Explainable AI

Jee Dong Jun

11.10.2020

Early Warning Scoring Systems

- Quick way to access degree of illness of patients.
- Increase in score or high score produces escalated response.

National Early Warning Score (NEWS)

PHYSIOLOGICAL PARAMETERS	3	2	1	0	1	2	3
Respiration Rate	≤8		9 - 11	12 - 20		21 - 24	≥25
Oxygen Saturations	≤91	92 - 93	94 - 95	≥96			
Any Supplemental Oxygen		Yes		No			
Temperature	≤35.0		35.1 - 36.0	36.1 - 38.0	38.1 - 39.0	≥39.1	
Systolic BP	≤90	91 - 100	101 - 110	111 - 219			≥220
Heart Rate	≤40		41 - 50	51 - 90	91 - 110	111 - 130	≥131
Consciousness Level				A			V, P, or U

* a weighting score of 2 should be added for any patient requiring supplemental oxygen

Figure 1: early warning score

Transparency and Explainability

- Prediction is not the only important factor in clinical practice.
- Wrong prediction can have serious consequences.
- Therefore, clinicians must understand the reasoning behind predictions.
- Must produce comprehensible reasons for prediction.

Current System

- When patients are admitted, various health conditions are checked in regular time interval.
- Based on this data, EWS score is calculated.
- When clinicians observe either a high EWS or an increase in EWS, appropriate actions are taken.
- Targeted clinical interventions happen when the clinician understands which parameters have caused high EWS.

Patients show symptoms beforehand

- EWS is based on assumption that clinical deterioration can be seen by various physiological changes or large change in one variable.
- Therefore, time-series data is used for prediction.
- Standard deep learning models are black-box predictions that cannot be used in explanation.

RNN model

- RNN model with gated recurrent units was proposed for mortality prediction.
- Self-attention was used to highlight particular time steps of the time series that is most important in model's prediction.
- Variation of RNN with attention model was used for interpretable framework.

Convolution Neural Network

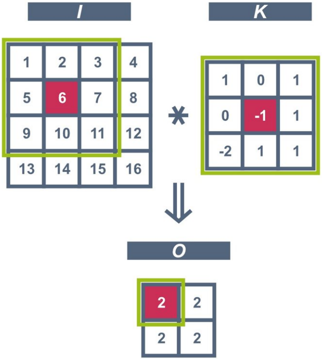


Figure 2: 2d convolution

- Capture spatial dependency

1D Convolution

- Just like 2D convolution captures spatial dependency, 1D convolution captures time dependency.
- Instead of 2 dimension kernel, one dimension kernel is used which runs through time axis.
- Input x is of the form (n, t, m) where n represents sample number, t is time step and m is feature value compared to (n, x, y, c) for 2D convolution.

1D vs 2D

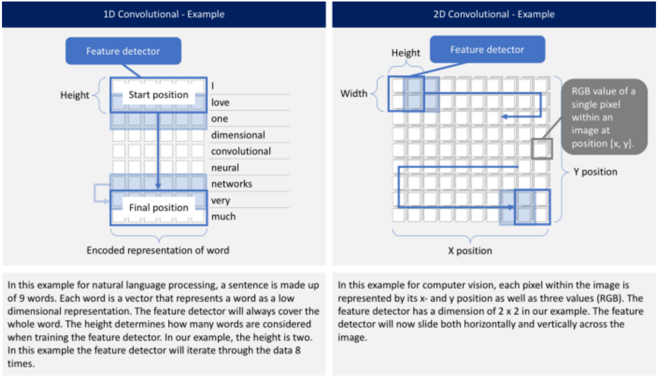


Figure 3: 1dvs2d

- Note all features are summed with weight.
- weight $W \in R^{dxF_l}$ where d is kernel size and F_l is input feature dimension

Causal Convolution

- Main difference of TCN from 1D CNN is that convolution is causal.
- Convolution filter at time t only depends on the inputs that are no later than t.
- $F(s) = x *_f = \sum_i^{k-1} f(i) x_{s-i}$ where k is kernel size.
- By using zero padding to ensure that length is conserved, TCN can take any length sequence and output same length sequence. (just like RNN)

Dilated Convolution

- For RNN, hidden state at time t is only depend on t-1. Long-term interactions tend to be exponentially smaller.
- TCN solves this issue by using dilated convolution.
- By having exponential stride, dilated convolution enables larger receptive field compared to normal CNN with lesser parameters.
- At same time, long-term interactions are still captured.

