**XNAP: Making LSTM-based Next Activity Predictions Explainable by Using LRP**

<https://arxiv.org/pdf/2008.07993.pdf>

**0. DEFINITIONS**

|  |
| --- |
| **Definition 1 (Vector, Matrix, Tensor)**  **Vector :** 숫자 값들의 배열로, **i번째 값은** 이다.  **Matrix :** 숫자 값들의 2차원 배열로, 이다.  **Tensor T**는 숫자 값들의 n차원 배열이다. 예를 들어 n=3이면 이고, 이때 이다.  **Definition 2 (Event, Trace, Event Log)**  **Event**는 **(c, a, t) 형태의 tuple**이며, 여기서 c는 case id, a는 activity (event type)이고 t는 timestamp이다.  **Trace**는 event의 비어 있지 않은 sequence로, 와 같은 형태로 표현하며, 이때 **한 trace 내부의 모든 event에 대하여 그 case id가 같다.** 즉 이다.  **Event log L**은 trace의 집합으로, 과 같이 표현한다.   * Trace는 vector의 sequence라고 할 수 있다.   **Definition 3 (Prefix and Label)**  **Prefix**는 trace의 부분 sequence로, 가 주어졌을 때 그것의 length k인 prefix는 이다.  **Label**은 prefix에 대해 정의되는 것으로, **length k인 prefix에 대한 label**은 이다. |

**1. Layer-wise Relevance Propagation for LSTMs**

|  |
| --- |
| **LRP (Layer-wise Relevance Propagation)**은 Deep Neural Network의 예측에 대해 설명하기 위한 기술로, 다음과 같이 작동한다.   * 주어진 **입력 sequence** 에 대해, 학습된 DNN 모델 를 이용하여 **출력 을 계산**한다. * LRP는 출력 **를 모델 에 reverse-propagate**하고, 각 의 input variable에 **relevance value를 할당**한다. * 여기서 relevance value는 **입력 변수가 예측에 얼마나 기여했는지**를 나타낸다. |
| **NEXT PAGE** |

|  |
| --- |
| **PREVIOUS PAGE** |
| LSTM에서는 신경망의 연결이 다음 2가지 경우 중 하나이므로, 해당 경우로 한정한다.   * **Many-to-one** weighted linear connections * **Two-to-one** multiplicative interactions   **<Many-to-one weighted connections>**   |  | | --- | | 가 lower-layer neuron이고 가 upper-layer neuron일 때, forward pass에서의 이 값은 다음과 같이 계산된다.  Upper-layer neuron 에 대한 각 relevance 가 주어졌을 때, **LRP는 lower-layer neuron 에 대한 각 relevance 를 계산**한다.   * 처음에는 로 설정되어 있다. * **Lower-layer로의 Relevance distribution**은 다음의 두 단계로 구성된다.   + 처음에 upper-layer neuron 로부터 lower-layer neuron 로의 **relevance message 를 계산**한다. 이때 다음 수식을 이용한다.   + 그 다음에, 각 lower-layer neuron 에 대해 들어오는 message를 합산하여 **relevance score 를 계산**한다. |   **<Two-to-one multiplicative interactions between lower-layer neurons>**   |  | | --- | | Upper-layer neuron 에 대해서 forward pass에서의 값은 **2개의 lower-layer neuron의 값 와 의 곱**으로 계산된다. 즉 이다.   * 이와 같은 multiplicative interaction에서, 2개의 lower-layer neuron 중 **value range가 [0, 1]인, gate를 나타내는 하나가 sigmoid activation function**을 사용한다.   + 이 neuron을 라 하고, 다른 하나를 라고 한다. * 여기서 upper-layer neuron 에 대한 relevance를 라고 하면 relevance는 다음과 같이 lower-layer neuron으로 재분배된다.   + 이 규칙을 통해서 gate neuron은 forward pass에서 **source neuron에 포함되어 있는 information이 얼마나 많이 남아 있어야 하는지를 결정**한다. | |

