Paper review: RuleNN

Eszter Iklódi

TU Wien

Mar 24, 2022
Learning Explainable Linguistic Expressions with Neural Inductive Logic Programming for Sentence Classification

Prithviraj Sen  
IBM Research  
San Jose, CA, USA  
senp@us.ibm.com

Marina Danilevsky  
IBM Research  
San Jose, CA, USA  
mdanile@us.ibm.com

Yunyao Li  
IBM Research  
San Jose, CA, USA  
yunyaoli@us.ibm.com

Siddhartha Brahma  
Google Research  
Mountain View, CA, USA  
sidbrahma@gmail.com

Matthias Boehm  
Graz University of Technology  
Graz, Austria  
m.boehm@tugraz.at

Laura Chiticariu  
IBM Watson  
San Jose, CA, USA  
chiti@us.ibm.com

Rajasekar Krishnamurthy  
IBM Watson  
San Jose, CA, USA  
rajase@us.ibm.com
Brief summary

**Topic**
transparent learning

**Task**
sentence classification

**Model**
first-order logic rules

**Method**
neural network

**Results**
outperforms statistical and neuro-symbolic methods, comparable to RNN

**Implication**
explainable and editable models
Context

**Background**
- Black-box models: difficult-to-interpret, undesirable bias
- There is interest in interpretability.
- Attempts to explain
  - surrogate models
  - neural network layer activation (attention)

**Knowledge gap**
- Do surrogate models correctly reflect the process?
- Treat explainability as first-class citizen and not as an after-thought

**Research question**
- Is it possible to devise a neural network that directly learns a model expressed in a clear, human-readable dialect?
Suggested approach

Architecture

- Dictionaries
  - Sentences → NLU
  - Facts and predicates
- RuleNN
  - LEs
- Verify
  - LEs

Idea

- Model NLP with first-order logic (FOL)
- Use shallow semantic parsing
- Combine the two into **linguistic expression (LE)**
Linguistic Expression (LE) - Example

Shallow semantic parsing

Notices may be transmitted electronically, by registered mail.

Learned rule = Linguistic expression

\[
\text{communication}(s) \leftarrow \text{Contains}(s, a) \land a.A1 = \text{notice} \\
\land (a.\text{verb} = \text{inform} \lor a.\text{verb} = \text{transmit})
\]
High level overview

Inputs

- Dict1
  - inform
  - notify
  - transmit

- Dict2
  - notice
  - communication

Hand-crafted dictionaries

Sentences

- Notice may be transmitted electronically. by registered mail

NLU

- Shallow semantic parsing
  - A1 transmit.01 Argm
    - notice
    - (transmit)
    - (electronically)

  - register.02 A0
    - (register)
    - (registered mail)

- Facts and predicates
  - Contains(S₁, transmit.01)
  - Contains(S₁, register.02)
  - MatchesDict(Dict₁, transmit.01.verb)
  - MatchesDict(Dict₂, transmit.01.A₁)

- Predicate matrix
  - transmit.01 1 1
  - register.02 0 0

RuleNN

- Parametrized predicate
  \[ PP(P; \alpha) = \sum_{P \in P} \alpha_i P_i(a), \ \forall a \in R_i, \ \forall x \in D \]
  such that \( \sum_i \alpha_i = 1 \), \( \alpha_i \geq 0 \ \forall i = 1, \ldots |P| \)

- Neural network
  - Concat
  - \( \exists v_{i+1} \wedge \sum_i \alpha_i \theta_i \)
  - Concat

- Predicate Generation Module
  - Clause Generation Module

FOL retrieval algorithm

**Algorithm 1: Post-hoc LE retrieval**

```plaintext
input: Learned \( \alpha_1, \ldots, \alpha_m \) and training data \( D \).
output: List of LEs.

1. \( S \leftarrow \{\} \)  // Loop goes over \( \binom{|P|}{m} \) combinations
2. while more predicate combinations exist do
   3. \( (p_1, \ldots, p_m) \leftarrow \) get next predicate combination
   4. if \( \prod_{i=1}^{m} \alpha_i p_i > 0 \) \( \exists x \in D \) such that \( (p_1, \ldots, p_m) \sim x \)
      then \( S \leftarrow S \cup \{(p_1, \ldots, p_m)\} \)
5. return \( S \)
```
Shallow semantic parsing

Syntactic structures - Penn Treebank (Marcinkiewicz, 1994)

(S (NP I)
  (VP have
    (NP a dog)
    (PP from
      (NP a Hungarian shelter))))

