Learning beyond Datasets: Knowledge Graph Augmented Neural Networks for Natural Language Processing

Authors

- Annervaz K M, Indian Institute of Science & Accenture Technology Labs
- Somnath Basu Roy Chowdhury, IIT Kharagpur
- Ambedkar Dukkipati, Indian Institute of Science

Motivation

- Learning is still heavily on the specific training data
- Propose to enhance learning models with world knowledge
  - In the form of Knowledge Graph (KG) fact triples
- Develop a deep learning model that can extract relevant prior support facts from knowledge graphs
  - Depending on the task
  - Using attention mechanism

Example: Natural Language Inference

- *A: The couple is walking on the sea shore and*
- *B: The man and woman are wide awake*
- Need common knowledge: *"The man and woman and The couple means the same"*
- this information may not be specific to a particular inference

Example: Classify the News Snippet

- *"Donald Trump offered his condolences towards the hurricane victims and their families in Texas. "*
- Need to know the *facts <Donald Trump, president, United States>* and *<Texas, state, United States>.*
- Or we cannot classify it as a political news

Overview

- Extract relevant support facts on demand from a knowledge base
- Incorporate it in the feature space along with features learned from the training data
- Jointly model this look up mechanism along with the task specific training of the model
- Generic enough so that it can be augmented to any task specific learning model to boost the performance

Knowledge Graph Representations

- Structure-based embeddings
  - e.g. TransE (h + r = t)
- Semantically-enriched embeddings:
  - learns to represent entities/relations of the KG along with its semantic information.
  - NTN:
    - Initialize entity vectors with the average word embeddings followed by tensor-based operations.
  - DKRL:
    - Take into descriptive nature of text
    - Keep the simple structure of TransE model

- Conventional:

$$\max_\theta \; P(y|x,\theta)$$

- Augment the supervised learning process by incorporation of world knowledge feature x_w
- World knowledge features are retrieved using the data x:

$$x_w = F\left(x, \theta^{(2)}\right)$$

- Modified objective fuction:

$$\max_\theta \; P\left(y\middle|x, x_w, \theta^{(1)}\right)$$

- Where $\theta = \{\theta^{(1)}, \theta^{(2)}\}$
- Optimize: $\theta = \operatorname{argmax}_\theta \; P\left(y\middle|x, F\left(x, \theta^{(2)}\right), \theta^{(1)}\right)$

