Statistical Natural Language Generation

Tsung-Hsien Wen, Milica Gasic

Dialogue Systems Group, Cambridge University Engineering Department

February 25, 2016
Outline

• Evaluation Metrics

• Traditional Approaches
  – Template-based
  – Tree-based

• Language Modeling for NLG
  – Class-based language model
  – Phrased-based Dynamic Bayesian Network

• Long Short-term Memory for NLG
  – Vanishing gradient problem and LSTM
  – Semantically conditioned LSTM for NLG
I'm looking for a restaurant
inform(type=restaurant)

What kind of food do you have in mind?
request(food)
Evaluating NLG

• What makes a generator a good generator?

• Aspects: [Stent et al, 2005]
  
  – Adequacy : Correct meaning
  
  – Fluency : Linguistic fluency
  
  – Readability : Fluency in the dialogue context
  
  – Variation : Multiple realisations for the same concept

• However, none of the above is trivial!
**BLEU score [Papineni et al, 2002]**

- Evaluating *similarity* between paired sentences (n-gram match).
- The gap between human perception and automatic metrics.

<table>
<thead>
<tr>
<th>Correlation</th>
<th>Adequacy</th>
<th>Fluency</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLEU</td>
<td>0.388</td>
<td>-0.492</td>
</tr>
</tbody>
</table>

[Stent et al, 2005]

- Real user trial is always the best way to evaluate NLG.
Template-based NLG

- Define a set of rules to map semantics to utterances.

- Pros:
  - simple, error-free (usually), easy-control

- Cons:
  - time-consuming, rigid, not scalable

confirm()  “Please tell me more about the product your are looking for.”
confirm(area=$V)  “Do you want somewhere in the $V?”
confirm(food=$V)  “Do you want a $V restaurant?”
confirm(food=$V,area=$W)  “Do you want a $V restaurant in the $W.”

...
Trainable generator [Walker et al, 2002]

• Divide the problem into a pipeline,

• Apply machine learning to sentence plan ranker.

Inform(
  name=Z_House,
  price=cheap
)

Z House is a cheap restaurant.
Sentence Plan Generator [Walker et al, 2002]

• Text plan (Dialogue Act):
  implicit-confirm(orig-city:NEWARK)
  implicit-confirm(dest-city:DALLAS)
  implicit-confirm(month:9)
  implicit-confirm(day-number:1)
  request(depart-time)

• Example sentence plan:

```
  soft-merge-general
    1
  soft-merge
    2
  imp-confirm(day)
  imp-confirm(month)
  request(time)
  soft-merge-general
    3
  imp-confirm(dest-city)
  imp-confirm(orig-city)

  period
    1
  period
    2
  imp-confirm(month)
  imp-confirm(day)
  soft-merge-general
    4
  request(time)
  imp-confirm(orig-city)
  imp-confirm(dest-city)
```
Sentence Plan Ranker [Walker et al, 2002]

• Frame it as an ML problem using RankBoost [Freund et al, 1998]

• Extracting features from trees using indicator function $f_i$,
  - Traversal features, ancestor features, leaf features, … etc. size 3291.
  
  $F(x) = \sum_i \alpha_i f_i(x)$
  $loss = \sum_{x,y \in D} e^{-(F(x) - F(y))}$
  assume $x$ is preferred than $y$
  
  – $\alpha_i$ are parameters to learn.
  – $x,y$ are sp-trees labeled with user preference.
  – $D$ is the set of sp-trees regarding to that text plan (DA).
Other similar approaches

- Learning sentence planning generation rules. [Stent et al, 2009]
- Statistical surface realisers. [Dethlefs et al, 2013]

**Pros:**
- Can generate sentences with complex linguistic structures.

**Cons:**
- Many rules, heavily engineered.
Class-based LM for NLG [Oh&Rudnicky, 2000]

- Language Modeling

\[ P(W) = \prod_{t} P(w_t|w_0, w_1, \ldots, w_{t-1}) \]

- Class-based LM

\[ P(W|u) = \prod_{t} P(w_t|w_0, w_1, \ldots, w_{t-1}, u) \]

- Decoding

\[ W^* = \operatorname{argmax}_W P(W|u) \]

Classes:
- inform_area
- inform_address
- inform_phone
- request_area
- request_postcode
Class-based LM for NLG [Oh&Rudnicky, 2000]

- Generation process
  - Generate utterances by sampling words from a particular class language model in which the dialogue act belongs to.
  - Re-rank utterances according to scores.
- Pros: no complicated rules, easy to implement, easy to understand.
- Cons: inefficient, error-prone
Phrase-based NLG [Mairesse et al, 2010]

- Phrase-based generation using Dynamic Bayesian Network (DBN)

```
Charlie Chan is a Chinese Restaurant near Cineworld in the centre
```

Inform(type= restaurant, name=Charlie Chan, food=chinese, near=Cineworld, area=centre)
Phrase-based NLG [Mairesse et al, 2010]

• Pros:
  – Computationally more efficient.
  – Better performance

• Cons:
  – A lot of effort involved in data collection: semantic alignments
Can we do better?