Semantic structures - PropBank (Palmer et al., 2005)

- Proposition Bank
- Penn Treebank structures + semantic role labels
- Shallow: no higher-order phenomena, e.g. coreference
- Predicate-argument information
  - ArgNr: numbered arguments; can be mapped to any theory
  - ArgM: adjunct-like arguments, like location, cause, time etc.
Frameset **decline.01** “go down incrementally”

Arg1: entity going down

Arg2: amount gone down by, EXT

Arg3: start point

Arg4: end point

Ex: … [\(\text{Arg}_1\) its net income] **declining** [\(\text{Arg}_2\cdot\text{EXT} 42\%\)] [\(\text{Arg}_4\) to $121$ million] [\(\text{Arg}_M\cdot\text{TMP} \) in the first 9 months of 1989]. (wsj_0067)

Frameset **decline.02** “demure, reject”

Arg0: agent

Arg1: rejected thing

Ex: [\(\text{Arg}_0\) A spokesman] **declined** [\(\text{Arg}_1\) *trace*\(_i\) to elaborate] (wsj_0038)
First-order logic (FOL)

First order logic // Predicate logic // Quantificational logic

Definition

- ∃ existential quantification
- ∀ universal quantification
- ∧ conjunction
- ∨ disjunction
- ¬ negation

How to learn?

- already done: inductive logic programming (ILP), statistical learning (StarAI), neuro-symbolic AI
- knowledge gap: none of these target NLP so far
Rule a.k.a. LE

Rule $R_3$:

\[
\text{communication}(s) \iff \text{Contains}(s, a) \\
\land (a.A0 \text{ contains } \text{notice}) \\
\lor (a.A0 \text{ contains } \text{communication}) \\
\land a.\text{tense} = \text{future}
\]

Distinguished predicate for LE

Semantic predicate

Syntactic predicate

\[
S_3 : \text{Notices required in writing under this agreement will be made to the appropriate contact(s) . . .}
\]
**Linguistic Expression (LE)**

### Logical Predicate
- Boolean-valued function returning true or false.
- Atom: a predicate whose variables denote constants.
- Fact: an atom that holds true.

### SRL (semantic role labeling) Predicate
- True, if the given dictionary contains the action’s surface form corresponding to a specific semantic attribute.
- Example semantic attributes: *verb*, *A0*, *A1*, *ARGM*

### Syntactic Predicate
- True if action’s value is equal to a specific value of a syntactic attribute.
- Example syntactic attributes: *tense*, *voice*
Linguistic Expression (LE)

Rule
- Head predicate: label
- Body predicate: conditions

LE
- A clause over a sentence and action
- distinguished Contains predicate + SRL and syntactic predicates
$R_3$ : \begin{align*}
\text{communication}(s) & \leftarrow \text{Contains}(s, a) \\
& \wedge (a.A0 \text{ contains } \text{notice}) \\
& \vee (a.A0 \text{ contains } \text{communication}) \\
& \wedge a.\text{tense} = \text{future} 
\end{align*}

$S_3 : \text{Notices required in writing under this agreement will be made to the appropriate contact(s) }$
Task 1: TREC - question classification

**Labels**

```
ABBREVIATION
abb
exp
```

**Examples**

ABBR:exp What is the full form of .com ?
HUM:title What is the oldest profession ?
DESC:def What are liver enzymes ?

**Label distribution**

<table>
<thead>
<tr>
<th>Label</th>
<th>Skew</th>
<th></th>
<th></th>
<th>Label</th>
</tr>
</thead>
<tbody>
<tr>
<td>LOC</td>
<td>0.18</td>
<td>133</td>
<td></td>
<td>Location</td>
</tr>
<tr>
<td>HUM</td>
<td>0.28</td>
<td>109</td>
<td></td>
<td>Human</td>
</tr>
<tr>
<td>NUM</td>
<td>0.17</td>
<td>127</td>
<td></td>
<td>Numeric</td>
</tr>
<tr>
<td>ENTY</td>
<td>0.22</td>
<td>137</td>
<td></td>
<td>Entity</td>
</tr>
<tr>
<td>DESC</td>
<td>0.22</td>
<td>122</td>
<td></td>
<td>Description</td>
</tr>
<tr>
<td>ABBR</td>
<td>0.02</td>
<td>38</td>
<td></td>
<td>Abbreviation</td>
</tr>
</tbody>
</table>
Task 2: Contracts

