Predictive Clinical Decision Support System with RNN Encoding and Tensor Decoding

Yinchong Yang
Ludwig-Maximilians-Universität München, Munich, Siemens AG, Corporate Technology, Munich
ynichong.yang@siemens.com

Peter A. Fasching
Department of Gynecology and Obstetrics, University Hospital Erlangen,
Comprehensive Cancer Center Erlangen-EMN,
Friedrich-Alexander University Erlangen-Nuremberg, Erlangen, Germany
peter.fasching@uk-erlangen.de

Markus Wallwiener
Department of Gynecology and Obstetrics, University Hospital Heidelberg, Heidelberg, Germany

Tanja N. Fehm
Department of Gynecology and Obstetrics, University Hospital Düsseldorf,
Heinrich-Heine University Düsseldorf, Düsseldorf, Germany

Sara Y. Brucker
Department of Gynecology and Obstetrics, University Hospital Tübingen, Tübingen, Germany

Volker Tresp
Ludwig-Maximilians-Universität München, Munich, Siemens AG, Corporate Technology, Munich
volker.tresp@siemens.com

1 Introduction

With the introduction of the Electric Health Records (EHR), large amounts of digital data become available for analysis and decision support [1]. These data, as soon as carefully cleaned and well preprocessed, could enable a large variety of analysis and modeling tasks that can improve healthcare services and patients experience [2]. When physicians are prescribing treatments to a patient, they need to consider a large range of data variety and volume. These data might include patients' genetic profiles and their entire historical clinical protocols. With the growing amounts of data decision making becomes increasingly complex. Machine learning based Clinical Decision Support (CDS) systems can be a solution to the data challenges [3] [4] [5]. Machine learning models and decision support systems have been proven to be capable of handling —and actually even profiting from— large amount of data in high dimensional space and with complex dependency characteristics. Some powerful machine learning models generate abstract and yet informative features from a usually sparse feature space.

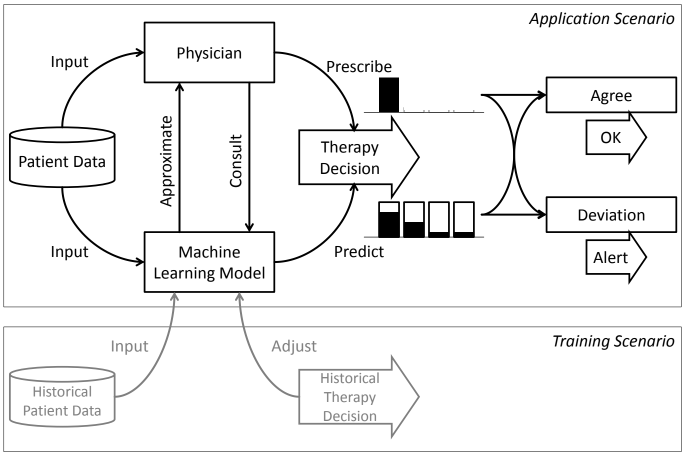
There are multiple ways that a machine learning model may impact the decision process of a physician: either indirectly, by predicting the possible outcome of each decision; or directly by calculating recommendation scores for all possible actions. As an example of the former case, [5] provides physicians with endpoint predictions of each patient queuing for a kidney transplantation. Based

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

on the predicted probabilities of kidney rejection, kidney loss and death, within the next 6 and 12 months, the physician may decide which candidate should receive a donated kidney.

In this work we focus on another other class of decision support in which the physicians' decision is directly predicted. Concretely, the model would assign higher probabilities to decisions that it presumes the physician are more likely to make. Thus the CDS system can provide physicians with rational recommendations. As a concrete use case, as soon as a physician prescribes a treatment that is not in the top-n ranked recommendations made by the CDS system, an alert would be triggered to ask the physician to reconsider the prescription and/or to document the arguments for the decision in more details. This scenario is illustrated in the upper part of the Figure 1.

Figure 1: The concept of a machine learning based CDS system. The system, as long as sufficiently trained on historic data, could predict the physician's decision set.



The proposed system is based on the predictive power of machine learning models, which are trained using historical data as illustrated in the lower half of the Figure 1. During training, the model attempts to predict historical decisions based on corresponding patient data. The actually documented decisions can adjust the model so that it can improve its predictions throughout the training epochs.

In our work we also address a problem that has not yet drawn much attention in the designing of a CDS, i.e. that a physician is required to make multiple decisions in a block, say, the intention and the type of a radiotherapy, and that these decisions are mutually dependent. In machine learning, this is known as the issue of correlation in target features. We propose a solution to the target correlation problem using a tensor factorization model.

