Marrying Up Regular Expressions with Neural Networks: A Case Study for Spoken Language Understanding

Bingfeng Luo, Yansong Feng, Zheng Wang, Songfang Huang, Rui Yan and Dongyan Zhao

2018/07/18
Data is Limited

- Most of the popular models in NLP are data-driven
- We often need to operate in a specific scenario → Limited data
Data is Limited

- Take **spoken language understanding** as an example
  - Understanding user query
  - Need to be implemented for many domains

**Intent Detection**
flights from Boston to Tokyo \(\rightarrow\) intent: **flight**

**Slot Filling**
flights from **Boston** to **Tokyo** \(\rightarrow\) fromloc.city: **Boston** toloc.city: **Tokyo**
Data is Limited

- **Take spoken language understanding** as an example
  - Need to be implemented for many domains → Limited data
  - E.g., intelligent customer service robot

- What can we do with limited data?

**Intent Detection**
flights from Boston to Tokyo → intent: *flight*

**Slot Filling**
flights from **Boston** to **Tokyo** → fromloc.city: *Boston* toloc.city: *Tokyo*
Regular Expression Rules

- When data is limited → Use **rule-based system**

- **Regular expression** is the most commonly used rule in NLP
  - Many regular expression rules in company

```
/^flights? from/    flights from Boston to Tokyo → intent: flight

/from (_,CITY) to (_,CITY)/    flights from Boston to Tokyo → fromloc.city: Boston toloc.city: Tokyo
```

_CITY=Boston | Tokyo | Beijign | ...
However, regular expressions are hard to generalize.

Neural networks are potentially good at generalization.

Can we combine the advantages of two worlds?

**Regular Expressions**

```
/^flights? from/  
```

- **Pro**: controllable, do not need data
- **Con**: need to specify every variation

**Neural Network**

```
[0.23, 0.11, -0.32, ...]  
```

- **Pro**: semantic matching
- **Con**: need a lot of data
Which Part of Regular Expression to Use?

- Regular expression (RE) output is useful
  - As feature
  - Fusion in output

### Intent Detection

```
/^flights? from/  
flights from Boston to Tokyo  
intent: flight
```

### Slot Filling

```
/from (_CITY) to (_CITY)/  
flights from Boston to Tokyo  
fromloc.city: Boston
toloc.city: Tokyo
```
Which Part of Regular Expression to Use?

- Regular expression (RE) output is useful

- RE contains **clue words**

  - NN should attend to these clue words for prediction

- Guide attention module

---

**Intent Detection**

/^flights? from/  
flights from Boston to Tokyo  ➔  intent: flight

**Slot Filling**

/from(_CITY) to(_CITY)/  
flights from Boston to Tokyo  ➔  fromloc.city: Boston toloc.city: Tokyo
Method 1: RE Output - As Features

- Embed the REtag, append to input

Intent Detection

REtag: flight

Softmax Classifier

Attention Aggregation

BLSTM

RE Instance

flights from Boston to Miami

/^flights? from/
Method 1: RE Output - As Features

- Embed the REtag, append to input

Slot Filling

```
REtag: O   O   B-loc.city   O   B-loc.city
/f/e/ /from__CITY to__CITY/
```
Method 2: RE Output - Fusion in Output

- \( \logit_k = \logit'_k + w_k z_k \)

- \( \logit'_k \) is the NN output score for class k (before softmax)

- \( z_k \in \{0, 1\} \), whether regular expression predict class k

Intent Detection
Method 2: RE Output - Fusion in Output

- $\text{logit}_k = \text{logit}'_k + w_k z_k$

- $\text{logit}'_k$ is the NN output score for class k (before softmax)

- $z_k \in \{0, 1\}$, whether regular expression predict class k

Slot Filling
Method 3: Clue Words - Guide Attention

- Attention should match clue words
  - Cross Entropy Loss

Intent Detection

```
Intent: flight

Attention Loss

Softmax Classifier

Attention Aggregation

BLSTM

s

h_1 h_2 h_3 h_4 h_5

RE Loss

RE Instance

Gold Att:

0.5 0.5 0
0 0 0

flights from

^
flights? from/

flights from Boston to Miami
```
Method 3: Clue Words - Guide Attention

- Attention should match clue words
- Cross Entropy Loss

Slot Filling

Gold Att:

```
0 1 0 0 0 0
```

/from __CITY to __CITY/
Method 3: Clue Words - Guide Attention

- Positive Regular Expressions (REs) & Negative REs
  - REs can indicate the input belong to class k, or does not belong to class k
  - Correction of wrong predictions

How long does it take to fly from LA to NYC?

