NPFL099 Statistical Dialogue Systems
4. Language Understanding

http://ufal.cz/npfl099

Ondřej Dušek & Vojtěch Hudeček

20. 10. 2020
Natural Language Understanding

• **words → meaning**
  • whatever “meaning” is – can be different tasks
  • typically structured, explicit representation

• alternative names/close tasks:
  • *spoken language understanding*
  • *semantic decoding/parsing*

• integral part of dialogue systems, also explored elsewhere
  • stand-alone semantic parsers
  • other applications:
    • human-robot interaction
    • question answering
    • machine translation (not so much nowadays)
NLU Challenges

• non-grammaticality  find something cheap for kids should be allowed
• disfluencies
  • hesitations – pauses, fillers, repetitions  uhm I want something in the west the west part of town
  • fragments
  • self-repairs (~6%!)  uhm I’m looking for a cheap
• ASR errors  I’m looking for a for a chip Chinese rest or rant
• synonymy  Chinese city centre
  I’ve been wondering if you could find me a restaurant that has Chinese food close to the city centre please
• out-of-domain utterances  oh yeah I’ve heard about that place my son was there last month
Semantic representations

- **syntax/semantic trees**
  - typical for standalone semantic parsing
  - different variations

- **frames**
  - technically also trees, but smaller, more abstract
  - (mostly older) DSs, some standalone parsers

- **graphs (AMR)**
  - trees + co-reference
    (e.g. pronouns referring to the same object)

- **dialogue acts** = intent + slots & values
  - flat – no hierarchy
  - most DSs nowadays

inform(date=Friday, stay="2 nights")

(Seneff, 1992)  
https://www.aclweb.org/anthology/J92-1004

ouï l’hôtel don’t le prix ne dépasse pas cent dix euros

(Bonneau-Maynard et al., 2005)  
https://www.isca-speech.org/archive/interspeech_2005/i05_3457.html

http://cohort.inf.ed.ac.uk/amreager.html
Handling ASR noise

• ASR produces **multiple hypotheses**
• Combine & get resulting NLU hypotheses
  • NLU: \( p(\text{DA}|\text{text}) \)
  • ASR: \( p(\text{text}|\text{audio}) \)
  • we want \( p(\text{DA}|\text{audio}) \)
• Easiest: **sum it up**

\[
p(\text{DA}|\text{audio}) = \sum_{\text{texts}} P(\text{DA}|\text{text})P(\text{text}|\text{audio})
\]

• Alternative: **confusion nets** with weighted words
  • a more concise way of showing the same thing
Handling out-of-domain queries

• Handcrafted: **no pattern matches** → out-of-domain
• Datasets – rarely taken into account!
• **Low confidence** on any intent → out-of-domain?
  • might work, but likely to fail (no explicit training for this)
• Out-of-domain data + **specific OOD intent**
  • adding OOD from a different dataset
    • problem: “out-of-domain” should be broad, not just some different domain
  • collecting out-of-domain data specifically
    • worker errors for in-domain
    • replies to specifically chosen irrelevant queries
• always need to ensure that they don’t match any intent randomly
• not so many instances needed (expected to be rare)

(Larson et al., 2019)
NLU as classification

• using DAs – treating them as a **set of semantic concepts**
  • concepts:
    • intent
    • slot-value pair
  • binary classification: is concept Y contained in utterance X?
  • independent for each concept

• consistency problems
  • conflicting intents (e.g. `affirm` + `negate`)
  • conflicting values (e.g. `kids-allowed=yes` + `kids-allowed=no`)
  • need to be solved externally, e.g. based on classifier confidence
NER + delexicalization

• Approach:
  1) **identify** slot values/named entities
  2) **delexicalize** = replace them with placeholders (indicating entity type)
     • or add the NE tags as more features for classification
• generally needed for NLU as classification
  • otherwise in-domain data is too sparse
     • this can vastly reduce the number of concepts to classify & classifiers
• NER is a problem on its own
  • but general-domain NER tools may need to be adapted
  • in-domain gazetteers, in-domain training data

What is the phone number for Golden Dragon?
What is the phone number for **<restaurant-name>**?

