ECE209AS (Winter 2021)

Lecture 4: Data Quality Challenges and Solutions

Mani Srivastava
mbs@ucla.edu
Networked & Embedded Systems Lab
ECE & CS Departments
UCLA

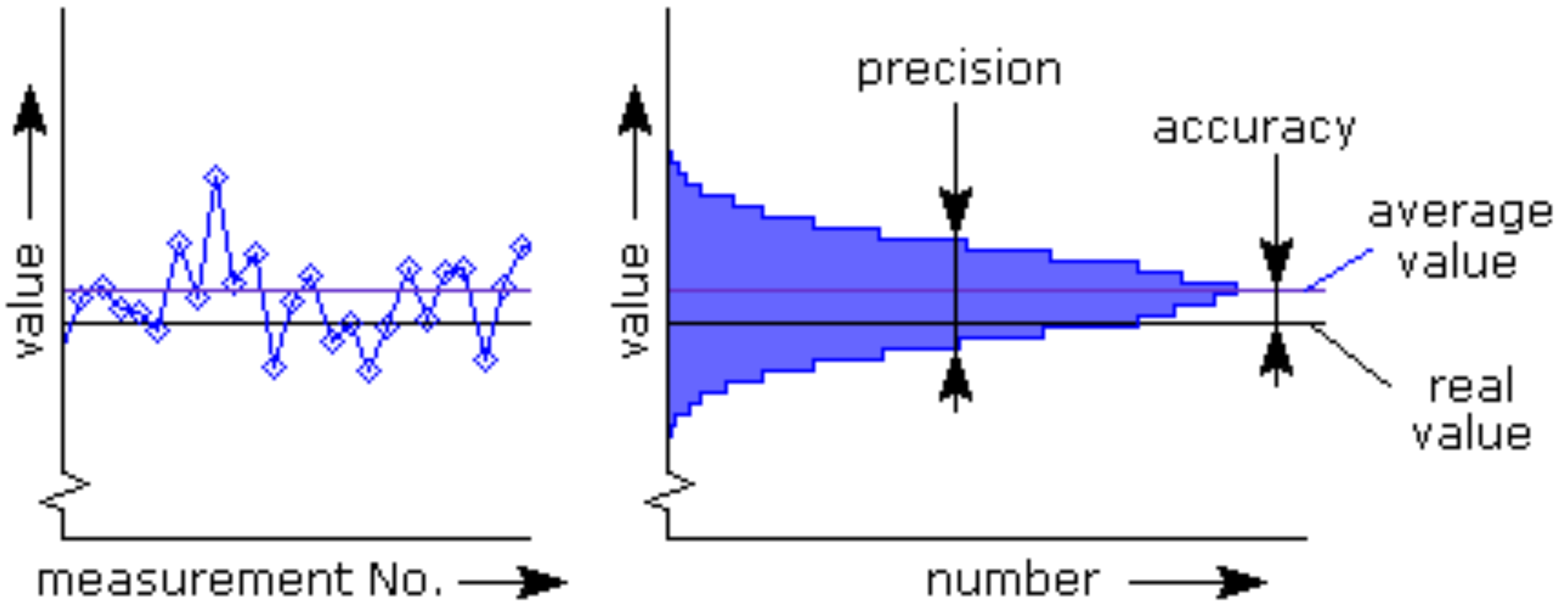
Data Quality Problems

- Measurement uncertainty
  - resolution, accuracy, precision, bias, …

Measurement Uncertainty

- Sensors, humans, and upstream ML algorithms always provide data that have a degree of uncertainty
  - systematic and random errors

- Key concepts
  - **Accuracy:** The error between the real and measured value (i.e. correctness)
  - **Precision:** The random spread of measured values around the average measured values (random error)
  - **Resolution:** The smallest to be distinguished magnitude from the measured value
  - Important Note: in classification, *accuracy* and *precision* have different meanings

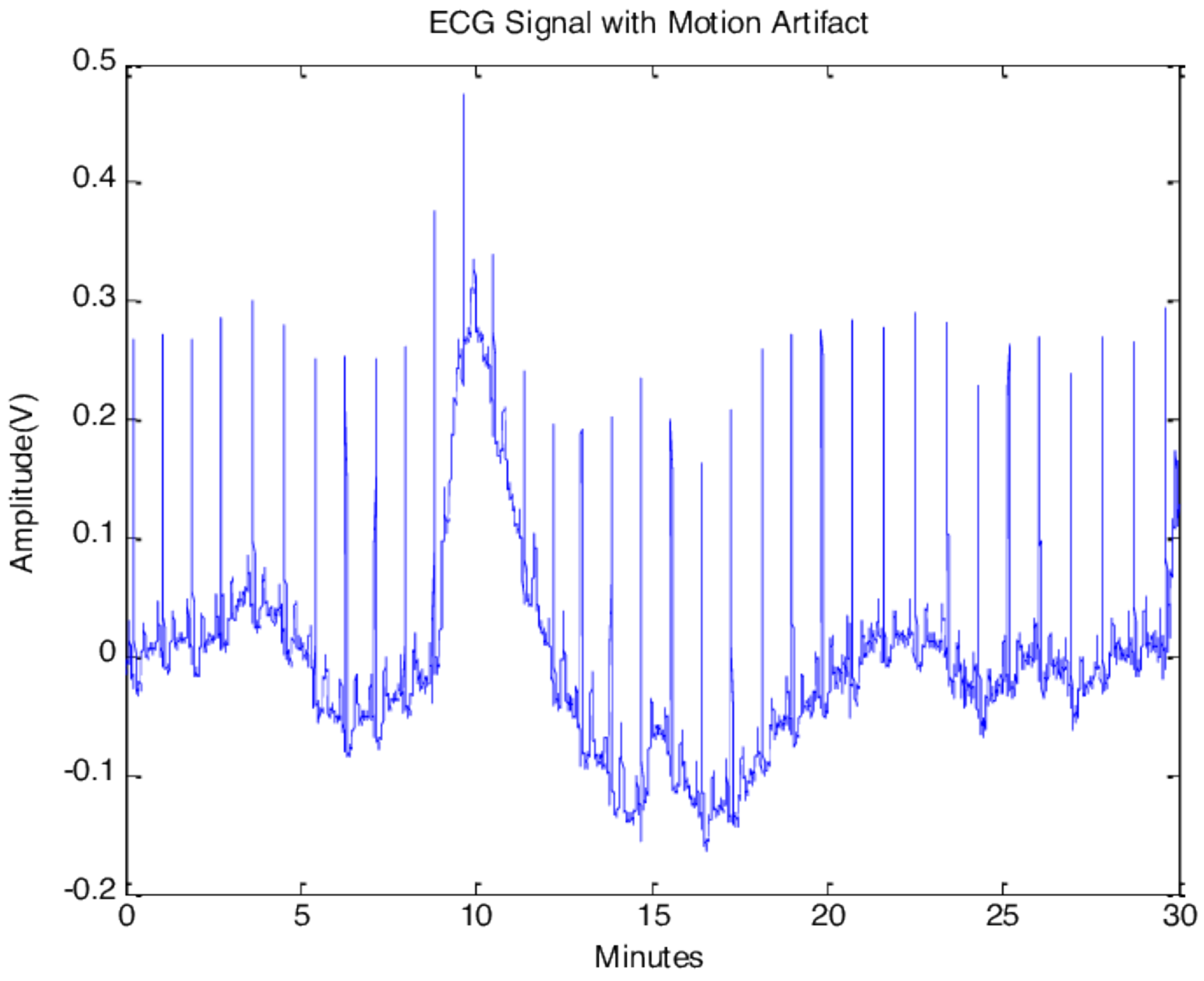


https://meettechniek.info/measurement/accuracy.html

Data Quality Problems

- Measurement uncertainty
  - resolution, accuracy, precision, bias, …

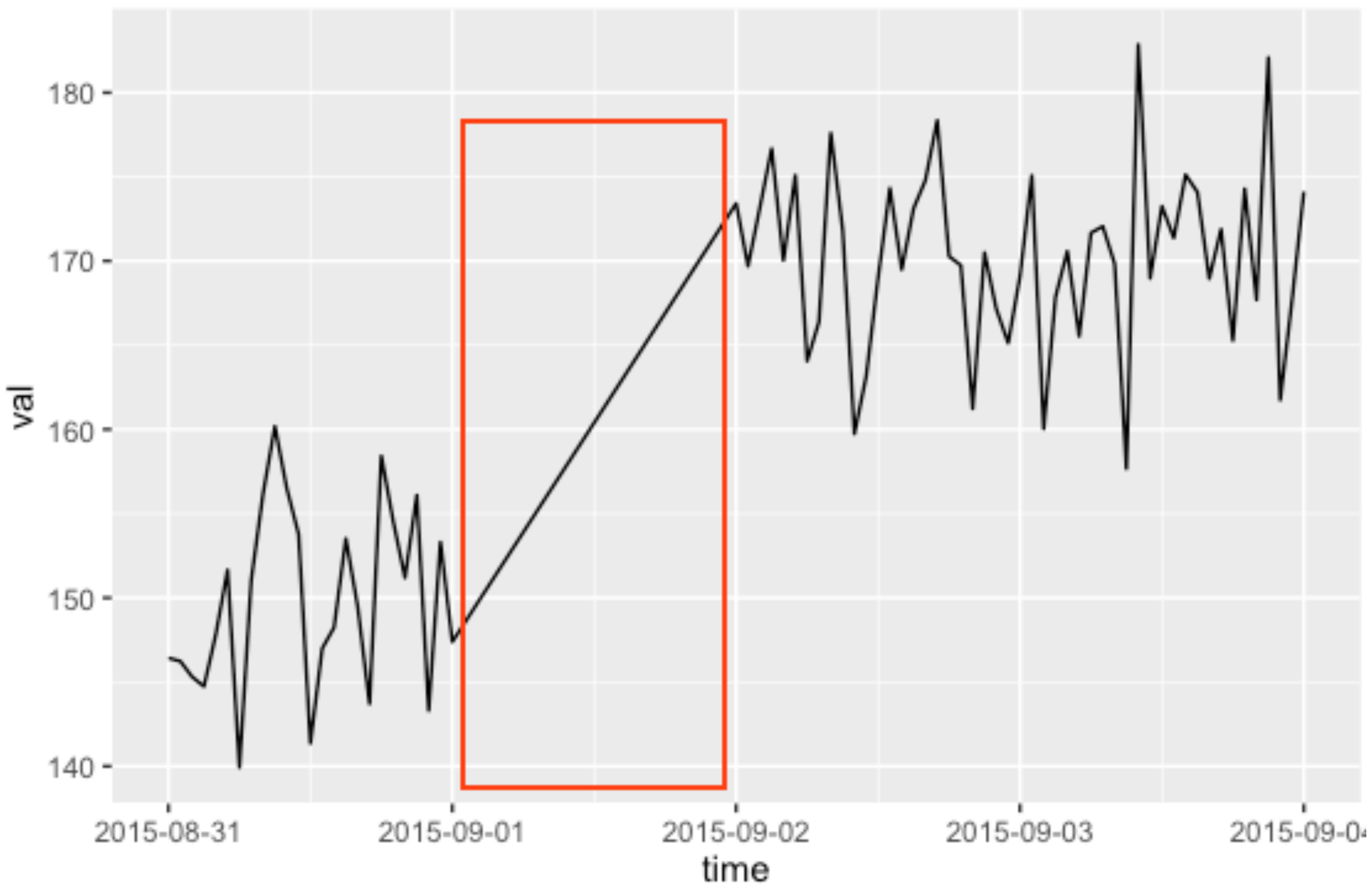
- Environmental factors
  - temperature, humidity, biofouling, light, dust, wind, rain, motion, …



https://www.researchgate.net/figure/ECG-signal-with-motion-artifact_fig1_264547500

Data Quality Problems

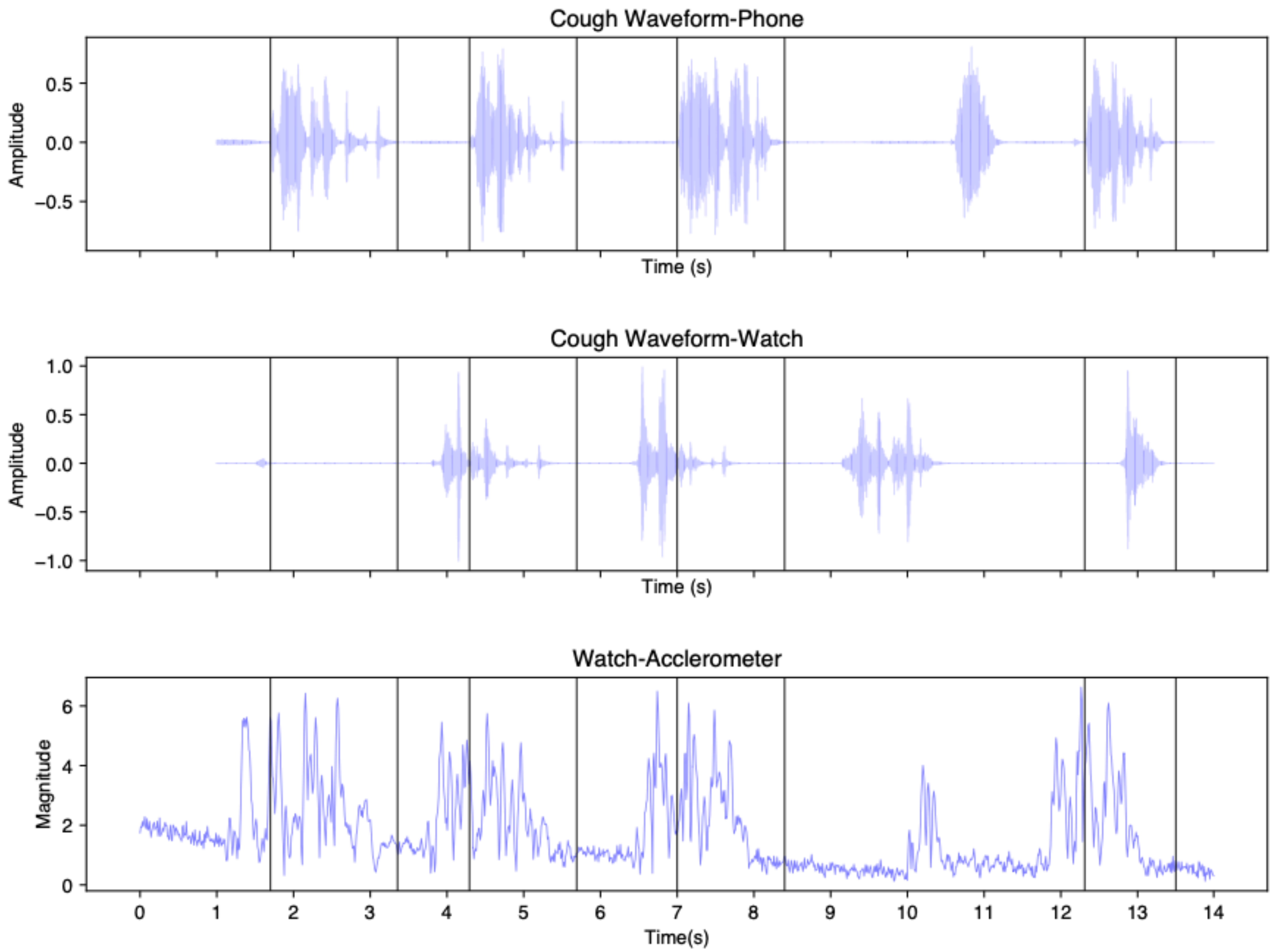
- Measurement uncertainty
  - resolution, accuracy, precision, bias, …
- Environmental factors
  - temperature, humidity, biofouling, light, dust, wind, rain, motion, …
- Missing values
  - gaps in time series, missing sensors



https://stackoverflow.com/questions/40087340/differentiate-missing-values-from-main-data-in-a-plot-using-r

Data Quality Problems

- Measurement uncertainty
  - resolution, accuracy, precision, bias, …
- Environmental factors
  - temperature, humidity, biofouling, light, dust, wind, rain, motion, …
- Missing values
  - gaps in time series, missing sensors
- Metadata may also have quality issues
  - timestamp
  - location



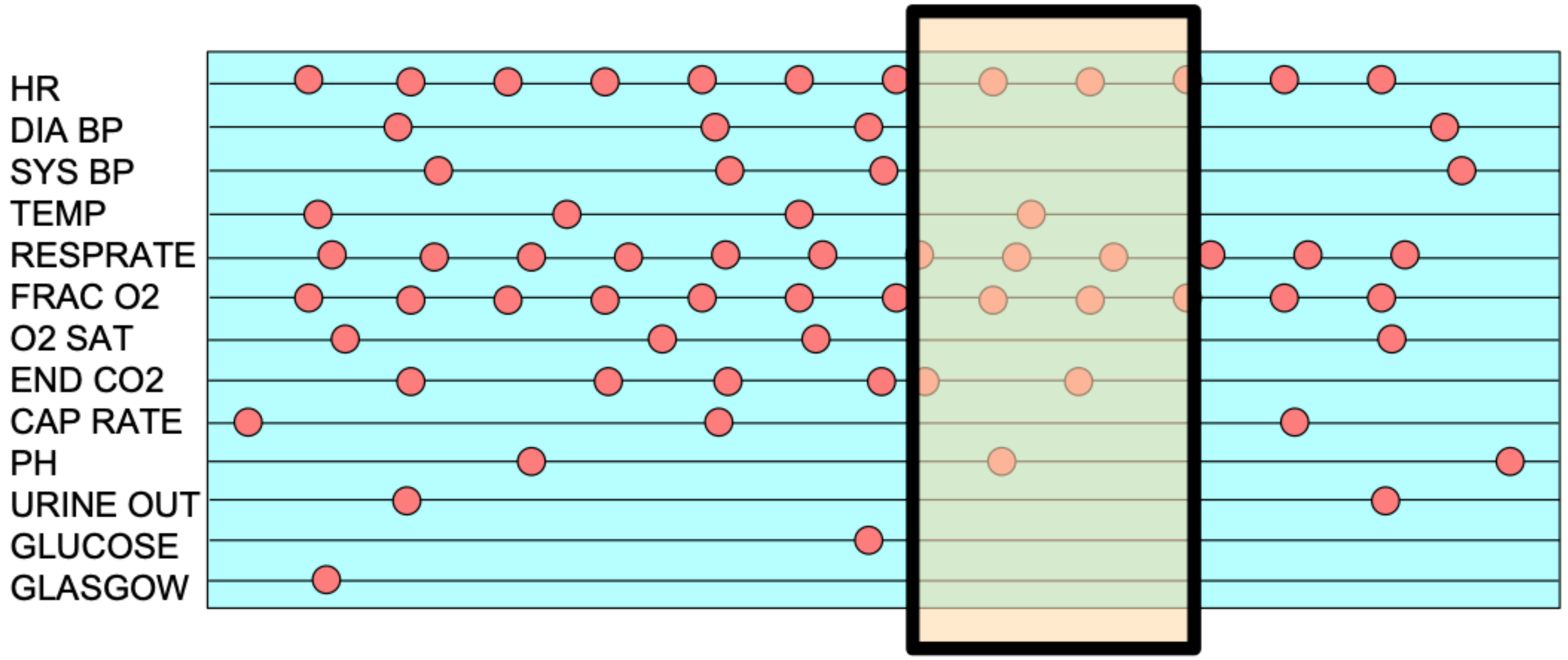
https://dl.acm.org/doi/pdf/10.1145/3382507.3418855

Missing Values

Many reasons for missing values in real-world applications

- Irregular observations
- Software crash
- Communication outage
- Energy availability
- Power management
- Privacy and other human factors

… and more …



http://proceedings.mlr.press/v56/Lipton16.pdf

Missingness artifacts created when irregular observations observations organized as a sequence with discrete, fixed-width time steps

Informative Missingness

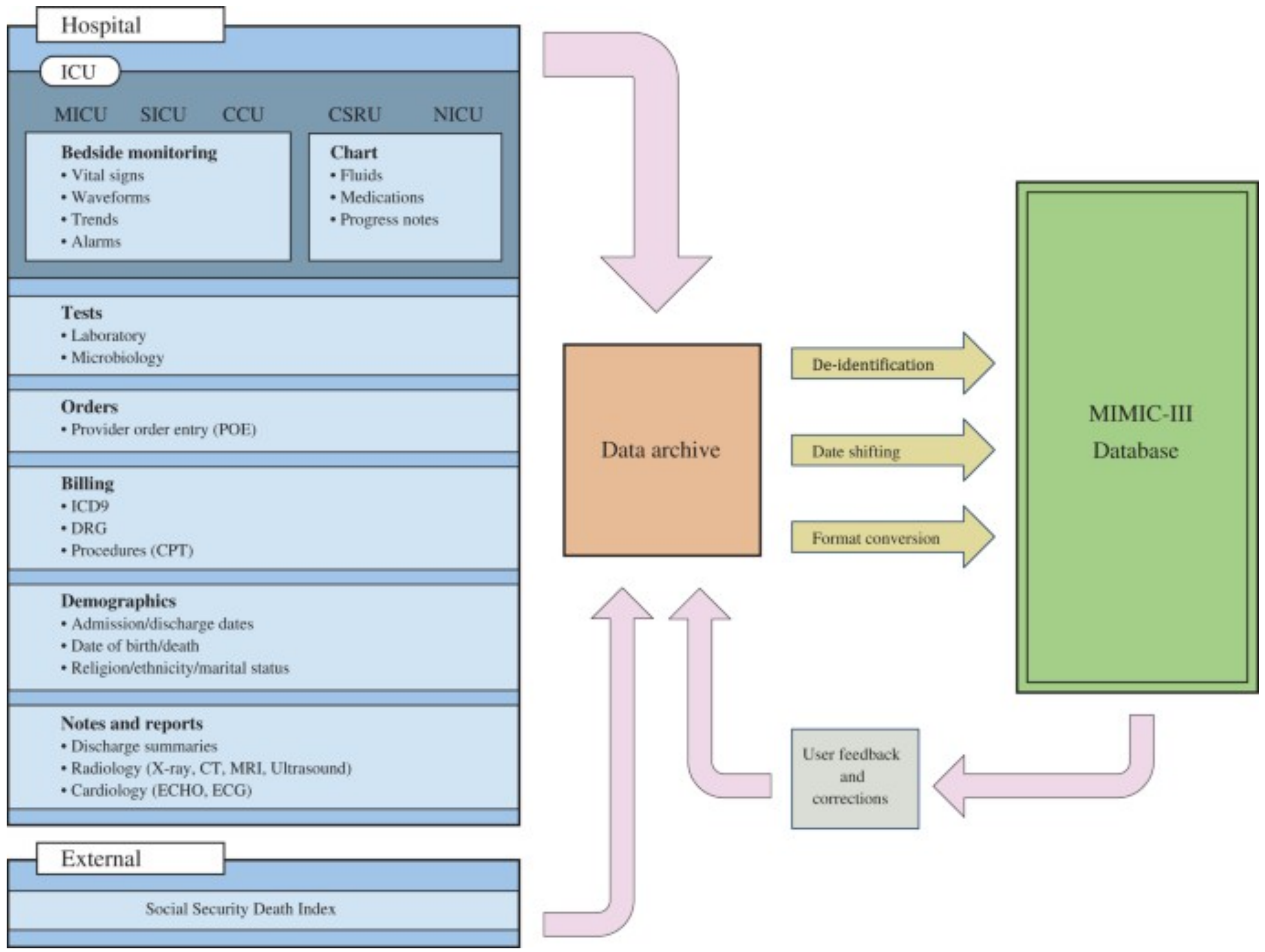
- Missing values and patterns often provide rich information about target labels in supervised learning tasks, e.g, time series classification
  - Because they may not occur randomly, e.g. may reflect clinical care decisions

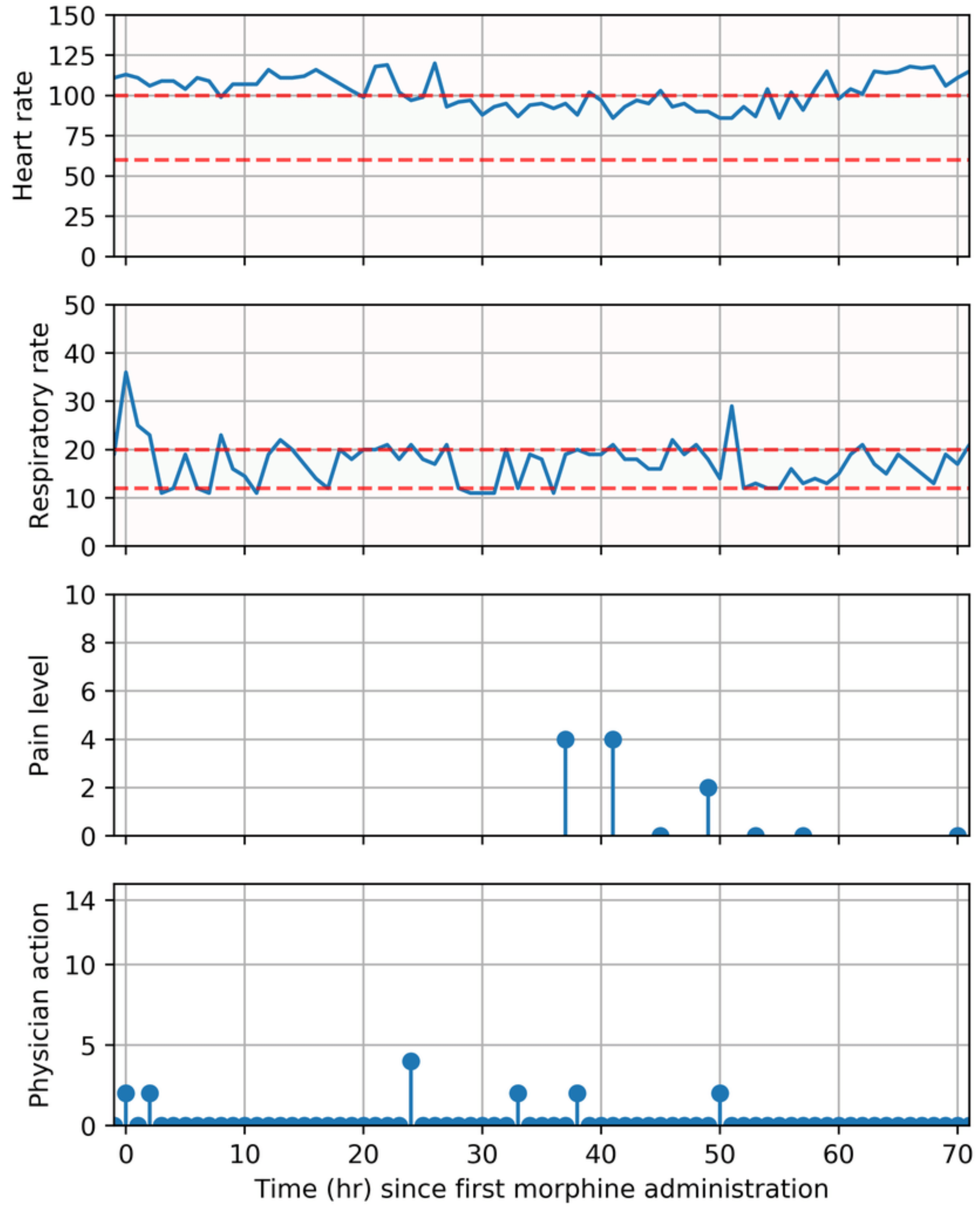
- Consider MIMIC III…

https://www.nature.com/articles/s41598-018-24271-9

MIMIC-IIII

- A freely accessible critical care database from MIT Lab for Computational Physiology
  - https://mimic.physionet.org

