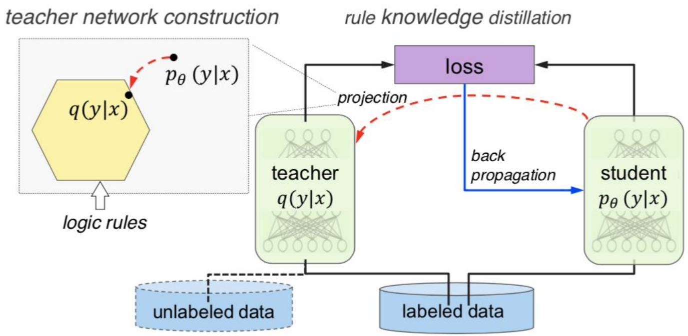
Harnessing Deep Neural Networks with Logic Rules



Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, Eric P. Xing
ACL 2016
Presenter: Weiyu Du

Motivation

- Hard to encode human intention in Deep Neural Nets
- But…
  - People not only learn from concrete examples, but also from general knowledge
  - Logic rules is an expressive language for that
- Therefore
  - We wish to enhance Neural Nets with logic rule knowledge
  - E.g. learn sentiment from sentence examples, but also follow the rule “A-but-B = B”
- Our framework uses iterative rule knowledge distillation procedure to learn from labeled data and logic rules simultaneously

Background

- Denote data as $\mathbf{D = \{(x_n, y_n)\}}$ where $\mathbf{x} \in \mathbf{X}$ is input and $\mathbf{y} \in \mathbf{Y}$ is target
- Denote first-order logic(FOL) rules as $\mathbf{R = \{(R_l, \lambda_l)\}}$ where $\mathbf{R}$ is the rule over space (X, Y) and $\boldsymbol{\lambda} \in \mathbf{[0, \infty]}$ is confidence level
  - FOL: extension to propositional logic, which can only express facts (either true or false)

Constant	1, 2, A, John, Mumbai, cat,....
Variables	x, y, z, a, b,....
Predicates	Brother, Father, >,....
Function	sqrt, LeftLegOf, ...
Connectives	∧, ∨, ¬, ⇒, ⇔
Equality	==
Quantifier	∀, ∃

FOL syntax

1. All birds fly.

In this question the predicate is "**fly(bird)**."

And since there are all birds who fly so it will be represented as follows.

$$\forall x\ \mathrm{bird}(x) \rightarrow \mathrm{fly}(x).$$

2. Every man respects his parent.

In this question, the predicate is "**respect(x, y)," where x=man, and y= parent**.

Since there is every man so will use ∀, and it will be represented as follows:

$$\forall x\ \mathrm{man}(x) \rightarrow \mathrm{respects}\ (x, \mathrm{parent}).$$

FOL examples

Background

- FOL rules: **$R = \{(R_l, \lambda_l)\}$** , **R** is the rule, **$\lambda \in [0, \infty]$** is confidence level
  - Grounding: logic expression with all variables instantiated
  - $\lambda = \infty$ indicates hard rule, all groundings have to be true
  - Denote the set of groundings of R_l as **$\{r_{lg}(X, Y)\}$**
- Encode FOL rules using soft logic
  - Soft logic are continuous from [0, 1], instead of {0, 1}
  - & vs Λ
    - & is selection operator
    - A&B = B when A = 1, A&B = A when A = 0
    - Λ is averaging operator

