Natural Language Generation for Dialog Systems

Presenters:
Thamme Gowda  isi.edu/~tg
Jacob Antony
NLG in a Dialog System
**NLG for Dialog Systems**

- **Input**: Dialog Manager’s Intent
- **Output**: Text (maybe to the TTS engine)
- **Methods (in general)**:
  - Fixed text
  - Template based
  - **Sentence Planning & Realization** *(we focus on this in this session)*
  - Grammar based (CFG)
    - Probabilistic Grammar Based (PCFG)
  - Statistical language model
    - Unigram, Bigram, Trigram, ... n-gram models [KenLM, BerkeleyLM]
    - Recurrent Neural Network (RNN) such as LSTM, GRU
‘Other’ Applications of NLG

● Autocomplete → Smart Compose Demo
● Machine Translation
● Summary Generation
● Question Answering
● Creative Language Generation
  ○ Poetry Generation
  ○ Story Writing
  ○ Movie Script Writing → Sunspring (2016) Demo

● ....

Note: Dialog System requires real-time NLG with context management
Goal: “generating language that varies along personality dimensions”

Lexical Hypothesis and its Two Postulates:
1. Personality traits that are important to a group of people will become part of group’s language.
2. More important traits are more likely to be encoded into the language as a single word.

⇒ Take away: Correlation between linguistic variables and personality traits
Personality Traits: ‘Big Five’ Model

**Five dimensions:** O.C.E.A.N.

1. Openness to Experiences
2. Conscientiousness
3. Extraversion
4. Agreeableness
5. Neuroticism

## Ten-Item Personality Inventory (TIPI)

<table>
<thead>
<tr>
<th>Disagree strongly</th>
<th>Disagree moderately</th>
<th>Disagree a little</th>
<th>Neither Agree nor Disagree</th>
<th>Agree a little</th>
<th>Agree moderately</th>
<th>Agree strongly</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>#</th>
<th>Rating</th>
<th>Traits</th>
<th>#</th>
<th>Rating</th>
<th>Traits</th>
<th>Dimension</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>____</td>
<td>Extraverted, enthusiastic</td>
<td>6R</td>
<td>____</td>
<td>Reserved, quiet</td>
<td>Extraversion</td>
</tr>
<tr>
<td>2R</td>
<td>____</td>
<td>Critical, quarrelsone</td>
<td>7</td>
<td>____</td>
<td>Sympathetic, warm</td>
<td>Agreeableness</td>
</tr>
<tr>
<td>3</td>
<td>____</td>
<td>Dependable, self-disciplined</td>
<td>8R</td>
<td>____</td>
<td>Disorganized, careless</td>
<td>Conscientiousness</td>
</tr>
<tr>
<td>4R</td>
<td>____</td>
<td>Anxious, easily upset</td>
<td>9</td>
<td>____</td>
<td>Calm, emotionally stable</td>
<td>Neuroticism</td>
</tr>
<tr>
<td>5</td>
<td>____</td>
<td>Open to new experiences, complex</td>
<td>10R</td>
<td>____</td>
<td>Conventional, uncreative</td>
<td>Openness</td>
</tr>
</tbody>
</table>

(Gosling et al., 2003)

‘R’ denotes reverse scored item
PERSONAGE: Personality Generation for Dialogue

François Mairesse and Marilyn Walker
Dept of Computer Science, University of Sheffield, UK

In Proceedings of the 45th Annual Meeting of
the Association for Computational Linguistics (ACL), Prague, June 2007.

- Focus on ‘Extraversion’ personality trait from Big-Five Model
- Parameter settings suggested by the psychology literature
- Built and evaluated NLG system to produce Restaurant Recommendation

Note:
→ Forward: understanding the language to detect personality traits (in psycholinguistic literature)
← Reverse: using personality traits to influence language generation (in this NLG work)
Challenges: Mapping Psychology Params to NLG

1. Studies (found in lit.) are in the domains such as consciousness essays, informal conversations
   ○ may not apply to the domain of NLG system (eg: restaurant rec sys).
2. Findings are based on self-reports of personality
   ○ but we want to affect observer’s perceptions.
3. Findings consist of weak but significant correlations
   ○ individual params may not have a strong enough effect to produce recognizable variation within a single utterance
4. Many possible mappings of the findings to generation params
## Sneak Peek: Restaurant Recommender