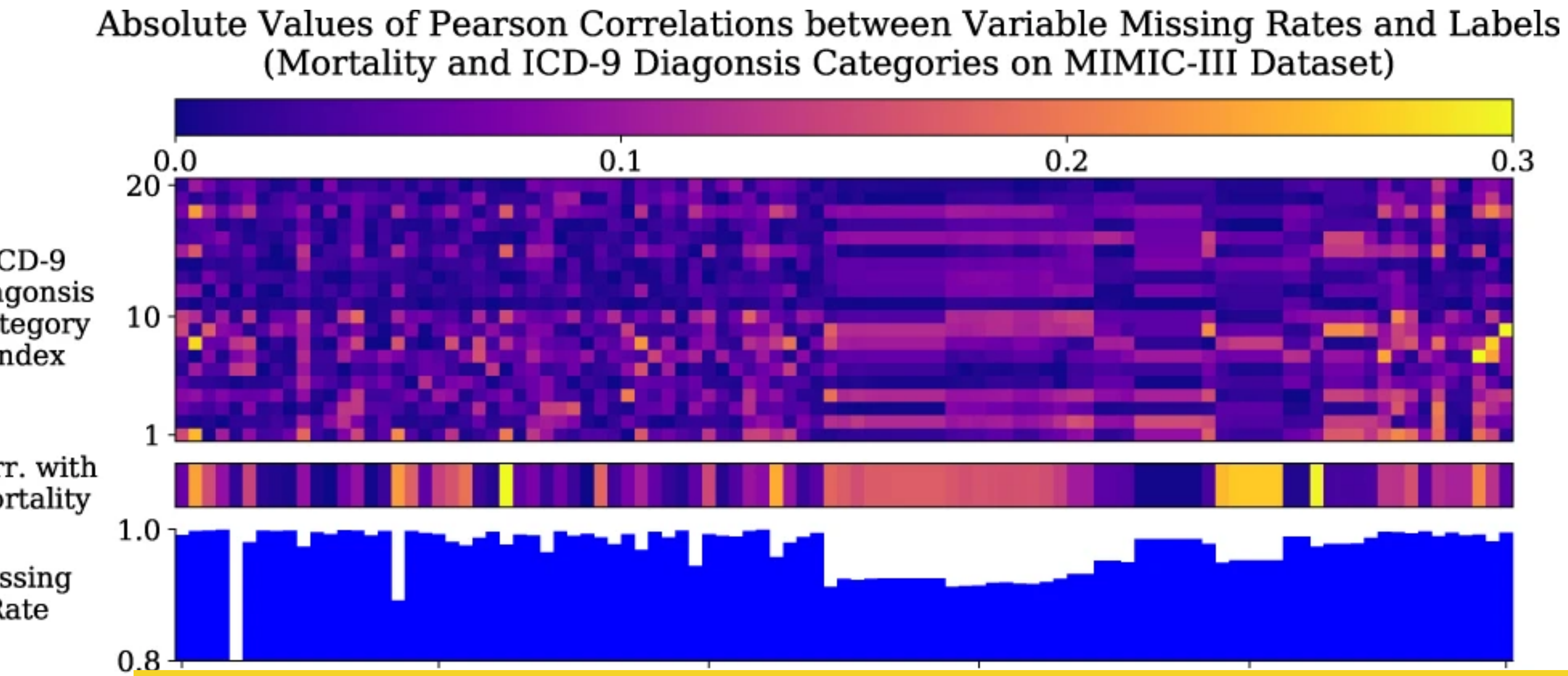




https://www.researchgate.net/publication/332669683_Deep_Reinforcement_Learning_for_Optimal_Critical_Care_Pain_Management_with_Morphine_using_Dueling_Double-Deep_Q_Networks

Informative Missingness

- Missing values and patterns often provide rich information about target labels in supervised learning tasks, e.g, time series classification



Demonstration of informative missingness on MIMIC-III dataset.

The bottom figure shows the missing rate of each input variable. The middle figure shows the absolute values of Pearson correlation coefficients between missing rate of each variable and mortality. The top figure shows the absolute values of Pearson correlation coefficients between missing rate of each variable and each ICD-9 diagnosis category.

Observation: The value of missing rate is correlated with the labels, and the missing rate of variables with low missing rate are usually highly (either positive or negative) correlated with the labels. I.e., the missing rate of variables for each patient is useful, and this information is more useful for the variables which are observed more often in the dataset.

https://www.nature.com/articles/s41598-018-24271-9

Traditional Approaches for Missing Data

- Omit the missing data and perform analysis only on the observed data
  - Does not provide good performance when the missing rate is high and inadequate samples are kept
  - Does not work if processing algorithm cannot handle missing data

- Fill in the missing values with substituted values: known as *data imputation*
  - Simple approaches: default value (e.g. 0), mean, median, mode, smoothing, interpolation, spline, …
    - Do not capture variable correlations
    - May not capture complex pattern when performing imputation
  - Better imputation methods
    - Spectral analysis, kernel methods, EM algorithm, KNN, matrix completion, matrix factorization, …

Limitations of Traditional Approaches

- Two-step process of imputation followed by prediction can be suboptimal
  - missing patterns are not effectively explored in the prediction model
- Most imputation methods also have other requirements which may not be satisfied in real applications
  - many of them work on data with small missing rates only
  - assume the data is missing at random or completely at random
  - or can not handle time series data with varying lengths
- Training and applying imputation methods are often computationally expensive

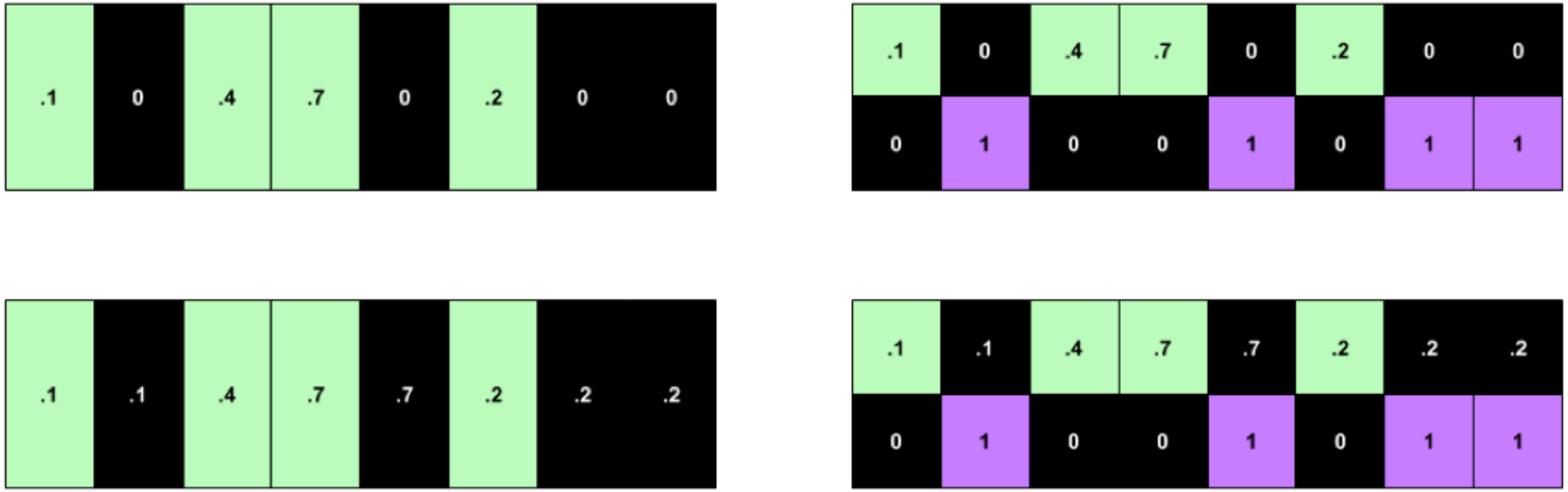
Strategies for RNN with Missing Data in Clinical Time Series [Lipton16]

- Zero-imputation strategy
  - If $x_i^{(t)}$ is missing then set $x_i^{(t)} = 0$
- Forward-filling strategy
  - If there is at least one previously recorded measurement of variable i at a time $t' < t$, perform forward-filling by setting $x_i^{(t)} = x_i^{(t')}$
  - If there is no previous recorded measurement (or if the variable is missing entirely), then impute the median estimated over all measurements in the training data.
    - motivated by the intuition that clinical staff record measurements at intervals proportional to rate at which they are believed or observed to change
- Indicator (mask) variable approach
  - Augment inputs with binary variables $m_i^{(t)}$ for every $x_i^{(t)}$, where $m_i^{(t)} = 1$ if $x_i^{(t)}$ is imputed and 0 otherwise
  - Through their hidden state computations, RNNs can use the indicators to learn arbitrary functions of the past observations and missingness pattern

http://proceedings.mlr.press/v56/Lipton16.pdf

Strategies for RNN with Missing Data in Clinical Time Series [Lipton16]

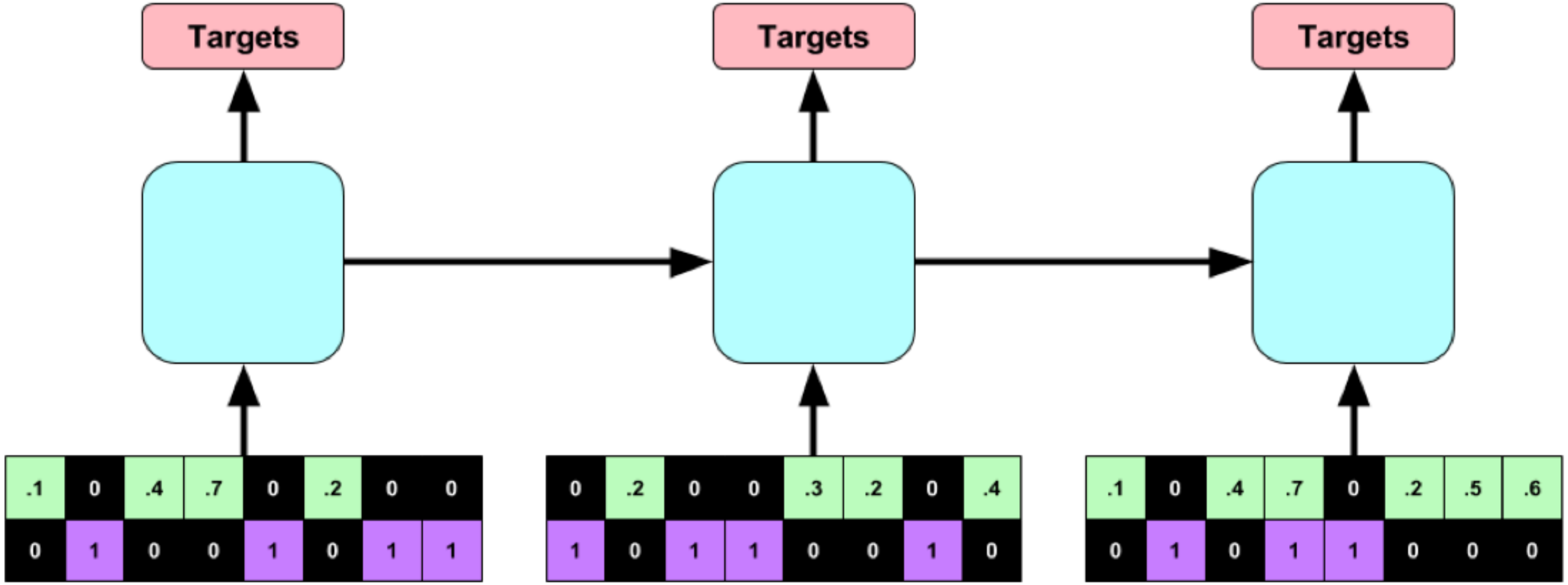
- Zero-imputation strategy
- Forward-filling strategy
- Indicator variable approach



(top left) no imputation or indicators, (bottom left) imputation absent indicators, (top right) indicators but no imputation, (bottom right) indicators and imputation. Time flows from left to right.

http://proceedings.mlr.press/v56/Lipton16.pdf

RNN with zero-filled inputs and missing data indicators



A Better Representation of Informative Missingness Patterns: Marking with Time Intervals

- Definitions
  - $X = (x_1, x_2, \ldots, x_T)^T \ \in \mathbb{R}^{T \times D}$ is a multivariate time series with D variables of length T
    - for each $t \in \{1,2,\ldots,T\}$, $x_t \in \mathbb{R}^D$ represents the t-th measurement of all variables while x_t^d denotes the measurement of the d-th variable of x_t
  - $s_t \in \mathbb{R}$ denotes the time-stamp when the t-th observation is obtained
    - we assume that the first observation is made at time-stamp 0 (i.e., $s_1 = 0$)
- Approach: indicate which variables are missing and how long they have been missing
  - A *masking vector* $m_t \in \{0,1\}^D$ denotes which variables are missing at time step t
  - Also maintain the *time interval* $\delta_t^d \in \mathbb{R}$ for each variable d since its last observation

$$\boldsymbol{m}_t^d = \begin{cases} 1, & \text{if } \boldsymbol{x}_t^d \text{ is observed} \\ 0, & \text{otherwise} \end{cases}$$

$$\delta_t^d = \begin{cases} s_t - s_{t-1} + \delta_{t-1}^d, & t > 1, m_{t-1}^d = 0 \\ s_t - s_{t-1}, & t > 1, m_{t-1}^d = 1 \\ 0, & t = 1 \end{cases}$$

$\boldsymbol{X}$: Input time series (2 variables);
$\boldsymbol{s}$: Timestamps for $\boldsymbol{X}$;

$\boldsymbol{M}$: Masking for $\boldsymbol{X}$;
$\boldsymbol{\Delta}$: Time interval for $\boldsymbol{X}$.

$$\boldsymbol{X} = \begin{bmatrix} 47 & 49 & NA & 40 & NA & 43 & 55 \\ NA & 15 & 14 & NA & NA & NA & 15 \end{bmatrix}$$

$$\boldsymbol{s} = \begin{bmatrix} 0 & 0.1 & 0.6 & 1.6 & 2.2 & 2.5 & 3.1 \end{bmatrix}$$

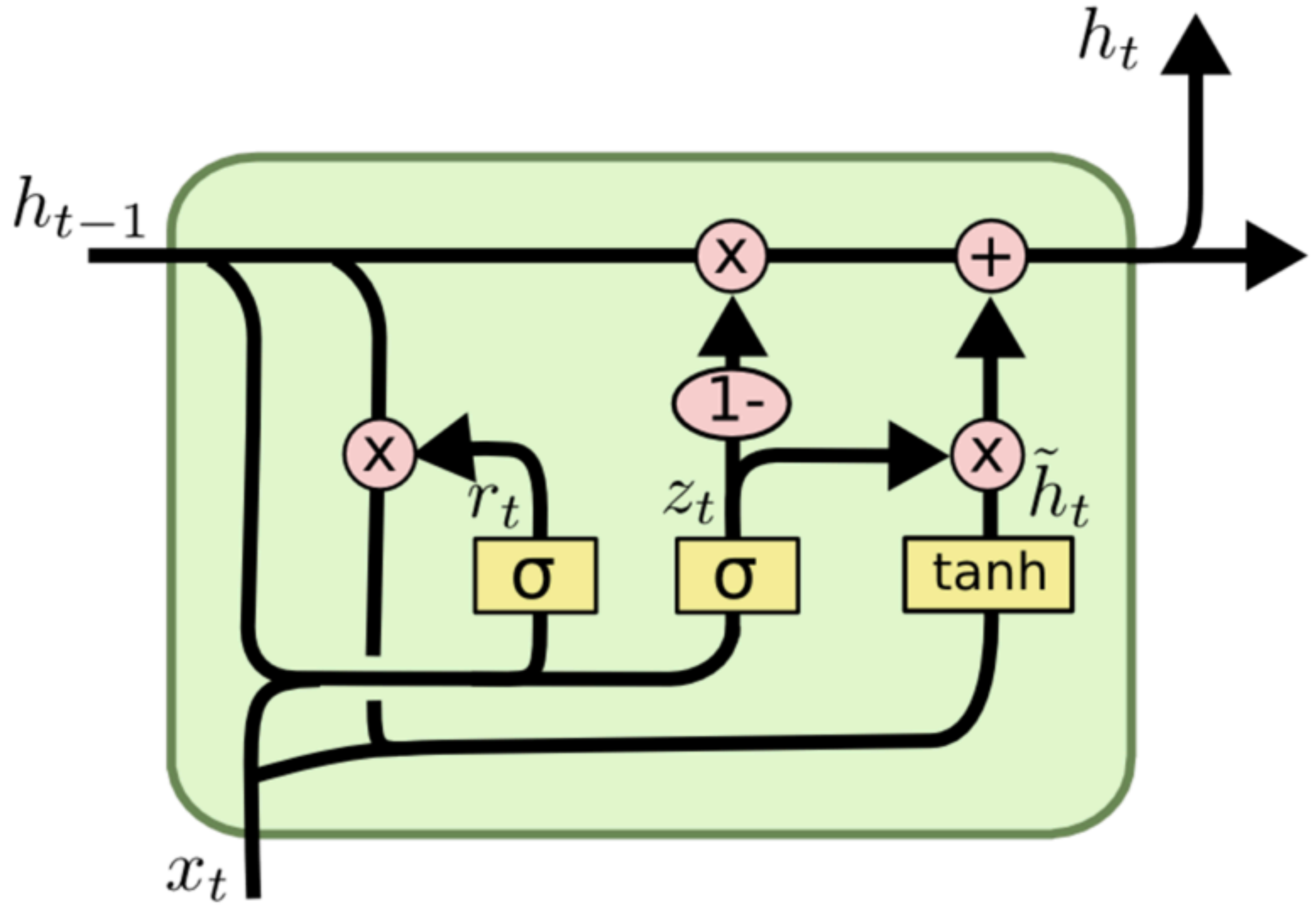
$$\boldsymbol{M} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\boldsymbol{\Delta} = \begin{bmatrix} 0.0 & 0.1 & 0.5 & 1.5 & 0.6 & 0.9 & 0.6 \\ 0.0 & 0.1 & 0.5 & 1.0 & 1.6 & 1.9 & 2.5 \end{bmatrix}$$

https://www.nature.com/articles/s41598-018-24271-9

Recall: GRU

(a) Long Short-Term Memory



(b) Gated Recurrent Unit

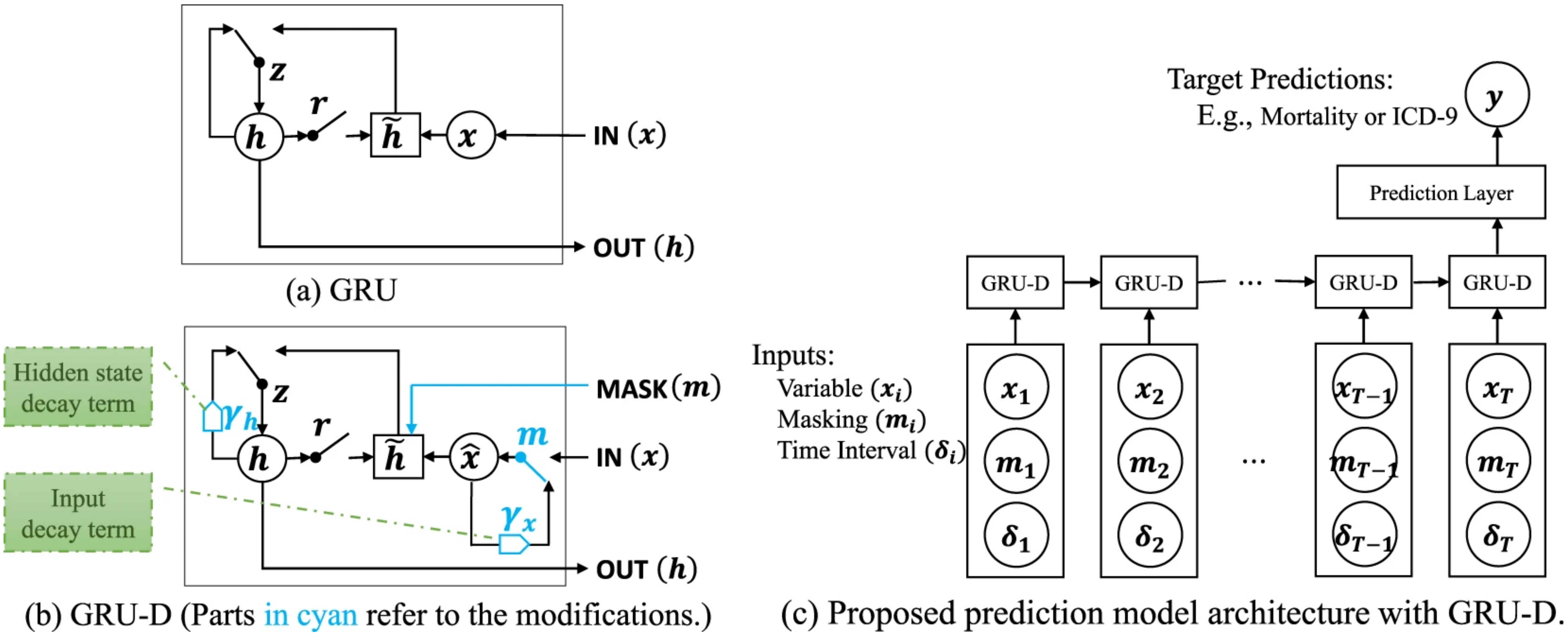
$$r_t = \sigma(W_r x_t + U_r h_{t-1} + b_r)$$

$$z_t = \sigma(W_z x_t + U_z h_{t-1} + b_z)$$

$$\tilde{h}_t = \tanh(W x_t + U(r_t \odot h_{t-1}) + b)$$

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \tilde{h}_t$$

RNN with modified GRUs containing trainable decay term



https://www.nature.com/articles/s41598-018-24271-9

Model performance for mortality prediction

Non-RNN Models						**RNN Models**	
Mortality Prediction On MIMIC-III Dataset						LSTM-Mean	0.8142 ± 0.014
LR-Mean	0.7589 ± 0.015	SVM-Mean	0.7908 ± 0.006	RF-Mean	0.8293 ± 0.004	GRU-Mean	0.8252 ± 0.011
LR-Forward	0.7792 ± 0.018	SVM-Forward	0.8010 ± 0.004	RF-Forward	0.8303 ± 0.003	GRU-Forward	0.8192 ± 0.013
LR-Simple	0.7715 ± 0.015	SVM-Simple	0.8146 ± 0.008	RF-Simple	0.8294 ± 0.007	GRU-Simple w/o δ[22]	0.8367 ± 0.009
LR-SoftImpute	0.7598 ± 0.017	SVM-SoftImpute	0.7540 ± 0.012	RF-SoftImpute	0.7855 ± 0.011	GRU-Simple w/o m[23,24]	0.8266 ± 0.009
LR-KNN	0.6877 ± 0.011	SVM-KNN	0.7200 ± 0.004	RF-KNN	0.7135 ± 0.015	GRU-Simple	0.8380 ± 0.008
LR-CubicSpline	0.7270 ± 0.005	SVM-CubicSpline	0.6376 ± 0.018	RF-CubicSpline	0.8339 ± 0.007	GRU-CubicSpline	0.8180 ± 0.011
LR-MICE	0.6965 ± 0.019	SVM-MICE	0.7169 ± 0.012	RF-MICE	0.7159 ± 0.005	GRU-MICE	0.7527 ± 0.015
LR-MF	0.7158 ± 0.018	SVM-MF	0.7266 ± 0.017	RF-MF	0.7234 ± 0.011	GRU-MF	0.7843 ± 0.012
LR-PCA	0.7246 ± 0.014	SVM-PCA	0.7235 ± 0.012	RF-PCA	0.7747 ± 0.009	GRU-PCA	0.8236 ± 0.007
LR-MissForest	0.7279 ± 0.016	SVM-MissForest	0.7482 ± 0.016	RF-MissForest	0.7858 ± 0.010	GRU-MissForest	0.8239 ± 0.006
						Proposed GRU-D	**0.8527 ± 0.003**
Mortality Prediction On PhysioNet Dataset						LSTM-Mean	0.8025 ± 0.013
LR-Mean	0.7423 ± 0.011	SVM-Mean	0.8131 ± 0.018	RF-Mean	0.8183 ± 0.015	GRU-Mean	0.8162 ± 0.014
LR-Forward	0.7479 ± 0.012	SVM-Forward	0.8140 ± 0.018	RF-Forward	0.8219 ± 0.017	GRU-Forward	0.8195 ± 0.004
LR-Simple	0.7625 ± 0.004	SVM-Simple	0.8277 ± 0.012	RF-Simple	0.8157 ± 0.014	GRU-Simple	0.8226 ± 0.010
LR-SoftImpute	0.7386 ± 0.007	SVM-SoftImpute	0.8057 ± 0.019	RF-SoftImpute	0.8100 ± 0.016	GRU-SoftImpute	0.8125 ± 0.005
LR-KNN	0.7146 ± 0.011	SVM-KNN	0.7644 ± 0.018	RF-KNN	0.7567 ± 0.012	GRU-KNN	0.8155 ± 0.004
LR-CubicSpline	0.6913 ± 0.022	SVM-CubicSpline	0.6364 ± 0.015	RF-CubicSpline	0.8151 ± 0.015	GRU-CubicSpline	0.7596 ± 0.020
LR-MICE	0.6828 ± 0.015	SVM-MICE	0.7690 ± 0.016	RF-MICE	0.7618 ± 0.007	GRU-MICE	0.8153 ± 0.013
LR-MF	0.6513 ± 0.014	SVM-MF	0.7515 ± 0.022	RF-MF	0.7355 ± 0.022	GRU-MF	0.7904 ± 0.012
LR-PCA	0.6890 ± 0.019	SVM-PCA	0.7741 ± 0.014	RF-PCA	0.7561 ± 0.025	GRU-PCA	0.8116 ± 0.007
LR-MissForest	0.7010 ± 0.018	SVM-MissForest	0.7779 ± 0.008	RF-MissForest	0.7890 ± 0.016	GRU-MissForest	0.8244 ± 0.012
						Proposed GRU-D	**0.8424 ± 0.012**

(average AUC score)

https://www.nature.com/articles/s41598-018-24271-9

Model performance for multi-task predictions

Models	ICD-9 20 Tasks on MIMIC-III Dataset	All 4 Tasks on PhysioNet Dataset
GRU-Mean	0.7070 ± 0.001	0.8099 ± 0.011
GRU-Forward	0.7077 ± 0.001	0.8091 ± 0.008
GRU-Simple	0.7105 ± 0.001	0.8249 ± 0.010
GRU-CubicSpline	0.6372 ± 0.005	0.7451 ± 0.011
GRU-MICE	0.6717 ± 0.005	0.7955 ± 0.003
GRU-MF	0.6805 ± 0.004	0.7727 ± 0.003
GRU-PCA	0.7040 ± 0.002	0.8042 ± 0.006
GRU-MissForest	0.7115 ± 0.003	0.8076 ± 0.009
Proposed GRU-D	**0.7123 ± 0.003**	**0.8370 ± 0.012**