Vanilla Model

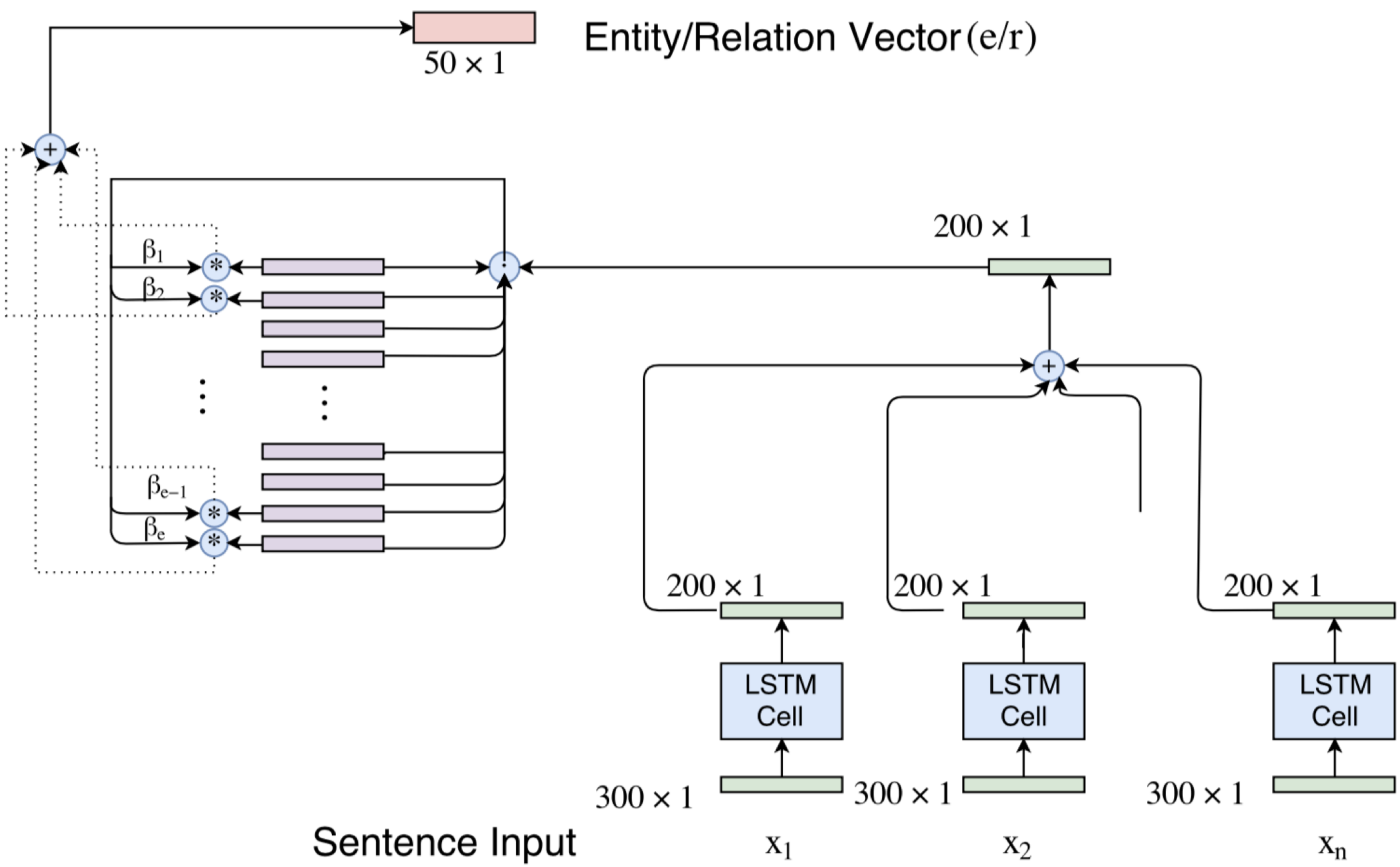

Fig. 2: Vanilla Entity/Relationship Retrieval Block Diagram

Vanilla Model - Encoder

- LSTM encoder:

$$h_t = f(x_t, h_{t-1})$$

$$O = \frac{1}{T}\sum_{t=1}^{T} h_t$$

- Context vector:

$$C = ReLU(O^T W)$$

- The same procedure is duplicated with separate LSTMs to form two separate context vectors
  - entity retrieval C_E
  - relationship retrieval C_R

Vanilla Model - Retriever

- Attention:

$$\alpha_{e_i} = \frac{\exp(C_E^T e_i)}{\sum_{j=0}^{|E|} \exp(C_E^T e_j)}, e = \sum_{j=0}^{|E|} \alpha_{e_i} e_i$$

$$\alpha_{r_i} = \frac{\exp(C_R^T e_i)}{\sum_{j=0}^{|R|} \exp(C_R^T e_j)}, r = \sum_{j=0}^{|R|} \alpha_{r_i} r_i$$

- DKRL uses the TransE model assumption (h + r ≈ t)
- Thus the fact triplet retrieved is F = [e, r, e + r]

Vanilla Model

- Concatenate context vector and fact triple
- $F' = ReLU(F^T V)$
- $y = softmax([F' : C]^T U)$
- V, U are parameters, y is used to compute the cross entropy loss

