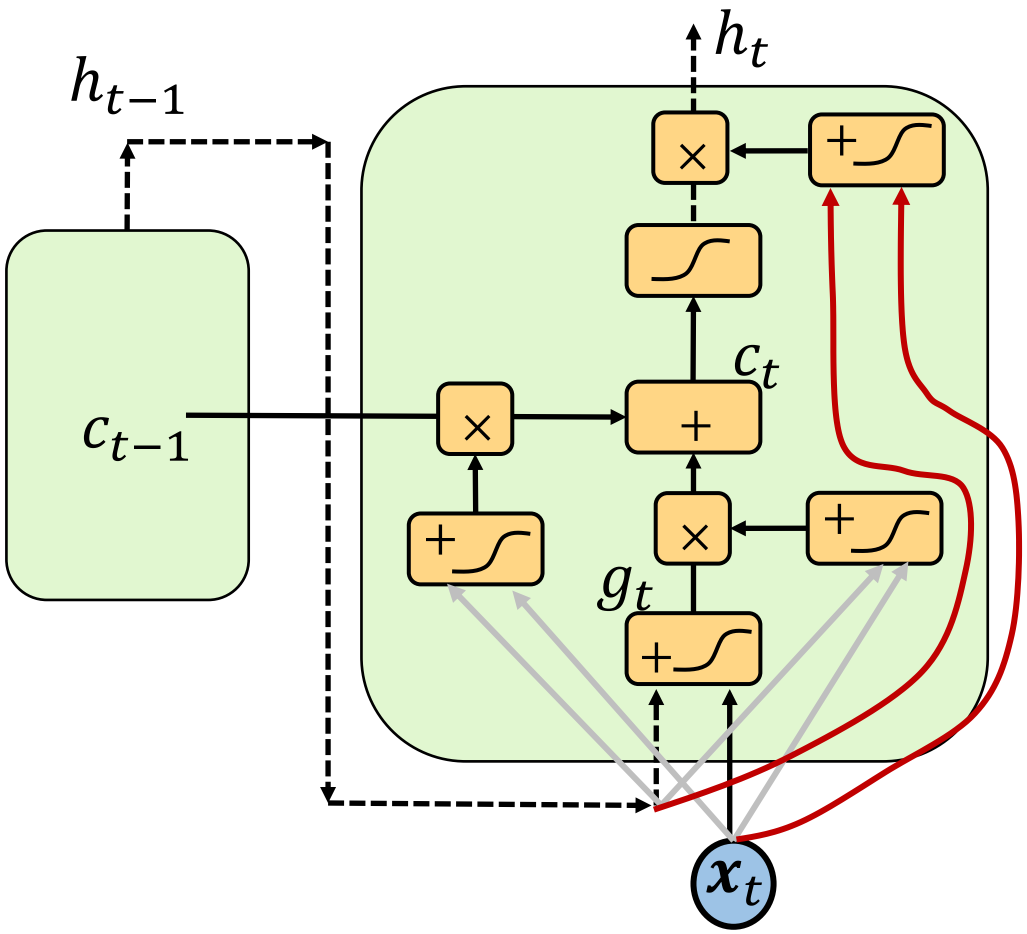
Forget gate

Controls how past internal state c_{t-1} contributes to c_t

$$F_t(x_t, h_{t-1})$$

Logistic sigmoid

LSTM Design – Step 2 (Gates)



Output gate

Controls what part of the internal state is propagated out of the cell

$$O_t(x_t, h_{t-1})$$

Logistic sigmoid

LSTM in Equations

1) Compute activation of input and forget gates

$$\boldsymbol{I}_t = \sigma(\boxed{\boldsymbol{W}_{Ih}}\boldsymbol{h}_{t-1} + \boxed{\boldsymbol{W}_{Iin}}\boldsymbol{x}_t + \boxed{\mathbf{b}_{\mathrm{I}}})$$

$$\boldsymbol{F}_t = \sigma(\boxed{\boldsymbol{W}_{Fh}}\boldsymbol{h}_{t-1} + \boxed{\boldsymbol{W}_{Fin}}\boldsymbol{x}_t + \boxed{\mathbf{b}_{\mathrm{F}}})$$

2) Compute input potential and internal state

$$\boldsymbol{g}_t = tanh(\boxed{\boldsymbol{W}_h}\boldsymbol{h}_{t-1} + \boxed{\boldsymbol{W}_{in}}\boldsymbol{x}_t + \boxed{\mathbf{b}_{\mathrm{h}}})$$

$$\boldsymbol{c}_t = \boldsymbol{F}_t \odot \boldsymbol{c}_{t-1} + \boldsymbol{I}_t \odot \boldsymbol{g}_t$$

3) Compute output gate and output state

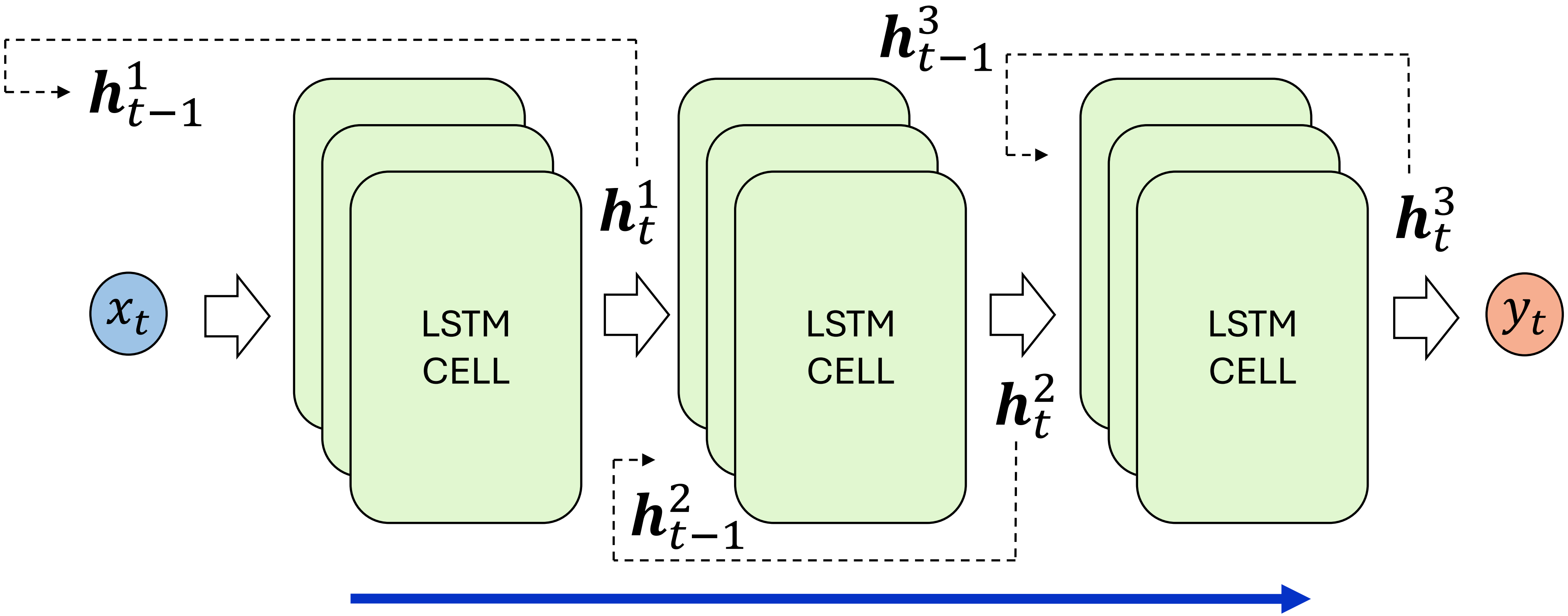
$$\boldsymbol{O}_t = \sigma(\boxed{\boldsymbol{W}_{Oh}}\boldsymbol{h}_{t-1} + \boxed{\boldsymbol{W}_{Oin}}\boldsymbol{x}_t + \boxed{\mathbf{b}_{\mathrm{O}}})$$

$$\boldsymbol{h}_t = \boldsymbol{O}_t \odot tanh(\boldsymbol{c}_t)$$

Training works by BPTT as in vanilla RNNs (including truncation)

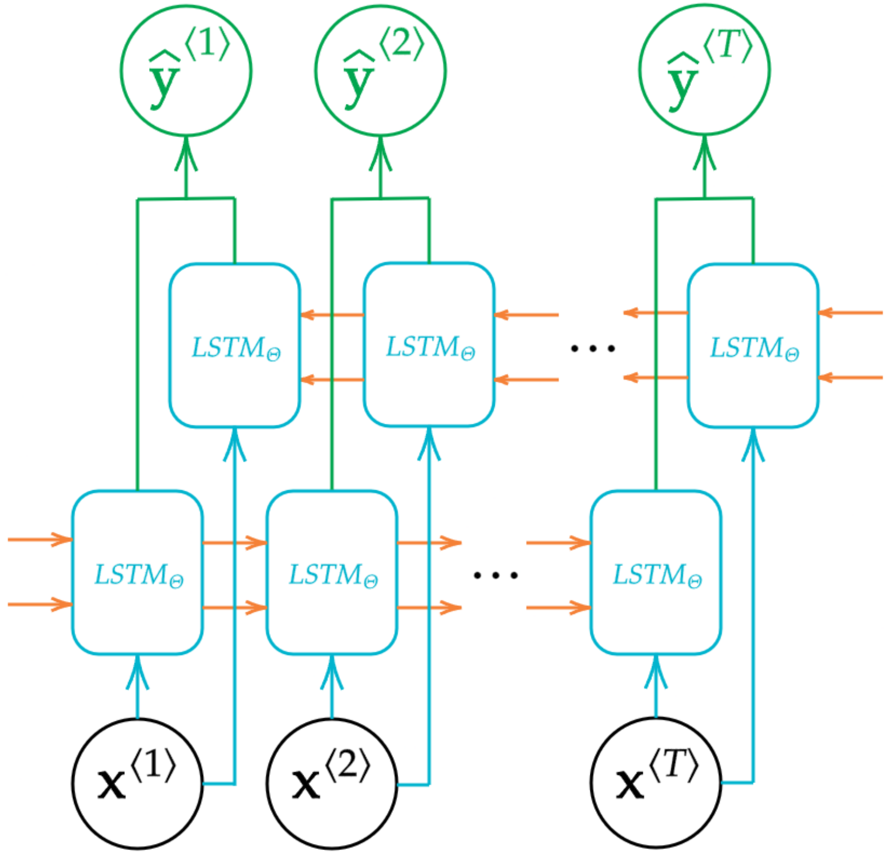
$\odot$ element-wise multiplication

Deep LSTM



LSTM layers extract information at increasing levels of abstraction (enlarging context)