(average AUC score)

https://www.nature.com/articles/s41598-018-24271-9

GRU-D Summary

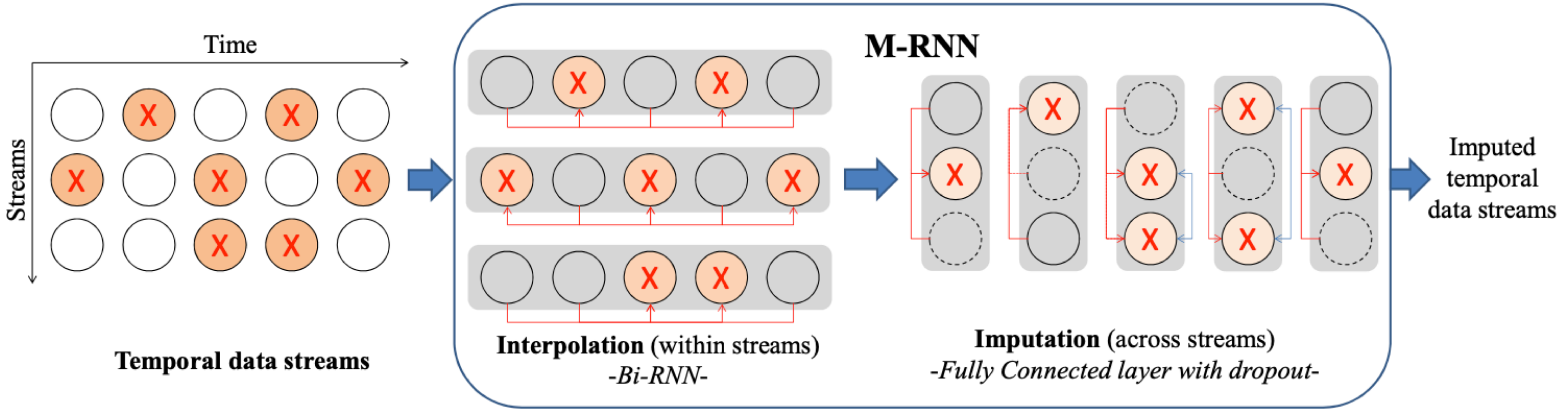
- GRU-D addresses the problem that off-the-shelf RNN architectures with imputation can only achieve performance comparable to Random Forests and SVMs, and moreover, they do not demonstrate the full advantage of representation learning

- However, GRU-D approach has limitations
  - It may fail if the missingness is not informative at all, or the inherent correlation between the missing patterns and the prediction tasks are not clear
  - Decay mechanism needs to be explicitly designed for the informative missingness present in an application domain (e.g. traffic, climate)
  - Not explicitly designed for filling in the missing values in the data, and can not be directly used in unsupervised settings without prediction labels

https://www.nature.com/articles/s41598-018-24271-9

Estimating Missing Data Using Multi-directional RNNs

- Exploits the fact that information is often correlated both within and across data streams
- Uses a hierarchical learning framework that limits the number of parameters to be learned to be linear in the number data streams



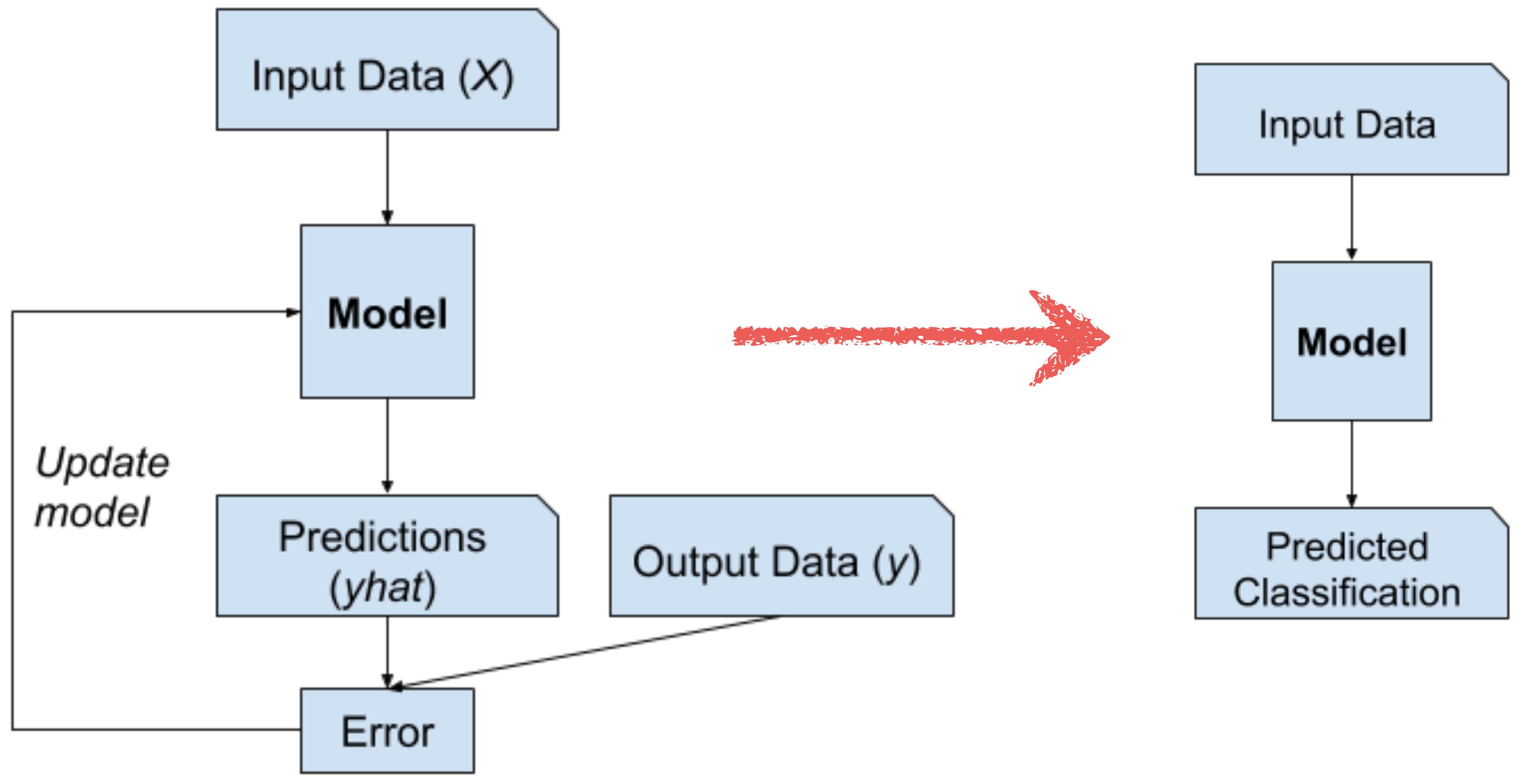
https://arxiv.org/pdf/1711.08742.pdf

Imputation via Generative Adversarial Network (GAN)

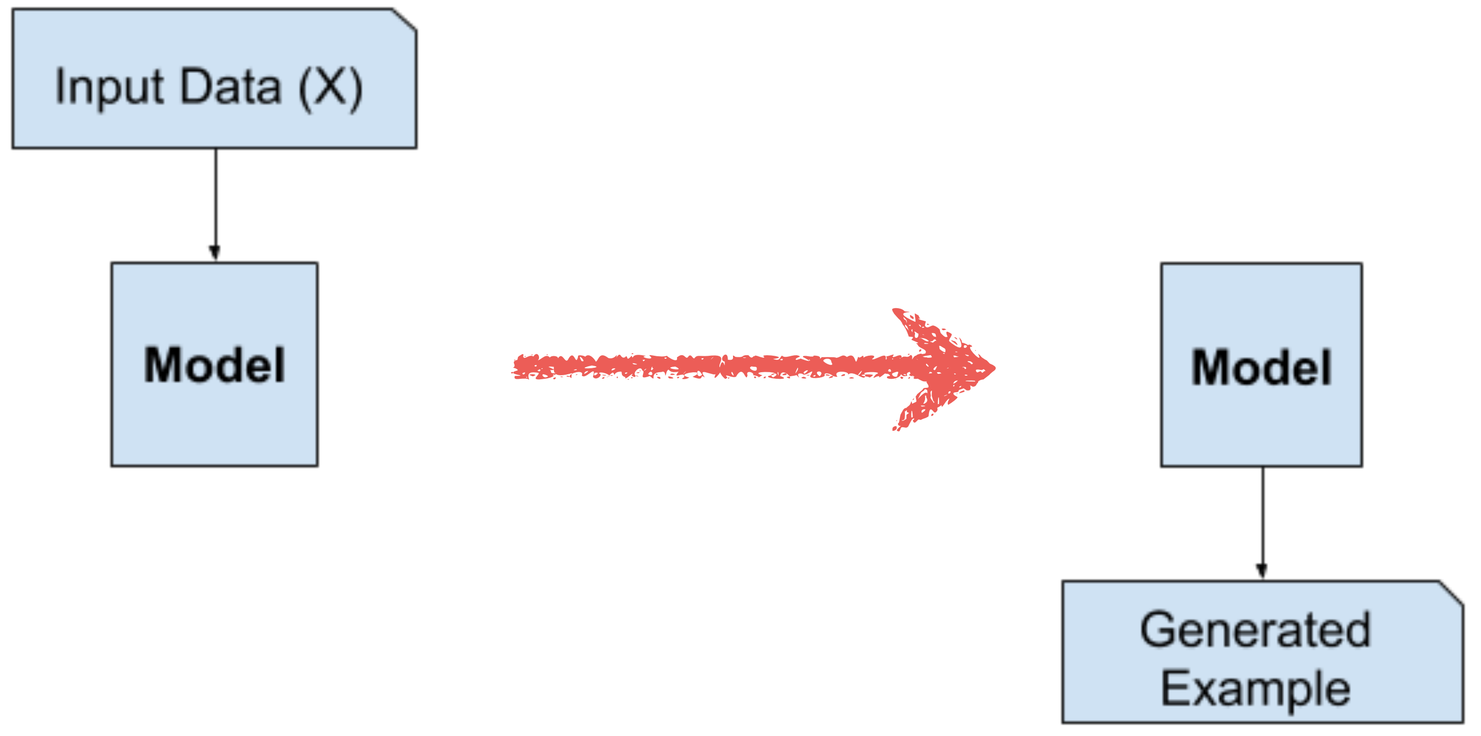
- But first what is a GAN?

Generative Adversarial Networks

- **Generative modeling** is an unsupervised learning task that involves automatically discovering and learning the regularities or patterns in input data
  - The learnt model can then be used to generate or output new examples that plausibly could have been drawn from the original dataset



Supervised Learning for Discriminative Modeling



Unsupervised Learning for Generative Modeling

https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

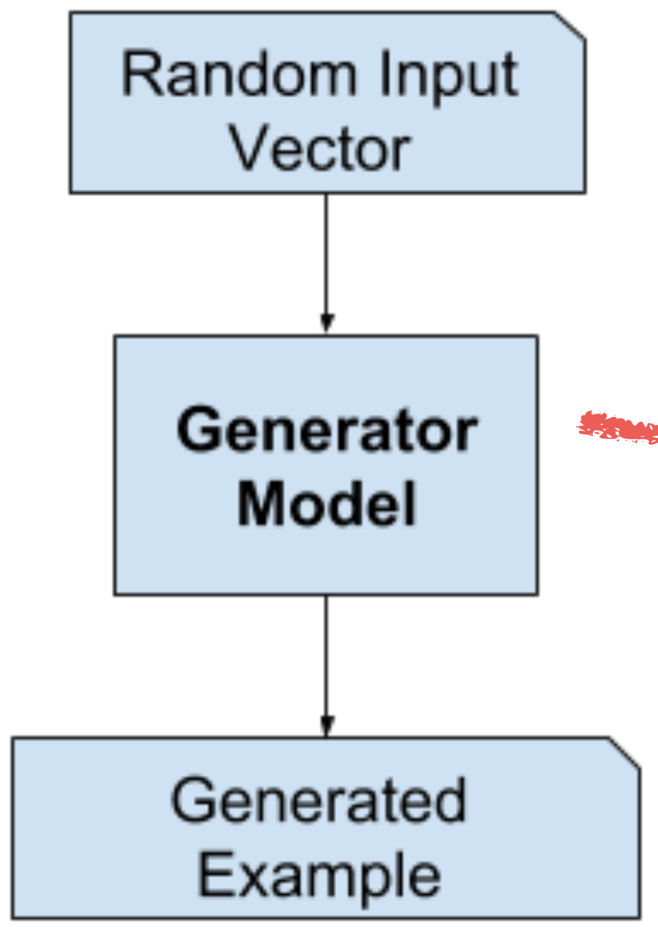
Generative Adversarial Networks (GANs)

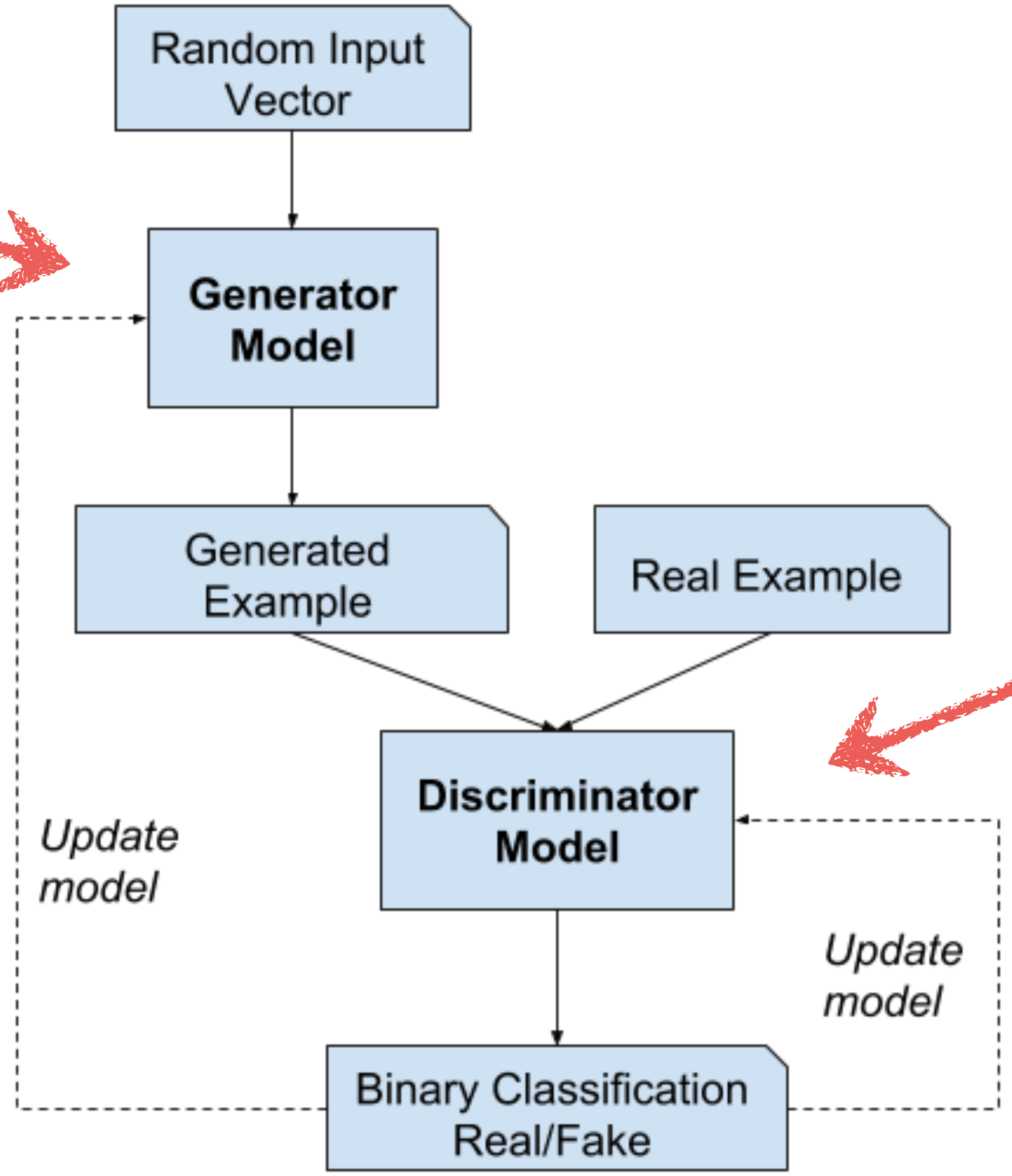
- **Generative modeling** is an unsupervised learning task that involves automatically discovering and learning the regularities or patterns in input data
  - The learnt model can then be used to generate or output new examples that plausibly could have been drawn from the original dataset

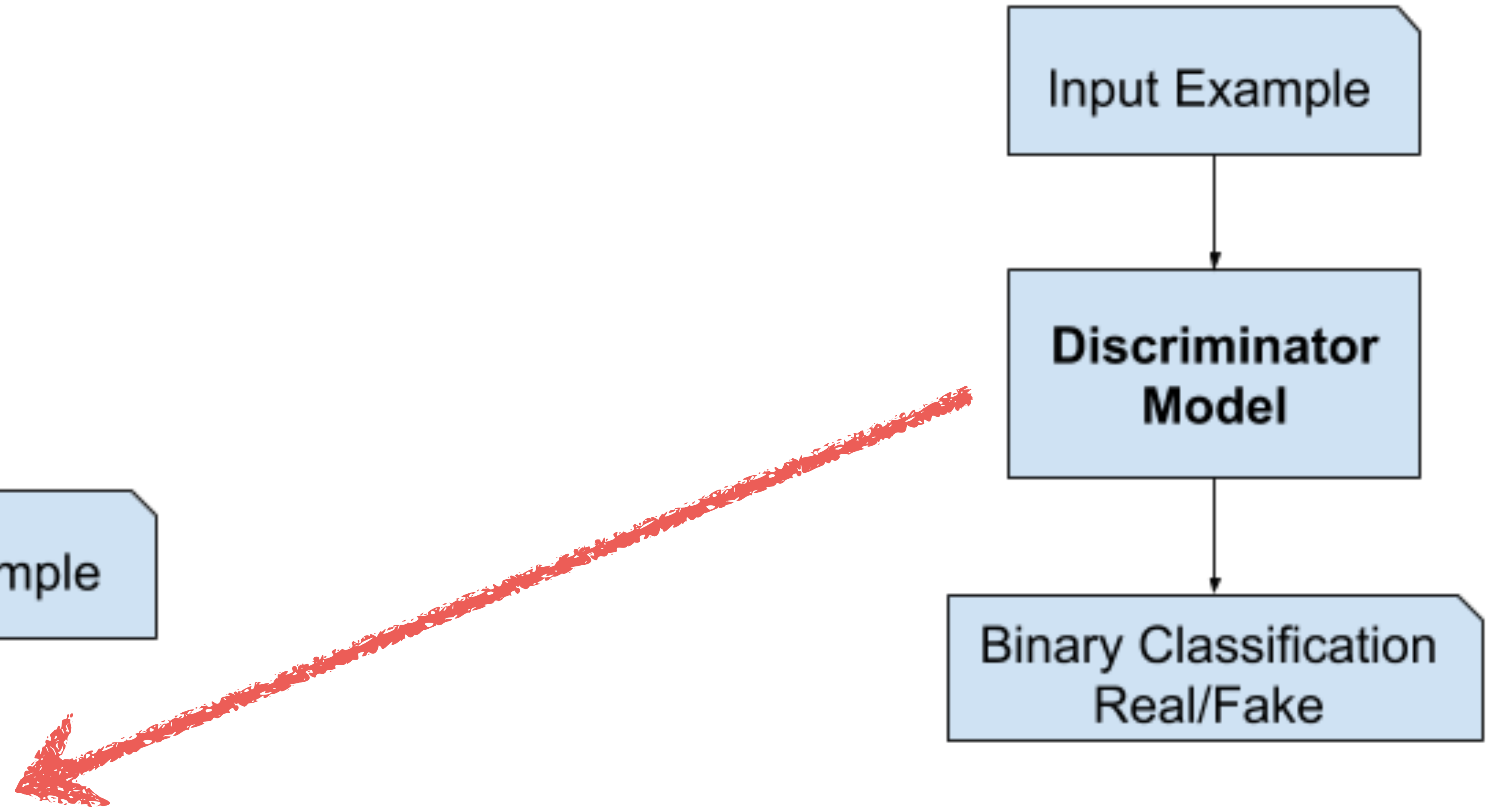
- GAN is a clever approach to generative modeling using deep learning methods
  - Frames the problem as a supervised learning problem with two sub-models: a *generator* model that we train to generate new examples, and a *discriminator* model that tries to classify examples as either real (from the domain) or fake (generated)
  - The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples

https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

Generative Adversarial Networks (GANs)







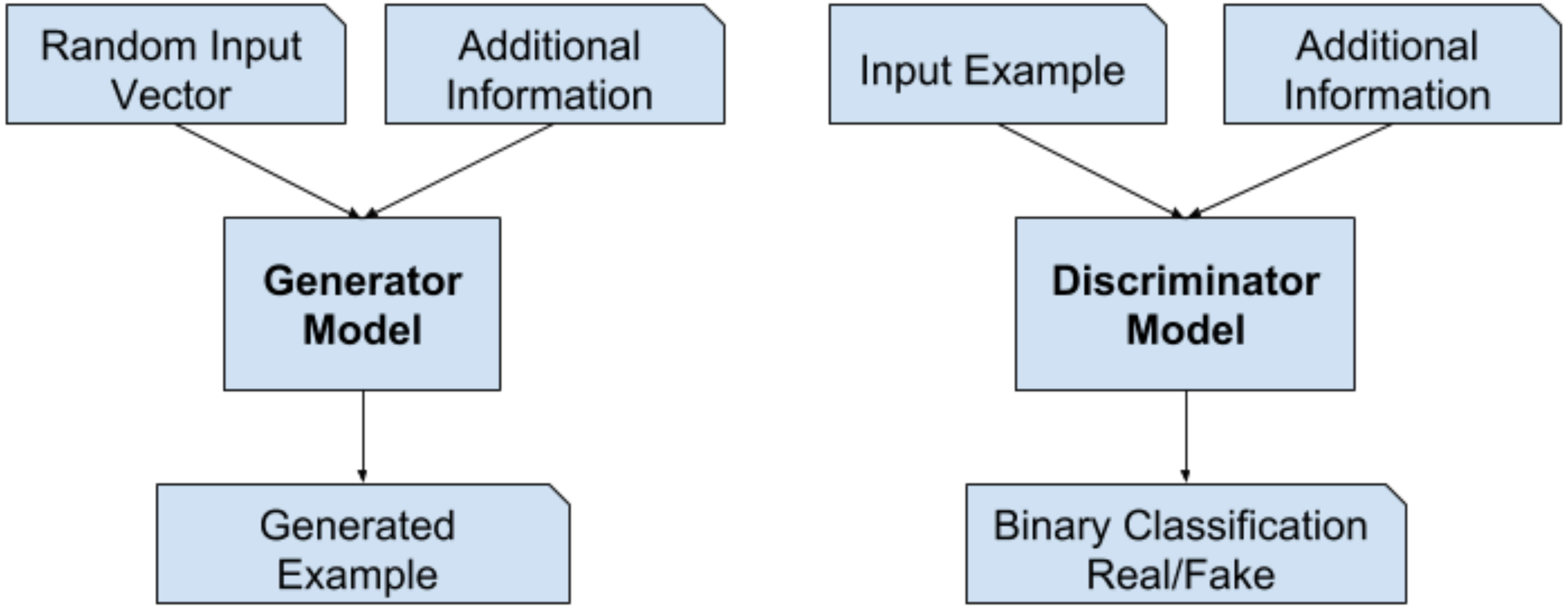
Modeled as a 2-party zero sum game via a Minimax optimization strategy.

Discriminator seeks to maximize the probability of assigning the correct label to both training examples and samples from G.

The generator seeks to minimize the log of the inverse probability predicted by the discriminator for fake images. This has the effect of encouraging the generator to generate samples that have a low probability of being fake.

https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

Enhancement: Conditional GANs



- Additional input could be a class value
  - e.g. “walking” in generation of accelerometer time series data
  - Such a conditional GAN can be used to generate examples from a domain of a given type
- Can also be conditioned on an example from a domain
  - This can allow GANs to do style transfer, domain translation etc.

Generative Adversarial Networks (GANs)

- **Generative modeling** is an unsupervised learning task that involves automatically discovering and learning the regularities or patterns in input data
  - The learnt model can then be used to generate or output new examples that plausibly could have been drawn from the original dataset

- GAN is a clever approach to generative modeling using deep learning methods
  - Frames the problem as a supervised learning problem with two sub-models: a *generator* model that we train to generate new examples, and a *discriminator* model that tries to classify examples as either real (from the domain) or fake (generated)
  - The two models are trained together in a zero-sum game, adversarial, until the discriminator model is fooled about half the time, meaning the generator model is generating plausible examples

- GANs very powerful to model high-dimensional data, handle missing data etc.
  - But a pain to train :-(

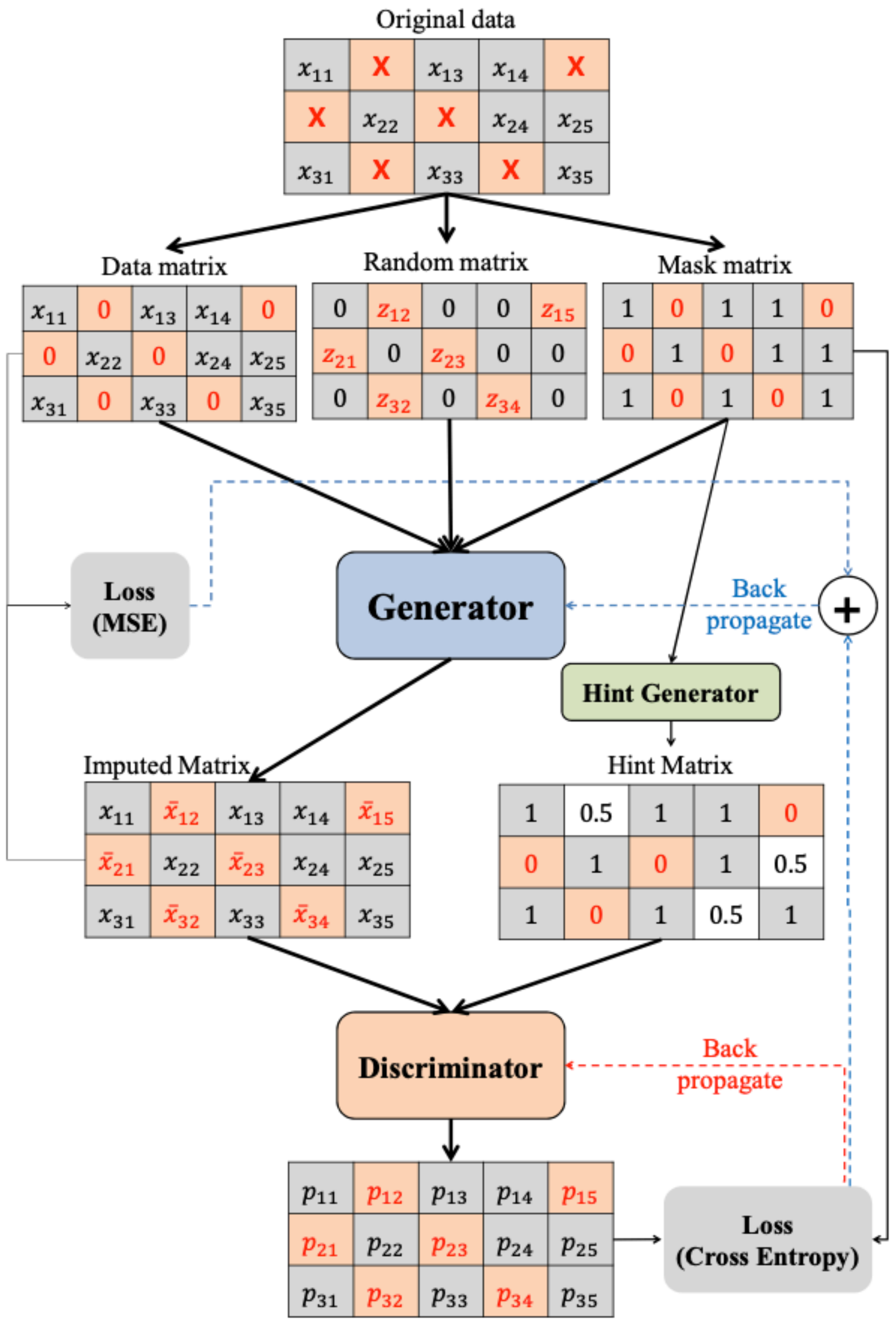
https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/

Imputation via Generative Adversarial Network (GAN)

Generative Adversarial Imputation Nets (GAIN) Architecture

In GAIN, the generator's goal is to accurately impute missing data, and the discriminator's goal is to distinguish between observed and imputed components. The discriminator is trained to minimize the classification loss (when classifying which components were observed and which have been imputed), and the generator is trained to maximize the discriminator's misclassification rate.