In order to handle the patients' historical information as sequential data, we apply the so-called Encoder-Decoder-Framework which is based on Recurrent Neural Networks (RNN) as encoders and a tensor factorization model as a decoder, a combination which is novel in machine learning.

2 Decision Support for Breast Cancer Therapies

2.1 Data Description

The data, provided by the PRAEGNANT study network[6], were collected on patients suffering from metastatic breast cancer.

After preprocessing we could extract information on 1245 patients: The 1) basic information, 2) primary tumor and 3) history of metastasis before entering the study. These data are considered to be static and consist in total of 26 features of binary, categorical or real types. We performed dummy-coding on the former both cases and could extract for each patient i a static feature vector denoted with $\boldsymbol{m}_i \in \mathbb{R}^{114}$

The sequential information consists of two categories. Firstly, 4) local recurrences, 5) metastasis and 6) visits cover information on the clinical events, such as progression and live status, to which we refer as *events*. Secondly, there are three types of *therapy* informations: 7) radiotherapies, 8) systemic therapies and 9) operations. These sequential data, being only either binary or categorical, are also dummy-coded. The total number of actual events for each patient varies from 1 to 23 and is on average 5. We retrieve for each patient i at a time step t a feature vector $\boldsymbol{x}_i^{[t]} \in \{0,1\}^{182}$ and denote the whole sequence for patient i up to time t using a set of $\{\boldsymbol{x}_i^{[t']}\}_{t'=1}^{t}$.

Based on the static and all the sequential information up to a time step $t-1$, we attempt to model how the radiotherapy at t should be prescribed, which consists of two 3-D feature vectors: The first one is the therapy intention, being either curative, palliative or unknown. The second target feature is the therapy type, being either percutaneous, Brachytherapy or unknown. We denote these as $\boldsymbol{y}_{\langle i,t\rangle} \in \{0,1\}^3$ and $\boldsymbol{z}_{\langle i,t\rangle} \in \{0,1\}^3$ for patient i at time t. We perform Pearson's χ-squared tests and G-Tests on the two target features and could verify the existence of correlations, with test-statistics of $\chi^2 = 197.17$ and $G = 146.48$ respectively (DF=4). The p-values in both cases are less than 2.2e-16 according to the implementation in the statistical programming framework of R [7] [8].

2.2 Modeling Solution

First we associate the sequential and static data on a patient i up to a time step t by constructing a latent representation vector:

$$\boldsymbol{a}_{\langle i,t\rangle} = (g(\{\boldsymbol{x}_i^{[t']}\}_{t'=1}^{t}) \mid \sigma(\boldsymbol{H}^T \boldsymbol{m}_i)), \tag{1}$$

where g denotes an RNN encoder that outputs only the last hidden state given the input sequence $\{\boldsymbol{x}_i^{[t']}\}_{t'=1}^{t}$. This approach was first proposed by [9] where such RNN is proven to be able to encode all information in a sequence of variable length into a single fixed size real vector. We exploit this aspect of such RNN encoders and apply this vector to represent the patient-specific temporal information indexed by $\langle i,t\rangle$. We augment this information by concatenating it with with the static patient information, which is log-linearly encoded by a matrix $\boldsymbol{H}$ and the sigmoid activation function [5].

In order to take into account the verified correlation between target features we propose to model the probability of each possible *pair* of target feature values, corresponding to the *joint* probability distribution of each feature value. The probability of observing a value pair $\langle$intention$= j$,type$= k\rangle$ given the static and sequential input data would be calculated as:

$$\mathbb{P}((\boldsymbol{y}_{\langle i,t\rangle})_j = 1 \wedge (\boldsymbol{z}_{\langle i,t\rangle})_k = 1 | \boldsymbol{m}_i, \{\boldsymbol{x}_i^{[t']}\}_{t'=1}^{t}) =: (\boldsymbol{U}_{\langle i,t\rangle})_{j,k} \text{ where} \tag{2}$$

$$\boldsymbol{U}_{\langle i,t\rangle} = \boldsymbol{y}_{\langle i,t\rangle} \otimes \boldsymbol{z}_{\langle i,t\rangle} \in \{0,1\}^{3\times 3}, \tag{3}$$

where we construct a new target feature *matrix* $\boldsymbol{U}$ by calculating the outer product of the dummy coded feature *vectors*. Therefore each entry in the matrix represents the probability that a certain pair of target feature values is observed on these two features. Considering also the $\langle$patient, time$\rangle$-dimension we have a 3-way tensor as modeling targets.

