

DEEP LEARNING & APPLICATIONS IN NON-COGNITIVE DOMAINS

PART II: PRACTICE

Truyen Tran

Deakin University

truyen.tran@deakin.edu.au

prada-research.net/~truyen

AusDM'16, Canberra, Dec 7th 2016

**GET
YOUR
HANDS
DIRTY**

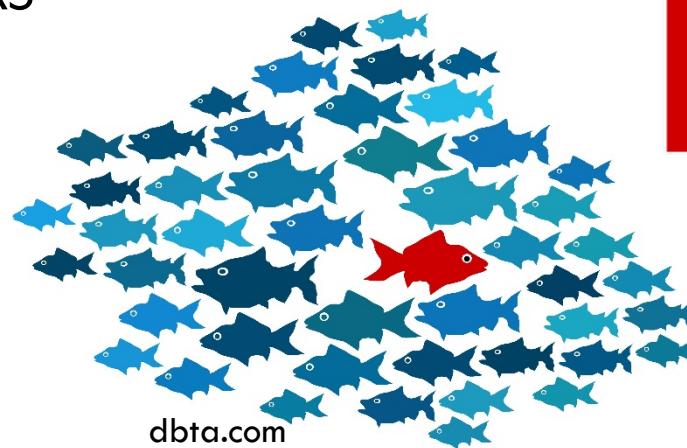
PART II: PRACTICE APPLYING DEEP LEARNING TO NON-COGNITIVE DOMAINS

Hand-on:

- Introducing programming frameworks
(Theano, TensorFlow, Mxnet)

Domains how-to:

- Healthcare
- Software engineering
- Anomaly detection



THEANO & TENSORFLOW

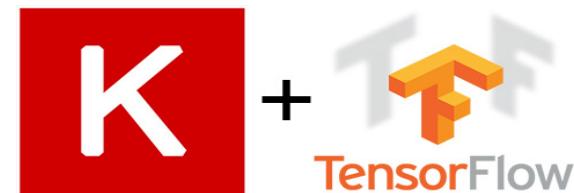
Two most popular frameworks at present. Both in Python.

Theano

- Academic-driven. Pioneer.
- Symbolic computation → can be tricky to debug
- Wrapper: Lasagne, Keras

TensorFlow

- Google → Native distributed computing support
- A lot of support, huge community
- Slightly bigger/messier code
- Linux/Mac only but VirtualBox will help in Windows
- Wrapper: Keras





- ✓ Excellent support for many languages
- ✓ Fast, portable
- ✓ Intuitive syntax
- ✓ Recent choice by AWS

Tensorflow vs. MXNET

	Languages	MultiGPU	Distributed	Mobile	Runtime Engine
Tensorflow	Python	Yes	No	Yes	?
MXNET	Python, R, Julia, Go	Yes	Yes	Yes	Yes
(Googlenet) E5-1650/980	Tensor flow	Torch7	Caffe	MXNET	
Time	940ms	172ms	170ms	180ms	
Memory	all (OOM after 24)	2.1GB	2.2GB	1.6GB	

Marianas Labs

Carnegie Mellon University

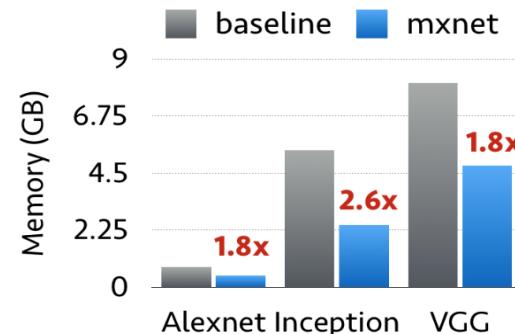
<http://bickson.blogspot.com.au/2016/02/mxnet-vs-tensorflow.html>

Portable

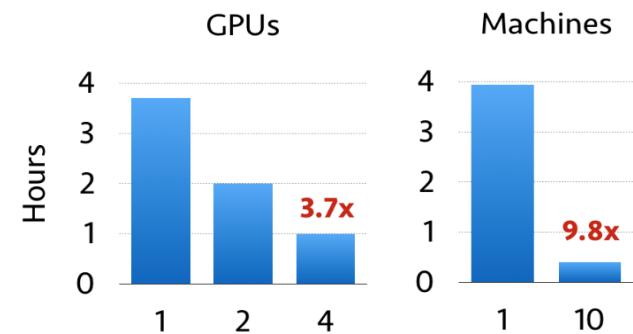


<https://github.com/dmlc/mxnet>

Efficient



Scalable



BUILDING A MODEL

Check the model assumption

- Is this only the vector → FNN?
- Is this a regular sequence → RNN?
- Is there repeated motifs → CNN?
- Is there a mix of static and dynamic features?
- What does the output look like?
 - A class
 - A sequence
 - An image?
- What are performance measures? → Surrogate smooth objective functions

Everything is a computational graph

From here to there is a tensor

So simple stacking is fine (the idea behind Keras)

Fit small datasets first to test the water

- But be cautious: small data do not always generalize

Always monitor the gap between train/validation sets: small gap indicates underfitting, big widening gap indicates overfitting.

STEPS

Prepare a clean big dataset

Design a suitable architecture → the main ART

Choose an optimizer (sgd, momentum, adagrad, adadelta, rmsprop, adam)

Normalise data (very important for fast training & well-behaved learning curve)

Shuffle data randomly (extremely important!)

Run the optimizer

Sit back & wait (in fact, should spend time monitor the convergence)

Grid search if time permits (sometimes very important to get correct convergence!)

Ensemble if time permits

Reiterate if needed

THINGS TO TAKE CARE OF

Data quality

Leakage

- Never touch validation data for feature engineering
- Be aware of overlapping between training/validation in time-sensitive data

Memory limitation

CPU/GPU time

Always shuffle the data BEFORE training – create a mixing of labels

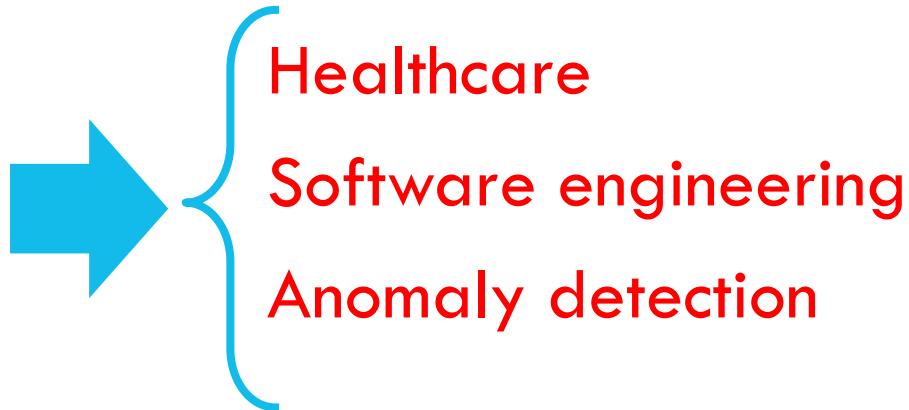
Initialisation matters

Dropouts: almost always help, normally with bigger models. But be careful with RNNs.

Numerical overflow/underflow: exp of large number, log of or division by zeros

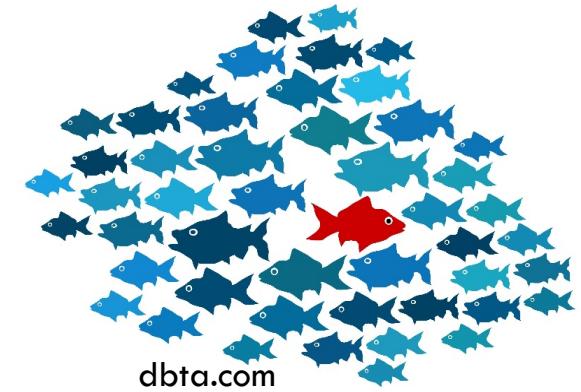
APPLYING TO NON-COGNITIVE DOMAINS

- Where humans need extensive training to do well
- Domains that demand transparency & interpretability.



```
string hql = "from Person p where ";
if(parameters.contains("name")){
    hql += " and p.name = :name";
}
if(parameters.contains("age")){
    hql += " and p.age = :age";
}
TypedQuery<Person> query = em.createQuery(hql, Person.class);
if(parameters.contains("name")){
    query.setParameter("name", values[0].toString());
}
if(parameters.contains("age")){
    query.setParameter("age", Integer.valueOf(values[1]));
}
```

http://www.bentoaktechnologies.com/images/code_scrn.jpg



WHAT MAKE NON-COGNITIVE DOMAINS HARD?

Great diversity but may be small in size

High uncertainty, low-quality/missing data

Reusable models do not usually exist

However, at the end of the day, we need few generic things:

- Vector -> DNN (e.g., highway net)
- Sequence -> RNN (e.g., LSTM, GRU)
- Repeated Motifs -> CNN
- Set -> attention (*Will visit in Part III*)
- Graphs -> Column Networks (*Will visit in Part III*)



HEALTHCARE





HOW DOES AI WORK FOR HEALTH?



Diagnosis



Discovery



Prognosis



Efficiency

HEALTHCARE: CHALLENGES + OPPORTUNITIES

Long-term dependencies

May include textual notes

Irregular timing

May contain physiological signals (e.g., EEG/ECG)

Mixture of discrete codes and continuous measures

May contain images (e.g., MRI, X-ray, retina)

Complex interaction of diseases and care processes

Genomics

Cohort of interest can be small (e.g., <1K)

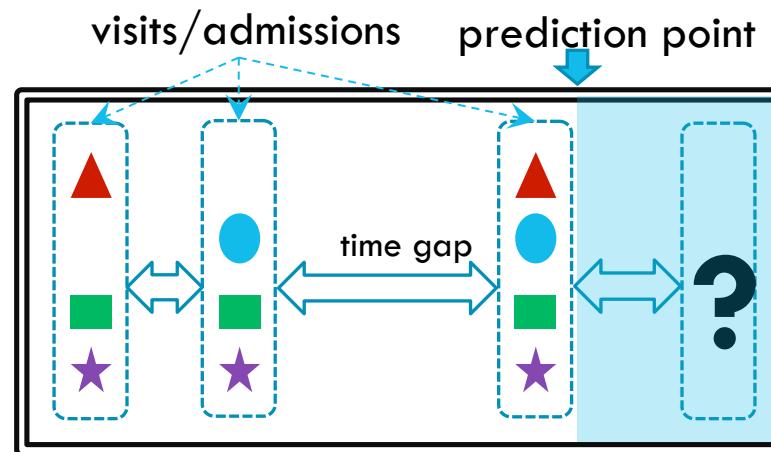
Detailed neuronal mapping (US) & simulation (EU)

Rich domain knowledge & ontologies

New modalities: social medial, wearable devices

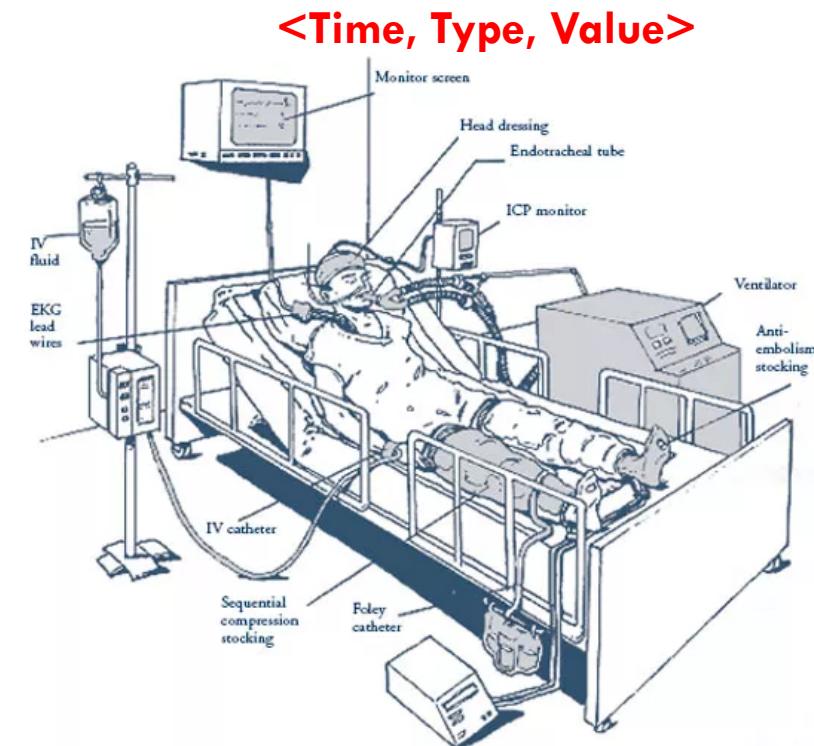
THIS TUTORIAL WILL COVER:

Electronic medical records (EMR)



