Medical Recommender Systems based on continuous-valued logic and multi-criteria decision operators, using interpretable neural networks

Juan G. Diaz Ochoa

PerMediQ GmbH

Orsolya Csiszár (✉ csiszar.orsolya@nik.uni-obuda.hu)

Óbuda University

Thomas Schimper

Knowledgepark GmbH

Research Article

Keywords: Recommender System, Artificial Intelligence, Deep Learning, Continuous Valued Logics, Data Privacy, Health Records

Posted Date: February 16th, 2021

DOI: <https://doi.org/10.21203/rs.3.rs-199580/v1>

License: This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Medical Recommender Systems based on continuous-valued logic and multi-criteria decision operators, using interpretable neural networks

Juan G. Diaz Ochoa
PerMediQ GmbH
Salzbergweg 18 • 85368 Wang

Orsolya Csiszár
Faculty of Basic Sciences
University of Applied Sciences, Esslingen • Esslingen
&
Physiological Controls Research Center, Óbuda University • Budapest, Hungary

Thomas Schimper
Knowledgepark GmbH
Leonrodstr. 68 • 80636 München

Abstract

Background

Out of the pressure of Digital Transformation, the major industrial domains are using advanced and efficient digital technologies to implement processes that are applied on a daily basis. Unfortunately, this still does not happen in the same way in the medical domain. For this reason, doctors usually do not have the time or knowledge to evaluate all alternative treatment options for an individual patient accurately and individually. However, physicians can reduce their workload by using Recommender Systems, still having every decision under control. In this way, they also get an insight into how other physicians make treatment decisions in each situation. In this work, we report the development of a novel recommender system that uses predicted outcomes based on continuous-valued logic and multi-criteria decision operators. The advantage of this methodology is that it is transparent, since the model outcomes emulate logical decision processes based on the hierarchy of relevant physiological parameters, and second, it is safer against adversarial attacks than conventional deep learning methods since it drastically reduces the number of trainable parameters.

Methods

We test our methodology in a patient population with diabetes and heart insufficiency that becomes a therapy (beta-blockers, ACE or Aspirin). The original database (Pakistan database) is publicly available and accessible via the internet. However, to explore methods to protect the patient's identity and guarantee data privacy we implemented a methodology on a variable-by-variable basis by fitting a sequence of regression models and drawing synthetic values from the corresponding predictive distributions using linear regressions and norm rank.

Furthermore, we implemented a deep-learning model based *on* logical gates modeled by perceptrons with fixed weights and biases. While a first trainable layer automatically recognizes a meaningful parameter hierarchy, the implemented Logic-Operator Neuronal Network (LONN) simulates cognitive processes like a rational, logical thinking process, considering that this logic is joined by fuzziness, i.e., logical operations are not exact but essentially fuzzy due to the implemented continuous-valued operators.

The predicted outcomes of the model (kind of therapy -ACE, Aspirin or beta-blocker- and expected therapy time of the patient) are then implemented in a recommender system that compares two different models: model 1 trained on a population excluding negative outcomes (patient group 1, with no patient dead and long therapy times) and a model 2 trained on the whole patient population (patient group 2). In this way, we provide a recommendation of the best possible therapy based on the outcome of model 1 and the confidence of this recommendation when the outcome of model 1 is compared with the outcome of model 2.

Results

With the applied method for data synthetization, we obtained an error of about 1% for all the relevant parameters. Furthermore, we demonstrate that the LONN models reach an accuracy of about 75%. After comparing the LONN models against conventional deep-learning models we observe that our implemented models are less accurate (accuracy loss of about 8%). However, the loss of accuracy is compensated by the fact that LONN models are transparent and safe because the freezing of training parameters makes them less prone to adversarial attacks. Finally, we predict the best therapy as well as the expected therapy time. We were able to predict individualized therapies, which were classified as optimal (binary value) when the prediction fully matched predictions made with models 1 and 2. The results provided by the recommender system are displayed using a graphical interface. The current is a proof of concept to improve the quality of the disease management, while the methods are continuously visualized to preserve transparency for the customers.

Conclusions

This work contributes to boost administrative functions and quality of management of patients and improve the quality of healthcare with models that are both transparent and safe. Our methodology can be extended to different clinical scenarios where recommender systems can be applied. The acceptance and further development of the app is one of the next important steps and still requires further development depending on specific requirements of the health management, the physicians or health professionals, and the patient population.

Keywords

Recommender System; Artificial Intelligence; Deep Learning; Continuous Valued Logics; Data Privacy; Health Records

Background

The management and quality of hospital services depend to a large extent on individual medical decisions. For example, based on their experience, each physician may select treatments coded into specific keys (Treatment Keys, or TK) for disease management and the cost of treatment; each of these treatments is then combined with a cost refund respectively which is encoded in specified keys in the Electronic Health Records (EHR) and on medical bills. These keys would have an ‘economic effect’ on the health system (Reddy and Aggarwal, 2015). The reimbursement system's behavior, in some cases, enable what physicians could do to offer more health services to help a patient, regardless of whether the additional care is economically optimal. At the same time, physicians can also ignore their colleagues' experiences by using alternative treatments that might be more suitable for treating a disease. Other deficiencies may stem from the fact that physicians are either overloaded by a large number of patients and/or by several therapeutic options to consider, which is part of a portfolio that has more than 100 different alternative treatments per patient (Hall and Walton, 2004). Because of this, physicians do not usually have the time or knowledge to evaluate all of the different alternative options accurately and individually.

By using Recommender System, physicians can reduce their workload, yet still have each decision made under their control. They will also get an insight into how other physicians recommend a treatment in a same given situation.