Bidirectional LSTM (BiLSTM)



- We combine (sum, average) the two directions before the output
- This is more powerful, as it takes the entire sequence into account to make a prediction
- But the entire sequence must be available (which is not always possible)

Gated Recurrent Unit (GRU)

Reset acts directly on output state (no internal state and no output gate)

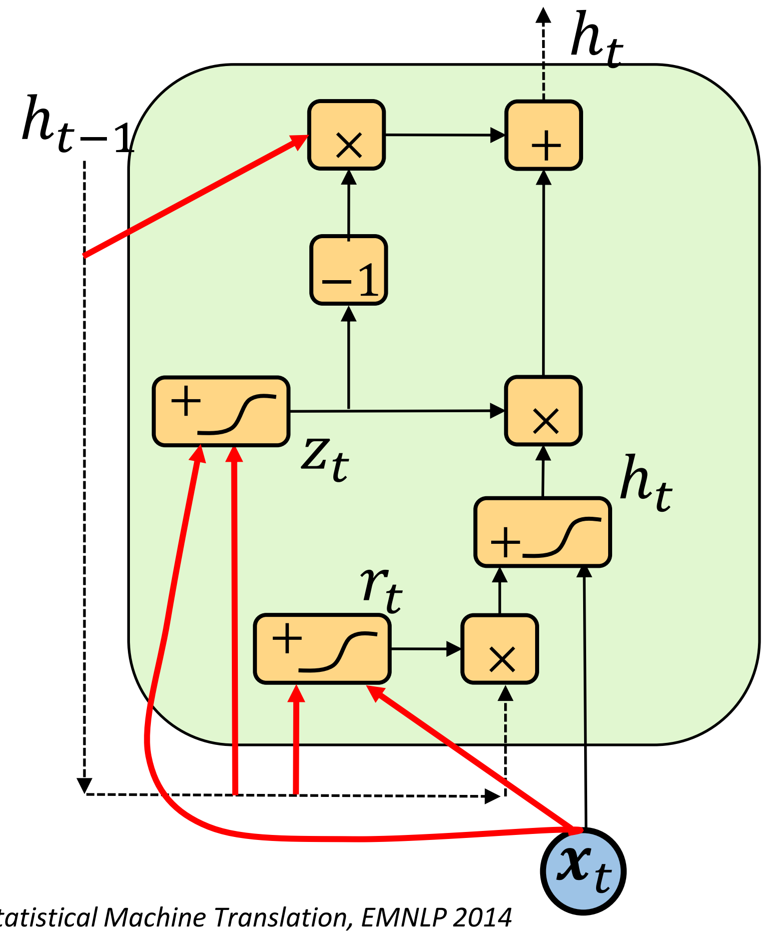
$$\boldsymbol{h}_t = (1 - \boldsymbol{z}_t) \odot \boldsymbol{h}_{t-1} + \boldsymbol{z}_t \odot \boldsymbol{h}_t$$

$$\boldsymbol{h}_t = tanh(\boldsymbol{W}_{hh}(\boldsymbol{r}_t \odot \boldsymbol{h}_{t-1}) + \boldsymbol{W}_{hin}\boldsymbol{x}_t + \mathbf{b}_\mathrm{h})$$

Reset and update gates when coupled act as input and forget gates

$$\boldsymbol{z}_t = \sigma(\boldsymbol{W}_{zh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{zin}\boldsymbol{x}_t + \mathbf{b}_\mathrm{z})$$

$$\boldsymbol{r}_t = \sigma(\boldsymbol{W}_{rh}\boldsymbol{h}_{t-1} + \boldsymbol{W}_{rin}\boldsymbol{x}_t + \mathbf{b}_\mathrm{r})$$

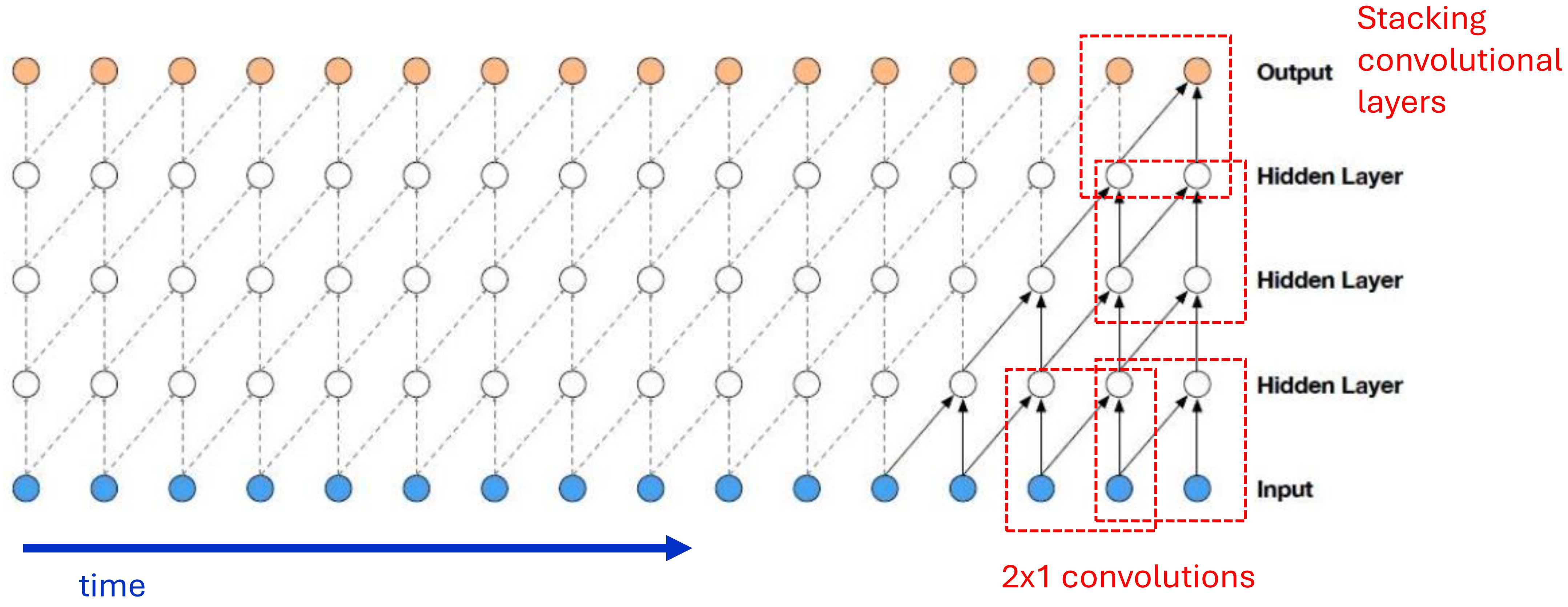


C. Kyunghyun et al, Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation, EMNLP 2014

Convolutional Recurrent Networks

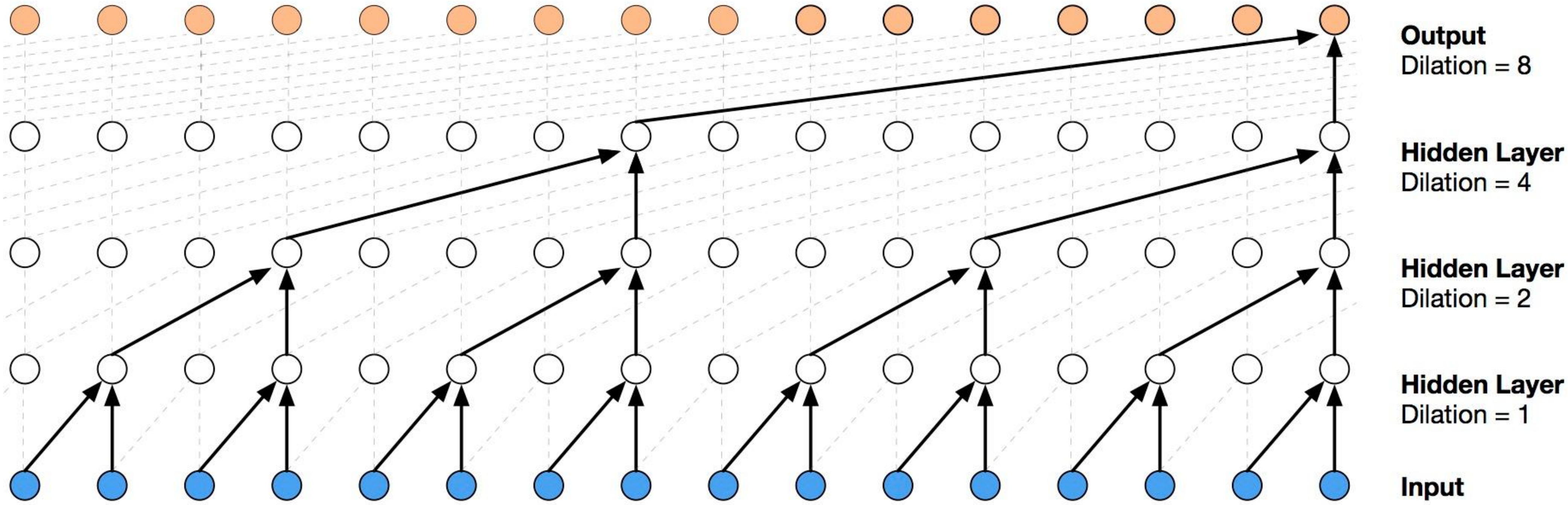
Convolutional neural networks on timeseries

It should not be surprising to think that convolutional filters can be defined to be mono-dimensional for their use on timeseries