I’m looking for a Japanese restaurant in Notting Hill.
I’m looking for a **<food>** restaurant in **<area>**.

I need to leave after 12:00.
I need to leave after **<time>**.
leave_at -> **leave_at**
arrive_by -> **none**
Both can be **<time>**
NLU Classifier models

• note that data is usually scarce!

• **handcrafted / rules**
  • simple mapping: word/n-gram/regex match → concept
  • can work really well for a limited domain
  • no training data, no retraining needed (tweaking on the go)

• **linear classifiers**
  • logistic regression, SVM…
  • need handcrafted features

• **neural nets** (=our main focus today)
NN neural classifiers

- **intent** = **multi-class** (softmax)
- **slot** tagging = set of **binary classifiers** (logistic loss)
- using word embeddings (task-specific or pretrained)
  - no need for handcrafted features
  - still needs delexicalization (otherwise data too sparse)
- different architectures possible
  - bag-of-words feed-forward NN
  - RNN / CNN encoders + classification layers
  - attention-based

(Raffel & Ellis, 2016)
Slot filling as sequence tagging

• get slot values directly – no need for delexicalization
  • each word classified
  • classes = slots & **IOB format** (inside-outside-beginning)
  • slot values taken from the text (where a slot is tagged)
  • NER-like approach
• rules + classifiers still work
  • keywords/regexes found at specific position
  • apply classifier to each word in the sentence left-to-right
• linear classifiers are still an option

*I need a flight from Boston to New York tomorrow*

```
O O O O B-dept O B-arr I-arr B-date
```
Neural sequence tagging

- Basic neural architecture:
  RNN (LSTM/GRU) → softmax over hidden states
  - + some different model for intents (such as classification)

- Sequence tagging problem: overall consistency
  - slots found elsewhere in the sentence might influence what’s classified now
  - may suffer from label bias
    - trained on gold data – single RNN step only
    - during inference, cell state is influenced by previous steps – danger of cascading errors

- Solution: structured/sequence prediction
  - conditional random fields (CRF)
    - can run CRF over NN outputs

Joint Intent & Slots Model

• Same network for both tasks
• Bidirectional encoder
  • 2 encoders: left-to-right, right-to-left
  • “see everything before you start tagging”
• Decoder – tag word-by-word, inputs:
  • attention
  • input encoder hidden states (“aligned inputs”)
  • both
• Intent classification:
  softmax over last encoder state
  • + specific intent context vector \( c_{\text{intent}} \) (attention)

(Liu & Lane, 2016)
http://arxiv.org/abs/1609.01454
NN for Joint Intent & Slots

• Extended version: **use slot tagging results in intent classification**
  • Bidi encoder
  • Slots decoder with encoder states & attention
  • Intent decoder
    – attention over slots decoder states

• Training for both intent & slot detection improves results on ATIS flights data
  • this is multi-task training 😊
  • intent error lower (2% → 1.5%)
  • slot filling slightly better (F1 95.7% → 95.9%)

• Variant: treat **intent detection as slot tagging**
  • append <EOS> token & tag it with intent

(Liu & Lane, 2016)
http://arxiv.org/abs/1609.01454

(Hakkani-Tür et al, 2016)
https://doi.org/10.21437/Interspeech.2016-402
Joint intents & slots with contextual embeddings

• shared “word contextualization”
  • feed-forward – \( \sum \) word + trained position embeddings
  • CNNs
  • (Transformer-style) attention with relative position
    • trained relative position embeddings instead of Transformer fixed absolute position embedding
  • LSTM

• task-specific network parts
  • intent: weighted sum of contextualized embeddings + softmax
  • slots tagging:
    • independent – non-recurrent, depend only on current embedding: \( P(l_i|h_i) \)
    • label-recurrent – depend on past labels & current embedding: \( P(l_i|l_{1,...i-1}, h_i) \)
      • faster than word-recurrent

(Gupta et al., 2019)
http://arxiv.org/abs/1903.08268
Joint intents & slots w/context embeddings