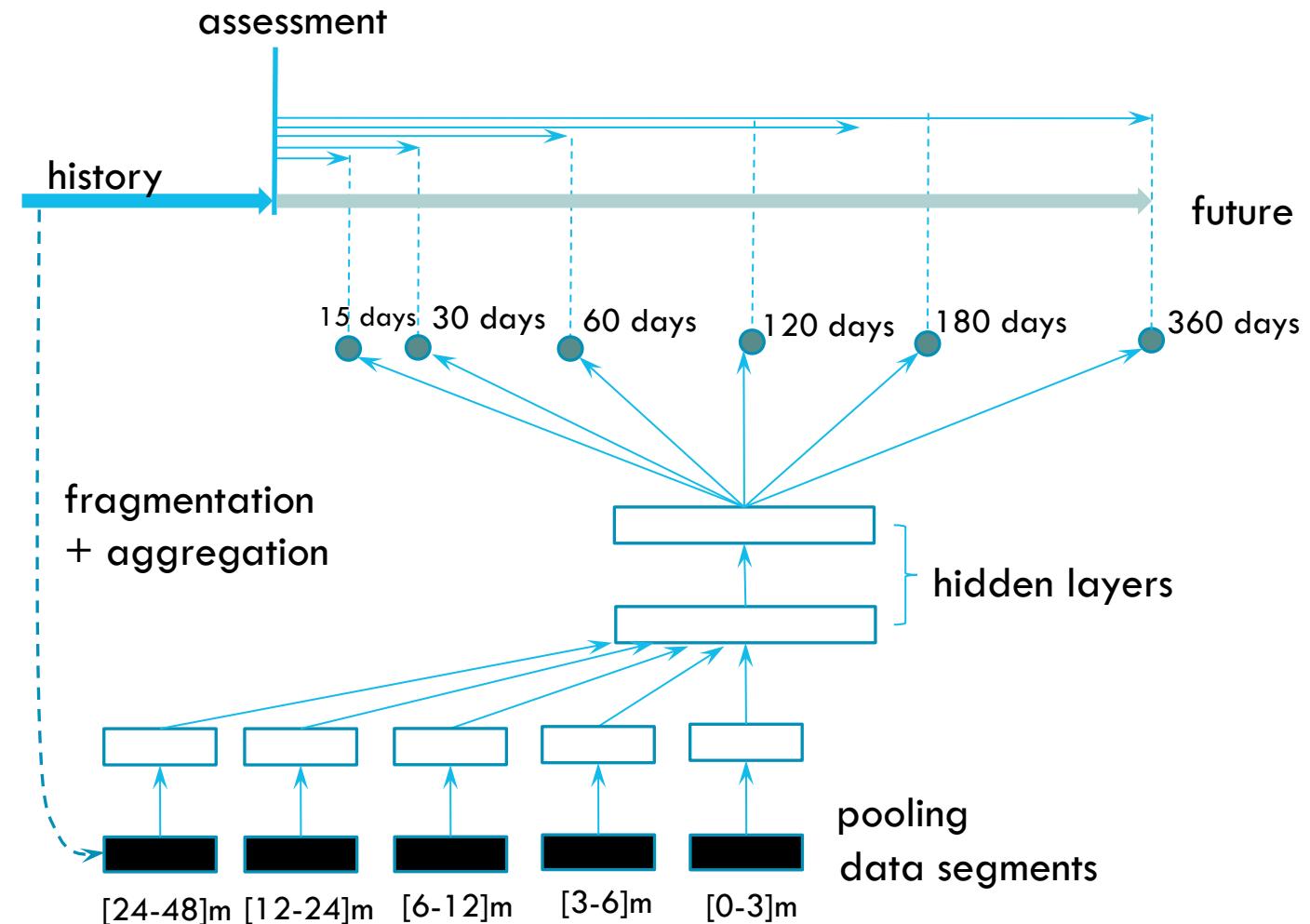
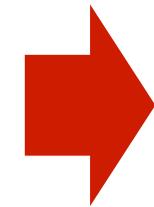
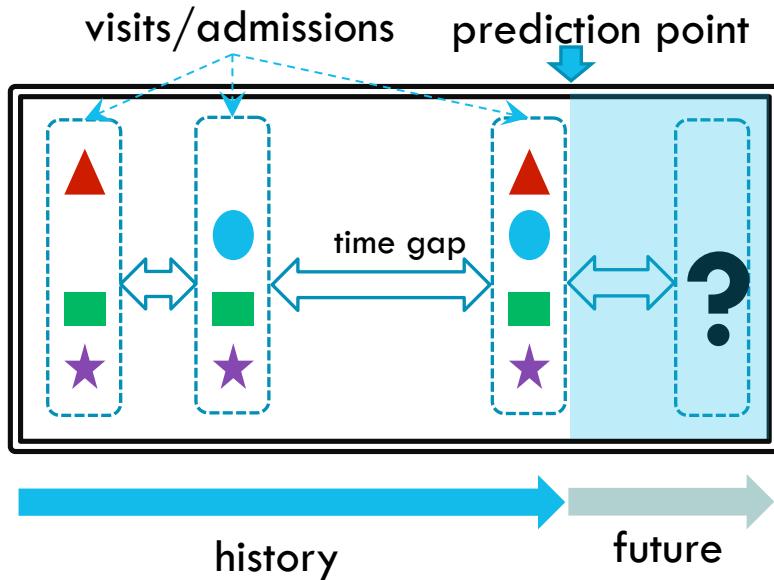
- Time-stamped
- Coded data: diagnosis, procedure & medication
- Text not considered, but in principle can be mapped in to vector using LSTM

Physiological measures in Intensive Care Unit (CU)

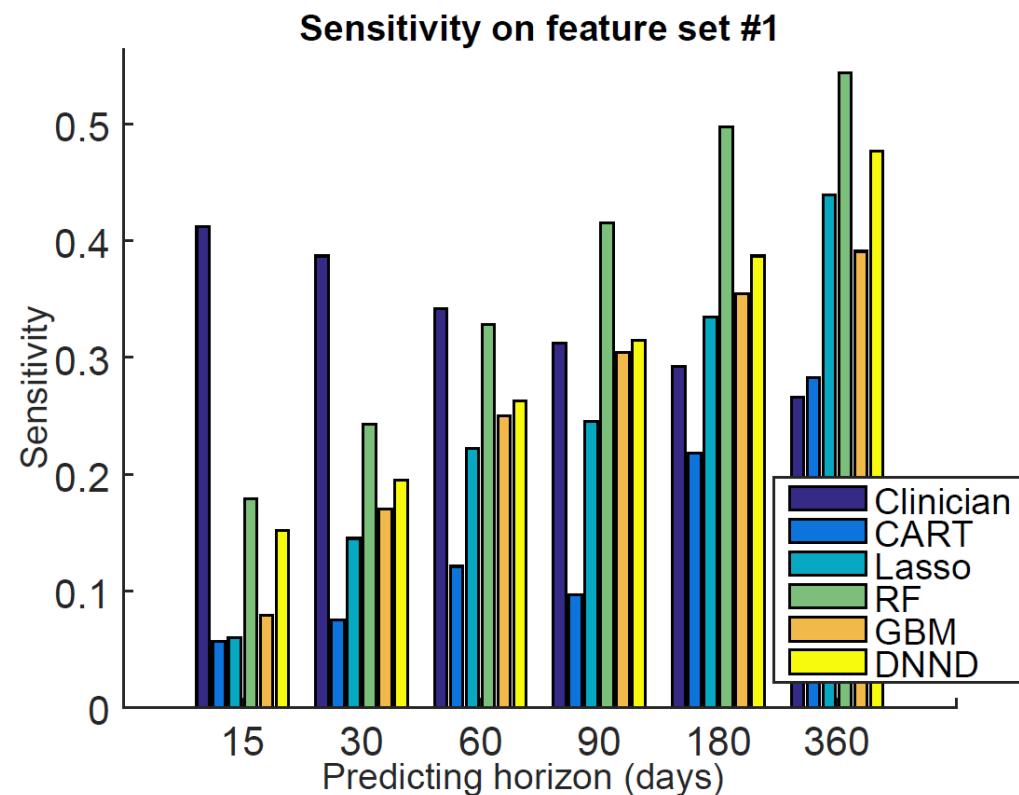


<http://www.healthpages.org/brain-injury/brain-injury-intensive-care-unit-icu/>

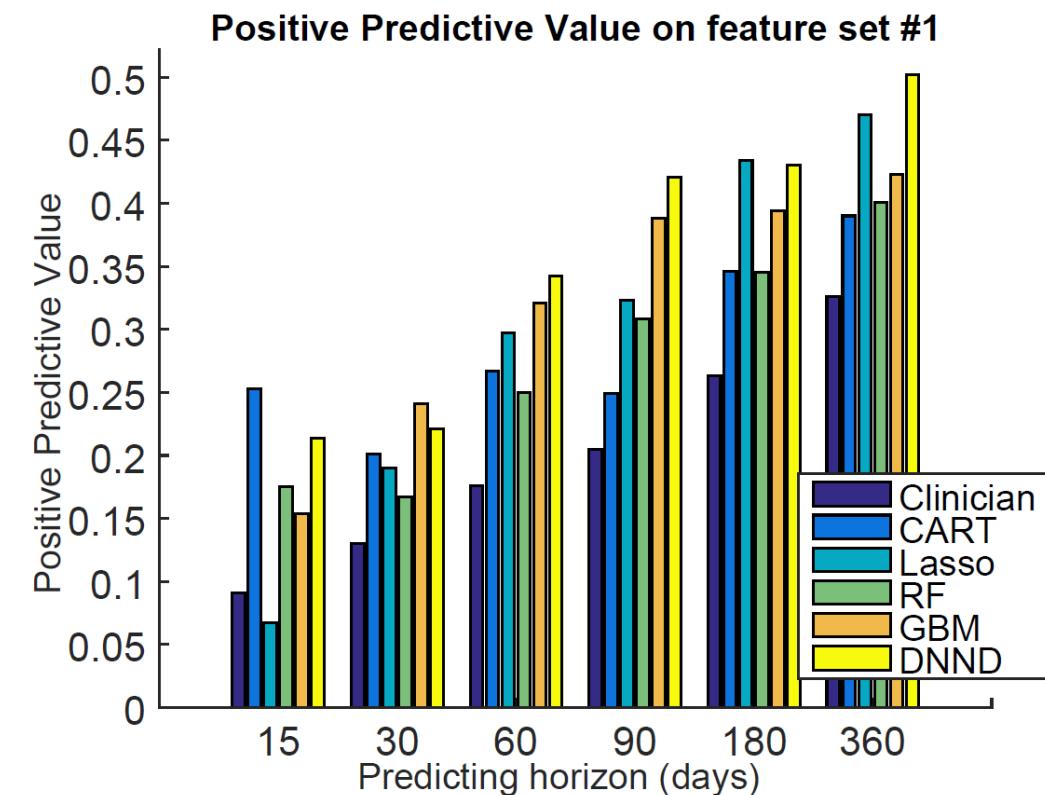
MEDICAL RECORDS: FEEDFORWARD NETS



SUICIDE RISK PREDICTION: MACHINE VERSUS CLINICIAN

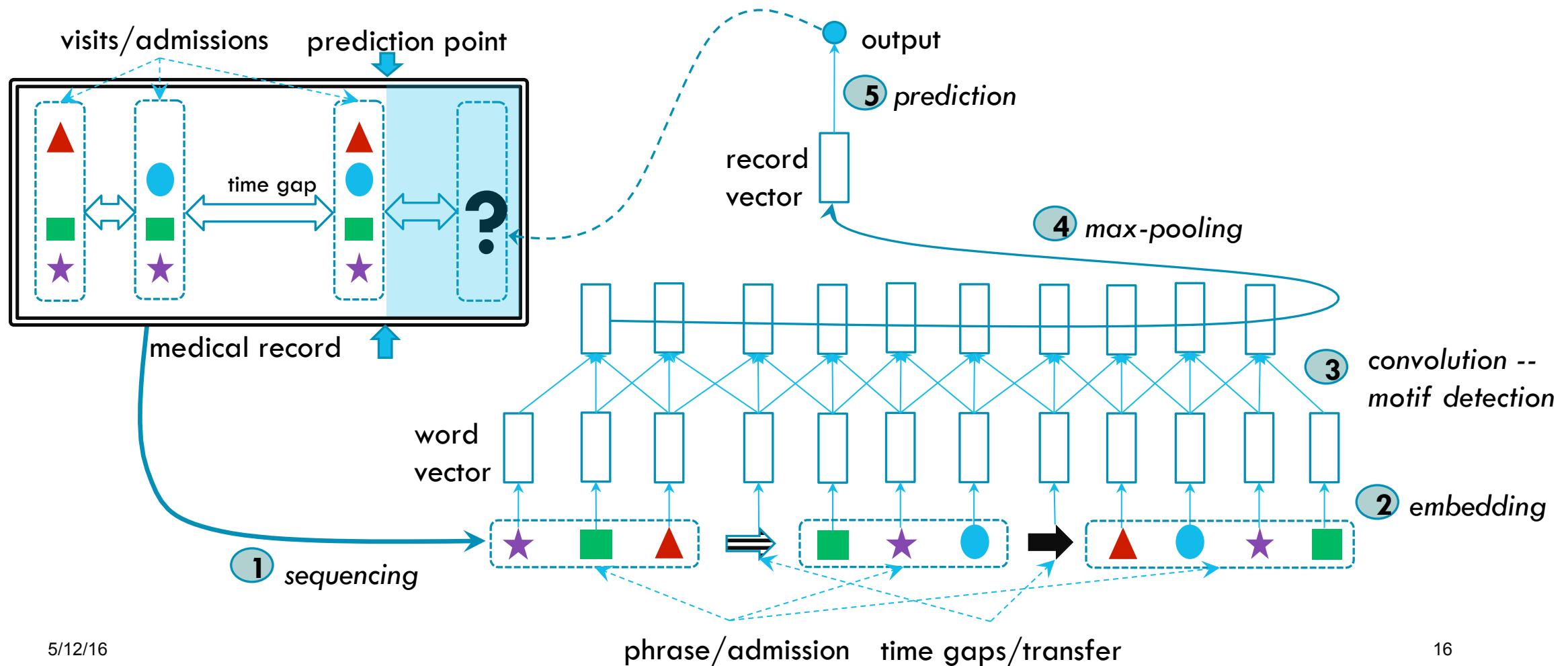


(a) Recall



(b) Precision

DEEPR: CNN FOR REPEATED MOTIFS AND SHORT SEQUENCES (NGUYEN ET AL, J-BHI, 2016)