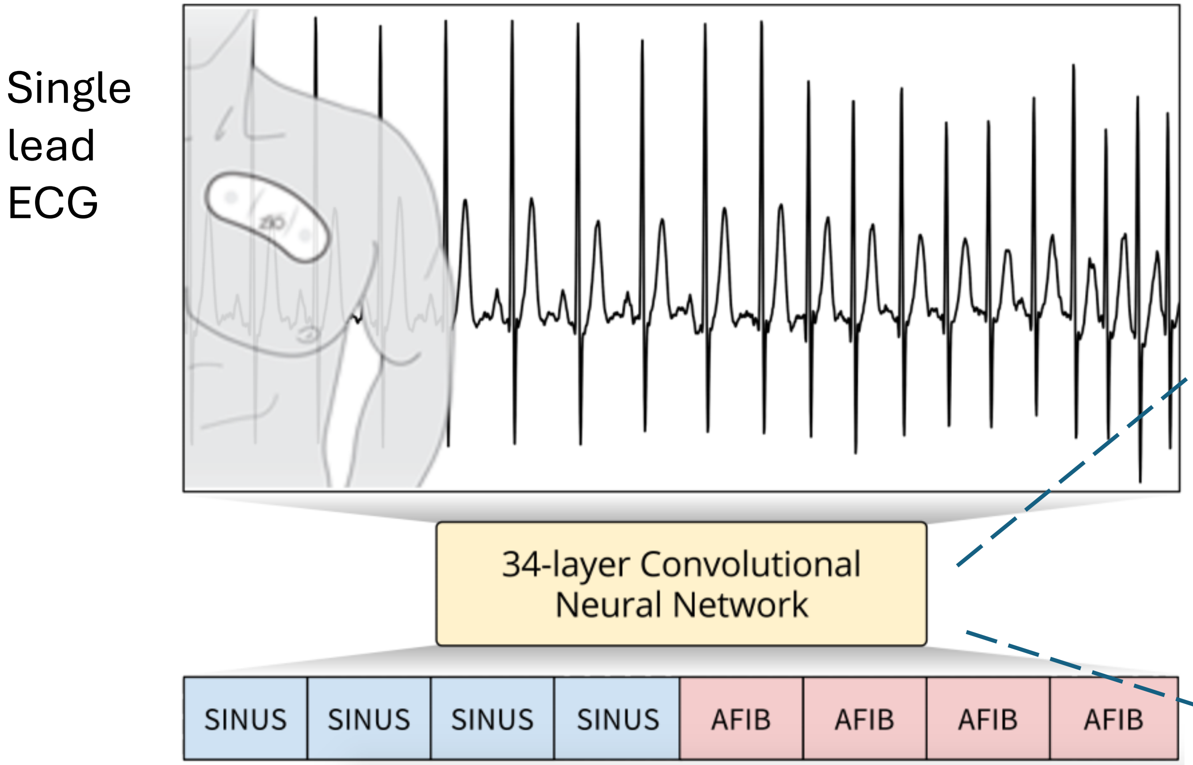


Temporal Convolutional Networks (TCNs)

The return of dilated convolutions

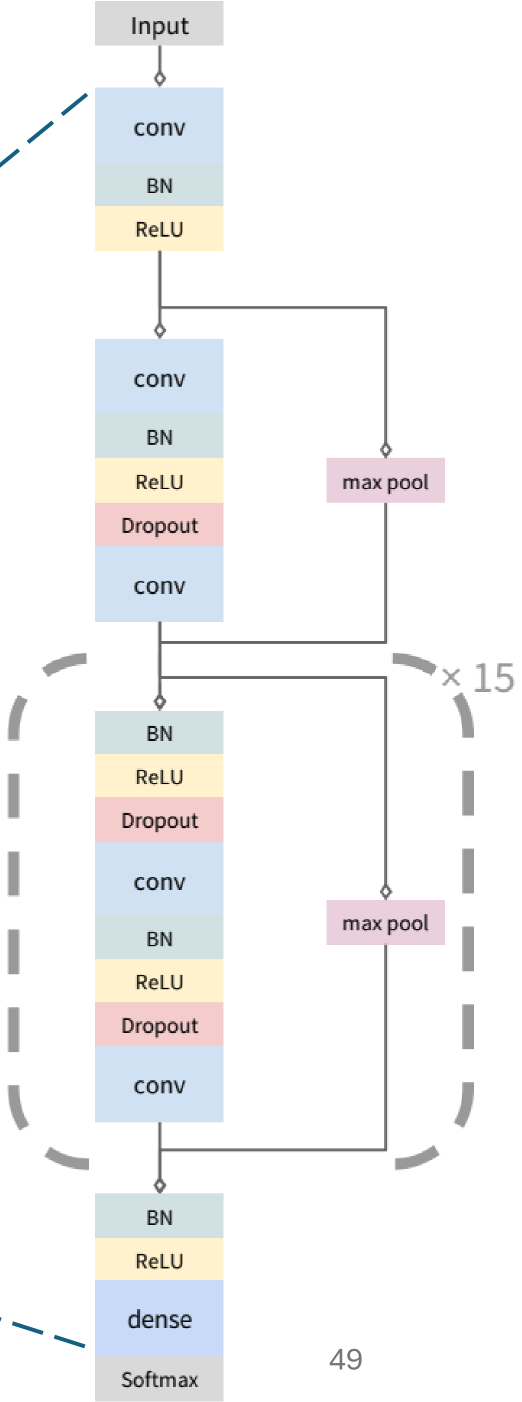


You can reuse all your CNN building blocks and knowledge here

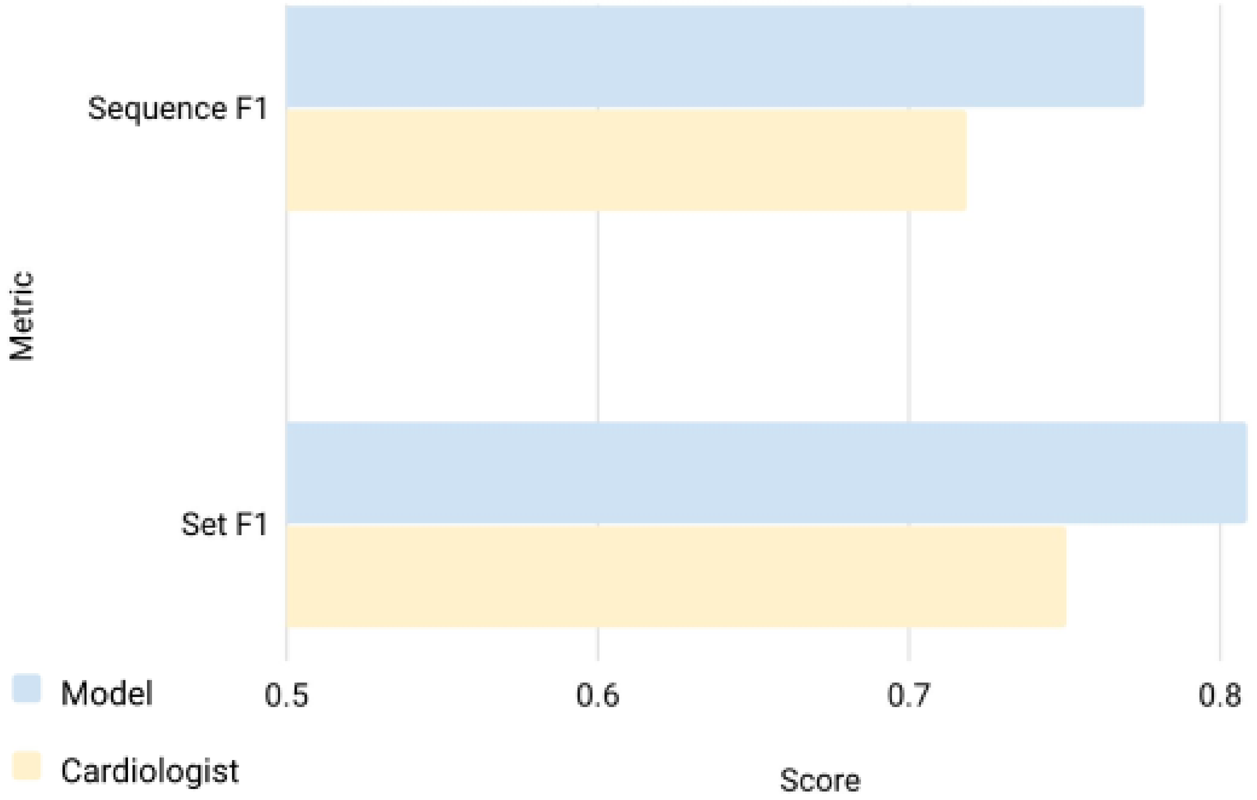


Predicting Sinus rhythm (SINUS) Vs Atrial Fibrillation for a total of 14 classes

Source: [Arxiv]



To get you cardiologist level predictions



	Seq		Set	
	Model	Cardiol.	Model	Cardiol.
Class-level F1 Score				
AFIB	**0.604**	0.515	**0.667**	0.544
AFL	**0.687**	0.635	**0.679**	0.646
AVB_TYPE2	**0.689**	0.535	**0.656**	0.529
BIGEMINY	**0.897**	0.837	**0.870**	0.849
CHB	**0.843**	0.701	**0.852**	0.685
EAR	**0.519**	0.476	**0.571**	0.529
IVR	**0.761**	0.632	**0.774**	0.720
JUNCTIONAL	0.670	**0.684**	**0.783**	0.674
NOISE	**0.823**	0.768	**0.704**	0.689
SINUS	**0.879**	0.847	**0.939**	0.907
SVT	**0.477**	0.449	**0.658**	0.556
TRIGEMINY	**0.908**	0.843	**0.870**	0.816
VT	0.506	**0.566**	0.694	**0.769**
WENCKEBACH	**0.709**	0.593	**0.806**	0.736
Aggregate Results				
Precision (PPV)	**0.800**	0.723	**0.809**	0.763
Recall (Sensitivity)	**0.784**	0.724	**0.827**	0.744
F1	**0.776**	0.719	**0.809**	0.751

Source: [Arxiv]

Healthcare Applications

LSTM for clinical timeseries (and risk prediction)

- LSTMs vs logistic regression in predictive tasks on MIMIC-III
- Used a subset of 17 clinical variables in input
  - All required some imputation

Four predictive tasks:

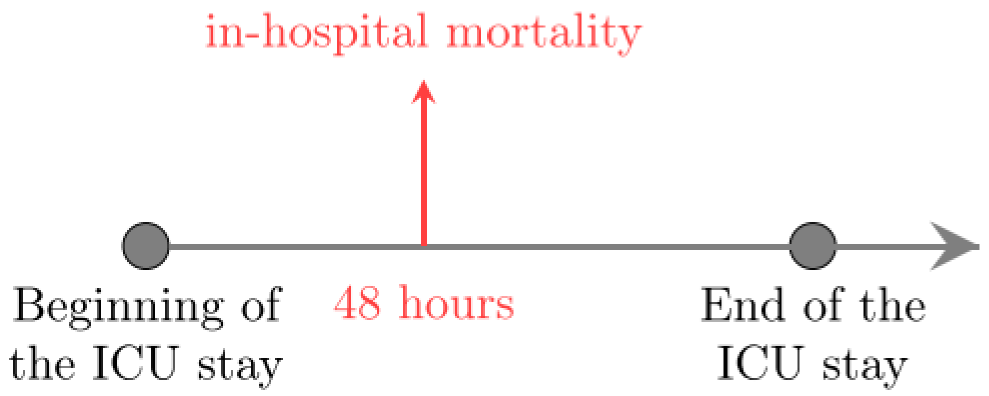
- in-hospital mortality
- decompensation
- length-of-stay
- Phenotype classification

