6.S897/HST.956 Machine Learning for Healthcare

Lecture 6: Physiological time-series

Instructors: David Sontag, Peter Szolovits

1 Survival Modeling Continued

1.1 Recap of previous lecture

As discussed in the previous lecture, the goal of survival modeling is to predict the wait time until the occurrence of an event. Examples of events of interest include death, divorce, graduation, and recidivism. However, unlike typical supervised learning tasks, the data are right censored, meaning that the labels are not fully observed. Figure 1 shows an example of a right censored dataset.

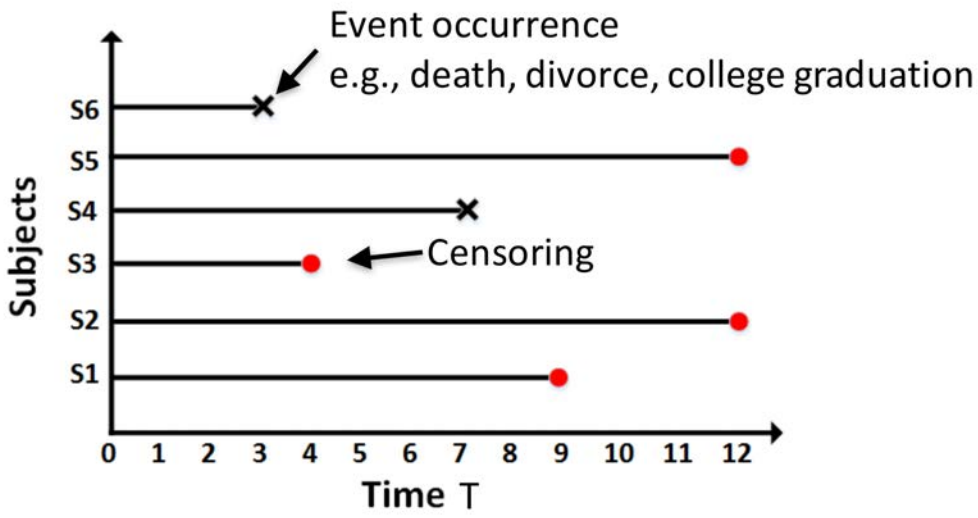


Figure 1: Each line represents a subject in the study. Black crosses represent the occurrence of an event, and red dots represent censored time points. In the case of a censored time point, all we know is that the event we actually care about occurred after the censored time point. From [WLR17]

Due to the right censored nature of the data, typical supervised learning approaches are unsatisfactory. For example, modeling this as a regression task would mean not using any of the censored datapoints. Not only is this data inefficient, but it may also result in a biased model, as censoring is not random.

1.2 Notation and formalization

In survival modeling, each datapoint can be represented as a tuple $(\mathbf{x}, T, b)$, where

- $\mathbf{x}$ is a vector of features
- T is the time of the event
- b is an binary indicator, which is 1 if the event is censored, and 0 otherwise

The main function we are interested in modeling is the death density $f(t)$, which can be interpreted as the probability of death at time t. It is also common to refer to the survival function, which is the probability of an individual surviving past time t, defined as $S(t) = P(T > t) = \int_t^\infty f(t)dt$. As we are interested in how the features $\mathbf{x}$ affect the survival function, it is often the case that $f(t) = f(t|\mathbf{x})$ is a function of $\mathbf{x}$ as well.

1.3 Learning Survival Models

1.3.1 Parameterization

In the last lecture, we discussed non-parametric methods for learning survival models, such as the Kaplan-Meier estimator. Now, we discuss parametric methods, which assume a specific form for the density $f(t)$. Table 2 shows commonly used forms for $f(t)$, and the corresponding survival function $S(t)$. These functions are non-negative and have long tails, which reflect the nature of wait times.

Distribution	Survival function $S(t)$	Density function $f(t)$
Exponential ($\lambda > 0$)	$\exp(-\lambda t)$	$\lambda \exp(-\lambda t)$
Weibull ($\lambda, \phi > 0$)	$\exp(-\lambda t^\phi)$	$\lambda\phi t^{\phi-1} \exp(-\lambda t^\phi)$
Log-normal ($\sigma > 0, \mu \in R$)	$1 - \Phi\{(\ln t - \mu)/\sigma\}$	$\varphi\{(\ln t - \mu)/\sigma\}(\sigma t)^{-1}$
Log-logistic ($\lambda > 0, \phi > 0$)	$1/(1 + \lambda t^\phi)$	$(\lambda\phi t^{\phi-1})/(1 + \lambda t^\phi)^2$
Gamma ($\lambda, \phi > 0$)	$1 - I(\lambda t, \phi)$	$\{\lambda^\phi / \Gamma(\phi)\} t^{\phi-1} \exp(-\lambda t)$
Gompertz ($\lambda, \phi > 0$)	$\exp\{\frac{\lambda}{\phi}(1 - e^{\phi t})\}$	$\lambda e^{\phi t} \exp\{\frac{\lambda}{\phi}(1 - e^{\phi t})\}$