Dilated Convolution

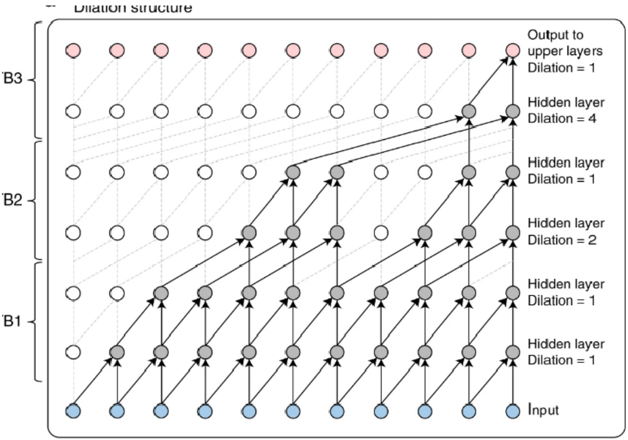


Figure 4: dilation

- $F(s) = x * f = \sum_i^{k-1} f(i)x_{s-di}$ where k is kernel size and d is dilation size.

AI-ews

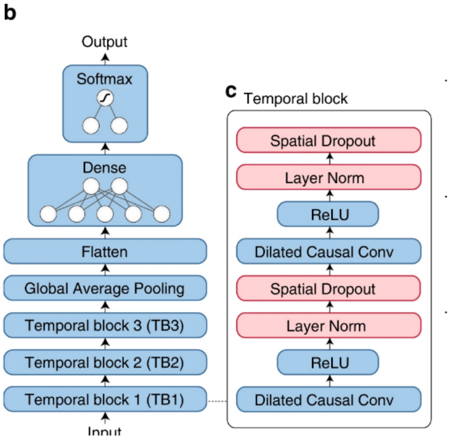


Figure 5: Structure of AI-EWS

Structure of AI-EWS

- In each temporal block, there are 2 convolution layer. Dilation is increased between each temporal block but kept constant within block.
- By stacking temporal block, the receptive field increase exponentially.
- $1 + \sum_i (k-1)(2^{i-1}+1)$
- Global Average Pooling across time step in final layer.

Summary

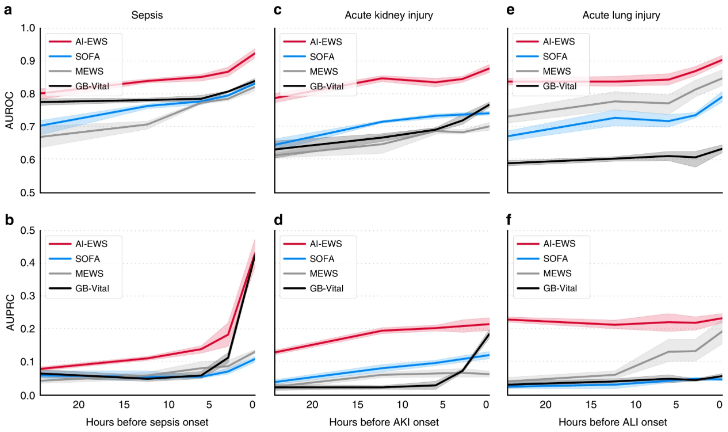
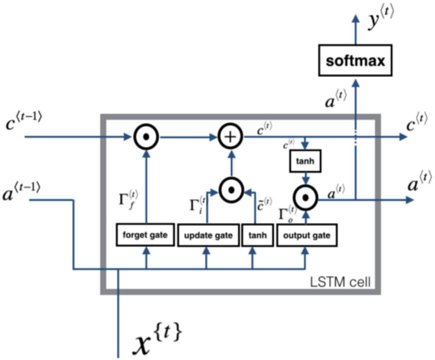


Figure 6: Prediction

RNN LSTM model



$\cdots$

$$\Gamma_f^{\langle t \rangle} = \sigma(W_f[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_f)$$
$$\Gamma_u^{\langle t \rangle} = \sigma(W_u[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_u)$$
$$\tilde{c}^{\langle t \rangle} = \tanh(W_C[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_C)$$
$$c^{\langle t \rangle} = \Gamma_f^{\langle t \rangle} \circ c^{\langle t-1 \rangle} + \Gamma_u^{\langle t \rangle} \circ \tilde{c}^{\langle t \rangle}$$
$$\Gamma_o^{\langle t \rangle} = \sigma(W_o[a^{\langle t-1 \rangle}, x^{\langle t \rangle}] + b_o)$$
$$a^{\langle t \rangle} = \Gamma_o^{\langle t \rangle} \circ \tanh(c^{\langle t \rangle})$$

Figure 7: LSTM

- GRU structure is similar to LSTM both difficult to train.

why TCN better?