Variable	MIMIC-III table	Impute value	Modeled as
Capillary refill rate	chartevents	0.0	categorical
Diastolic blood pressure	chartevents	59.0	continuous
Fraction inspired oxygen	chartevents	0.21	continuous
Glascow coma scale eye opening	chartevents	4 spontaneously	categorical
Glascow coma scale motor response	chartevents	6 obeys commands	categorical
Glascow coma scale total	chartevents	15	categorical
Glascow coma scale verbal response	chartevents	5 oriented	categorical
Glucose	chartevents, labevents	128.0	continuous
Heart Rate	chartevents	86	continuous
Height	chartevents	170.0	continuous
Mean blood pressure	chartevents	77.0	continuous
Oxygen saturation	chartevents, labevents	98.0	continuous
Respiratory rate	chartevents	19	continuous
Systolic blood pressure	chartevents	118.0	continuous
Temperature	chartevents	36.6	continuous
Weight	chartevents	81.0	continuous
pH	chartevents, labevents	7.4	continuous

Harutyunyan et al, Nature Sci. Data 2019

In-hospital mortality task

	Train	Test
Negative	15480	2862
Positive	2423	374

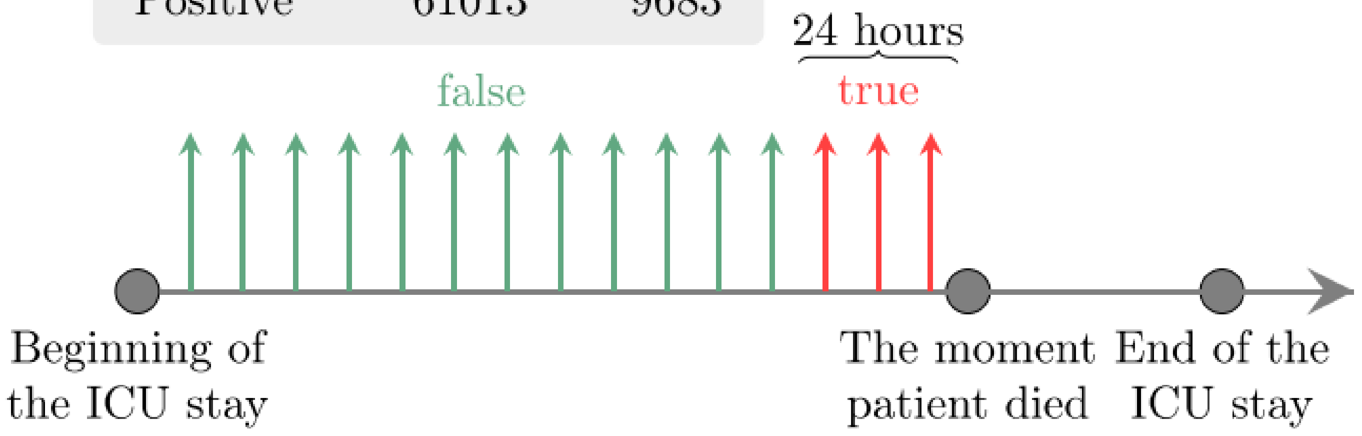


- Predicting in-hospital mortality based on the first 48 hours of an ICU stay
- Binary classification task
- AUC-ROC as metric

Harutyunyan et al, Nature Sci. Data 2019

Decompensation prediction task

	Train	Test
Negative	2847401	513525
Positive	61013	9683

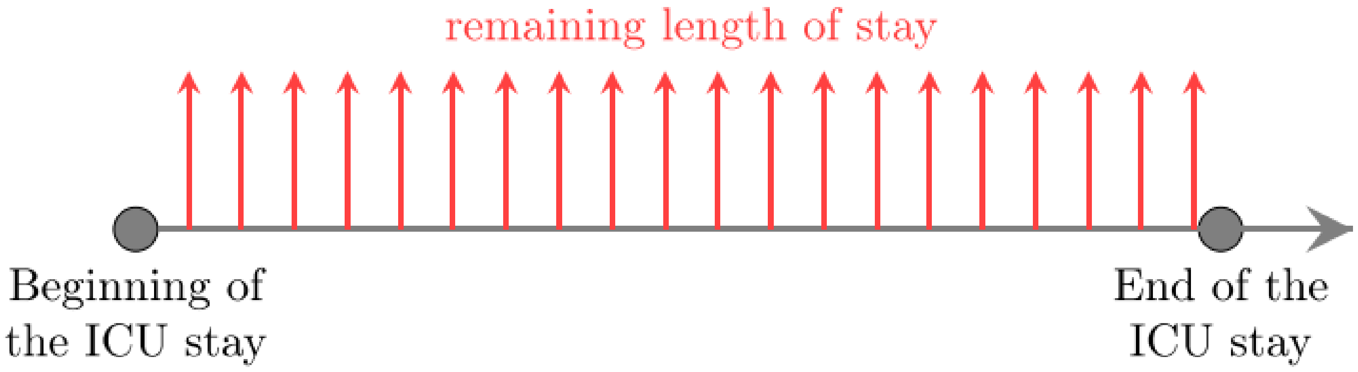


- Decompensation prediction (as mortality in the next 24hours)
- Multiple binary classification task
- AUC-ROC as metric

Harutyunyan et al, Nature Sci. Data 2019

Length-of-stay prediction task

Train	Test
2925434	525912

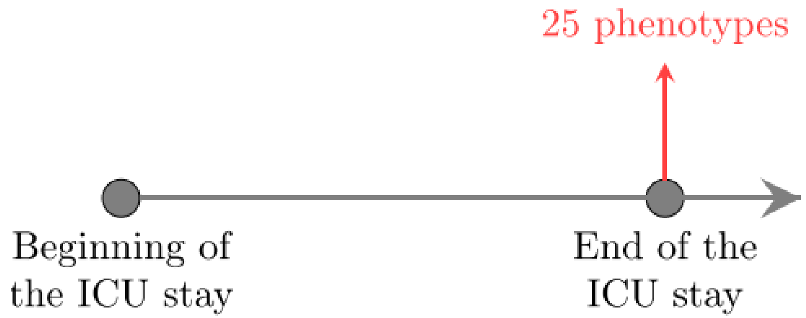


- Remaining time spent in ICU at each hour of stay
- Multiclass task on 10 classes (one for ICU stays shorter than a day, 7 day-long buckets for each day of the first week, one for stays of over one week but less than two, and one for stays of over two weeks)
- Cohen’s linear weighted kappa score

Harutyunyan et al, Nature Sci. Data 2019

Phenotype classification task

Train	Test
35621	6281



- Classifying which of 25 acute care conditions are present in each patient ICU stay record
- Multilabel classification problem
- Average AUC-ROC

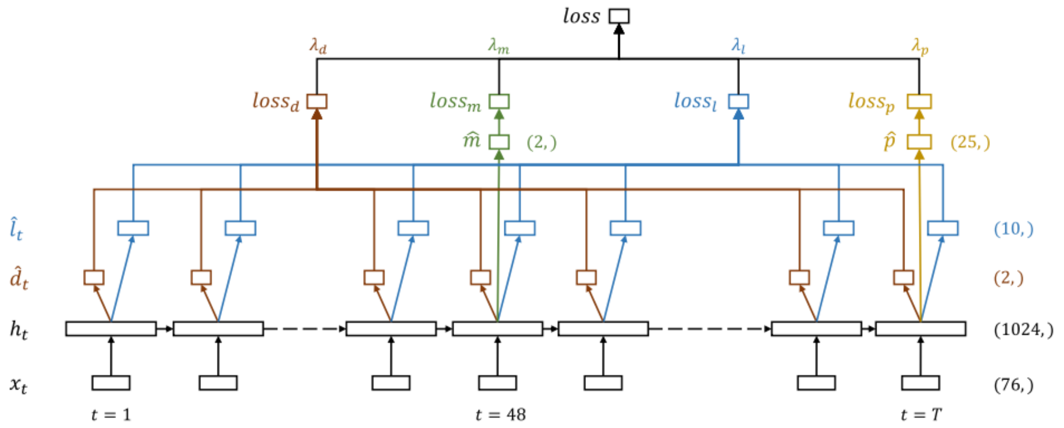
Harutyunyan et al, Nature Sci. Data 2019

Phenotype	Type	Prevalence		AUC-ROC
		Train	Test	
Acute and unspecified renal failure	acute	0.214	0.212	0.806
Acute cerebrovascular disease	acute	0.075	0.066	0.909
Acute myocardial infarction	acute	0.103	0.108	0.776
Cardiac dysrhythmias	mixed	0.321	0.323	0.687
Chronic kidney disease	chronic	0.134	0.132	0.771
Chronic obstructive pulmonary disease	chronic	0.131	0.126	0.695
Complications of surgical/medical care	acute	0.207	0.213	0.724
Conduction disorders	mixed	0.072	0.071	0.737
Congestive heart failure; nonhypertensive	mixed	0.268	0.268	0.763
Coronary atherosclerosis and related	chronic	0.322	0.331	0.797
Diabetes mellitus with complications	mixed	0.095	0.094	0.872
Diabetes mellitus without complication	chronic	0.193	0.192	0.797
Disorders of lipid metabolism	chronic	0.291	0.289	0.728
Essential hypertension	chronic	0.419	0.423	0.683
Fluid and electrolyte disorders	acute	0.269	0.265	0.739
Gastrointestinal hemorrhage	acute	0.072	0.079	0.751
Hypertension with complications	chronic	0.133	0.130	0.750
Other liver diseases	mixed	0.089	0.089	0.778
Other lower respiratory disease	acute	0.051	0.057	0.694
Other upper respiratory disease	acute	0.040	0.043	0.785
Pleurisy; pneumothorax; pulmonary collapse	acute	0.087	0.091	0.709
Pneumonia	acute	0.139	0.135	0.809
Respiratory failure; insufficiency; arrest	acute	0.181	0.177	0.907
Septicemia (except in labor)	acute	0.143	0.139	0.854
Shock	acute	0.078	0.082	0.892

Some interesting insights (and tricks)

