Predicting Clinical Events by Combining Static and Dynamic Information Using Recurrent Neural Networks

Cristóbal Esteban[1], **Oliver Staeck**[2], **Yinchong Yang**[1] **and Volker Tresp**[1]
[1] Siemens AG and Ludwig Maximilian University of Munich, Munich, Germany
[2] Charité University Hospital of Berlin, Berlin, Germany
{cristobal.esteban, yinchong.yang.ext, volker.tresp}@siemens.com
oliver.staeck@charite.de

Abstract

In clinical data sets we often find static information (e.g. gender of the patients, blood type, etc.) combined with sequences of data that are recorded during multiple hospital visits (e.g. medications prescribed, tests performed, etc.). Recurrent Neural Networks (RNNs) have proven to be very successful for modelling sequences of data in many areas of Machine Learning. In this work we present an approach based on RNNs that is specifically designed for the clinical domain and that combines static and dynamic information in order to predict future events. We work with a database collected in the Charité Hospital in Berlin that contains all the information concerning patients that underwent a kidney transplantation. After the transplantation three main endpoints can occur: rejection of the kidney, loss of the kidney and death of the patient. Our goal is to predict, given the Electronic Health Record of each patient, whether any of those endpoints will occur within the next six or twelve months after each visit to the clinic. We compared different types of RNNs that we developed for this work, a model based on a Feedforward Neural Network and a Logistic Regression model. We found that the RNN that we developed based on Gated Recurrent Units provides the best performance for this task. We also performed an additional experiment using these models to predict next actions and found that for such use case the model based on a Feedforward Neural Network outperformed the other models. Our hypothesis is that long-term dependencies are not as relevant in this task.

1 Introduction

As a result of the recent trend towards digitization, an increasing amount of information is recorded in clinics and hospitals, and this increasingly overwhelms the human decision maker. This issue is one of the main reasons why Machine Learning (ML) is gaining attention in the medical domain, since ML algorithms can make use of all the available information to predict the most likely future events that will occur to each individual patient. Physicians can include these predictions in their decision processes which can lead to improved outcomes.

Eventually ML can also be the basis for a decision support system that provides personalized recommendations for each individual patient (e.g., which is the most suitable medication, which procedure should be applied next, etc.).

It is also worth noticing that medical data sets are becoming both longer (i.e. we have more samples collected through time) and wider (i.e. we store more variables). Therefore we need to use ML algorithms capable of modelling complex relationships among a big number of time-evolving variables.

A kind of models that can capture very complex relationships are Neural Networks, which have proven to be successful in other areas of ML [1]. Particularly there is a notable parallelism among the prediction of clinical events and the field of Language Modelling, where Deep Learning has also proven to be very successful. One could imagine that each word of a text represents an event: A text would correspond to a a stream of events and the task of Language Modelling would be to predict the next event in the stream. For this reason, we can get inspired by Language Modelling to create models that predict clinical events. However, the medical domain has a set of characteristics that make it an almost unique scenario: multiple events can occur at the same time, there are multiple sequences (i.e. multiple patients), each sequence has an associated set of static variables and both inputs and outputs can be a combination of boolean variables and real numbers. For these reasons we need to develop approaches specifically designed for the medical use case.

For our work we use a large data set collected from patients that suffered from kidney failure. The data was collected in the Charité hospital in Berlin and it is the largest data collection of its kind in Europe. Once the kidney has failed, patients face a lifelong treatment and periodic visits to the clinic for the rest of their lives. Until the hospital finds a new kidney for the patient, he or she must attend to the clinic multiple times per week in order to receive dialysis, which is a treatment that replaces many of the functions of the kidney. After the transplant has been performed, the patient receives immunosuppressive therapy to avoid the rejection of the transplanted kidney. The patient must be periodically controlled to check the status of the kidney, adjust the treatment and take care of associated diseases, such as those that arise due to the immunosuppressive therapy. This data set started being

recorded more than 30 years ago and it is composed of more than 4000 patients that underwent a renal transplantation or are waiting for it. The database has been the basis for many studies in the past [2, 3, 4, 5].

