Dialogue management: integrated approaches to understanding and tracking

Milica Gašić

Dialogue Systems and Machine Learning Group, Heinrich Heine University Düsseldorf
Hybrid approach to tracking and understanding

Delexicalisation

Word-vector embeddings

Self-attention & transformer architecture in tracking
Limitations of modular approach to dialogue systems

- Modular approaches suffer from information loss between the components.
- Labeled data not always available to train individual modules.
Hybrid approach

- Dialogue act output of NLU module is an intermediate designer-defined step.
- We could directly predict dialogue state or dialogue belief state.
- We then do not need dialogue act labels for the user input.
Alternative dialogue system architecture
Integrated approaches to semantic decoding and belief tracking [Henderson et al., 2014]

- Instead of extracting features from semantic decoding hypotheses extract features from ASR hypotheses
- Apply the same neural network structure
- Avoids information loss resulting from compact semantic representation of traditional approach
- Output: distribution over slot-value pairs
Feature extraction from ASR hypotheses

- For limited vocabulary dialogue system possible to extract N-gram features from ASR
- In order to deal with data sparsity need to delexicalise input
- Unlike for semantic decoding output, here it is not obvious which word corresponds to which slot and value
- Semantic dictionary is therefore needed to define possible values
## Results from dialogue state tracking challenge

<table>
<thead>
<tr>
<th></th>
<th>Goals</th>
<th>Method</th>
<th>Requested</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Acc.</td>
<td>L2</td>
<td>Acc.</td>
</tr>
<tr>
<td>SD features</td>
<td>0.742</td>
<td>0.387</td>
<td>0.922</td>
</tr>
<tr>
<td>ASR features</td>
<td>0.768</td>
<td>0.346</td>
<td>0.940</td>
</tr>
</tbody>
</table>
Delexicalisation - elephant in the room

- Most of the performance gain comes from delexicalised features.
- This requires a separate semantic dictionary which for all values from ontology defines their possible realisations, for example expensive → luxurious, upmarket, pricey.
- In real systems this poses a major problem.
Understanding the context

- Speech recognition performs extremely well in noise-free conditions for a high-resource language.
- Still personal assistants even in such circumstances do not perform well.
- Their understanding of context is not adequate!
Video
Hybrid approach
Word-vector embeddings

- Instead of 1-hot feature vectors, delexicalised features, or n-gram features, each word is represented by a dense vector.
- Semantically similar words are represented by vectors that are close to each other in the vector space.
What does semantic similarity mean for dialogue modelling?

- I would like something in the **north** part of town.
- I would like something in the **south** part of town.
- How close are embeddings for **north** and **south**?
Attract-repel algorithm [Mrkšić et al., 2016]

- Start from a given static word embedding
- Modify the word embeddings iteratively
  - **Attract** reduce the distance of synonyms
  - **Repel** increase the distance of antonyms

while keeping the distance between any other words the same
Static vs contextual word embeddings

- Contextual word embeddings have the potential to model context better.
- This is achieved through transformer framework.
An attention network maintains a set of hidden representations that scale with the size of the source.

The model uses an internal inference step to perform a soft selection over these representations.

\[
\begin{align*}
\mathbf{x} &= (x_1, \ldots, x_n) \quad \text{input sequence} \\
q &= \text{query} \\
z &\sim p(z|x, q) \quad \text{attention distribution with } \mathbf{x} \text{ as keys} \\
f(\mathbf{x}, z) &= \text{attention function with } \mathbf{x} \text{ as values} \\
c &= E_{p(z|x, q)} f(\mathbf{x}, z) \quad \text{context}
\end{align*}
\]
Example of attention network in translation

- Translation task with encoder-decoder architecture
- \( x \) is the sequence of hidden states of encoder
- \( q \) is the (current) hidden state of the decoder
- \( p(z|x, q) \) modelled as a neural network with softmax output
- \( f(x, z) = x_z \) selected hidden state to attend to during translation
Example of attention network in question answering

- $x$ is the sequence of facts
- $q$ is the question
Theory: Self-attention

- Attention relates input to output in order to determine which part of input should be used as context to output.
- Self-attention relates different parts of the input sequence to produce a better representation for that sequence.
Theory: Transformer

