aNMM: Ranking Short Answer Texts with Attention-Based Neural Matching Model

Liu Yang\textsuperscript{1}, Qingyao Ai\textsuperscript{1}, Jiafeng Guo\textsuperscript{2}, W. Bruce Croft\textsuperscript{1}

\textsuperscript{1}Center for Intelligent Information Retrieval, University of Massachusetts Amherst

\textsuperscript{2}Institute of Computing Technology, Chinese Academy of Sciences
Outline

• Motivation

• Related Works
  – Learning to rank for QA
  – Deep learning for QA

• Attention-based Neural Matching Model

• Experiments
  – Data Set and Experiment Settings
  – Model Learning Results
  – Experimental Results for Ranking Answers

• Conclusions and Future Work
Motivation

• **Question answering** plays a central role in many popular mobile search systems and intelligent assistant systems
  – Google Assistant, Microsoft Cortana, Microsoft Xiaoice, IBM Watson, etc.

• Users are more likely to expect direct answers instead of a rank list of documents from search results
  – Retrieve finer grained text units such as *passages or sentences* as *answers* for *Web queries* or *questions*
Types of Questions

- **Factoid queries**: WH questions like when, who, where.
- **Yes/No queries**: Is Berlin capital of Germany?
- **Definition queries**: what is leukemia?
- **Cause/consequence queries**: How, Why, What. what are the consequences of the Iraq war?
- **Procedural queries**: which are the steps for getting a Master degree?
- **Comparative queries**: what are the differences between the model A and B?
- **Queries with examples**: list of hard disks similar to hard disk X.
- **Queries about opinion**: What is the opinion of the majority of Americans about the Iraq war?
QA Approaches

• IR based approaches
  – TREC QA, IBM Watson, Google

• Knowledge base based approaches
  – Apple Siri, IBM Watson

• Many QA systems used hybrid approaches
Many previous QA systems used a learning to rank approach:

- Encode question/answers with complex linguistic features including lexical, syntactic and semantic features.
- E.g. Surdeanu et al. [1,2] investigated a wide range of feature types for learning to rank answers.

Problems with learning to rank approaches:

- Reply on feature engineering, which is time consuming and requires domain dependent expertise.
- Need additional NLP parsers or external knowledge sources:
  - may not be available for some languages.

Recently researchers have been studying deep learning approaches to learn semantic match between questions and answers

- Convolutional Neural Networks (CNN) [3, 4, 5]
- Long Short-Term Memory (LSTM) networks [6]
- Benefit of not requiring hand-crafted linguistic features and external resources except pre-trained word embedding
- Some of them [5] achieve state-of-the-art performance for answer sentence selection task benchmarked by the TREC QA Data

Deep Learning for QA

• Problems with current deep learning architectures for answer sentence selection
  – The proposed models, either based on CNN or LSTM, need to be combined with additional features such as word overlap features [3,5] and BM25 [6] to perform well
  – Without combining additional features, the performance of their model is significantly worse
    • Comparing with the results from the state-of-the-art methods using linguistic feature engineering [7]

• Research question:
  – Could we build deep learning models that can achieve comparable or even better performance without combining additional features than methods using feature engineering?
  – End-to-end question answering?

Observations From the Current Deep Learning Architectures for Ranking Answers

- Architectures not specifically designed for question/answer matching
  - CNN
    - Uses position-shared weights with local perceptive filters to learn spatial regularities as in many CV tasks
    - Such spatial regularities may not exist in the semantic matching between questions and answers
    - Complex linguistic property of natural languages
  - LSTM
    - View the question/answer matching problem in a sequential way
    - No direct interactions between question and answer terms
    - Can not capture sufficiently detailed matching signals

- Our solution
  - Introduce a novel value-shared weighting scheme in deep neural networks
  - Learn value regularities rather than spatial regularities
• Lack of modeling question focus
  – Understanding the focus of questions which are important terms is helpful for ranking answers correctly
    • E.g. Where was the first burger king restaurant opened?
  – Most existing text matching deep learning models do not explicitly model question focus

