# Harnessing Deep Neural Networks with Logic Rules

By Abhilasha

# Introduction

1. This method allows neural network to learn from both specific examples and general rules.

2. This approach can be seen as the combination of the knowledge distillation (Hinton et al., 2015; Bucilu et al., 2006) and the posterior regularization (PR) method (Ganchev et al., 2010)

# Distilling Knowledge

1. The main Idea of Knowledge distillation is to compress trained network.

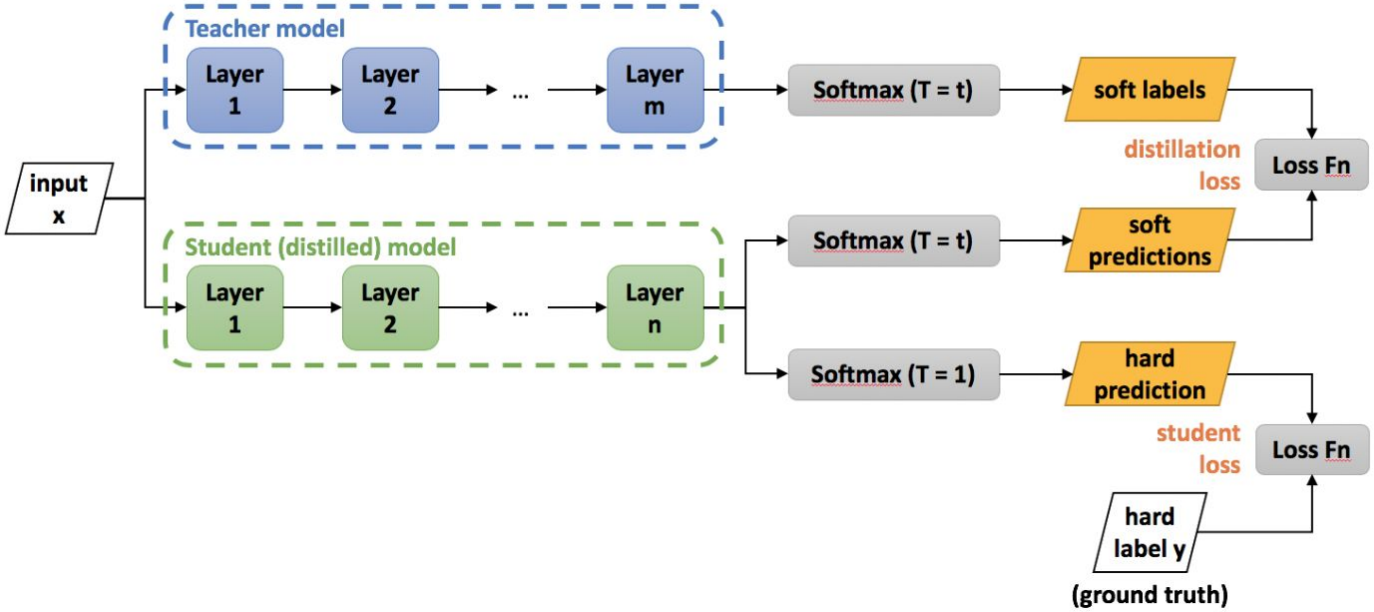
2. VGGNet requires 540 MB of storage which is not suitable for mobile devices. Idea is to compress this network once it has been trained(Teacher) and using this network another model(student) is trained to mimic the output of the large network(Teacher) instead of training it on raw data directly.

# Distilling Knowledge

$$P_t(a) = \frac{\exp(q_t(a)/\tau)}{\sum_{i=1}^{n} \exp(q_t(i)/\tau)},$$

Softmax with Temperature

**So how is this transfer of knowledge done?**



# Harnessing Deep Neural Networks with Logic Rules

1. This work integrates first order logic rules with neural networks to transfer  human intention and domain knowledge to neural models and regulate learning process.

2. This work present framework which enables neural network to learn simultaneously from labelled instances as well as logic rules through an iterative rule knowledge distillation procedure that transfer the structured information encoded in logic rules into network parameters.

# Algorithm

**Algorithm 1** Harnessing NN with Rules

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

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

**Output:** Distill student network $p_\theta$ and teacher network $q$

# Details

1. Training Data D = $(x_n, y_n)_{n=1}^{N}$
2. FOL R = $(R_l, \lambda_l)_{n=1}^{L}$ R_l is lth rule
3. Grounding is the logic expression with all variables being instantiated.
4. Given a set of examples (X,Y ) ⊂ (X , Y) (e.g., a minibatch from D), the set of groundings of Rl are denoted as {r_lg(X,Y )} g=1 to G_l
5. λ_l ∈ [0, ∞] is the confidence level with λ_l = ∞ indicating a hard rule, i.e., all groundings are required to be true (=1).

