Using Paraphrasing and Memory-Augmented Models to Combat Data Sparsity in Question Interpretation with a Virtual Patient Dialogue System

Lifeng Jin, David King, Amad Hussein, Michael White and Douglas Danforth

Department of Linguistics, Department of Computer Science, Wexner Medical Center
The Ohio State University

June 5 @ 13th BEA
Outline

1. Introduction

2. Addressing the long tail
   - Memory augmentation
   - Data augmentation

3. Experiments

4. Conclusion
Introduction

Virtual Patients: A dialogue system

One virtual patient: Mr. Wilkins
An example dialogue session

The student asks:
What brings you in today?

System predicts the label as “what brings you in today” and responds:
I was hoping you could help me with my back pain, it really hurts! ...

The student asks:
Could you tell me more about the pain?

System predicts the label as “describe the pain” and responds:
It’s a dull ache right in the middle of my lower back.

...
| The student asks: | System predicts the label as “what pills did you take” and responds: |
|------------------|-------------------------------------------------------------------|
| What medications did you take? | I took some ibuprofen for a few days. |

| The student asks: | System predicts the label as “have you exercised recently” and responds: |
|------------------|-------------------------------------------------------------------|
| Have you had any surgery recently? | No, my back hurt and I couldn't really exercise. |
Virtual patient dataset

- 94 dialogues with 4330 turns
- hand-corrected question labels
- 359 unique labels

Number of instances for a label.
Introduction

Examples of frequent and rare labels

Example frequent label: what brings you in today
- can you tell me a little about your issue
- what brings you in today
- so can you tell me what brings you in today
- what brings you into the office today
- what is it you want to talk about today
...

Example rare label: do you feel safe.
- that sounds like a fun job. do you feel safe at home and work
Previous work: an ensemble of CNNs

- **Stacking**
  - **Ystacking**
    - **Ychar_ensemble**
      - **Majority voting**
        - **Ychar_1**
        - **...**
        - **Ychar_5**
    - **Yword_ensemble**
      - **Majority voting**
        - **Yword_1**
        - **...**
        - **Yword_5**

- **CharCNN_1**
- **...**
- **CharCNN_5**

- **WordCNN_1**
- **...**
- **WordCNN_5**

*Jin et al (Ohio State)*
Stacked CNN significantly better than other machine learning models

| System     | Simple | Ensemble |
|------------|--------|----------|
| ChatScript | 79.8   | n/a      |
| Baseline   | 77.2   | n/a      |
| CharCNN    | 76.16  | 78.20    |
| WordCNN    | 76.92  | 77.67    |
| Stacked    | n/a    | 79.02*   |

Mean 10-fold Accuracy by System Type. Numbers reported are on the test set. The improvement between the stacked model and any other model is significant. Ensembling character CNNs provides significant performance boost, but not word CNNs.
Addressing the long tail

Frequency quintile analysis: data is good for CNNs

System Accuracy by Label Frequency, in Quintiles. Note the high performance in the least frequent labels for ChatScript, the hand-crafted pattern matching system. With more data, the CNNs perform better.
We approach the problem of rare labels from two different angles:

1. Make the model good at dealing with such items:
   - **Few-shot learning**

2. Make them no longer rare:
   - **Data augmentation**
Unlike most machine learning algorithms, humans are perfectly capable of learning with few examples. (Lake et al, 2009)
In order to give neural networks the ability to remember specific examples, one approach is to give them memory to remember specific past events. Let’s see how the memory module works.
How the memory operates

representation of a sentence

e \quad s = Ke

The memory module

the memory bank \quad labels \quad age

K \quad V \quad A

update rules:

s[0]

s[n]

CNN Encoder

What brings you in today?
How the memory operates

representation of a sentence

The memory module

| the memory bank | labels | age |
|-----------------|--------|-----|
| K               | V      | A   |

| e -> s = Ke |
|-------------|

- loss = [s[0] - s[n] + α]_+
- update rules:
  - $K[0] \leftarrow \frac{e + K[0]}{\|e + K[0]\|}$
  - $A \leftarrow A + 1$
  - $A[0] \leftarrow 0$

What brings you in today?
How the memory operates

The memory module

representation of a sentence

\[ e \rightarrow s = Ke \]

The memory bank

labels

age

\[ K \]

\[ V \]

\[ A \]

What brings you in today?

loss = \[ [s[0] - s[n] + \alpha]_+ \]

update rules:

\[ K[n'] \leftarrow e \]
\[ V[n'] \leftarrow v \]
\[ A \leftarrow A + 1 \]
\[ A[n'] \leftarrow 0 \]
Episodic training

- Sample an episode
- Init shot
- Training shots

Training data

Memory module
Episodic training

- Addressing the long tail
- Memory augmentation

Jin et al (Ohio State)
Paraphrasing and Mem models
June 5 @ 13th BEA
Episodic training

- Addressing the long tail
- Memory augmentation

- Sample an episode
- Init shot
- Training shots
- Training data
- The memory module
- Encoder
- Loss

Jin et al (Ohio State)
Memory module

The memory module (Kaiser et al., 2017) is like an external database which has a storage for question representations, question labels and age of entries. The neural network (encoder) can read and write it to keep it updated. This helps the neural network to remember the rare instances. Training the memory module also requires balancing the training data, which also helps give the rare labels better representations.
Episodic evaluation

- Sample a support set
- Support shots
- Training data
- Test data
- The memory module

Jin et al (Ohio State)
Episodic evaluation

- Sample a support set
- Support shots
- Training data
- Test data

Encoder

Memory module
Episodic evaluation

Addressing the long tail
Memory augmentation

Jin et al (Ohio State)
Episodic evaluation

- Sample a support set
- Support shots
- Training data
- Encoder
- Test predictions
- Test data