Figure 2: Useful parametric distributions for survival models

Note that the densities $f(t)$ in the table are parameterized by λ, which we want to learn. It can be a function of the features $\mathbf{x}$, i.e. $\lambda = g(\mathbf{x})$, where g is any function, such as a deep neural network or a random forest.

1.3.2 Maximum Likelihood Estimation

Now that we have a functional form for the density, we can learn the parameters through maximum likelihood estimation. The likelihood of an uncensored event is $P_\theta(t|\mathbf{x}) = f(t)$, and the likelihood of a censored event is $P_\theta^{\text{censored}}(t|\mathbf{x}) = P_\theta(T > t|\mathbf{x}) = S(t)$. Putting the two together, we find that the maximum likelihood estimator is

$$\text{MLE} = \sum_{i=1}^{n} b_i \log P_\theta^{\text{censored}}(t|\mathbf{x}) + (1 - b_i) \log P_\theta(t|\mathbf{x}) \tag{1}$$

This objective can now be optimized using stochastic gradient ascent.

1.4 Evaluation Metrics

The primary metric used to evaluate survival models is the concordance index, otherwise known as the C-statistic. The idea of the concordance index is that a good model should rank individuals who survive longer higher than individuals who die sooner. To formulate this mathematically, consider the function produced by the survival model, $S(E)$, that gives a score to event E indicating the relative time at which E occurs. Then the concordance index is the probability that an event E_j, which occurs after event an event E_i, is ranked higher ($S(E_j) > S(E_i)$), where E_j is any event, censored or uncensored, and E_i is an uncensored event:

$$\hat{c} = \frac{1}{\text{num pairs}} \sum_{E_i \text{ uncensored}} \sum_{E_j > E_i} I[S(E_j) > S(E_i)] \tag{2}$$

The reason that E_i must be uncensored is because we don't know the true time of occurrence of an uncensored event. Thus it does not make sense to say that another event E_j occurs after an uncensored event.

Other metrics we may be interested in include

- mean squared error for uncensored individuals
- the likelihood of the censored (held out) data

2 Dealing with non-stationarity

We've talked about some of the issues of having non-stationary data, and about ways to detect it, but how can we actually build models that are robust to non-stationarity?

In the case that we have lots of data, one way to get around the issue is to train on only the most recent data i.e. the past three months. With such a short time between the train and test data, there is unlikely to be many non-stationary effects. However, this approach is not data efficient, and it is generally not the case that there is enough data for this approach to be useful.

There is a large amount of literature on the topic of how to best use historical data - here, we will discuss a few of the main approaches:

1. Impute or transform historical data to look like current data (e.g., [GUA+16], [FHS+17])

   The idea of this approach is to adjust the training data to look like the test data. A very simple example would be to convert ICD-9 codes in the training data to the ICD-10 codes used in the test data. Another example would be to develop a model to impute missing data in the training and test data as in [FHS+17]. This makes the model more robust to biases in the missing data.

2. Reweight historical data to look like current data (see [SK12])

   The idea of this approach is to up-weight the subsets of the training data that are more similar to the test data, and down-weight the subsets that are less similar.

3. Online algorithm that adapts quickly (see e.g. [BRJ+18])

   Many learning algorithms work with a static training set and treat all datapoints equally. These are called offline algorithms. An alternative is to use online algorithms which process data points in a given order, such as chronologically, and can train on new data as it comes in. These algorithms can be made to adapt quickly to changes in the data distribution and thus be robust to non-stationarity.

3 Physiological time-series

3.1 Overview

In the past, physiological time-series were prevalent mostly only in hospitals. For example, Figure 3 shows various physiological signals typically recorded from a baby born prematurely, such as temperature and heart rate. Now, however, physiological time-series are coming to consumers in the form of devices such as smartwatches, which may contain monitoring devices such as an EKG. Thus the ability to make inferences from physiological time-series is becoming more and more important.

Unfortunately, physiological time-series are often very noisy. A typical use case is then to infer the true signal from a noisy signal. These improved signals can then be used for downstream tasks such as risk stratification.

The approach to this problem depends heavily on the amount of labeled data that is available. In this lecture, we will see two opposite ends of the spectrum - in one case, there is almost no labeled data, and in the other, labeled data are plentiful and easy to acquire.