There are a set of endpoints that can happen after a transplantation and it would be very valuable for the physicians to be able to know beforehand when one of these is going to happen. Specifically we will predict whether the patient will die, the transplant will be rejected, or the transplant will be lost. For each visit that a patient makes to the clinic, we will anticipate which of those three events (if any) will occur both within 6 months and 12 months after the visit.

In order to predict endpoints we developed a new model based on Recurrent Neural Networks (RNNs).

The paper is organized as follows. Next section contains related work. In section 3 we make an introduction to the kidney transplantation problem and how our work could contribute to alleviate it. In section 4 we present an overview of the existing types of RNNs and introduce our approach for combining static and dynamic data. Section 5 contains experiments and results. Finally, section 6 contains our conclusions.

2 Related Work

Regarding the task of predicting clinical events, Esteban et al. [6] introduced the Temporal Latent Embeddings model which is based on a Feedforward Neural Network. This model outperforms its baselines for the task of predicting which events will be observed next given the previous events recorded (i.e. the goal was to predict which laboratory analyses and medication prescriptions will be observed in the next visit for each patient). The architecture of this model can be seen in Figure 1.

The Temporal Latent Embeddings model requires to explicitly define the number of time steps in the past that we want to consider in order to predict the next step. In some scenarios, this constrain can actually be an advantage since many recent papers have shown how attention mechanisms can actually improve the performance of Neural Networks [7] [8]. The Temporal Latent Embeddings model puts all the attention on the last "n" samples and therefore it provides an advantage over RNNs for data sets where the events that we want to predict are dependent just on the "n" previous events. However, in order to capture long term dependencies on the data with this model, we have to aggregate the whole history of each patient in one vector (e.g. computing the mean values of each laboratory measurement), and therefore many long-term dependencies can be lost in this aggregation step (e.g. a very high value in one measurement followed by a very low value in other measurement).

In recent work, Choi et al. [9] use an RNN with Gated Recurrent Units (GRUs) combined with skip-gram vectors [10] in order to predict diagnosis, medication prescription and procedures. However in this work the static information of the patients was not integrated into the Neural Network.

Outside of the medical domain, regarding the tasks of using sequences of data together with RNNs in order to predict events, we can find in the literature multiple successful examples, specially in the field of Natural Language Processing [11] [12]. However to our knowledge none of them includes both sequential and static information, which is the typical setting in sequences of clinical data, and in general these approaches are not designed for the specific characteristics of the medical data sets.

3 Kidney Transplantation Endpoints

Chronic kidney disease is a worldwide health burden with increasing prevalence and incidence. It affects multiple organ systems and may result in end stage renal disease. The growing number of patients with end stage renal disease requiring dialysis or transplantation is a major challenge for healthcare systems around the world [13]. For most patients kidney transplantation is the treatment of choice offering lowest morbidity, significant survival benefit, lowest costs and highest quality of life. The main goals after transplantation are the reduction of complications and the increase of long-term graft and patient survival. However, kidney transplant recipients are at high risk of severe complications like rejections, infections, drug toxicity, malignancies and cardiovascular diseases. The majority of patients have to take 5-10 different medications every day during their entire life. Due to complexities of post-transplant management kidney transplant recipients should remain in life-long specialized care. The medical records of kidney transplant recipients mostly cover a very long history of disease (years to decades) and include a vast number of diagnoses, symptoms, results, medications and laboratory values.

Due to this complexity of medical data, decision making is complex and difficult in clinical practice of kidney transplantation. Treating 20-40 patients per day for the medical practitioner it is generally not possible to review all available medical information during every bed side or outpatient consultation. However, early prediction of clinical events on the base of current data enables informed decision making on how best choose future therapy, identify current problems and thus help to avoid complications and improve survival of the patient and graft, reduce morbidity, improve health related quality of life. Assessing the risk of graft failure or death in renal transplant recipients can be crucial for the individual patient management detecting patients that need intensified medical care. Integrating a computerized tool to predict relevant end points (deteriorating of the graft function, infections, rejections, graft loss, death) on the base of all available medical data may be of great benefit in the clinical routine and provide medical professional with significant decision support.