Nowadays, recommender systems have proven to be invaluable for online users to cope with information overload in many different fields (e.g. e-commerce, decision support systems, etc.) (Ricci et al., 2011), (Bouayad et al., 2020). However, the architecture of recommender systems and their evaluation in real-world problems is still an active area of research. We have investigated and implemented a recommender system intending to transfer this technology in the field of medicine to support physicians seeking to predict the *rating* or *preference* of a treatment-key for new patients.

Recommender systems in medicine is not new. There are about 911 search items in published medical journals, with articles reporting recommender systems for patients using personal health systems (Wiesner and Pfeifer, 2014). Until the last few years, most of these techniques were used for analyzing rich EHR-data based on traditional machine learning and statistical techniques such as logistic regression, support vector machines (SVM) (Shickel et al., 2018), and random forests. However recently, deep learning techniques have achieved great success in many domains through deep hierarchical feature construction and capturing long-range dependencies in data effectively, particularly for the analysis of EHRs in medicine (Rajkomar et al., 2018).

Here, we are introducing a ‘different recommender system’ which is a combination of methods to generate synthetic populations while keeping personal data protected when needed for Artificial Intelligence (AI) applications and the use of novel methods based on continuous-valued logic and multi-criteria decision operators

aimed for a robust, safer, and more understandable use of deep learning. By combining neural networks with continuous logic and multi-criteria decision-making tools, thus reducing in this way the black-box nature of neural models (Csiszár et al., 2020a, 2020b; Riegel et al., 2020; Shi et al., 2019). By doing this, we are exploring the (often overlooked) possibility of combining neural networks with continuous logical systems. This strategy provides a clear advantage in the medical field since it is a system that, due to its nature, can be easily understood by physicians and/or medical practitioners, who often make their decisions relying on continuous logical rules. We aim to reach more transparency of AI applications in medicine while preserving efficient deep learning methods. The synthetic populations are mainly used for training the Deep Learning machinery. They are completely anonymized yet keep their original structure of the original data sample used for ML use.

Our customers consist of physicians handling individual patient diagnosis. In this way, by providing these recommendations, we are seeking to reach both an economic and a clinical efficacy:

- Persuade physicians to make use of best-suited treatment keys aim at reducing the costs in the management of patients,
- Improve the management of a diagnose of the disease by helping the physicians to discover additional keys that could improve the treatment of patients.

These two bullets seem to be contradicting, but they represent the typical conflict of interest what each physician faces daily. Physicians should act accordingly to the following principles:

- Best medical care for the patient with optimum economic impact (i.e cost efficiency) for those who have to pay the bill finally. So, if there are two equal treatments available, the system should recommend the cheaper one with the same if not better treatment quality result than the more expensive option.
- Depending on the health system of a particular country and how billing/charging is carried out, there may be another conflict that the physician must contend with: optimizing the cost-income ratio of the organization that is treating patients.

As the system is learning from the best Physicians in the country, the recommendations made could be seen as best advice from the best physicians around.

Our recommender system consists of a collaborative filtering approach. The goal is to build a model from the past behavior of several physicians selecting treatment keys that are correlated to the patient's diagnosis keys (codified by the ICD system). Therefore, when a new patient with specific ICDs enters the system, the physician is shown a recommendation list of the plausible treatment keys for this patient with the most effective therapy. Thus, to extract structured medical concepts, such as diseases and treatment procedures, we use single-concept extraction in electronic medical records (Shickel et al., 2018).

In the next section, we will introduce the architecture implemented in this project. After that, we will describe the implemented methodology and present the validation results compared to conventional dense networks. Then we will further show the visualization method implemented in a recommender system. Finally, we will discuss the implication of this implementation and conclude with a deviation on how this methodology can be further improved.

Methods

Solution architecture

The implemented recommender system analyzes the frequency of medical events in the EHR and delivers a recommendation based on the preferred events. Therefore, our system works like collaborative filtering, i.e. items are chosen based on the patients' rating history. This implies that our system, unlike systems implemented by Amazon, do not use details of the registered user's profile (i.e., the physician)

The workflow of the implemented recommendation system requires the ICDs, age, sex to predict the more probable TKs per patient as parameters; and the following steps ensue:

- Synthesize the patient's information and store the result with the relevant patient parameters: age, sex, IDs, ICDs.
- Cluster ICDs and TKs. This step is required to reduce the dimensionality of both parameters (high number of items) and perform predictions of TKs group number depending on patient parameters, including ICDs groups.
- Train (deep learning model), validate, and export model to medical/hospital documentation and information system.
- Introduce a user interface for the recommender system, using the trained deep learning model based on the medical information system's data to recommend the treatment keys. As part of the combination of medical/hospital documentation and information system/recommender system this is the process where the physician accepts or discards the recommended TK in his/her professional autonomy and charges it via the charging system.

Figure 1 Structure of the implemented recommender system

The different steps in the workflow are resumed in Figure 1. Once the recommender system is defined, using it implies the following steps:

- The physician asks for a recommendation of the most frequently used TKs using the recommender system.
- The physician gets a recommendation.
- The physician selects the appropriate recommendations and does the treatment based on his own decision.
- The new updated information about selected TKs is stored then in the database (DB).

The implementation of this system is far from trivial, and we have to overcome several challenges:

- Data privacy:** generating of synthetic data from the original microdata containing confidential information so that they are safe to be released to users. Synthetic data are generated from sensitive records by replacing them with values simulated from probability distributions specified to preserve the actual observed data's key features.
- Data extraction:** By using our techniques, we can synthesize large numbers of patients' data, however we also require robust techniques to filter large data amounts to synthesized data efficiently based on parallelization methods. To this end, data pre-processing was implemented using Spark.
- Model selection:** Selection of a robust model enable us to establish a relation between diagnoses and TKs.
- Model validation and group deciphering.**
- Final model consumption and visualization.**

In the following sections, we provide solution strategies for each one of these challenges.