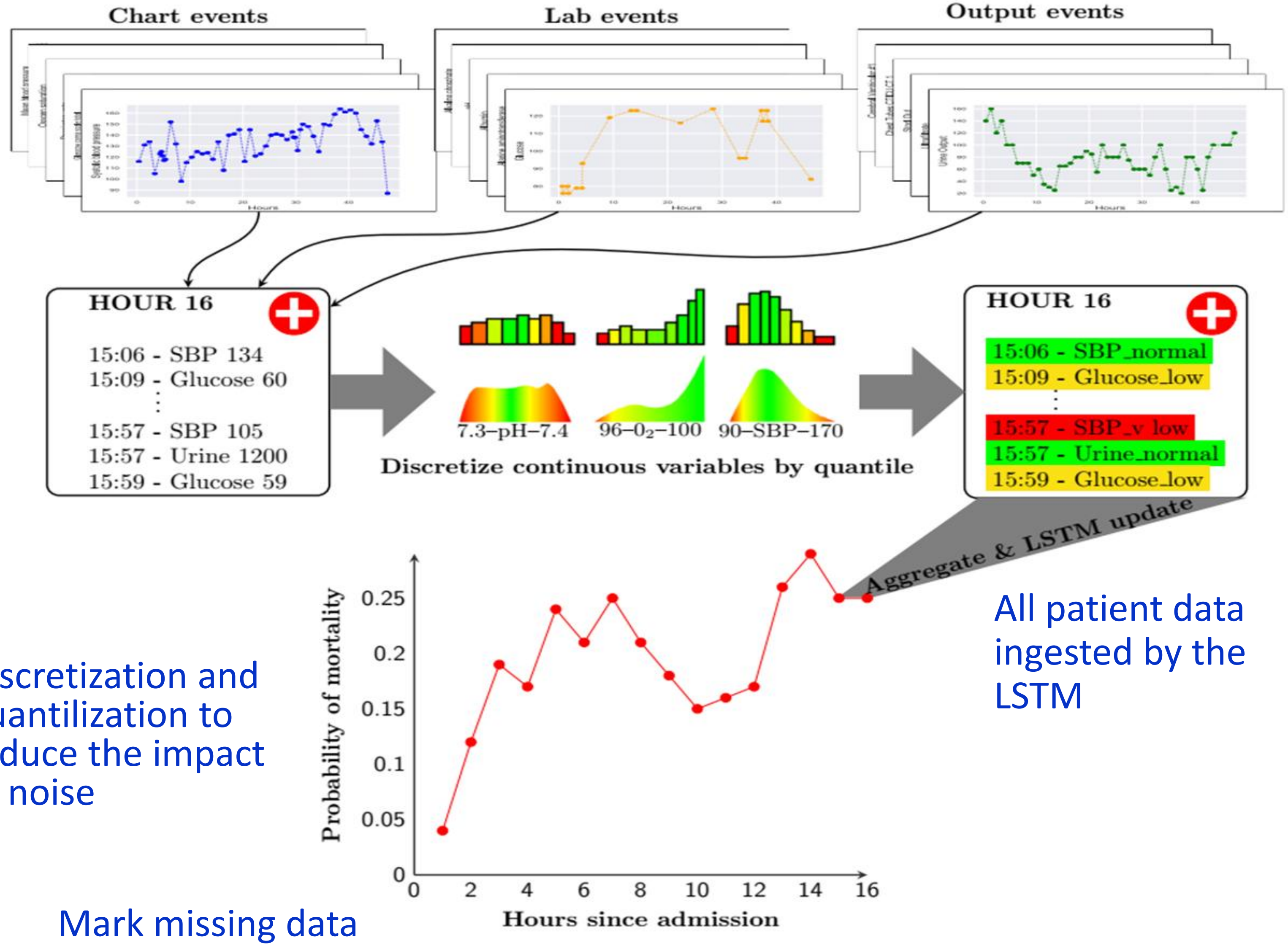
- Working with multi-channel data requires a lot of alignment of timesteps (subsampling, supersampling and imputation)
  - Authors also experimented with channel specific (Bi)LSTMs (one for each of the 17 channels)
- Pair each channel with a binary variable (in time) indicating whether the specific channel was observed at time step t

- Multitask learning can help: training the LSTM to solve all four task altogether rather than independently
- Deep supervision helps on sequence prediction tasks
  - Use target replication for each time step if it makes sense
  - Doesn't work for length-of-stay and decompensation



A “less data curation” approach

Survival prediction in ICU from MIMIC-III data



Discretization and quantilization to reduce the impact of noise

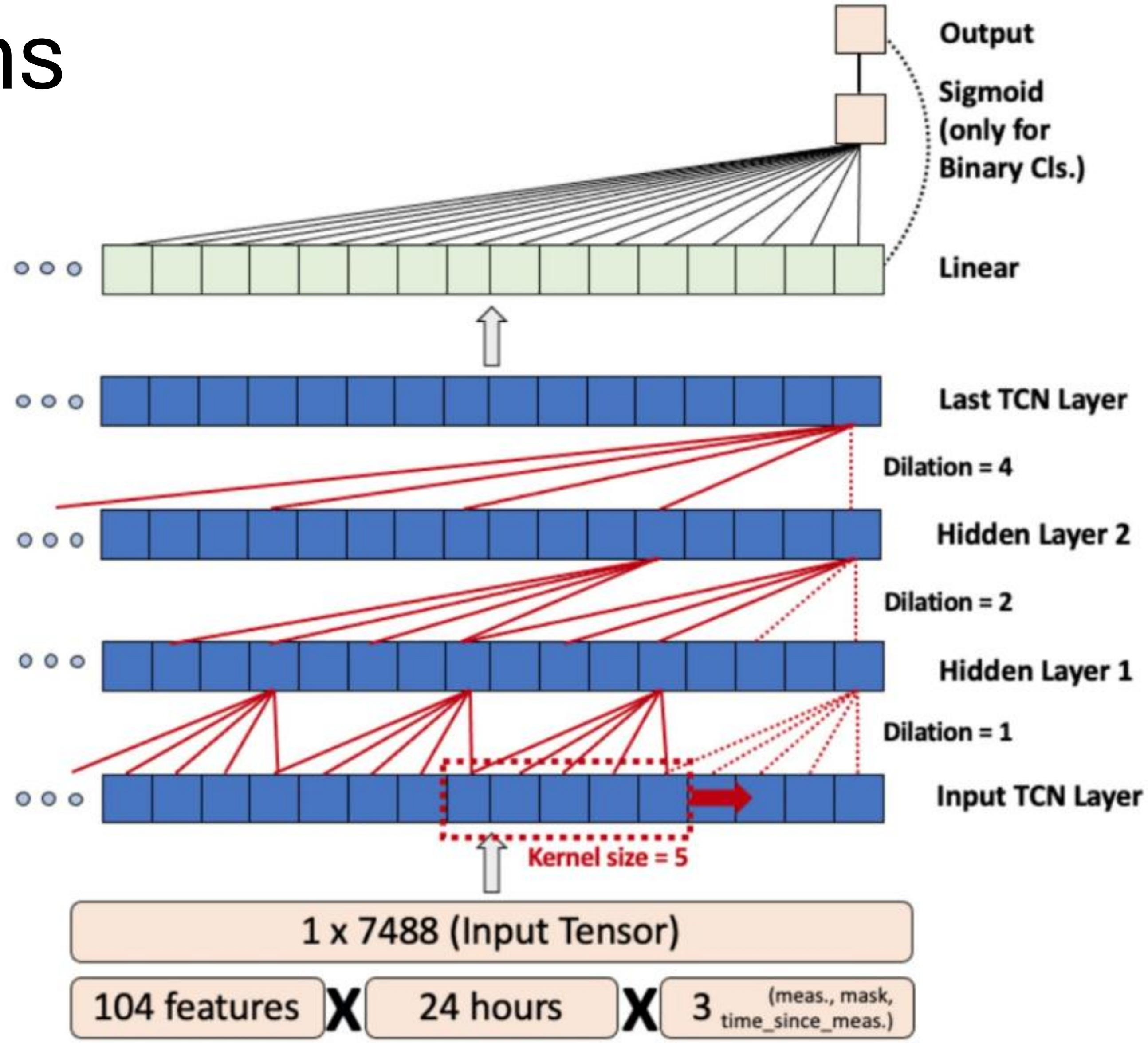
Mark missing data and do not impute

All patient data ingested by the LSTM

Deasy et al, Scientific Report, 2020

Temporal convolutions on MIMIC-III tasks

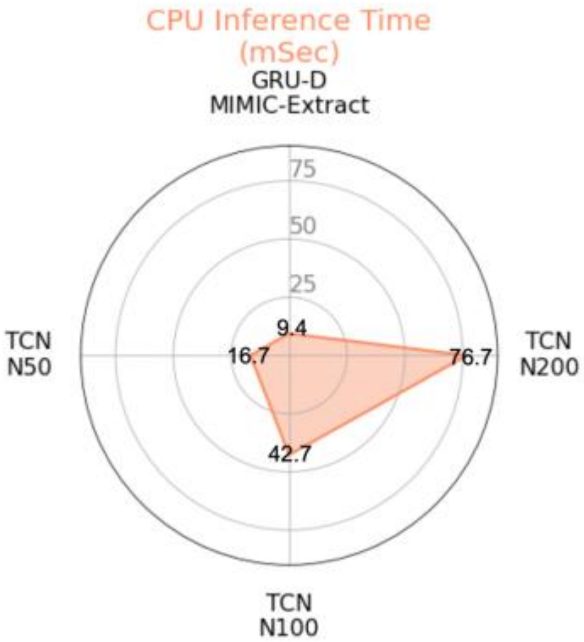
Thanks to the dilation factor can gain a longer-time insight into the history of the input signal than Gated RNNs, without incurring in fading gradients