$$A \& B = \max\{A + B - 1, 0\}$$

$$A \vee B = \min\{A + B, 1\}$$

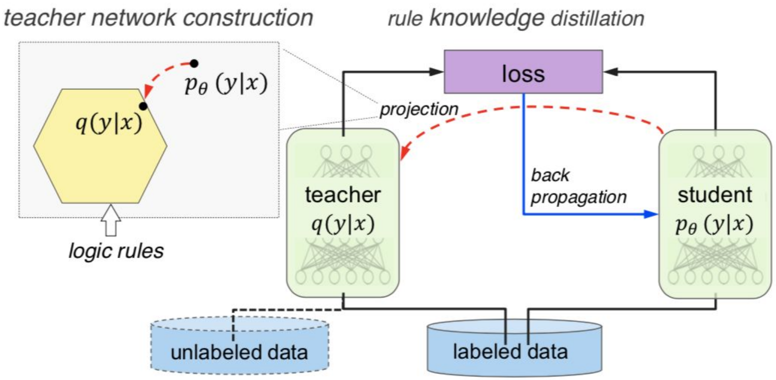
$$A_1 \wedge \cdots \wedge A_N = \sum_i A_i / N$$

$$\neg A = 1 - A$$

Iterative rule knowledge distillation

Consider K-way classification

- "Student network": Learn from labeled instances, defines conditional probability $p_\theta(y|x)$
- "Teacher network": Constructed by projecting $p_\theta(y|x)$ to a subspace constrained by FOL rules, denoted $q(y|x)$



Iterative rule knowledge distillation

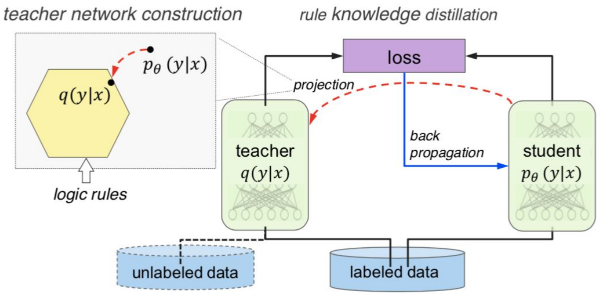
Algorithm 1 Harnessing NN with Rules

Input: The training data $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^N$,
The rule set $\mathcal{R} = \{(R_l, \lambda_l)\}_{l=1}^L$,
Parameters: π – imitation parameter
C – regularization strength

1: Initialize neural network parameter $\boldsymbol{\theta}$
2: **repeat**
3: Sample a minibatch $(\mathbf{X}, \mathbf{Y}) \subset \mathcal{D}$
4: Construct teacher network q with Eq.(4)
5: Transfer knowledge into p_θ by updating $\boldsymbol{\theta}$ with Eq.(2)
6: **until** convergence

Output: Distill student network p_θ and teacher network q

Overall Algorithm



Transfer knowledge into p_θ

- We wish to balance between imitating q(y|x) and learning supervised labels, therefore define objective:

Learn from labels

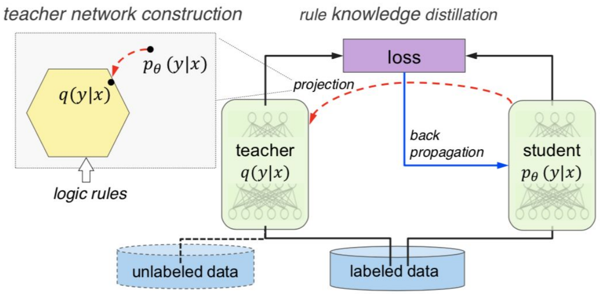
$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\theta \in \Theta} \frac{1}{N} \sum_{n=1}^{N} (1-\pi) \boxed{\ell(\mathbf{y}_n, \boxed{\boldsymbol{\sigma}_\theta(\mathbf{x}_n)})} + \pi \boxed{\ell(\boxed{\mathbf{s}_n^{(t)}}, \boxed{\boldsymbol{\sigma}_\theta(\mathbf{x}_n)})},$$

prediction from p_θ(y|x)

