# Learning and Knowledge Transfer with Memory Networks for Machine Comprehension

Mohit Yadav. Lovekesh Vig. Gautam Shroff

TCS Research New-Delhi

*Presented by Kyo Kim*

April 24, 2018

# Overview

1. Motivation
2. Background
3. Proposed Method
4. Dataset and Experiment Results
5. Summary

# Motivation

# Problem

- Obtaining high performance in "machine comprehension" requires abundant human annotated dataset.
  - Measured by question answering performance.
- In a real-world dataset with small amount of data, wider range of vocabulary can be observed and the grammar structure is often complex.

# High-level Overview of Proposed Method

1. Curriculum based training procedure.
2. Knowledge transfer to increase the performance in dataset with less abundant labeled data.
3. Pre-trained memory network on small dataset.

# Background

# End-to-end Memory Networks

1. Vectorize the problem tuple.
2. Retrieve the corresponding memory attention vector.
3. Use the retrieved memory to answer the question.

# End-to-end Memory Networks Cont.

Vectorize the problem tuple

- Problem tuple: $(q, C, S, s)$
    - $q$: question
    - $C$: context text
    - $S$: set of answer choices
    - $s$: correct answer $(s \in S)$
- Question and context embedding matrix $A \in \mathbb{R}^{p \cdot d}$
    - Query vector: $\vec{q} = A\Phi(q)$
        - $\Phi$: Bag of words
    - Memory vector: $\vec{m}_i = A\Phi(c_i)$ for $i = 1, \cdots, n$ where $n = |C|$ and $c_i \in C$

# End-to-end Memory Networks Cont.

Retrieve the corresponding memory attention vector

- Attention distribution: $a_i = softmax(\vec{m}_i^\top \vec{q})$.
- Second memory vector: $\vec{r}_i = B\Phi(c_i)$ where $B$ is another embedding matrix similar to $A$.
- Aggregated vector: $\vec{r}_o = \sum_{i=1}^{n} a_i \vec{r}_i$
- Prediction vector: $\hat{a}_i = softmax((\vec{r}_o + \vec{q})^\top U\Phi(s_i))$
  - $U$ is the embedding matrix for the answers

# End-to-end Memory Networks Cont.

Answer the question

- Pick $s_i$ that corresponds to the highest $\hat{a}_i$.

## Cross-entropy Loss

$$L(P, D) = \frac{1}{N_D} \sum_{n=1}^{N_D} \Big[ a_n \cdot log(\hat{a}_n(P, D)) + (1 - a_n) \cdot log(1 - \hat{a}_n(P, D)) \Big]$$

# Curriculum Learning

- First proposed by Bengio et al. (2009)
- Introduce samples with increasing "difficulty".
- Better local minima even under non-convex loss.

# Pre-training and Joint-training

Pre-training

- Have a pre-trained model to initially guide the training process in a similar domain.

Joint-training

- Exploit the similarity between two different domains by training the model is two different domains simultaneously.

# Proposed Method

# Curriculum Inspired Training (CIT)

## Difficulty Measurement

$$SF(q, S, C, s) = \frac{\sum_{word \in \{q \cup S \cup C\}} log(Freq(word))}{\#\{q \cup S \cup C\}}$$

- Partition the dataset into a fixed number *chapter_size* with increasing difficulty.
- Each chapter consists of $\bigcup_{i=1}^{current\_chapter} parition[i]$.
- The model is trained with fixed number of epochs per chapter.

# CIT Cont.

## Loss Function

$$L(P, D, en) = \frac{1}{N_D} \sum_{n=1}^{N_D} \Big[ (a_n \cdot log(\hat{a}_n(P, D))) + (1 - a_n) \cdot log(1 - \hat{a}_n(P, D)) \cdot 1_{en >= c(n) \cdot epc} \Big]$$

- $en$ : Current epoch
- $c(n)$ : Chapter number that the example $n$ is assigned to
- $epc$ : Epochs per chapter

# Joint-Training

## General Joint Loss Function

$$\hat{L}(P, TD, SD) = 2\gamma \cdot L(P, TD) + 2(1-\gamma) \cdot L(P, SD) \cdot F(N_{TD}, N_{SD})$$

- $TD$: Target dataset
- $SD$: Source dataset
- $N_D$: Number of examples in the dataset $D$
- $\gamma$: Tunable weight parameter

# Loss Functions

## Joint-training

$\gamma = \frac{1}{2}$ and $F(N_{TD}, N_{SD}) = 1$

$$\hat{L}(P, TD, SD) = L(P, TD) + L(P, SD)$$

## Weighted joint-training

$\gamma \in (0, 1)$ and $F(N_{TD}, N_{SD}) = \frac{N_{TD}}{N_{SD}}$.

$$\hat{L}(P, TD, SD) = 2\gamma \cdot L(P, TD) + 2(1-\gamma) \cdot L(P, SD) \cdot \frac{N_{TD}}{N_{SD}}$$