- CNN > LSTM > attention > feed-forward
- CNNs are also faster than anything other than FF
- label-recurrent models mostly better than independent
- except intent classification (non-recurrent task) on 1 dataset

<table>
<thead>
<tr>
<th>Model</th>
<th>label recurrent</th>
<th>intent classif. accuracy</th>
<th>slot labelling F1</th>
<th>Inference ms/utterance</th>
<th>Epochs to converge</th>
<th>s/epoch</th>
<th># params</th>
</tr>
</thead>
<tbody>
<tr>
<td>FEED-FORWARD</td>
<td>No</td>
<td>98.56</td>
<td>75.35</td>
<td>88.72</td>
<td>1.82</td>
<td>83</td>
<td>2.52</td>
</tr>
<tr>
<td>FEED-FORWARD</td>
<td>Yes</td>
<td>98.54</td>
<td>97.46</td>
<td></td>
<td>1.82</td>
<td>83</td>
<td>2.52</td>
</tr>
<tr>
<td>CNN, 5KERNEL, 1L</td>
<td>No</td>
<td>98.56</td>
<td>98.40</td>
<td>94.11</td>
<td>0.82</td>
<td>23</td>
<td>1.90</td>
</tr>
<tr>
<td>CNN, 5KERNEL, 3L</td>
<td>No</td>
<td>99.04</td>
<td>92.21</td>
<td>96.68</td>
<td>1.37</td>
<td>55</td>
<td>2.16</td>
</tr>
<tr>
<td>CNN, 3KERNEL, 4L</td>
<td>No</td>
<td>98.81</td>
<td>91.65</td>
<td>96.75</td>
<td>1.28</td>
<td>57</td>
<td>2.24</td>
</tr>
<tr>
<td>CNN, 5KERNEL, 1L</td>
<td>Yes</td>
<td>98.85</td>
<td>93.12</td>
<td>96.39</td>
<td>2.13</td>
<td>51</td>
<td>2.77</td>
</tr>
<tr>
<td>CNN, 5KERNEL, 3L</td>
<td>Yes</td>
<td>99.10</td>
<td>96.22</td>
<td>96.95</td>
<td>2.68</td>
<td>59</td>
<td>3.34</td>
</tr>
<tr>
<td>CNN, 3KERNEL, 4L</td>
<td>Yes</td>
<td>98.96</td>
<td>93.71</td>
<td>96.95</td>
<td>2.60</td>
<td>53</td>
<td>3.43</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 1L, NO-POS</td>
<td>No</td>
<td>98.50</td>
<td>97.51</td>
<td>69.31</td>
<td>1.95</td>
<td>25</td>
<td>1.94</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 1L</td>
<td>No</td>
<td>98.53</td>
<td>97.74</td>
<td>93.22</td>
<td>4.75</td>
<td>117</td>
<td>4.34</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 3L</td>
<td>No</td>
<td>98.74</td>
<td>81.51</td>
<td>94.07</td>
<td>7.68</td>
<td>160</td>
<td>4.32</td>
</tr>
<tr>
<td>ATTN, 2HEAD, 3L</td>
<td>No</td>
<td>98.31</td>
<td>83.02</td>
<td>94.61</td>
<td>7.86</td>
<td>79</td>
<td>4.87</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 1L, NO-POS</td>
<td>Yes</td>
<td>98.63</td>
<td>97.68</td>
<td>88.60</td>
<td>3.24</td>
<td>60</td>
<td>2.66</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 1L</td>
<td>Yes</td>
<td>98.61</td>
<td>98.00</td>
<td>94.53</td>
<td>6.12</td>
<td>89</td>
<td>5.53</td>
</tr>
<tr>
<td>ATTN, 1HEAD, 3L</td>
<td>Yes</td>
<td>98.51</td>
<td>88.04</td>
<td>94.99</td>
<td>9.03</td>
<td>109</td>
<td>6.06</td>
</tr>
<tr>
<td>ATTN, 2HEAD, 3L</td>
<td>Yes</td>
<td>98.48</td>
<td>89.31</td>
<td>95.86</td>
<td>9.17</td>
<td>93</td>
<td>6.54</td>
</tr>
<tr>
<td>LSTM, 1L</td>
<td>No</td>
<td>98.82</td>
<td>98.34</td>
<td>97.28</td>
<td>2.65</td>
<td>45</td>
<td>2.91</td>
</tr>
<tr>
<td>LSTM, 2L</td>
<td>No</td>
<td>98.77</td>
<td>93.10</td>
<td>97.36</td>
<td>4.72</td>
<td>58</td>
<td>5.09</td>
</tr>
<tr>
<td>LSTM, 1L</td>
<td>Yes</td>
<td>98.68</td>
<td>93.83</td>
<td>97.37</td>
<td>3.98</td>
<td>54</td>
<td>4.62</td>
</tr>
<tr>
<td>LSTM, 2L</td>
<td>Yes</td>
<td>98.71</td>
<td>93.88</td>
<td>97.28</td>
<td>6.03</td>
<td>69</td>
<td>6.82</td>
</tr>
</tbody>
</table>