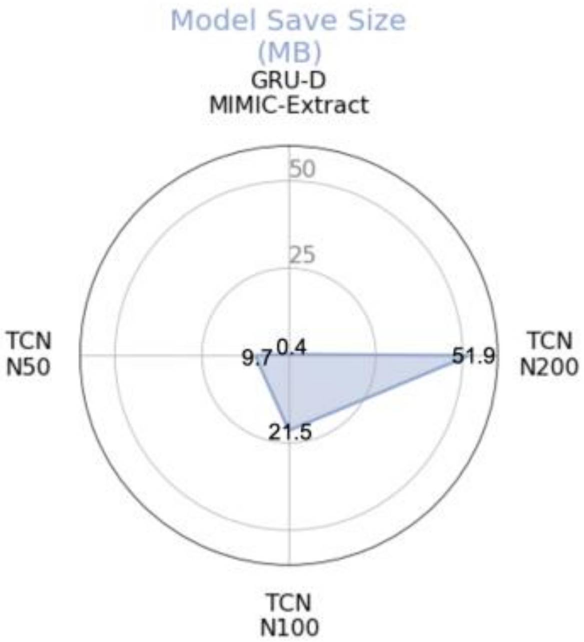


Bednarsky et al, Scientific Report, 2022

TCN - Good cost-for-performance trade-off

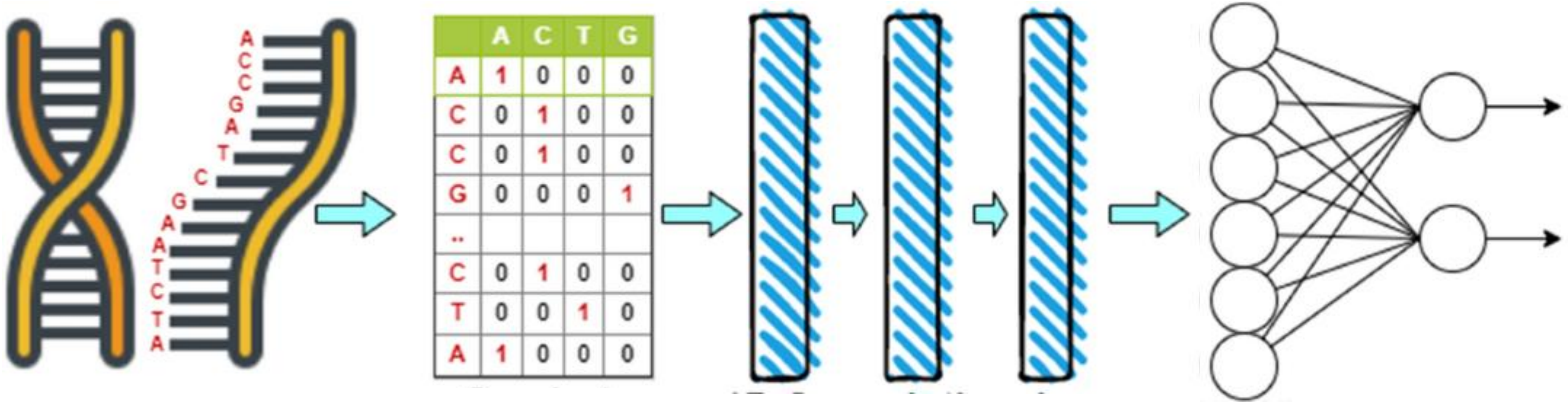
Model	AUROC	AUPRC	Accuracy	F-1	Precision	Recall
In-ICU mortality						
LR	85.1 ± 3.2	39.5 ± 7.2	93.4 ± 0.6	30.1 ± 7.6	55.0 ± 11.6	20.7 ± 6.1
RF	89.1 ± 2.2	45.9 ± 7.3	93.5 ± 0.3	14.2 ± 6.5	81.8 ± 19.2	7.8 ± 3.9
GRU-D	**89.4 ± 2.3**	**50.8 ± 6.8**	94.0 ± 0.6	38.9 ± 8.1	66.2 ± 10.3	27.6 ± 6.5
TCN	89.2 ± 2.5	**50.8 ± 7.0**	**94.3 ± 0.6**	**46.6 ± 7.3**	64.5 ± 8.7	36.5 ± 7.1
In-hospital mortality						
LR	83.6 ± 2.6	44.7 ± 5.7	91.0 ± 0.7	35.7 ± 6.0	61.4 ± 9.3	25.2 ± 5.3
RF	86.4 ± 2.3	49.3 ± 5.9	90.7 ± 0.4	14.5 ± 5.8	85.1 ± 14.0	7.9 ± 3.4
GRU-D	87.3 ± 2.3	52.1 ± 5.6	**91.6 ± 0.8**	44.2 ± 6.0	65.4 ± 7.5	33.4 ± 5.8
TCN	**87.7 ± 2.1**	**53.0 ± 6.0**	91.2 ± 0.9	**47.2 ± 6.0**	58.7 ± 6.7	39.5 ± 6.2
Length of stay (LOS > 3)						
LR	69.0 ± 2.1	61.7 ± 2.8	65.5 ± 1.8	53.5 ± 2.7	63.6 ± 2.8	46.2 ± 2.9
RF	71.4 ± 2.0	65.5 ± 2.8	67.3 ± 1.7	55.3 ± 2.7	67.1 ± 2.8	47.0 ± 3.0
GRU-D	**72.2 ± 2.0**	**65.7 ± 2.7**	**68.1 ± 1.7**	**59.4 ± 2.5**	65.6 ± 2.6	54.2 ± 3.0
TCN	71.6 ± 2.2	65.0 ± 2.7	67.0 ± 1.7	55.6 ± 2.7	66.0 ± 2.8	48.0 ± 2.9
Length of stay (LOS > 7)						
LR	66.8 ± 4.2	15.9 ± 3.3	91.7 ± 0.3	2.3 ± 2.8	15.2 ± 17.7	1.3 ± 1.6
RF	**75.3 ± 3.5**	22.0 ± 4.5	**92.1 ± 0.0**	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
GRU-D	74.4 ± 3.8	**22.4 ± 4.5**	92.0 ± 0.4	**9.8 ± 5.3**	44.9 ± 20.4	5.5 ± 3.2
TCN	73.5 ± 3.6	18.8 ± 3.5	91.8 ± 0.3	3.7 ± 3.5	25.0 ± 21.9	2.0 ± 1.9





Bednarsky et al, Scientific Report, 2022

Working with Genomic Sequences



Sequenced genetic material

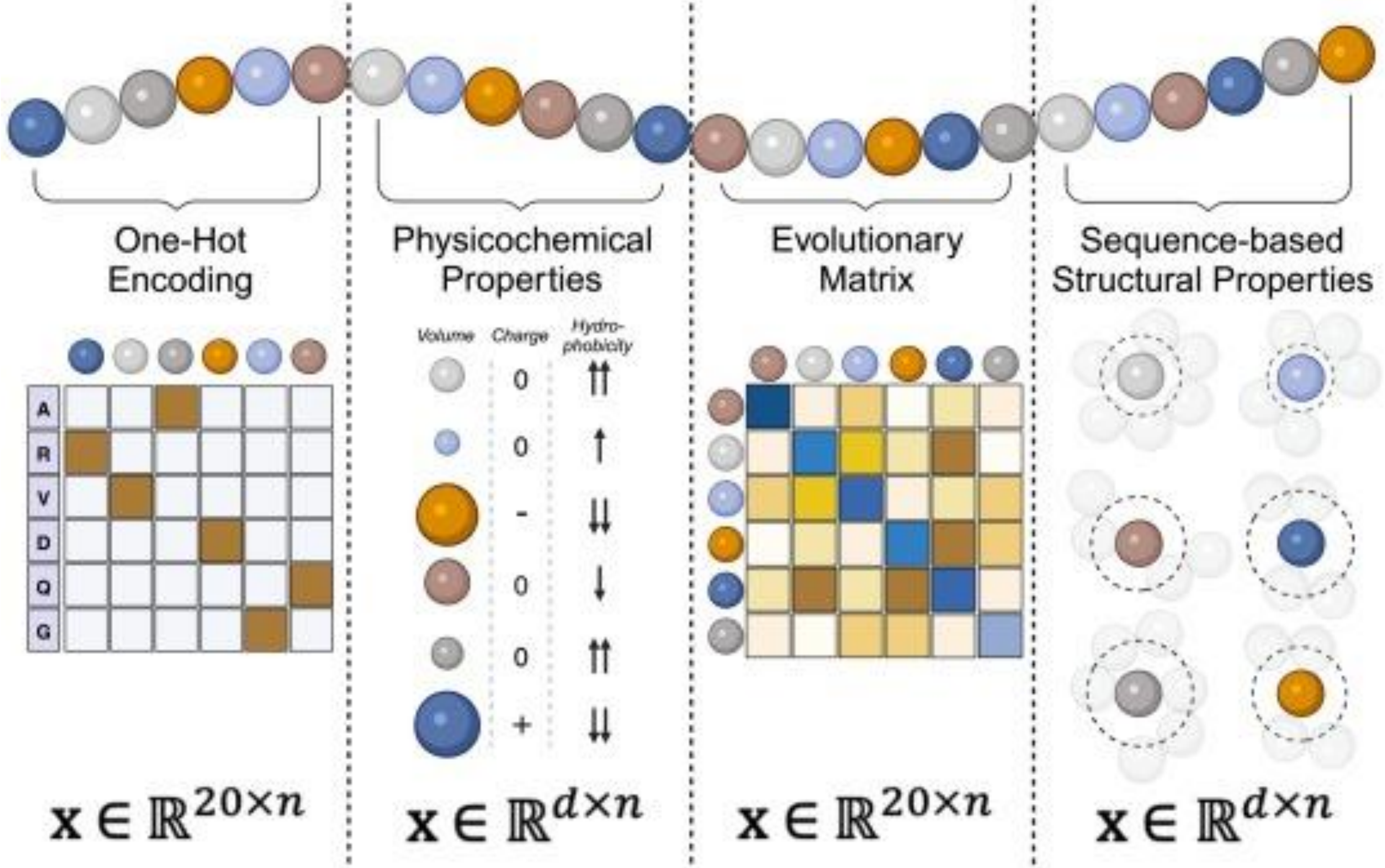
one-hot encoding

neural layers for sequences

prediction layer

The approach can be extended also to protein

Source: D. Harding-Larsen et al, Biotechnology Advances, 2024

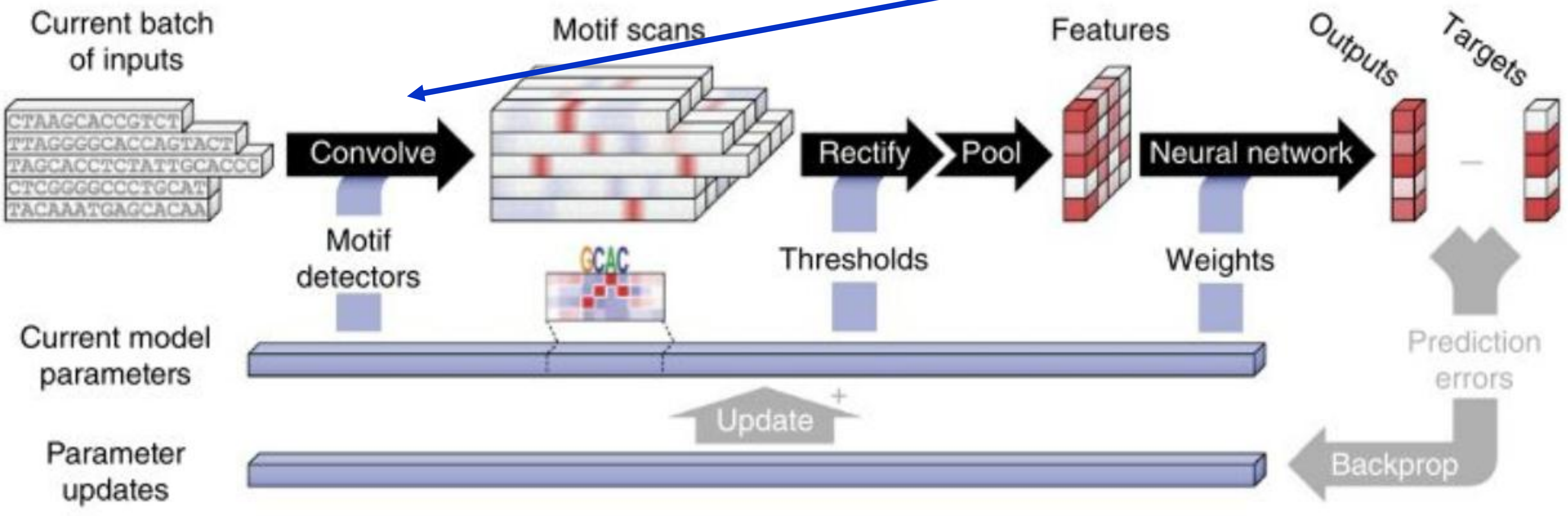


Although the vocabulary grows and many different representations can be thought of (including graph-based ones)