Detecting a higher risk of graft failure or death 6-12 months in advance may timely allow identifying of toxicities, side effects, interactions of medications, infections, relevant comorbidities and other complications in order to intervene and avoid a poor outcome. Furthermore, predicting rejections during consultations enables the medical practitioner to change immunosuppressive medications and thus prevent deterioration of the graft function. Therefore, computerized prediction of relevant events has the potential to significantly change daily medical practice in patient care.

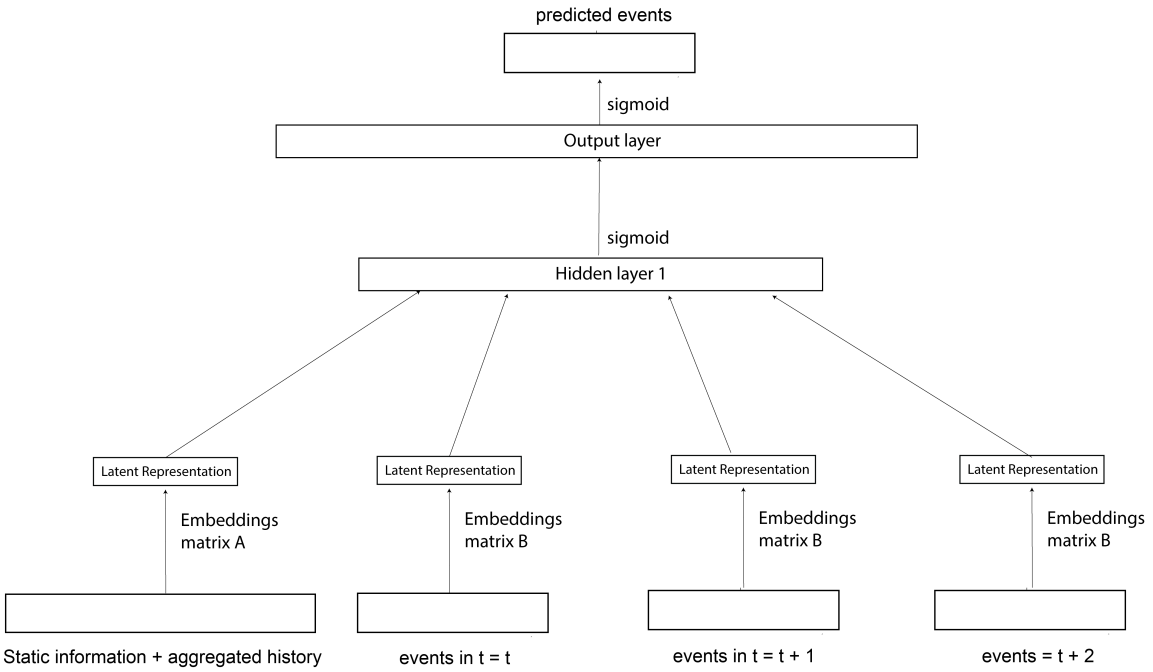

Figure 1: Temporal Latent Embeddings Model.

There are three major endpoints that can happen after a kidney transplantation: rejection of the kidney, graft loss and death of the patient. A rejection means that the body is rejecting the kidney. In such situation, physicians try to fight the rejection with medications and if they are not able to stop the rejection, then the kidney will stop working.

Our goal with this work is to predict which of these endpoints (if any) will occur to each patient 6 months and 12 months after each visit to the clinic, given the medical history of the patient. Specifically, in order to make these predictions we will exploit from the medical history of each patient the sequence of medications prescribed, the sequence of laboratory tests performed together with their results, and the following static information: age, gender, blood type, time from first dialysis, time from the first time the patient was seen, weight and primary disease. Therefore each patient can be seen as a sequence of events (medications prescribed and laboratory tests performed) combined with static data. We will use both static and dynamic data in order to predict the explained endpoints.

4 Recurrent Neural Networks for Clinical Event Prediction

RNNs are a type of Neural Network where the hidden state of one time step is computed by combining the current input with the hidden state of the previous step. This way RNNs can learn to remember events from the past that are relevant to predict future outcomes. In Figure 2 you can see the architecture of an RNN.