Imitation

prediction from q(y|x) at iteration t

- π: imitation parameter
- Teacher and student are learned simultaneously



Construct teacher network

- Goal:
  - 1) fits the rule
    - Impose rule constraints through expectation operator
    - For each rule, expect **$E_{q(Y|X)}[r_{lg}(X, Y)] = 1$** with confidence λ_l
  - 2) stay close to p_θ
    - Minimize KL-divergence between q and p_θ
- Combining above, we form the optimization problem

$$\min_{q,\boldsymbol{\xi}\geq 0} \mathrm{KL}(q(\mathbf{Y}|\mathbf{X}) \| p_\theta(\mathbf{Y}|\mathbf{X})) + C \sum\nolimits_{l,g_l} \xi_{l,g_l}$$

$$\text{s.t.} \quad \lambda_l (1 - \mathbb{E}_q[r_{l,g_l}(\mathbf{X}, \mathbf{Y})]) \leq \xi_{l,g_l}$$

$$g_l = 1, \ldots, G_l, \quad l = 1, \ldots, L,$$

Construct teacher network

- $\xi_{l,g_l} \geq 0$: slack variable for each rule, C: regularization parameter

$$\min_{q,\boldsymbol{\xi}\geq 0} \ \mathrm{KL}(q(\mathbf{Y}|\mathbf{X})\|p_\theta(\mathbf{Y}|\mathbf{X})) + C\sum\nolimits_{l,g_l} \xi_{l,g_l}$$
$$\text{s.t.} \ \ \lambda_l(1 - \mathbb{E}_q[r_{l,g_l}(\mathbf{X},\mathbf{Y})]) \leq \xi_{l,g_l}$$
$$g_l = 1,\ldots,G_l, \quad l = 1,\ldots,L,$$

- Problem is convex, can be efficiently solved in dual form with closed-form solutions

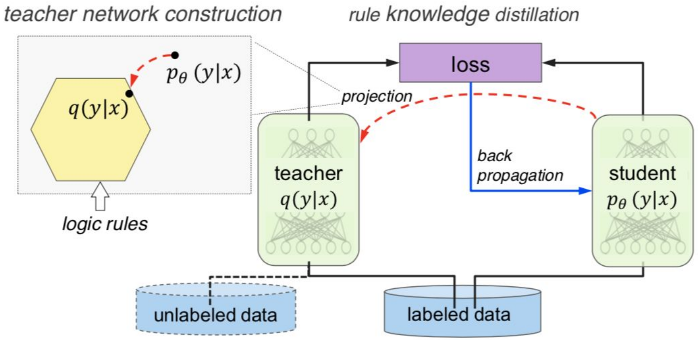
$$q^*(\mathbf{Y}|\mathbf{X}) \propto p_\theta(\mathbf{Y}|\mathbf{X}) \exp\left\{-\sum_{l,g_l} C\lambda_l(1 - r_{l,g_l}(\mathbf{X},\mathbf{Y}))\right\}$$

Algorithm 1 Harnessing NN with Rules

Input: The training data $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$,
The rule set $\mathcal{R} = \{(R_l, \lambda_l)\}_{l=1}^{L}$,
Parameters: π – imitation parameter
C – regularization strength

```
Initialize neural network parameter θ
repeat
    Sample a minibatch (X, Y) ⊂ D
    Construct teacher network q with Eq.(4)
    Transfer knowledge into p_θ by updating θ with Eq.(2)
until convergence
```

Output: Distill student network p_θ and teacher network q



$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\theta \in \Theta} \frac{1}{N} \sum_{n=1}^{N} (1-\pi)\ell(\mathbf{y}_n, \boldsymbol{\sigma}_\theta(\mathbf{x}_n)) + \pi \ell(\mathbf{s}_n^{(t)}, \boldsymbol{\sigma}_\theta(\mathbf{x}_n)),$$

Equation 2

- π : at beginning of training, p_θ prediction is bad, therefore we favor true labels. As training goes on, gradually bias towards emulating teacher

$$q^*(\mathbf{Y}|\mathbf{X}) \propto p_\theta(\mathbf{Y}|\mathbf{X}) \exp\left\{ -\sum_{l,g_l} C\lambda_l (1 - r_{l,g_l}(\mathbf{X}, \mathbf{Y})) \right\}$$