DeepBind

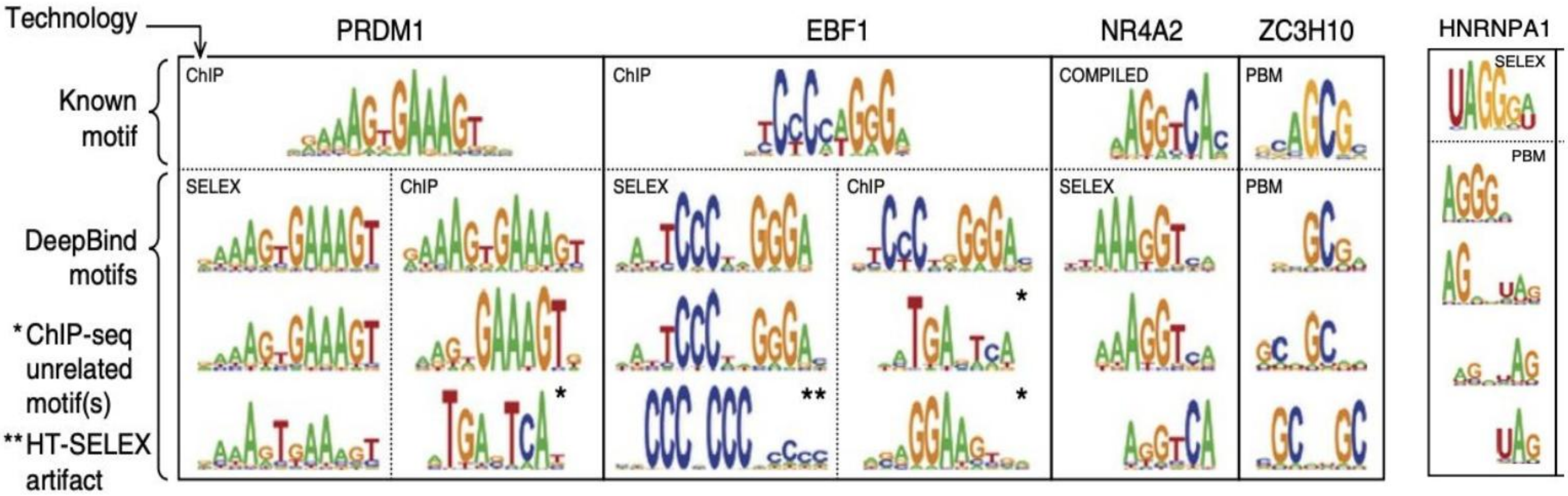
Predicting scores of whether particular proteins will bind to the sequence or not

Convolutional neural network for sequences



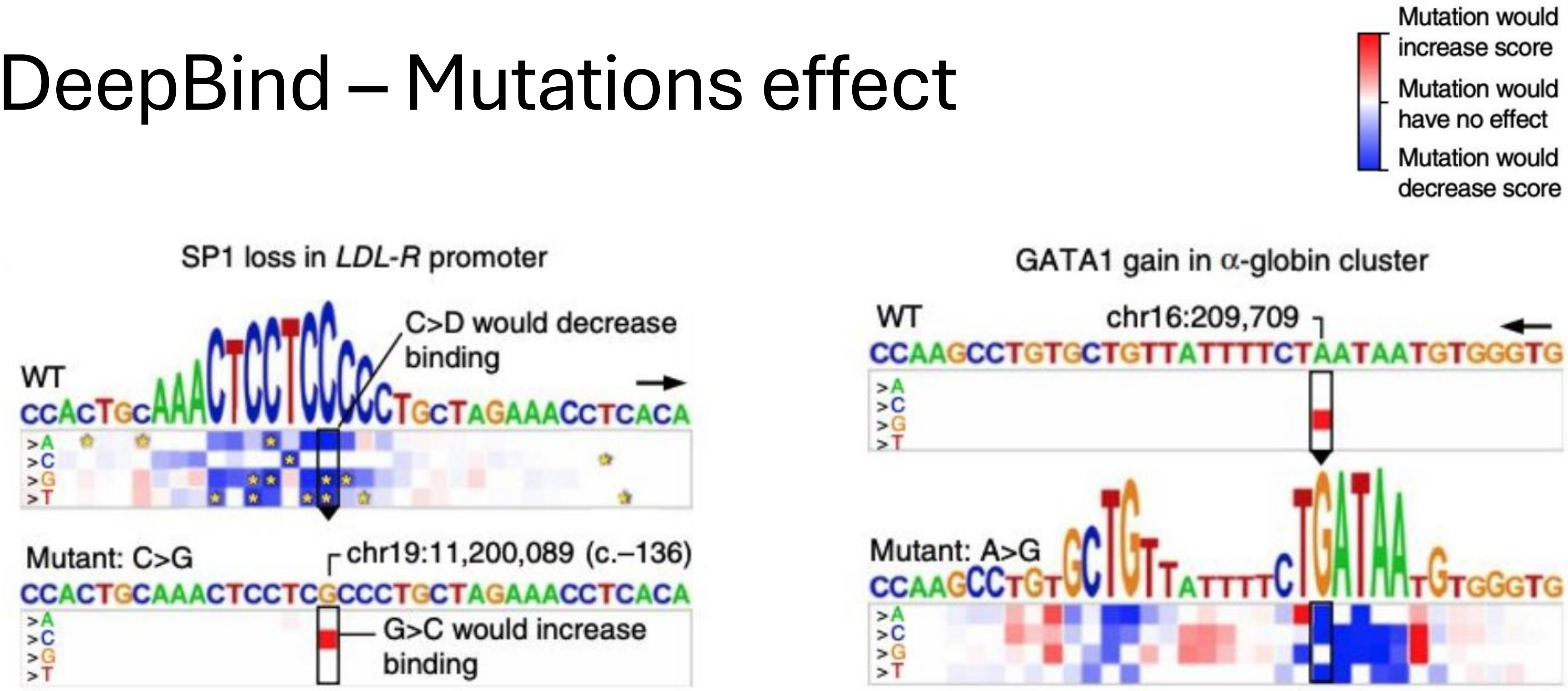
Alipanahi et al, Nature 2015

DeepBind – Intepreting filters



Alipanahi et al, Nature 2015

DeepBind – Mutations effect



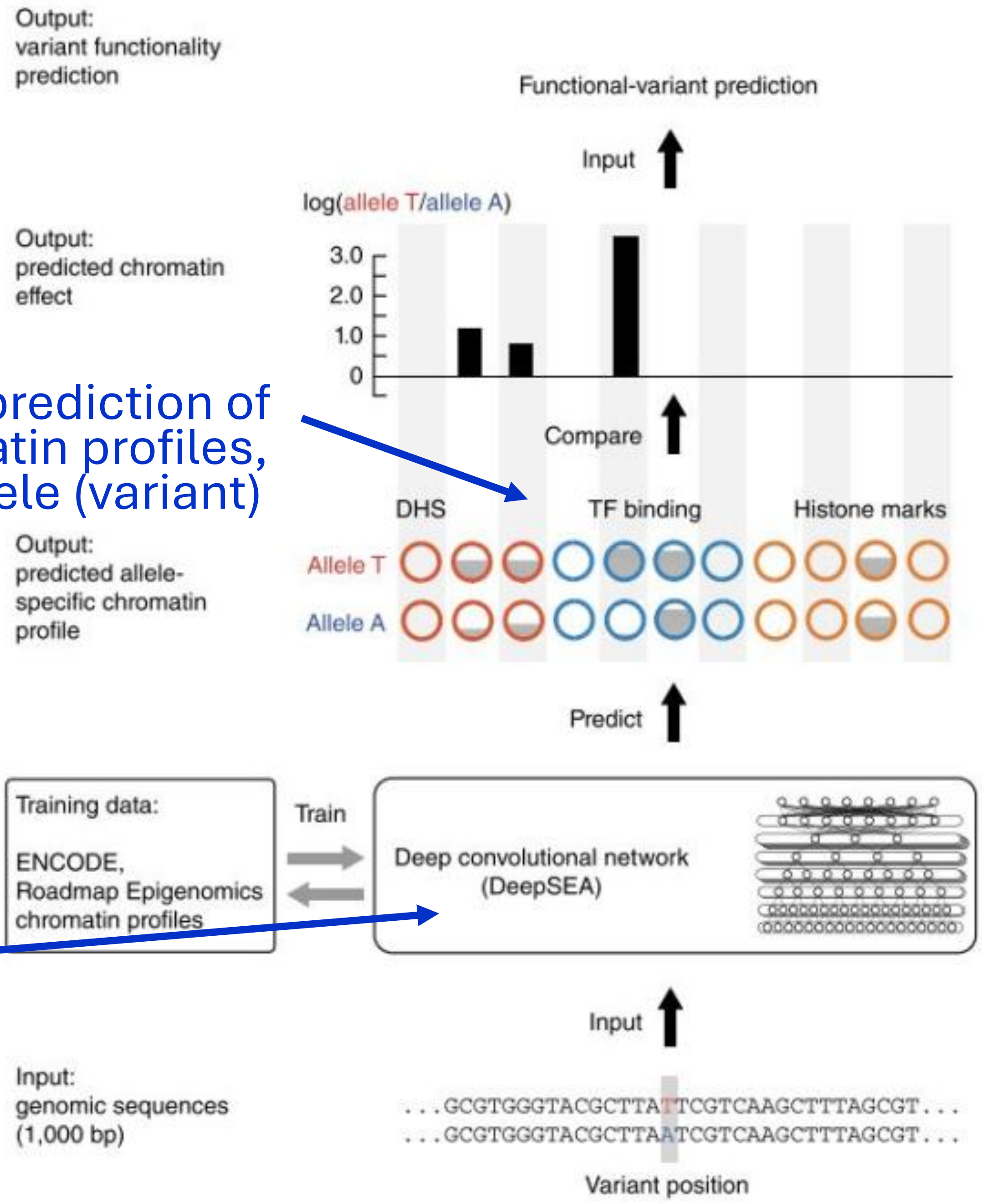
Predict the effect of sequence mutation through interpretability techniques