As opposed to the models based on Feedforward Neural Networks, when sequential data is modelled using RNNs, we

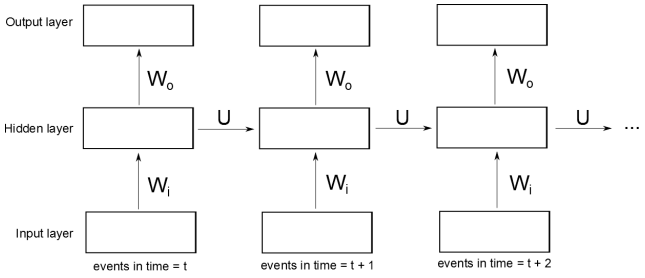

Figure 2: Recurrent Neural Network.

do not have to specify how many time steps from the past we want to consider to predict future events. Therefore, RNNs can theoretically learn to remember any event (or combination of events) that occur in the life of the patient that is useful to predict future events. This feature can be very valuable when our clinical data set presents such kind of long term dependencies.

Another advantage of using RNNs for learning with sequences is that, given a new patient, we can start predicting future events for such patient right after his or her first visit to the clinic. On the other hand, if we model our data with a Feedforward Neural Network and we have decided to use the "n" previous visits to predict future events, we will have to wait until we have accumulated at least "n" visits for a patient to start predicting his or her future events. In some scenarios, this ability of making predictions using a variable amount of previous time steps can be a very useful feature.

More formally, the output of an RNN is computed as:

$$\hat{y}_t = \sigma(W_o h_t) \tag{1}$$

where σ is some differentiable function such as the logistic sigmoid function and h is the hidden state of the Neural Network.

Given a sequence of vectors $x = (x_1, x_2, \cdots, x_T)$, the hidden state of an RNN is computed the following way:

$$h_t = f(h_{t-1}, x_t). \tag{2}$$

We will summarize the most common options for the f function in the next sections.

4.1 Standard Recurrent Neural Network

This is the classic way of updating the hidden state in RNNs:

$$h_t = \sigma(Wx_t + Uh_{t-1}). \tag{3}$$

As we mentioned earlier, one of our main interests for using RNNs is their ability of capturing long-term dependencies. However, it was observed by Bengio et al. [14] that it is difficult for standard RNNs to capture such long-term dependencies due to the gradient vanishing problem.

4.2 Long Short-Term Memory Units

In order to alleviate the gradient vanishing problem, Hochreiter et al. [15] developed a gating mechanism that dictates when and how the hidden state of an RNN has to be updated.

There are different versions with minor modifications regarding the gating mechanism in the Long short-term memory (LSTM) units. We will use in here the one defined by Graves et al.[16]:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + W_{ci}c_{t-1} + b_i) \tag{4}$$
$$r_t = \sigma(W_{xr}x_t + W_{hr}h_{t-1} + W_{cr}c_{t-1} + b_r) \tag{5}$$
$$c_t = r_t \odot c_{t-1} + i_t \odot \tanh(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{6}$$
$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + W_{co}c_{t-1} + b_o) \tag{7}$$
$$h_t = o_t \odot \tanh(c_t) \tag{8}$$

where $\odot$ is the element-wise product and i, r and o are the input, forget and output gates respectively. As it can be seen, the gating mechanism regulates how the current input and the previous hidden state must be combined to generate the new hidden state.

LSTM units have been successfully used many times before [7] [8].

4.3 Gated Recurrent Units

Another gating mechanism named Gated Recurrent Units (GRUs) was introduced by Cho et al. [17] with the goal of making each recurrent unit to adaptively capture dependencies of different time scales. We follow the equations as defined by Chung et al. [18]:

$$r_t = \sigma(W_r x_t + U_r h_{t-1}) \tag{9}$$
$$\tilde{h}_t = \tanh(Wx_t + U(r_t \odot h_{t-1})) \tag{10}$$
$$z_t = \sigma(W_z x_t + U_z h_{t-1}) \tag{11}$$
$$h_t = (1 - z_t)h_{t-1} + z_t\tilde{h}_t \tag{12}$$

where z and r are the update and reset gates respectively.