DISEASE EMBEDDING & MOTIFS DETECTION

E11 I48 I50

Type 2 diabetes mellitus

Atrial fibrillation and flutter

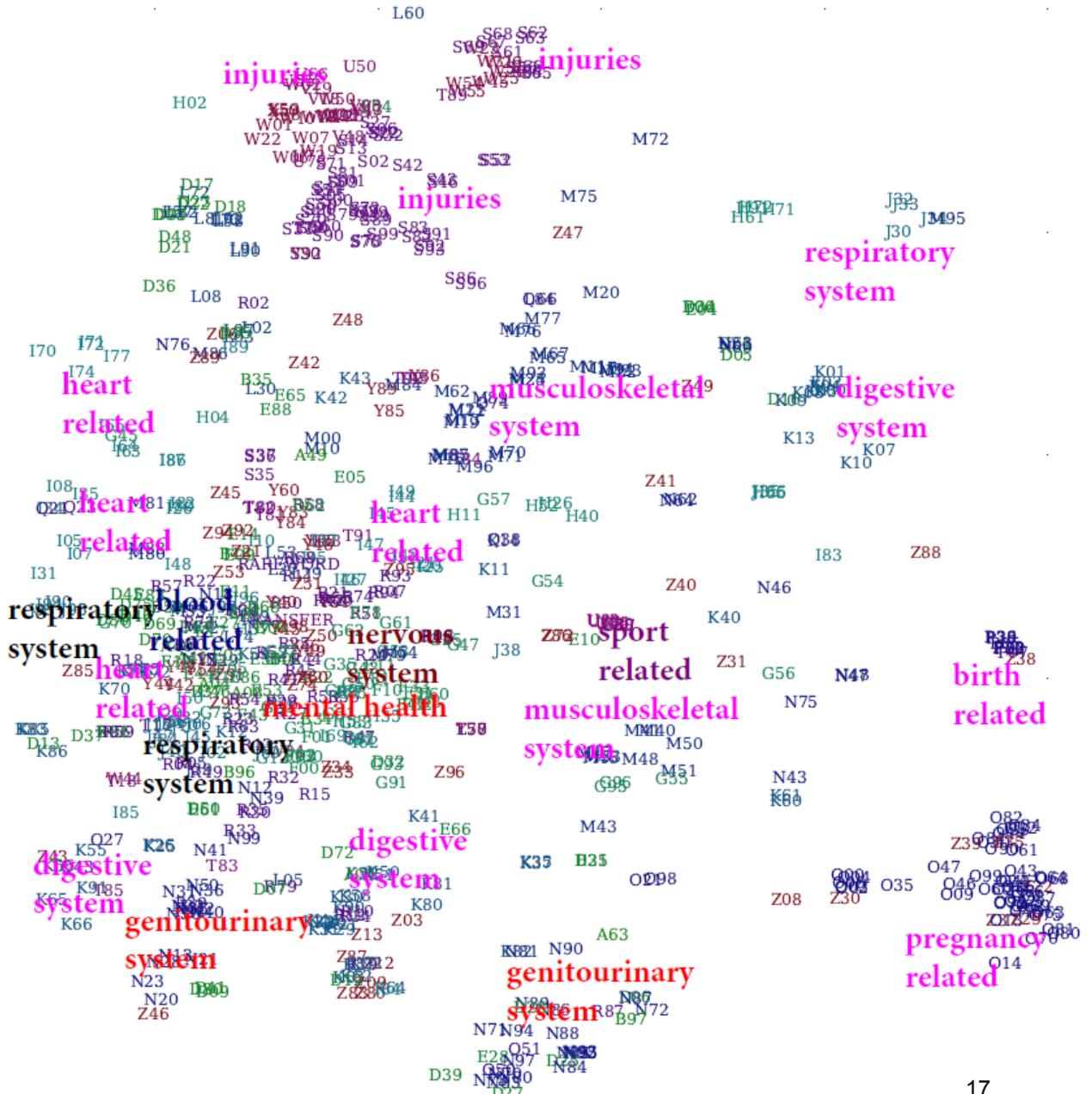
Heart failure

E11 I50 N17

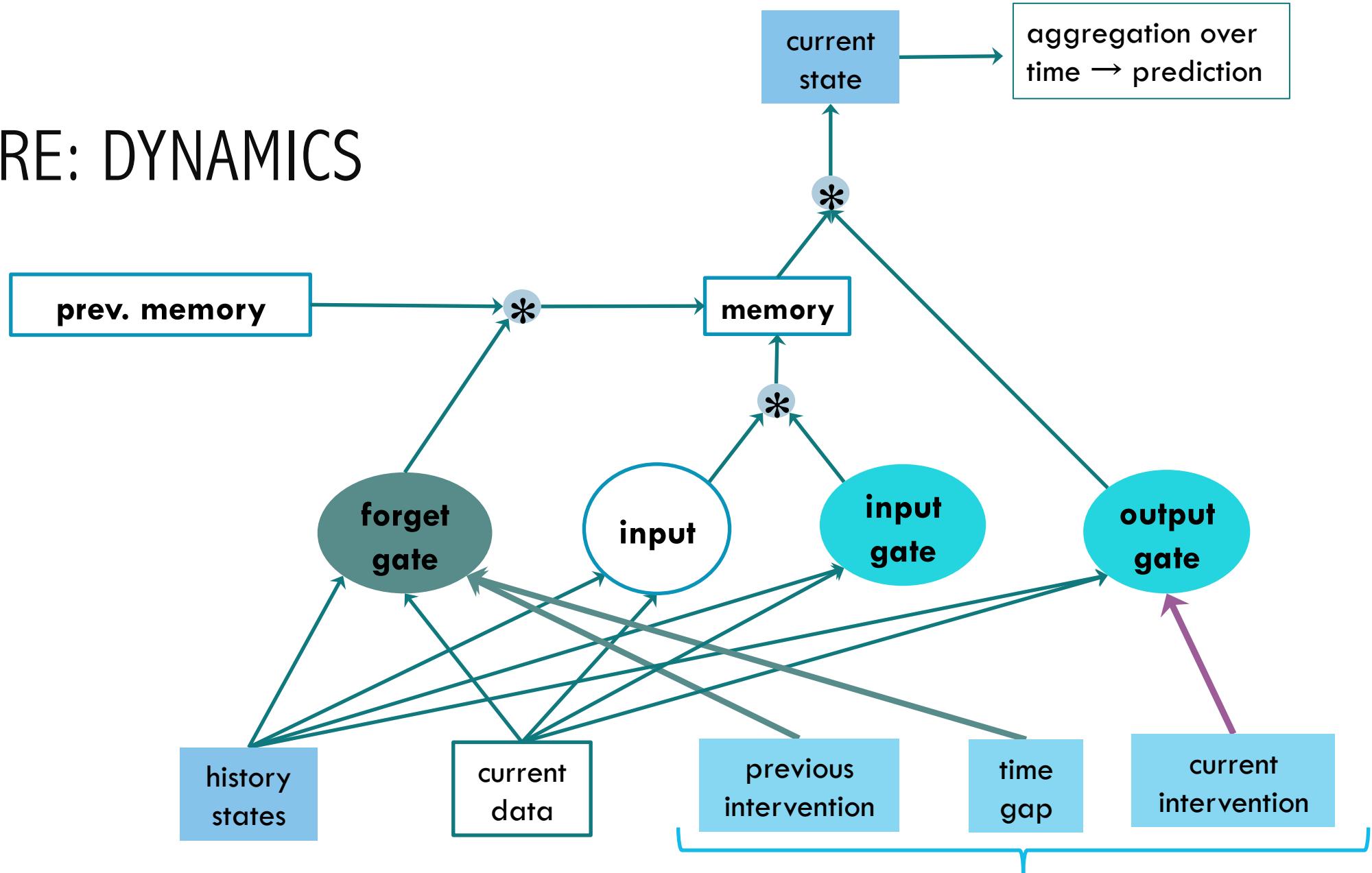
Type 2 diabetes mellitus

Heart failure

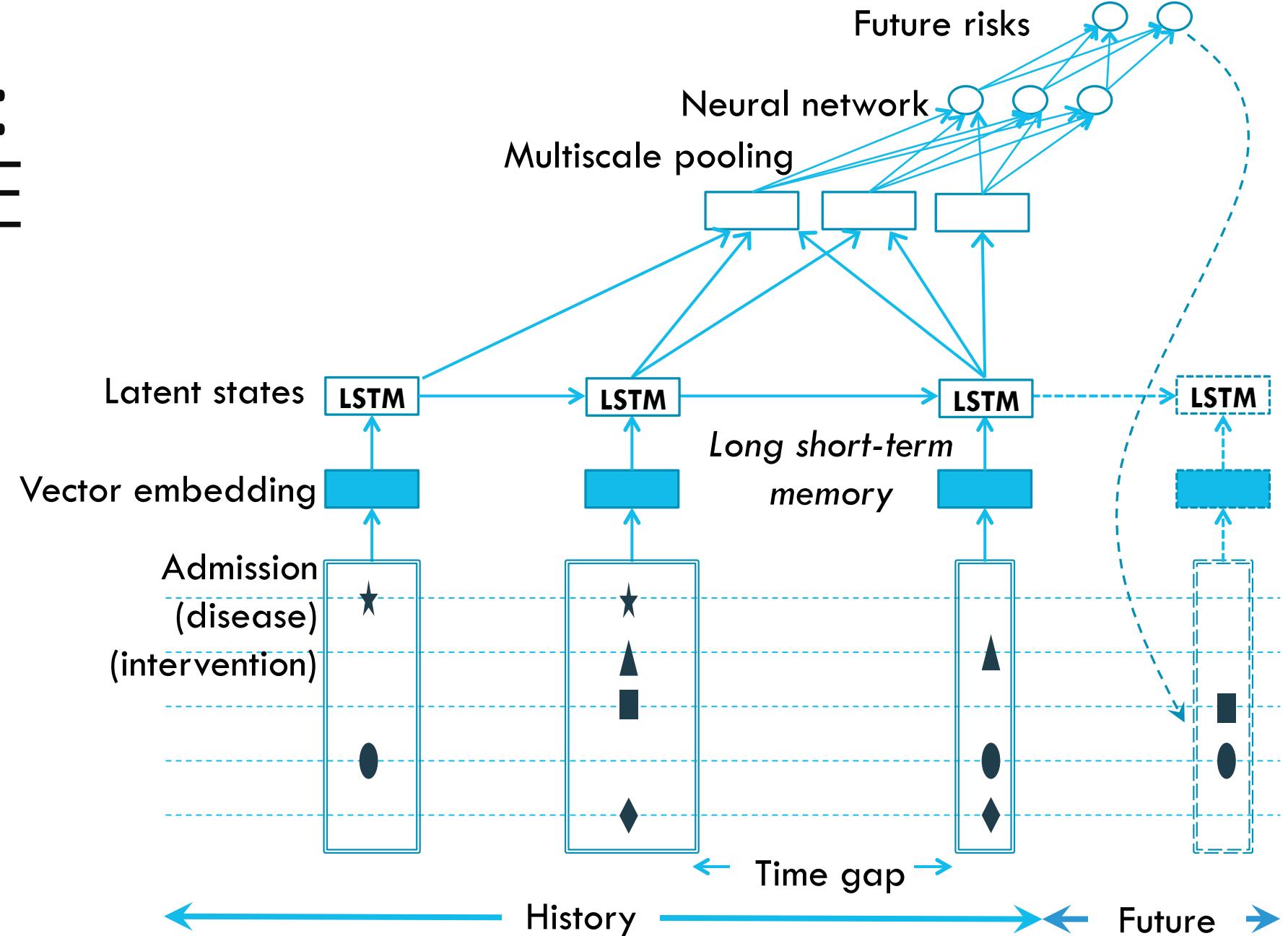
Acute kidney failure



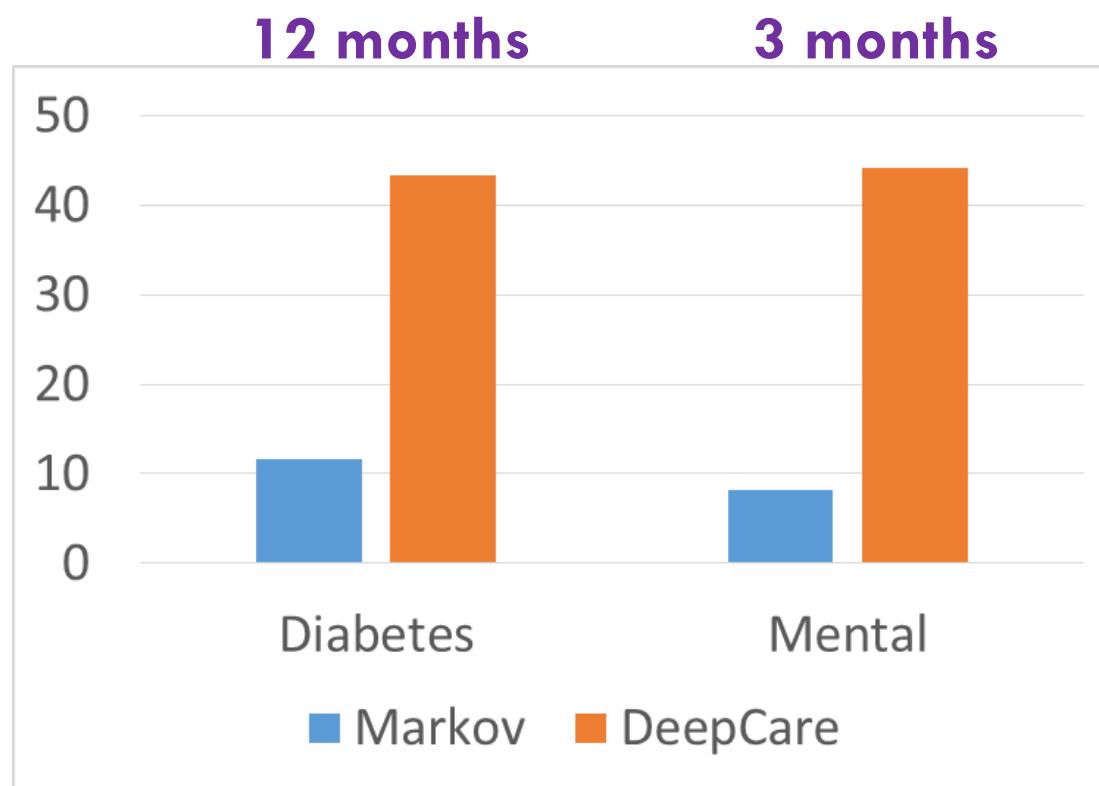
DEEPCARE: DYNAMICS



DEEPCARE: STRUCTURE

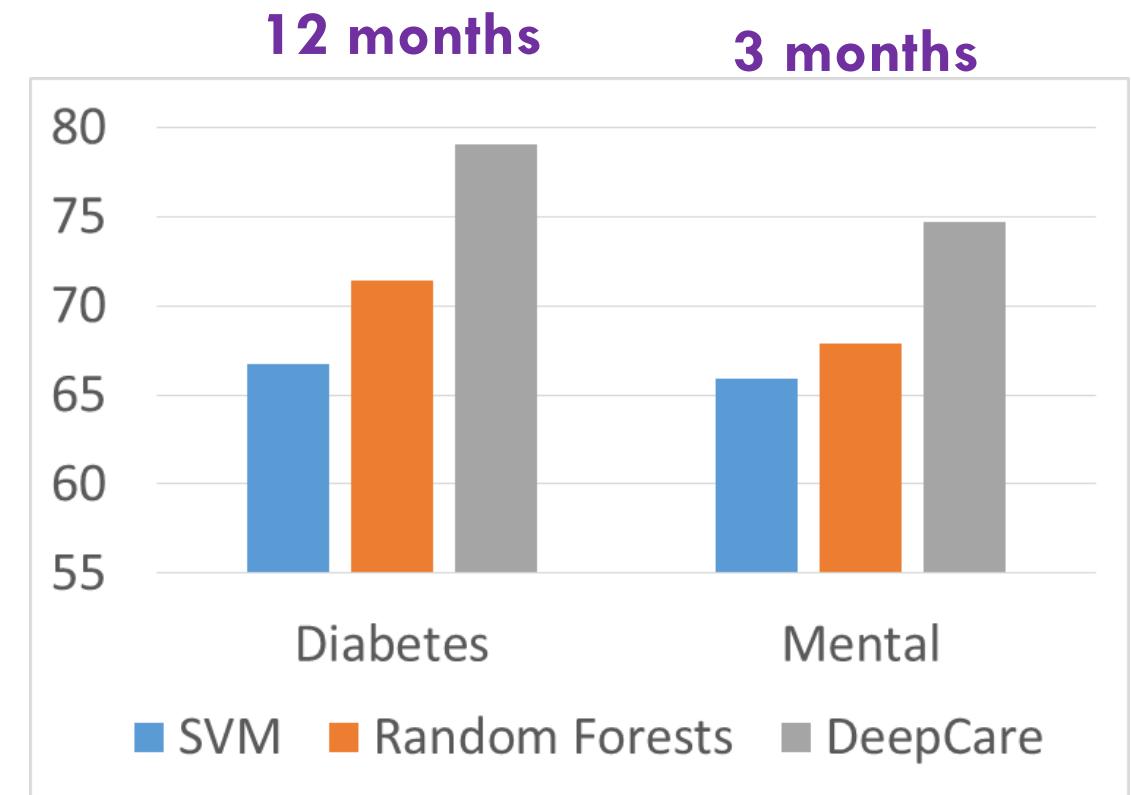


DEEPCARE: PREDICTION RESULTS



Intervention recommendation (precision@3)

5/12/16



Unplanned readmission prediction (F-score)

20

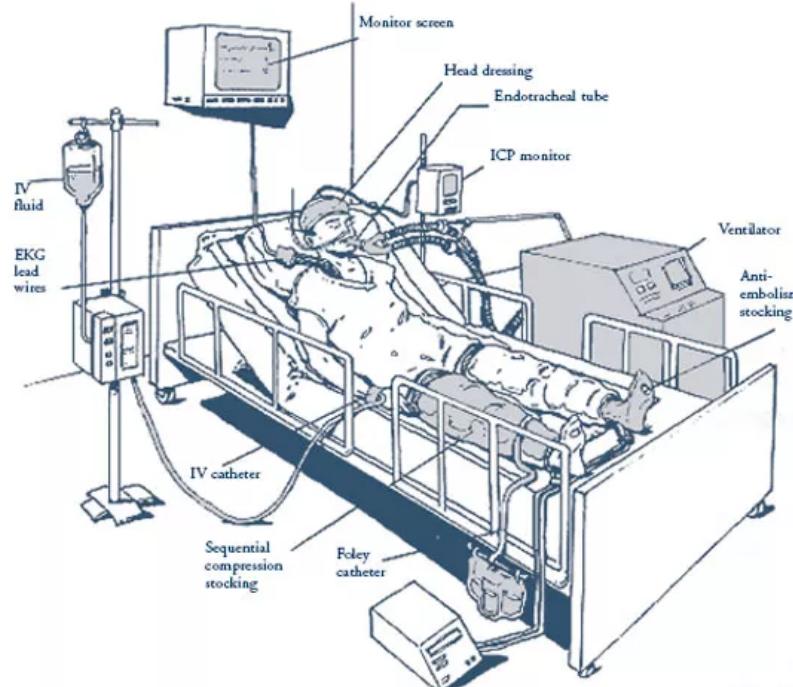
DEEPIIC: MORTALITY PREDICTION IN INTENSIVE CARE UNITS (WORK IN PROGRESS)