• RNN as language generator
  – Natural model for modeling sequences
  – Long-term dependencies
  – Flexible to conditioned on auxiliary inputs

• Long-term dependencies in NLG?
  – Example: The restaurant (in the north) is a nice Chinese place.
RNN & Vanishing gradient [Pascanu et al,2013]

\[ h_j = \sigma(W_r h_{j-1} + W_i w_j + b_h) \]

\[ y_j = \text{softmax}(W_o h_j + b_o) \]

\[
\frac{\partial E_3}{\partial W_r} = \sum_{k=0}^{3} \frac{\partial E_3}{\partial y_3} \frac{\partial y_3}{\partial h_3} \frac{\partial h_3}{\partial h_k} \frac{\partial h_k}{\partial W_r} \\
= \sum_{k=0}^{3} \frac{\partial E_3}{\partial y_3} \frac{\partial y_3}{\partial h_3} \left( \prod_{j=k+1}^{3} \frac{\partial h_j}{\partial h_{j-1}} \right) \frac{\partial h_k}{\partial W_r}
\]

\[
\frac{\partial h_j}{\partial h_{j-1}} = W_r^T \text{diag}(\sigma'(x_j))
\]

\[ x_j = W_r h_{j-1} + W_i w_j + b_h \]

Ignore proof here.
\[ \|W_r\| \cdot \|\text{diag}(\sigma'(x_j))\| < 1 \]
Vanishing gradient!
Long Short-term Memory
[Hochreiter and Schmidhuber, 1997]

- Sigmoid gates
  \[ i_t = g(W_{wi}w_t + W_{hi}h_{t-1}) \]
  \[ f_t = g(W_{wf}w_t + W_{hf}h_{t-1}) \]
  \[ o_t = g(W_{wo}w_t + W_{ho}h_{t-1}) \]

- Proposed cell value
  \[ \hat{C}_t = tanh(W_{wc}w_t + W_{hc}h_{t-1}) \]

- Update cell and hidden layer
  \[ C_t = i_t \odot \hat{C}_t + f_t \odot C_{t-1} \]
  \[ h_t = o_t \odot tanh(C_t) \]
Long Short-term Memory [Hochreiter and Schmidhuber, 1997]

• How it prevents vanishing gradient problem?
  
  – Consider memory cell, where recurrence actually happens

\[
C_t = i_t \odot \hat{C}_t + f_t \odot C_{t-1}
\]

  – We can back-propagate the gradient by chain rule.

\[
\frac{\partial E_t}{\partial C_{t-1}} = \frac{\partial E_t}{\partial C_t} \frac{\partial C_t}{\partial C_{t-1}} = \frac{\partial E_t}{\partial C_t} f_t
\]

  – If \( f_t \) maintains a value of 1, gradient is perfectly propagated.
RNN Language Model for NLG [Wen et al, 2015a]

Inform(name=Seven_Days, food=Chinese)

```
[ 0, 0, 1, 0, 0, ..., 1, 0, 0, ..., 1, 0, 0, 0, 0... ]
```

dialog act 1-hot representation

```
SLOT_NAME serves SLOT_FOOD .
```

delexicalisation
Inform(name=Seven_Days, food=Chinese)
Learned alignments
Results

![Bar chart showing BLEU scores for classlm and sc-lstm models for Restaurant and Hotel domains.]

<table>
<thead>
<tr>
<th>Human Evaluation</th>
<th>Informativeness</th>
<th>Naturalness</th>
</tr>
</thead>
<tbody>
<tr>
<td>sc-lstm</td>
<td>2.59</td>
<td>2.50</td>
</tr>
<tr>
<td>classlm</td>
<td>2.46**</td>
<td>2.45</td>
</tr>
</tbody>
</table>

* $p < 0.05$  ** $p < 0.005$
inform_no_match(area=tenderloin)

there are no restaurants in the tenderloin area.
there are 0 restaurants in the tenderloin area.
unfortunately there are 0 restaurants in the tenderloin area.
i could not find any restaurants in tenderloin.
Conclusion

• Evaluating NLG is hard. The best way is human evaluation.

• Tree-based NLG is a highly linguistically motivated approach. By introducing machine learning in the pipeline enables the model to learn from data.

• Language Modeling casts NLG as a sequential prediction problem. Both word-based and phrase-based approaches were introduced.

• RNN is a sequential model that can theoretically model very long-term dependencies, but in practice it suffers from the vanishing gradient problem.

• LSTM overcomes vanishing gradient by sophisticated gating mechanism. The same idea was applied to NLG resulting in semantically conditioned-LSTM, a generator that can learn realisation and semantic alignments jointly.
References