As it can be seen, GRUs present an architecture less complex than LSTM units, and the former usually perform at least as good than the latter.

4.4 Combining RNNs with Static Data

It often happens that Electronic Health Records (EHRs) contain both dynamic information (i.e. new data is recorded every time a patient visits the clinic or hospital) and static or slowly changing information (e.g. gender, blood type, age, etc.).

Therefore, we modified the RNN architecture to include static information of the patients, such as gender, blood type, cause of the kidney failure, etc. The modified architecture is depicted in Figure 3.

As it can be seen, we process the static information on an independent Feedforward Neural Network whereas we process the dynamic information with an RNN. Afterwards we concatenate the hidden states of both networks and we put an output layer on top of them.

We feed our network with the latent representation of the inputs, as it is often done in Natural Language Processing [19] [12]. In this case we compute such latent representation with a linear transformation.

More formally, we first compute the latent representation of the input data as:

$$\tilde{x}_i^e = A\tilde{x}_i \tag{13}$$
$$x_{t,i}^e = Bx_{t,i} \tag{14}$$

where $\tilde{x}_i$ is a vector containing the static information for patient i being $\tilde{x}_i^e$ its latent representation, and $x_{t,i}$ is a vector containing the information recorded during the visit made by patient i at time t being $x_{t,i}^e$ its latent representation.

Then we compute the hidden state of the static part of the network:

$$\tilde{h}_i = \tilde{f}(\tilde{x}_i^e) \tag{15}$$

where we are current using the input and hidden layers of a Feedforward Neural Network as $\tilde{f}$.

In order to compute the hidden state of the recurrent part of the network we do:

$$h_{t,i} = f(x_{t,i}^e, h_{t-1,i}) \tag{16}$$

where f can be any of the update functions explained above (standard RNN, LSTM or GRU).

Finally, we use both hidden states in order to predict our target:

$$\hat{y}_{t,i} = g(\tilde{h}_i, h_{t,i}). \tag{17}$$

In this work the function g consists of passing the concatenated hidden states through an output layer:

$$g = \sigma(W_o(\tilde{h}_i, h_{t,i}) + b). \tag{18}$$

We derive a cost function based on the Bernoulli likelihood function, also known as Binary Cross Entropy, which has the form:

$$\text{cost}(A, B, W, U) = \sum_{t,i \in \text{Tr}} -y_{t,i}\log(\hat{y}_{t,i}) - (1 - y_{t,i})\log(1 - \hat{y}_{t,i}) \tag{19}$$

where Tr stands for the training data set, $y_{t,i}$ stands for the true target data and $\hat{y_{t,i}}$ stands for the predicted data.