Jin et al. (Ohio State)
Episodic evaluation

- Sample a support set
- Support shots
- Training data
- Test data
- The memory module
- Compare and predict
- Encoder
- Test predictions
MA-CNN on rare labels

| System     | Full Acc | Rare Acc |
|------------|----------|----------|
| StackedCNN | 79.02    | 46.54    |
| MA-CNN     | 75.22    | **51.78***** |

Test results for the stacked CNN ensemble (Jin et al., 2017) and the memory-augmented CNN classifier (MA-CNN) without any generated paraphrases. The difference of performance on the rare items is highly significant ($p = 9.5 \times 10^{-5}$, *McNemar’s test*).
Paraphrase generation

We augment the rare labels by generating new training instances with different paraphrasing methods.

1. **Lexical substitution**
   - Sources: WordNet, Word2Vec, PPDB
   - Ranked by likelihood ratio between the original and the generated

2. **Neural machine translation**
   - $10 \times 10$ back translation (Mallinson et al., 2017)
   - Scores combined to produce a ranking
   - Used German as the pivot language
Filtering the generated phrases

There are still too many paraphrases for human filtering, and too noisy.

Ideally all paraphrases are filtered automatically, but we need to know if they are any good at all.
Filtering the generated phrases

1. Pseudo-oracle (automatic)
   1. Captures surface similarity
   2. Keeps a generated paraphrase when its n-gram recall is higher than that of the original when compared to a test item
   3. Could still be noisy

2. Manual
   1. Captures semantic similarity
   2. Keeps a generated paraphrase only if it has novel n-grams compared to training items

They validate the quality of generated paraphrases.
Generated paraphrases in training data

| System               | Full Acc | Rare Acc |
|----------------------|----------|----------|
| StackedCNN           | 79.02    | 46.54    |
| MA-CNN               | 75.22    | 51.78    |
| StackedCNN w/ GPs    | 78.45    | 53.04    |
| MA-CNN w/ GPs        | 75.33    | **56.14*** |

Test results for the stacked CNN ensemble and the memory-augmented CNN classifier (MA-CNN) with the manually filtered paraphrases. The gain brought by the adding the automatically generated paraphrases into training data for MA-CNN is highly significant \((p = 1.6 \times 10^{-4}, \text{McNemar’s test})\).
Ablation of filtering methods

| System                              | Rare Acc |
|-------------------------------------|----------|
| MA-CNN                              | 51.78    |
| +Pseudo-oracle                      | 54.87    |
| +Pseudo-oracle + Manual             | **56.14**|

Test results for the memory-augmented CNN classifier (MA-CNN) with different filtering techniques.
## Quality of generated paraphrases

| Paraphrases                              | Rare Acc |
|------------------------------------------|----------|
| No paraphrases                           | 51.78    |
| +Lexical substitution                    | 53.16    |
| +Neural Machine Translation              | 55.22    |
| +Both                                    | **56.14**|

Test results for the memory-augmented CNN classifier (MA-CNN) with different subsets of the manual filtered paraphrases generated using different paraphrase methods.
Combining the stacked CNN and the MA-CNN

| System      | Full Acc | Rare Acc |
|-------------|----------|----------|
| StackedCNN  | 79.02    | 46.54    |
| MA-CNN      | 75.33    | 56.14    |
| Combiner    | **79.86**| **50.98**|

Test results for the combiner as well as the two combined subsystems: the stacked CNN ensemble trained with gold and the memory-augmented CNN classifier trained with gold and generated paraphrases. The gain compared to stacked CNN on full accuracy is highly significant ($p = 1.9 \times 10^{-9}$, McNemar’s test).
Lexical substitution is good and neural back-translation is better.
Memory-augmented CNN classifier is better on low frequency labels with a smaller model.
MA-CNN and StackCNN can work together to be better.
Future work

- Automatic filtering
- Advanced paraphrasing
  - deep generative paraphrasing
  - syntactic paraphrasing
  - using aligned paraphrases to induce paraphrase templates
Thank you all for your attention.

Thanks to Kellen Maicher for creating the virtual environment and to Evan Jaffe, Eric Fosler-Lussier and William Schuler for feedback and discussion. This project was supported by funding from the Department of Health and Human Services Health Resources and Services Administration (HRSA D56HP020687), the National Board of Medical Examiners Edward J. Stemmler Education Research Fund (NBME 1112-064), and the National Science Foundation (NSF IIS 1618336). The project does not necessarily reflect NBME policy, and NBME support provides no official endorsement. We thank Ohio Supercomputer Center (1987) for computation support.
Lifeng Jin, Michael White, Evan Jaffe, Laura Zimmerman, and Douglas Danforth. 2017. Combining CNNs and Pattern Matching for Question Interpretation in a Virtual Patient Dialogue System. In Proceedings of the 12th Workshop on Innovative Use of NLP for Building Educational Applications. pages 11–21.
https://aclanthology.coli.uni-saarland.de/papers/W17-5002/w17-5002
http://aclweb.org/anthology/W17-5002.

Lukasz Kaiser, Ofir Nachum, Aurko Roy, and Samy Bengio. 2017. Learning to Remember Rare Events. In Proceedings of the International Conference on Learning Representations.
https://arxiv.org/pdf/1703.03129.pdf.

Jonathan Mallinson, Rico Sennrich, and Mirella Lapata. 2017. Paraphrasing revisited with neural machine translation. In Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers. volume 1, pages 881–893.

The Ohio Supercomputer Center. 1987. Ohio Supercomputer Center.