Vanilla Model - Problems

- Vanilla model attends over the entire entity/relation space
- Gradient for each attention value gets saturated easily as observed
- While training the classification and retrieval module together, the model tends to ignore the KG part and gradient propagates only through the classification module
- Most pertinent information for the task at hand comes from the training samples, only background aiding information comes from KG
- After few epochs of training, the KG retrieved fact always converged to a fixed vector

Pre-training KG Retrieval

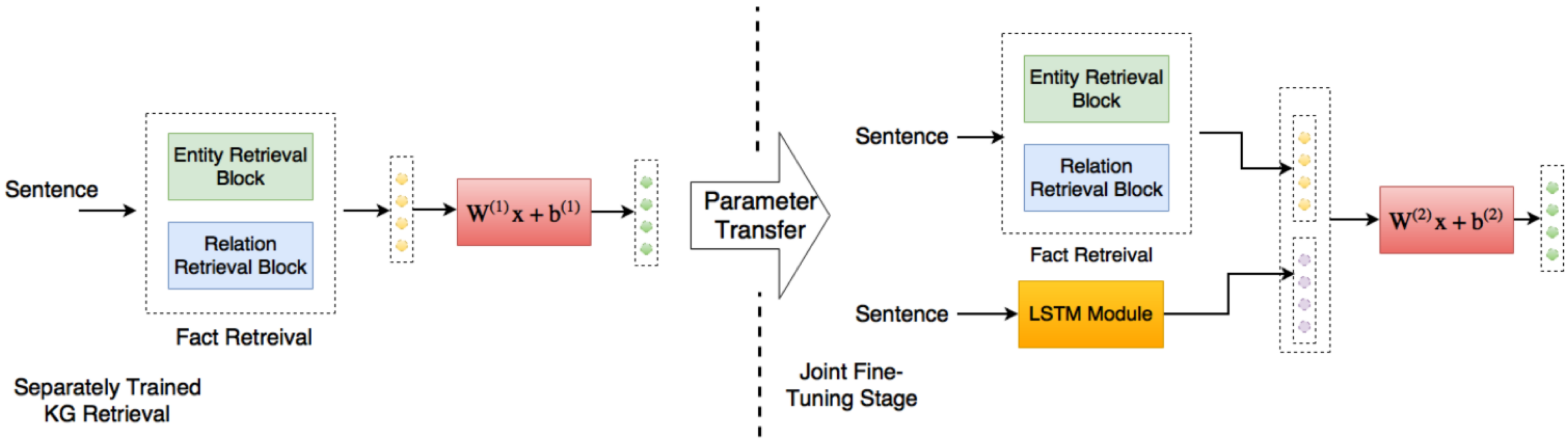

Fig. 3: Separately Training Knowledge Graph Retrieval and Jointly Training the Full Model

Pre-training KG Retrieval

- Pre-trained KG model is used to retrieve the facts
- Joint training:
  - Concatenate with the classification module
  - Allow error to be propagate through the pre-trained model
- The separate KG part alone shows significant performance
  - 59% for News20 and 66% for SNLI
  - KG doesn’t return noise and has essential information for the task

Convolution-based Cluster Representation

- Reduce the attention space
- Learning the representation of similar entity/relation vectors
- Use k-means clustering to form l clusters
  - With equal number of entity/relation vectors in each cluster
- Each clusters were then encoded using convolutional filters
  - 1D convolution