# Loss Functions Cont.

## Curriculum joint-training

$\gamma = \frac{1}{2}$ and $F(N_{TD}, N_{SD}) = 1$

$$\hat{L}(P, TD, SD) = L(P, TD, en) + L(P, SD, en)$$

## Weighted Curriculum joint-training

$\gamma \in (0, 1)$ and $F(N_{TD}, N_{SD}) = \frac{N_{TD}}{N_{SD}}$.

$$\hat{L}(P, TD, SD) = 2\gamma \cdot L(P, TD, en) + 2(1-\gamma)L(P, SD, en) \cdot \frac{N_{TD}}{N_{SD}}$$

## Source only

$\gamma = 0$ and $c \in \mathbb{R}^+$

$$\hat{L}(P, TD, SD) = c \cdot L(P, SD)$$

# Dataset and Experiment Results

# Dataset

| | MCTest-160 | MCTest-500 | CNN-11K | CNN-22K | CNN-55K | Dailymail-55K |
|---|---|---|---|---|---|---|
| # Train | 280 | 1400 | 11,000 | 22,000 | 55,000 | 55,000 |
| # Validation | 120 | 200 | 3,924 | 3,924 | 3,924 | 2,500 |
| # Test | 200 | 400 | 3,198 | 3,198 | 3,198 | 2,000 |
| # Vocabulary | 2856 | 4279 | 26,550 | 31,932 | 40,833 | 42,311 |
| # Words $\notin$ Dailymail-55K | — | — | 1,981 | 2,734 | 6,468 | — |

Figure: Dataset used for experiments.

# Experiment Results

| | CNN-11 K | | | CNN-22 K | | | CNN-55 K | | |
|---|---|---|---|---|---|---|---|---|---|
| Model + Training Methods | Train | Valid | Test | Train | Valid | Test | Train | Valid | Test |
| SW § | 21.33 | 20.35 | 21.48 | 21.80 | 20.61 | 20.76 | 21.54 | 19.87 | 20.66 |
| SW+D § | 25.45 | 25.40 | 25.90 | 25.61 | 25.25 | 26.47 | 25.85 | 25.74 | 26.94 |
| SW+W2V § | 43.90 | 43.01 | 42.60 | 45.70 | 44.10 | 42.23 | 45.06 | 44.50 | 43.50 |
| MemNN § | 98.98 | 45.96 | 46.08 | 98.07 | 49.28 | 51.42 | 97.31 | 54.98 | 56.69 |
| **MemNN+CIT** § | 96.44 | 47.17 | **49.04** | 98.36 | 52.43 | **52.73** | 91.14 | 57.26 | **57.68** |
| SW+Dailymail ‡ | 30.19 | 31.21 | 30.60 | 31.70 | 30.87 | 32.01 | 31.56 | 33.07 | 31.08 |
| MemNN+W2V ‡ | 86.57 | 43.78 | 45.99 | 94.1 | 49.98 | 51.06 | 95.2 | 51.47 | 53.66 |
| MemNN+SrcOnly ‡ | 25.12 | 26.78 | 27.08 | 25.43 | 26.78 | 27.08 | 24.79 | 26.78 | 27.08 |
| MemNN+Pre-train ‡ | 92.82 | 52.87 | 52.06 | 95.12 | 53.59 | 55.35 | 96.33 | 56.64 | 59.19 |
| MemNN+Jo-train ‡ | 65.78 | 53.85 | 55.06 | 64.85 | 55.94 | 55.69 | 77.32 | 57.76 | 57.99 |
| **MemNN+CIT+Jo-train** ‡ | 77.74 | 55.93 | 55.74 | 78.96 | 55.98 | 56.85 | 71.89 | 56.83 | 59.07 |
| **MemNN+W+Jo-train**‡ | 71.72 | 54.30 | 55.70 | 79.64 | 55.91 | 56.73 | 71.15 | 57.62 | 58.34 |
| **MemNN+W+CIT+Jo-train** ‡ | 80.14 | 56.91 | **57.02** | 79.04 | 57.90 | **57.71** | 76.91 | 58.14 | **59.88** |

Figure: The table has two major rows. The upper row are models that only used the target dataset. The lower rows are models that used both the target and source dataset.