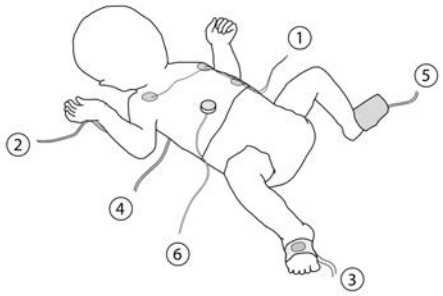


Fig. 4. Probes used to collect vital signs data from an infant in intensive care. 1) Three-lead ECG, 2) arterial line (connected to blood pressure transducer), 3) pulse oximeter, 4) core temperature probe (underneath shoulder blades), 5) peripheral temperature probe, 6) transcutaneous probe.

Figure 3: From [MQW08]

3.2 Neonatal Condition Monitoring

Our first case study is a case where there is very little labeled data. Clinicians in the ICU often use physiological time-series such as those in Figure 3 to monitor the condition of a prematurely born baby. They are typically able to identify the state of the neonate, normal or abnormal, based on the time-series data. We would like to create a computer algorithm to do the same, but the problem is complicated by artifacts in the data. These artifacts may look abnormal at first glance, but are actually not related to changes in the baby's state. For example, a clinician may temporarily remove a temperature probe, resulting in the temperature probe reading suddenly disappearing. This is then followed by an abnormally low reading when it is reconnected, as the probe has cooled off since disconnection. This situation is termed "dropout". Figure 4 shows the effect of dropout and various other confounding interventions.

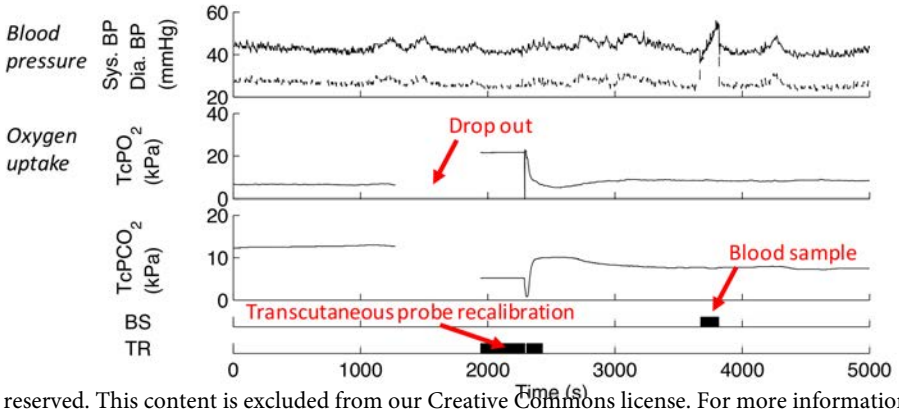


Figure 4: Effect of confounding interventions. From [MQW08].

If we can identify these artifactual processes, then we can remove them for use in a downstream task. An algorithm which makes use of this de-noised time series can then also reduce alarm fatigue, by not alerting clinicians unnecessarily. Next, we'll discuss one approach to this problem, taken by the authors of [MQW08].

3.3 (Switching) linear dynamical systems

3.3.1 Overview

As we have said, measurements that are observed (Y) can be noisy and don't always reflect the true state of a patient (X).

For instance, blood pressure measurements are extremely noisy due to the noise induced by the machines but also by the state of the person (is he/she tired, has he/she just smoked)?

$$\begin{aligned} \mathbf{x}_t &\sim \mathcal{N}\left(\mathbf{A}^{(s_t)}\mathbf{x}_{t-1} + \mathbf{d}^{(s_t)}, \mathbf{Q}^{(s_t)}\right) \\ \mathbf{y}_t &\sim \mathcal{N}\left(\mathbf{C}^{(s_t)}\mathbf{x}_t, \mathbf{R}^{(s_t)}\right) \end{aligned}$$

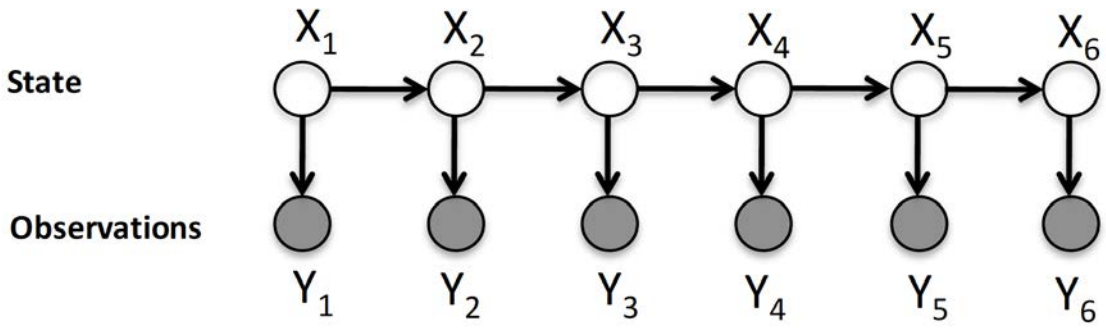


Figure 5: Kalman filters

Kalman filters can be used to understand the true state of a patient.