<table>
<thead>
<tr>
<th>#</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>Err... it seems to me that Le Marais isn’t as bad as the others.</td>
</tr>
<tr>
<td>b</td>
<td>Right, I mean, Le Marais is the only restaurant that is any good.</td>
</tr>
<tr>
<td>c</td>
<td>I am sure you would like Le Marais, you know. The atmosphere is acceptable, the servers are nice and it’s a french, kosher and steak house place. Actually, the food is good, even if its price is 44 dollars.</td>
</tr>
<tr>
<td>d</td>
<td>Basically, actually, I am sure you would like Le Marais. It features friendly service and acceptable atmosphere and it’s a french, kosher and steak house place. Even if its price is 44 dollars, it just has really good food, nice food.</td>
</tr>
</tbody>
</table>
SPaRKy NLG architecture: Text Planning and Realization

Ehud Reiter, and Robert Dale (1997) on Building Applied NLG systems:

1. Text Planner
   - Content Determination
   - Discourse Planning

2. Sentence Planner
   - Sentence Aggregation
   - Lexicalization
   - Referring Expression Generation

3. Linguistic Realizer

This architecture is used by SPaRKy generator
Sentence Planning & Realization, Reiter and Dale (1997)

An NLG system has six sub-activities:

- Content Determination
- Discourse Planning
- Sentence Aggregation
- Lexicalization
- Referring Expression Generation
- Linguistic realization

These sub activities maybe implemented independently, or combined in few sub modules depending on NLG architecture.
# Mapping: Linguistic Variables to NLG Params

<table>
<thead>
<tr>
<th>NLG modules</th>
<th>Introvert findings</th>
<th>Extravert findings</th>
<th>Parameter</th>
<th>Intro</th>
<th>Extra</th>
</tr>
</thead>
</table>
| **Content selection and structure** | Single topic  
Strict selection  
Problem talk, dissatisfaction | Many topics  
Think out loud*  
Pleasure talk, agreement, compliment | **VERBOSITY**  
**RESTATEMENTS**  
**REPETITIONS**  
**CONTENT POLARITY**  
**REPETITIONS POLARITY**  
**CLAIM POLARITY**  
**CONCESSIONS**  
**CONCESSIONS POLARITY**  
**POLARISATION**  
**POSITIVE CONTENT FIRST** | low   | high  |
| **Syntactic templates selection** | Few self-references  
Elaborated constructions  
Many articles | Many self-references  
Simple constructions*  
Few articles | **SELF-REFERENCES**  
**CLAIM COMPLEXITY** | low   | high  |
| **Aggregation Operations** | Many words per sentence/clause  
Many unfilled pauses | Few words per sentence/clause  
Few unfilled pauses | **RELATIVE CLAUSES**  
**WITH CUE WORD**  
**CONJUNCTION**  
**PERIOD** | high  | low   |

- **Note**: This table is incomplete, refer to the original paper for full mapping.
- **Parameter** column is for the SPaRKy Generator (an NLG framework used).
Experiments and Evals

- Total of 240 utterances for 20 Content Plans
  - 2 using Introvert parameters (with 15% standard deviation) ⇒ 40
  - 2 using Extravert parameters (with 15% standard deviation) ⇒ 40
  - 8 using Random parameters ⇒ 160

- Rated by 3 Human Judges (experts); scale: 1 to 7; averaged
  - Extraversion rating
    - Two questions in the “Ten-Item Personality Inventory (TIPI)”
  - Naturalness rating
<table>
<thead>
<tr>
<th>Setting</th>
<th>Output</th>
<th>Judge</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intro</td>
<td>Err... it seems to me that Le Marais isn’t as bad as the others.</td>
<td>1.83</td>
</tr>
<tr>
<td>Intro</td>
<td>Right, I mean, Le Marais is the only restaurant that is any good.</td>
<td>2.83</td>
</tr>
<tr>
<td>Extra</td>
<td>I am sure you would like Le Marais, you know. The atmosphere is acceptable, the servers are nice and it’s a french, kosher and steak house place. Actually, the food is good, even if its price is 44 dollars.</td>
<td>6.00</td>
</tr>
<tr>
<td>Extra</td>
<td>Basically, actually, I am sure you would like Le Marais. It features friendly service and acceptable atmosphere and it’s a french, kosher and steak house place. Even if its price is 44 dollars, it just has really good food, nice food.</td>
<td>6.17</td>
</tr>
</tbody>
</table>