\(^{/ \text{how long}/}^{\text{intention: abbreviation}}\)
Method 3: Clue Words - Guide Attention

- Positive Regular Expressions (REs) & Negative REs
  - Corresponding to positive / negative REs
  - $\text{logit}_k = \text{logit}_{k; \text{positive}} - \text{logit}_{k; \text{negative}}$

```
/^how long/  

How long does it take to fly from LA to NYC?  
```

intent: abbreviation
Method 3: Clue Words - Guide Attention

- Positive REs and Negative REs interconvertible
  - A positive RE for one class can be negative RE for other classes

```
/^flights? from/ flights from Boston to Tokyo
```

- intent: flight
- intent: abbreviation
- intent: airfare
- ...
Experiment Setup

- **ATIS Dataset**
  - 18 intents, 63 slots

- **Regular Expressions (RE)**
  - Written by a paid annotator
  - Intent: 54 REs, 1.5 hours
  - Slot: 60 REs, 1 hour (feature & output); 115 REs, 5.5 hours (attention)
We want to answer the following questions:

- Can regular expressions (REs) improve the neural network (NN) when data is limited (only use a small fraction of the training data)?
- Can REs still improve NN when using the full dataset?
- How does RE complexity influence the results?
# Few-Shot Learning Experiment

## Intent Detection

- **Macro-F1 / Accuracy**

- **5/10/20-shot:** every intent have 5/10/20 sentences

<table>
<thead>
<tr>
<th></th>
<th>5-shot</th>
<th>10-shot</th>
<th>20-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>45.28 / 60.02</td>
<td>60.62 / 64.61</td>
<td>63.60 / 80.52</td>
</tr>
<tr>
<td>feat</td>
<td>49.40 / 63.72</td>
<td>64.34 / 73.46</td>
<td>65.16 / 83.20</td>
</tr>
<tr>
<td>output</td>
<td>46.01 / 58.68</td>
<td>63.51 / 77.83</td>
<td>69.22 / <strong>89.25</strong></td>
</tr>
<tr>
<td>att</td>
<td><strong>54.86 / 75.36</strong></td>
<td><strong>71.23 / 85.44</strong></td>
<td><strong>75.58 / 88.80</strong></td>
</tr>
<tr>
<td>RE</td>
<td></td>
<td>70.31 / 68.98</td>
<td></td>
</tr>
</tbody>
</table>

Regular expressions help
Few-Shot Learning Experiment

- Intent Detection
  - Macro-F1 / Accuracy
  - 5/10/20-shot: every intent have 5/10/20 sentences

<table>
<thead>
<tr>
<th></th>
<th>5-shot</th>
<th>10-shot</th>
<th>20-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>45.28 / 60.02</td>
<td>60.62 / 64.61</td>
<td>63.60 / 80.52</td>
</tr>
<tr>
<td>feat</td>
<td>49.40 / 63.72</td>
<td>64.34 / 73.46</td>
<td>65.16 / 83.20</td>
</tr>
<tr>
<td>output</td>
<td>46.01 / 58.68</td>
<td>63.51 / 77.83</td>
<td>69.22 / <strong>89.25</strong></td>
</tr>
<tr>
<td>att</td>
<td><strong>54.86 / 75.36</strong></td>
<td><strong>71.23 / 85.44</strong></td>
<td><strong>75.58 / 88.80</strong></td>
</tr>
<tr>
<td>RE</td>
<td><strong>70.31 / 68.98</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Using clue words to **guide attention** performs best for intent detection.
Few-Shot Learning Experiment

- Slot Filling
  - Macro/Micro-F1
  - 5/10/20-shot: every intent have 5/10/20 sentences