GAIN designed for non-sequential data



Yoon, Jinsung, James Jordon, and Mihaela Schaar. "Gain: Missing data imputation using generative adversarial nets." In International Conference on Machine Learning, pp. 5689-5698. PMLR, 2018. http://proceedings.mlr.press/v80/yoon18a/yoon18a.pdf

GAIN Performance: Imputation and Prediction

Table 2. Imputation performance in terms of RMSE (Average ± Std of RMSE)

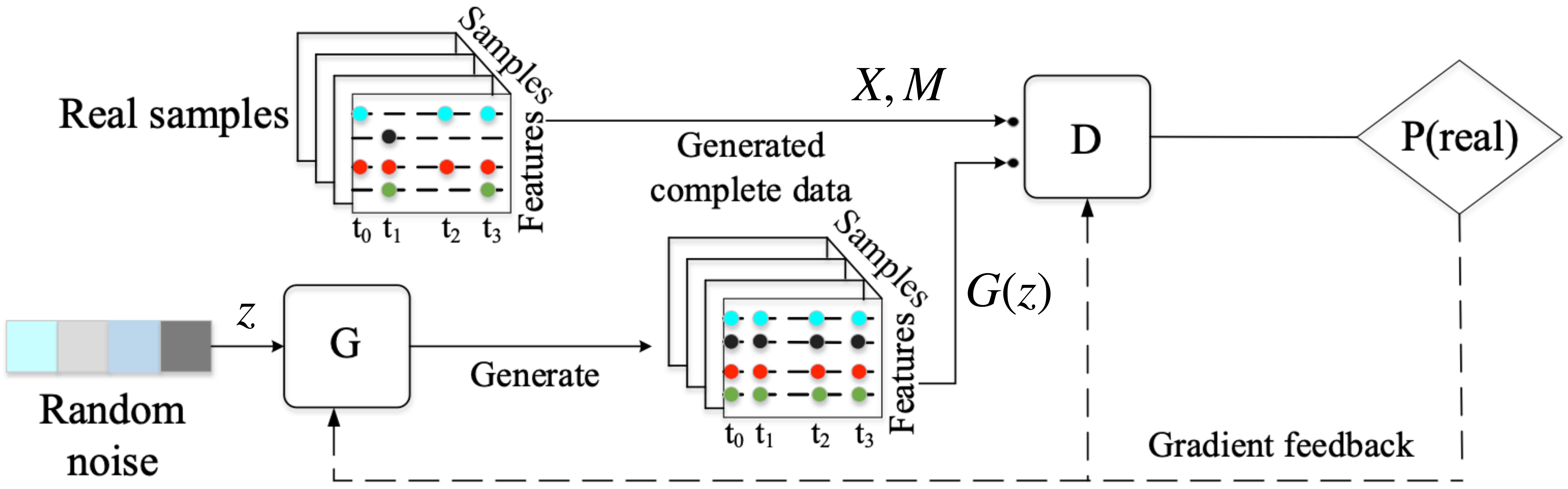
Algorithm	Breast	Spam	Letter	Credit	News
GAIN	**.0546 ± .0006**	**.0513± .0016**	**.1198± .0005**	**.1858 ± .0010**	**.1441 ± .0007**
MICE	.0646 ± .0028	.0699 ± .0010	.1537 ± .0006	.2585 ± .0011	.1763 ± .0007
MissForest	.0608 ± .0013	.0553 ± .0013	.1605 ± .0004	.1976 ± .0015	.1623 ± 0.012
Matrix	.0946 ± .0020	.0542 ± .0006	.1442 ± .0006	.2602 ± .0073	.2282 ± .0005
Auto-encoder	.0697 ± .0018	.0670 ± .0030	.1351 ± .0009	.2388 ± .0005	.1667 ± .0014
EM	.0634 ± .0021	.0712 ± .0012	.1563 ± .0012	.2604 ± .0015	.1912 ± .0011

Table 3. Prediction performance comparison

Algorithm	AUROC (Average ± Std)			
	Breast	**Spam**	**Credit**	**News**
GAIN	**.9930 ± .0073**	**.9529 ± .0023**	**.7527 ± .0031**	**.9711 ± .0027**
MICE	.9914 ± .0034	.9495 ± .0031	.7427 ± .0026	.9451 ± .0037
MissForest	.9860 ± .0112	.9520 ± .0061	.7498 ± .0047	.9597 ± .0043
Matrix	.9897 ± .0042	.8639 ± .0055	.7059 ± .0150	.8578 ± .0125
Auto-encoder	.9916 ± .0059	.9403 ± .0051	.7485 ± .0031	.9321 ± .0058
EM	.9899 ± .0147	.9217 ± .0093	.7390 ± .0079	.8987 ± .0157

Yoon, Jinsung, James Jordon, and Mihaela Schaar. "Gain: Missing data imputation using generative adversarial nets."
In International Conference on Machine Learning, pp. 5689-5698. PMLR, 2018. http://proceedings.mlr.press/v80/yoon18a/yoon18a.pdf

GAN-based Imputation for Time Series Data



$$L_r(\boldsymbol{z}) = ||\boldsymbol{X} \odot \boldsymbol{M} - G(\boldsymbol{z}) \odot \boldsymbol{M}||_2$$

$$L_d(\boldsymbol{z}) = -D(G(\boldsymbol{z}))$$

$$L_{imputation}(\boldsymbol{z}) = L_r(\boldsymbol{z}) + \lambda L_d(\boldsymbol{z})$$

- The generator G learns a mapping $G(z) = z \mapsto x$ that maps the random noise vector z to a complete time series which contains no missing value
- Uses Gated Recurrent Unit (GRU) but is adapted to cope with varying time lags between two consecutive valid observations due to data incompleteness
- Two part Imputation Loss function
  - *Masked Reconstruction Loss:* masked squared errors between the original sample x and the generated sample
  - *Discriminative Loss*: generated sample's degree of authenticity

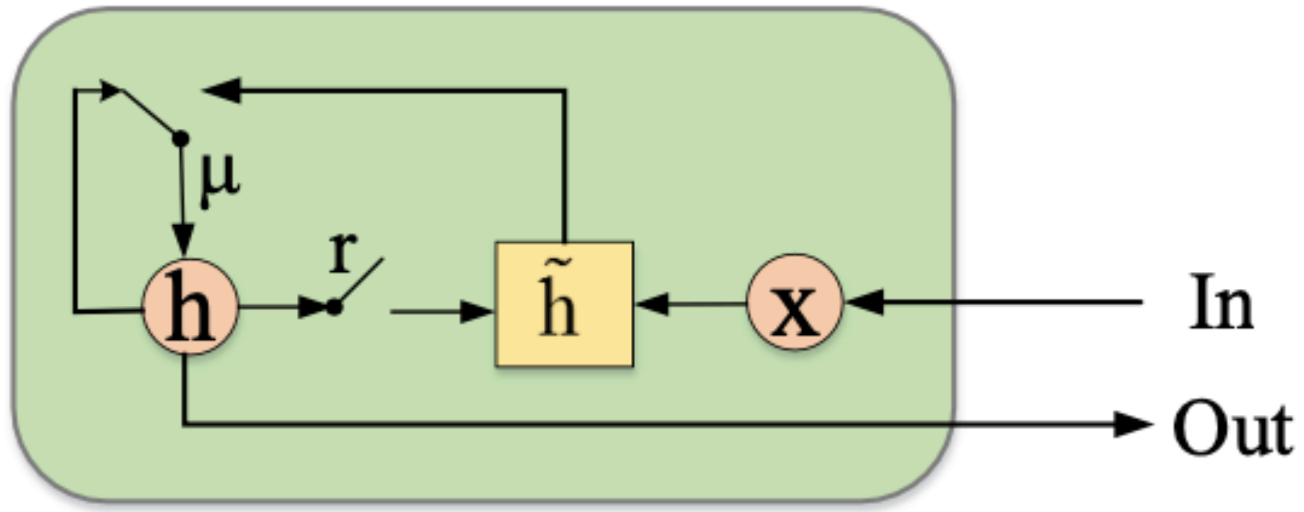
Luo, Yonghong, Xiangrui Cai, Ying Zhang, Jun Xu, and Xiaojie Yuan. "Multivariate time series imputation with generative adversarial networks." In Proceedings of the 32nd International Conference on Neural Information Processing Systems, pp. 1603-1614. 2018. https://proceedings.neurips.cc/paper/2018/file/96b9bff013acedfb1d140579e2fbeb63-Paper.pdf

Gated Recurrent Unit for data Imputation (GRUI) cell

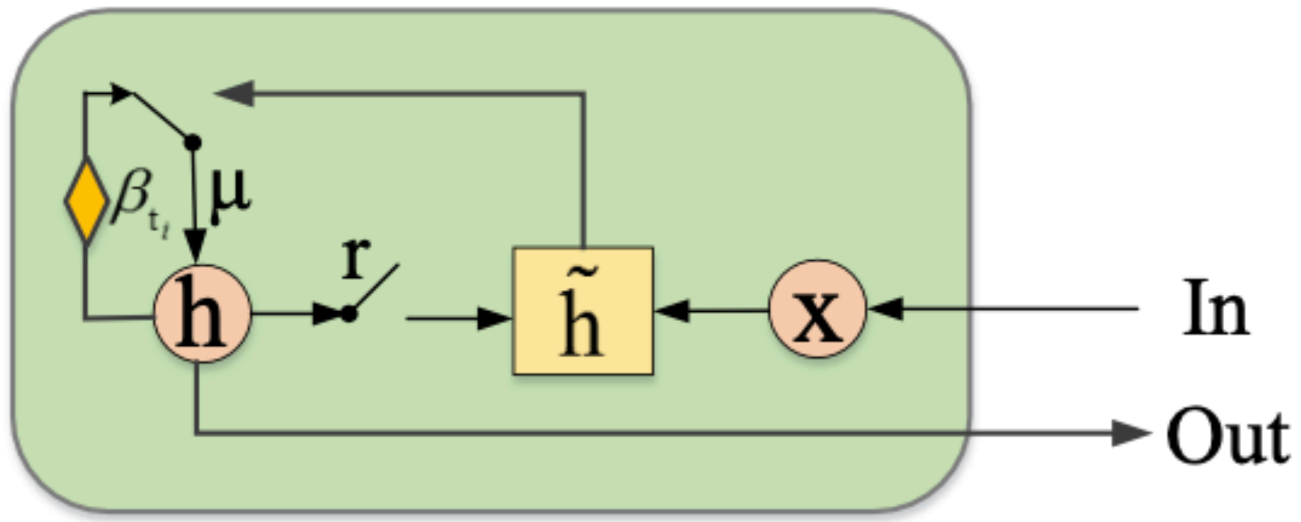
- Time lag matrix δ to record the time lag between current value and last valid value
- Example

$$\boldsymbol{X} = \begin{bmatrix} 1 & 6 & none & 9 \\ 7 & none & 7 & none \\ 9 & none & none & 79 \end{bmatrix}, T = \begin{bmatrix} 0 \\ 5 \\ 13 \end{bmatrix} \qquad \delta_{t_i}^j = \begin{cases} t_i - t_{i-1}, & M_{t_{i-1}}^j == 1 \\ \delta_{t_{i-1}}^j + t_i - t_{i-1}, & M_{t_{i-1}}^j == 0 \,\&\, i > 0 \\ 0, & i == 0 \end{cases} \; ; \quad \boldsymbol{\delta} = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 5 & 5 & 5 & 5 \\ 8 & 13 & 8 & 13 \end{bmatrix}$$

- A time decay vector β to control the influence of the past observations
  - $0 < \beta < 1$, and the larger the δ, the smaller the decay vector
  - Model β as a function of δ with parameters that need to be learnt $\boldsymbol{\beta}_{t_i} = 1/e^{\max(\mathbf{0}, \boldsymbol{W}_\beta \boldsymbol{\delta}_{t_i} + \boldsymbol{b}_\beta)}$
- GRUI Structure



Conventional GRU Cell



Modified GRUI Cell

$$\boldsymbol{h}'_{t_{i-1}} = \boldsymbol{\beta}_{t_i} \odot \boldsymbol{h}_{t_{i-1}},$$

$$\boldsymbol{\mu}_{t_i} = \sigma(\boldsymbol{W}_\mu \left[\boldsymbol{h}'_{t_{i-1}}, \boldsymbol{x}_{t_i}\right] + \boldsymbol{b}_\mu),$$

$$\tilde{\boldsymbol{h}}_{t_i} = tanh(\boldsymbol{W}_{\tilde{h}} \left[\boldsymbol{r}_{t_i} \odot \boldsymbol{h}'_{t_{i-1}}, \boldsymbol{x}_{t_i}\right] + \boldsymbol{b}_{\tilde{h}}),$$

$$\boldsymbol{r}_{t_i} = \sigma(\boldsymbol{W}_r \left[\boldsymbol{h}'_{t_{i-1}}, \boldsymbol{x}_{t_i}\right] + \boldsymbol{b}_r),$$

$$\boldsymbol{h}_{t_i} = (\mathbf{1} - \boldsymbol{\mu}_{t_i}) \odot \boldsymbol{h}_{t'_{i-1}} + \boldsymbol{\mu}_{t_i} \odot \tilde{\boldsymbol{h}}_{t_i},$$

https://proceedings.neurips.cc/paper/2018/file/96b9bff013acedfb1d140579e2fbeb63-Paper.pdf

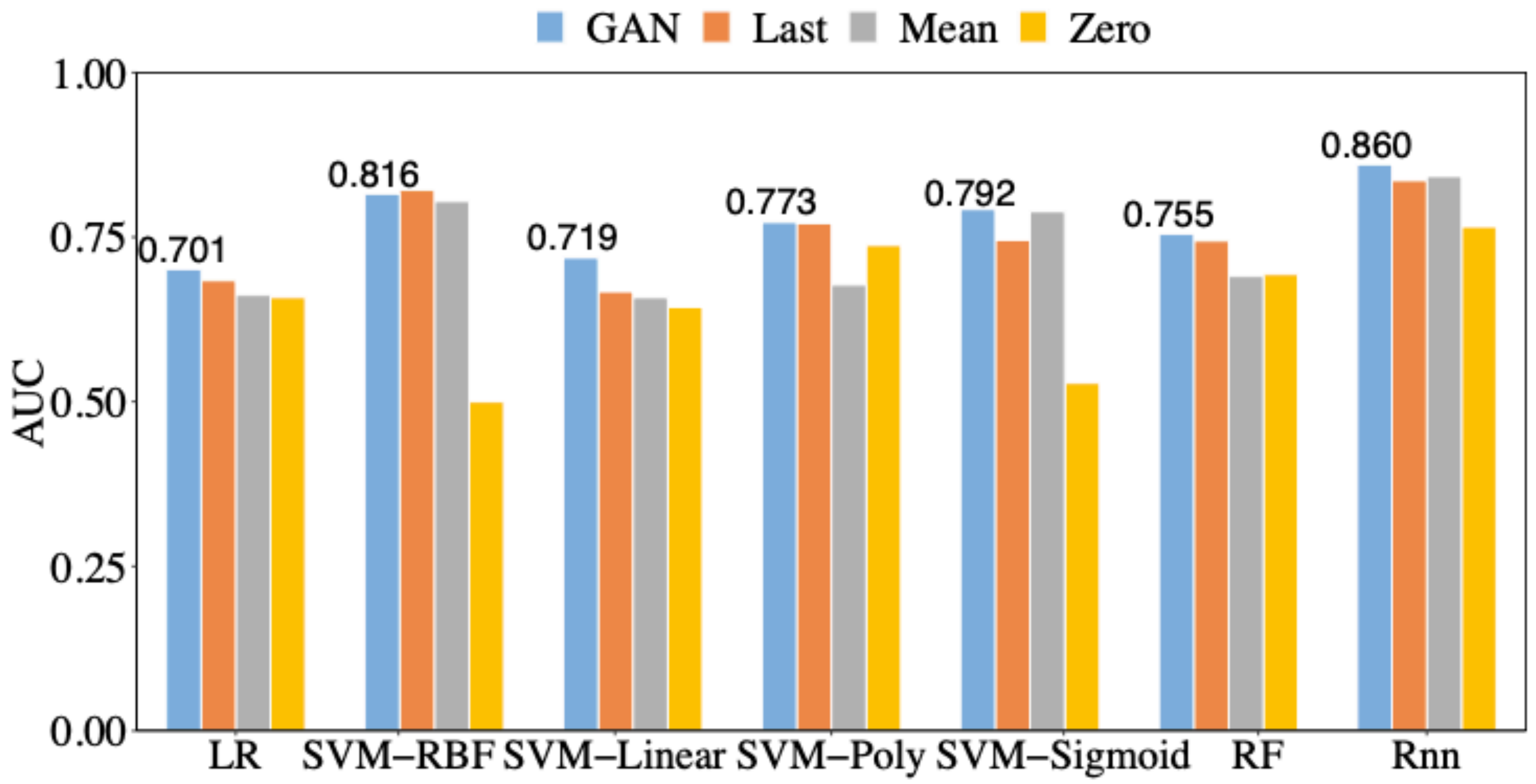
Performance of Imputation with GRUI-based GAN

Model	Result
Neural Network model called GRUD [7]	0.8424
Hazard Markov Chain model [29]	0.8381
Regularized Logistic Regression model [25]	0.848
GAN based imputation & RNN model	**0.8603**

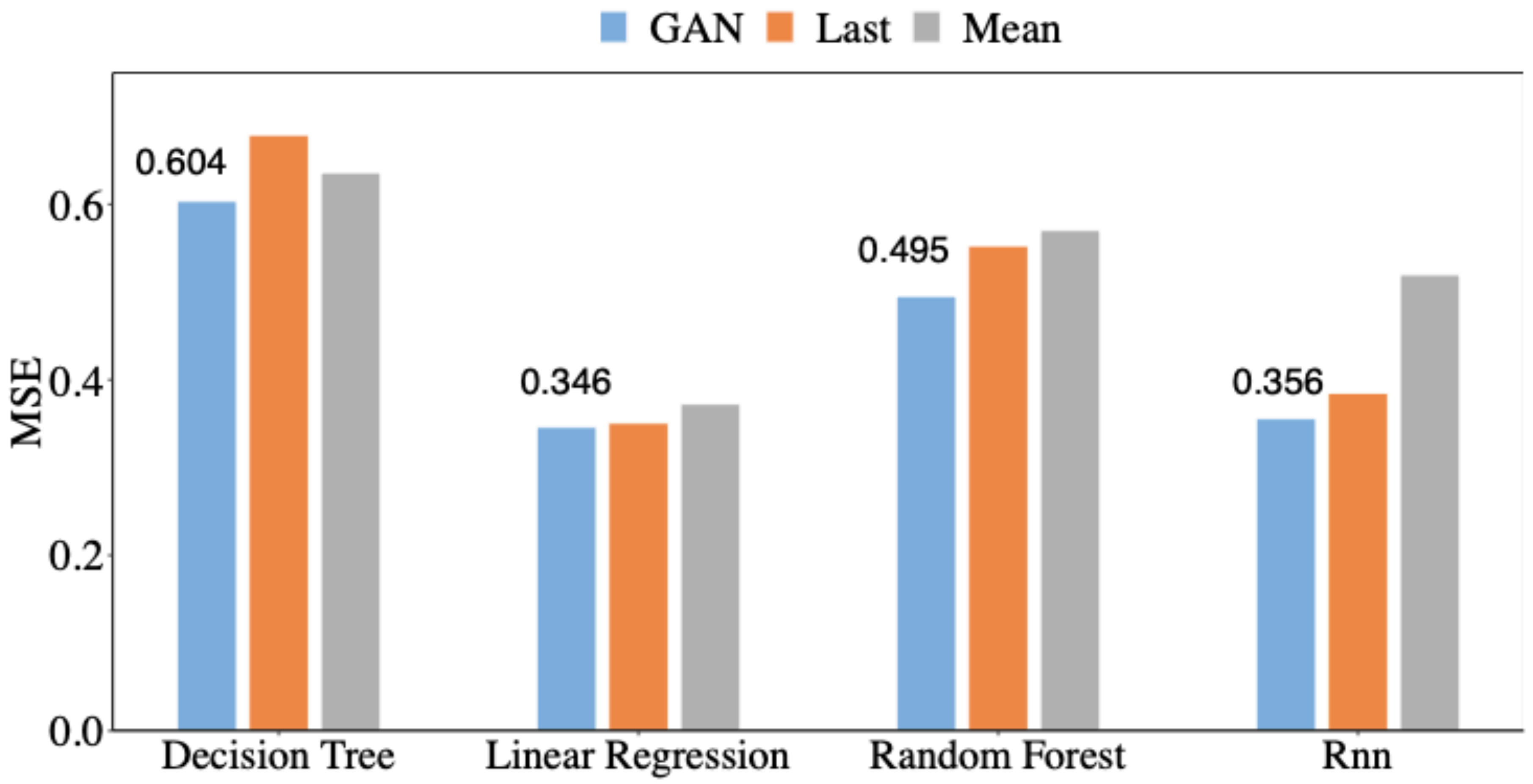
The AUC score of the mortality prediction task on the Physionet dataset.

Missing-rate	Last filling	Mean filling	KNN filling	MF filling	GAN filling
90%	2.870	**1.002**	1.243	1.196	1.018
80%	1.689	0.937	0.873	0.860	**0.837**
70%	1.236	0.935	0.852	0.805	**0.780**
60%	1.040	0.973	0.856	0.834	**0.803**
50%	0.990	0.923	0.798	0.772	**0.743**
40%	0.901	0.914	0.776	0.787	**0.753**
30%	0.894	0.907	0.803	0.785	**0.780**
20%	1.073	0.916	0.892	0.850	**0.844**

The MSE results of the proposed method and other imputation methods on the KDD Air Quality dataset



The AUC score of mortality prediction by different classification models trained on different imputed datasets



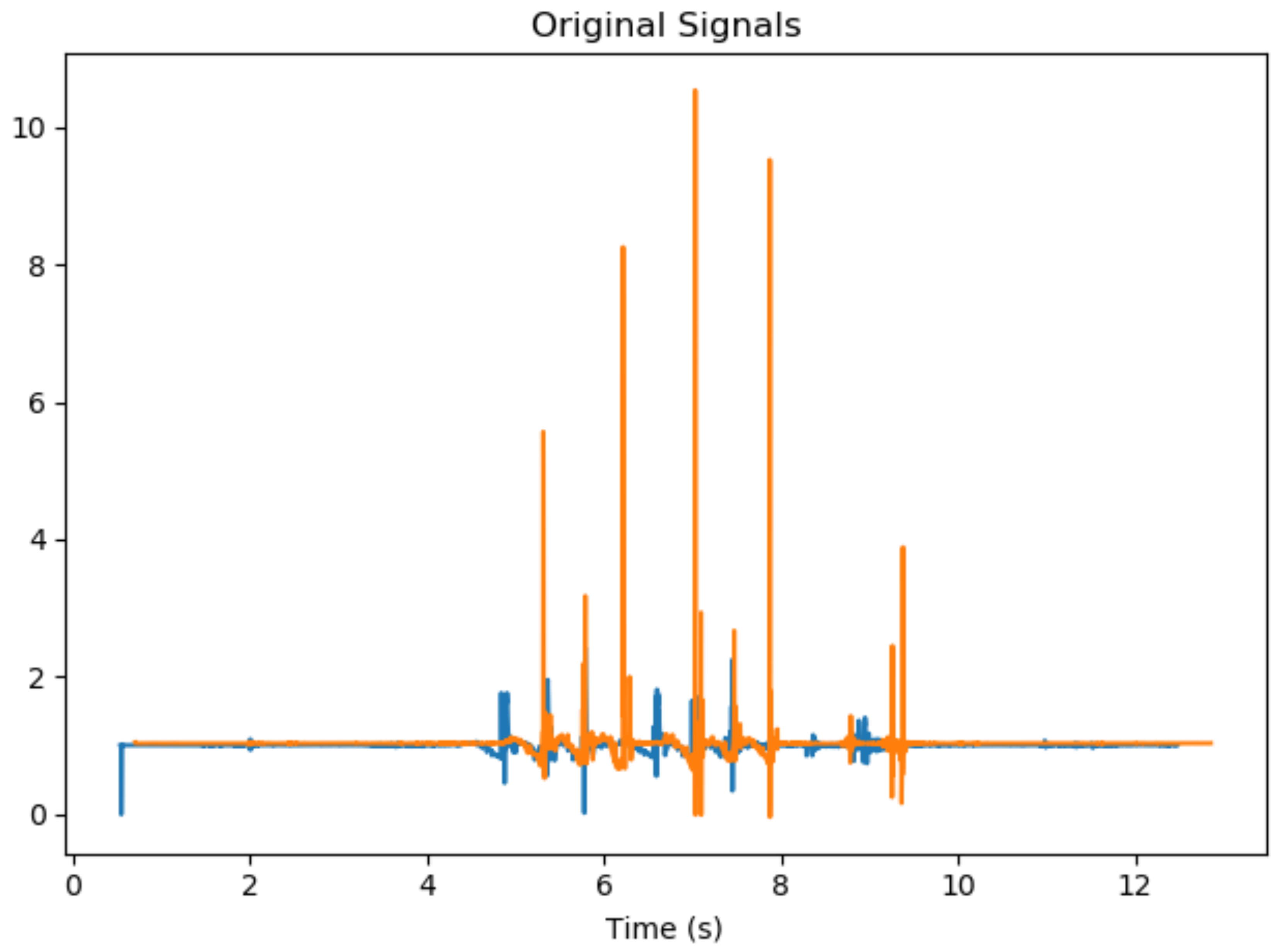
The MSE of air quality prediction by different regression models trained on different imputed datasets

https://proceedings.neurips.cc/paper/2018/file/96b9bff013acedfb1d140579e2fbeb63-Paper.pdf

