# Adverse Event Prediction by Telemonitoring and Deep Learning

A. Prouvost, A. Lodi, L.-M. Rousseau, and J. Vallee

**Abstract** Home health care comes as a potential solution to increasing stress on health-care systems, as well as concerns for medical patients comfort. However, additional distance from the care workers to the patients lead to more challenges, some of which can be addressed with machine learning (ML) and operations research (OR) algorithms. In this paper, we focus on automating a risk assessment of remote patients. Namely, we describe a risk prediction framework for home telemonitoring patients and show that learning a risk from weak signals in the patient's data outperforms simple risk threshold proposed by care workers to automate the task. We combine recurrent neural networks with a ranking objective from survival analysis to evaluate the risk of patient's adverse events. Training and testing of our methodology is achieved on a retrospective dataset gathered by an Ontario home health care agency during the course of a multi-year pilot home telemonitoring program. Results are benchmarked against alerts that were manually engineered by registered nurses, and against a simple linear baseline. This is an additional step in the application of machine learning in health care for patient-centered personalized treatments.

Antoine Prouvost
Canada Excellence Research Chair
École Polytechnique de Montréal, e-mail: antoine.prouvost@polymtl.ca

Andrea Lodi
Canada Excellence Research Chair
École Polytechnique de Montréal, e-mail: andrea.lodi@polymtl.ca

Louis-Martin Rousseau
École Polytechnique de Montréal, e-mail: louis-martin.rousseau@polymtl.ca

Jonathan Vallee
AlayaCAre, e-mail: jonathan.vallee@alayacare.com

# 1 Introduction

Population ageing comes with increased care needs since 85% of elderly will develop chronic conditions [Ward et al., 2014]. From 6.9% in 2000 to an estimated proportion of 19.3% of the global population in 2050 [Gavrilov and Heuveline, 2003], the elderly account for a growing proportion of the health care costs.

Keeping elderly healthy and at home longer is thus a critical endeavour. Home Health Care (HHC) is starting to be widely adopted since it is seen as a cost effective alternative to traditional care and because patients often prefer it.

Home telemonitoring (HT), a specialized form of HHC is a potential alternative that empowers patients to take charge of their health, generates reliable data that can be leveraged to better assess the patients' states and that may improve the patient's medical condition [Paré et al., 2007].

Given the growing demand for HHC and HT, data is accumulating at an extreme velocity, in a great volume and in a variety of forms. The advancements in monitoring devices is also contributing to the velocity and volume of data generated by HT programs. Valuable information lies in this data. There is thus a pressing need for improved decision systems that can use the information.

When a patient is admitted to a home care agency, she generally gets visited by a registered nurse who will perform an initial needs assessment [Rasmussen et al., 2012]. If the agency offers a HT program, patients can be admitted to it. While on a HT program, the patient answers a periodic questionnaire during which she will be asked to take some vital signs readings. This information is then transmitted to the HHC agency where a nurse monitors a HT case load. Based on the patient diagnosed conditions and initial assessment, the care workers create alerts based on acceptable range of each of the measured vital signs as shown in [Suh et al., 2010]. Sometimes, with more advanced systems, complex rules can be developed to get alerted based on combinations of suspect readings.

In all cases, care workers bear the weight of setting up patients with the right set of alerts based on their conditions. The manually engineered rules then need to evolve with the patient's condition in order to remain reliable. When a vital sign reading is out of the acceptable range, the monitoring nurse can perform one or two of the following actions: (1) call the patient to determine next steps, and/or (2) schedule an in-person visit. The challenge is to prevent costly hospital readmissions and emergency room visits, but there is also a cost to each intervention. To add complexity, most of the alarms are false positives, not leading to adverse events.

Early detection of these events serves the purpose of the triple aim of improving outcomes: (1) quality of health services, (2) improving health of populations and (3) reducing costs [Berwick et al., 2008].

The contribution of this paper is to propose a patient ranking approach to adverse events prediction and to show that it performs well on a retrospective patient cohort.