Existing methods: LSTM with missingness and time-gap as input.

New method: Deepic

Steps:

- Measurement quantization
- Time gap quantization
- Sequencing words into sentence
- CNN

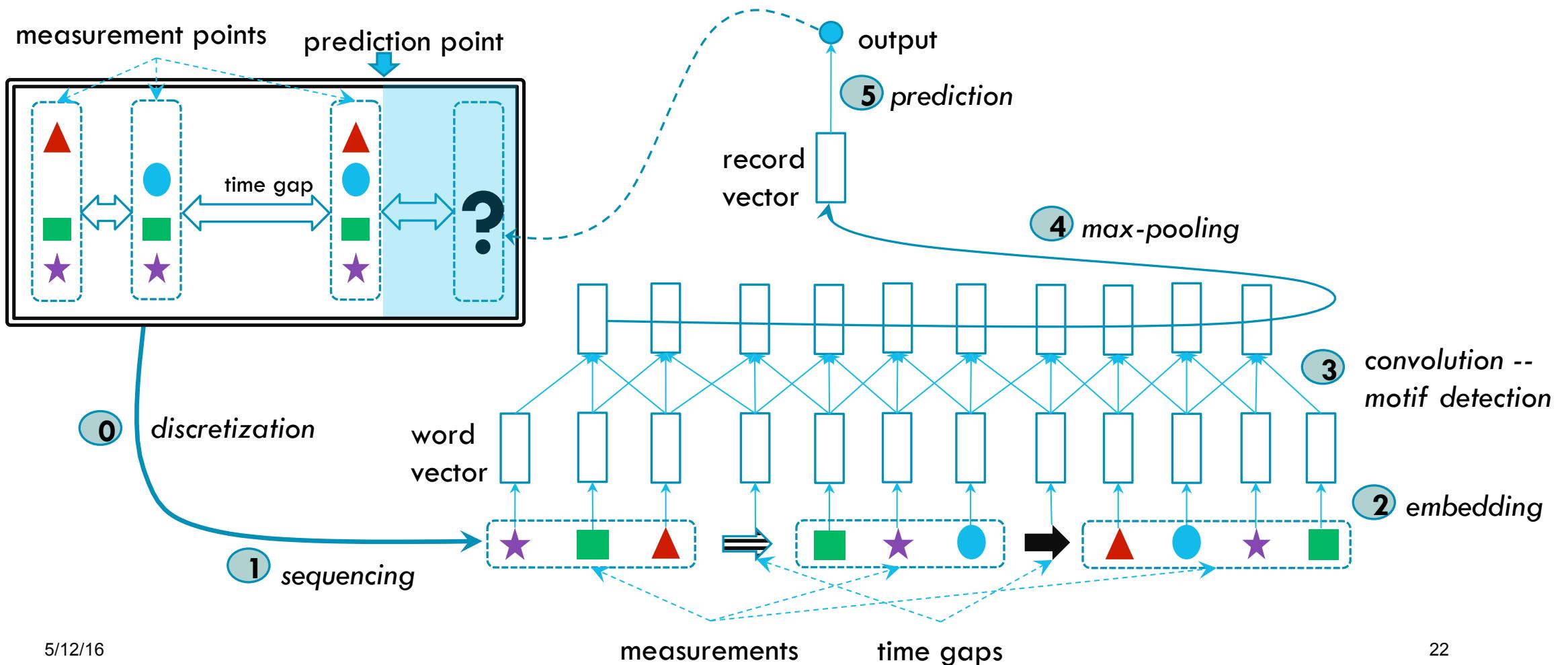


<http://www.healthpages.org/brain-injury/brain-injury-intensive-care-unit-icu/>

Time	Parameter	Value
00:00	RecordID	132539
00:00	Age	54
00:00	Gender	0
00:00	Height	-1
00:00	ICUType	4
00:00	Weight	-1
00:07	GCS	15
00:07	HR	73
00:07	NIDiasABP	65
00:07	NIMAP	92.33
00:07	NISysABP	147
00:07	RespRate	19
00:07	Temp	35.1
00:07	Urine	900
00:37	HR	77
00:37	NIDiasABP	58
00:37	NIMAP	91
00:37	NISysABP	157
00:37	RespRate	19
00:37	Temp	35.6
00:37	Urine	60