Modeling

Synthetic populations

Synthetic patients are generated from a representative patient database with samples of diabetic patients, some with heart insufficiency (Pakistan Database, from the UCI repository¹). Diabetic patients with heart insufficiency often impose onerous requirements and restrictions for their lives. A clinical, as well as the economic impact, can be obtained with improved management of the disease based on a recommender system applied to Electronic Health Records (EHR) predicting “preference” (best clinical outcome) and “rating” (best reimbursement) that a physician would give to an item² encoded in treatment keys (TK).

In the first instance, we only focus on heart insufficiency patients and combine this database with already known kinds of therapies. The goal is to combine the patient data with data from diabetic patients (group numbers extracted and analyzed in the recommendation system). To complement the data corresponding to the therapy, we would also include the treatment of heart insufficiency (Edelmann et al., 2011). More than 90% of heart failure patients with reduced ejection fraction (systolic heart failure, or **SHF**) and diabetes were treated with an ACE inhibitor (ACEi) or angiotensin receptor blocker (ARB) or with beta-blockers. By contrast, patients with diabetes and preserved ejection fraction (Heart Failure with Preserved Ejection Fraction, or **HFNEF**) were less likely to receive these substance classes ($p < 0.001$) and had the worst blood pressure control ($p < 0.001$). Compared to patients without diabetes, the probability of receiving these therapies was increased in diabetic HFNEF patients ($p < 0.001$), but not in diabetic SHF patients. Aldosterone receptor blockers were given more often on diabetic patients with reduced ejection fraction ($p < 0.001$), and the presence and severity of diabetes decreased the probability to receive this substance class, irrespective of renal function.

Therefore, the hybrid database with typical TK and ICD groups (non-identifiable) combined with the Pakistan database for diabetic patients with heart insufficiency had the following attributes:

- Attributes from the Pakistan Database, including age and sex.
- ICD Groups (numbers – non-identifiable from original database).
- TK Groups + treatment group for heart insufficiency.

The total number of parameters, included input (V1 to V10) and output (O1 and O2) parameters, is illustrated in Table 1.

Table 1 Principal input and output parameters extracted from the HER of diabetic patients with heart insufficiency.

Input / Output		Variable	Kind of parameter
-	ID		Character
V1	Sex		Binary
V2	Age		Real value
V3	Creatinine_phosphokinase		Real value
V4	Ejection_fraction		Real value
V5	High blood pressure		Binary
V6	Platelets		Real value
V7	Serum_creatinine		Real value
V8	Serum_sodium		Real value
V9	Smoking		Binary

¹ The data has been obtained from the [uci repository](<https://archive.ics.uci.edu/ml/datasets/Heart+failure+clinical+records>)

² In this case, the best treatment

V10	Diagnose (kind of hearth insufficiency - HF)	Character
O1	Time (feedback period)	Real value
O2	Treatment key TK (AEC, Aspirin or Betablocker)	Natural number

Notice that the parameter “Time” is the total number of days that the patient has been treated, in relation to the entire six months when this database has been obtained. Short times can mean that either the patient has recently entered into the system or that the therapy outcome was negative and/or the patient has deceased.

To better assess the quality of therapy, we used the treatment outcome as a reference parameter to generate a control population with a positive therapy outcome (no death patients).

The steps required to generate the synthetic population are (see Figure 2):

1. first, a query of different databases to extract a table with anonymized patient’s parameters; before modeling the patient’s clinical profile,
2. then we model the distribution of each one of the parameters in the population and use the distribution functions to model and clone fully synthetic patients,
3. finally, the data is stored into a database for further modeling when needed.

Figure 2 Workflow for the generation of synthetic patients. From this data, we analyze the parameter distribution and generate entirely new synthetic parameters using these distributions, which meant that we could generate synthetic patient populations completely and that this data provides more anonymization of the original clinical data.

Personal information, like the identification number (ID), age, and sex, are randomly generated. While the first parameter can be generated using a simple uniform random distribution, the last two parameters depend on the population's structure, i.e., whether there are more men than women and how the patients' age influenced the distribution of the disease. These are parameters that cannot be modeled using a model and thus require the estimation of the distribution structure. For the simulation of the distribution of diagnoses and treatment keys (TKs, in figure 2), we used the “synthpop” package³ (Nowok et al., 2016). A more detailed description of the performed data synthesis is provided in the supplementary material.

Deep-learning methods

The recommender system is based on the fuzzy classification system, such that the ICD groups, patients' age, and sex are used as input parameters to predict the TKs.

³ <https://www.r-bloggers.com/generating-synthetic-data-sets-with-synthpop-in-r/>

To this end, a deep-learning method based on a multilayer neural net (NN) has been implemented. A dense NN is a type of artificial neural network (ANN) composed of multiple hidden layers, where every neuron in layer i is fully connected to every other neuron in layer $i + 1$. Typically, these networks are limited to a few hidden layers, and the data flows only in one direction, unlike recurrent or undirected models⁴.

Extending the notion of a single layer ANN, each hidden unit computes h_i a weighted sum of the outputs from the previous layer, followed by a non-linear activation σ of the calculated sum as

$$h_i = \sigma \left(\sum_{j=1}^d x_j w_{ij} + b_i \right)$$

Here, d is the number of units in the previous layer, x_j is the output from the previous layer's j th node, and w_{ij} and b_i represent the weight and bias terms associated with each x_j .

Considering that neural networks emulate the spikes in neuronal processes, its result as a logical choice to select sigmoidal activation functions between different neurons in the different layers is clear. Recent investigations have demonstrated that rectified linear functions are the most effective in representing data processing in neural networks⁵, particularly for networks with many layers, and thus more effective to process information (Urenda et al., 2020a). In our investigation, we replaced these kinds of dense neuronal layers with continuous-valued logical operators in the last layers that can emulate fuzzy logical operations.