- Deploys encoder-decoder architecture
- Relies solely on attention (incorporates neither recurrent nor convolutional connections)
Theory: Dot-product attention

\[ q = x_1 W_Q \]
\[ k_1 = x_1 W_K \]
\[ v_1 = x_1 W_V \]

\[ k_2 = x_2 W_K \]
\[ v_2 = x_2 W_V \]

\[ k_3 = x_3 W_K \]
\[ v_3 = x_3 W_V \]

\[ \text{softmax}(qK^T) \]

\[ \text{softmax}(qK^T) V \]

\[ \text{output} \]

Theory: Multi-head attention

- Instead of performing a single attention function, we project queries, keys and values $h$ times

\[
Attention(Q, K, V) = \text{softmax} \left( \frac{QK^T}{\sqrt{d_k}} \right) V
\]

\[
head_i = Attention(QW_Q^i, KW_k^i, VW_V^i)
\]

\[
MultiHead(Q, K, V) = \text{Concat}(head_1, \ldots, head_h)W^0
\]
Theory: Attention in transformer

**Encoder** keys, values and queries come from the output of the previous layer. Each position in the encoder can attend to all positions in the previous layer.

**Encoder-Decoder** queries come from previous decoder layer and the keys and values come from the output of the encoder.

**Decoder** keys, values and queries come from the output of the previous layer BUT in order to preserve autoregressive property all connections going from right to left are masked.
Theory: Positional encoding

- Dot-product attention does not incorporate any information about the order of words
- In order to mitigate this issue we utilise positional encoding with following properties:
  - unique and deterministic encoding $e_t$ for each position $t$
  - distance between any two positions consistent $e_{t+k} = L_k e_t$
  - the values should be bounded
Theory: Positional encoding

- By choosing

\[ e_t(2i) = \sin \left( \frac{t}{10000} \right) \]
\[ e_t(2i + 1) = \cos \left( \frac{t}{10000} \right) \]

- \( L_k \) in this case is a block diagonal matrix consisting of rotation matrices that do not depend on \( t \) but only on \( k \) and \( d \).
Theory: Transformer architecture [Vaswani et al., 2017]
Computational complexity

<table>
<thead>
<tr>
<th></th>
<th>Complexity per layer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Self-attention</td>
<td>$O(n^2d)$</td>
</tr>
<tr>
<td>Recurrent</td>
<td>$O(nd^2)$</td>
</tr>
</tbody>
</table>

- $n$ sequence length
- $d$ representation dimension

Self attention could be restricted to consider only a neighbourhood of size $r$ in the input sequence centered around the respective output distribution. Then computational complexity is $O(rnd)$. 
Application of transformers in dialogue systems

**Encoder** Represent words via BERT

**Structure** Utilise self-attention in the system’s structure
In order to avoid delexicalisation we need a way to calculate the similarity between slots and values and the input. Multi-head attention can be used in this respect.
SUMBT: Slot-Utterance Matching for Universal and Scalable Belief Tracking [Lee et al., 2019]
TripPy: A Triple Copy Strategy for Value Independent Neural Dialog State Tracking [Heck et al., 2020]

Input:
- last user utterance
- last system utterance
- dialogue history (as is)

Dialogue state copy mechanisms: slot-value of the dialogue is
- mentioned by the user
- mentioned by the system
- referred to in the history from another slot

Span prediction: for value independence
- slot-value is directly extracted from the input
Evaluation

- MultiWOZ dataset collected via Amazon MTurk portal where humans take roles of user and system — *Wizard of Oz* set-up
- Contains more than 10K dialogues spanning multiple domains

<table>
<thead>
<tr>
<th></th>
<th>Joint goal acc.</th>
</tr>
</thead>
<tbody>
<tr>
<td>MDBT(^1)</td>
<td>15%</td>
</tr>
<tr>
<td>GCE(^2)</td>
<td>36%</td>
</tr>
<tr>
<td>SUMBT(^3)</td>
<td>46%</td>
</tr>
<tr>
<td>TripPy(^4)</td>
<td>55%</td>
</tr>
</tbody>
</table>

- \(^1\)[Ramadan et al., 2018]
- \(^2\)[Nouri and Hosseini-Asl, 2018]
- \(^3\)[Lee et al., 2019]
- \(^4\)[Heck et al., 2020]
Summary

- To avoid the information loss and the need for intermediate labels the process of understanding and tracking can be integrated.
- The biggest gains come from delexicalisation and this necessitates semantic dictionaries, which in practice is undesirable.
- To avoid the need for delexicalisation word-vector embeddings can be used.
- Static embeddings can be modified to be more suited for dialogue.
- Contextualised embeddings however are particularly useful for tracking.
References 1