The remainder of the paper is organized as follows: In Section 2, we review the body of work related to adverse events predictions. In Section 3, we review the technical background required for our proposed solution to adverse events predic-

tion. Section 4 details our proposed framework and Section 5 reports our results. We conclude the paper with a discussion and perspectives in Section 6.

## 2 Adverse Event Prediction Related Work

Since about 20% of Medicare patients are readmitted within 30 days of discharge [Jencks et al., 2009] and since [Huntington et al., 2011] established financial penalties to hospital with the highest readmission rates 30 days after discharge, prediction of adverse events such as hospital readmissions has been extensively done in health care research.

Linear models such as multivariate logistic regression and Cox Proportional Hazard [Ross et al., 2008, Hansen et al., 2011, Wallace et al., 2014, Kansagara et al., 2011] are often used because of their understandable nature. Indeed, most of the work so far has been interested in understanding the significant factors that lead to adverse events. Conversely, neural networks have not been used as much because they are seen as hard to interpret black-boxes [Zhu et al., 2014] despite their success in many industries, from computer vision to market finance.

In health care, some examples of neural network use are as diagnostic tool such as in [Baxt et al., 2002], as prediction tools in [Harrison and Kennedy, 2005, Luck et al., 2017], in emergency states detectors [Swiercz et al., 1998], and in psychology [Price et al., 2000]. In particular, [Baxt et al., 2002] hypothesizes that neural networks could outperform linear models because of their capacity to capture relationships between input variables that are not seen by simpler models. More recently, additional work has been done using neural networks to anticipate patient outcome (mortality, readmission, extended stay, *etc.*) from their electronic health records [Rajkomar et al., 2018, Avati et al., 2018], including using recurrent neural networks [Esteban et al., 2016].

## 3 Technical background

Neural networks are parametric functions approximators built by composition. The network is built by alternatively composing matrix multiplications and a non linear (element-wise) function, called activation function (such as the Rectified Linear Units (ReLU), $x \mapsto \max(x, 0)$). This type of architecture is that of a multi-layer perceptron. Finding the coefficients of the matrices building the network is an optimization problem, also called "*learning*" or "*fitting*" the model. The loss function of this problem depends on the input data. In supervised learning, neural networks are optimized on a training set to minimize a loss function between their prediction and an observed target. The nature of the loss function depends on the task. Usually, mean square error is used for regression and cross-entropy for classification. To minimize the training error, neural networks are designed differentiable, and optimized

using Stochastic Gradient Descent (SGD), a gradient descent approach where the loss is approximated over a subset of the training examples (called a "*mini-batch*"). Optimizing on a training set eventually leads to degrading performances on unseen examples (this is known as *over-fitting*). To curtail this, the performance of the neural network is monitored on a dataset of unseen examples, known as the *validation* set, and optimization is stopped once the validation metrics worsen. This is done in combination with regularization techniques: any method that empirically reduces the test error, at the expense of the training error.

Recurrent neural networks, reuse their internal parameters sequentially to be able to process and learn over time series data of arbitrary length. At every step, they combine information coming from the current time step with past information condensed by the neural network into a hidden state vector. Popular models include the long short term memory (LSTM) [Hochreiter and Schmidhuber, 1997] and Gated Recurrent Units (GRUs) [Cho et al., 2014]. When time series are long, recurrent neural networks can be trained with truncated back propagation through time [Williams and Peng, 1990]. Namely, the gradients are approximated, and optimization steps are taken, over consecutive subsets of data along the time dimension. Data global to a time series can be combined with the neural network, for instance through the initial hidden vector. The reader is referred to [Goodfellow et al., 2016] for an extensive textbook on deep learning.