Task

- IBM proprietary data
- Multi-label task, but treated as binary classification
- Sentences from legal contracts among enterprise
- Training set: IBM first-party
- Test set: diverse companies
- Generalization?

| Label | Skew | \(|P|\) |
|-------|------|--------|
| W     | 0.09 | 101    |
| SoW   | 0.07 | 48     |
| DR    | 0.06 | 80     |
| IP    | 0.05 | 79     |
| C     | 0.06 | 39     |
| P&T   | 0.10 | 117    |
| T&T   | 0.08 | 77     |
| P&B   | 0.05 | 95     |
| L     | 0.04 | 71     |

Labels

- Payment Terms & Billing: Elements that detail how and when a party is to pay or get paid, as well as the items or fees the parties are paying or billed for. Includes references to modes of payment or payment mechanisms.
- Pricing & Taxes: Elements that refer to specific amounts or figures that are associated with individual deliverables that are exchanged (for example, how much something costs) as part of satisfying the terms of the contract. Includes references to specific figures or methods for calculating prices or tax amounts.
RuleNN architecture

Steps to take
1. Learn discriminative predicates, a.k.a. FOL rules
2. Combine them into LEs
3. Learn multiple LEs

Architecture

RuleNN
1. PGM: Predicate Generation Module
2. CGM: Clause Generation Module
input: sentences and dictionaries

dictionaries: hand-crafted for each label

output: generated facts and predicates

Figure 3: Generating (c) facts and predicates from (a) shallow semantic parsing and (b) dictionaries.
Task formulation: Input representation

Predicate matrix

\[
\begin{array}{cc}
\text{transmit.01} & 1 & 1 \\
\text{register.02} & 0 & 0 \\
\end{array}
\]

\[M \in \{0, 1\}^{\left|R_i\right| \times \left|\mathcal{P}\right|}\]

Figure 4: Example predicate matrix where rows and columns denote actions and predicates, respectively.
Task formulation: Learnable parameters

Parametrized predicate (PP)

- linear (convex) combination of predicates
- one hot encoding $\rightarrow$ corner of the convex hull
- update: backpropagation

$$PP_\mathcal{P}(a; \alpha) = \sum_{P_i \in \mathcal{P}} \alpha_i P_i(a), \quad \forall a \in R_i, \quad x_i \in \mathcal{D}$$

such that $\sum_i \alpha_i = 1, \quad \alpha_i \geq 0 \quad \forall i = 1, \ldots |\mathcal{P}|$
### Method: Predicate Generation Module (PGM)

This label contains all together 4 predicates.

This sentence has 3 actions.

<table>
<thead>
<tr>
<th>Name</th>
<th>Global/Local</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$</td>
<td>Local</td>
<td>Predicate matrix for instance</td>
</tr>
<tr>
<td>$\mathcal{R}$</td>
<td>Local</td>
<td>Actions in instance</td>
</tr>
<tr>
<td>$PP_{ji}^j$</td>
<td>Local</td>
<td>Responses for actions belonging to instance</td>
</tr>
<tr>
<td>$\alpha_{ij}^j$</td>
<td>Global</td>
<td>Attention weights defining a learned predicate</td>
</tr>
<tr>
<td>$\gamma_{ij}^j$</td>
<td>Global</td>
<td>Log attention weights for learned predicate</td>
</tr>
<tr>
<td>$k$</td>
<td>Global</td>
<td>Number of LEs (hyperparameter)</td>
</tr>
<tr>
<td>$m$</td>
<td>Global</td>
<td>Length of LEs (hyperparameter)</td>
</tr>
</tbody>
</table>
Method: Clause Generation Module (CGM)

- Conjunction is replaced by differentiable, element-wise product t-norm.
- Combining different LEs: max across all CGM outputs. It can then be compared to the label of the sentence.

<table>
<thead>
<tr>
<th>Name</th>
<th>Global/Local</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$</td>
<td>Local</td>
<td>Predicate matrix for instance</td>
</tr>
<tr>
<td>$\mathcal{R}$</td>
<td>Local</td>
<td>Actions in instance</td>
</tr>
<tr>
<td>$PP_i^j$</td>
<td>Local</td>
<td>Responses for actions belonging to instance</td>
</tr>
<tr>
<td>$\alpha_i^j$</td>
<td>Global</td>
<td>Attention weights defining a learned predicate</td>
</tr>
<tr>
<td>$\gamma_i^j$</td>
<td>Global</td>
<td>Log attention weights for learned predicate</td>
</tr>
<tr>
<td>$k$</td>
<td>Global</td>
<td>Number of LEs (hyperparameter)</td>
</tr>
<tr>
<td>$m$</td>
<td>Global</td>
<td>Length of LEs (hyperparameter)</td>
</tr>
</tbody>
</table>
Figure 7: RuleNN for learning $k$ $m$-length clauses.
Method: Retrieve LEs expressed as FOL