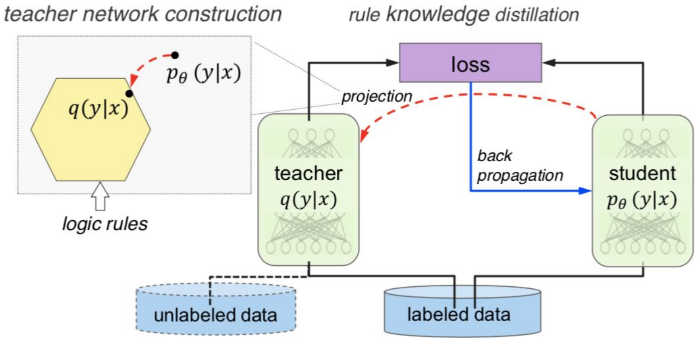
Equation 4

Algorithm 1 Harnessing NN with Rules

Input: The training data $\mathcal{D} = \{(\mathbf{x}_n, \mathbf{y}_n)\}_{n=1}^{N}$,
The rule set $\mathcal{R} = \{(R_l, \lambda_l)\}_{l=1}^{L}$,
Parameters: π – imitation parameter
C – regularization strength

Initialize neural network parameter $\boldsymbol{\theta}$
repeat
    Sample a minibatch $(\mathbf{X}, \mathbf{Y}) \subset \mathcal{D}$
    Construct teacher network q with Eq.(4)
    Transfer knowledge into p_θ by updating $\boldsymbol{\theta}$ with Eq.(2)
until convergence

Output: Distill student network p_θ and teacher network q



$$\boldsymbol{\theta}^{(t+1)} = \arg\min_{\theta \in \Theta} \frac{1}{N} \sum_{n=1}^{N} (1-\pi)\ell(\mathbf{y}_n, \boldsymbol{\sigma}_\theta(\mathbf{x}_n)) + \pi \ell(\mathbf{s}_n^{(t)}, \boldsymbol{\sigma}_\theta(\mathbf{x}_n)),$$

Equation 2

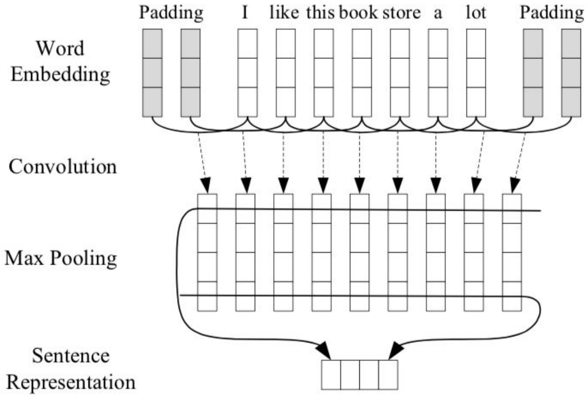
$$q^*(\mathbf{Y}|\mathbf{X}) \propto p_\theta(\mathbf{Y}|\mathbf{X}) \exp\left\{ -\sum_{l, g_l} C\lambda_l (1 - r_{l,g_l}(\mathbf{X}, \mathbf{Y})) \right\}$$

Equation 4

- student p vs teacher q at test time
    - We can use either p or q at test time
    - In general, q performs better than p
        - q more suitable when rules requires joint inference (spanning over multiple example)
        - p more lightweight and efficient

Sentence-level sentiment analysis

- Task: identify the sentiment (positive / negative) underlying individual sentence
- Base Network: single-channel conv net
  - Max-over-time pooling
  - Fully-connected layer after sentence representation
- Logic Rule:
  - Consider A-but-B structure, B dominates
  - "I'm stuck at home but I get to watch Friends."



$$\text{has-`A-but-B'-structure}(S) \Rightarrow$$
$$(\mathbf{1}(y=+) \Rightarrow \boldsymbol{\sigma}_\theta(B)_+ \ \wedge \ \boldsymbol{\sigma}_\theta(B)_+ \Rightarrow \mathbf{1}(y=+))$$