Continuous-valued logic multi-criteria decision operators and interpretability

Our strategy consists of implementing networks based on logical gates, modeled by Perceptron with fixed weights and biases. This hybrid neural model was introduced in (Csiszár et al., 2020a). Here, a single Perceptron in the NN network is activated by so-called Squashing activation functions, differentiable, parametric family of functions that satisfy natural invariance requirements and contain rectified linear units as a particular case (Urenda et al., 2020a; Zeltner et al., 2020). These Squashing functions approximate the cutting function in the nilpotent logical operators. A relevant characteristic of this family is its differentiability, which is vital for employing gradient-based optimization techniques.

In this investigation, we implemented the following function:

$$S_{\beta}(x) = \frac{1}{\beta} \ln \left(\frac{1 + e^{\beta \cdot x}}{1 + e^{\beta \cdot (x-1)}} \right), \quad (2)$$

where β is a real nonzero value that must be adjusted to let the model be convergence.

Thus, the Perceptron in the neural networks' hidden layers can model a threshold-based nilpotent operator (Csiszár et al., 2020a, 2020b): a conjunction, a disjunction, or even an aggregative operator. *This means that the weights of the first layer is to be learned, while the hidden layers of the pre-designed neural block, worked as logical operators with frozen weights and biases.* This means:

1. In the first layer, the activation functions are fuzzy membership functions M_i , representing truth values of inequalities. We implemented an activation function $S_{\beta_{inp}}(x)$ shown in Equation 2, with β_{inp} defined exclusively for the input function.
2. At the same time, the activation functions in the hidden layers, model the cutting function to avoid the vanishing gradient problem with the so-called squashing function $S_{\beta_{int}}(x)$ in the nilpotent logical operators (Urenda et al., 2020b) (defined by equation 2), with β_{int} representing the internal layers. Besides logical operators, preference operators can also be modeled this way (Csiszár et al., 2020c).

⁴ https://en.wikipedia.org/wiki/Multilayer_perceptron

⁵ https://scholarworks.utep.edu/cgi/viewcontent.cgi?article=2170&context=cs_techrep

Table 2 Some examples of logical operators and their corresponding implementation (Csiszár et al., 2020c)

Logical operation	w_{ij}	b_i
AND	1	-1
OR	1	0
NOT (x)	0	1
NOT (y)	-1	1
Not (x) and Not (x)	-1	1

The weights in the first, H_i , and last layer, O_i , are optimized during training to establish an association between input x and output y . In the second layer, we define the nodes M_i , layers with different and frozen weights w_{ij} and biases b_{ij} (see Equation 1), grouping different relations between the input parameters. Thus, each of these nodes is essentially a hypothesis grouping of all the parameters with different statistical weights. Finally, the additional internal layers perform logical operations; some of them are resumed in Table 2. In Figure 3, we illustrate this architecture, considering 4 M_i nodes.

Figure 3 For now, with simplicity in mind, we implement two AND layers (conjunctions), followed by an OR layer (disjunction), to logically evaluate the nodes M_i , which is a process modeling human reasoning in the decision process.

Therefore, our implemented **Logic-Operator neural network (LONN)** simulates cognitive processes like rational, logical thinking process, considering that this logic is joined by fuzziness, i.e., logical operations are not exact but essentially fuzzy due to the implemented continuous-valued operators (see Figure 4). (Alvarez et al., 2020).

Figure 4 Representation of natural thinking processes (A) and by neural networks (B). Logical processes, like the combination of “and” and “or” processes, are implemented as fuzzy logic, representing natural uncertainties in the thinking process.

Even though we were inspired in this design by cognitive processes, our aim is not to reproduce the native human thinking process *in silico* but rather to implement processes much closer to the natural processes in making information processing interpretable.

Results

Model validation

The ANN has been implemented using tensor flow in an R environment. The data ingestion and pre-processing are made with Spark. The data ingested in the model is normalized, and the internal model validation is performed using this normalized data. From this data, we use 70% for model training and 30% for model validation. We implemented a control model using “ReLU” activation functions with the same topology as the LONN. The main parameters of both the control and the continuous-valued multi-criteria network are listed below in Table 3.

Table 3 Parameters for comparison a dense Layer network ReLu vs. a LONN

	NN – Dense Layers / ReLU activation	LONN
β_{inp}	-	1.0
β_{int}	-	1.5
# Internal Layers	4	4
# Neurons per layer	10, 4, 3, 2	10, 4, 3, 2
# Trainable Parameters	293	116
# Non-Trainable Parameters	0	59

For the parameters β , we performed different validation runs to explore the optimal value for the first (β_{inp}) and internal (β_{int}) layers. The effect of this parameter on the slope of the function $S_{\beta}(x)$ is shown in Figure 5.

Figure 5 Squashing function $S_\beta(x)$ for different parameters β .

Observe that the Squashing functions go through the point $(0.5, 0.5)$ for all values of β and that:

$\beta > 0$, squashing function is increasing;
 $\beta < 0$, squashing function is decreasing.

Moreover, $S_\beta(x) = S_{-\beta}(1 - x)$ holds, i.e., $S_{-\beta}(x)$ is the reflection of $S_\beta(x)$ over the axis $x=0.5$. This means that in the interpretation, for a negative β value, a negation operator is applied. This fact explains the symmetry of the training error with respect to the β values (see Figure 6).

Figure 6 Training error of the model presented in Figure 3 for different β_{inp} (different shapes) and β_{int} (x-axis) estimated after 50 epochs for each validation run.

The systematic test of the mean error considering different β_{inp} and β_{int} (Figure 6) confirms that optimal model performance is obtained for $\beta \geq 0.5$, such that single neurons in the network can be activated and respond to the external input parameters to train a model.

We performed a model validation with these parameters in mind and compared the dense control layer NN and the LONN shown in Figure 3. The results are presented in Figure 7.

Figure 7 Internal model validation with a dense NN (two internal layers, with 10 and 4 neurons with ReLU activation function, figure A), vs. the logic-operators NN (LONN, figure B) for 150 epochs.