1. Consider each m-combination of predicates.
2. Return LE if:
   - associated weight is non-zero
   - evaluates true on some sentences

---

Algorithm 1: Post-hoc LE retrieval

```plaintext
input : Learned $\alpha_1, \ldots, \alpha_m$ and training data $\mathcal{D}$.
output : List of LEs.

1. $S \leftarrow \{\} \quad // \text{ Loop goes over } |\mathcal{P}| \choose m \text{ combinations }
2. while more predicate combinations exist do
3.   $(p_1, \ldots, p_m) \leftarrow \text{ get next predicate combination }
4.   if $\prod_{i=1}^{m} \alpha_{ip_i} > 0 \land \exists x \in \mathcal{D} \text{ such that } (p_1, \ldots, p_m) \sim x$
5.     then $S \leftarrow S \cup \{(p_1, \ldots, p_m)\}
6. return S
```
Method: Algorithmic details

General
- RuleNN learns $k$ LEs containing *up to* $m$ PPs each.
- skewed data: negative sampling
- dropout before max-pooling
- exponential in $m$, but efficient for small $m$

Setup
- $k=50$, $m=4$
- dropout=0.5, batchsize=64, stepsize=0.01
- SGD with momentum=0.9
### Comparison: Overview

<table>
<thead>
<tr>
<th>Method</th>
<th>Category</th>
<th>Explainability</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>MG</td>
<td>inductive logic programming</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MG&lt;sub&gt;NT&lt;/sub&gt;</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSM</td>
<td>Markov logic network</td>
<td>white-box</td>
<td>predicate-based</td>
</tr>
<tr>
<td>BSRL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MITI</td>
<td>multiple instance learning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MIRI</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>NeralLP</td>
<td>neuro-symbolic AI</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RuleNN</strong></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>MINet</td>
<td>deep neural network</td>
<td>black-box</td>
<td></td>
</tr>
<tr>
<td>BiLSTM</td>
<td></td>
<td></td>
<td>token-based</td>
</tr>
</tbody>
</table>

Eszter Iklódi (TU Wien)  Paper review: RuleNN  Mar 24, 2022 27/43
Comparison: Metric

AUC-PR (area under the precision-recall curve)

- average of precision scores calculated for each recall threshold
- useful for imbalanced data
## Background: Inductive logic programming (ILP)

### Inductive logic programming - Muggleton (1991)
- **Input:** background knowledge (B) and examples (E)
- **Output:** logic program that entail ALL positive and NONE of the negative examples

### Used implementations

**Metagol (MG) - Cropper and Muggleton (2015)**
- top-down approach (generates rules before testing them on data)
- generates only 0-error rules

**Noise-tolerant metagol (MG_{NT}) - Muggleton et al. (2018)**
- minimized error instead of 0-error rules,
- more suited for noisy real-world data
Background: Markov logic network (MLN)

Markov logic network - Richardson and Domingos (2006)
- Markov-network: like Bayesian-network, but undirected and may be cyclic
- combines first order logic and probabilistic graphical methods
- each formula/clause has an attached weight
- A highly expressive representation!

Used implementations
LSM (Learning using Structural Motifs) - Kok and Domingos (2010)
- first, search for clauses and then learn weights
BSRL (Boost Statistical Relational Learning) - Khot et al. (2011)
- learns weights and the structure of the MLN simultaneously
Background: Multiple instance learning (MIL)

Multiple instance learning - Dietterich et al. (1997)

- dataset: bag of instances
- classification: for each bag, whether it contains at least one positive instance or not
- sentences are bags of actions
- label is assigned if there exists at least one action, for which the learned predicates hold true. RuleNN is also MIL

Used implementations

MITI (multi-instance tree inducer) - Blockeel et al. (2005)
- learns an instance (not bag) tree classifier with best-first node expansion strategy

MIRI (multi-instance rule induction) - Bjerring and Frank (2011)
- output is more naturally a set of classification rules (if-then rules)
Background: Neuro-symbolic AI

Neuro-symbolic AI
- traditional rules-based AI approaches with modern deep learning techniques