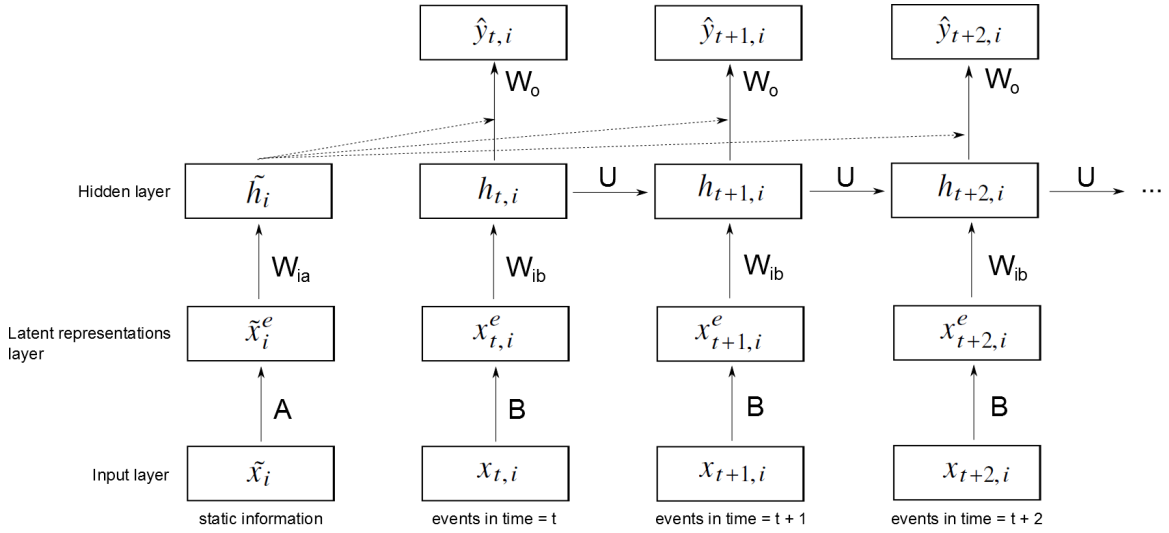

Figure 3: Recurrent Neural Network with static information. *i* stands for the index of the patient.

It is worth mentioning that we tried other structures in order to combine static data together with dynamic data using RNNs. For example we experimented with the approach of using an RNN to compute a latent representation of the whole history of the patient, in order to use such representation together with the latent representations of the last "n" visits as the inputs of a Feedforward Neural Network. We thought it could be an interesting approach since the network would put a higher attention on the most recent visits and would still have access to the full history of the patient. We also tried an attention mechanism similar to the ones shown in [7] [8], which also has the advantage of bringing a high interpretability to the model (i.e. it provides information about why certain event was predicted). However none of these approaches was able to improve the performance of the models presented in this article.

5 Experiments

5.1 Data Pre-processing and Experimental Setup

We have three groups of variables in our data set: endpoints, medications prescribed and laboratory results. Both endpoints and medications prescribed are binary data. On the other hand, the laboratory results are a set of real numbers.

Not every possible laboratory measurement is performed each time a patient visits the clinic (e.g. sometimes a doctor may order to measure calcium if he suspects the patient might have low calcium, otherwise he will not order such measurement). In order to deal with such missing data, we experimented with mean imputation and median imputation. Also, we experimented with both scaling and normalizing the data before inputting it to the Neural Network.

It turned out that encoding the laboratory measurements in a binary way by representing each of them as three event types as in [6], i.e., LabValueHigh, LabValueNormal and LabValueLow, provided a better predictive performance than the approach of doing mean or median imputation and normalizing or scaling the data. If a certain measurement was not done, its corresponding three events will be all equal to 0. We believe that this improvement is due to the increase in the number parameters of the model and due to the fact that with this strategy we remove the need of doing data imputation, which can add significant noise to the data set.

In this work we consider events from year 2005 and onwards due to the improvement of the data quality from that year. Our final data set consists of 173111 visits with 6568 features each.

The model contains several hyperparameters that need to be optimized, being the most relevant ones the rank of the latent representations, the number of hidden units in the Neural Network, the learning rate and the drop out regularization parameter. Besides we will test our models with two optimization algorithms, which are Adagrad [20] and RMSProp [21].

In order to fit these hyperparameters, we randomly split the data into three subsets: 60% of the patients were assigned to the training set, 20% of the patients were assigned to the validation set and another 20% to the test set. Under this configuration, we evaluate the performance of the model by predicting the future events of patients that the model has never seen before, and therefore increasing the difficulty of the task.

When comparing the performance between these models, we report for each model the mean area under the Precision-Recall curve (AUPRC) and mean area under Receiver Operating Characteristics curve (AUROC) together with their associated standard errors after repeating each experiment five times with different random splits of the data. We made sure that these five random splits were identical for each model.

We run the experiments using the models presented in this paper and also using Logistic Regression, since it provided the second best performance in the task of predicting sequences of clinical data in [6].

5.2 Results

Table 1 shows the results of predicting endpoints without considering different types of them (i.e. for evaluation we concatenate all the predictions of the set). "GRU + static", "LSTM + static" and "RNN + static" stand for the architectures presented in this paper that combine an RNN with static information. TLE stands for the Temporal Latent Embeddings model [6]. Random stands for the scores obtained when doing random predictions.

We can see how the recurrent models outperform the other models both in the AUROC score and AUPRC score, being the one using GRUs the best one with an AUPRC of 0.345 and an AUROC of 0.833. The AUPRC scores are pretty low compared to its maximum possible value (i.e. 1), but are fairly good compared to the random baseline of 0.073, which is that low due to the high sparsity of the data.