• Our solution
  – Incorporate attention scheme over question terms
    • Introduce attention mechanisms with a gating function
    • Explicitly discriminate the question term importance
• Motivation
• Related Works
  – Learning to rank for QA
  – Deep learning for QA
• Attention-based Neural Matching Model
• Experiments
  – Data Set and Experiment Settings
  – Model Learning Results
  – Experimental Results for Ranking Answers
• Conclusions and Future Work
• QA Matching Matrix
  – A matrix represents the semantic matching information of term pairs from a question and answer pair
  – Given a question $q$ with length $M$ and an answer $a$ with length $N$
    • An $M$ by $N$ matrix $P$
    • $P_{j,i}$ is the semantic similarity between $q_j$ and $a_i$ using word embedding
    • Assign value 1 if $q_j$ and $a_i$ are the same term
    • Inspired by the ARC-II model proposed by Hu et al. [8]

Attention-based Neural Matching Model

\[ y = \sum_{j=1}^{M} \tau(v \cdot q_j) \cdot \delta(\sum_{t=0}^{T} r_t \delta(\sum_{k=0}^{K} w_{kt} x_{jk})) \]

\( \tau \): softmax gate function
\( \delta \): sigmoid function

- Neural network architecture with value-shared weights
• In CNN, the weight associated with a node only depends on its position as specified by the filters
• In aNMM, the weight associated with a node depends on its value

\[ y = \sum_{j=1}^{M} \tau(v \cdot q_j) \cdot \delta(\sum_{t=0}^{T} r_t \cdot \delta(\sum_{k=0}^{K} w_{kt}x_{jk})) \]

\( \tau \): softmax gate function
\( \delta \): sigmoid function

• Neural network architecture with value-shared weights
Question Attention Network

\[ y = \sum_{j=1}^{M} \tau(v \cdot q_j) \cdot \delta\left(\sum_{t=0}^{T} r_t \cdot \delta\left(\sum_{k=0}^{K} w_{kt} x_{jk}\right)\right) \]

\( \tau \): softmax gate function
\( \delta \): sigmoid function

- Neural network architecture with attention schemes
Two Variations: aNMM-1 and aNMM-2

- aNMM-1: basic architecture
  \[ aNMM-1: y = \sum_{j=1}^{M} \tau(v \cdot q_j) \cdot \delta(\sum_{k=0}^{K} W_k x_{jk}) \]

- aNMM-2: Extension with multiple sets of value-share weights
  \[ aNMM-2: y = \sum_{j=1}^{M} \tau(v \cdot q_j) \cdot \delta(\sum_{t=0}^{T} r_t \cdot \delta(\sum_{k=0}^{K} W_{kt} x_{jk})) \]
Back Propagation for Model Training

- Backward propagation with stochastic gradient descent
- Pairwise Learning
- Given a triple \((q, a^+, a^-)\) where
  - \(q\) question sentence
  - \(a^+\) correct answer sentence
  - \(a^-\) wrong answer sentence
  - Hinge Loss function \(e(q, a^+, a^-; w, r, v) = \max(0, 1 - S(q, a^+) + S(q, a^-))\)
  - Compute \(\Delta S = 1 - S(q, a^+) + S(q, a^-)\)
  - If \(\Delta S \leq 0\) Skip this triple
    - If \(\Delta S > 0\) Compute the gradients w.r.t \(v, r, w\)
    - Update the model parameters to minimize the loss function with BP algorithm
Outline

• Motivation
• Related Works
  – Learning to rank for QA
  – Deep learning for QA
• Attention-based Neural Matching Model
• Experiments
  – Data Set and Experiment Settings
  – Model Learning Results
  – Experimental Results for Ranking Answers
• Conclusions and Future Work
Experimental Data and Settings

- TREC QA data set from TREC QA track 8-13
  - One of the most widely used benchmarks for answer sentence selection/ranking
  - Contains a set of factoid questions with candidate answers which are limited to a single sentence
  - Judgements in TRAIN and TRAIN-ALL
  - Word embedding: pre-trained with English Wikipedia dump with the Word2Vec tool by Mikolov et. al [9, 10]