## 4 Adverse Events Prediction

Because predicting adverse events, either as a regression (for the time to the next adverse event), or as a very in-balanced classification (classifying if an adverse event will occur in the next $k$ days), is hard, we chose to model the problem as a survival task. Namely, we predict a *latent* patient risk of experiencing an adverse event. It is important to understand that this risk is interpreted only relative to other patient risks, that is a patient risk is higher than another if the former is more likely to experience an adverse event than the latter. In other words, this risk is only introduced to output a ranking of patients. Given a predicted ranking, and the true ranking (computed from the times to the next adverse event), we can compute a score metric called *concordance index* (C-index) [Harrell et al., 1982]. This measure is not differentiable and therefore cannot be optimized through gradient based methods (as it is done for neural networks). Hence, we use a surrogate loss derived from the maximum likelihood of the Cox proportional hazard model for survival analysis [Cox, 1972]. This likelihood loss function is used to compute gradients for the neural network, but model selection and final scores are expressed in terms of C-index. The detail of these loss function can be burdensome and we deliberately omit it here. In short, the C-index is a measure counting the percentage of pairs (of patients here) properly ranked. The Cox model makes more assumptions on the mathematical form of the risk function in order to derive a likelihood. Both are able to deal with censored data, *i.e.* patients exiting the program (or the program end-

ing). The interested reader is referred to the aforementioned literature, as well as the adaptation for neural networks introduced in [Katzman et al., 2018]. Unlike in Cox proportional hazard model, the problem contains a strong time component as we wish to predict a risk for every patient *on every day* (with new information coming in). The metrics are therefore evaluated on a daily basis across all patients. Modeling the problem as a ranking problem is in line with the application pursued here. Indeed, on every day, it is sufficient for the care giver to be able to rank the patients by risk, in order to provide an intuition on where to prioritize, as care workers cannot visit all patients on a daily basis.

As depicted in Figure 1, we use a recurrent neural network to process patient data, both static (patient information and diagnoses) and time distributed (vital signs measured on a daily basis). The network outputs a risk for every patient, on every day. We use the Cox log-likelihood as a loss function to train the network, as was previously done by [Katzman et al., 2018], and report the C-index as well.

Static patient data contain medical diagnoses codes from the International Statistical Classification of Diseases and Related Health Problems (ICD9, ICD10) [Organization, 2004], which are not very informative on their own. To tackle this issue [Choi et al., 2016] uses an unsupervised deep learning approach to embed these codes into a more meaningful vector space, using additional information from the disease, as well as other types of codes. The embedded codes show desirable properties such as diseases with similar symptoms or prescriptions are close together in Euclidian distance. We used their pre-trained embeddings to represent this part of our data. Because patients have a variable number of diseases, we need to use another (small) recurrent neural network (hidden inside the orange parallelogram in Figure 1) to process these diseases before passing them on to the main (larger) recurrent neural network. Alternatively, we also try not to include that information, and simply pass a null vector (as usually done), with the intuition that learning should be easier.

We face more challenges as we have some missing data in the vital signs observed on a daily basis. Although more complex solutions exist to model this (*e.g.*, [Che et al., 2018]), we chose the simple approach of modelling missing data with additional binary variables representing if the data is missing.

The data is split into training (60%), validation (20%) and test (20%) sets. We made sure that no information from the future could be used to make prediction and hence computed the splits as dates: from the beginning to the first split date would constitute the training set *etc.* Past events (previous targets) however can be used as input after their occurring date. Some patients appear in multiple sets, while other are new in every one. This is because of the nature of the application as we want to keep assessing the risk of patients throughout their participation in the program and not just once.

Neural networks and their optimization algorithms come with a a number of so called "*hyper-parameters*": parameters that cannot be optimized directly. In our case, these hyper-parameters divide into two groups. The first group controls the optimization algorithm itself: SGD, or the Adam variant [Kingma and Ba, 2014], the gradient step size, the $L_2$ regularization multiplier, the number of examples in