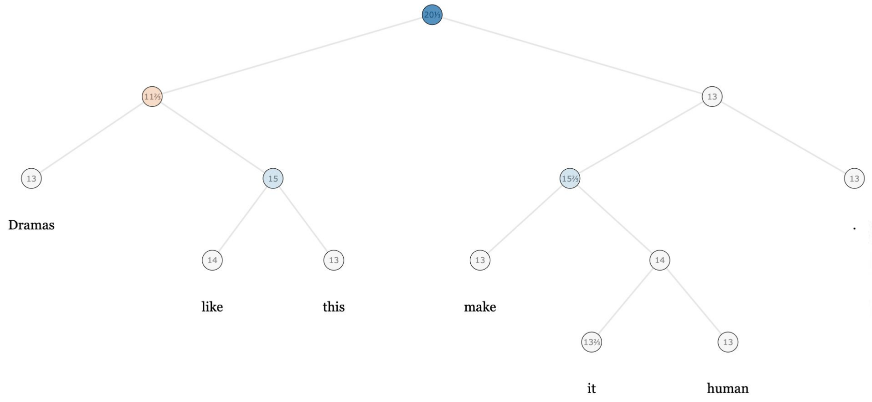
Truth value evaluates to $(1+\boldsymbol{\sigma}_\theta(B)_+)/2$ when $y=+$, and $(2-\boldsymbol{\sigma}_\theta(B)_+)/2$ otherwise

Sentiment analysis experiment

- Datasets: Around 15% sentences contains "but"
  - SST2 (Stanford Sentiment Treebank)
  - MR: movie reviews
  - CR: customer reviews of various products

All the more disquieting for its relatively gore-free allusions to the serial murders , but it falls down in its attempts to humanize its subject .

MR



"Dramas like this make it human."

SST2

```
[t]excellent phone , excellent service .
##i am a business user who heavily depend on mobile service .
phone[+3], work[+2]##there is much which has been said in other
reviews about the features of this phone , it is a great
phone , mine worked without any problems right out of the box .
```

CR

Sentiment analysis experiment

- Compare against different methods
  - Superior performance, q improves over p
  - On SST2, MVCNN has better performance -- diverse sets of pre-trained word embeddings, more layers and parameters

	Model	SST2	MR	CR
1	CNN (Kim, 2014)	87.2	81.3±0.1	84.3±0.2
2	CNN-Rule-*p*	88.8	81.6±0.1	85.0±0.3
3	CNN-Rule-*q*	89.3	**81.7±0.1**	**85.3±0.3**
4	MGNC-CNN (Zhang et al., 2016)	88.4	–	–
5	MVCNN (Yin and Schutze, 2015)	**89.4**	–	–
6	CNN-multichannel (Kim, 2014)	88.1	81.1	85.0
7	Paragraph-Vec (Le and Mikolov, 2014)	87.8	–	–
8	CRF-PR (Yang and Cardie, 2014)	–	–	82.7
9	RNTN (Socher et al., 2013)	85.4	–	–
10	G-Dropout (Wang and Manning, 2013)	–	79.0	82.1

Sentiment analysis experiment

- Compare against different rule integration methods on SST2
  - -but-clause: takes the clause after “but” as input
  - -l2-reg: adds regularization term $\gamma \|\boldsymbol{\sigma}_\theta(S) - \boldsymbol{\sigma}_\theta(Y)\|_2$
  - -project: project trained CNN to rule-constrained space
  - -opt-project: optimize projected CNN
  - -pipeline: distills pre-trained “opt-project” to plain CNN

	Model	Accuracy (%)
1	CNN (Kim, 2014)	87.2
2	-but-clause	87.3
3	-ℓ_2-reg	87.5
4	-project	87.9
5	-opt-project	88.3
6	-pipeline	87.9
7	-Rule-p	88.8
8	-Rule-q	**89.3**