Kalman filters are a special case of (dynamic) Bayesians network. Given an observed Y caused by an X we try to infer X, and in Kalman filter we do this across time/different states.

We want to maximize the likelihood of observing X given Y.
In this example we would want to maximize:

$$P(X_1, X_2, X_3, X_4, X_5, X_6 | Y_1, Y_2, Y_3, Y_4, Y_5, Y_6) \tag{3}$$

We make an assumption that this joint distribution can be factorized:

$$P(X_1, X_2, X_3, X_4, X_5, X_6 | Y_1, Y_2, Y_3, Y_4, Y_5, Y_6) = P(X_1)P(Y_1|X_1)P(X_2|X_1)P(Y_2|X_2).... \tag{4}$$

Which allow us to characterize a complex distribution by multiple simple distributions.
The assumption made here is pretty strong and is true only for Markov Models, memory-less models where the next observation depends only on the previous observation and not the full history of observations:

$$\begin{aligned} &\forall n \geq 0, \forall (i_0, \ldots, i_{n-1}, i, j) \in E^{n+2} \\ &\mathbb{P}\left(X_{n+1} = j | X_0 = i_0, X_1 = i_1, \ldots, X_{n-1} = i_{n-1}, X_n = i\right) = \mathbb{P}\left(X_{n+1} = j | X_n = i\right) \end{aligned} \tag{5}$$

Another assumption made with Kalman filters is that:
X_t depends only on x_{t-1}, with a gaussian distribution and a fixed covariance matrix and y_t only on x_t with another fixed covariance matrix:

$$\begin{aligned}\mathbf{x}_t &\sim \mathcal{N}\left(\mathbf{A}^{(s_t)}\mathbf{x}_{t-1} + \mathbf{d}^{(s_t)}, \mathbf{Q}^{(s_t)}\right) \\ \mathbf{y}_t &\sim \mathcal{N}\left(\mathbf{C}^{(s_t)}\mathbf{x}_t, \mathbf{R}^{(s_t)}\right)\end{aligned} \tag{6}$$

Y_1 should be very close to X_1 if it is a good observation. If it's noisy it's not going to be case (e.g.: if the prob disconnects then there is no correlation between y_t and x_t).

In order to better understand these connections between X and Y, and just how X can also be affected by external factors it's useful to add confounding factors (e.g.: artifactual events).

In that case the full model is the following:

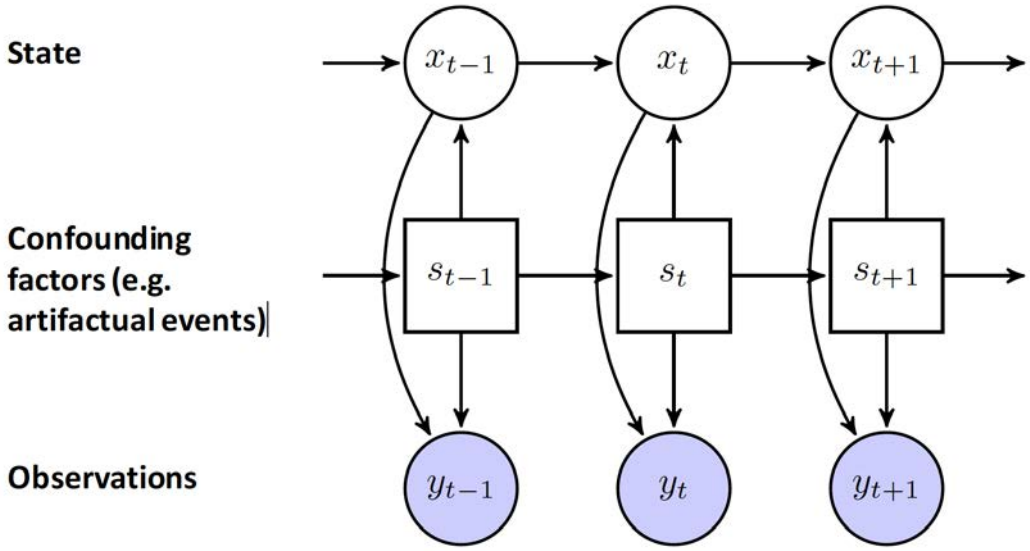


Figure 6: Full Kalman filters

Here **s** are others random event denoting artifactual events. To get more context on this: [PL15]

When running these models you have to ask yourself what type of data do you have.