Judge’s Rating scale: 1 to 7; where 7 means extravert
# Average Ratings

<table>
<thead>
<tr>
<th>Rating</th>
<th>Introvert (40)</th>
<th>Extravert (40)</th>
<th>Random (160)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extraversion</td>
<td>2.96</td>
<td>5.98</td>
<td>5.02</td>
</tr>
<tr>
<td>Naturalness</td>
<td>4.93</td>
<td>5.78</td>
<td>4.51</td>
</tr>
</tbody>
</table>

Scale: 1 to 7
Statistical Model

- Each of 160 Randomly generated sentences had feature vector of parameters using and Judges rating
- Used Weka Toolbox to learn regression models
  - Loss function: Mean Absolute Error (smaller the better)
  - Models (Evaluation: K fold cross validation)
    - Baseline: 0.83
    - **Linear Regression**: 0.65
    - M5’ Model Tree: 0.66  (and visualized the tree)
    - SVM with Linear Kernel: 0.72
    - SVM with RBF Kernel: 0.70
- Future work: Use this model to refine the parameter mappings
Questions from Piazza

1. a. Is it sufficient for work in dialogue (and human-computer interactions) to typically use the "Big 5" to model personality, and not other psychology models, such as Myers Briggs?
   b. I also want to echo some of the other questions that asked about the "Big 5" as the go-to personality traits. There are a fair number of systems out there that could be incorporated into a dialogue generator, what is it about this one that makes it stand above the others?

2. The paper describes an approach to manually map psychological personality findings to the parameters of an NLG system. I was wondering if there has been work on learning linguistic characteristics automatically from data (labeled with personality types).

3. The paper cites another study that states that introverts tend to use more negations in speech. They incorporate this into their model by having introverted personality types say "it's not nasty" rather than "it's nice", as an example. What I'm uncertain about is how a statement like this can be definitely considered introverted, rather than simply a lukewarm view on the restaurant being discussed.

4. since personality can be a dimension affecting speaking behaviors, can we extend the approach to other dimensions, e.g. cultures, ages, careers, etc?
Evaluating NLG in Virtual Environments

Report on the Second NLG Challenge on Generating Instructions in Virtual Environments (GIVE-2)

Alexander Koller
Kristina Striegnitz
Andrew Gargett
Donna Byron

Justine Cassell
Robert Dale
Johanna Moore
Jon Oberlander

6th International Natural Language Generation Conference (INLG 2010)
Summary

- Outcomes of 2nd installment of Generating Instructions in Virtual Environments (GIVE-2), happened in 2009-10
  - GIVE-1 was held in 2008-09 (GIVE-2 is an improvement over this)
  - GIVE-2.5 was held in 2011-2012 (not touching this part)

- A shared task for NLG community
  - Game: Treasure hunt in virtual 3D world (online game)
  - Task: Guide human player by generating NL instructions, in real-time
  - 7 NLG systems were submitted and evaluated
  - 1825 games played over the Internet, between Feb - May 2010

Vision-and-Language Navigation:
Interpreting visually-grounded navigation instructions in real environments
Three ways to finish game:
1. Successful hunt
2. Trigger Alarm → lose
3. Cancel Game
3 Evaluation Worlds

World 1

World 2

World 3
NLG Systems

Seven systems were submitted from these teams:

- Dublin Institute of Technology ("D")
- Trinity College Dublin ("T")
- Universidad Complutense de Madrid ("M")
- University of Heidelberg ("H")
- Saarland University ("S")
- two systems from INRIA Grand-Est in Nancy ("NA" and "NM")

⇒ There was a matchmaker server which chose random NLG system and a random Evaluation World (i.e. 7 x 3 worlds = 21 choices)
⇒ **Objective and Subjective measures** were collected from all games
Objective measures

- **Task success**: Did the player get the trophy?
- **Duration**: Time in seconds from the end of the tutorial until the retrieval of the trophy.
- **Distance**: Distance traveled (measured in distance units of the virtual environment).
- **Actions**: Number of object manipulation actions.
- **Instructions**: Number of instructions produced by the NLG system.
- **Words per instruction**: Average number of words the NLG system used per instruction.
Subjective Measures