Data: **Physionet 2012**

DEEPIC: SYMBOLIC & TIME GAP REPRESENTATION OF DATA



SOFTWARE ANALYTICS

DATA-DRIVEN SOFTWARE ENGINEERING



TOWARDS INTELLIGENT ASSISTANTS



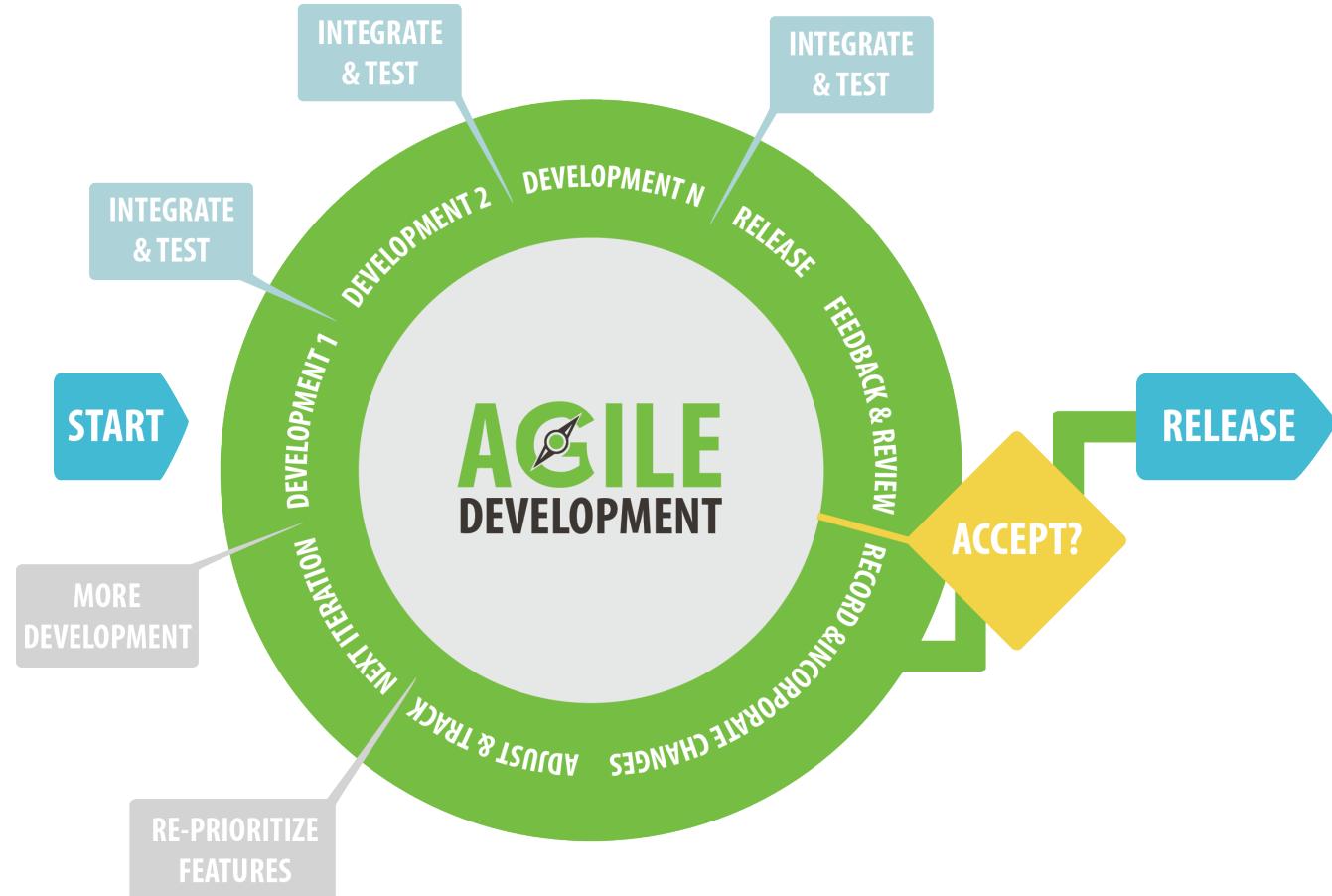
http://lifelong.engr.utexas.edu/images/course/swpm_b.jpg

Goal: To model code, text, team, user, execution, project & enabled business process → answer any queries by developers, managers, users and business

For now:

- DeepSoft vision
- LSTM for code language model
- LD-RNN for report representation
- Stacked/deep inference (later)

ANALYTICS FOR AGILE SOFTWARE PROJECT MANAGEMENT



CHALLENGES: LONG-TERM TEMPORAL DEPENDENCIES IN SOFTWARE

Software is similar to an **evolving** organism

- What will happen next to a software system depends heavily on what has previously been done to it.
- E.g. the implementation of a functionality may constraint how other functionalities are implemented in the *future*.
- E.g. a *previous* change (to fix a bug or add a new feature) may inject *new* bugs and lead to further changes.
- E.g. refactoring a piece of code may have *long-term* benefits in *future* maintenance.

Today's software products undergo rapid cycles of development, testing and release

- A software **project** typically has many **releases**
- A release requires the completion of some tasks (i.e. resolution of some **issues**).
- An issue is described using natural language (*raw data*).
- The resolution of an issue may result in code patches (*raw data*).

A DEEP LANGUAGE MODEL FOR SOFTWARE CODE (DAM ET AL, FSE'16 SE+NL)

A good language model for source code would capture the long-term dependencies

The model can be used for various prediction tasks, e.g. defect prediction, code duplication, bug localization, etc.

The model can be extended to model software and its development process.



UNIVERSITY
OF WOLLONGONG
AUSTRALIA



DEAKIN
UNIVERSITY AUSTRALIA

CHARACTERISTICS OF SOFTWARE CODE

Repetitiveness

- E.g. `for (int i = 0; i < n; i++)`

Localness

- E.g. `for (int size` may appear more often than `for (int i` in some source files.

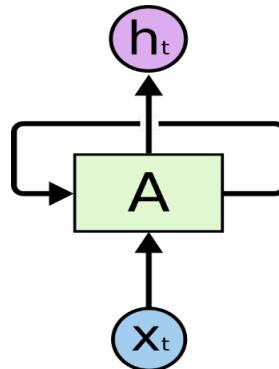
Rich and explicit structural information

- E.g. nested loops, inheritance hierarchies

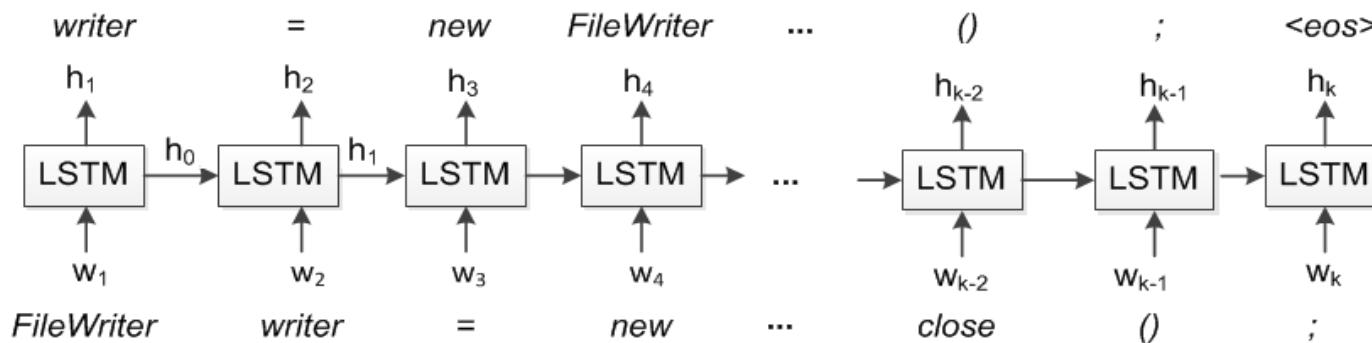
Long-term dependencies

- `try` and `catch` (in Java) or file `open` and `close` are not immediately followed each other.

CODE LANGUAGE MODEL



```
FileWriter writer = new FileWriter( file );
writer.write( "This is an example" );
int count = 0;
System.out.println( "Long gap" );
.....
writer.flush();
writer.close();
```



Previous work has applied RNNs to model software code (*White et al, MSR 2015*)

RNNs however do not capture the long-term dependencies in code

EXPERIMENTS

Built dataset of 10 Java projects: Ant, Batik, Cassandra, Eclipse-E4, Log4J, Lucene, Maven2, Maven3, Xalan-J, and Xerces.

Comments and blank lines removed. Each source code file is tokenized to produce a sequence of code tokens.

- Integers, real numbers, exponential notation, hexadecimal numbers replaced with <num> token, and constant strings replaced with <str> token.
- Replaced less “popular” tokens with <unk>

Code corpus of **6,103,191 code tokens**, with a vocabulary of **81,213 unique tokens**.

EXPERIMENTS (CONT.)

sent-len	embed-dim	RNN	LSTM	improv %
10	50	13.49	12.86	4.7
20		10.38	9.66	6.9
50		7.93	6.81	14.1
100		7.20	6.40	11.1
200		6.64	5.60	15.7
500		6.48	4.72	27.2
100	20	7.96	7.11	10.7
	50	7.20	6.40	11.1
	100	7.23	5.72	20.9
	200	9.14	5.68	37.9

Table 1: Perplexity on test data (the smaller the better).

Both RNN and LSTM improve with more training data (whose size grows with sequence length).

LSTM consistently performs better than RNN: 4.7% improvement to 27.2% (varying sequence length), 10.7% to 37.9% (varying embedding size).

STORY POINT ESTIMATION

Traditional estimation methods require experts, LOC or function points.

- Not applicable early
- Expensive

Feature engineering is not easy!

Needs a cheap way to start from just a documentation.

 Spring XD / XD-2970

Standardize XD logging to align with Spring Boot Title

Type:	Story	Status:	DONE
Priority:	Major	Resolution:	Complete
Affects Version/s:	1.2 GA	Fix Version/s:	1.2 RC1
Story Points:	8		
Sprint:	Sprint 49		

Description

In XD today we use commons-logging or slf4j APIs bound to log4j at runtime (configured with log4j.properties).

Boot uses slf4j APIs backed by logback. This causes some build incompatibilities building a component that depends on spring-xd-dirt and spring-boot, requiring specific dependency exclusions. In order to simplify building and troubleshooting log dependencies, XD should standardize on slf4j APIs (replace any commons-logging Loggers with Slf4j). This is internal only, and would not impact users who are used to seeing log4j.properties. An additional step is to replace log4j with logback. This change would be visible to end users but will provide us greater affinity with boot and improve the developer experience. If we make this change it should go into 1.2 GA.

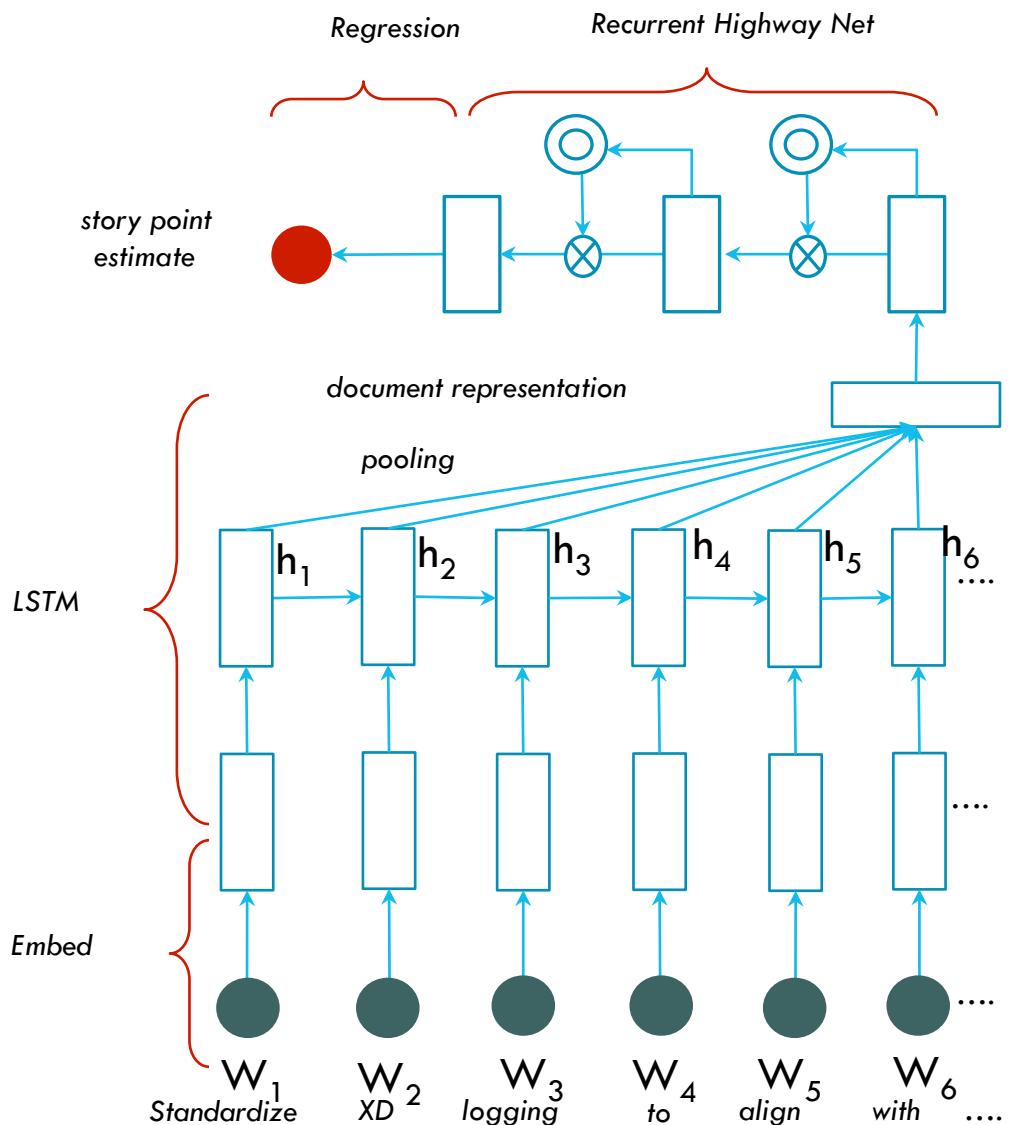
LD-RNN FOR REPORT REPRESENTATION

(CHOETKIERTIKUL ET AL, WORK IN PROGRESS)