Poor Quality Temporal Metadata

Problems in time-related metadata

- Case 1: Sampling period in time series with regular sampling
  - Actual sample interval may differ from configured sampling period
  - Systematic errors and drifts in clock frequency
    - manufacturing variations, variations due to temperature, indeterministic software delays
  - Lead to erroneous inference and control
    - estimate of physical variables such as location, speed, …
    - classes distinguished by latent temporal properties



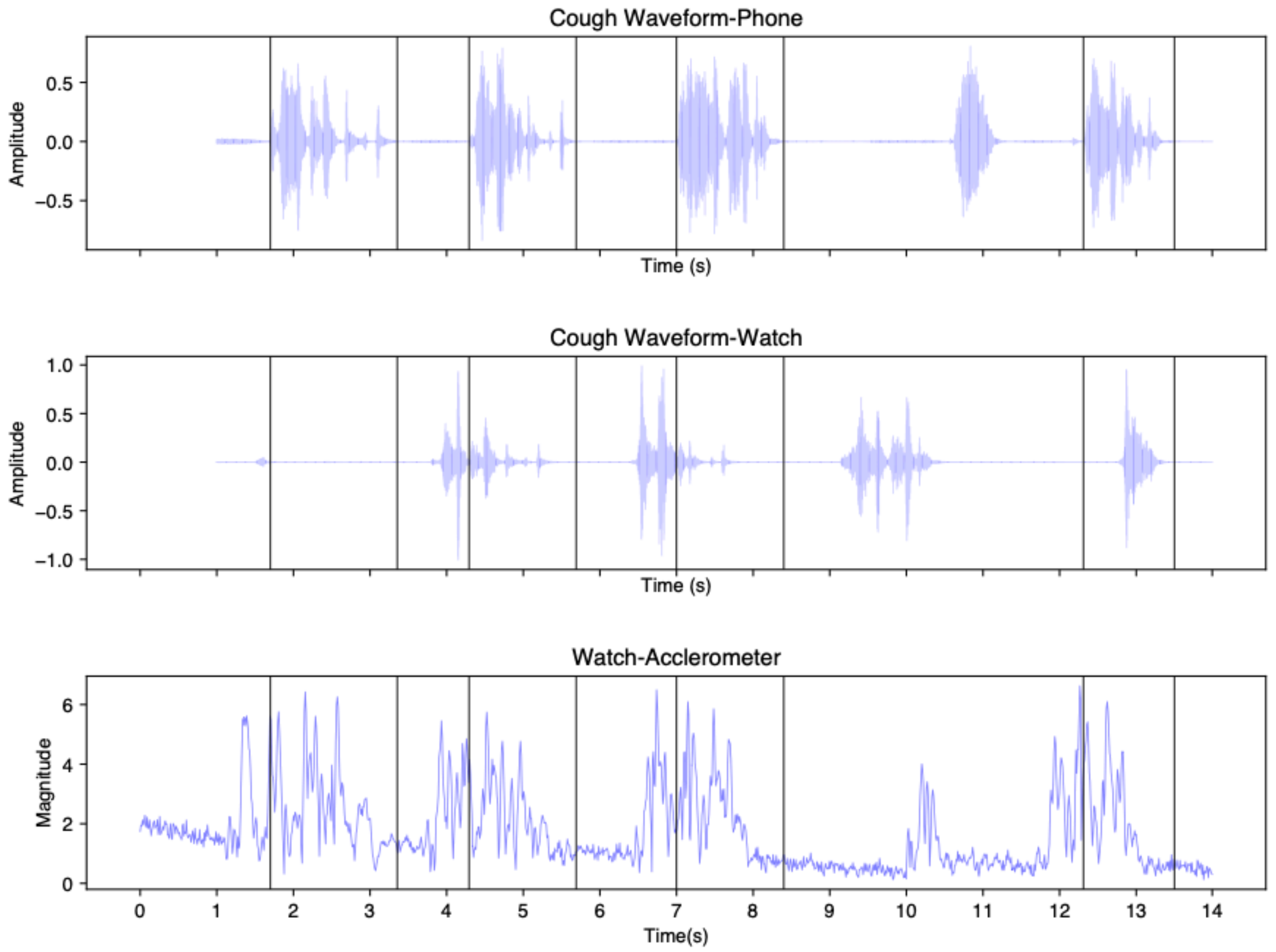
The blue and orange signals were recorded on two separate devices. Their recordings are similar, but one of them is shifted in time, and one of them is misinformed about it's sampling rate (https://blog.endaq.com/synchronizing-signals)

Problems in time-related metadata

- Case 1: Sampling period in time series with regular sampling
  - Actual sample interval may differ from configured sampling period
  - Systematic errors and drifts in clock frequency
    - manufacturing variations, variations due to temperature, indeterministic software delays
  - Lead to erroneous inference and control
    - estimate of physical variables such as location, speed, …
    - classes distinguished by latent temporal properties

- Case 2: Timestamps in time series
  - Explicit timestamps used with events and irregular sampling
  - Timestamps may be incorrect
    - Poorly synchronized sensor clocks (sender-side time-stamping)
    - Indeterminate OS & network delays (receiver-side time-stamping)
  - Lead to erroneous inference and control
    - incorrect ordering of events
    - domain shift due to temporal misaligned
    - estimate of physical variables such as location, speed, …
    - classes distinguished by latent temporal properties

Time Offset ~ 2.6s



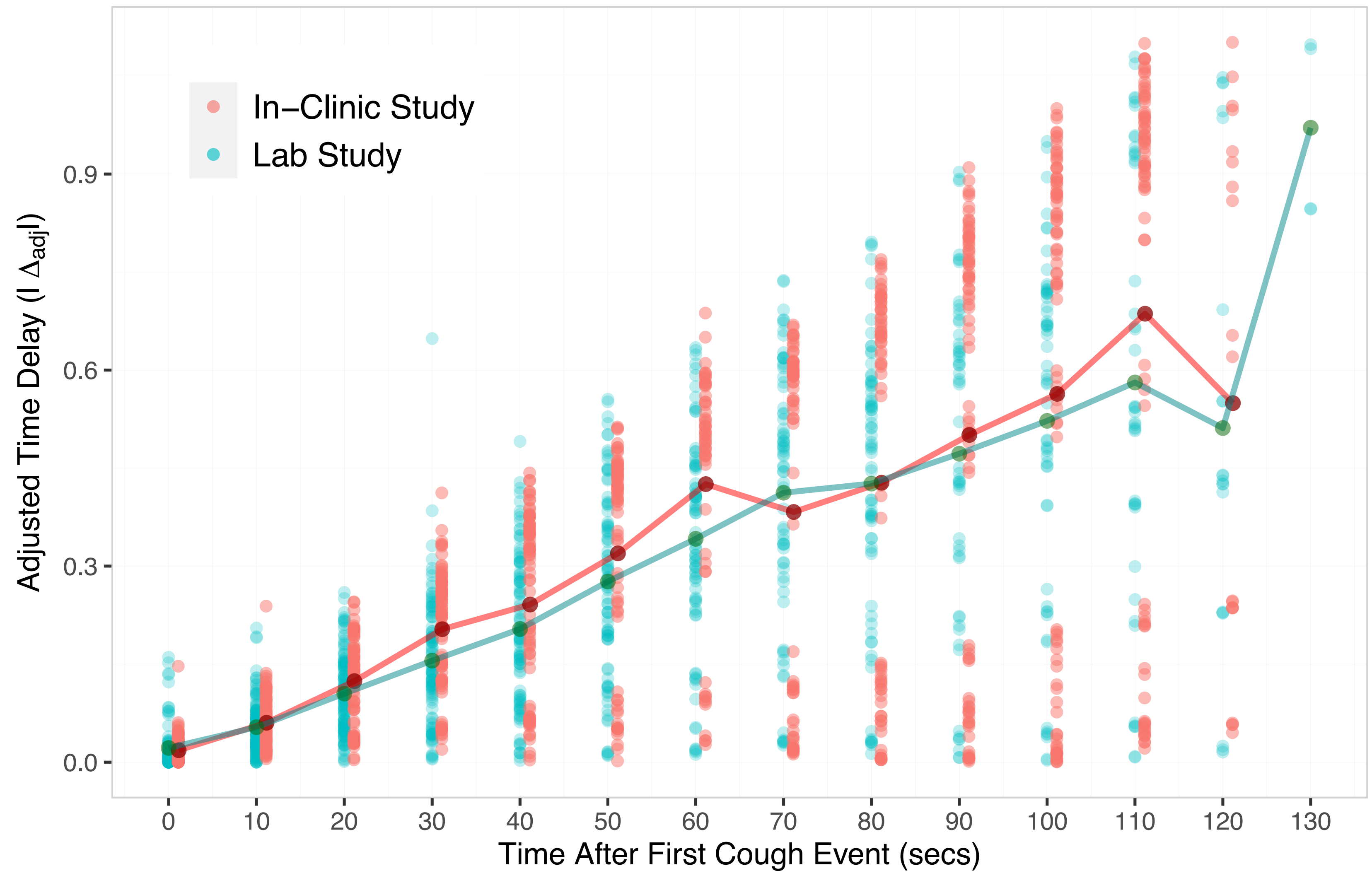
https://dl.acm.org/doi/pdf/10.1145/3382507.3418855

Problems in time-related metadata

- Case 1: Sampling period in time series with regular sampling
  - Actual sample interval may differ from configured sampling period
  - Systematic errors and drifts in clock frequency
    - manufacturing variations, variations due to temperature, indeterministic software delays
  - Lead to erroneous inference and control
    - estimate of physical variables such as location, speed, …
    - classes distinguished by latent temporal properties

- Case 2: Timestamps in time series
  - Explicit timestamps used with events and irregular sampling
  - Timestamps may be incorrect
    - Poorly synchronized sensor clocks (sender-side time-stamping)
    - Indeterminate OS & network delays (receiver-side time-stamping)
  - Lead to erroneous inference and control
    - incorrect ordering of events
    - domain shift due to temporal misaligned
    - estimate of physical variables such as location, speed, …
    - classes distinguished by latent temporal properties

Time Drifts



https://dl.acm.org/doi/pdf/10.1145/3382507.3418855

Aside: *Uncertainty* vs. *Variability*

- **Uncertain**: not knowable at runtime; may change
  - Tradeoff between uncertainty and cost of data acquisition
  - E.g., Inaccurate clock can result in *data timestamp uncertainty*

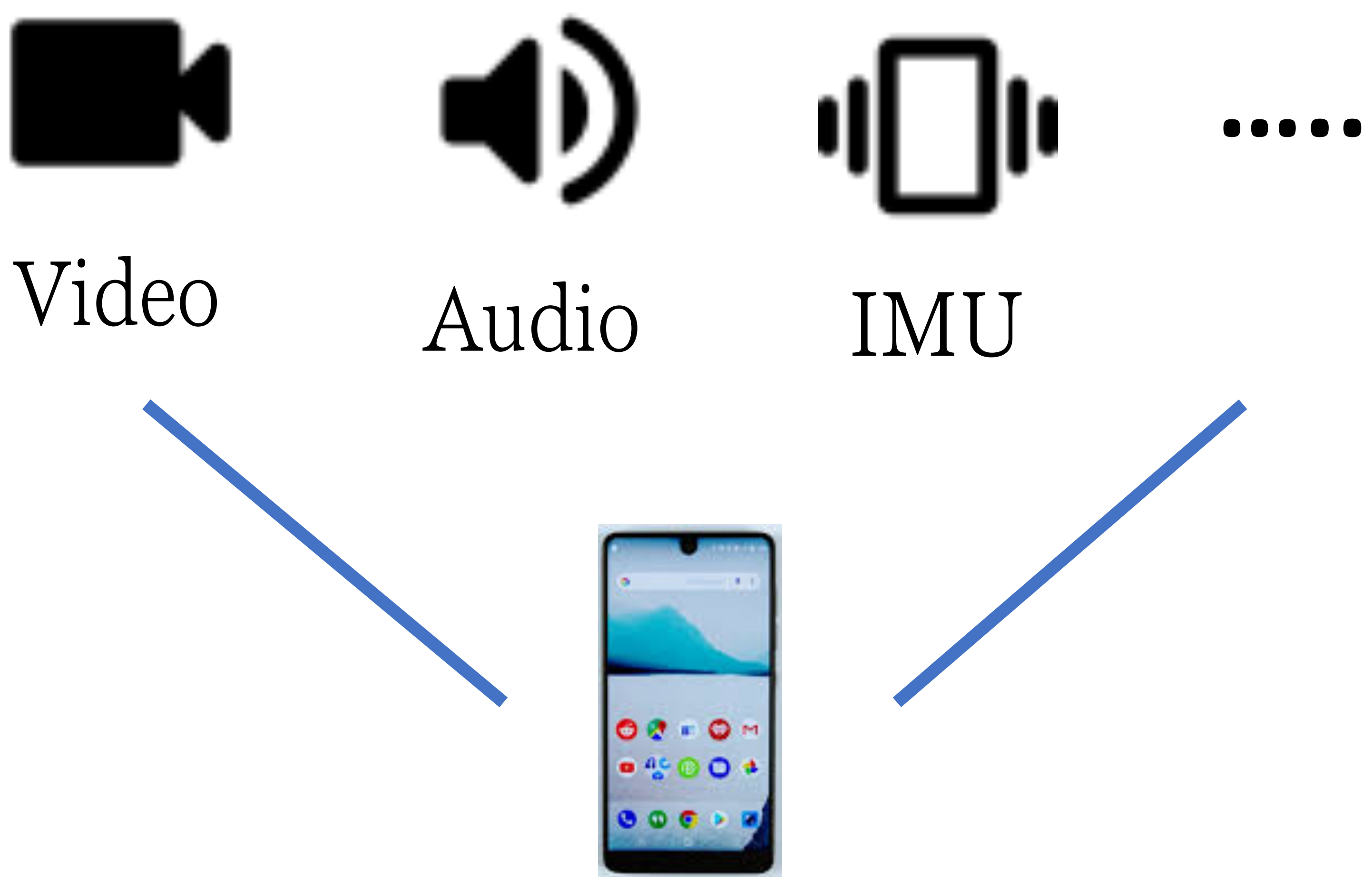
- **Variable**: not consistent but inconsistency may be measurable at runtime
  - Can be measured directly/indirectly at runtime via other sensors
  - E.g., Variation in sampling rate can be measured by the systems

- Temporal metadata has both of these issues

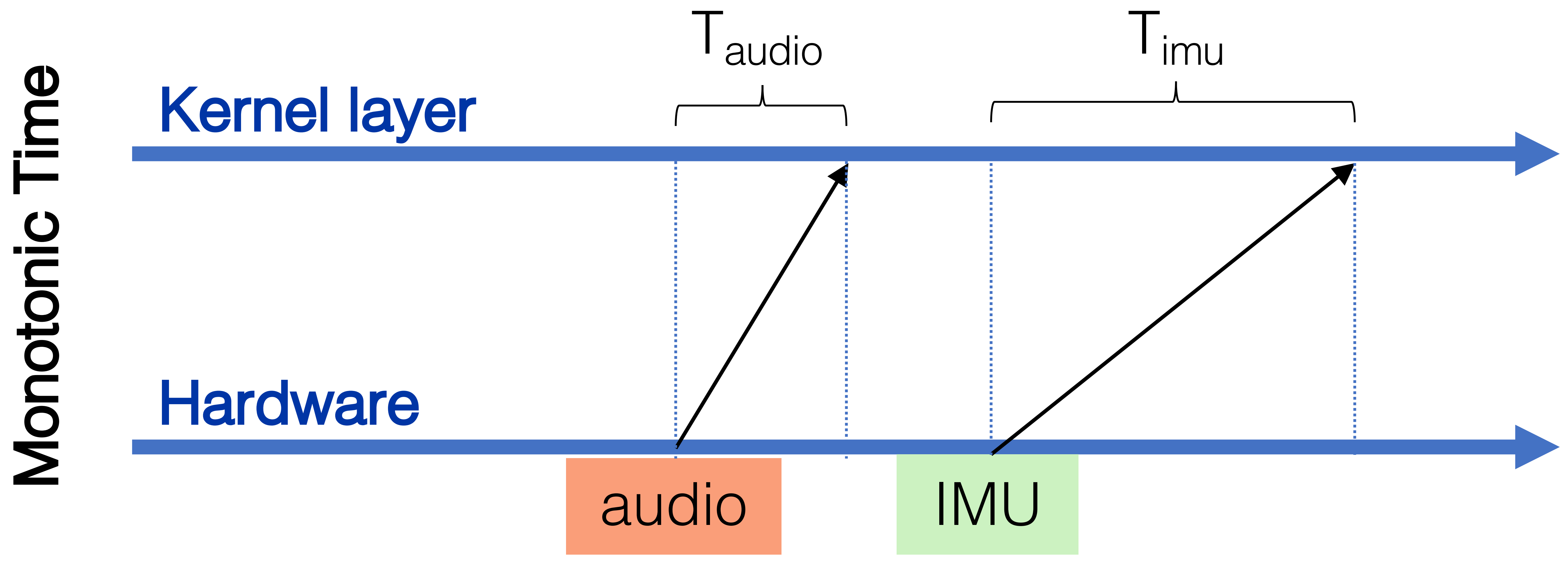
Multimodal Fusion

Multimodal Fusion on a Single Device

- Modalities can be misaligned
  - E.g. audio and IMU data are timestamped by kernel along different software pathways
  - Stack latencies can be different.



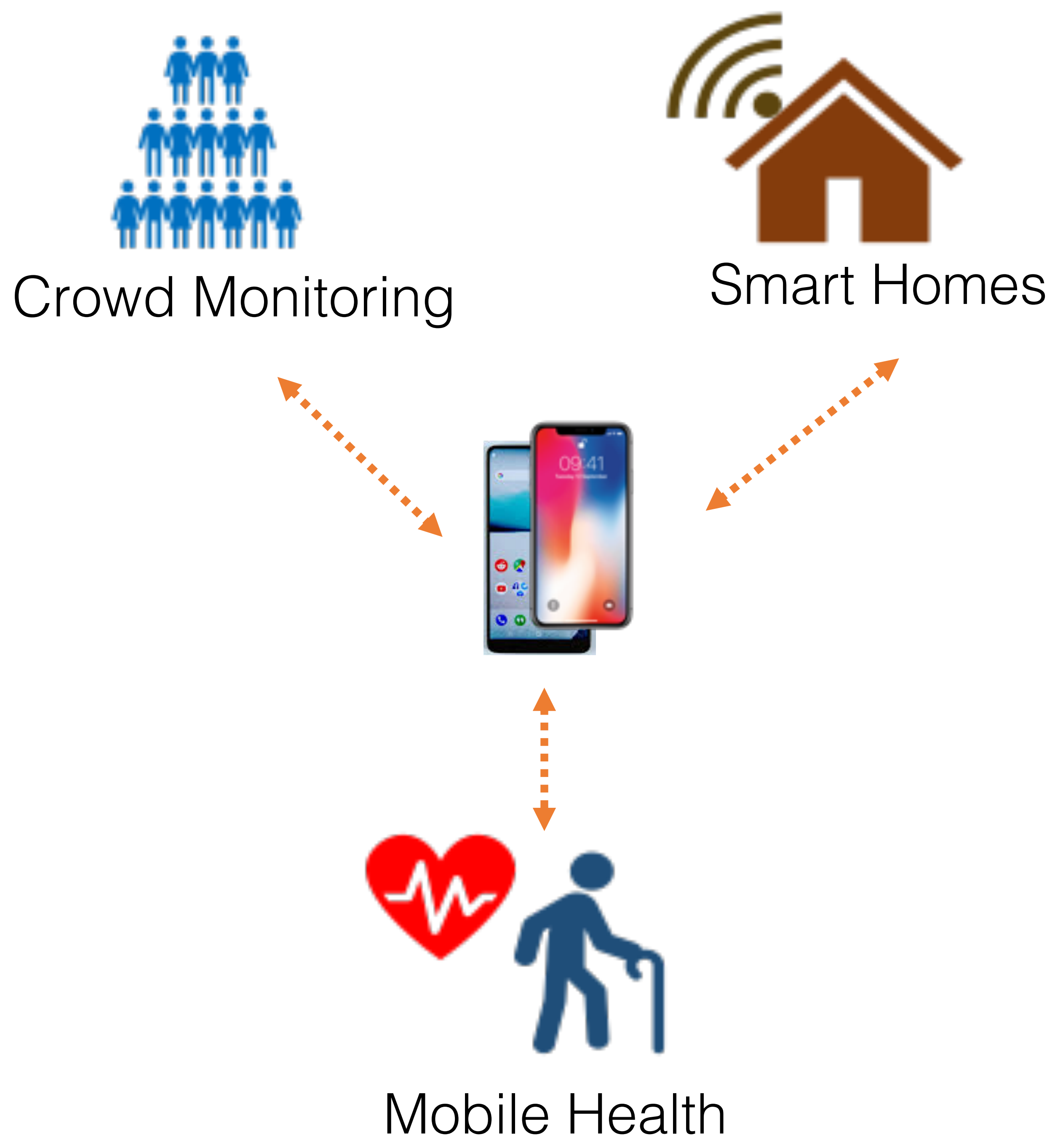
Audio latency: ~ (10 ms - 40 ms)
Jitter in IMU latency: ~10 ms [Stisen15]



Timestamp for audio and IMU

Audio Latency: https://superpowered.com/
[Stisen15] Stisen, Allan, et al. "Smart devices are different: Assessing and mitigatingmobile sensing heterogeneities for activity recognition." Proceedings of the 13th ACM conference on embedded networked sensor systems. 2015.

Multimodal Fusion across Multiple Devices (e.g. Smartphones)



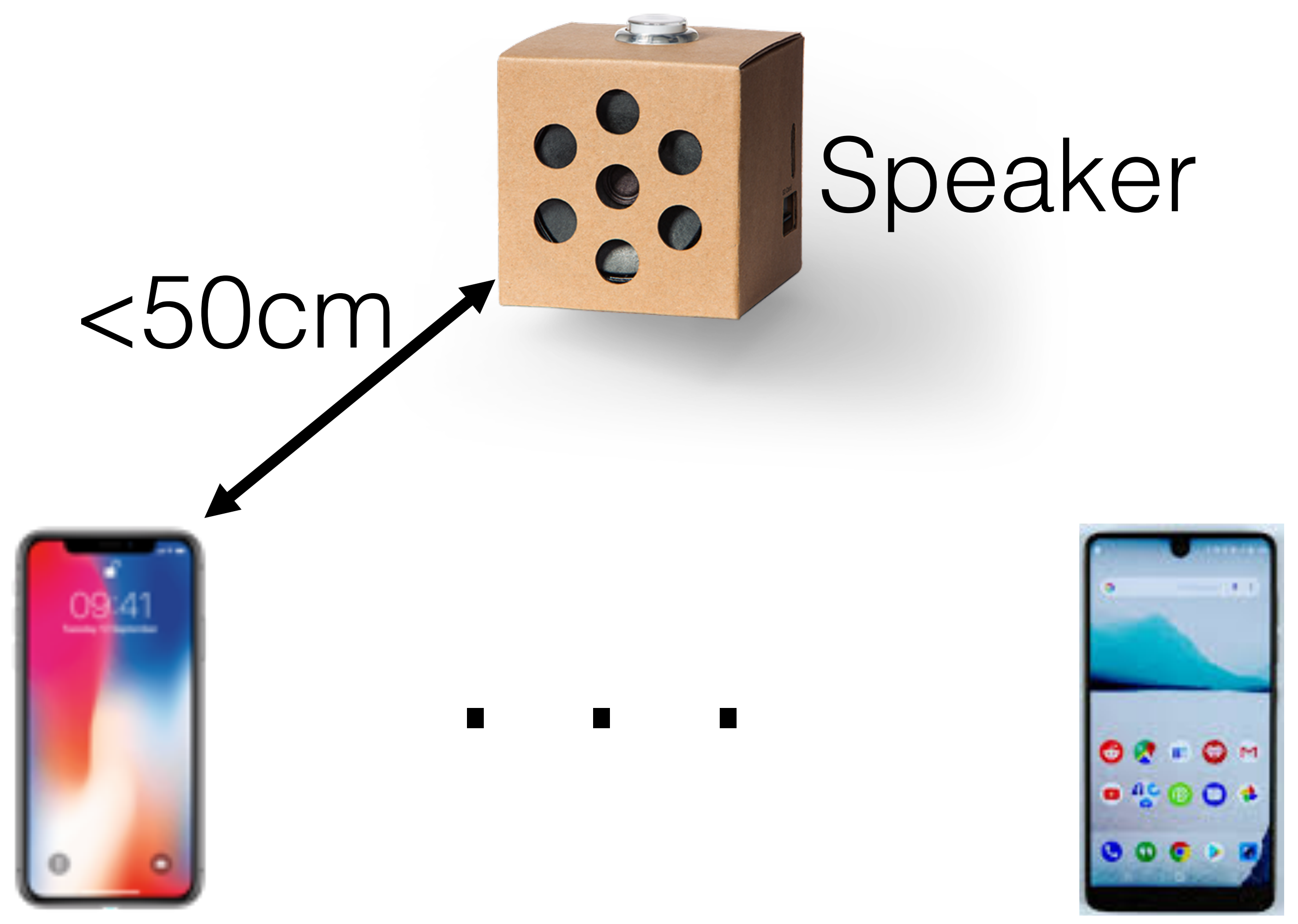
The most straightforward approach to align the modalities is to use the timestamps.

Are the timestamps across devices such as smartphones reliable ?

A Study of System Clock Accuracy

ID	Device	OS	Year	SIM?
I1	iPhone 6	iOS 12.1.4	2014	N
I2	iPad Pro 9"	iOS 12.1.4	2016	N
I3	iPhone 7+	iOS 12.1.4	2016	N
I4	iPhone 6S	iOS 12.1.4	2015	N
I5	iPhone 6	iOS 12.1.4	2014	Y
A1	Nexus 5X	Android 8.1.0	2015	Y
A2	Nexux 7 Tab	Android 6.0.1	2012	N
A3	Huawei P9	Android 7.0	2016	N
A4	OnePlus A1	Android 5.1.1	2014	N
A5	Samsung GTS2	Android 7.0	2015	N
A6	Nexus 5X	Android 8.1.0	2015	Y
A7	Nexus 7 Tab	Android 6.0.1	2012	N
A8	Pixel 3	Android 9.0	2018	Y

The study was conducted in March 2019.
A patch was submitted to Google.
Changes have been done to new Android versions.