Q1: The system used words and phrases that were easy to understand.
Q2: I had to re-read instructions to understand what I needed to do.
Q3: The system gave me useful feedback about my progress.
Q4: I was confused about what to do next.
Q5: I was confused about which direction to go in.
Q6: I had no difficulty with identifying the objects the system described for me.
Q7: The system gave me a lot of unnecessary information.
Q8: The system gave me too much information all at once.
Q9: The system immediately offered help when I was in trouble.
Q10: The system sent instructions too late.
Q11: The system’s instructions were delivered too early.
Subjective Measures ...

Q12: The system’s instructions were visible long enough for me to read them.
Q13: The system’s instructions were clearly worded.
Q14: The system’s instructions sounded robotic.
Q15: The system’s instructions were repetitive.
Q16: I really wanted to find that trophy.
Q17: I lost track of time while solving the overall task.
Q18: I enjoyed solving the overall task.
Q19: Interacting with the system was really annoying.
Q20: I would recommend this game to a friend.
Q21: The system was very friendly.
Q22: I felt I could trust the system’s instructions.
Extra Stuff
Case Study 1: Google Duplex


Google Duplex’s conversations sound natural thanks to advances in understanding, interacting, timing, and speaking.

“At the core of Duplex is a recurrent neural network (RNN) ...”

Demo: https://www.youtube.com/embed/GoXp1leA5Qc
Case Study 2: Google Smart Compose


Subject and previous email message are encoded by averaging the word embeddings in each field. The averaged embeddings are then fed to the RNN-LM at each decoding step.

Demo: https://www.youtube.com/embed/M7pyXXGjko0?start=13
Deep Learning: Encoder-Decoder

- NLP is mainly NLU and NLG
  - NLU is an Encoder Module
    - \((x_0, x_1, \ldots, x_n; \theta_E) \to \mathbb{R}^d\)
  - NLG is a Decoder Module
    - \((\mathbb{R}^d; \theta_D) \to y_0, y_1, \ldots, y_m\)
- Parameters \(\theta_E\) and \(\theta_D\) are approximated using a corpus and an optimizer.
Word Embeddings

Word Embeddings are a way of mapping words into numerical vectors. These vectors capture the semantic and syntactic relationships between words. The goal is to learn a vector space where similar words are mapped to similar vectors, which can be used for various natural language processing tasks.

### Key Word Embedding Models

- **Word2Vec (2013)**
- **GloVe (2014)**
- **Subwords (2016)**
- **FastText (2016)**
- **Elmo (2018)**

#### Example Vectors

- **Male-Female:**
  - king
  - man
  - woman
  - queen

- **Verb Tense:**
  - walked
  - swimming
  - swam

- **Country-Capital:**
  - Spain
  - Italy
  - Germany
  - Turkey
  - Russia
  - Canada
  - Japan
  - Vietnam
  - China

---

Image Credits: Tensorflow

[https://www.tensorflow.org/tutorials/representation/word2vec](https://www.tensorflow.org/tutorials/representation/word2vec)
RNN/LSTM based NLU: Encoding

Image Credits: Guillaume Genthial
RNN/LSTM based NLG: Decoding

Image Credits: Guillaume Genthial
RNN/LSTM based NLG: Training

Image Credits: Guillaume Genthial
~END~
Adversarial Learning for Neural Dialogue Generation

Jiwei Li, Will Montro, et. al.
Stanford University


- Inspired by Turing test, i.e. "Can a human tell if it's talking to a machine?"
- Built an generative adversarial learning model
- Described a method of evaluating adversarial models
Turing Test

- Proposed in 1950 by Alan Turing
- Given a conversation in which a human and a machine are participants, can a blind evaluator identify which entity is human and which is a machine?
- Can a machine exhibit intelligent behavior (dialogue) that is indistinguishable from a human's?
Prior work on dialogue generation

- Phrase-based machine translation, end-to-end neural systems
- Predict next utterance conditioned on dialogue history using MLE.
- "Despite its success, this over-simplified training objective leads to problems: responses are dull, generic, repetitive, and short-sighted."
- ^Not "human" enough.
Adversarial Learning