At first glance, we observe that the model defined with continuous-valued logic multi-criteria decision operators (i.e., fuzzy network) is less accurate than conventional networks. Based on the results presented in figures 6 and 7, we observe that the implemented model delivers a precision of about 74%, which is somewhat lower than the NN with dense layers and ReLU activation functions.

Figure 8 Test of two LONNs with two different topologies with different numbers of internal layers and logical operators.

An inspection of the dependency of the model accuracy on the number of neurons in the internal layers (4 M_i neurons, **Figure 8-A**, and 10 M_i neurons with a corresponding modification of the connection of the logical operators, **Figure 8-B**) demonstrates that an increase of the number of neurons has a slight influence on the mean absolute error and that the consideration of additional logical rules does not necessarily imply a more accurate result.

In a nutshell, the observed loss of accuracy in fuzzy logic networks is compensated by the model interpretability and by the fact that we dramatically reduce the number of trainable parameters.

Model consumption and visualization of recommendations

We have trained two different models for the final model consumption, one with the whole database (Model 1, Figure 9-A) and another with a database that selects only positive outcomes (Model 2, Figure 9-A). The goal of this training method is to make predictions based on the positive outcomes and then evaluate the confidence of the prediction:

- High confidence when the predictions of both models overlap (Figure 9 - B)
- Low confidence when there is no matching. In this case, we provide recommendations based on the outcomes of model 2, but with a warning that this prediction has low confidence (Figure 9 - B)

In this way, we aim to setup a recommender system, leading to outcomes, that should be almost positive for the patients.

Figure 9 Training method for the recommender system. We train two different models based on the whole data set (Model 1) and a dataset consisting only of positive outcomes (Model 2). After that, we use both models to make different predictions. If there is a matching in the predictions, then Prediction 1 is used as the standard with high confidence; otherwise, predictions from Model 2 are provided but have low confidence.

The trained model is finally exported and used for consumption and data ingestion (as shown in Figure 1):

- The physician asks for a recommendation once he has a clear diagnose of the patient, correspondingly encoded in ICDs.
- The trained model ingests the ICDs, as well as the sex and age of the patient, and delivers corresponding TKs recommendations encoded as group numbers.
- The final TKs are decoded.
- The final recommendation is finally deployed and visualized, for instance, in an application or the physician's software.

To improve the model interpretability, we require a visualization of the distribution of the input parameters, automatically generated by the model, as shown in the supplementary section 2 (Figure S2-1). From this result, we discover a parameter hierarchy, with the creatinine concentration (serum and phosphokinase) as the relevant parameter. This result makes sense regarding the fact that the creatinine concentration is a metabolite that indicates how good the patient adherence to a therapy is. (Love et al., 2016).

Figure 10 Diagram of a potential app based on the model outcomes.

The final recommendations are then visualized in a dashboard, as is shown in figure 10. Not only the recommendations, but also the confidence of the prediction based on positive outcomes (binary value: 1 for confidence, 0 for non-confidence) is visualized. In the case of low confidence, the two alternative treatments from the two trained models are deployed. In this Dashboard the physician can provide a subjective feedback about how useful or accurate are the deployed recommendations.

Finally, indifferent to whether the recommended TKs are accepted or not by the physician, the recommendation is stored in the database. This final step implies that the database is continuously updated with new information, implying that models must be trained periodically to guarantee their quality. This implies that a robust implementation of this system requires an automatic system that allows its new training.

Discussion

As it is well-known, deep learning methods are advantageous because they allow the modeling of non-linear systems while at the same time being robust against small changes of training data, unlike methods as random forest (“4.7 Other Interpretable Models | Interpretable Machine Learning,” n.d.). With our implemented methodology, we aim to further improve deep learning methods in the following three ways:

- **Interpretability:** Implementing a combination of logical operators/gates, i.e., the different parameters are logically combined. Since the parameters of the deep layers are frozen, we can rely on the parameter classification in the first layer, which provides interpretable information about how a parameter hierarchy influences the model outcome. Furthermore, these operators are implemented using fuzzy logic, which is closer to natural thinking processes.
- **Safety:** the statistical weights and biases in the internal layers are frozen, i.e., they do not change in the training process. This guarantees that these parameters are robust, even when adversarial examples are employed.
- **Increased efficiency:** fewer parameters to be trained.

These aspects are extremely relevant when neural networks are implemented in a medical field, not only because of the high safety standards required but also because physicians must understand how the network delivers results. With our implementation of the **LONN in a deep-learning environment**, we have demonstrated that we can implement more transparent models with highly efficient computational tools.

Observe that in this approach we are providing a rather precise interpretation for an epistemological problem in machine learning that has not been fully solved (Krishnan, 2020): with our implemented LONN we know that from the extracted features in the layer M_i we perform a logical combination of the recommendations, i.e. we know which are the internal operations in the network. However, our LONN implementation is partially interpretable since the feature extraction has been trained in the Keras environment and cannot be fully interpreted. For this reason, with our methodology we can better justify a result, and we can provide a partial explanation about how the algorithm works, without pretending to provide a full causal relation between the input parameters and the model outputs.

Besides this technical milestone, several problems will persist in the implementation of recommender systems in medicine. Patient databases are dynamic and evolve depending on the disease distribution in the patient population, which can eventually invalidate trained models. Once activated, recommender models can influence physicians' decision-making, which will be reflected in how treatments are suggested to patients who are registered in the database. This fact implies a co-evolution of the database and the implemented AI model coupled to the recommender system, considering that AI models must be trained and validated regularly.

Besides selecting the correct methodology to process and analyze the data for predictive modeling, the most challenging problem is protecting patient's data. When it comes to data protection, three main aspects are problematic:

- The access to the patient's data for machine learning is limited as it always requires the patient's consent.
- Recommendation systems can generate incorrect patient profiling, which simultaneously can be misused.
- Physicians seldom get enough support in their day-to-day work.