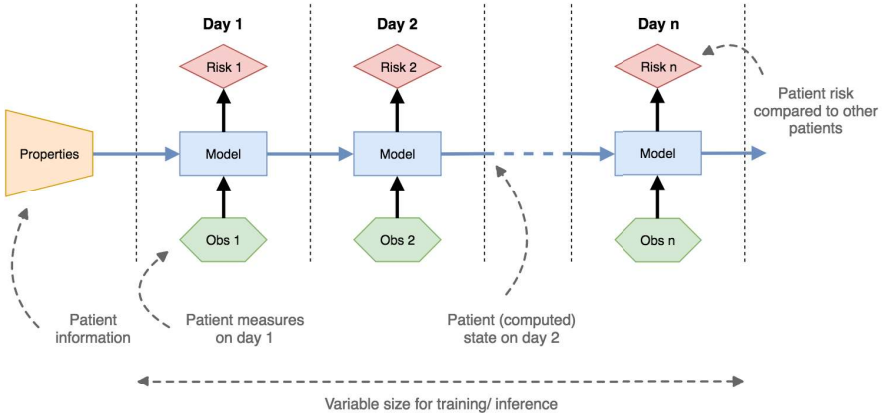



**Fig. 1** A recurrent neural network is used to combine the patients *a priori* information with their daily vital signs and output a risks.

a SGD mini-batch, the number of time steps considered in the truncated version of back-propagation through time, and the use of dropout (a regularization technique) [Kingma and Ba, 2014]. The second group controls architectural decisions: number and size of hidden layers (the matrices involved in the neural network), the type of recurrent layers (LSTM or GRU), and whether to use the patient static data as the initial hidden vector or to simply omit it. A pragmatic way to select these hyper-parameters is to generate randomly some configurations, train the networks for each of them and finally keep the one performing best (in C-index) on the validation set.

Expanding on the evaluation of machine learning performance, we compared it to manually engineered alerts (four levels of severity). We also compare to a simple linear survival baseline using the time distributed readings independently (time dependencies are not taken into account) without using static patient data.

## 5 Results

The dataset we use has been gathered by an Ontario private HHC agency during the course of a multi-year pilot HT program and is fully anonymized. The input data include the patient static information (sex, age, and medical records through ICD codes), and daily vital sign readings (blood glucose, systolic, diastolic, heart beat, $SpO_2$ oximeter, and weight). The dataset also contains observed adverse events experienced by the patients and used to compute the losses (either Cox log-likelihood or C-index). On any given day, past events are also added as input.[1] The 320 patients in our study were aged from 31 to 101 with an average age of 79. Women represented 56.25% of the patient group. Moreover, 36.25% of patients experienced

[1] This is correct as no information from the future is added to the input.

at least one hospital readmission or emergency room visit while on the HT program. The average number of events per patient was 0.72 with a standard deviation of 1.30. The average number of events for the 36.25% of patients that experienced at least one was 1.98 with a standard deviation of 1.47. Finally, 91.56% of patients had co-morbidities, hypertension being the more frequent at 55.31% followed by chronic heath failure with 46.25%, diabetes with 39.69% and chronic obstructive pulmonary disease at 38.13%. In total, we have access to 76,359 daily patient observations.

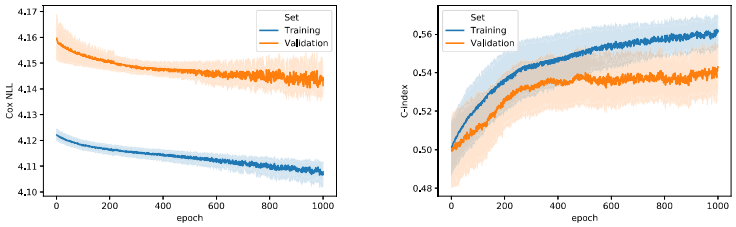
The training process of neural networks was highly stochastic, with results often close to random, as there is a strong noise to signal ratio in the data. This made it difficult to differentiate promising models from random luck. Therefore, hyper-parameters configurations were manually selected from across all runs (over a hundred), based not only on validation performance, but as well on stability (reduced stochasticity during training) in addition to small gap between the training and validation scores.