3.3.2 Learning SLDS models

If we follow the BP example again: we only get to observe the very noisy BP point across time. We have to think on **X** and **s** as latent variables and we want to maximize their likelihood.

However it's often a non-convex problem, very hard to solve.

We cannot use the gradient descent algorithm (it would only converge to a local optimum). We can use instead the EM (Estimation-Maximization) algorithm.

- We have to assume that we have some labeled training data :$\{s, y\}$
- The true state x is assumed to never be observed
- Learn using the expectation maximization algorithm

As we observe only y, we can wonder will this algorithm ever recover the true state?
It's why it's essential to bring in domain knowledge.
With this domain, we then try to constrain the model as much as we can.

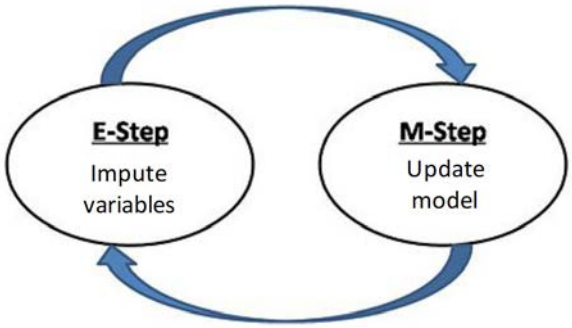


Figure 7: Estimation-Maximization algorithm

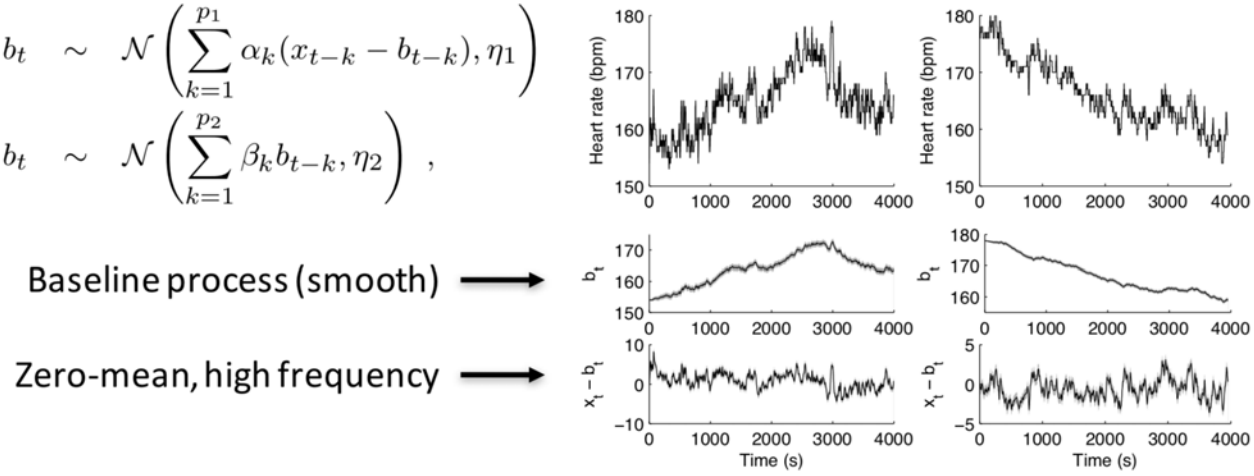
3.3.3 Parameterizing model

Problem: Indeed, if we leave to many degrees of liberty to the model (as in Neural Networks), it will be harder to learn complex relations with few data points. The model is maybe more likely to overfit.

Solution: We can use domain knowledge to help parameterize the model. For instance, normal heart rate (HR) dynamics are well-modeled using autoregressive (AR) processes:

$$x_t - b_t \sim \mathcal{N}\left(\sum_{k=1}^{p_1} \alpha_k (x_{t-k} - b_{t-k}), \eta_1\right)$$

$$b_t \sim \mathcal{N}\left(\sum_{k=1}^{p_2} \beta_k b_{t-k}, \eta_2\right) ,$$



(Quinn et al., TPAMI 2008)

Figure 8: Heart rate denoised

The true signal is basically what you observe when you squint your eyes, when you denoise it.
The noise signal has a zero mean. The sum of b(t) and the residual corresponds to the exact HR signal.
We can model both by random walk, going slowly (for the true signal) and quickly for the error/noise.
Please refer to the paper [MQW08].