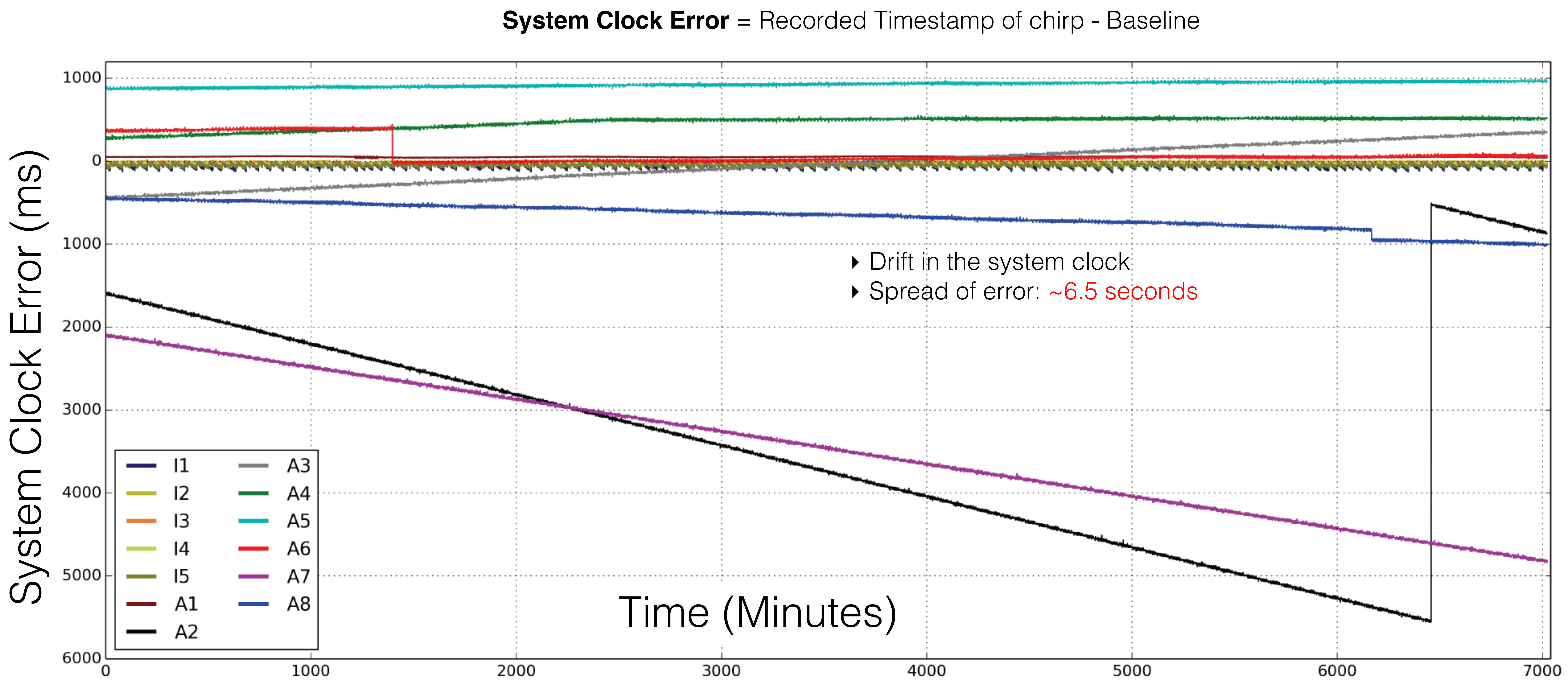


Speaker generates periodic chirp (~20 Seconds)
Baseline: Average of NTP clients from all phones
NTP variability: ~10 ms [Mani16].
Audio latency: ~(10 ms - 40 ms).

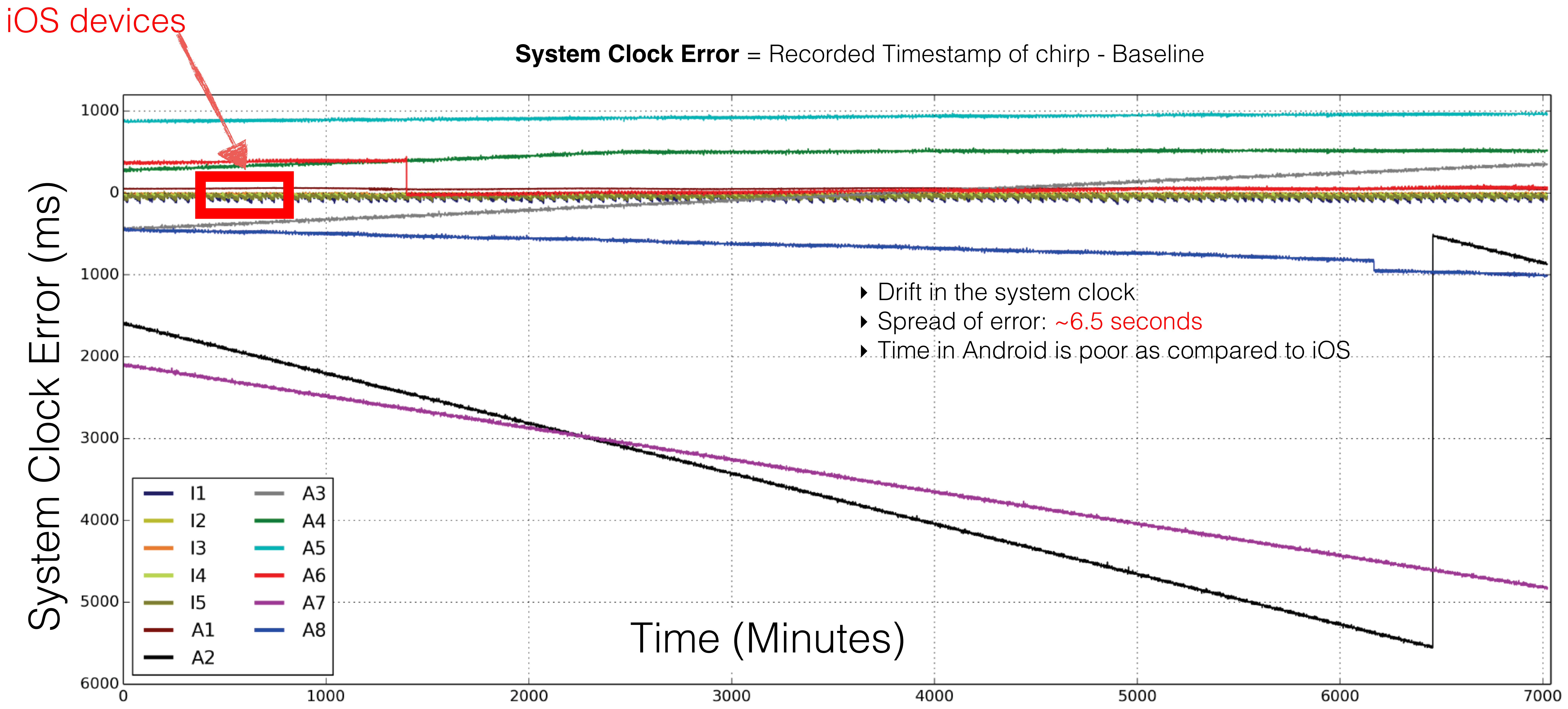
[Mani16] Mani, Sathiya Kumaran, et al. "Mntp: Enhancing time synchronization for mobile devices. Proceedings of the 2016 Internet Measurement Conference. 2016.
Audio Latency: https://superpowered.com/

Sandha, Sandeep Singh, Joseph Noor, Fatima M. Anwar, and Mani Srivastava. "Time awareness in deep learning-based multimodal fusion across smartphone platforms." In 2020 IEEE/ACM Fifth International Conference on Internet-of-Things Design and Implementation (IoTDI), pp. 149-156. IEEE, 2020.

Five Day Study of iOS and Android Devices

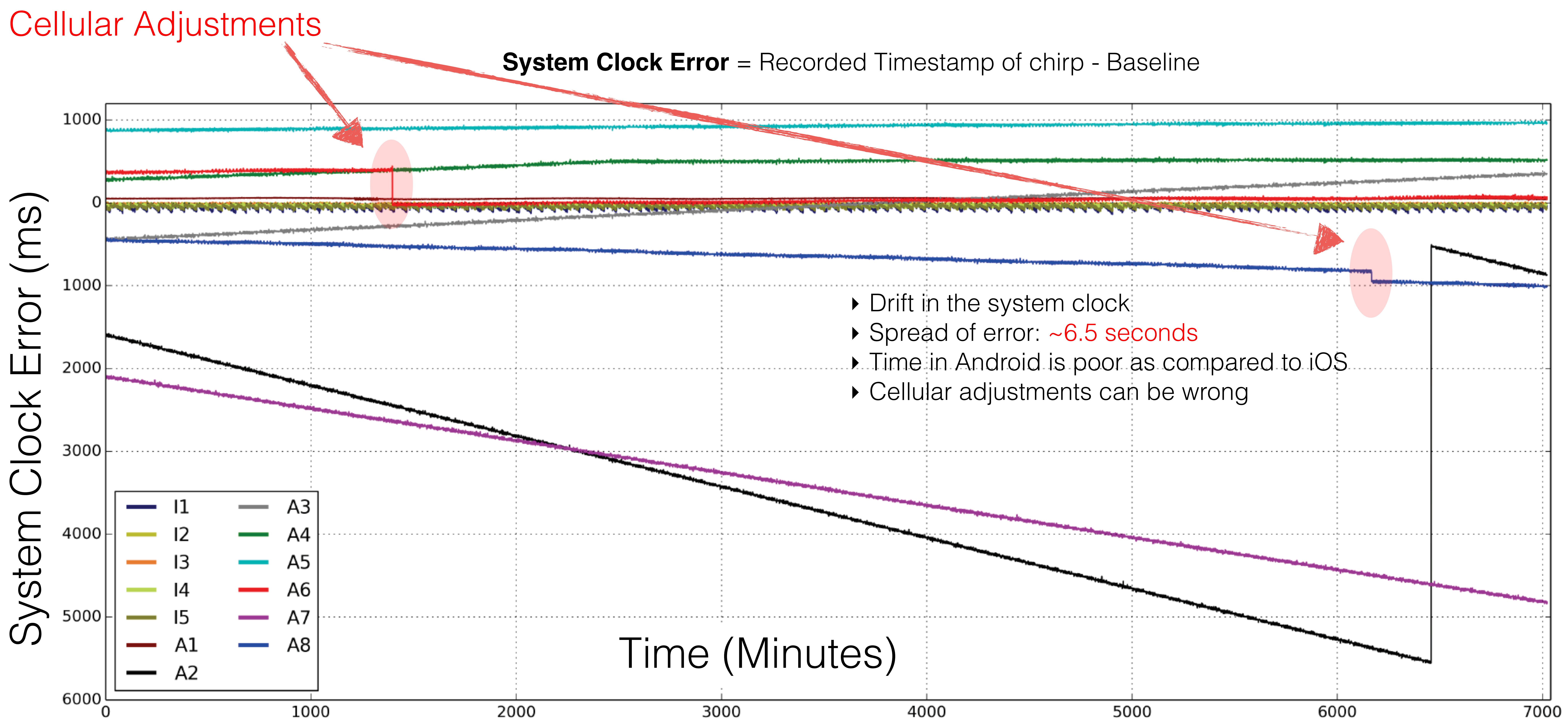


Five Day Study of iOS and Android Devices



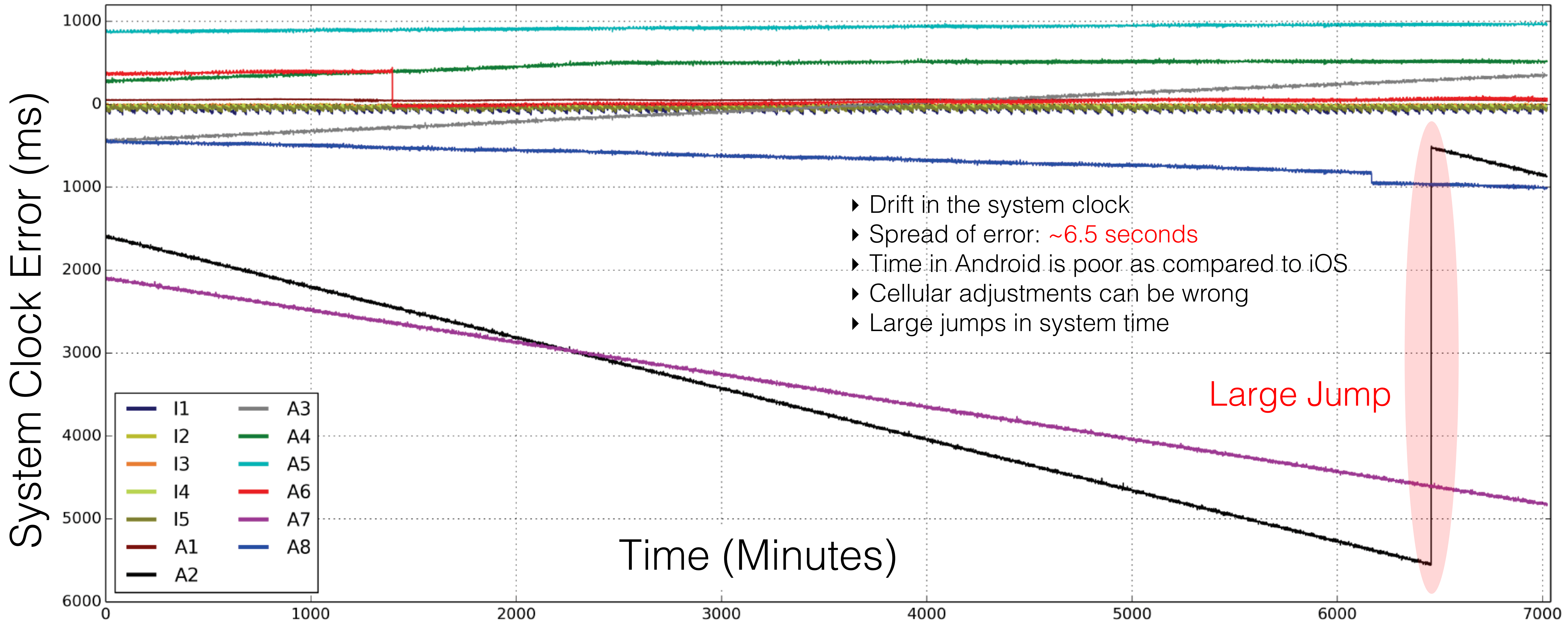
- Drift in the system clock
- Spread of error: ~6.5 seconds
- Time in Android is poor as compared to iOS

Five Day Study of iOS and Android Devices

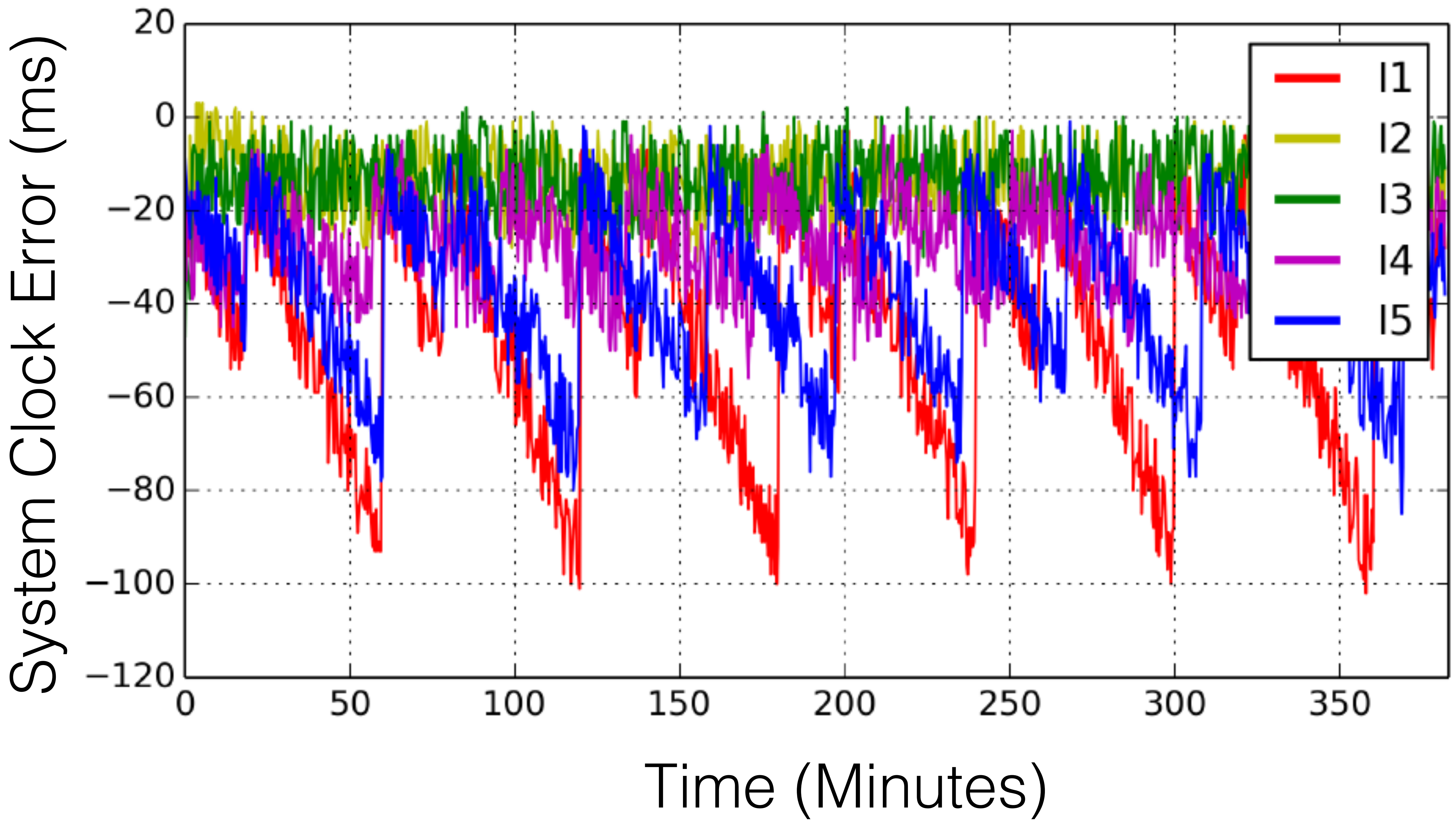


Five Day Study of iOS and Android Devices

System Clock Error = Recorded Timestamp of chirp - Baseline



Five Day Study of iOS and Android Devices



iOS

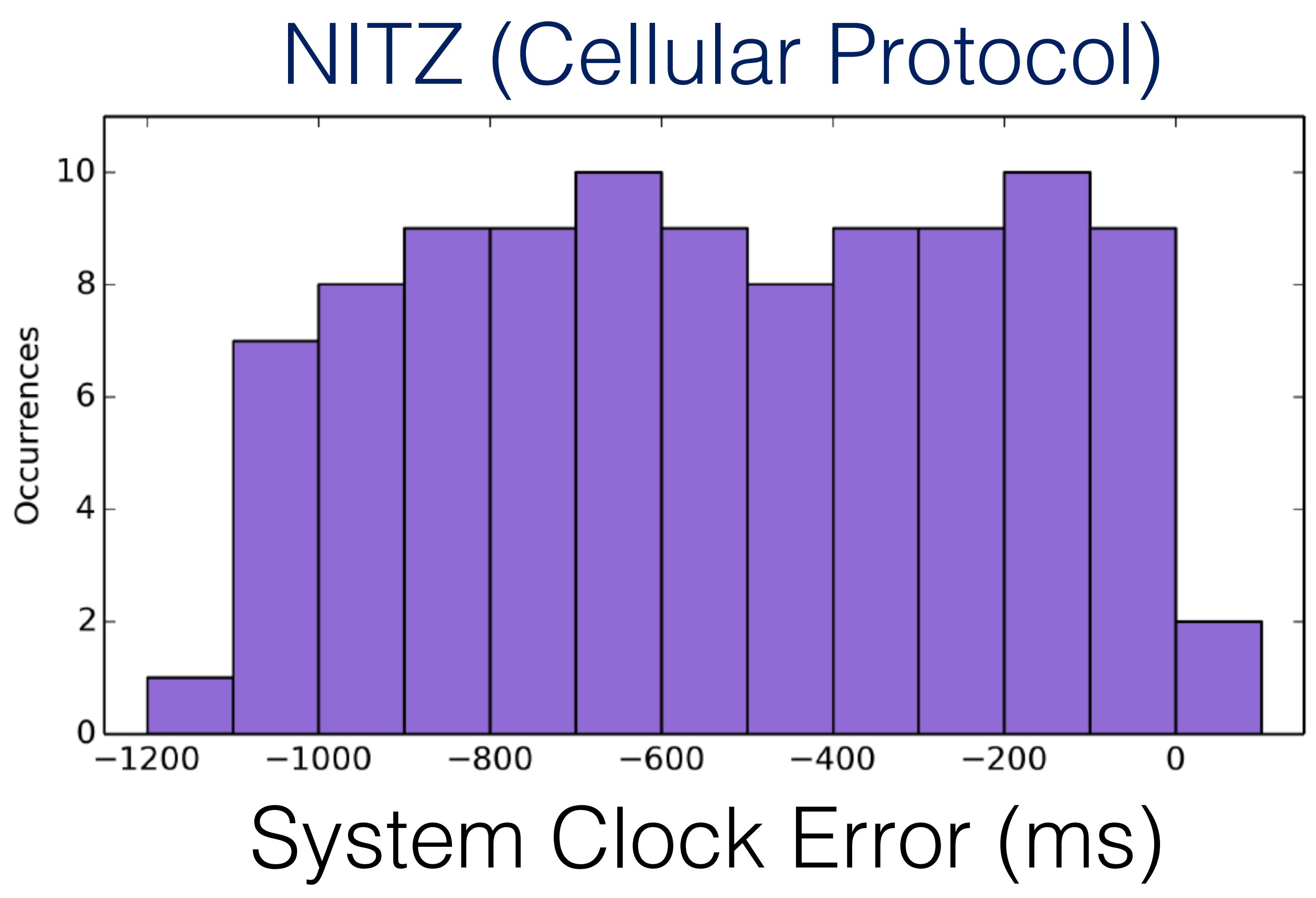
- Error within ~100ms
- Aggressive synchronization

Android

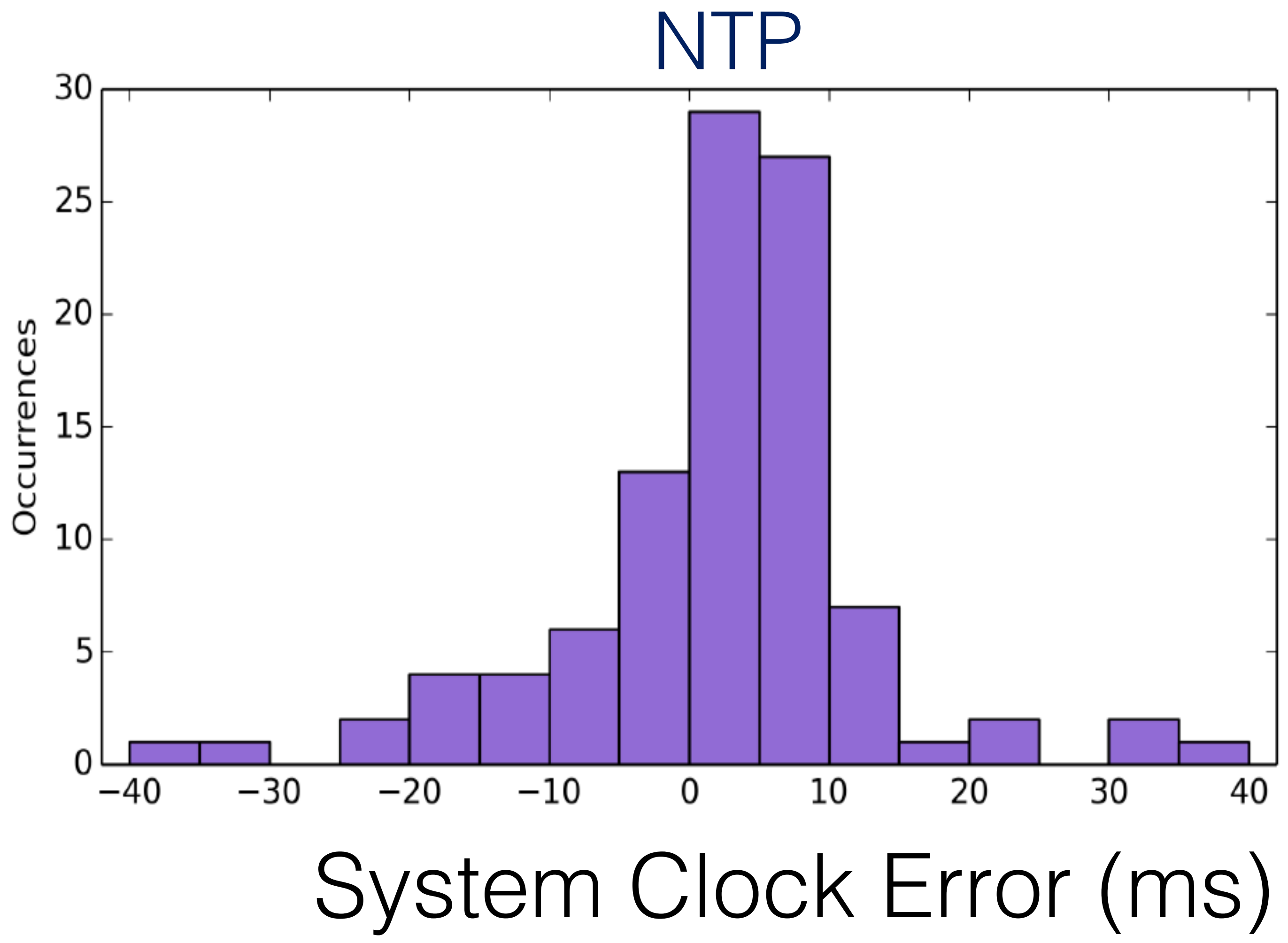
- Error ~5000ms
-

Why does Android exhibit high timing errors?