These configurations were then retrained over twenty random seeds to average the results. The training for the best performing set of hyper-parameters is depicted in Figure 2. Important details of this configuration are: three gated recurrent unit (GRU) layers with eight units each and dropout for the network architecture, an SGD optimizer with a batch size of 64, and a truncation length for back-propagation through time of 15 days for the training procedure. It it worth noting that the neural network presented in Figure 2 does *not* use static information about the patients (this was a configurable hyper-parameter choice). That is among all the configurations trained, the best performing model was one that did not make use of the static information. This is, further discussed in Section 6.

We removed from these twenty models a few that did not perform well on the training or validation set (three of them). Due to the stochasticity in the training process, some trained networks can under-perform. We can legitimately filter them out, as long as we do so on validation or training sets.[2] Then, we computed their final score on the test set. The results are given in Figure 3. The box plot reports a test concordance index of $58.8 \pm 4.6\%$. We performed a Student T-test against the value of 50% and rejected with p-value $3.7 \times 10^{-7}$ that our model is equivalent to a random one.

We compares against two baselines. The first ones are the manual alerts set by the care workers. We have a history of four levels of urgency (none, low, medium, high) for every patient and every day. As done with the latent risk modeled by the neural network, these alerts yield a natural ranking that can be used to compute a concordance index. These alerts achieve concordance indices of 48.7%, 50.7%, and 51.1% on respectively the training, the validation, and the test set. Note that these alerts were not set based on training data, so these results could be aggregated. However, we provide them separated so that the test error could be compared to the neural network on the same data points. The second baseline is a linear survival regression model, where data points are the vital signs for every patient and every day, as if they were independent (no time dependencies are taken into account). This

[2] The only difference is that the training procedure now includes a filtering phase.




(a) Cox negative log-likelihood. (b) Concordance index.

**Fig. 2** Training of the most promising set hyper-parameters, averaged over twenty runs.

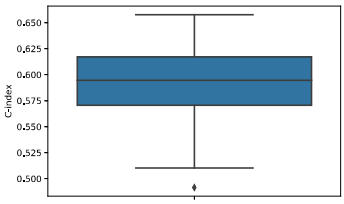



**Fig. 3** A box plot of the test error of the most promising set hyper-parameters, over seventeen runs (three dropped due to under-performance on training or validation set).

model achieves a training error (on training and validation sets combined) of 49.7% and a test error of 48.0%. Even if these two baselines are simplistic, the fact that they do not achieve better than a random draw shows the difficulty of the problem.

We implemented the neural network in PyTorch [Paszke et al., 2017], used Lifelines [Davidson-Pilon et al., 2019] to compute the linear baseline and the C-index, made use of Numpy [Van Der Walt et al., 2011], Scipy [Jones et al., 01 ], Pandas [McKinney, 2010], Scikit-Learn [Pedregosa et al., 2011], IPython [Perez and Granger, 2007], and Jupyter [Kluyver et al., 2016] for pre-processing of the data and post-processing of the results, and rendered the figures with Matplotlib [Hunter, 2007] and Seaborn [Waskom et al., 2018].

## 6 Discussion

The high stochasticity of the problem makes training hard and long, therefore making comparison between different neural network architectures and training procedures either expensive (through averaging) or unfair (some configuration randomly performing abnormally well or poorly). As stated in the previous section, passing the patient static information as the first hidden vector of the recurrent neural network (as opposed to just passing a null vector), as proposed in Section 4, did not

improve the performances and was therefore not selected through hyper-parameter search. Further inquiring should be done to find out if a better model could be obtained using the static patient information. Our hypothesis is that this data does have predictive power for this task, but that the specific part of the neural network resposible for it failed to learn, due to optimization hardships. Indeed, not only this part of the model adds more parameters (layers) to train, these parameters are also updated less frequently in the truncated variant of back-propagation through time. Potential directions could be to focus more on the training of this part of the model (for instance through pertaining), or to include this data at every time-step (explicitly or through an attention mechanism).