Used implementations
NeuralIP - Yang et al. (2017)
- problem: learning of probabilistic first-order logical rules
- end-to-end parameter and structure learning
Background: Black-box methods

Used implementations

MINet - Wang et al. (2018)

- deep neural network with fully connected layers
- used for multiple instance learning

BiLSTM

- replaces tokens with GloVe embeddings
- bi-directional LSTM
- label: aggregation of hidden LSTM layers
- hidden layer units: 200–600
- an order of magnitude larger parameter set than RuleNN
Comparison: Experiments summary

<table>
<thead>
<tr>
<th></th>
<th>MG</th>
<th>MG&lt;sub&gt;NT&lt;/sub&gt;</th>
<th>MITI&lt;sup&gt;·&lt;/sup&gt;</th>
<th>MIRI&lt;sup&gt;·&lt;/sup&gt;</th>
<th>MINet&lt;sup&gt;·&lt;/sup&gt;</th>
<th>LSM&lt;sup&gt;·&lt;/sup&gt;</th>
<th>BSRL&lt;sup&gt;·&lt;/sup&gt;</th>
<th>NeuralLP&lt;sup&gt;·&lt;/sup&gt;</th>
<th>RuleNN&lt;sup&gt;·&lt;/sup&gt;</th>
<th>BiLSTM&lt;sup&gt;·&lt;/sup&gt;</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>NR 0.07</td>
<td>0.184</td>
<td>0.156</td>
<td>0.294</td>
<td>—</td>
<td>0.183</td>
<td>0.537</td>
<td>0.685</td>
<td>0.805 ± 0.010</td>
<td></td>
</tr>
<tr>
<td>SoW</td>
<td>NR 0.011</td>
<td>0.011</td>
<td>0.018</td>
<td>—</td>
<td>0.015</td>
<td>0.438</td>
<td>0.658</td>
<td>0.689 ± 0.030</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DR</td>
<td>NR 0.05</td>
<td>0.144</td>
<td>0.147</td>
<td>0.258</td>
<td>—</td>
<td>0.021</td>
<td>0.614</td>
<td>0.848</td>
<td>0.807 ± 0.030</td>
<td></td>
</tr>
<tr>
<td>IP</td>
<td>NR 0.13</td>
<td>0.145</td>
<td>0.153</td>
<td>0.244</td>
<td>—</td>
<td>0.148</td>
<td>0.550</td>
<td>0.844</td>
<td>0.787 ± 0.050</td>
<td></td>
</tr>
<tr>
<td>C</td>
<td>NR 0.38</td>
<td>0.157</td>
<td>0.149</td>
<td>0.580</td>
<td>—</td>
<td>0.545</td>
<td>0.574</td>
<td>0.788</td>
<td>0.653 ± 0.020</td>
<td></td>
</tr>
<tr>
<td>P&amp;T</td>
<td>NR 0.30</td>
<td>0.111</td>
<td>0.083</td>
<td>0.314</td>
<td>—</td>
<td>0.269</td>
<td>0.516</td>
<td>0.813</td>
<td>0.802 ± 0.030</td>
<td></td>
</tr>
<tr>
<td>T&amp;T</td>
<td>NR 0.11</td>
<td>0.406</td>
<td>0.372</td>
<td>0.586</td>
<td>—</td>
<td>0.591</td>
<td>0.560</td>
<td>0.837</td>
<td>0.846 ± 0.020</td>
<td></td>
</tr>
<tr>
<td>P&amp;B</td>
<td>NR 0.10</td>
<td>0.111</td>
<td>0.122</td>
<td>0.154</td>
<td>—</td>
<td>0.192</td>
<td>0.533</td>
<td>0.819</td>
<td>0.786 ± 0.010</td>
<td></td>
</tr>
<tr>
<td>L</td>
<td>NR 0.07</td>
<td>0.115</td>
<td>0.126</td>
<td>0.12</td>
<td>—</td>
<td>0.205</td>
<td>0.464</td>
<td>0.750</td>
<td>0.741 ± 0.070</td>
<td></td>
</tr>
<tr>
<td>LOC</td>
<td>NR 0.710</td>
<td>0.699</td>
<td>0.833</td>
<td>0.473</td>
<td>0.835</td>
<td>0.470</td>
<td>0.904</td>
<td>0.998 ± 0.001</td>
<td></td>
<td></td>
</tr>
<tr>
<td>HUM</td>
<td>NR 0.36</td>
<td>0.771</td>
<td>0.770</td>
<td>0.922</td>
<td>0.565</td>
<td>0.927</td>
<td>0.558</td>
<td>0.912</td>
<td>0.999 ± 0.000</td>
<td></td>
</tr>
<tr>
<td>NUM</td>
<td>NR 0.687</td>
<td>0.680</td>
<td>0.821</td>
<td>0.497</td>
<td>0.756</td>
<td>0.497</td>
<td>0.856</td>
<td>0.996 ± 0.004</td>
<td></td>
<td></td>
</tr>
<tr>
<td>ENTY</td>
<td>NR 0.365</td>
<td>0.373</td>
<td>0.591</td>
<td>0.481</td>
<td>0.425</td>
<td>0.576</td>
<td>0.745</td>
<td>0.957 ± 0.020</td>
<td></td>
<td></td>
</tr>
<tr>
<td>DESC</td>
<td>NR 0.52</td>
<td>0.331</td>
<td>0.334</td>
<td>0.540</td>
<td>0.498</td>
<td>0.519</td>
<td>0.437</td>
<td>0.789</td>
<td>0.995 ± 0.003</td>
<td></td>
</tr>
<tr>
<td>ABBR</td>
<td>NR 0.731</td>
<td>0.731</td>
<td>0.688</td>
<td>0.542</td>
<td>0.735</td>
<td>0.443</td>
<td>0.774</td>
<td>1.000 ± 0.000</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(c) NR and — denote no LEs learned and non-convergence, resp. **Bold-font and underscore denotes best performing approach and best predicate-based method**, resp. RuleNN (ours) is the best predicate-based method. Variation of BiLSTM’s AUC-PR due to changing hidden dimensions is shown after ±.
Experiment: Explainability