- Android System Time
  - Two update mechanisms: NITZ and NTP
  - NITZ has priority, and it directly updates the system time.
  - NTP done every 24 hour.
  - NTP updates system time only if the error is more than 5000ms ⇒ Large jumps



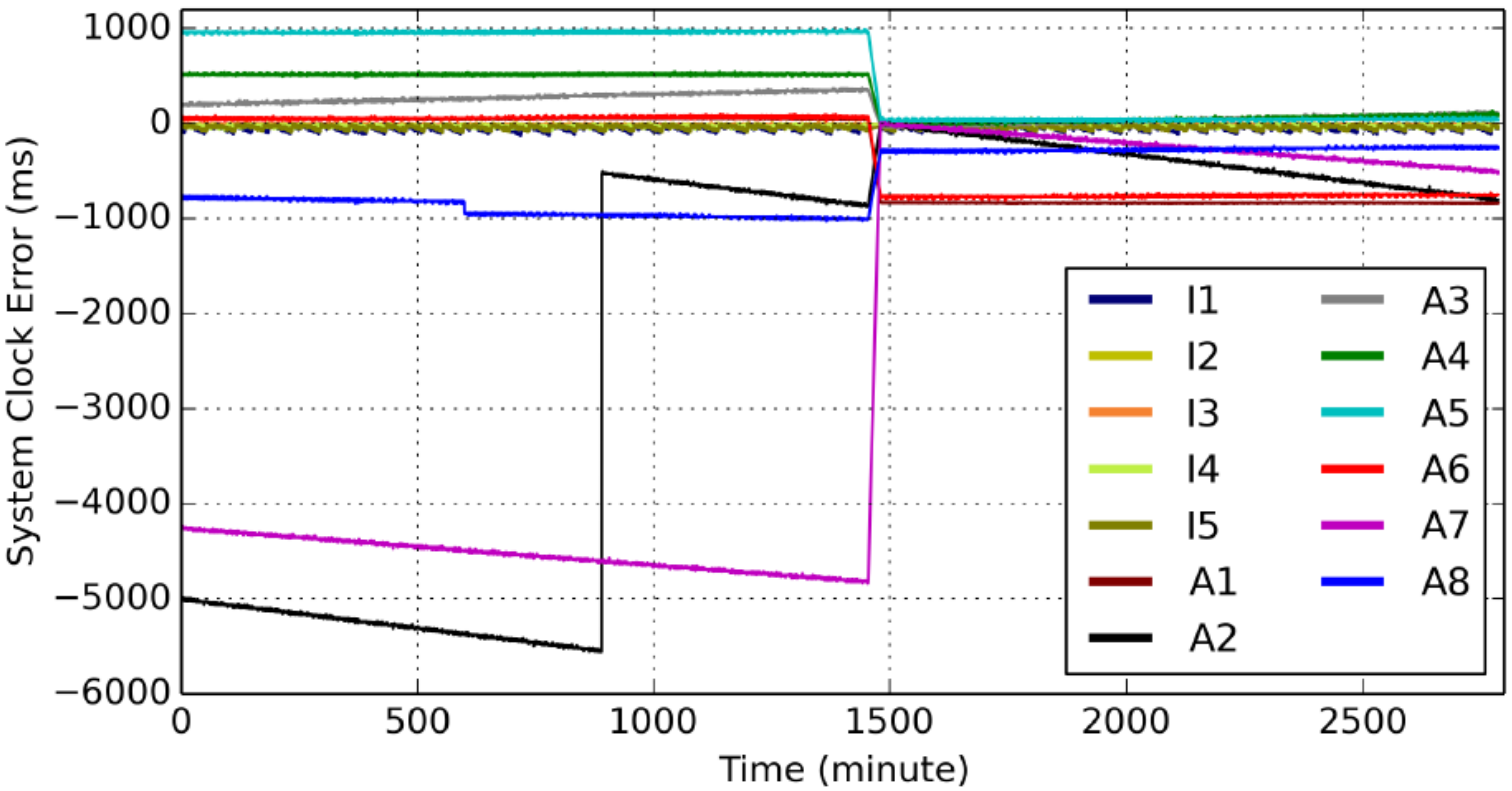

```
if (DBG) Log.d(TAG, "Ntp time is close enough = " + ntp);
```



- Changes in Android 10
  - 5000ms is modified to 2000ms

Simple Trick: Restart all Phones

- Phones with SIM have error ranging from 300 to 800ms as they use NITZ
- Phones without SIM use NTP and so have offset < 40ms



Impact of Timestamp Errors on Multimodal Fusion

Use case: Human activity recognition

Audio from one smartphone and IMU from another
(Dataset: https://github.com/nesl/CMActivities-DataSet)

Activity	Number of Videos	Duration (sec)
Go Upstairs	162	1338
Go Downstairs	161	1113
Walk	119	1143
Run	115	891
Jump	73	995
Wash Hand	73	1070
Jumping Jack	90	958

Tianwei Xing, Sandeep Singh Sandha, Bharathan Balaji, Supriyo Chakraborty, Mani Srivastava,*
"Enabling edge devices that learn from each other: Cross modal training for activity recognition,"EdgeSys-18

Impact of Timestamp Errors on Multimodal Fusion

Use case: Human activity recognition

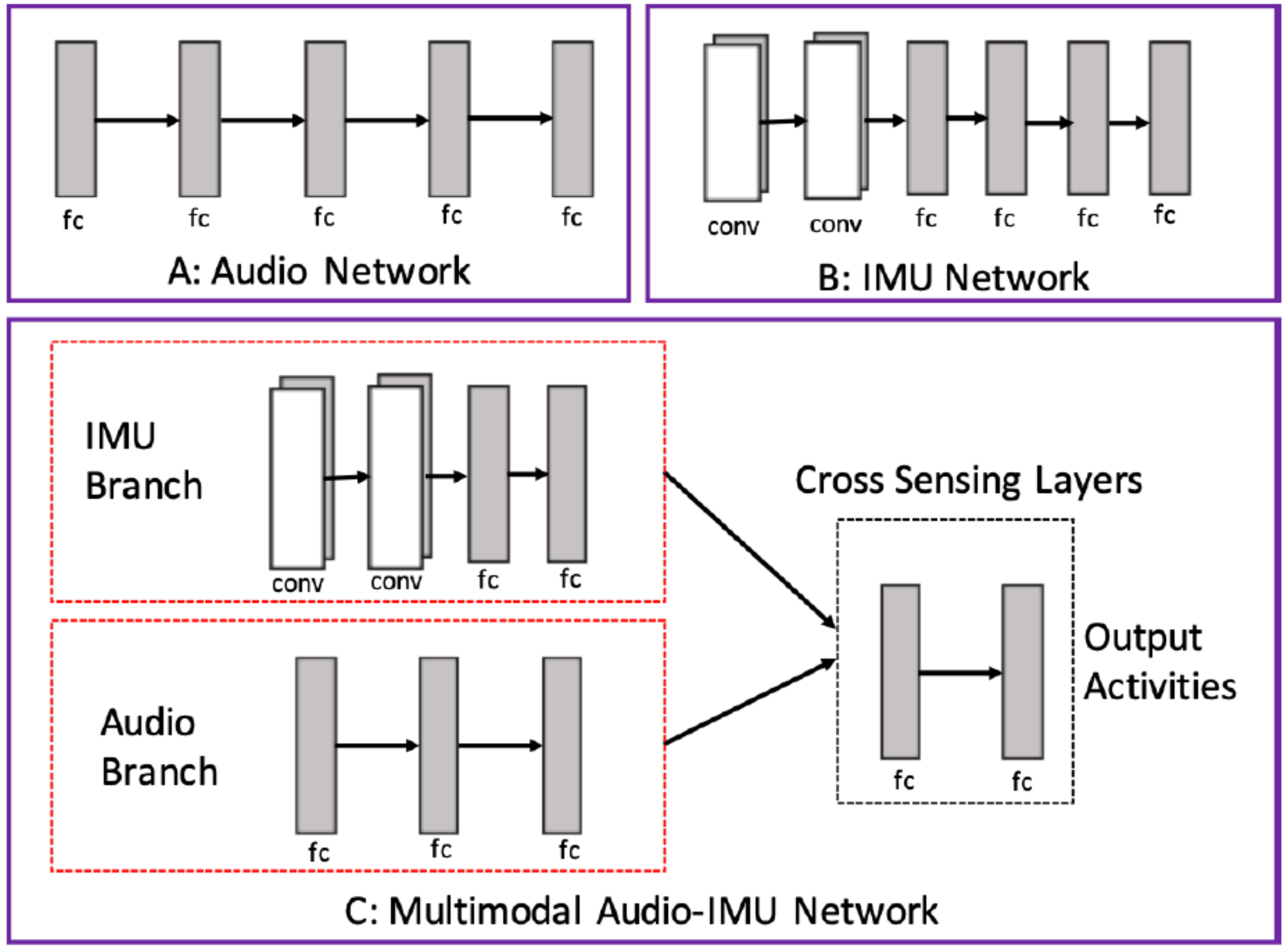
Audio from one smartphone and IMU from another
(Dataset: https://github.com/nesl/CMActivities-DataSet)

Baseline Accuracy

Networks	Audio	IMU	Multimodal Audio-IMU
Test Accuracy	91.34%	90.10%	96.12%

Multimodal fusion improves accuracy by ~5%.

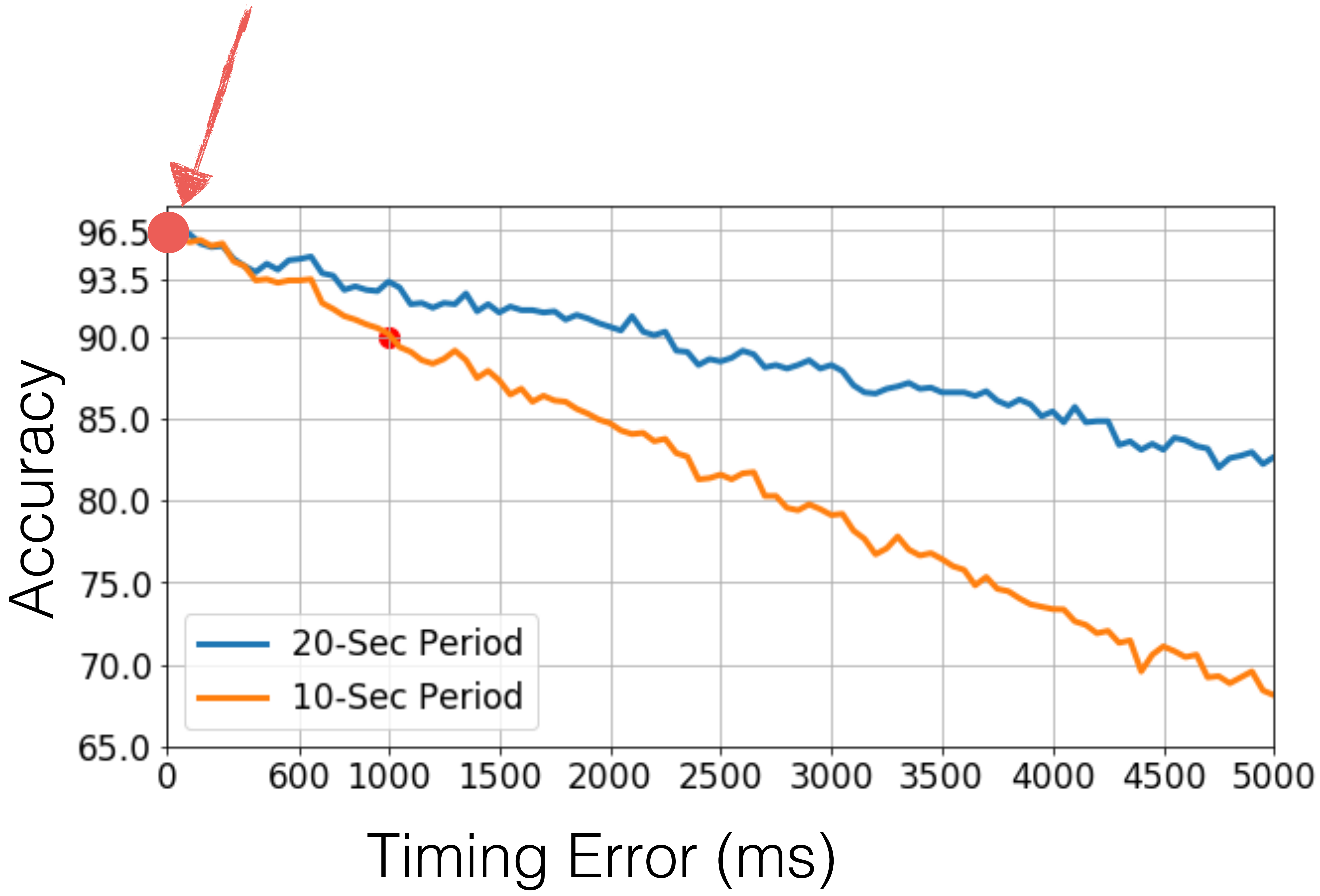
Feature level fusion [Ngiam11, Radu18]

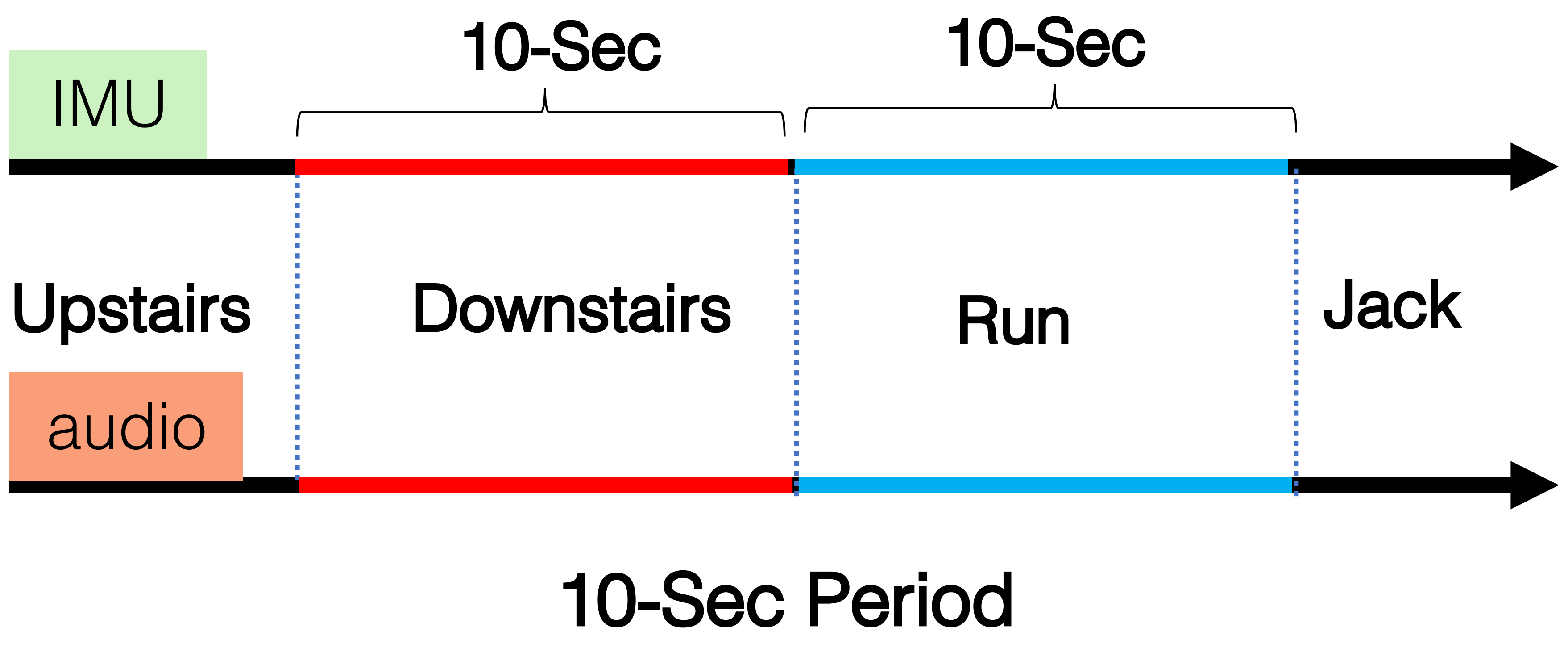


[Ngiam11] Ngiam, Jiquan, et al. "Multimodal deep learning." In International Conference on Machine Learning, 2011.
[Radu18] Radu, Valentin, et al. "Multimodal deep learning for activity and context recognition." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies, 2018.

Impact of Timing Errors on Multimodal Fusion Classifier

No time error 96.1% accuracy

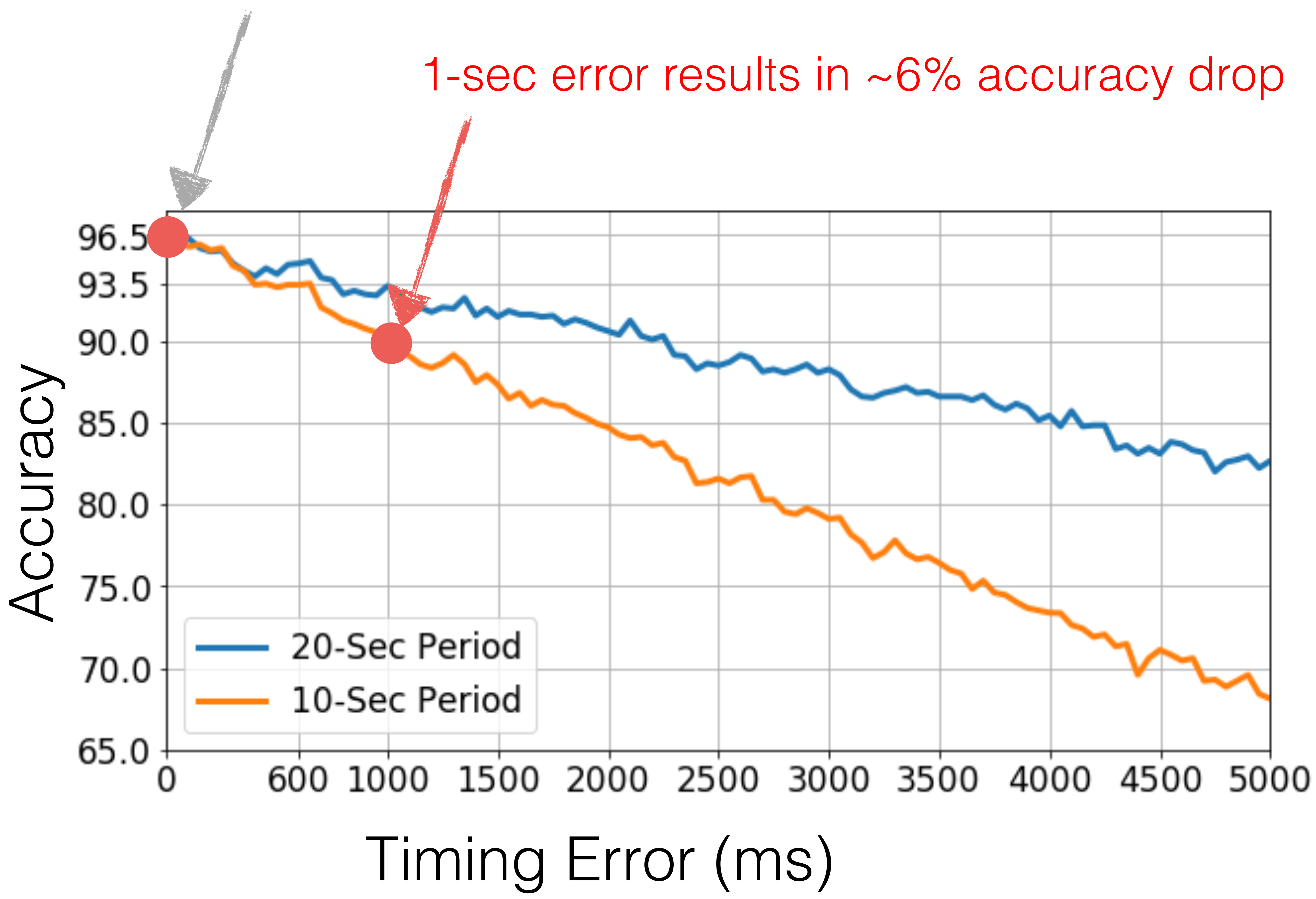


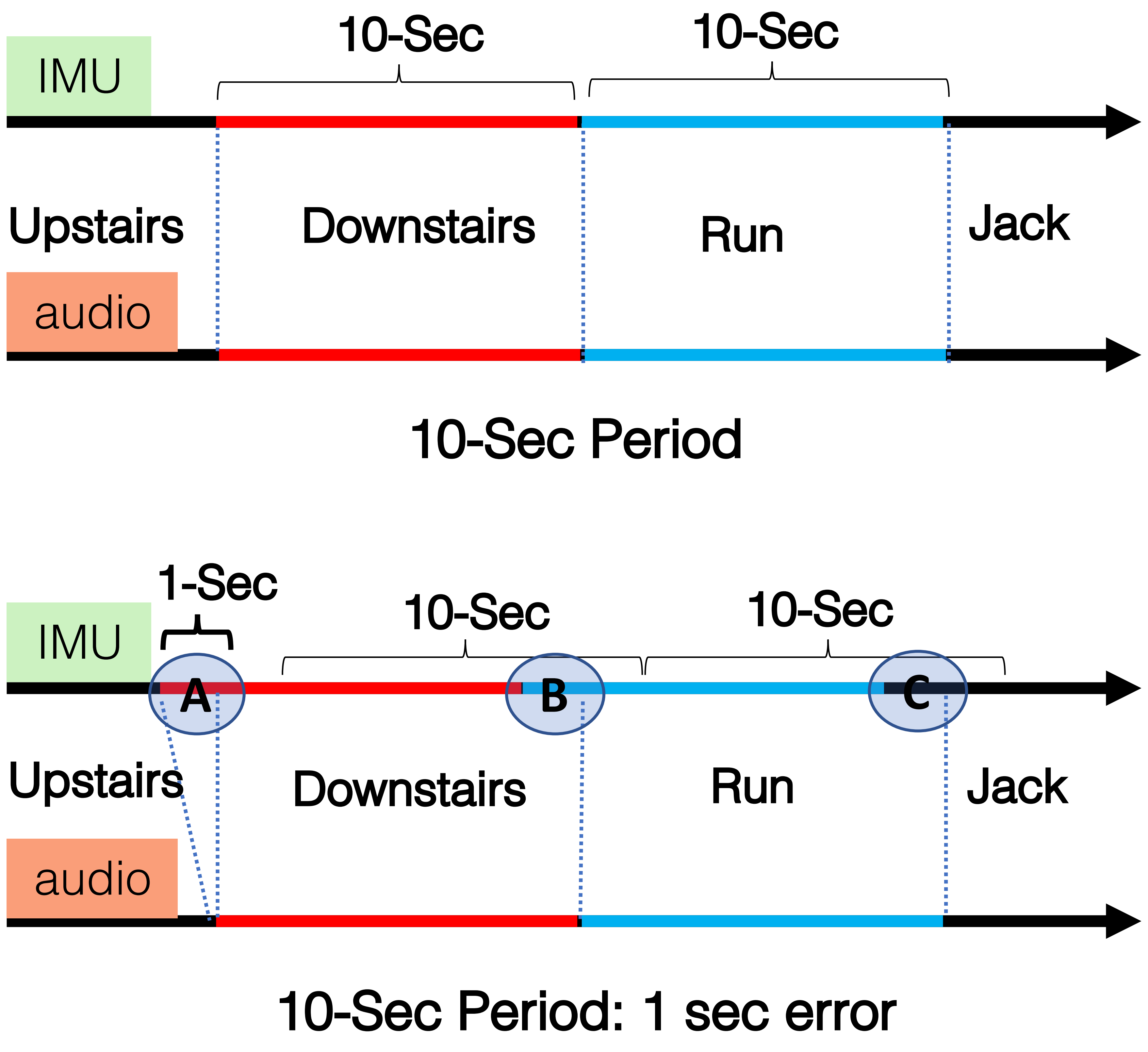


Impact of Timing Errors on Multimodal Fusion Classifier

No time error 96.1% accuracy

1-sec error results in ~6% accuracy drop

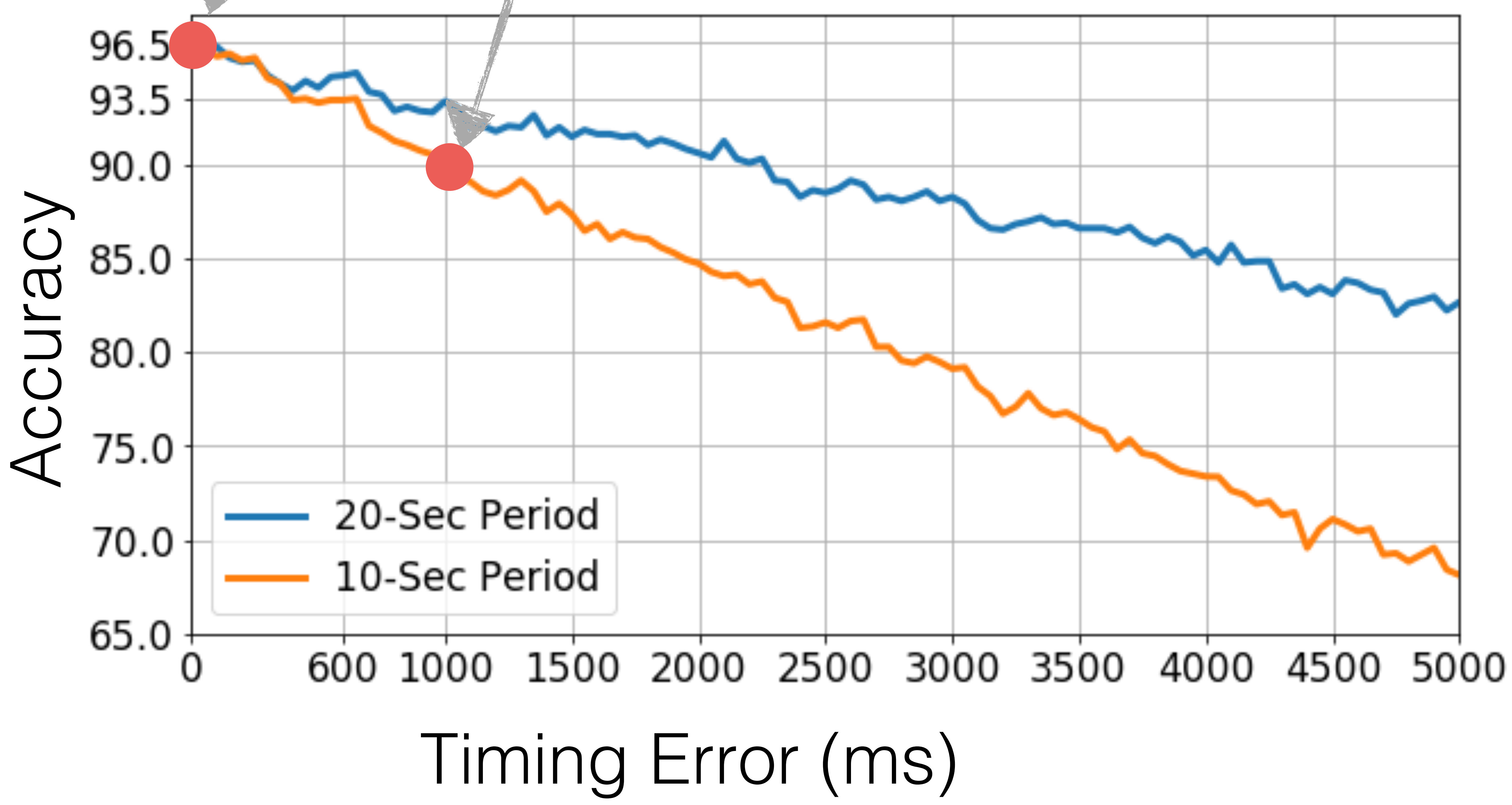




Impact of Timing Errors on Multimodal Fusion Classifier

No time error 96.1% accuracy

1-sec error results in ~6% accuracy drop



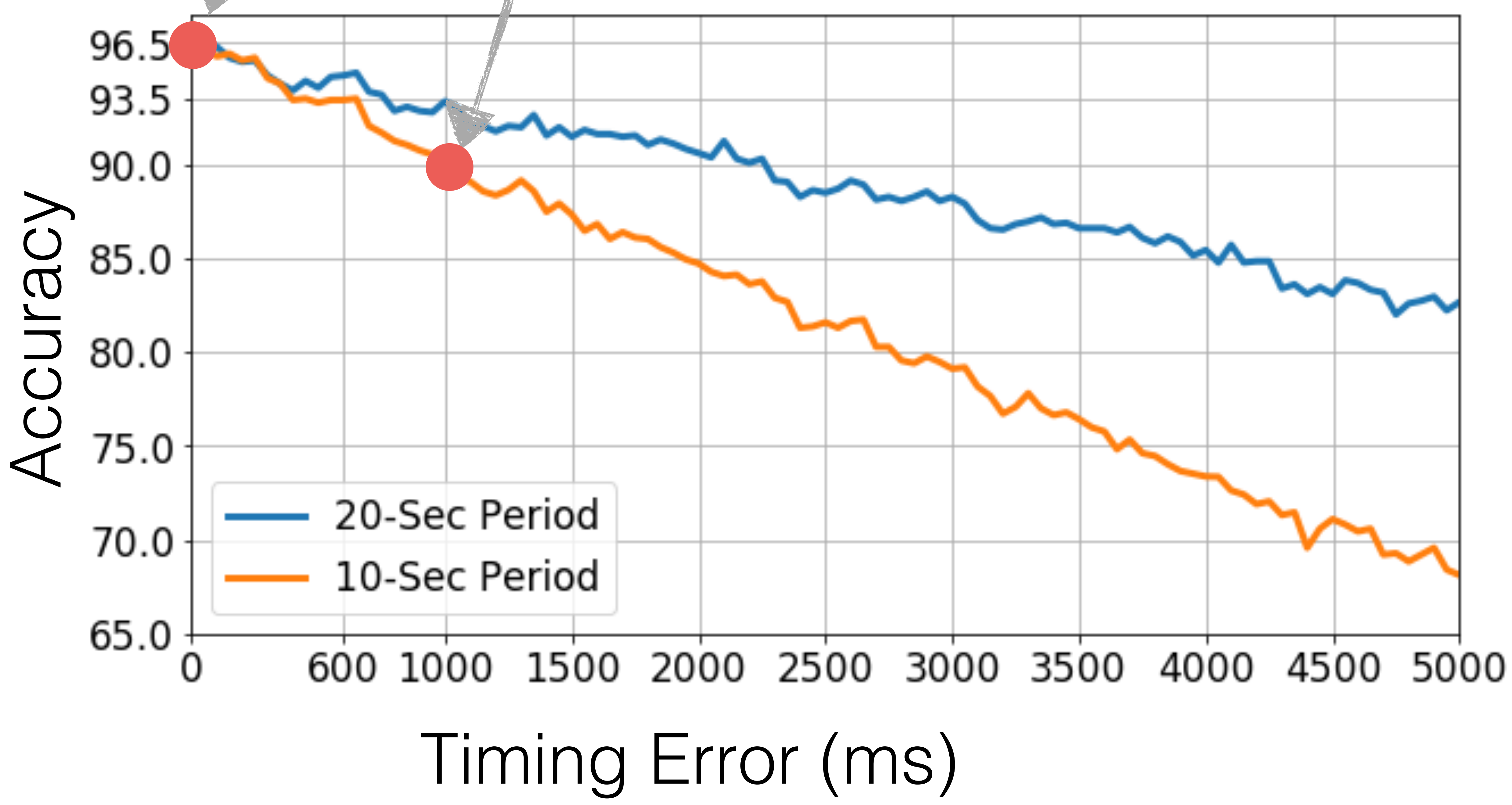
Time-Aware Fusion

1. **Improve** the quality of clock in and across devices
2. **Fix** the incorrect timestamps
3. **Modify training** pipeline for the imperfect timestamps