Our results do not show that a linear model could not perform as well as the neural network, as less effort was given to this model. Improving the linear baseline would however mean additional engineering of the data to include time dependencies, patient static information, and perform feature selection.

These results suggest that combining, even weak, signals from remote monitoring in the homecare context can outperform simple manual baselines which open the door to better models.

Nonetheless, a self-fulfilling prophecy problem could occur with a better prediction accuracy. Care workers using machine learning generated alerts would prevent events from happening and reduce the observations labeled as events. A potential alternative is to ask care workers if the prediction was useful or not, *i.e.*, if they want such prediction happening again in the future. While far from perfect, this methodology has the advantage of enabling model retraining as the data is gathered. More research is required to evaluate the risk of this problem and performance of the proposed alternative.

In addition, further research is needed to better understand the factors that contribute to higher risk days for home telemonitoring patients. Indeed, the black-box nature of neural networks makes them difficult to implement in the health care industry since physicians and other care workers generally want to understand why an adverse event probability is predicted. For example, what action should a care worker take if manual alerts are triggered but neural networks say that nothing is happening? The model performance suggests that no action should be taken, but this is clearly a difficult call.

## References

[Avati et al., 2018] Avati, A., Jung, K., Harman, S., Downing, L., Ng, A., and Shah, N. H. (2018). Improving palliative care with deep learning. *BMC Medical Informatics and Decision Making*, 18(4):122.

[Baxt et al., 2002] Baxt, W. G., Shofer, F. S., Sites, F. D., and Hollander, J. E. (2002). A neural network aid for the early diagnosis of cardiac ischemia in patients presenting to the emergency department with chest pain. *Annals of Emergency Medicine*, 40(6):575–583.

[Berwick et al., 2008] Berwick, D. M., Nolan, T. W., and Whittington, J. (2008). The triple aim: care, health, and cost. *Health Affairs*, 27(3):759–769.

[Che et al., 2018] Che, Z., Purushotham, S., Cho, K., Sontag, D., and Liu, Y. (2018). Recurrent Neural Networks for Multivariate Time Series with Missing Values. *Scientific Reports*, 8(1).

[Cho et al., 2014] Cho, K., van Merrienboer, B., Gulcehre, C., Bougares, F., Schwenk, H., and Bengio, Y. (2014). Learning phrase representations using rnn encoder-decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing (EMNLP 2014)*.

[Choi et al., 2016] Choi, Y., Chiu, C. Y.-I., and Sontag, D. (2016). Learning Low-Dimensional Representations of Medical Concepts. *AMIA Summits on Translational Science Proceedings*, 2016:41–50.

[Cox, 1972] Cox, D. R. (1972). Regression models and life-tables. *Journal of the Royal Statistical Society. Series B*, 34:187–220.

[Davidson-Pilon et al., 2019] Davidson-Pilon, C., Kalderstam, J., Zivich, P., Kuhn, B., Fiore-Gartland, A., Moneda, L., Gabriel, WIlson, D., Parij, A., Stark, K., Anton, S., Besson, L., Jona, Gadgil, H., Golland, D., Hussey, S., Noorbakhsh, J., Klintberg, A., Jordan, J., Rose, J., Slavitt, I., Martin, E., Ochoa, E., Albrecht, D., dhuynh, Zgonjanin, D., Chen, D., Fournier, C., Arturo, and Rendeiro, A. F. (2019). Camdavidsonpilon/lifelines: v0.19.2.

[Esteban et al., 2016] Esteban, C., Staeck, O., Baier, S., Yang, Y., and Tresp, V. (2016). Predicting clinical events by combining static and dynamic information using recurrent neural networks. In *2016 IEEE International Conference on Healthcare Informatics (ICHI)*, pages 93–101.

[Gavrilov and Heuveline, 2003] Gavrilov, L. and Heuveline, P. (2003). Aging of population. Technical report.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep learning*. MIT press.