# Objective function to train the network

1. A neural network defines a conditional probability $p_\theta(y|x)$ by using a softmax output layer that produces a K-dimensional soft prediction vector denoted as $\sigma_\theta(x)$.
2. To integrate the information encoded in the rules, we propose to train the network to also imitate the outputs of a rule-regularized projection of $p_\theta(y|x)$, denoted as $q(y|x)$, which explicitly includes rule constraints as regularization terms.
3. The new objective is then formulated as a balancing between imitating the soft predictions of q and predicting the true hard labels:

# Objective function

1. l denotes the loss function selected according to specific applications (e.g., the cross entropy loss for classification).
2. s_n is the soft prediction vector of q on x_n at iteration t.
3. π is the imitation parameter calibrating the relative importance of the two objectives.

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

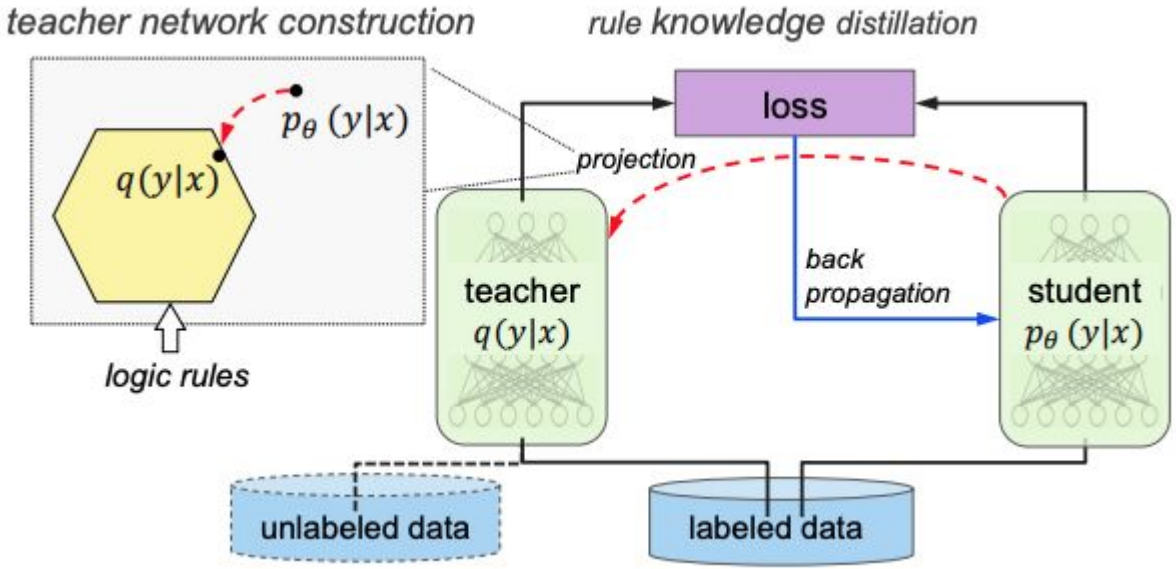
# Teacher Network Construction

1. This approach construct the teacher network q(y|x) at each iteration from p_θ(y|x).
2. Our goal is to find the optimal q that fits the rules while at the same time staying close to p_θ.
3. first property, we apply a commonly-used strategy that imposes the rule constraints on q through an expectation operator. That is, for each rule (indexed by l) and each of its groundings (indexed by g) on (X,Y ), we expect E_q(Y |X) [r_lg(X,Y )] = 1, with confidence λ_l.
4. For the second property, we measure the closeness between q and pθ with KL-divergence, and wish to minimize it.

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

# Parameters Initialization

1. Imitation parameters(π) : Since the teacher network is constructed from pθ, which, at the beginning of training, would produce low-quality predictions, we thus favor predicting the true labels more at initial stage. As training goes on, we gradually bias towards emulating the teacher predictions to effectively distill the structured knowledge. Specifically, we define $\pi(t) = \min\{\pi_0, 1 - \alpha t\}$ at iteration $t \geq 0$, where $\alpha \leq 1$ specifies the speed of decay and $\pi_0 < 1$ is a lower bound.


teacher network construction
rule knowledge distillation
$p_\theta(y|x)$
$q(y|x)$
logic rules
loss
projection
back
propagation
teacher
$q(y|x)$
student
$p_\theta(y|x)$
unlabeled data
labeled data