The first problem is solved by generating databases with synthetic patients. In solving this problem, we used a small random but representative dataset of anonymized representative patients to model their individual clinical data with characteristic distribution patterns to generate completely synthetic patients. We have also developed our methodology and workflow, based solely on this synthetic database. Naturally, after training models on synthetic data, models can be further trained on real patient data with consent inside closed repositories.

The other two aspects are critical since it concerns the rights that patients do not have to be subject to decisions based solely on automated processing (Kamarinou et al., 2016). For this purpose, patients reserve the right to be informed about the existence of automated decision-making, including profiling, and their right to receive meaningful information about the logic involved and the significance and the envisaged consequences of such processing. However, whatever is recommended to a physician, it must be seen as a recommendation. The final decision on how a patient receives treatment will always be the physician's call.

The behavioral features involved in the decision-making process, i.e. how a recommender system may influence physician's decision making (Timotijevic et al., 2020), and how this technology could have a positive impact on patient's care and better economic management is a problem that not only depends on the technical implementation and the AI component but also on the final user's interface. For instance, in a recent investigation, it has been shown that, *when designed well, recommender systems that incorporate treatment costs can result in significant cost savings, while providing similar or better health outcomes. Scenarios, where practitioners do not feel time pressure and have access to accurate cost information, are most conducive for adopting recommendations and creating change* (Bouayad et al., 2020).

A deeper analysis of this aspect lies beyond the scope of the present POC-implementation and must be analyzed in future once the recommender system and its corresponding user interface run in real conditions⁶.

Conclusion

We have implemented a recommender system based only on the statistical analysis of data stored in HER and working as a collaborative filtering. The implemented system estimates the therapy time and treatment keys, TKs (in this case the use of ACE, Aspirin, or Betablocker), and implemented deep learning to predict TKs depending on the patient's diagnoses and essential phenotypic information.

We can demonstrate that our methodology reaches a precision of up to 78%. This accuracy is lower than the one obtained using conventional NN implemented with dense layers and ReLU activation functions. However, in this kind of implementation, the interpretability (architecture and emulation of rational decision processes) and safety (parameters of internal layers are frozen) are two characteristics that are perhaps much more valuable than the accuracy of the model.

At this point, it is relevant to point out that the recommender system coevolves with the whole system, i.e. the recommender system can eventually influence physicians' decisions, while such decisions, stored in the databases,

⁶ All products which offer decision support must be built under the design and software development process for medical devices.

can influence the recommender system when it is periodically re-trained. Such coupled dynamics between model re-training and model used by physicians is fundamental since such models are not static and dependent on how they are used. Such evolution must be the focus of future analysis, considering behavioral aspects related to the use of recommender systems by physicians.

Finally, recent advances in neural networks allow the visualization and recording of the learning process to leverage the safety and transparency in the use of deep learning methods (Olah et al., 2018). Such methods must be implemented in the future to increase the safety in the validation and use of these methods in the medical field.

List of Abbreviations

TK	Therapy Key
EHR	Electronic Health Record
SVM	Support Vector Machine
AI	Artificial Intelligence
DB	Data Bank
SHF	Systolic Heart Failure
HFPEF	Heart Failure with Preserved Ejection Fraction
ID	Identification Number
ANN	Artificial Neural Network
NN	Neural Network
LONN	Logic Operator Neural Network

Declarations

Ethics approval and consent to participate

No research in human / animal participants

Consent for publication

Hereby the authors provide permission to publish this article.

Availability of materials

All models and relevant data inputs are available upon request to the corresponding author.

Competing interests

JDO is associated to PerMediQ GmbH;

TS is associated to Knowledgepark GmbH;

OC declares no conflict of interest.

Funding

No funding.

Author contribution

TS Suggested the conceptual framework.

JGD & TS defined the design of the recommender system as well as the implementation strategy of the AI solution.

OC & JGD defined and implemented the continuous-valued logic and multi-criteria decision operators.

JGD implemented the AI solution and conceived the final design of the recommender system.

JGD, OC & TS worked on the manuscript.

Acknowledgements

We are very grateful for the discussions and contribution of Martin Grundman in the conceptual development of the recommendation system.

Bibliography

4.7 Other Interpretable Models | Interpretable Machine Learning [WWW Document], n.d. URL <https://christophm.github.io/interpretable-ml-book/other-interpretable.html> (accessed 12.8.20).

Alvarez, K., Urenda, J., Csiszár, O., Csiszár, G., Dombi, J., Eigner, G., Kreinovich, V., 2020. Towards Fast and Understandable Computations: Which “And”- and “Or”-Operations Can Be Represented by the Fastest (i.e., 1-Layer) Neural Networks? Which Activations Functions Allow Such Representations? Dep. Tech. Rep. CS.

Bouayad, L., Padmanabhan, B., and Chari, K. (2020). Can Recommender Systems Reduce Healthcare Costs? The Role of Time Pressure and Cost Transparency in Prescription Choice. *Manag. Inf. Syst. Q.* 44, 1859–1903.

Csiszár, O., Csiszár, G., Dombi, J., 2020a. Interpretable neural networks based on continuous-valued logic and multi-criteria decision operators. *Knowl.-Based Syst.* 199, 105972. <https://doi.org/10.1016/j.knosys.2020.105972>

Csiszár, O., Csiszár, G., Dombi, J., 2020b. How to implement MCDM tools and continuous logic into neural computation?: Towards better interpretability of neural networks. *Knowl.-Based Syst.* 210, 106530. <https://doi.org/10.1016/j.knosys.2020.106530>

Edelmann, F., Wachter, R., Düngen, H.-D., Störk, S., Richter, A., Stahrenberg, R., Neumann, T., Lüers, C., Angermann, C.E., Mehrhof, F., Gelbrich, G., Pieske, B., the German “Competence Network Heart Failure,” 2011. Heart failure therapy in diabetic patients-comparison with the recent ESC/EASD guideline. *Cardiovasc. Diabetol.* 10, 15. <https://doi.org/10.1186/1475-2840-10-15>

Hall, A., Walton, G., 2004. Information overload within the health care system: a literature review. *Health Inf. Libr. J.* 21, 102–108. <https://doi.org/10.1111/j.1471-1842.2004.00506.x>

Kamarinou, D., Millard, C., Singh, J., 2016. Machine Learning with Personal Data (SSRN Scholarly Paper No. ID 2865811). Social Science Research Network, Rochester, NY.