- Statistics of the TREC QA data set

<table>
<thead>
<tr>
<th>Data</th>
<th>#Questions</th>
<th>#QA pairs</th>
<th>%Correct</th>
<th>#Answers/Q</th>
<th>Judgement</th>
</tr>
</thead>
<tbody>
<tr>
<td>TRAIN-ALL</td>
<td>1,229</td>
<td>53,417</td>
<td>12.00%</td>
<td>43.46</td>
<td>automatic</td>
</tr>
<tr>
<td>TRAIN</td>
<td>94</td>
<td>4,718</td>
<td>7.40%</td>
<td>50.19</td>
<td>manual</td>
</tr>
<tr>
<td>DEV</td>
<td>82</td>
<td>1,148</td>
<td>19.30%</td>
<td>14.00</td>
<td>manual</td>
</tr>
<tr>
<td>TEST</td>
<td>100</td>
<td>1,517</td>
<td>18.70%</td>
<td>15.17</td>
<td>manual</td>
</tr>
</tbody>
</table>

[9] https://code.google.com/archive/p/word2vec/
Model Learning Results

- Visualization of learned question term importance

<table>
<thead>
<tr>
<th></th>
<th>Term Importance</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>test_14</td>
<td>when</td>
<td>4.91E-03</td>
<td>did</td>
<td>7.18E-04</td>
<td>the</td>
<td>8.97E-04</td>
<td>khmer</td>
<td>5.67E-01</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>test_66</td>
<td>where</td>
<td>2.16E-04</td>
<td>was</td>
<td>5.67E-04</td>
<td>the</td>
<td>1.96E-04</td>
<td>first</td>
<td>2.57E-03</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>train_84</td>
<td>at</td>
<td>5.06E-02</td>
<td>what</td>
<td>2.54E-03</td>
<td>age</td>
<td>6.17E-02</td>
<td>did</td>
<td>2.68E-03</td>
</tr>
</tbody>
</table>
Experimental Results

• Learning without combining additional features

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wang et al. (2007) [27]</td>
<td>0.6029</td>
<td>0.6852</td>
</tr>
<tr>
<td>Heilman and Smith (2010) [5]</td>
<td>0.6091</td>
<td>0.6917</td>
</tr>
<tr>
<td>Wang and Manning (2010) [26]</td>
<td>0.5951</td>
<td>0.6951</td>
</tr>
<tr>
<td>Yao et al. (2013) [31]</td>
<td>0.6307</td>
<td>0.7477</td>
</tr>
<tr>
<td>Severyn et al. (2013) [17]</td>
<td>0.6781</td>
<td>0.7358</td>
</tr>
<tr>
<td>Yih et al. (2013) [32]</td>
<td>0.7092</td>
<td>0.7700</td>
</tr>
<tr>
<td>aNMM-2</td>
<td>0.7407</td>
<td>0.7969</td>
</tr>
<tr>
<td>aNMM-1</td>
<td>0.7385</td>
<td>0.7995</td>
</tr>
</tbody>
</table>

Compare with methods using feature engineering (on TRAIN-ALL)

Compare with deep learning methods

<table>
<thead>
<tr>
<th>Training Data</th>
<th>TRAIN</th>
<th>TRAIN-ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Method</td>
<td>MAP</td>
<td>MRR</td>
</tr>
<tr>
<td>Yu et al. (2014) [34]</td>
<td>0.5476</td>
<td>0.6437</td>
</tr>
<tr>
<td>Wang et al. (2015) [25]</td>
<td>/</td>
<td>/</td>
</tr>
<tr>
<td>Severyn et al. (2015) [18]</td>
<td>0.6258</td>
<td>0.6591</td>
</tr>
<tr>
<td>aNMM-2</td>
<td>0.7191</td>
<td>0.7974</td>
</tr>
<tr>
<td>aNMM-1</td>
<td>0.7334</td>
<td><strong>0.8020</strong></td>
</tr>
</tbody>
</table>