(Gupta et al., 2019)
http://arxiv.org/abs/1903.08268
Seq2seq-based NLU

• seq2seq with **copy mechanism** = pointer-generator net
  • normal seq2seq with attention – generate output tokens (softmax over vocabulary)
  • **pointer net**: select tokens from input (attention over input tokens)
  • prediction = **weighted combination** of

• can work with out-of-vocabulary
  • e.g. previously unseen restaurant names
  • (but IOB tagging can, too)

• generating slots/values + intent
  • it’s not slot tagging (doesn’t need alignment)
    • works for slots expressed implicitly or not as consecutive phrases
  • treats intent as another slot to generate

---

Can I bring my kids along to this restaurant?
I want a Chinese place with a takeaway option.

confirm(kids_friendly=yes)
inform(food=Chinese_takeaway)

DSTC2 results
BERT-based NLU

• slot tagging on top of pretrained BERT
  • standard **IOB approach**
  • just feed final hidden layers to **softmax over tags**
    • classify only at 1st subword in case of split words (don’t want tag changes mid-word)

• special start token tagged with intent

• optional CRF on top of the tagger
  • for global sequence optimization

<table>
<thead>
<tr>
<th>Models</th>
<th>Intent</th>
<th>Snips Slot</th>
<th>Sent</th>
<th>Intent</th>
<th>Slot</th>
<th>Sent</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN-LSTM (Hakkani-Tür et al., 2016)</td>
<td>96.9</td>
<td>87.3</td>
<td>73.2</td>
<td>92.6</td>
<td>94.3</td>
<td>80.7</td>
</tr>
<tr>
<td>Atten.-BiRNN (Liu andLane, 2016)</td>
<td>96.7</td>
<td>87.8</td>
<td>74.1</td>
<td>91.1</td>
<td>94.2</td>
<td>78.9</td>
</tr>
<tr>
<td>Slot-Gated (Goo et al., 2018)</td>
<td>97.0</td>
<td>88.8</td>
<td>75.5</td>
<td>94.1</td>
<td>95.2</td>
<td>82.6</td>
</tr>
<tr>
<td>Joint BERT</td>
<td><strong>98.6</strong></td>
<td><strong>97.0</strong></td>
<td><strong>92.8</strong></td>
<td><strong>97.5</strong></td>
<td><strong>96.1</strong></td>
<td><strong>88.2</strong></td>
</tr>
<tr>
<td>Joint BERT + CRF</td>
<td>98.4</td>
<td>96.7</td>
<td>92.6</td>
<td><strong>97.9</strong></td>
<td>96.0</td>
<td><strong>88.6</strong></td>
</tr>
</tbody>
</table>