Now we construct our regression model toward the target tensor in a fashion similar to Tucker factorization:

$$\hat{u}_{\langle i,t\rangle,j,k} = \sigma([\![\boldsymbol{\mathcal{G}};\ \boldsymbol{a}_{\langle i,t\rangle}, \boldsymbol{b}_j, \boldsymbol{c}_k]\!] = \sigma(\boldsymbol{a}_{\langle i,t\rangle}^T \boldsymbol{G}_{(1)} vec((\boldsymbol{c}_k \otimes \boldsymbol{b}_j)^T)) \tag{4}$$

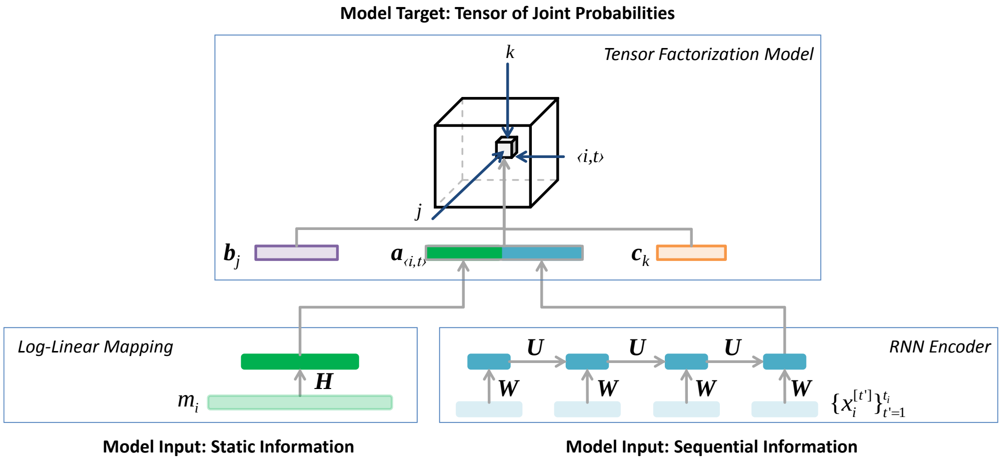
where $\boldsymbol{G}_{(1)}$ represents the unfolding of tensor $\boldsymbol{\mathcal{G}}$ w.r.t the 1st dimension following [10].

The entire model architecture, illustrated in Figure 2, is trained end-to-end.

3 Experiments

We assign 80% of all the 1245 patients to the training set and 20% to the test set. Then we query all the static and sequential information as well as targets belonging to the training patients and test patients respectively. As the RNN model we choose the LSTM with hidden state of size 25 and map the static information into a latent space of size 15 using a simple log-linear mapping matrix $\boldsymbol{H}$ as in Figure 2. As tensor model we implement a rank 5 Tucker-3 model. We apply 0.1-dropout regularization for the RNN part and 0.01-ridge regularization for the rest of the parameters in the model. All models are implemented using Theano [11] [12] and Keras [13]. All models, including

Figure 2: An overview of the entire model architecture. The latent embedding vector $\boldsymbol{a}_{\langle i,t\rangle}$ is derived from the static and sequential information, and is the input to the target tensor.


Model Target: Tensor of Joint Probabilities
Tensor Factorization Model
k
‹i,t›
j
$\boldsymbol{b}_j$
$\boldsymbol{a}_{\langle i,t\rangle}$
$\boldsymbol{c}_k$
Log-Linear Mapping
H
m_i
U
U
U
W
W
W
W
RNN Encoder
$\{x_i^{[t']}\}_{t'=1}^{t_i}$
Model Input: Static Information
Model Input: Sequential Information

the trainable baseline model for the marginal distribution of feature values, are trained up to 1000 epochs using RMSprop algorithm with learning rate 0.001. We sample 20% within the training set to be a validation set for the early-stopping mechanism and generate 5 different splittings of the data to perform cross-validations and report the mean and standard deviation of all 5 predictions for each target group. As objective function we define the sum of all binary cross entropies for each target feature as

$$-\sum_{\forall i}\sum_{t=1}^{T_i}\left[\sum_{j=1}^{3}\sum_{k=1}^{3}\delta\left(u_{\langle i,t\rangle,j,k},\ \hat{u}_{\langle i,t\rangle,j,k}\right)\right] \quad \text{with } \delta(y,\hat{y}) = y\cdot\log(\hat{y}) + (1-y)\cdot\log(1-\hat{y}) \quad (5)$$

As baseline models we test 1) random guessing; 2) most popular prediction, which is to constantly predict the frequency of each target feature value in the training data; and 3) a standard model that predicts the marginal distributions. Since our tensor model predicts the joint distribution, the predictions of both models are not directly comparable. We calculate the outer product of the marginal distributions from the baseline models, which would serve as a pseudo joint probability with which we can compare against our tensor model.