Alipanahi et al, Nature 2015

DeepSea

Predict chromatin effects of (non-coding) sequence alterations with single-nucleotide polimorphisms (SNPs)

Multi-task prediction of 919 chromatin profiles, for each allele (variant)

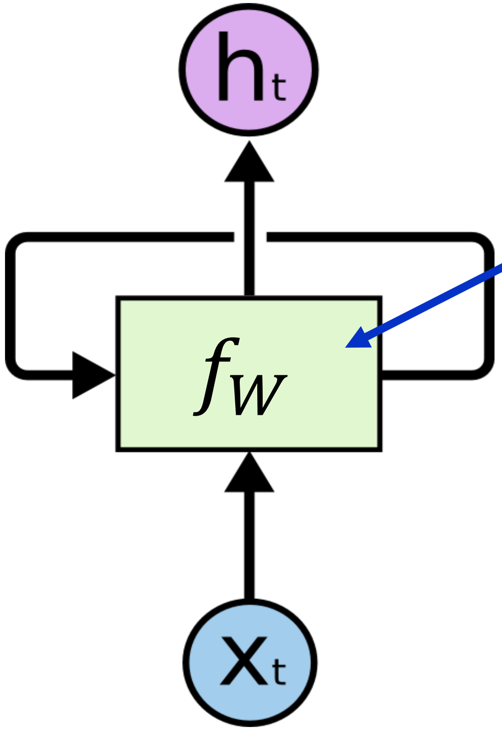


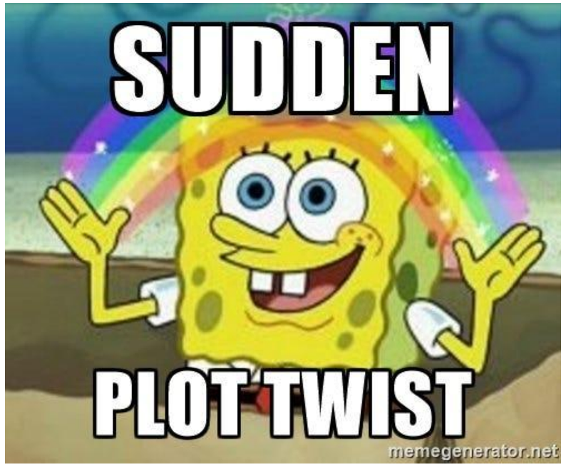
Convolutional (8x1) and pooling (4x1) layers

Zhou and Troyanskaya, Nature 2015

Bonus Track

What if we don’t train this?

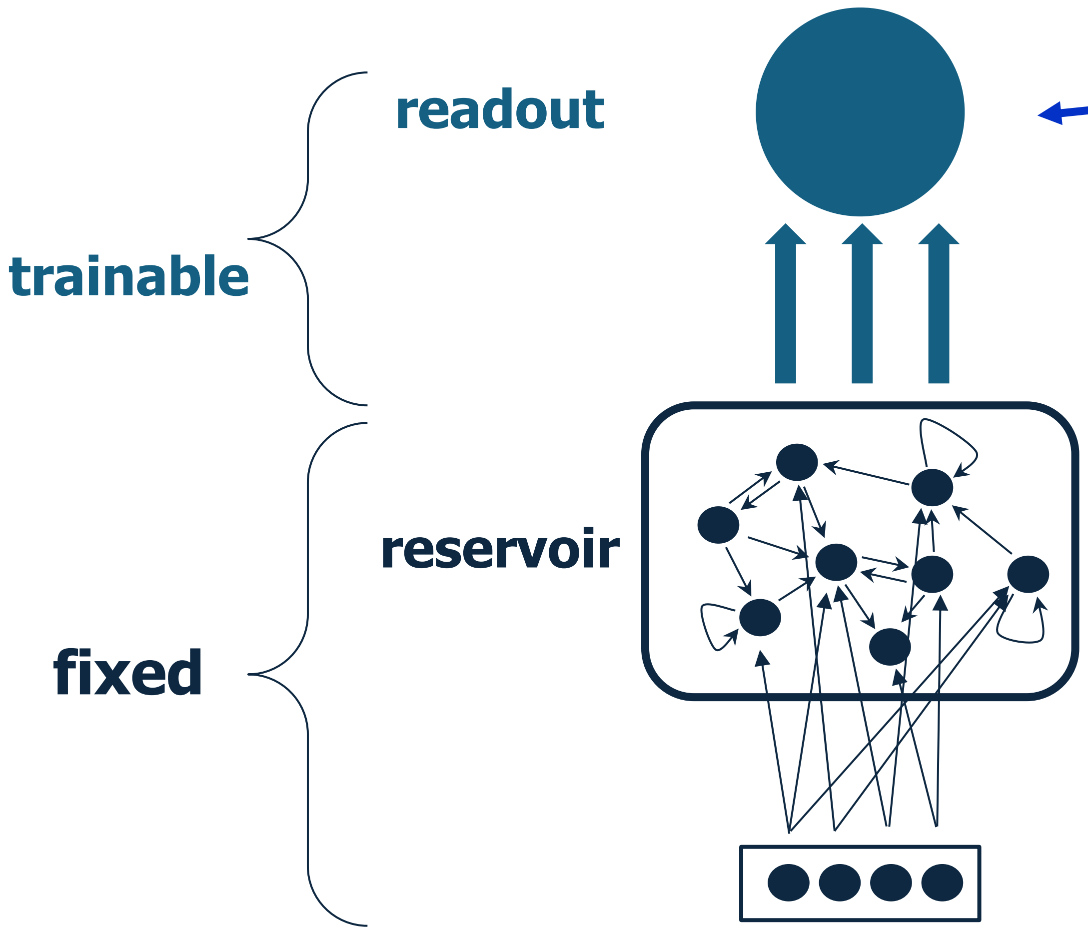




$$\boldsymbol{h}_t = \tanh(\boldsymbol{W}_h \boldsymbol{h}_{t-1} + \boldsymbol{W}_{in} \boldsymbol{x}_t)$$

It means that these are initialized but not trained!

You get Reservoir Computing



Simple linear layer trainable in closed from (remember least-mean square!)

High dimensional but sparsely connected recurrent layer

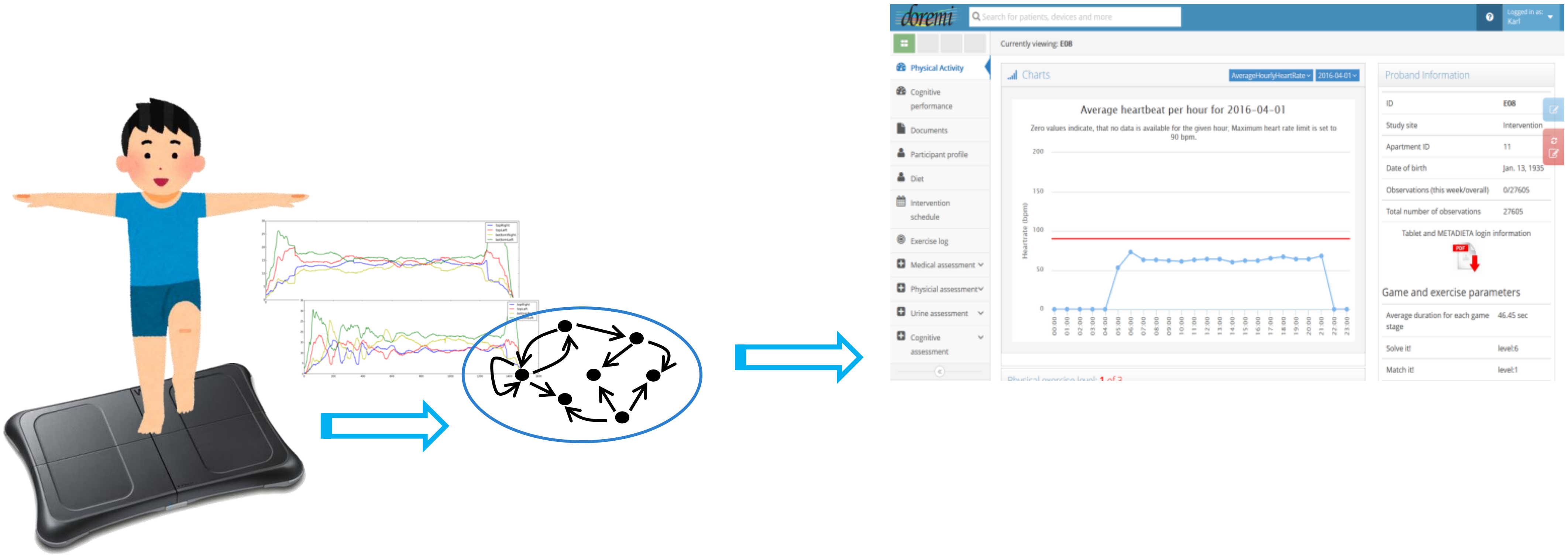
It works under certain conditions concerning the eigenvalues of the recurrent weight matrix (randomly initialized and not learned)

Reservoir Computing – What do I gain?

- Good predictive performance on highly noisy input signals and short-term memory tasks
- Computational and memory efficiency
  - Trains in seconds (Vs hours/days)
  - Even on embedded devices (computation, memory and energy constraints)
  - Consider physiological monitoring applications
- Comes also in deep learning fashion (and with some adaptivity reintroduced in the recurrent layer)
- Can be implemented in hardware (neuromorphic)

A Reservoir Computing Application

- Automatic assessment of balance skills
- Predict the outcome of the Berg Balance Scale (BBS) clinical test from time-series of pressure sensors (in 10 secs Vs 10 minutes)



Wrap-up

Take Home Lessons

- Recurrent neural networks create a dynamic memory of past inputs which influences neural activation besides the current input
  - Good inductive bias for sequential data
  - Amounts to weight sharing in time
- Learning long-term dependencies can be difficult due to gradient vanish/explosion so you need smarter solutions than vanilla RNNs
  - Gated RNNs: control memory reading and writing by gates
  - Temporal convolution networks: use dilation factor to broaden the scope of how much past a neuron can see
  - Reservoir computing: use randomization in place of learning when you have computational constraints (and the right task)
- Dealing with physiological timeseries typically requires preprocessing carefully

Next Lectures

- Laboratory tutorial (Tuesday)

Next 3 lectures:

- Deep learning fundamentals
  - Sequence-to-sequence learning and encoder-decoder architectures
  - Neural attention
  - Transformers and vision transformers
- Natural language and text data processing
  - Learning dense embeddings
  - Natural language processing pipeline and tasks
  - Language modelling
- Application verticals
  - Language models for healthcare
  - Dealing with language in HER