No objective measure for LE’s explainability :(

GUI
- filtering and ranking LEs based on precision and recall
- dropping/adding predicates
- instant re-evaluation
Experiment: Human-in-the-loop learning

Imitate iterative and collaborative development

- 188 LEs learned for C (Communication label in Contracts)
- 4 data scientists with knowledge of FOL and NLU
- in half an hour time 6–8 LE selection
- → with human expertise LEs can be made smaller and more interpretable
- 4 explainable models of each subset of 3 participants
Learned rule

\[ R_3 : \text{communication}(s) \leftarrow \text{Contains}(s, a) \]
\[ \land (a.\text{A0} \text{ contains } \text{notice}) \]
\[ \lor a.\text{A0} \text{ contains } \text{communication} \]
\[ \land a.\text{tense} = \text{future} \]

\[ S_3 : \text{Notices required in writing under this agreement will be made to the appropriate contact(s)} .. \]
Experiment: Human-in-the-loop learning

Edited rule

$R_3 : \text{communication}(s) \leftarrow \text{Contains}(s, a)$

$\land (a.A0 \text{ contains } \textit{notice})$

$\lor a.A0 \text{ contains } \textit{communication}$

$\land a.tense = \textit{future}$

$S_3 : \textit{Notices} \text{ required in writing under this agreement will be made to the appropriate contact(s)} \ldots$
Discussion

Context
- A proposed neuro-symbolic learning method for sentence classification.
- Compared to similar methods: better efficiency and quality.

Key findings
- Yes, it is possible to learn human-interpretable models by designing NN-s with explainability in mind.

Strength and limitations
- It can be used for any MIL tasks assuming the predicates are given.
- It can learn rules combining any previously built classifier’s output probabilities.

Disclaimer
- approach is different from explainable AI (!)

What’s next?
- Learn rules on top of embeddings
- Learn rules and dictionaries jointly
Relevance to my research

- BRISE data is highly structured → great potential for rule learning
- Current rules are defined as graph or text patterns.
- Potential experiment: change UD/4lang semantic representation to a shallow semantic predicate-argument representation?
- Challenges: PropBank for German?
  https://github.com/System-T/UniversalPropositions
...still unclear

- How dictionaries are constructed? (For TREC: "automatically capture surface forms that discriminate well among labels")
- How is the semantic/syntactic parser implemented?
- How are the facts and predicates constructed, given the dictionaries and shallow-semantic notations?
- How can we get an OR-relation within one SRL predicate? (I guess it's just a syntactic sugar for "contains" in case the dictionary has more than one element.)
- Why is it good to have multiple CGMs \((k > 1)\)? The actual number of LEs is coming out as a result of the post-hoc algorithm.
- Generally, the term LE is a bit overloaded.
References


Thank you for your attention :)