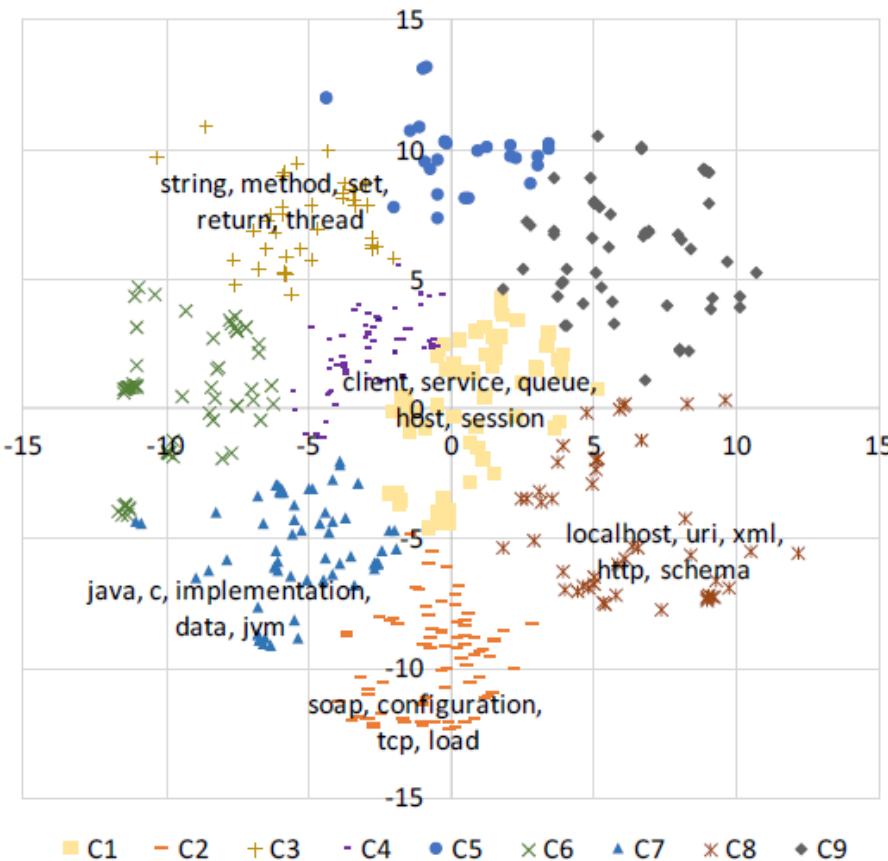
LD = Long Deep

LSTM for document representation

Highway-net with tied parameters for story point estimation



RESULTS



MAE = Mean Absolute Error

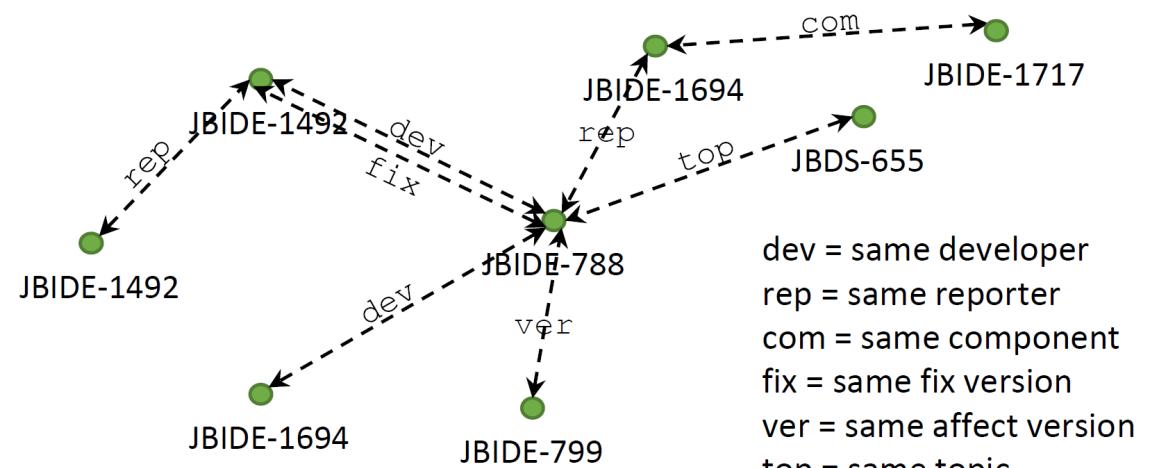
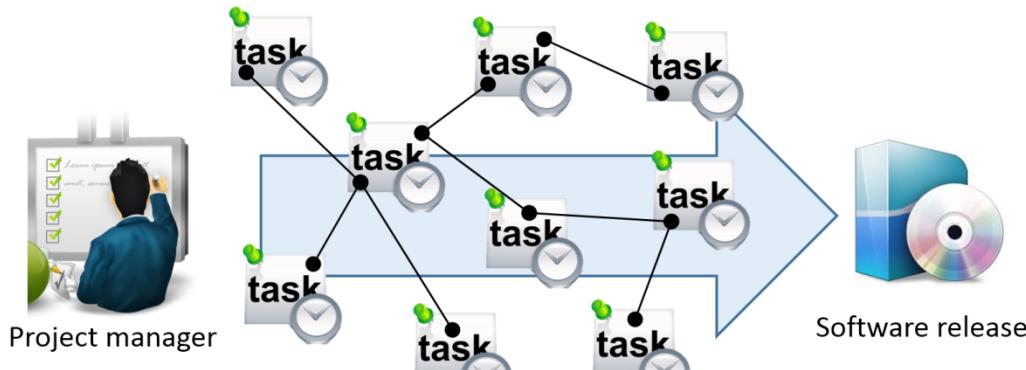
$$SA = \left(1 - \frac{MAE}{MAE_{rguess}} \right) \times 100$$

Proj	Technique	MAE	SA	Proj	Technique	MAE	SA
ME	LD-RNN	1.02	59.03	JI	LD-RNN	1.38	59.52
	LSTM+RF	1.08	57.57		LSTM+RF	1.71	49.71
	BoW+RF	1.31	48.66		BoW+RF	2.10	38.34
	Mean	1.64	35.61		Mean	2.48	27.06
	Median	1.73	32.01		Median	2.93	13.88
UG	LD-RNN	1.03	52.66	MD	LD-RNN	5.97	50.29
	LSTM+RF	1.07	50.70		LSTM+RF	9.86	17.86
	BoW+RF	1.19	45.24		BoW+RF	10.20	15.07
	Mean	1.48	32.13		Mean	10.90	9.16
	Median	1.60	26.29		Median	7.18	40.16
AS	LD-RNN	1.36	60.26	DM	LD-RNN	3.77	47.87
	LSTM+RF	1.62	52.38		LSTM+RF	4.51	37.71
	BoW+RF	1.83	46.34		BoW+RF	4.78	33.84
	Mean	2.08	39.02		Mean	5.29	26.85
	Median	1.84	46.17		Median	4.82	33.38
AP	LD-RNN	2.71	42.58	MU	LD-RNN	2.18	40.09
	LSTM+RF	2.97	37.09		LSTM+RF	2.23	38.73
	BoW+RF	2.96	37.34		BoW+RF	2.31	36.64
	Mean	3.15	33.30		Mean	2.59	28.82
	Median	3.71	21.54		Median	2.69	26.07

Fig. 4. Top-500 word clusters used in the Apache's issue reports

TASK DEPENDENCY IN SOFTWARE PROJECT

(CHOETKERTIKUL ET AL, WORK IN PROGRESS)

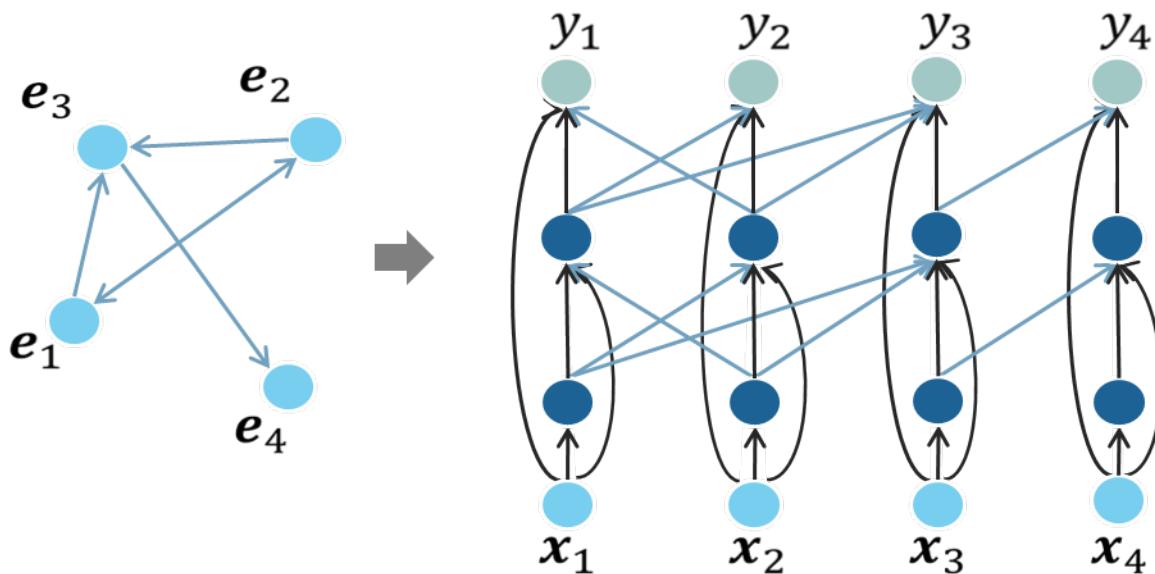


dev = same developer
rep = same reporter
com = same component
fix = same fix version
ver = same affect version
top = same topic

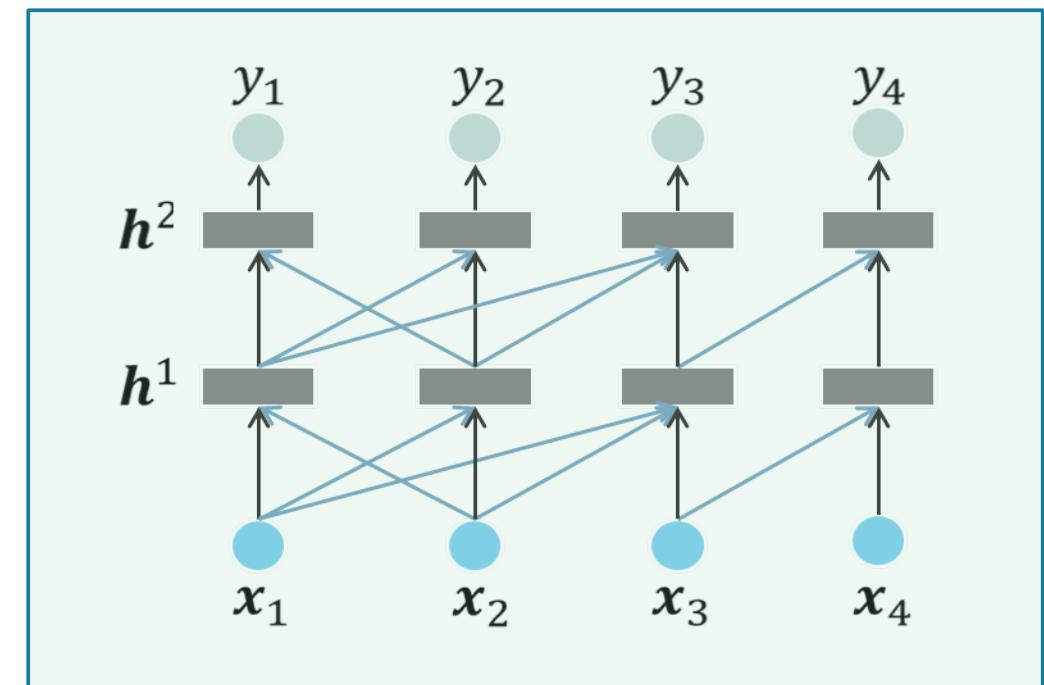
Approximately, **one-third** of IT projects went over the scheduled time