• Achieve better performance comparing with other methods using feature engineering
• Show significant improvements comparing with previous deep learning methods
• Results of aNMM-1 and aNMM-2 are very close
• aNMM-1 could be trained with higher efficiency
Experimental Results

- Learning with combining additional features

Compare with deep learning methods
Severyn et al. (SIGIR 2015) is the state-of-the-art result

Overview of previously published results on TREC QA data (the best setting of each model trained on TRAIN-ALL)

<table>
<thead>
<tr>
<th>Training Data</th>
<th>METHOD</th>
<th>TRAIN</th>
<th>TRAIN-ALL</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MAP</td>
<td>MRR</td>
<td>MAP</td>
</tr>
<tr>
<td>Method</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yu et al. (2014) [34]</td>
<td>0.7058</td>
<td>0.7800</td>
<td>0.7113</td>
</tr>
<tr>
<td>Wang et al. (2015) [25]</td>
<td>/</td>
<td>/</td>
<td>0.7134</td>
</tr>
<tr>
<td>Severyn et al. (2015) [18]</td>
<td>0.7329</td>
<td>0.7962</td>
<td>0.7459</td>
</tr>
<tr>
<td>aNMM-2</td>
<td>0.7306</td>
<td>0.7968</td>
<td>0.7484</td>
</tr>
<tr>
<td>aNMM-1</td>
<td><strong>0.7417</strong></td>
<td><strong>0.8102</strong></td>
<td><strong>0.7495</strong></td>
</tr>
</tbody>
</table>

- Combine the score of aNMM-1/aNMM-2 with QL score
- With the combined feature, both aNMM-1 and aNMM-2 have better performances
- aNMM-1 also outperforms CDNN by Severyn et al. ([5] in SIGIR 2015) which is the current state-of-the-art method
• Motivation
• Related Works
  – Learning to rank for QA
  – Deep learning for QA
• Attention-based Neural Matching Model
• Experiments
  • Data Set and Experiment Settings
  • Model Learning Results
  • Experimental Results for Ranking Answers
• Conclusions and Future Work
Conclusions and Future Work

• Propose an attention based neural matching model for ranking short answer text
  – Adopt value-shared weighting scheme instead of position-shared weighting scheme for combining matching signals
  – Incorporate question term importance learning using a question attention network

• Perform a thorough experimental study with TREC QA data and show promising results
  – Without combining additional features
    • Outperform previous deep learning methods and feature engineering methods with large gains
  – With one simple additional feature
    • Outperform the state-of-the-art method
Conclusions and Future Work

• Additional results on Microsoft Research WikiQA data [11]
  – Double confirms the advantages of the attention based neural matching models for ranking answer sentences.

<table>
<thead>
<tr>
<th>Method</th>
<th>MAP</th>
<th>MRR</th>
</tr>
</thead>
<tbody>
<tr>
<td>WordCount</td>
<td>0.4891</td>
<td>0.4924</td>
</tr>
<tr>
<td>WeightedWordCount</td>
<td>0.5099</td>
<td>0.5132</td>
</tr>
<tr>
<td>LCLR</td>
<td>0.5993</td>
<td>0.6086</td>
</tr>
<tr>
<td>PV</td>
<td>0.5110</td>
<td>0.5160</td>
</tr>
<tr>
<td>CNN</td>
<td>0.6190</td>
<td>0.6281</td>
</tr>
<tr>
<td>PV-Count</td>
<td>0.5976</td>
<td>0.6058</td>
</tr>
<tr>
<td>CNN-Count</td>
<td>0.6520</td>
<td>0.6652</td>
</tr>
<tr>
<td>aNMM-2</td>
<td>0.6455</td>
<td>0.6527</td>
</tr>
<tr>
<td>aNMM-1</td>
<td>0.6562</td>
<td>0.6687</td>
</tr>
</tbody>
</table>

• Future work
  – Extend our work to include non-factoid question answering data sets
    • Yahoo CQA /Stack Overflow/ WebAP
  – Interactive QA & Natural language dialogue for FAQ search

Thank You

Q&A

Email: lyang@cs.umass.edu
https://sites.google.com/site/lyangwww/