Krishnan, M. (2020). Against Interpretability: a Critical Examination of the Interpretability Problem in Machine Learning. *Philos. Technol.* 33, 487–502.

Love, S.A., Seegmiller, J.C., Kloss, J., and Apple, F.S. (2016). Urine Creatinine Concentrations in Drug Monitoring Participants and Hospitalized Patients. *J. Anal. Toxicol.* 40, 659–662.

Nowok, B., Raab, G.M., Dibben, C., 2016. synthpop: Bespoke Creation of Synthetic Data in R. *J. Stat. Softw.* 74, 1–26. <https://doi.org/10.18637/jss.v074.i11>

Olah, C., Satyanarayan, A., Johnson, I., Carter, S., Schubert, L., Ye, K., Mordvintsev, A., 2018. The Building Blocks of Interpretability. *Distill* 3, e10. <https://doi.org/10.23915/distill.00010>

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., Dean, J., 2018. Scalable and accurate deep learning with electronic health records. *Npj Digit. Med.* 1, 1–10. <https://doi.org/10.1038/s41746-018-0029-1>

Reddy, C.K., Aggarwal, C.C., 2015. Healthcare Data Analytics. Chapman & Hall/Crc Data Mining and Knowledge Discovery Series, Boca Raton.

Ricci, F., Rokach, L., Shapira, B., Kantor, P.B. (Eds.), 2011. Recommender Systems Handbook. Springer US. <https://doi.org/10.1007/978-0-387-85820-3>

Riegel, R., Gray, A., Luus, F., Khan, N., Makondo, N., Akhalwaya, I.Y., Qian, H., Fagin, R., Barahona, F., Sharma, U., Iqbal, S., Karanam, H., Neelam, S., Likhani, A., Srivastava, S., 2020. Logical Neural Networks. *ArXiv200613155* Cs.

- Shi, S., Chen, H., Zhang, M., Zhang, Y., 2019. Neural Logic Networks. ArXiv191008629 Cs Stat.
- Shickel, B., Tighe, P.J., Bihorac, A., Rashidi, P., 2018. Deep EHR: A Survey of Recent Advances in Deep Learning Techniques for Electronic Health Record (EHR) Analysis. *IEEE J. Biomed. Health Inform.* 22, 1589–1604. <https://doi.org/10.1109/JBHI.2017.2767063>
- Timotijevic, L., Hodgkins, C.E., Banks, A., Rusconi, P., Egan, B., Peacock, M., Seiss, E., Touray, M.M.L., Gage, H., Pellicano, C., Spalletta, G., Assogna, F., Giglio, M., Marcante, A., Gentile, G., Cikajlo, I., Gatsios, D., Konitsiotis, S., Fotiadis, D., 2020. Designing a mHealth clinical decision support system for Parkinson’s disease: a theoretically grounded user needs approach. *BMC Med. Inform. Decis. Mak.* 20, 34. <https://doi.org/10.1186/s12911-020-1027-1>
- Urenda, J., Csiszár, O., Csiszár, G., Dombi, J., Kosheleva, O., Kreinovich, V., Eigner, G., 2020a. Why Squashing Functions in Multi-Layer Neural Networks. *Dep. Tech. Rep. CS.*
- Wiesner, M., Pfeifer, D., 2014. Health Recommender Systems: Concepts, Requirements, Technical Basics, and Challenges. *Int. J. Environ. Res. Public. Health* 11, 2580–2607. <https://doi.org/10.3390/ijerph110302580>
- Zeltner, D., Schmid, B., Csiszar, G., Csiszar, O., 2020. Squashing activation functions in benchmark tests: towards eXplainable Artificial Intelligence using continuous-valued logic. ArXiv201008760 Cs.

Supplementary 1

For the simulation of the distribution of diagnoses and treatment keys (TKs, in Figure 2), we used the “synthpop” package⁷ (Nowok et al., 2016). *Synthesis is performed on a variable-by-variable basis by fitting a sequence of regression models and drawing synthetic values from the corresponding predictive distributions. The fitted models are conditioned on the original variables so that the number of covariates increases for subsequent variables. In this approach, models can be defined for each variable separately, and structural features of the data, such as logical constraints or missing data patterns, can be considered.*

In this research, we implemented linear regression and norm rank⁸. Using this method, synthetic values of normal deviates of ranks of the values in $\mathbf{y}$ are generated using the spread around the fitted linear regression line of normal deviates of ranks given $\mathbf{x}$. Then normal synthetic deviates of ranks are transformed back to get synthetic ranks, which are used to assign values from $\mathbf{y}$. First, the regression coefficients are drawn from a normal distribution with mean and variance from the fitted model for proper synthesis.

The regression is carried out on Normal deviates of ranks in the original variable. Synthetic values are assigned from the original values based on the synthesized ranks that are transformed from their normal synthesized deviates. With the applied method, we obtained an error⁹ of about 1% for all the relevant parameters (see **Figure S1-1**).