Impact of Timing Errors on Multimodal Fusion Classifier

No time error 96.1% accuracy

1-sec error results in ~6% accuracy drop

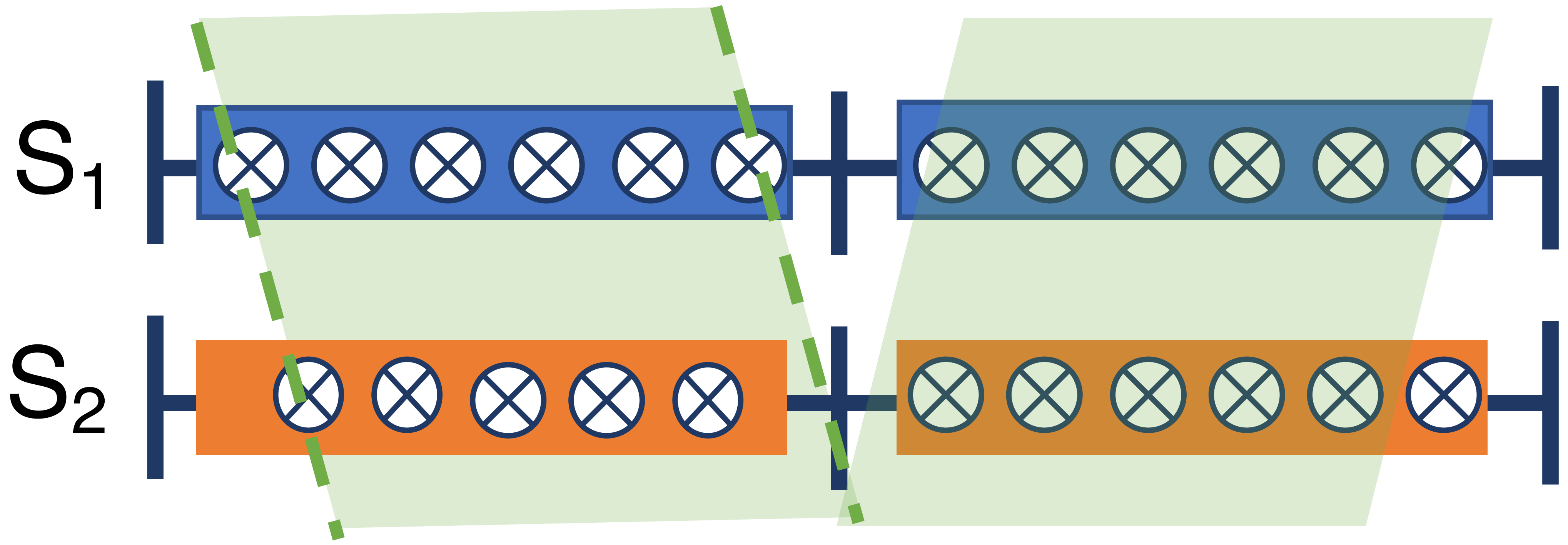


Time-Aware Fusion

1. **Improve** the quality of clock in and across devices
2. **Fix** the incorrect timestamps
3. **Modify training** pipeline for the imperfect timestamps

Time-Shift Data Augmentation

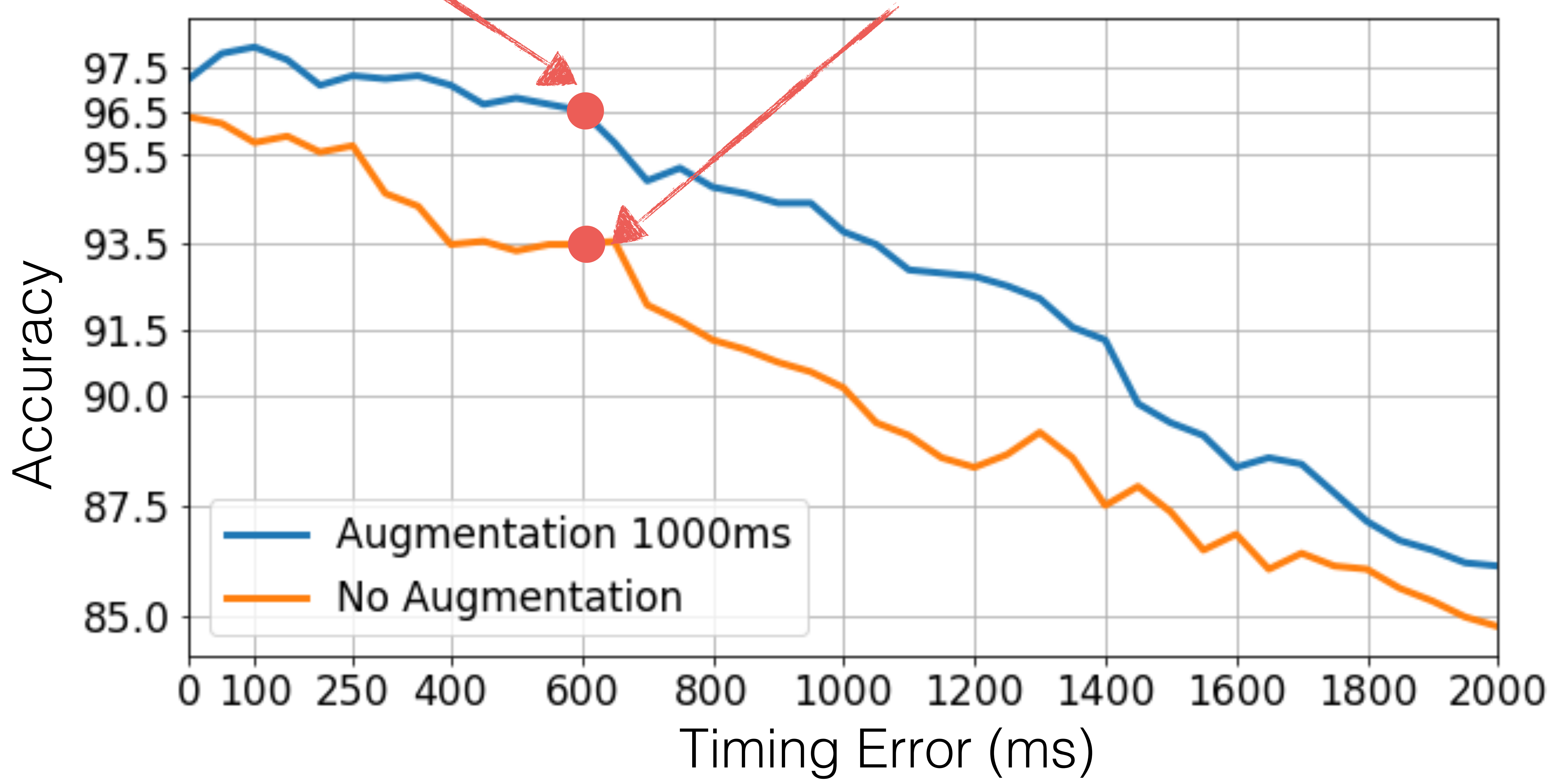
Idea: Add controlled artificial shifts during training (a form of *domain randomization*)



Time-Shift Data Augmentation

1000ms Augmentation can handle ~600ms time errors

No Augmentation: Accuracy drop by 3% with 600ms time errors

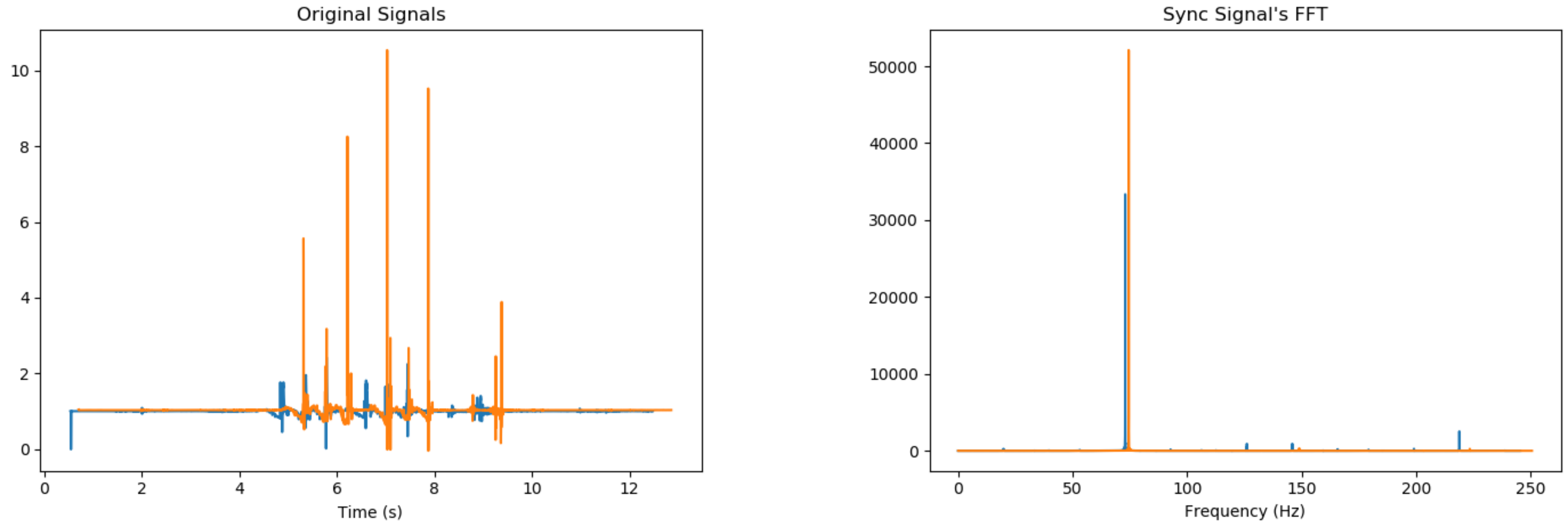


Time-Shift Data Augmentation

Idea: Add controlled artificial shifts during training

Setting: Up to 1000ms shift between modalities

Fixing Timestamps: Post-facto Synchronizing Signals

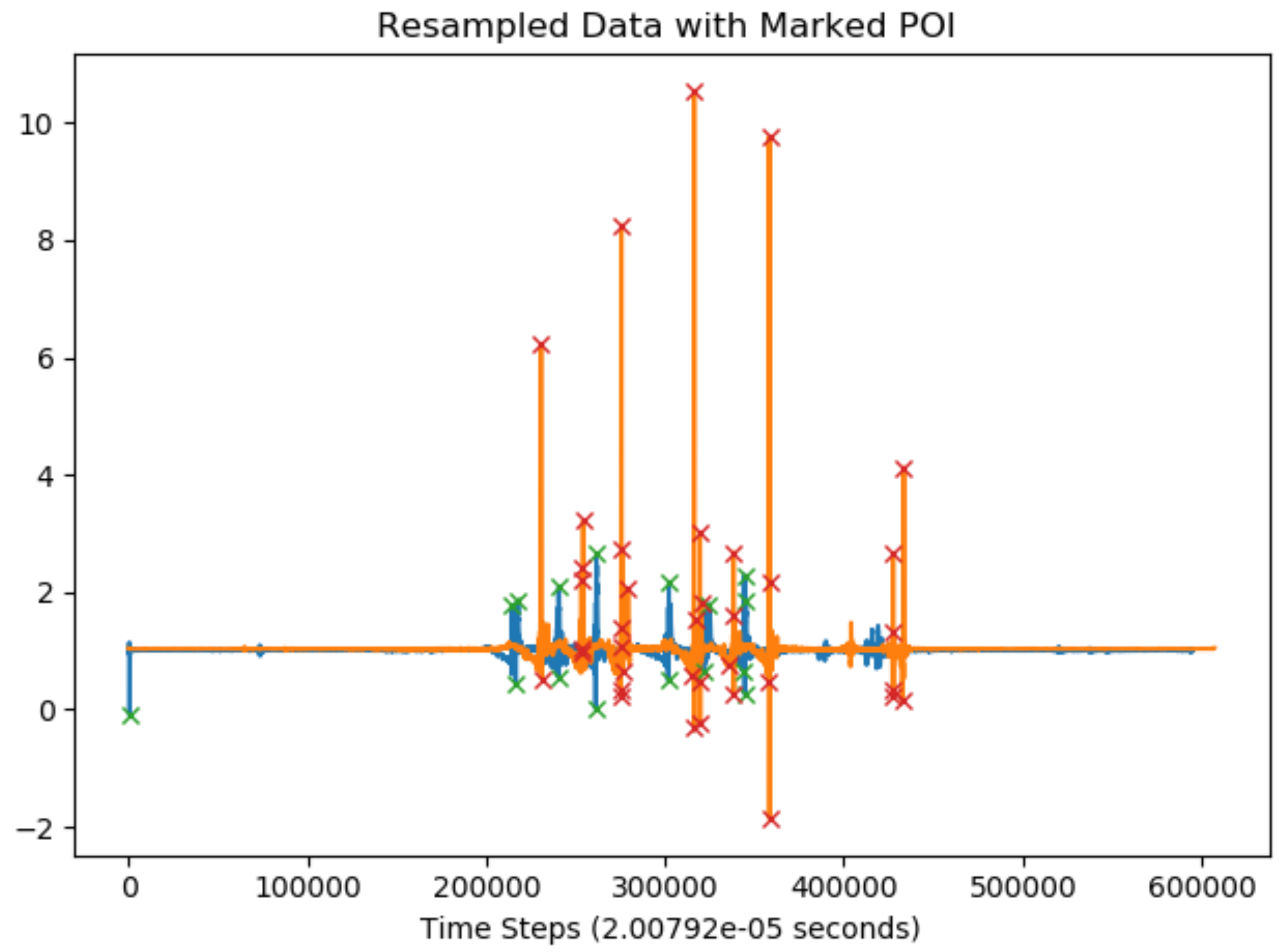


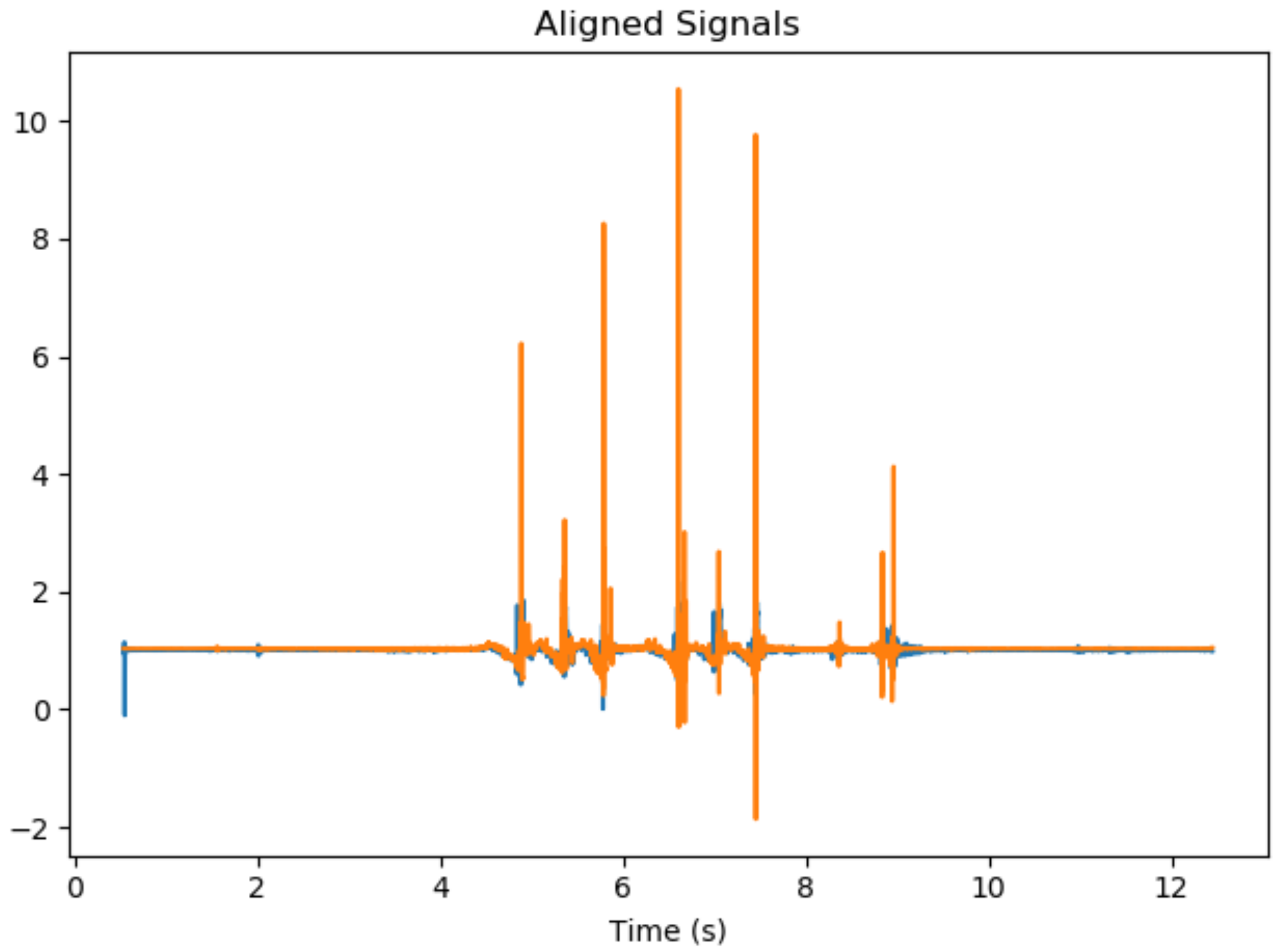
- Each sensor also produces a sine wave based on its local clock (sync signal)
- Receiver samples and records the sine wave from each source (sync data)
- Resample signals from different sources to a common sampling rate
  - comparing the frequencies of the sync signals by taking FFT of sync data
- What if the sources cannot provide a sync signal?
- Algorithmically find the optimal mutual offsets between the signals

https://blog.endaq.com/synchronizing-signals#AnalysisTechniques

Determining Offset

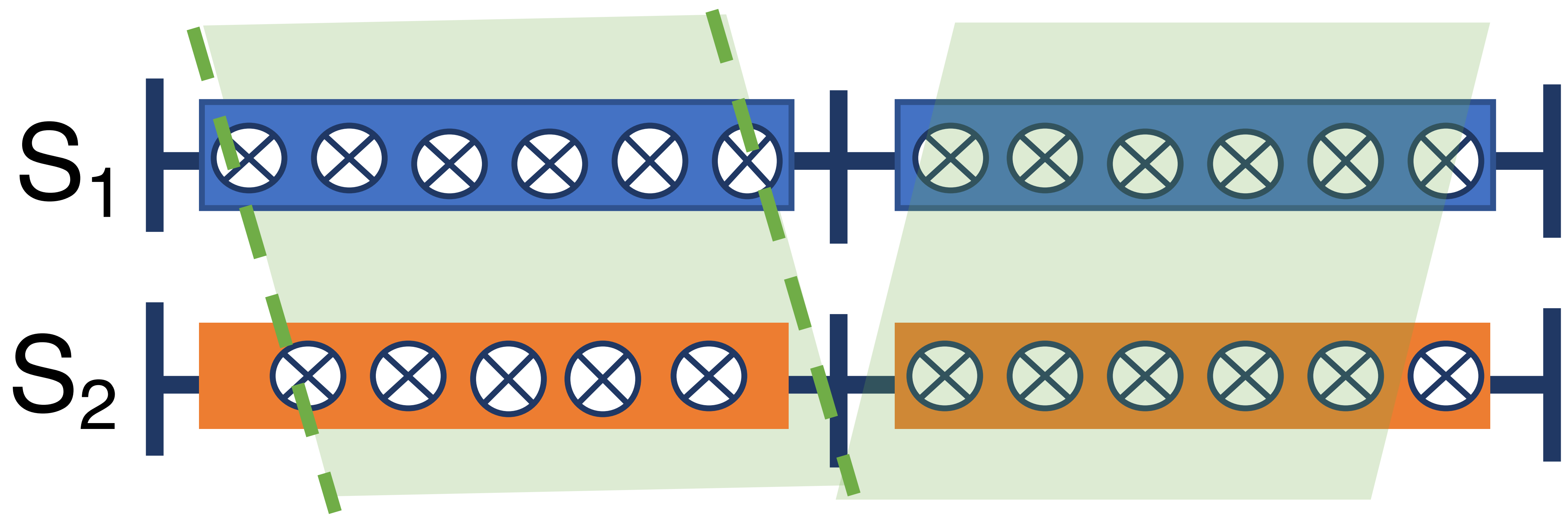
- Problem: Find the optimal offset for signal one $t \in (-\text{len}(signal1), \text{len}(signal2)) \subseteq \mathbb{Z}$ which best synchronizes the resampled signals by some metric (e.g. maximize normalized cross-correlation)
  - When synchronizing signals with a large number of samples, the search space for the optimal offset is large, and the computational cost of calculating how in-sync two signals are is high
- A plausible approach: synchronize 'major' events within the recordings, because these 'major' events should be recorded within both signals, and thus their offset should align the signals.
  - Generate a subset of the search space by taking 'points of interest' within each signal, and use the offsets which occur when choosing a pair of points (one from each signal) to align the signals





https://blog.endaq.com/synchronizing-signals#AnalysisTechniques

Another Application of Time-Shift Data Augmentation



Artificial shifts during training

Cooking Activity Recognition Challenge

- 4 accelerometers (2 wrist watches, 2 smartphones)
- 3 distinct macro and 10 distinct micro-activities

  Sandwich: cut, wash, take, put, other
  Fruit salad: cut, take, peel, add, mix, put, other
  Cereal: cut, take, pour, peel, put, open, other

- Data: 288 samples, each 30 seconds long

Accuracy	Without Time aug.	With Time aug.
Macro activity	77%	**83%**
Micro activity	48%	**72%**

Swapnil Sayan Saha, Sandeep Singh Sandha, Mani Srivastava, "Deep Convolutional Bidirectional LSTM for Complex Activity Recognition with Missing Data," Human Activity Recognition Challenge - Smart Innovations, Systems and Technologies, Ch. 4, Springer Singapore (2020).*

Research on Algorithmic Resynchronization of Multiple Time Series

- Plotz, Thomas, Chen Chen, Nils Y. Hammerla, and Gregory D. Abowd. "Automatic synchronization of wearable sensors and video-cameras for ground truth annotation--A practical approach." In 2012 16th international symposium on wearable computers, pp. 100-103. IEEE, 2012.
  - https://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6246150
- Chung, Joon Son, and Andrew Zisserman. "Out of time: automated lip sync in the wild." In Asian conference on computer vision, pp. 251-263. Springer, Cham, 2016.
  - https://ora.ox.ac.uk/objects/uuid:6bdd4768-6fbd-40ac-8efc-edca8a0325b3
- Fridman, Lex, Daniel E. Brown, William Angell, Irman Abdić, Bryan Reimer, and Hae Young Noh. "Automated synchronization of driving data using vibration and steering events." Pattern Recognition Letters 75 (2016): 9-15.
  - https://www.sciencedirect.com/science/article/pii/S0167865516000581
- Ahmed, Tousif, Mohsin Y. Ahmed, Md Mahbubur Rahman, Ebrahim Nemati, Bashima Islam, Korosh Vatanparvar, Viswam Nathan, Daniel McCaffrey, Jilong Kuang, and Jun Alex Gao. "Automated Time Synchronization of Cough Events from Multimodal Sensors in Mobile Devices." In Proceedings of the 2020 International Conference on Multimodal Interaction, pp. 614-619. 2020.
  - https://dl.acm.org/doi/pdf/10.1145/3382507.3418855
- Zhang, Yun C., Shibo Zhang, Miao Liu, Elyse Daly, Samuel Battalio, Santosh Kumar, Bonnie Spring, James M. Rehg, and Nabil Alshurafa. "SyncWISE: Window Induced Shift Estimation for Synchronization of Video and Accelerometry from Wearable Sensors." Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies 4, no. 3 (2020): 1-26.
  - https://dl.acm.org/doi/pdf/10.1145/3411824