slightly different numbers, most probably a reimplementation

(Chen et al., 2019)
http://arxiv.org/abs/1902.10909
Dialogue Pretrained Models

(Henderson et al., 2020)
http://arxiv.org/abs/1911.03688

- Pretraining on dialogue tasks can do better (& smaller) than BERT
- **ConveRT**: Transformer-based **dual encoder**
  - 2 Transformer encoders: context + response
  - optionally 3rd encoder with more context (concatenated turns)
  - feed forward + cosine similarity on top
- training objective: **response selection**
  - response that actually happened = 1
  - random response from another dialogue = 0
- trained on a large dialogue dataset (Reddit)
- can be used as a base to train models for:
  - **slot tagging** (top self-attention layer → CNN → CRF)
  - **intent classification** (top feed-forward → more feed-forward → softmax)
  - Transformer layers are fixed, not fine-tuned
  - works well for little training data (**few-shot**)
Regular Expressions & NNs for NLU

- Regexes as manually specified features
  - **binary**: any matching sentence (for intents) + any word in a matching phrase (for slots)
  - regexes meant to represent an intent/slot
  - combination at different levels
    1) “input”: aggregate word/sent + regex embeddings (at sentence level for intent, word level for slots)
    2) “network”: per-label supervised attentions (log loss for regex matches)
    3) “output”: alter final softmax (add weighted regex value)

- Good for limited amounts of data (few-shot)
  - works with 10-20 training examples per slot/intent
  - still improves a bit on full ATIS data

---


<table>
<thead>
<tr>
<th>Model</th>
<th>Intent Micro-F1</th>
<th>Intent Accuracy</th>
<th>Slot Micro-F1</th>
<th>Slot Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Liu&amp;Lane (2016)</td>
<td>- / 98.43</td>
<td>- / 95.98</td>
<td>- / 94.74</td>
<td>- / 95.47</td>
</tr>
<tr>
<td>no regex (BiLSTM)</td>
<td>92.50 / 98.77</td>
<td>85.01 / 95.47</td>
<td>86.7 / 95.55</td>
<td>86.34 / 95.42</td>
</tr>
<tr>
<td>(1) input</td>
<td>91.86 / 97.65</td>
<td>86.7 / 95.55</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(3) output</td>
<td>92.48 / 98.77</td>
<td>86.94 / 95.42</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(2) network</td>
<td><strong>96.20 / 98.99</strong></td>
<td><strong>85.44 / 95.27</strong></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
NLU as semantic parsing

- **transition-based parsing**
  - actions over input build semantic tree gradually
  - using stack:
    - create terminal node (+ select what kind)
    - create non-terminal node (+ select what kind)
    - reduce – pop node from stack
  - can parse into intent-slot-value shallow trees
  - found to improve cross-domain performance
    - multi-task learning/transfer learning (pretrain + tune)

(Damonte et al., 2019) http://arxiv.org/abs/1903.04521

(Dyer et al., 2015) http://arxiv.org/abs/1505.08075
Universal Intents

• typically DAs are domain-dependent

• ISO 24617-2 DA tagging **standard**
  • pretty complex: **multiple dimensions**
    • Task, Social, Feedback…
  • DA types (intents) under each dimension

• **Simpler approach** – non-hierarchical
  • **union** looking at different datasets
  • Mapping from datasets – manual/semi-automatic
    • mapping tuned on classifier performance
  • Intent tagging improved using multiple datasets/domains
    • generic intents only
  • Slots stay domain-specific

Unsupervised NLU

• **Clustering** intents & slots
• Features:
  • word embeddings
  • POS
  • word classes
  • topic modelling (biterm)
• Autoencoder to normalize # of dimensions for features
• Dynamic hierarchical clustering
  • decides # of clusters – stops if cluster distance exceeds threshold
• Slot clustering – word-level
  • over nouns, using intent clustering results

(Shi et al., 2018)
https://www.aclweb.org/anthology/D18-1072/
Unsupervised NLU with semantic frames

(Vojta’s work)

• Frame semantic parsing
  • Too general, not usable directly
  • Some frames redundant
  • Some frames overlap

• What about intents?
Unsupervised NLU with semantic frames

1. Frame semantic parser
2. Frame merging
3. Intent clustering
4. Candidate selection
5. Slot tagger training Corpus

Processes:
- Direction + Location
- Intent clustering
- Type: 1.86, 1.24
- Color: 0.05
Unsupervised NLU with semantic frames - selection