- In RNN, the latent state at each time step, t, is only a function of the data at t and hidden state and memory at t-1.
- TCN captures past events by convolution. Hence, can capture variant length sequence.
- To model long term interaction, we just have to adjust kernel size and layer number. (receptive size increase exponentially)

Good points of TCN

- Unlike RNN which is sequential, TCN is not sequential enabling parallel evaluation.
- Back-propagation direction is different from the time axis, hence avoid exploding/vanishing gradient problem.
- There are many ways to adjust receptive field size. (model design more flexible)
- No need to store complicated result of multiple cell gate during backpropgation.
- Overall, training is faster than RNN and result tends to be better.

Slight Problem

- Depending on data, receptive field may have to be adjusted.
- Data storage during evaluation may be longer.
- LSTM learns how much to forget but TCN has to decide beforehand.

Understanding the prediction

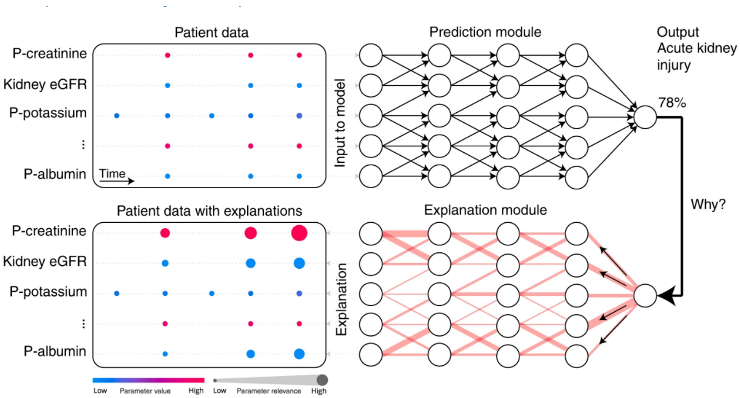


Figure 8: overview

Sensitivity vs Decomposition

- What makes human human?
- What makes more human?
- Sensitivity is related to decomposition.
- But what we are interested is decomposition.

Decomposition

- Decomposition and sensitivity are almost same in linear regression.
- $f(x) = w^T x = \sum w_i x_i$
- Decomposition of output $f(x)$ is $\sum_i [x_i]$ and each decomposition must be positive.

Use Backpropagation Idea

- Start from the output $f(x)$
- Decomposes an explanation into simpler local updates, defining the contribution to the explanation of each activating neurons in the previous layer.
- $R_{i \leftarrow j}$ represents local relevance updates connecting neuron i to neuron j
- R_i represents local relevance of neuron i which can be calculated by $\sum_j R_{i \leftarrow j}$
- Continue until we reach input layer.

How to choose update rule?

- Between each layer, update rule can be different.
- One possible way is $R_{i \leftarrow j} = \frac{w_{ij} a_i}{\sum_i w_{ij} a_i} R_j$ where a is activation and w is weight.
- Network is only consist of ReLU activations and linear projections without a bias term.
- Therefore, by changing w_{ij} with w_{ij}^{+}, we can enforce all relevance score to be positive.

Explanation result

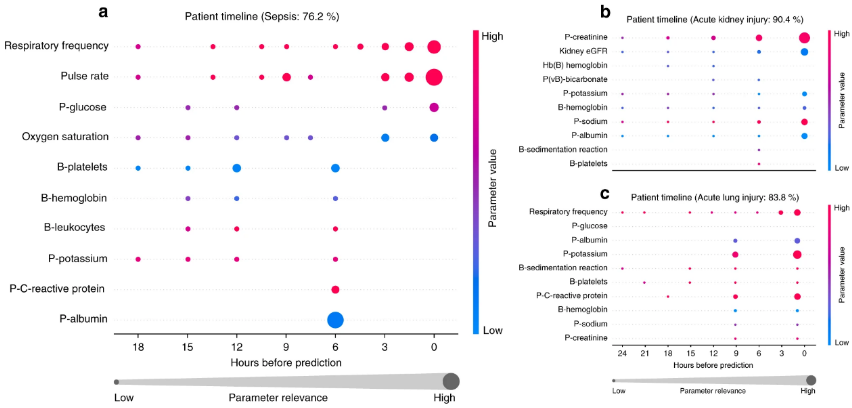


Figure 9: Explanation Result in the paper