Figure S1-1 Boxplot of error distribution of different exemplary parameters in the dataset

To generate a complete synthetic population, we modeled and simultaneously synthesized the data N for new patients, such that the final population is the number of the initial patient population multiplied by the number of clones per synthesized patient. We performed an arbitrary synthesizing of 10 patients for each patient, implying that we generated a synthetic population of approximately $N = 2000$ individuals. In this way, we obtained an extended database to apply deep learning and have a large population with balanced characteristics, but we restricted the maximal number of clones to avoid overfitting the models trained on this database.

⁷ <https://www.r-bloggers.com/generating-synthetic-data-sets-with-synthpop-in-r/>

⁸ https://www.unece.org/fileadmin/DAM/stats/documents/ece/ces/ge.46/20150/Paper_24_bnowok_synthpop.pdf

⁹ Difference between the number of observed and synthetic data in percent

Once the database is defined, we introduced efficient methods for data filtering based on Spark, for cluster-computing with implicit data parallelism¹⁰.

Supplementary 2

We use the weight distributions w_{ij} from equation 1 to assess the parameter classification in the different network layers.

The weights between the input parameters V_i and the first layer H_i are the single parameters (besides the output layer) that can be trained in this model; in Figure S2-1 we present the values obtained after 150 epochs as a heat map, where the x axis represents the mean weight of the link of each of the input parameter (from V1 to V10) on the first trainable layer (from H1 to H10; see model architecture in figure 3).

Figure S2-1 classification of the model weights w_{ij} in the first trainable input layer after 150 epochs

This information is useful to establish the hierarchy of the input parameters in the NN model and presents a clear interpretation as to which parameters are more relevant for the trained model: both parameters creatinine Phosphokinase and serum creatinine are particularly relevant for the final prediction of therapy outcome and therapy duration. Serum Creatinine Test and Creatinine Phosphokinase are a waste product from the normal breakdown of muscle tissue. As creatinine is produced, it's filtered through the kidneys and excreted in urine and is an indicator of the normal function of the kidney as well as its capacity to expel metabolites generated during a patient's treatment. Therefore, the current result suggests that the metabolization of the substances provided in a therapy is relevant for the estimation of the success of the therapy outcome. Furthermore, the observation of the creatinine is an important indicator of how good the patient's compliance and adherence to the therapy is, which is naturally directly correlated to the therapy's outcome¹¹.

Therefore, the analysis of the parameter distribution in the first layer is from a medical perspective meaningful and can be used as a proxy for a health professional.

The analysis of the model classification in the next layer is less meaningful from a medical perspective, since it is a random definition that remains frozen in all the training process. Essentially, this layer groups relations between the classified input parameters H_i differently and remains frozen during all the training process. This explains why there is no convergence, i.e it preserves its initial random configuration during the training process (figure S2-2).

¹⁰ https://en.wikipedia.org/wiki/Apache_Spark

¹¹ <https://academic.oup.com/jat/article/40/8/659/2445890>

Figure S2 -2 Distribution of model features M_i encoded in the second -non trainable- layer in the LONN model (figure 3)

The further classification of M_i , which is basically a feature extraction based on the parameter hierarchy, as is shown in figure S2-2, is then performed with the continuous logic multicriteria operators (figure 3). This means, features extracted from the initial parameter hierarchy are evaluated using the continuous logic operators implemented with our LONN methodology.

Figures

Figure 1

Structure of the implemented recommender system

Figure 2

Workflow for the generation of synthetic patients. From this data, we analyze the parameter distribution and generate entirely new synthetic parameters using these distributions, which meant that we could generate synthetic patient populations completely and that this data provides more anonymization of the original clinical data.

Figure 3

For now, with simplicity in mind, we implement two AND layers (conjunctions), followed by an OR layer (disjunction), to logically evaluate the nodes M_i , which is a process modeling human reasoning in the decision process.

A.

B.

Figure 4

Representation of natural thinking processes (A) and by neural networks (B). Logical processes, like the combination of "and" and "or" processes, are implemented as fuzzy logic, representing natural uncertainties in the thinking process.

Figure 5

Squashing function $S_\beta(x)$ for different parameters β .

Figure 6

Training error of the model presented in Figure 3 for different β_{inp} (different shapes) and β_{int} (x-axis) estimated after 50 epochs for each validation run.

Figure 7

Internal model validation with a dense NN (two internal layers, with 10 and 4 neurons with ReLU activation function, figure A), vs. the logic-operators NN (LONN, figure B) for 150 epochs.

A.

B.

Figure 8

Test of two LONNs with two different topologies with different numbers of internal layers and logical operators.

Figure 9

Training method for the recommender system. We train two different models based on the whole data set (Model 1) and a dataset consisting only of positive outcomes (Model 2). After that, we use both models to make different predictions. If there is a matching in the predictions, then Prediction 1 is used as the standard with high confidence; otherwise, predictions from Model 2 are provided but have low confidence.

Patients		Recommendations based on patient survival to therapy			
				✓ Validation in both control and reference groups	
				⚠ Validation only in control group	
		Show 10 entries			
		Search:			
Alternative Therapy		Therapy		Confidence	Survival_Prob
Pat. Nr 23425		Betablocker	⊙	*** ✓	Prob. Survival < 1 Year
Pat. Nr 12564		Betablocker	⊙	*** ✓	Prob. Survival < 1 Year
Pat. Nr 12256	Betablocker	ACE	⊙	* ⚠	Prob. Survival > 1 Year
Pat. Nr 82382		ACE	⊙	* ⚠	Prob. Survival > 1 Year
Pat. Nr 23124		Betablocker	⊙	*** ✓	Prob. Survival > 1 Year
Pat. Nr 63346		ARB	⊙	* ⚠	Prob. Survival > 1 Year
Pat. Nr 12643		Aspirin	⊙	*** ✓	Prob. Survival > 1 Year
Pat. Nr 23464	ACE	ARB	⊙	* ⚠	Prob. Survival > 1 Year

👍 Thanks! You are improving the life quality of our patients

Figure 10

Diagram of a potential app based on the model outcomes.