We evaluate the performances of all models in term of the AUROC which is standard for classification tasks. This metric can be here interpreted as the capability of the model to assign a patient group to a specific feature value. In our application scenario, however, we find it of more relevance that the CDS system can generate for each patient a set of rational therapy recommendations. In order to also test this aspect we evaluate the models using ranking-based metrics such as Coverage Error [14] and Ranking Precision [15] from the scikit-learn tool-box [16] and NDCG@k [17].

We report in Table 1 our experimental results. We can verify that, with the present experimental setting, the tensor-target model indeed provides better performances by considering the correlations between target feature values. Especially in term of NDCG@5, which is a common metric in assessing recommender systems, the tensor model improves the prediction by as much as 56% .

Table 1: Experimental Results

Metrics	Random	Most Popular	Standard Model	Tensor Model
AUROC	0.489±0.014	0.587±0.165	0.842±0.009	**0.874**±0.012
Coverage Error	138.846±0.740	118.569±11.053	28.197±1.344	**27.902**±1.084
Rank Precision	0.098±0.008	0.079±0.022	0.226±0.008	**0.296**±0.009
NDCG@5	0.066±0.002	0.047±0.031	0.172±0.002	**0.269**±0.014

4 Discussions

The major contribution of our work is to propose a novel model architecture by combining an RNN encoder with a tensor factorization model, both of which share a common latent representation. The tensor factorization component models the joint probability of target feature values instead of the marginal ones. We hypothesize that our model is therefore capable of capturing correlations among target features, which is often relevant in modeling clinical decisions. We show that the proposed model does achieve better prediction performances in experiments with real-world datasets. Within this paper we only conduct prediction of the radiotherapy. We shall extend our work to the other two therapies in our dataset: the systemic therapy and operations. Especially the former one has three targets and would require a 4-D tensor.

References

[1] R. Rahman and C. K. Reddy, "Electronic health records: a survey," *Healthcare Data Analytics*, vol. 36, p. 21, 2015.

[2] V. Tresp, M. Overhage, M. Bundschus, S. Rabizadeh, P. Fasching, and S. Yu, "Going digital: A survey on digitalization and large scale data analytics in healthcare," *arXiv preprint arXiv:1606.08075*, 2016.

[3] E. Choi, M. T. Bahadori, and J. Sun, "Doctor ai: Predicting clinical events via recurrent neural networks," *arXiv preprint arXiv:1511.05942*, 2015.

[4] C. Esteban, D. Schmidt, D. Krompaß, and V. Tresp, "Predicting sequences of clinical events by using a personalized temporal latent embedding model," in *Healthcare Informatics (ICHI), 2015 International Conference on*. IEEE, 2015, pp. 130–139.

[5] C. Esteban, O. Staeck, Y. Yang, and V. Tresp, "Predicting clinical events by combining static and dynamic information using recurrent neural networks," *arXiv preprint arXiv:1602.02685*, 2016.

[6] P. Fasching, S. Brucker, T. Fehm, F. Overkamp, W. Janni, M. Wallwiener, P. Hadji, E. Belleville, L. Häberle, F. Taran, D. Luftner, M. Lux, J. Ettl, V. Muller, H. Tesch, D. Wallwiener, and A. Schneeweiss, "Biomarkers in patients with metastatic breast cancer and the praegnant study network," *Geburtshilfe Frauenheilkunde*, vol. 75, no. 01, pp. 41–50, 2015.

[7] R Core Team, *R: A Language and Environment for Statistical Computing*, R Foundation for Statistical Computing, Vienna, Austria, 2016. [Online]. Available: https://www.R-project.org/

[8] A. S. et mult. al., *DescTools: Tools for Descriptive Statistics*, 2016, r package version 0.99.17. [Online]. Available: http://CRAN.R-project.org/package=DescTools

[9] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in neural information processing systems*, 2014, pp. 3104–3112.

[10] T. G. Kolda and B. W. Bader, "Tensor Decompositions and Applications," *SIAM Review*, 2009.

[11] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[12] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of SciPy*, 2010.

[13] F. Chollet, "Keras: Deep learning library for theano and tensorflow," https://github.com/fchollet/keras, 2015.

[14] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Mining multi-label data," in *Data mining and knowledge discovery handbook*. Springer, 2009, pp. 667–685.

[15] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. Džeroski, "An extensive experimental comparison of methods for multi-label learning," *Pattern Recognition*, vol. 45, no. 9, pp. 3084–3104, 2012.

[16] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] G. Shani and A. Gunawardana, "Evaluating recommendation systems," in *Recommender systems handbook*. Springer, 2011, pp. 257–297.