82% software projects missed schedules

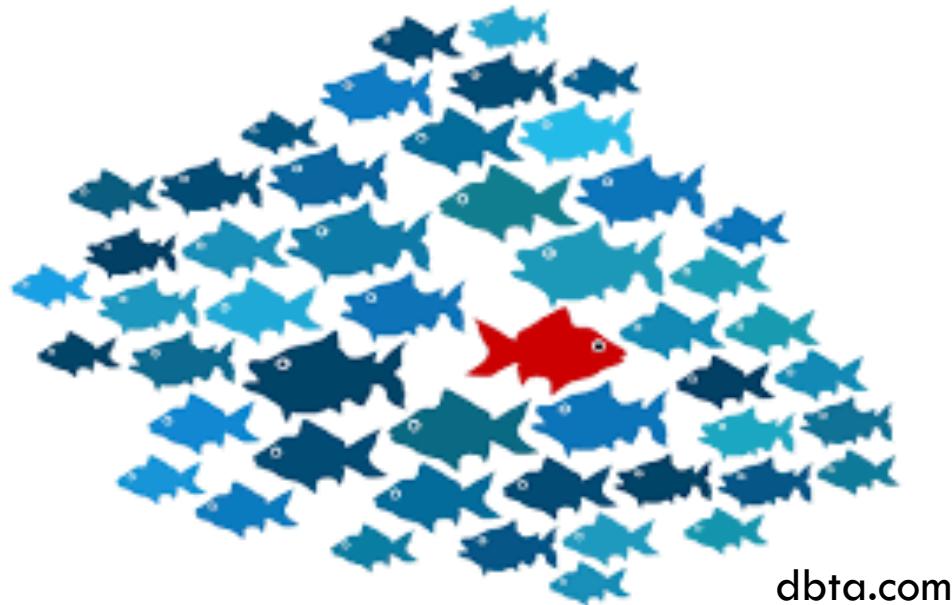
TASK DEPENDENCY IN SOFTWARE PROJECT (MORE ON PART III)



Stacked Inference



Column networks

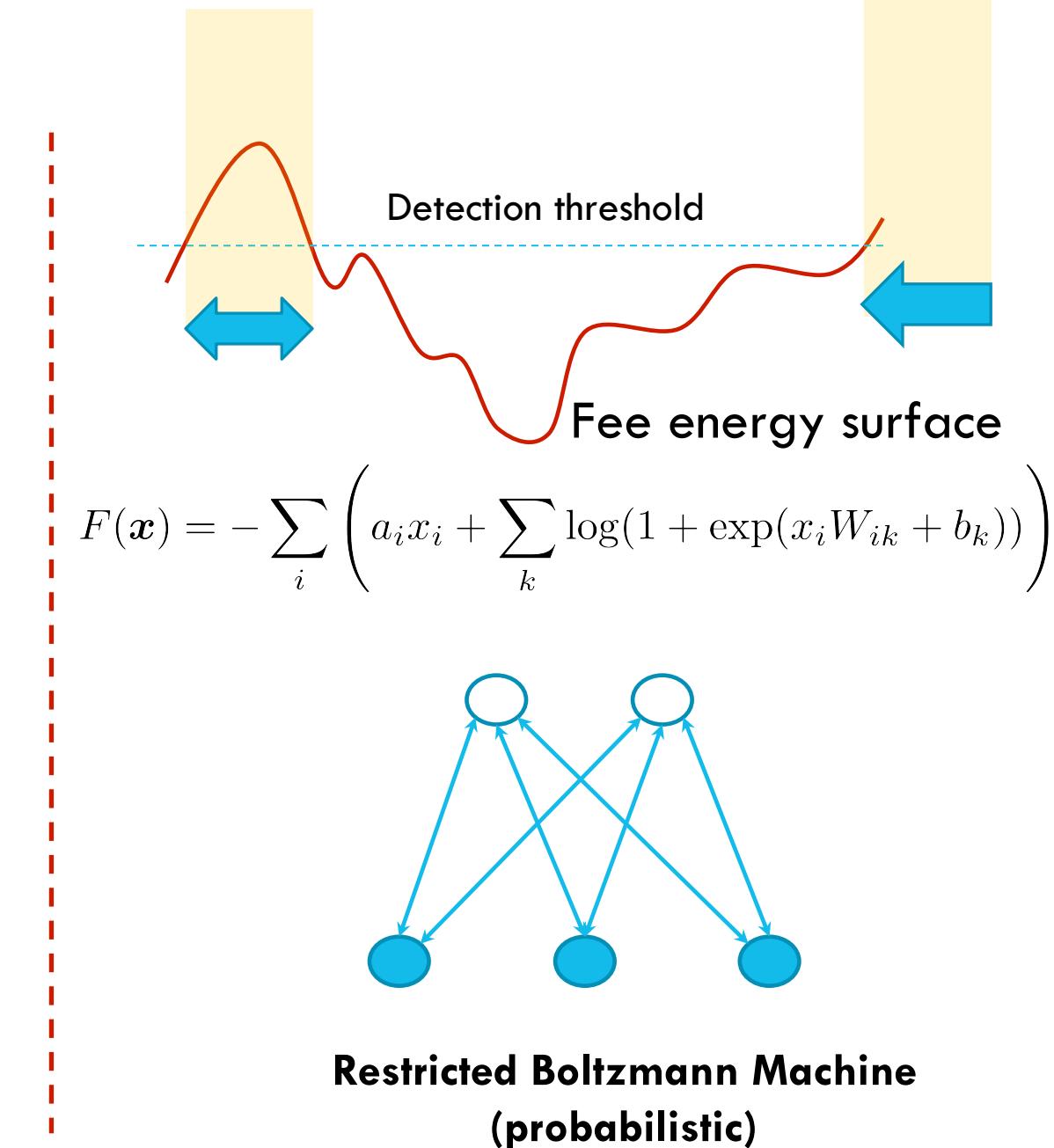
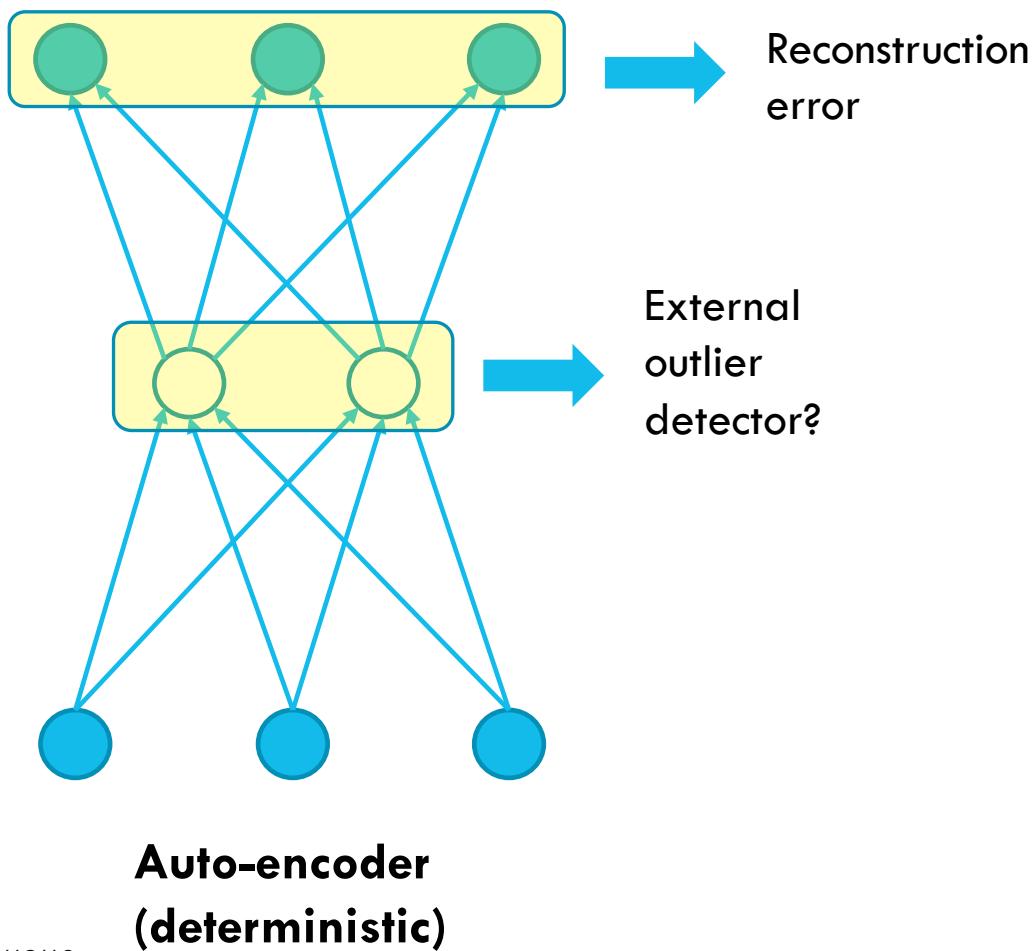


ANOMALY DETECTION USING UNSUPERVISED LEARNING (PART III)



This work is partially supported by the Telstra-Deakin Centre of Excellence in Big Data and Machine Learning

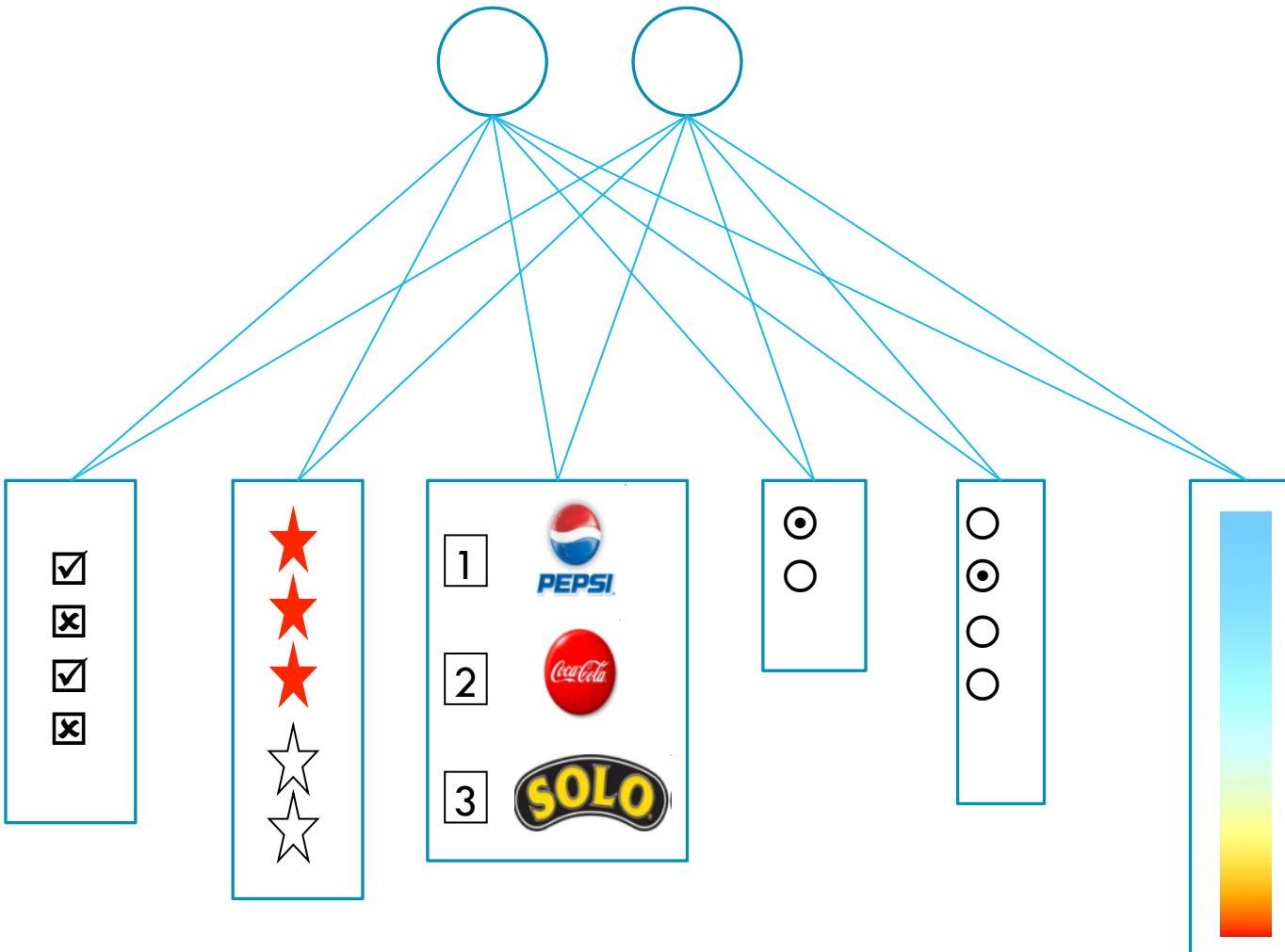
DETECTION METHODS



MIXED DATA

	A	B	C	D	E	F	G	H	I	J
1	Age	Sex	Chest pain type	Resting blood pressure	Serum cholestorol (mg/dl)	Fasting blood sugar > 120 mg/dl ?	Resting electrocardiographic result	Maximum heart rate achieved	Exercise induced angina	oldpeak = ST depression induced by exercise relative to rest
2	70	male	asymptomatic (4)	130.0	322.0	no	2	109.0	no	2.4
3	67	female	non-anginal pain (3)	115.0	564.0	no	2	160.0	no	1.6
4	57	male	atypical angina (2)	124.0	261.0	no	0	141.0	no	0.3
5	64	male	asymptomatic (4)	128.0	263.0	no	0	105.0	yes	0.2
6	74	female	atypical angina (2)	120.0	269.0	no	2	121.0	yes	0.2
7	65	male	asymptomatic (4)	120.0	177.0	no	0	140.0	no	0.4
8	56	male	non-anginal pain (3)	130.0	256.0	yes	2	142.0	yes	0.6
9	59	male	asymptomatic (4)	110.0	239.0	no	2	142.0	yes	1.2
10	60	male	asymptomatic (4)	140.0	293.0	no	2	170.0	no	1.2
11	63	female	asymptomatic (4)	150.0	407.0	no	2	154.0	no	4.0
12	59	male	asymptomatic (4)	135.0	234.0	no	0	161.0	no	0.5
13	53	male	asymptomatic (4)	142.0	226.0	no	2	111.0	yes	0.0
14	44	male	non-anginal pain (3)	140.0	235.0	no	2	180.0	no	0.0
15	61	male	typical angina (1)	134.0	234.0	no	0	145.0	no	2.6
16	57	female	asymptomatic (4)	128.0	303.0	no	2	159.0	no	0.0
17	71	female	asymptomatic (4)	112.0	149.0	no	0	125.0	no	1.6
18	46	male	asymptomatic (4)	140.0	311.0	no	0	120.0	yes	1.8
19	53	male	asymptomatic (4)	140.0	203.0	yes	2	155.0	yes	3.1
20	64	male	typical angina (1)	110.0	211.0	no	2	144.0	yes	1.8
21	40	male	typical angina (1)	140.0	199.0	no	0	178.0	yes	1.4
22	67	male	asymptomatic (4)	120.0	229.0	no	2	129.0	yes	2.6