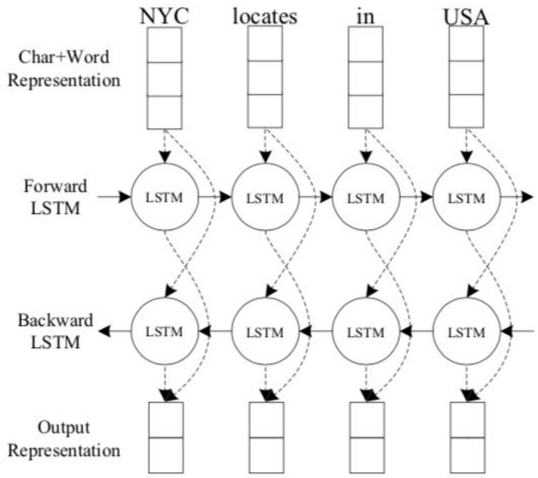
Sentiment analysis experiment

- Semi-supervised learning
  - Superior in performance sparse data context
  - Performance further improved with unlabeled data, they are used to better absorb logic rules

	Data size	5%	10%	30%	100%
1	CNN	79.9	81.6	83.6	87.2
2	-Rule-p	81.5	83.2	84.5	88.8
3	-Rule-q	82.5	83.9	85.6	**89.3**
4	-semi-PR	81.5	83.1	84.6	–
5	-semi-Rule-p	81.7	83.3	84.7	–
6	-semi-Rule-q	**82.7**	**84.2**	**85.7**	–

Named entity recognition

- Task: locate and classify elements in text into entity categories
  - Assign tag in "X-Y", where X is one of BIEOS (Beginning, Inside, End, Outside, Singleton) and Y is entity category
- Base Network: bi-directional LSTM
  - CNN + pre-trained word vectors for char+word repr
- Logic Rule:
  - Constraint on successive label for a valid tag sequence:
  - I-ORG (inside, organization) cannot follow B-PER (beginning
  - $\text{equal}(y_{i-1}, \text{I-ORG}) \Rightarrow \neg\, \text{equal}(y_i, \text{B-PER})$
  - List structures:
  - 1. Juventus, 2. Barcelona, … Barcelona has to be a club
  - $\text{is-counterpart}(X, A) \Rightarrow 1 - \|c(\mathbf{e}_y) - c(\boldsymbol{\sigma}_\theta(A))\|_2$



Named entity recognition

- Datasets: 1.7% named entities occur in lists
  - CoNLL-2003 NER benchmark [ORG U.N.] official [PER Ekeus] heads for [LOC Baghdad]
  - Close performance to SOTA, which is more complex and has more parameters

	Model	F1
1	BLSTM	89.55
2	BLSTM-Rule-trans	p: 89.80, q: 91.11
3	BLSTM-Rules	p: 89.93, q: **91.18**
4	NN-lex (Collobert et al., 2011)	89.59
5	S-LSTM (Lample et al., 2016)	90.33
6	BLSTM-lex (Chiu and Nichols, 2015)	90.77
7	BLSTM-CRF$_1$ (Lample et al., 2016)	90.94
8	Joint-NER-EL (Luo et al., 2015)	91.20
9	BLSTM-CRF$_2$ (Ma and Hovy, 2016)	**91.21**

BLSTM-Rule-trans: impose transition rule, BLSTM-Rules: further impose list rule

Discussion

- Summary:
  - Our framework combines learning knowledge and rules through an iterative distillation procedure. We transfer logic rules through a teacher network, constructed with posterior regularization principle.
- Contribution:
  - Provides a general distillation framework for FOL that can be applied to any specific network structures; very intuitive
- Limitations:
  - Dependent on hand-crafted rules as priors, lack the ability to induce and learn abstract knowledge from data; unsuitable to incorporate large amount of fuzzy human intuitions
- Comparison:
  - A Semantic Loss Function for Deep Learning with Symbolic Knowledge
    - Jingyi Xu, Zilu Zhang, Tal Friedman, Yitao Liang, Guy Van den Broeck
    - Combines propositional logic, limited but more convenient
  - Deep Neural Networks with Massive Learned Knowledge
    - Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, Eric P. Xing
    - Mutual distillation that iteratively transfers information between DNN and structured knowledge