[Hansen et al., 2011] Hansen, L. O., Young, R. S., Hinami, K., Leung, A., and Williams, M. V. (2011). Interventions to reduce 30-day rehospitalization: a systematic review. *Annals of internal medicine*, 155(8):520–528.

[Harrell et al., 1982] Harrell, Frank E., J., Califf, R. M., Pryor, D. B., Lee, K. L., and Rosati, R. A. (1982). Evaluating the Yield of Medical Tests. *JAMA*, 247(18):2543–2546.

[Harrison and Kennedy, 2005] Harrison, R. F. and Kennedy, R. L. (2005). Artificial neural network models for prediction of acute coronary syndromes using clinical data from the time of presentation. *Annals of emergency medicine*, 46(5):431–439.

[Hochreiter and Schmidhuber, 1997] Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

[Hunter, 2007] Hunter, J. D. (2007). Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95.

[Huntington et al., 2011] Huntington, W. V., Covington, L. A., Center, P. P., and Manchikanti, L. (2011). Patient protection and affordable care act of 2010: reforming the health care reform for the new decade. *Pain Physician*, 14(1):E35–E67.

[Jencks et al., 2009] Jencks, S. F., Williams, M. V., and Coleman, E. A. (2009). Rehospitalizations among patients in the medicare fee-for-service program. *New England Journal of Medicine*, 360(14):1418–1428.

[Jones et al., 01 ] Jones, E., Oliphant, T., Peterson, P., et al. (2001–). SciPy: Open source scientific tools for Python. [Online; accessed ¡today¿].

[Kansagara et al., 2011] Kansagara, D., Englander, H., Salanitro, A., Kagen, D., Theobald, C., Freeman, M., and Kripalani, S. (2011). Risk prediction models for hospital readmission: a systematic review. *Jama*, 306(15):1688–1698.

[Katzman et al., 2018] Katzman, J. L., Shaham, U., Cloninger, A., Bates, J., Jiang, T., and Kluger, Y. (2018). Deepsurv: personalized treatment recommender system using a cox proportional hazards deep neural network. *BMC Medical Research Methodology*, 18(1):24.

[Kingma and Ba, 2014] Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Kluyver et al., 2016] Kluyver, T., Ragan-Kelley, B., Pérez, F., Granger, B., Bussonnier, M., Frederic, J., Kelley, K., Hamrick, J., Grout, J., Corlay, S., Ivanov, P., Avila, D., Abdalla, S., and Willing, C. (2016). Jupyter notebooks – a publishing format for reproducible computational

workflows. In Loizides, F. and Schmidt, B., editors, *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, pages 87 – 90. IOS Press.

[Luck et al., 2017] Luck, M., Sylvain, T., Cardinal, H., Lodi, A., and Bengio, Y. (2017). Deep learning for patient-specific kidney graft survival analysis. *arXiv*, (1705.1024).

[McKinney, 2010] McKinney, W. (2010). Data structures for statistical computing in python. In van der Walt, S. and Millman, J., editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56.

[Organization, 2004] Organization, W. H. (2004). *International statistical classification of diseases and related health problems*, volume 1. World Health Organization.

[Paré et al., 2007] Paré, G., Jaana, M., and Sicotte, C. (2007). Systematic review of home telemonitoring for chronic diseases: the evidence base. *J Am Med Inform Assoc*, 14(3):269–77.

[Paszke et al., 2017] Paszke, A., Gross, S., Chintala, S., Chanan, G., Yang, E., DeVito, Z., Lin, Z., Desmaison, A., Antiga, L., and Lerer, A. (2017). Automatic differentiation in pytorch.

[Pedregosa et al., 2011] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research*, 12(Oct):2825–2830.

[Perez and Granger, 2007] Perez, F. and Granger, B. E. (2007). Ipython: A system for interactive scientific computing. *Computing in Science Engineering*, 9(3):21–29.

[Price et al., 2000] Price, R. K., Spitznagel, E. L., Downey, T. J., Meyer, D. J., Risk, N. K., and El-Ghazzawy, O. G. (2000). Applying artificial neural network models to clinical decision making. *Psychological assessment*, 12(1):40.