MIXED-VARIATE RBM (TRAN ET AL, 2011)

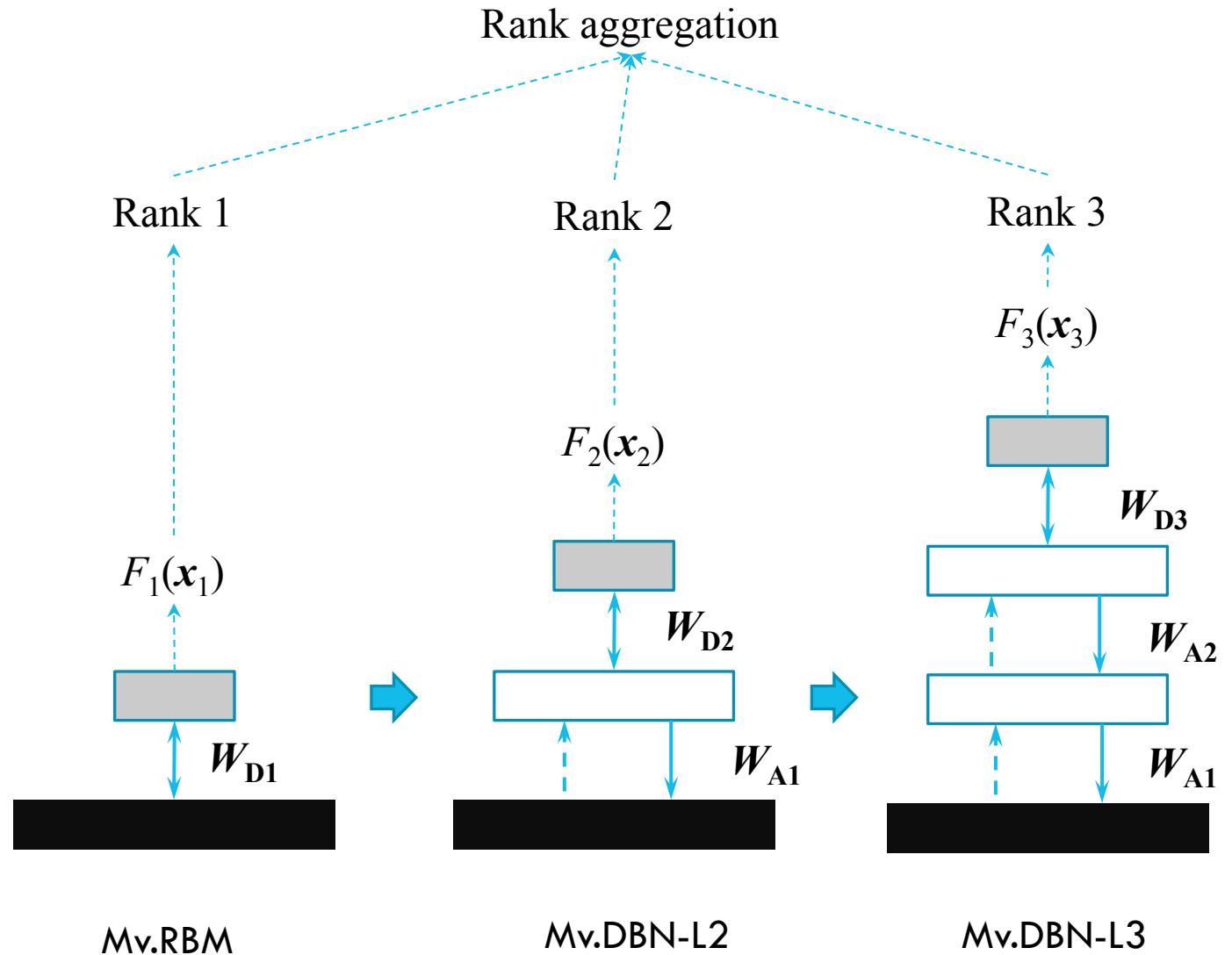


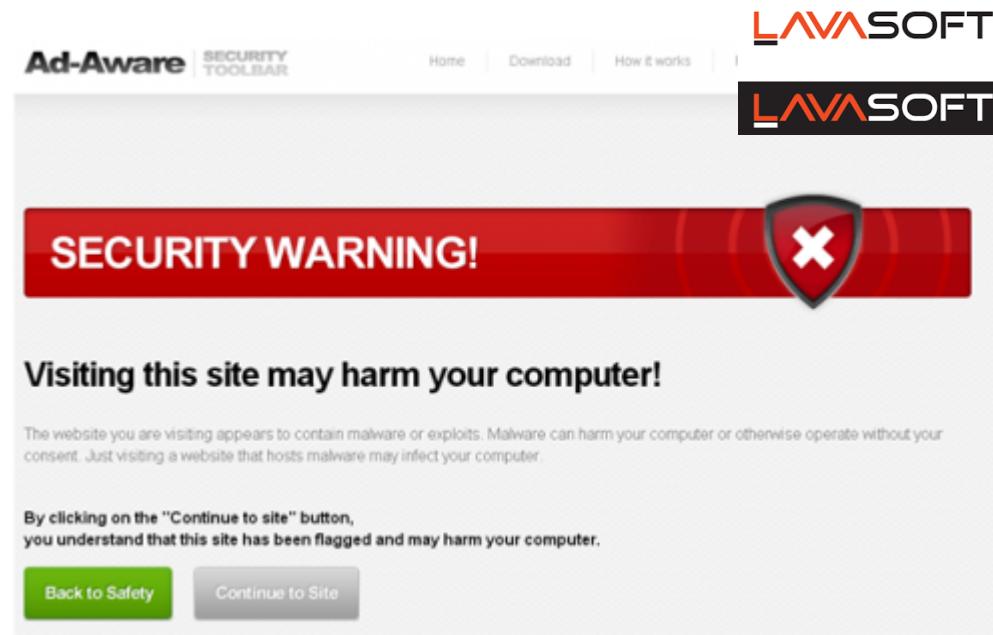
RESULTS OVER REAL DATASETS

Dataset	Single type			mixed-type			
	GMM	OCSVM	PPCA	BMM	ODMAD	GLM-t	Mv.RBM
<i>KDD99-10</i>	0.42	0.54	0.55	–	–	–	0.71
<i>Australian Credit</i>	0.74	0.84	0.38	0.972	0.942	–	0.90
<i>German Credit</i>	0.86	0.86	0.02	0.934	0.810	–	0.95
<i>Heart</i>	0.89	0.76	0.64	0.872	0.630	0.72	0.94
<i>Thoracic Surgery</i>	0.71	0.71	0.70	0.939	0.879	–	0.90
<i>Auto MPG</i>	1.00	1.00	0.67	0.625	0.575	0.64	1.00
<i>Contraceptive</i>	0.62	0.84	0.02	0.673	0.523	–	0.91
<i>Average</i>	0.75	0.79	0.43	0.84	0.73	0.68	0.91

ABNORMALITY ACROSS ABSTRACTIONS

$$\bar{r}_i(p) = \left(\sum_{l=1}^L r_{li}^p \right)^{1/p}$$





MALICIOUS URL CLASSIFICATION

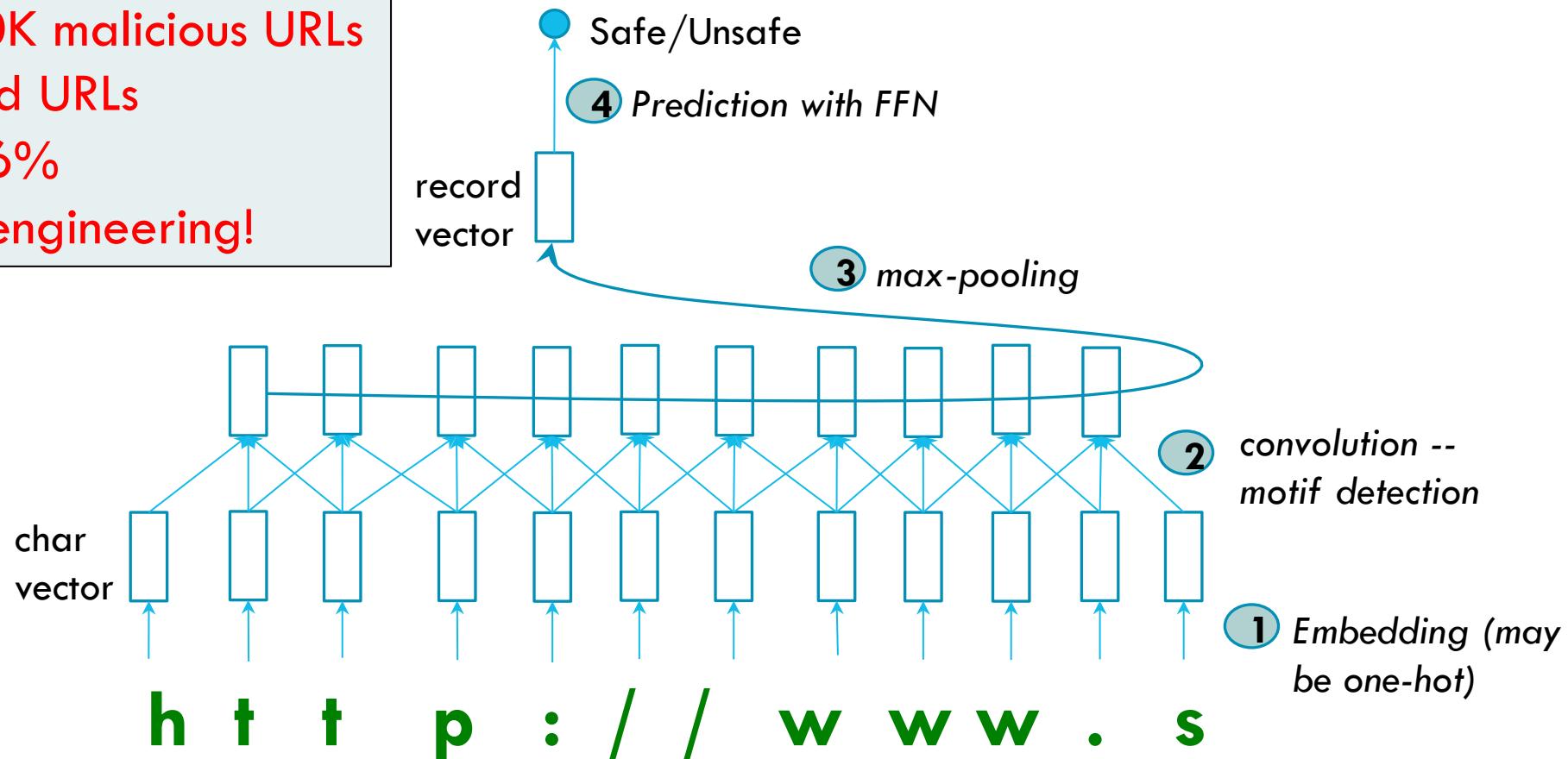
Countries with the highest number of users who clicked malicious URLs in 2015



● US	34%
● Japan	20%
● Australia	4%
● Taiwan	4%
● India	3%
● China	3%
● Germany	3%
● France	3%
● Canada	2%
● Italy	2%
● Others	22%

MODEL OF MALICIOUS URLs

Train on 900K malicious URLs
1,000K good URLs
Accuracy: 96%
No feature engineering!



SUMMARY OF PART II

Hand-on:

- Introducing programming frameworks (Theano, TensorFlow, Mxnet)

Domains how-to:

- Healthcare
- Software engineering
- Anomaly detection