**2. XNAP: Explainable Next Activity Prediction**

|  |  |
| --- | --- |
| XNAP (Explainable Next Activity Prediction)은 **offline component와 online component로 구성**된다.   * **Offline component:** 예측 모델은 **과거의 event log로부터 Bi-LSTM DNN**을 적용하여 학습한다. * **Online component:** 학습된 모델은 **다음 행동에 대한 예측**에 쓰인다. * 주어진 next activity prediction 모델과 learned predictive model에 대하여, LRP는 **각 activity의 relevance value를 결정**한다.   **<Offline Component: Learning a Bi-LSTM model>**   |  | | --- | | Offline component는 **event log를 받아서 pre-processing**한 다음 이것을 이용하여 학습한 **Bi-LSTM model**을 반환한다.  **<Pre-processing>**  Event log L을 data tensor X와 label matrix Y (next activity)로 바꾼다.   * 1. **Event log L을 matrix 로 변환**한다. 여기서 E는 event log의 size이고 U는 event tuple의 원소의 개수이다.   + 이때 예측을 위해 각 sequence의 끝부분에 activity를 추가한다. * 2. Bi-LSTM의 입력은 숫자 값이어야 하므로, **S에 있는 activity attribute의 string value들을 one-hot encoding**한다. * 3. **행렬** 를 얻는다. 이때 H는 event log에 있는 서로 다른 activity value의 개수이다.   + 그렇게 함으로써 **prefix가 R인 tuple**이 함수 를 적용하여, 그리고 **prefix가 K인 tuple**이 함수 을 적용하여 행렬 S로부터 만들어진다. * 4. 마지막으로, **prefix tuple R**에 기반하여 3차원 데이터 텐서 를, **label tuple K**에 기반하여 **label matrix** 를 생성한다.   **<Model Learning>**  XNAP은 Bi-LSTM model M을 이용하여 학습하는데, 이 모델은 **전처리 단계에서의 data tensor X와 label matrix Y**에 기반하여 **prefix들을 next activity label들로 mapping**한다.   * 여기서는 Bi-LSTM 구조, 즉 vanilla LSTM 구조의 확장을 이용하는데, 이것은 Bi-LSTM이 **control-flow information을 양방향의 sequence로부터 이용**할 수 있는 **forward and backward LSTM**이기 때문이다.   XNAP의 Bi-LSTM 구조는 input layer, hidden layer와 output layer로 구성되어 있다.   * Hidden layer는 **100차원의 Bi-LSTM layer**이다. * Overfitting을 방지하기 위해서 **20%의 random dropout**을 적용한다. | |
| **NEXT PAGE** |

|  |
| --- |
| **PREVIOUS PAGE** |
| **<Online Component: Producing predictions with explanations>**   |  | | --- | | Online component는 input을 running trace로 받아서 pre-processing을 수행한 후 next activity에 대한 예측을 생성한다. 또한 prediction을 고려하여, 각 running trace의 activity에 대해 relevance value를 생성하여 결론을 내린다.   * Prediction은 **offline component로부터의 학습된 Bi-LSTM model을 사용하여 생성**된다. * Prediction이 주어졌을 때, LRP는 **학습된 Bi-LSTM model을 backward passing하여 activity relevance를 결정**한다.   **<Pre-processing>**  Online component의 pre-processing step은 **running trace 를 data tensor와 label matrix로 변환**한다.   * 이때 **이면 online phase를 중지**하며, 그 이유는 base prediction와 relevance creation을 위한 데이터가 부족하기 때문이다. * 더 나아가서, 우리는 offline component에서 **longest trace를 포함한 모든 가능한 activity를 관측했다**고 가정한다.   **<Prediction creation>**  이전 step에서 주어진 data tensor 로부터, offline component로부터의 학습된 Bi-LSTM model M은 **모든 activity에 대한 확률 값을 포함한 확률분포 를 반환**한다.   * 이때 확률분포 로부터 를 이용하여 예측 를 찾는다.   **<Relevance creation>**  마지막으로, **LRP를 적용하여 예측 p의 explainability를 제공**한다.   * Next activity prediction p에 대해서, LRP는 **running trace 의 course에 있는 각 activity에 대해 relevance value를 결정**한다.   + Model을 decompose하여 toward하고, 모델을 따라 output layer로부터 input layer까지 backward한다. * Prediction p는 prediction creation 단계에서, **running trace 의 모든 activity**를 고려하여 생성되었다. * 이때, Arras et al이 제안한, **LSTM을 위해 설계된 LRP 접근 방법**을 사용한다.   Relevance value를 heatmap으로 시각화하기 위하여 다음과 같이 rescale한다.   * Positive relevance values: range [0.5, 1.0] * Negative relevance values: range [0.0, 0.5] | |

**3. Results**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Event log는 다음과 같이 구성된다.    실험 설계는 다음과 같다.   |  |  | | --- | --- | | Predictive quality | **Model generalization을 향상**시키기 위해서 **각 event log의 trace를 랜덤하게 shuffle한다.**   * 이렇게 하면 event log entry에 대한 process-instance-affiliation을 위한, **process-instance에 기반한 sampling**을 할 수 있다. | | Explainability | XNAP의 LAP의 explainability를 설명하기 위하여, **F1-Score의 값이 가장 큰 Bi-LSTM model**을 선택한다. |   실험 결과는 다음과 같다. |