Table 1: Scores for full visit predictions. AUPRC stands for Area Under Precision-Recall Curve. AUROC stands for Area Under ROC Curve.

	AUPRC	AUROC
GRU + static	0.345 ± 0.013	0.833 ± 0.006
LSTM + static	0.330 ± 0.014	0.826 ± 0.006
RNN + static	0.319 ± 0.012	0.822 ± 0.006
TLE	0.313 ± 0.010	0.821 ± 0.005
Logistic Regression	0.299 ± 0.009	0.808 ± 0.005
Random	0.073 ± 0.002	0.5

Since we repeated the experiment five times with different splits of the data, we have slight variations on the configuration of the winning model. However, the most repeated configuration was composed of 100 hidden units, a rank size of 50 for the latent representation, a dropout rate of 0.1, a learning rate of 0.1 and the Adagrad optimization algorithm.

In Table 2 we show the performance achieved for each specific endpoint. It can be seen how the it gets the best score for predicting the death of a patient whereas it obtains the worse AUPRC in the task of predicting kidney loss within the next 6 months.

Table 2: Scores for full visit predictions. AUPRC stands for Area Under Precision-Recall Curve. AUROC stands for Area Under ROC Curve.

	AUPRC	AUROC
Rejection 6 months	0.234 ± 0.010	0.778 ± 0.006
Rejection 12 months	0.279 ± 0.014	0.768 ± 0.009
Loss 6 months	0.167 ± 0.017	0.821 ± 0.009
Loss 12 months	0.223 ± 0.019	0.814 ± 0.009
Death 6 months	0.467 ± 0.018	0.890 ± 0.004
Death 12 months	0.465 ± 0.020	0.861 ± 0.004

5.3 Additional Experiments

As we mentioned earlier, the Temporal Latent Embeddings model outperformed the baselines presented in [6] for the task of predicting the events that will be observed for each patient in his or her next visit to the clinic (i.e. which laboratory analyses will be made next, which results will be obtained in such analyses and what medications will be prescribed next). However none of those baselines were based on RNNs.

Thus we reproduced the experiments presented in [6] including the models based on RNNs introduced in this article. Table 3 shows the result of such experiment, where we can appreciate that the Temporal Latent Embeddings model still provides better scores than the other models. We also included in Table 3 an entry named "Static Embeddings" which correspond to the predictions made with a Feedforward Neural Network using just the static information of each patient. We hypothesize that the Temporal Latent Embeddings model provides the best performance due to the lack of complex long term dependencies that are relevant for this task. Thus it would be an advantage to use a model that puts all the attention on the most recent events in situations where all the relevant information to predict the target was recorded during the previous "n" visits.

Table 3: Scores for full visit predictions. AUPRC stands for Area Under Precision-Recall Curve. AUROC stands for Area Under ROC Curve.

	AUPRC	AUROC
TLE	0.584 ± 0.0011	0.978 ± 0.0001
LSTM + static	0.571 ± 0.0048	0.975 ± 0.0002
GRU + static	0.566 ± 0.0034	0.975 ± 0.0002
Static embeddings	0.487 ± 0.0016	0.974 ± 0.0002
Random	0.011 ± 0.0001	0.5

6 Conclusion

We developed and compared different algorithms based on RNNs that are capable of combining both static and dynamic clinical variables, in order to solve the task of predicting endpoints on patients that underwent a kidney transplantation. This is an application that will provide critical information to the physicians and will support them to make better decisions.

We found that an RNN with GRUs combined with a Feedforward Neural Network provides the best scores for this task.

We also compared these recurrent models with other models for the task of predicting next actions. We found that for such use case the RNNs do not outperform another model based on a Feedforward Neural Network. We hypothesize that this is due to the lack of complex long term dependencies in the data that are relevant for this task, and therefore it is an advantage to use a model that puts all the attention on the most recent events.

Besides, we provided insight regarding how to pre-process clinical data and reported about other Neural Network architectures that did not work for our use case.

References

[1] LeCun Y, Bengio Y and Hinton G. Deep learning. Nature 521, 436444, Feb 2015.