Problem: There should not be any dependence between y and x after some time. However, we don't how fast this dependence will be removed. For instance, a thermometer takes time to cool. Know how fast it cools would be really helpful.

Solution: Another way to use domain knowledge is to specify specify parts of the artifacts model:

- Probe dropouts modeled by removing dependence of observation y_t on patient state x_t
- Temperature probe disconnection: exponential decay to room temperature

3.3.4 Evaluation

It's pretty difficult to evaluate these models. We have to assume that the first 30min consists of a normal distribution (with no artifact).

Then we can look at the ROC curve of the ability to predict each artifact (when a blood sample was taken, a probe disconnected).

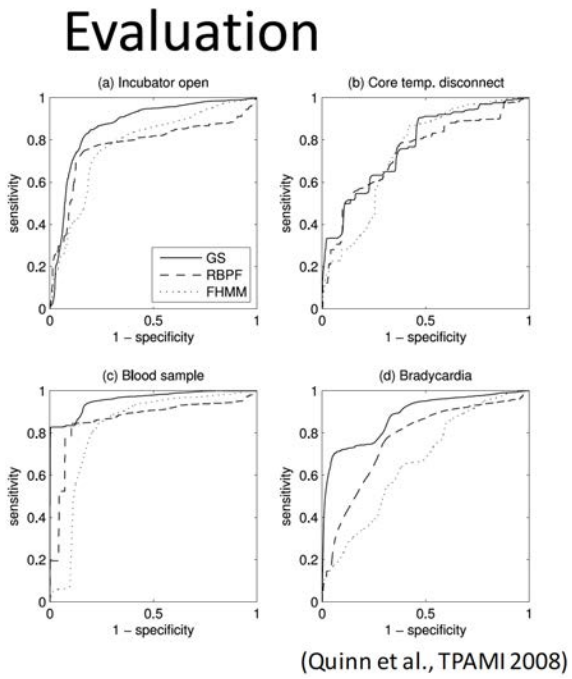


Figure 9: Evaluation ROC curve

In this graph we compare 3 different approaches:

- GS = Gaussian-sum approximation (used for inference)
- RBPF = Rao-Blackellized particle filtering approximation (used for inference)
- FHMM = Factorial HMM (simpler model not modeling normal physiologics dynamic)

We can see that the GS, which is deterministic, outperforms the other methods.
The black bar on the time represent when the blood sample was taken or when the probe was disconnected.

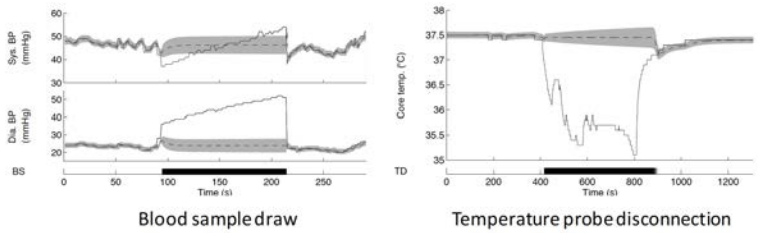


Figure 10: Inference of physiological state

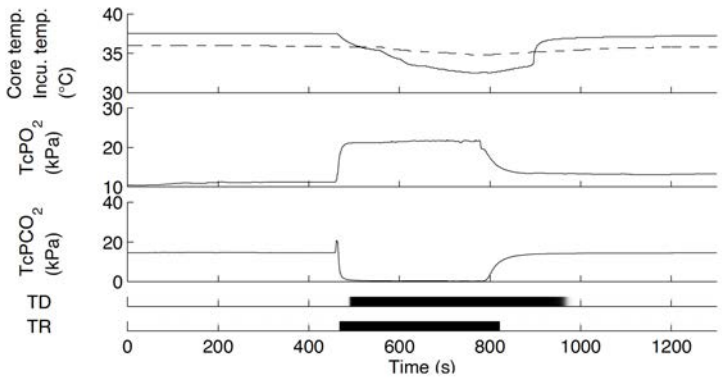


Figure 11: Inferred switch settings

3.4 Detecting Atrial Fibrillation

3.4.1 Description of the problem

Our second case study is a case where labeled data is plentiful. Atrial fibrillation (AF) is the irregular beating of the heart, which can lead to problems such as blood clots, heart failure, strokes, and other illnesses. One way AF can be detected is by using ECG recordings. Figure 12 shows ECG recordings for normal heart rhythms and AF heart rhythms.