$$\mathcal{E}'(i,j) = W^T[e_{i,j}, e_{i+1,j}, \ldots, e_{i+k-1,j}]^T$$

  - Window size k
  - pooling layer

Convolution-based Cluster Representation

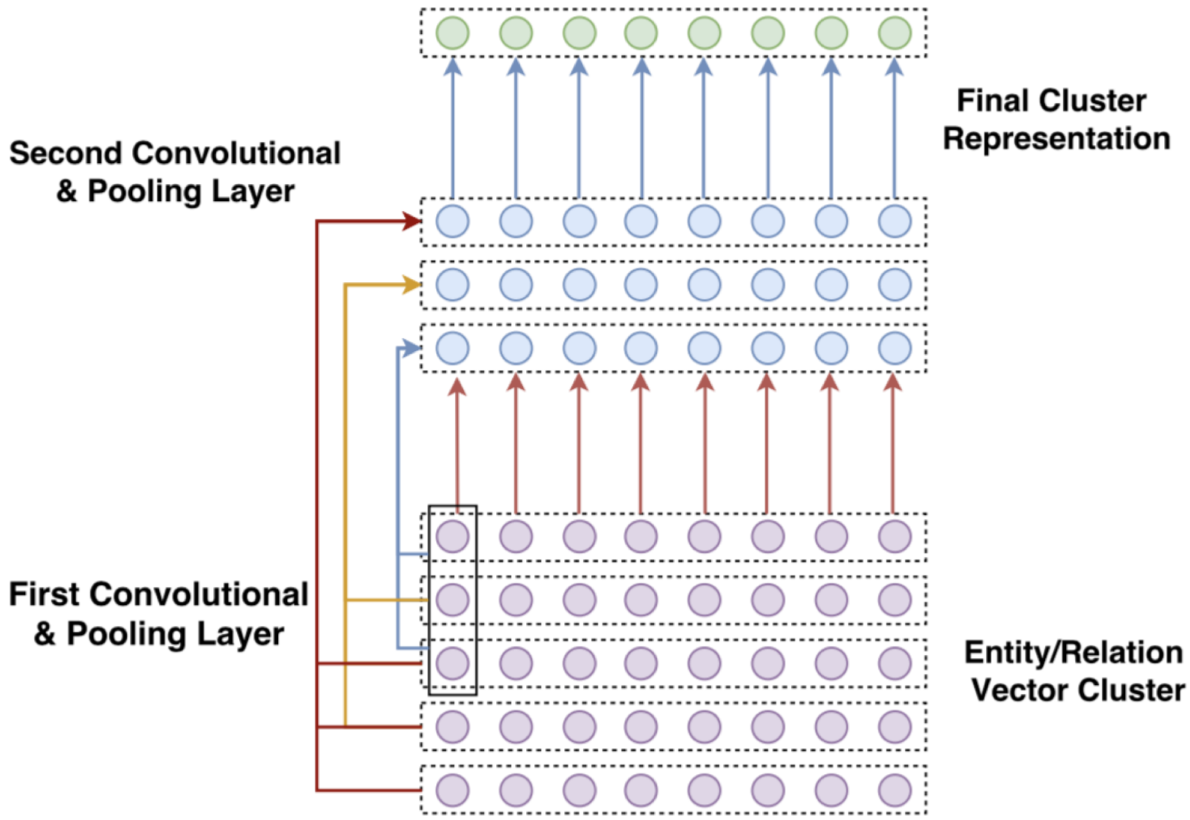

Fig. 4: Convolution model cluster representation

Experiments

Dataset	Train Size	Test Size	# Classes
News20	16000	2000	20
SNLI	549367	9824	3
DBPedia	553,000	70,000	14

Hyper-parameter	News20	SNLI
Batch size	256	1024
Learning rate	0.05	0.05
Word Vector Dimension	300	300
Sequence Length	300	85
LSTM hidden-state Dimension	200	200
KG Embedding Dimension	50	50
# Clusters	20	20
# Epochs	20	20

Experiments

Model	Accuracy	
	News20	SNLI
Plain LSTM	66.75%	68.73%
Vanilla KG Retrieval	67.30%	69.20%
Convolution-based KG	**69.34%**	**73.10%**

TABLE III: Test accuracy of approaches in News20 using FB15K & SNLI datasets using WN18

Summary

- Applicable for other domain task
- Future work:
  - Attention structure: flat – hierarchical attention
  - Soft attention – hard attention
  - Convolution based – similarity based

Thanks!