<table>
<thead>
<tr>
<th></th>
<th>5-shot</th>
<th>10-shot</th>
<th>20-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>60.78 / 83.91</td>
<td>74.28 / 90.19</td>
<td>80.57 / 93.08</td>
</tr>
<tr>
<td>feat</td>
<td>66.84 / 88.96</td>
<td>79.67 / 93.64</td>
<td>84.95 / 95.00</td>
</tr>
<tr>
<td>output</td>
<td>63.68 / 86.18</td>
<td>76.12 / 91.64</td>
<td>83.71 / 94.43</td>
</tr>
<tr>
<td>att</td>
<td>59.47 / 83.35</td>
<td>73.55 / 89.54</td>
<td>79.02 / 92.22</td>
</tr>
<tr>
<td>RE</td>
<td>42.33 / 70.79</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Few-Shot Learning Experiment

- **Slot Filling**

  - **Macro/Micro-F1**

  - **5/10/20-shot**: every intent have 5/10/20 sentences

<table>
<thead>
<tr>
<th></th>
<th>5-shot</th>
<th>10-shot</th>
<th>20-shot</th>
</tr>
</thead>
<tbody>
<tr>
<td>base</td>
<td>60.78 / 83.91</td>
<td>74.28 / 90.19</td>
<td>80.57 / 93.08</td>
</tr>
<tr>
<td>feat</td>
<td>66.84 / 88.96</td>
<td>79.67 / 93.64</td>
<td>84.95 / 95.00</td>
</tr>
<tr>
<td>output</td>
<td>63.68 / 86.18</td>
<td>76.12 / 91.64</td>
<td>83.71 / 94.43</td>
</tr>
<tr>
<td>att</td>
<td>59.47 / 83.35</td>
<td>73.55 / 89.54</td>
<td>79.02 / 92.22</td>
</tr>
</tbody>
</table>

**RE**

Using RE output as **feature** performs best for slot filling
Full Dataset Experiment

- Use all the training data
  - RE still works!

<table>
<thead>
<tr>
<th></th>
<th>Intent</th>
<th>Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>base</strong></td>
<td>92.50/98.77</td>
<td>85.01/95.47</td>
</tr>
<tr>
<td><strong>feat</strong></td>
<td>91.86/97.65</td>
<td><strong>86.70/95.55</strong></td>
</tr>
<tr>
<td><strong>output</strong></td>
<td>92.48/98.77</td>
<td>86.94/95.42</td>
</tr>
<tr>
<td><strong>att</strong></td>
<td><strong>96.20/98.99</strong></td>
<td>85.44/95.27</td>
</tr>
<tr>
<td><strong>RE</strong></td>
<td>70.31/68.98</td>
<td>42.33/70.79</td>
</tr>
<tr>
<td><strong>SoA (Joint Model)</strong></td>
<td>- / 98.43</td>
<td>- / 95.98</td>
</tr>
</tbody>
</table>
Complex RE v.s. Simple RE

- **Complex RE:** many semantically independant groups

**Complex RE:** /(_AIRCRAFT_CODE) that fly/

**Simple RE:** /(_AIRCRAFT_CODE)/

<table>
<thead>
<tr>
<th></th>
<th>Intent</th>
<th>Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Complex</td>
<td>Simple</td>
</tr>
<tr>
<td>base</td>
<td>80.52</td>
<td></td>
</tr>
<tr>
<td>feat</td>
<td>83.20</td>
<td>80.40</td>
</tr>
<tr>
<td>output</td>
<td>89.25</td>
<td>83.09</td>
</tr>
<tr>
<td>att</td>
<td>88.80</td>
<td>87.46</td>
</tr>
</tbody>
</table>

Complex REs yield better results
**Complex RE v.s. Simple RE**

- **Complex RE:** many semantically independant groups

  **Complex RE:** /(_AIRCRAFT_CODE) that fly/

  **Simple RE:** /(_AIRCRAFT_CODE)/

<table>
<thead>
<tr>
<th></th>
<th>Intent</th>
<th>Slot</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Complex</td>
<td>Simple</td>
</tr>
<tr>
<td>base</td>
<td>80.52</td>
<td></td>
</tr>
<tr>
<td>feat</td>
<td>83.20</td>
<td>80.40</td>
</tr>
<tr>
<td>output</td>
<td>89.25</td>
<td>83.09</td>
</tr>
<tr>
<td>att</td>
<td>88.80</td>
<td>87.46</td>
</tr>
</tbody>
</table>

Simple REs also clearly improves the baseline
Conclusion

- Using REs can help to train of NN when data is limited

- Guiding attention is best for intent detection (sentence classification)

- RE output as feature is best for slot filling (sequence labeling)

- We can start with simple REs, and increase complexity gradually