In the past, data was less plentiful and the best algorithms for detecting AF using ECG time series data were based on a combination of hand engineered features and machine learning. This was true as recently as 2017, when Clifford et al. held the PhysioNet challenge for AF Classification from a short single lead ECG recording [CLM+17]. Figures 13 and 19 give a sense of the kinds of hand engineered features used.

It might seem surprising that no CNN approaches worked. For the 2017 Physionet challenge we had only 8500 ECGs, maybe it was not enough...

"With so many parameters and hyperparameters to tune, the search space can be enormous and significant overtraining was seen" [CLM+17]

As it turns out, with a much larger dataset, we can avoid using hand engineered features, and use more expressive machine learning models such as convolutional neural networks.

3.5 Atrial fibrillation detection using convolution neural networks

The paper [PR19] "develop a model which can diagnose irregular heart rythms (arrhythmias) from single-lead ECG signals better than a cardiologist.".

3.5.1 Differences with previous work

Even if the model performs really well we have to outline a few differences with previous work:

- The sensor used was less noisy
- More ECG (90K ECG records annotated from 50K patients)
- This model identify 12 heart arrhythmias, sinus rhythm and noise for a total of 14 output classes (versus 4)
- The data was better (they look where the errors where and then they got more people with these errors)

Table 1: Set of features used to train the global classifier

`tSR`: Proportion of the record length interpreted as a regular rhythm (Normal rhythm, tachycardia or bradycardia).	`t1b`: Number of milliseconds from the beginning of the record to the first interpreted heartbeat.
`tOR`: Number of milliseconds interpreted as a non-regular rhythm.	`longTch`: Longest period of time with heart rate over 100bpm.
`RR`: Median RR interval of regular rhythms.	`RRd_std`: Standard deviation of the instant RR variation.
`RRd`: Median Absolute Deviation (MAD) of the RR interval in regular rhythms.	`MRRd`: Max. absolute variation of the RR interval in regular rhythms.
`RR_MIrr`: Max. RR irregularity measure.	`RR_Irr`: Median RR irregularity measure.
`PNN{10,50,100}`: Global PNNx measures.	`o_PNN50`: PNN50 of non-regular rhythms.
`mRR`: Min. RR interval of regular rhythms.	`o_mRR`: Min. RR interval of non-regular rhythms.
`n_nP`: Proportion of heartbeats with detected P-wave inside regular rhythms.	`n_aT`: Median of the amplitude of the T waves inside regular rhythms.
`n_PR`: Median PR duration inside regular rhythms.	`Psmooth`: Median of the ratio between the standard deviation and the mean value of P-waves' derivative signal.
`Pdistd`: MAD of the measure given by the P wave delineation method.	`MPdist`: Max. of the measure given by the P wave delineation method.
`prof`: Profile of the full signal.	`pw_profd`: MAD of `pw_prof`.
`xcorr`: Median of the maximum cross-correlation between QRS complexes interpreted in regular rhythms.	`o_xcorr`: Median of the maximum cross-correlation between QRS complexes interpreted in non-regular rhythms.
`PRd`: Global MAD of the PR durations.	`QT`: Median of the corrected QT measure.
`TP`: Median of the prevailing frequency in the TP intervals.	`TPfreq`: Median of the frequency entropy in the TP intervals.
`pw_prof`: Profile measure of the signal in the P-wave area.	`nT`: Proportion of QRS complexes with detected T waves.
`n_Txcorr`: Median of the maximum cross-correlation between T-waves inside regular rhythms.	`n_Pxcorr`: Median of the maximum cross-correlation between P-waves inside regular rhythms.
`baseline`: Profile of the baseline in regular rhythms.	`o_baseline`: Profile of the baseline in non-regular rhythms.
`wQRS`: Proportion of wide QRS complexes (duration longer than 110ms).	`wQRS_xc`: Median of the maximum cross-correlation between wide QRS complexes.
`wQRS_prof`: Median of the signal profile in the 300ms before each wide QRS complex.	`w_PR`: Proportion of heartbeats with long PR interval (longer than 210 ms).
`x_xc`: Median of the maximum cross-correlation between ectopic beats.	`x_rrel`: Median of the ratio between the previous and next RR intervals for each ectopic beat.

Figure 12: Descriptions of hand engineered features. From [TGCF18]

3.5.2 Architecture of the model used

"While artificial neural networks were first applied toward the interpretation of ECGs as early as two decades ago, until recently they only contained several layers and were constrained by algorithmic and computational limitations. More recent studies have employed deeper networks, although some only use DNNs to perform certain steps in the ECG processing pipeline, such as feature extraction33 or classification25. End-to-end DNN approaches have been used more recently showing good performance for a limited set of ECG rhythms, such as atrial fibrillation". [**?**]

Here the network is deeper network (34 layers), and thus a bit more a black box. This study "demonstrates that the paradigm shift represented by end-to-end deep learning may enable a new approach to automated ECG analysis."