- Iterative process
- Frames merging
  - Syntactic dependencies
  - 2 similar slots
- Candidates ranking
  - Based on frame semantic parser output
  - Multiple scoring functions (coherence, TextRank)
Unsupervised NLU with semantic frames - tagging

- LSTM B-I-O tagger
- Tagger trained on data previously labeled with our selection method
- Set threshold to improve recall
<table>
<thead>
<tr>
<th>method</th>
<th>CamRest676</th>
<th>CarSLU</th>
<th>WOZ-hotel</th>
<th>WOZ-attraction</th>
<th>ATIS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tag-supervised*</td>
<td>0.778 ± .004</td>
<td>0.724 ± .003</td>
<td>0.742 ± .008</td>
<td>0.731 ± .002</td>
<td>0.848 ± .003</td>
</tr>
<tr>
<td>Dict-supervised*</td>
<td>0.705 ± .005</td>
<td>0.753 ± .005</td>
<td>0.750 ± .018</td>
<td>0.665 ± .003</td>
<td>0.678 ± .002</td>
</tr>
<tr>
<td>Chen et al.</td>
<td>0.535 ± .002</td>
<td>0.590 ± .001</td>
<td>0.382 ± .001</td>
<td>0.375 ± .001</td>
<td>0.616 ± .001</td>
</tr>
<tr>
<td>Ours-nocl</td>
<td>0.311 ± .006</td>
<td>0.393 ± .011</td>
<td>0.122 ± .001</td>
<td>0.266 ± .008</td>
<td>0.631 ± .002</td>
</tr>
<tr>
<td>Ours-pars</td>
<td>0.552 ± .008</td>
<td>0.664 ± .007</td>
<td>0.388 ± .002</td>
<td>0.383 ± .002</td>
<td>0.627 ± .002</td>
</tr>
<tr>
<td>Ours-nothr</td>
<td>0.586 ± .024</td>
<td>0.569 ± .031</td>
<td>0.485 ± .032</td>
<td>0.435 ± .002</td>
<td>0.671 ± .005</td>
</tr>
<tr>
<td>Ours-full</td>
<td><strong>0.665 ± .012</strong></td>
<td><strong>0.692 ± .008</strong></td>
<td><strong>0.548 ± .004</strong></td>
<td><strong>0.439 ± .001</strong></td>
<td><strong>0.678 ± .002</strong></td>
</tr>
</tbody>
</table>
Unsupervised NLU - drawbacks

• How to estimate the output quality?
• How to use the inducted slots?
  • What do they represent?
  • How to align with the DB?
Summary

• NLU is mostly **intent classification + slot tagging**
• **Rules + simple methods work well** with limited domains
• Neural NLU:
  • various architectures possible: CNN, LSTM, attention, seq2seq + pointer nets
  • slot tagging: **sequence prediction – label bias**
  • it helps to do joint intent + slots
  • BERT et al. can help, but these models are huge & expensive
    • there are specific pretrained dialogue models, too
  • NNs can be combined with regexes/handcrafted features
    • helps with limited data
• Experimental/alternative neural NLU:
  • using parsing (syntactic, semantic)
  • **unsupervised approaches**
Contact us:
https://ufaldsg.slack.com/
{odusek,hudecek}@ufal.mff.cuni.cz
Skype/Meet/Zoom (by agreement)

Get the slides here:
http://ufal.cz/npfl099

References/Inspiration/Further:

• mostly papers referenced from slides
• Milica Gašić’s slides (Cambridge University): http://mi.eng.cam.ac.uk/~mg436/teaching.html
• Raymond Mooney’s slides (University of Texas Austin): https://www.cs.utexas.edu/~mooney/ir-course/
• Hao Fang’s slides (University of Washington): https://hao-fang.github.io/ee596_spr2018/syllabus.html
• Gokhan Tur & Renato De Mori (2011): Spoken Language Understanding

Labs in 10 minutes
Dialmonkey Framework

Next Tue 9:50am:
• State Tracking
• Lab Projects