[2] Huber L, Naik M, Budde K. Desensitization of HLA-incompatible kidney recipients. N Engl J Med. 27;365(17):1643, Oct 2011.

[3] Budde K, Becker T, Arns W, Sommerer C, Reinke P, Eisenberger U, Kramer S, Fischer W, Gschaidmeier H, Pietruck F. Everolimus-based, calcineurin-inhibitor-free regimen in recipients of de-novo kidney transplants: an open-label, randomised, controlled trial. The Lancet. 5;377(9768):837-47, Mar 2011.

[4] Budde K, Lehner F, Sommerer C, Arns W, Reinke P, Eisenberger U, Wthrich RP, Scheidl S, May C, Paulus EM, Mhlfeld A, Wolters HH, Pressmar K, Stahl R, Witzke O. Conversion from cyclosporine to everolimus at 4.5 months posttransplant: 3-year results from the randomized ZEUS study. Am J Transplant. 12(6):1528-40, Jun 2012.

[5] Bissler JJ, Kingswood JC, Radzikowska E, Zonnenberg BA, Frost M, Belousova E, Sauter M, Nonomura N, Brakemeier S, de Vries PJ, Whittemore VH, Chen D, Sahmoud T, Shah G, Lincy J, Lebwohl D, Budde K. Everolimus for angiomyolipoma associated with tuberous sclerosis complex or sporadic lymphangioleiomyomatosis: a multicentre, randomised, double-blind, placebo-controlled trial. The Lancet, 2012

[6] Esteban C, Schmidt D, Krompaß D, Tresp V. Predicting sequences of clinical events by using a personalized temporal latent embedding model. In Proceedings of the IEEE International Conference on Healthcare Informatics, 2015.

[7] Cho K, Courville A, Bengio Y, Describing multimedia content using attention-based encoder-decoder networks, arXiv preprint arXiv:1507.01053, 2015.

[8] Xu K, Ba J, Kiros R, Cho K, Courville A, Salakhutdinov R, Zemel R, Bengio Y. Show, attend and tell: Neural image caption generation with visual attention. In Proceedings of The 32-nd International Conference on Machine Learning, 2015.

[9] Choi E, Taha Bahadori M, Sun J. Doctor AI: Predicting clinical events via recurrent neural networks. arXiv preprint arXiv:1511.05942, 2015.

[10] Mikolov T, Sutskever I, Chen K, Corrado G, Dean J. Distributed representations of words and phrases and their compositionality. NIPS, 2013.

[11] Mikolov T. Statistical Language Models based on Neural Networks. PhD thesis, Brno University of Technology, 2012.

[12] Sutskever I, Vinyals O, V. Le Q. Sequence to Sequence Learning with Neural Networks. NIPS 2014.

[13] Coresh J, Selvin E, Stevens LA, Manzi J, Kusek JW, Eggers P, Van Lente F, Levey AS. Prevalence of chronic kidney disease in the United States. JAMA, 2007.

[14] Bengio Y, Frasconi P, Simard P. The problem of learning long-term dependencies in recurrent networks. pages 11831195, IEEE Press, 1993.

[15] Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Computation, 9(8):17351780, 1997.

[16] Graves A, Mohamed A, Hinton G. Speech recognition with deep recurrent neural networks. In ICASSP2013, pages 66456649. IEEE, 2013

[17] Cho K, van Merrienboer B, Bahdanau D, Bengio Y. On the properties of neural machine translation: Encoder-decoder approaches. arXiv preprint arXiv:1409.1259, 2014.

[18] Chung J, Gulcehre C, Cho K, Bengio Y. arXiv preprint arXiv:1412.3555.

[19] Cho K, van Merrienboer B, Bahdanau D. On the Properties of Neural Machine Translation: EncoderDecoder Approaches. arXiv preprint arXiv:1409.1259.

[20] Duchi J, Hazan E, Singer Y. Adaptive subgradient methods for online learning and stochastic optimization. The Journal of Machine Learning Research, pages 2121-2159, 2011.

[21] Tieleman T, Hinton G. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. Coursera: Neural Networks for Machine Learning, 4, 2012