[Rajkomar et al., 2018] Rajkomar, A., Oren, E., Chen, K., Dai, A. M., Hajaj, N., Hardt, M., Liu, P. J., Liu, X., Marcus, J., Sun, M., Sundberg, P., Yee, H., Zhang, K., Zhang, Y., Flores, G., Duggan, G. E., Irvine, J., Le, Q., Litsch, K., Mossin, A., Tansuwan, J., Wang, D., Wexler, J., Wilson, J., Ludwig, D., Volchenboum, S. L., Chou, K., Pearson, M., Madabushi, S., Shah, N. H., Butte, A. J., Howell, M. D., Cui, C., Corrado, G. S., and Dean, J. (2018). Scalable and accurate deep learning with electronic health records. *npj Digital Medicine*, 1(1):18.

[Rasmussen et al., 2012] Rasmussen, M. S., Justesen, T., Dohn, A., and Larsen, J. (2012). The home care crew scheduling problem: Preference-based visit clustering and temporal dependencies. *European Journal of Operational Research*, 219(3):598–610.

[Ross et al., 2008] Ross, J. S., Mulvey, G. K., Stauffer, B., Patlolla, V., Bernheim, S. M., Keenan, P. S., and Krumholz, H. M. (2008). Statistical models and patient predictors of readmission for heart failure: a systematic review. *Archives of internal medicine*, 168(13):1371–1386.

[Suh et al., 2010] Suh, M.-k. . K., Evangelista, L. S., Chen, C.-A. . A., Han, K., Kang, J., Tu, M. K., Chen, V., Nahapetian, A., and Sarrafzadeh, M. (2010). An automated vital sign monitoring system for congestive heart failure patients. In *Proceedings of the 1st ACM International Health Informatics Symposium*, pages 108–117. ACM.

[Swiercz et al., 1998] Swiercz, M., Mariak, Z., Lewko, J., Chojnacki, K., Kozlowski, A., and Piekarski, P. (1998). Neural network technique for detecting emergency states in neurosurgical patients. *Medical and Biological Engineering and Computing*, 36(6):717–722.

[Van Der Walt et al., 2011] Van Der Walt, S., Colbert, S. C., and Varoquaux, G. (2011). The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering*, 13(2):22.

[Wallace et al., 2014] Wallace, E., Stuart, E., Vaughan, N., Bennett, K., Fahey, T., and Smith, S. M. (2014). Risk prediction models to predict emergency hospital admission in community-dwelling adults: a systematic review. *Medical care*, 52(8):751.

[Ward et al., 2014] Ward, B. W., Schiller, J. S., and Goodman, R. A. (2014). Peer reviewed: Multiple chronic conditions among us adults: A 2012 update. *Preventing chronic disease*, 11.

[Waskom et al., 2018] Waskom, M., Botvinnik, O., O'Kane, D., Hobson, P., Ostblom, J., Lukauskas, S., Gemperline, D. C., Augspurger, T., Halchenko, Y., Cole, J. B., Warmenhoven, J., de Ruiter, J., Pye, C., Hoyer, S., Vanderplas, J., Villalba, S., Kunter, G., Quintero, E., Bachant, P., Martin, M., Meyer, K., Miles, A., Ram, Y., Brunner, T., Yarkoni, T., Williams, M. L., Evans, C., Fitzgerald, C., Brian, and Qalieh, A. (2018). mwaskom/seaborn: v0.9.0 (july 2018).

[Williams and Peng, 1990] Williams, R. J. and Peng, J. (1990). An efficient gradient-based algorithm for on-line training of recurrent network trajectories. *Neural Computation*, 2(4):490–501.

[Zhu et al., 2014] Zhu, M., Cheng, L., Armstrong, J. J., Poss, J. W., Hirdes, J. P., and Stolee, P. (2014). *Machine Learning in Healthcare Informatics*. Springer.