D-dimensional noise vector

Real Images → Discriminator Network → Predicted Labels

Generator Network → Fake Images

![Diagram of adversarial learning process](image.png)
Adversarial learning

- Generated examples are evaluated by discriminator and classified as human or machine-generated
- Discriminator output is used as reward in reinforcement learning for generator
- Generator is pushed towards outputs indistinguishable from human dialogue
Dialogue Generation

- Seq2seq model
Adversarial training

● Generative:
  ○ Seq2seq model generating response conditioned on dialogue history

● Discriminative:
  ○ Takes output of generative network, feeds into hierarchical encoder and uses 2-way softmax classification.
  ○ Returns $P(\text{machine-generated})$ and $P(\text{human-generated})$

● Policy Gradient Training:
  ○ Generator must maximize expected reward of an utterance given $P(\text{human-generated})$ from discriminator as reward
  ○ REINFORCE, Reward for Every Generation Step (REGS)
REGS

- Input: "What's your name?"
- Human response: "I am John.", Machine response: "I don't know."
- REINFORCE assigns negative reward to all tokens in "I don't know" but should assign reward based on token
  - "I": neutral reward; "don't", "know": negative reward
- Generated responses are split into subsequences which are labeled positive and negative. Samples are randomly selected as updates for discriminator.
REGS Modifications

- Generator can get "stuck" producing bad examples if discriminator catches them and returns negative reward
  - I.e. it is never pushed toward "good" examples
- Teacher forcing
  - Feed generator human examples with reward of 1 to force it to generate better examples.
Adversarial Evaluation

- Computationally resembles Turing test
- Train machine evaluator to replace human evaluator (which is too expensive and time-consuming)
- Trained on labeled data, tested on holdout dataset
Adversarial Success

- Fraction of generated samples capable of fooling the evaluator
- If generated and human samples are indistinguishable, evaluator will have 50% accuracy (passes the Turing test!).
- \( \text{AdverSuc}: 1 - \text{(accuracy of evaluator on test data)} \)
- Evaluation reliability error (ERE): Test evaluator on 100% human data, 100% generated data, human data labeled as positive, human data labeled as negative. Measure deviation from expected scores for each case.
Adversarial Success

- Since machine generated responses are not included in previous evaluation tests, need a metric to ensure generator can't fool evaluator by introducing noise.
- Test evaluator with generated data and randomly sampled human responses --> *machine vs. random* test
- Sampling can result in high AdverSuc score but low machine vs. random accuracy.
## Results

**Adversarial Evaluation:**

<table>
<thead>
<tr>
<th>Setting</th>
<th>FPR</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVM+Unigram</td>
<td>0.232</td>
</tr>
<tr>
<td>Concat Neural</td>
<td>0.209</td>
</tr>
<tr>
<td><strong>Hierarchical Neural</strong></td>
<td>0.193</td>
</tr>
<tr>
<td>SVM+Neural+multil-features</td>
<td>0.152</td>
</tr>
</tbody>
</table>
Results

Models:

<table>
<thead>
<tr>
<th>Model</th>
<th>AdverSuc</th>
<th>machine-vs-random</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLE-BS</td>
<td>0.037</td>
<td>0.942</td>
</tr>
<tr>
<td>ML-F-Greedy</td>
<td>0.049</td>
<td>0.945</td>
</tr>
<tr>
<td>MMI+$p(t</td>
<td>s)$</td>
<td>0.073</td>
</tr>
<tr>
<td>MMI-$p(t)$</td>
<td>0.090</td>
<td>0.880</td>
</tr>
<tr>
<td>Sampling</td>
<td>0.372</td>
<td>0.679</td>
</tr>
<tr>
<td>Adver-Reinforce</td>
<td>0.080</td>
<td>0.945</td>
</tr>
<tr>
<td>Adver-REGS</td>
<td>0.098</td>
<td>0.952</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Setting</th>
<th>adver-win</th>
<th>adver-lose</th>
<th>tie</th>
</tr>
</thead>
<tbody>
<tr>
<td>single-turn</td>
<td>0.62</td>
<td>0.18</td>
<td>0.20</td>
</tr>
<tr>
<td>multi-turn</td>
<td>0.72</td>
<td>0.10</td>
<td>0.18</td>
</tr>
</tbody>
</table>

Table 3: AdverSuc and machine-vs-random scores achieved by different training/decoding strategies.