# Experiment Results

| Model + Training Methods | Exact | Para. | Part.Clue | Multi.Sent. | Co-ref. | Ambi./Hard |
|---|---|---|---|---|---|---|
| SW § | 3(23.1%) | 12(29.2%) | 2(10.5%) | 0(0.0%) | 0(0.0%) | 2(11.7%) |
| SW+D § | 6(46.1%) | 14(34.1%) | 2(10.5%) | 0(0.0%) | 0(0.0%) | 3(17.6%) |
| SW+W2V § | 10(76.9%) | 20(48.7%) | 5(26.3%) | 0(0.0%) | 0(0.0%) | 7(41.1%) |
| MemNN § | 8(61.5%) | 20(48.7%) | 12(63.1%) | 1(50.0%) | 0(0.0%) | 2(11.7%) |
| **MemNN+CIT** § | 10(76.9%) | 19(46.3%) | 12(63.1%) | 1(50.0%) | 3(37.5%) | 2(11.7%) |
| SW+Dailymail ‡ | 6(46.1%) | 19(46.3%) | 5(26.3%) | 0(0.0%) | 0(0.0%) | 2(11.7%) |
| MemNN+W2V ‡ | 6(46.1%) | 27(65.8%) | 5(26.3%) | 0(0.0%) | 0(0.0%) | 7(41.1%) |
| MemNN+SrcOnly § | 6(46.1%) | 12(29.2%) | 2(10.5%) | 0(0.0%) | 0(0.0%) | 2(11.7%) |
| MemNN+Pre-train ‡ | 11(84.6%) | 25(60.9%) | 12(63.1%) | 0(0.0%) | 0(0.0%) | 1(5.9%) |
| MemNN+Jo-train ‡ | 8(61.5%) | 29(70.7%) | 10(52.6%) | 2(100%) | 0(0.0%) | 5(29.4%) |
| **MemNN+CIT+Jo-train** ‡ | 10(76.9%) | 27(65.8%) | 10(52.6%) | 0(0.0%) | 3(37.5%) | 5(29.4%) |
| **MemNN+W+Jo-train** ‡ | 11(84.6%) | 29(70.7%) | 10(52.6%) | 2(100%) | 0(0.0%) | 5(29.4%) |
| **MemNN+W+CIT+Jo-train** ‡ | **11(84.6%)** | 27(65.8%) | 10(52.6%) | **2(100%)** | **3(37.5%)** | **5(29.4%)** |
| Chen et al. (2016) $ | 13(100%) | 39(95.1%) | 17(89.5%) | 1(50.0%) | 3(37.5%) | 1(5.9%) |
| Sordoni et al. (2016) $ | 13(100%) | 39(95.1%) | 16(84.2%) | 1(50.0%) | 3(37.5%) | 5(29.4%) |
| Total Number Of Samples | 13 | 41 | 19 | 2 | 8 | 17 |

Figure: Categorical performance measurement in CNN-11 K . The table has two major rows. The upper row are models that only used the target dataset. The lower rows are models that used both the target and source dataset.

# Experiment Results

| | MCTest-160 | | | MCTest-500 | | |
|---|---|---|---|---|---|---|
| Training Methods | One | Multi. | All | One | Multi. | All |
| SW | 66.07 | 53.12 | 59.16 | 54.77 | 53.04 | 53.83 |
| SW+D | 75.89 | 60.15 | 67.50 | 63.23 | 57.01 | 59.83 |
| SW+D+W2V | 79.46 | 59.37 | 68.75 | 65.07 | 58.84 | 61.67 |
| SW+D+CNN-11K | 79.78 | 59.37 | **67.67** | 64.33 | 57.92 | **60.83** |
| SW+D+CNN-22K | 76.78 | 60.93 | **68.33** | 64.70 | 59.45 | **61.83** |
| SW+D+CNN-55K | 78.57 | 59.37 | **68.33** | 65.07 | 59.75 | **62.16** |
| SW+D+CNN-11K+W2V | 77.67 | 59.41 | 68.69 | 65.07 | 61.28 | 63.00 |
| SW+D+CNN-22K+W2V | 78.57 | 60.16 | 69.51 | 66.91 | 60.00 | 63.13 |
| SW+D+CNN-55K+W2V | 79.78 | 60.93 | **70.51** | 66.91 | 60.67 | **63.50** |

Figure: Knowledge transfer performance result.

# Experiment Results

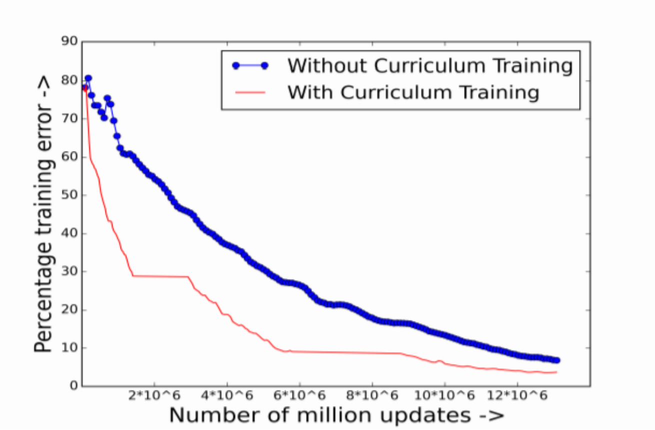


Figure: Loss convergence comparison between model trained with CIT and without CIT.

# Summary

- MemNN is often used in QA.
- Ordering the samples lead to better local minima.
- Joint-training is useful in obtaining better performance on small target dataset.
- Using pre-trained model improves performance.

# The End