The shortcuts connection allow the "information" to be passed quickly (similarly than in residual neural network).

The convolution layers apply a filter (a matrix, for instance 3x3) to all possible combination of matrix 3x3 in the input, apply an element-wise multiplication and then summing all the values.

3.5.3 Evaluation

In order to evaluate the model, they used a sequential evaluation= how good are we at labeling point (normal/abnormal) across a sequence of point in time.

It's interesting to notice that the performances of the model is nearly always as good as the performances of the cardiologist:

(In bold you can observe the best performances between cardiologist and the model).

"The average F1 score, which is the harmonic mean of the positive predictive value and sensitivity, for the DNN (0.837) exceeded that of average cardiologists (0.780). With specificity fixed at the average specificity achieved by cardiologists, the sensitivity of the DNN exceeded the average cardiologist sensitivity for all rhythm classes. These findings demonstrate that an endto- end deep learning approach can classify a broad range of distinct arrhythmias from single-lead ECGs with high diagnostic performance similar to that of cardiologists."

This is an example of model that has been deployed (apple watch works with a similar algorithm).

On the "ROC and precision-recall curves comparison" graph we can observe that for a given sensitivity the model nearly always outperforms cardiologist.

What is even more interesting is that the model seems to be making similar errors as the cardiologists.

4 Summary

- The lack of data is always there, and it has a huge impact of deep learning approaches
- Deeper, end-to-end/ black box, neural network performs better and better, especially when they are fed more data.
- However, modeling and incorporating prior domain knowledge remains critical to obtain good performance

A few design principles
- Model the distribution of physiological dynamics
- Derive features using existing clinical knowledge
- Start from the simplest possible model
- Share statistical strength across tasks

References

[BRJ+18] Spyridon Bakas, Mauricio Reyes, Andras Jakab, Stefan Bauer, Markus Rempfler, Alessandro Crimi, Russell Takeshi Shinohara, Christoph Berger, Sung Min Ha, Martin Rozycki, et al. Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge. *arXiv preprint arXiv:1811.02629*, 2018.

[CLM+17] Gari D Clifford, Chengyu Liu, Benjamin Moody, H Lehman Li-wei, Ikaro Silva, Qiao Li, AE Johnson, and Roger G Mark. Af classification from a short single lead ecg recording: the physionet/computing in cardiology challenge 2017. In *2017 Computing in Cardiology (CinC)*, pages 1–4. IEEE, 2017.

[FHS+17] Joseph Futoma, Sanjay Hariharan, Mark Sendak, Nathan Brajer, Meredith Clement, Armando Bedoya, Cara O'Brien, and Katherine Heller. An improved multi-output gaussian process rnn with real-time validation for early sepsis detection. *arXiv preprint arXiv:1708.05894*, 2017.

[GUA+16] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[MQW08] N. McIntosh, J. A. Quinn, and C. K. Williams. Factorial switching linear dynamical systems applied to physiological condition monitoring. *IEEE Transactions on Pattern Analysis Machine Intelligence*, 31:1537–1551, 07 2008.

[PL15] Konstantinos Georgatzis Christopher Hawthorne Paul McMonagle Ian Piper Martin Shaw Partha Lal, Christopher K. I. Williams. Detecting artifactual events in vital signs monitoring data. *Machine Learning for Healthcare Technologies*, 2015.

[PR19] Masoumeh Haghpanahi Codie Bourn Andrew Y. Ng Pranav Rajpurkar, Awni Y. Hannun. Cardiologist-level arrhythmia detection with convolutional neural networks. *Nature Medicine*, 2019.

[SK12] Masashi Sugiyama and Motoaki Kawanabe. *Machine Learning in Non-Stationary Environments: Introduction to Covariate Shift Adaptation*. The MIT Press, 2012.

[TGCF18] Tomás Teijeiro, Constantino A García, Daniel Castro, and Paulo Félix. Abductive reasoning as a basis to reproduce expert criteria in ecg atrial fibrillation identification. *Physiological measurement*, 39(8):084006, 2018.

[WLR17] Ping Wang, Yan Li, and Chandan K Reddy. Machine learning for survival analysis: A survey. *arXiv preprint arXiv:1708.04649*, 2017.

6.S897 / HST.956 Machine Learning for Healthcare

Spring 2019