Table 4: The gain from the proposed adversarial model over the mutual information system based on pairwise human judgments.
Conclusion

Adversarial models are significantly better at fooling evaluators compared to baseline models (MLE, etc.) on multiple metrics.

Future work: Adversarial models work well when entropy of reference target sequences is high. Not much performance boost on machine translation tasks.
Making Grammar-Based Generation Easier to Deploy in Dialogue Systems

David Devault, David Traum, Ron Artstein
USC ICT


- Grammar based generation for implemented dialogue systems.
- Need to balance resource demands along with coverage of semantic meanings and real-time interaction.
- Developer provides training examples and system will construct a high quality, runtime grammar-based generation component for a given dialogue system.
Background

- Research developed in context of a program for creating virtual human agents for social training purposes
  - Can use non-verbal cues like eye gaze, gestures, facial expressions in combination with spoken language.
- System should be able to produce a variety of utterances based on internal state, beliefs, and goals.
Case Study: Dr. Perez

- Designed to teach negotiation skills to a trainee
- Effectiveness: Generator should give Dr. Perez an utterance within 200ms.
- Rich internal state: beliefs, goals, plans, and emotions
- Simulation of non-native English speech
  - Doesn't need to be perfect
  - Disfluency is desirable to increase realism of conversation
Case Study: Dr. Perez

- Utterance selection, dialogue management, language generation use attribute-value matrix.
- AVM maps utterance to set of core speech and dialogue acts.
  - States, actions, polarity, modality.
- Can be represented as frames corresponding to leaf node values of AVM tree.

Figure 2: An example of Doctor Perez’s representations for utterance semantics: Doctor Perez tells the captain that there are no medical supplies at the market.
Methods

Three steps to grammar-based generation:

1. Specification of Training Examples
2. Grammar Induction
3. Search Optimization

Automatically constructs resources needed for generation component of dialogue system.
Specification of Training Examples

- Training Examples
  - Target output utterance, syntax, substrings -> semantic representations
  - \((u, \text{syntax}(u), \text{semantics}(u))\)
- syntax\((u)\) derived from Penn Treebank format w/ Charniak parser
- semantics\((u)\) corresponds to set \(M = \{m_1, \ldots, m_j\}\) of key-value pairs in generation frames.
  - Allows for arbitrary n-gram chunks to be associated with various meanings at the discretion of the author.
Automatic Grammar Induction

- Probabilistic tree-adjoining grammar format
  - Trained on examples with assigned derivations of grammar
  - Subtrees are incrementally detached from training example syntax and reused.

Figure 5: The linguistic resources automatically inferred from the training example in Figure 3.
Search Optimization

• Given desired semantic representation:
  ○ Use induced grammar (subtrees) to incrementally construct utterances until specified timeout
• Beam search to rank potential grammatical expansions at each timestep using features and weights selected by search optimization designed by (Daume and Marcu, 2005)
• Boosting (Collins and Koo, 2005) for feature and weight selection
Evaluation

- Hand authored 220 utterances based on frames requested by Dr. Perez in previous dialogues with users.
- Split into 198/22 train-test set, trained generator and selected highest-ranked utterance for each example.
- Some examples timed out, but task was completed with 95% success on training and 80% success on test.
- Compared to sentence retrieval, which classifies example to highest ranking utterance based on relative entropy.
- Ranked by judges familiar with the task based on fluency and accuracy with respect to given frame.
Evaluation

- 494 utterances: hand-authored, generated, and sentence retrieval based
- Retriever selections mostly very bad or very good
- Generated selections have graded quality.

Figure 4: Observed ratings of generated (uncorrected syntax) vs. retrieved sentences for test examples.
Evaluation

- Though generator suffered in test coverage compared to sentence retriever (80% vs 90%), it was because it would only give an output to a frame that could be 100% covered
- Generator also produced multiple high quality outputs, which is advantageous for response variety.
Future Work

- Analyzing how growing size of training set affects performance
- Identifying specific weaknesses and disfluencies identified by human judges in generator output.
- Running generator within the larger scope of Dr. Perez