Natural Language Processing
with Deep Learning

Natural Language Generation
(with summarization, dialog, etc.)
Natural Language Generation (NLG)

Natural Language Generation refers to any setting in which we generate (i.e. write) new text.

NLG is a subcomponent of:

- Machine Translation
- (Abstractive) Summarization
- Dialogue (chit-chat and task-based)
- Creative writing: storytelling, poetry-generation
- Freeform Question Answering (i.e. answer is generated, not extracted from text or knowledge base)
- Image captioning
- ...
Recap

- **Language Modeling**: the task of predicting the next word, given the words so far

\[ P(y_t | y_1, ..., y_{t-1}) \]

- A system that produces this probability distribution is called a **Language Model**

- If that system is an RNN, it’s called a **RNN-LM**
Recap

- **Conditional Language Modeling**: the task of predicting the next word, given the words so far, and also some other input $x$.

  $$P(y_t|y_1,...,y_{t-1},x)$$

- Examples of conditional language modeling tasks:
  - Machine Translation ($x$=source sentence, $y$=target sentence)
  - Summarization ($x$=input text, $y$=summarized text)
  - Dialogue ($x$=dialogue history, $y$=next utterance)
  - ...
Recap: training a (conditional) RNN-LM

This example: Neural Machine Translation

During training, we feed the gold (aka reference) target sentence into the decoder, regardless of what the decoder predicts. This training method is called **Teacher Forcing**.

$$J = \frac{1}{T} \sum_{t=1}^{T} J_t$$

= negative log prob of “he”

= negative log prob of “with”

= negative log prob of <END>

Probability dist of next word
Recap: decoding algorithms

• **Question:** Once you’ve trained your (conditional) language model, how do you use it to generate text?

• **Answer:** A *decoding algorithm* is an algorithm you use to generate text from your language model.

• We’ve learnt about two decoding algorithms:
  • Greedy decoding
  • Beam search
Recap: beam search decoding

Beam size = \( k = 2 \). Blue numbers =

\[
\text{score}(y_1, \ldots, y_t) = \sum_{i=1}^{t} \log P_{LM}(y_i|y_1, \ldots, y_{i-1}, x)
\]
What’s the effect of changing beam size $k$?

- **Small $k$** has similar problems to greedy decoding ($k=1$)
  - Ungrammatical, unnatural, nonsensical, incorrect

- **Larger $k$** means you consider more hypotheses
  - Increasing $k$ reduces some of the problems above
  - Larger $k$ is more computationally expensive
  - But increasing $k$ can introduce other problems:
    - For NMT, increasing $k$ too much decreases BLEU score (Tu et al, Koehn et al). This is primarily because large-k beam search produces too-short translations (even with score normalization!)
    - In open-ended tasks like chit-chat dialogue, large $k$ can make output more generic (see next slide)
Effect of beam size in chitchat dialogue

I mostly eat a fresh and raw diet, so I save on groceries

<table>
<thead>
<tr>
<th>Beam size</th>
<th>Model response</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I love to eat healthy and eat healthy</td>
</tr>
<tr>
<td>2</td>
<td>That is a good thing to have</td>
</tr>
<tr>
<td>3</td>
<td>I am a nurse so I do not eat raw food</td>
</tr>
<tr>
<td>4</td>
<td>I am a nurse so I am a nurse</td>
</tr>
<tr>
<td>5</td>
<td>Do you have any hobbies?</td>
</tr>
<tr>
<td>6</td>
<td>What do you do for a living?</td>
</tr>
<tr>
<td>7</td>
<td>What do you do for a living?</td>
</tr>
<tr>
<td>8</td>
<td>What do you do for a living?</td>
</tr>
</tbody>
</table>

**Low beam size:**
More on-topic but nonsensical; bad English

**High beam size:**
Converges to safe, “correct” response, but it’s generic and less relevant
Sampling-based decoding

• Pure sampling
  • On each step $t$, randomly sample from the probability distribution $P_t$ to obtain your next word.
  • Like greedy decoding, but sample instead of argmax.

• Top-n sampling*
  • On each step $t$, randomly sample from $P_t$, restricted to just the top-n most probable words
  • Like pure sampling, but truncate the probability distribution
  • $n=1$ is greedy search, $n=V$ is pure sampling
  • Decrease $n$ to get more diverse/risky output
  • Increase $n$ to get more generic/safe output

*Usually called top-$k$ sampling, but here we’re avoiding confusion with beam size $k$
**Softmax temperature**

- **Recall:** On timestep $t$, the LM computes a prob dist $P_t$ by applying the softmax function to a vector of scores $s \in \mathbb{R}^{|V|}$
  \[
P_t(w) = \frac{\exp(s_w)}{\sum_{w' \in V} \exp(s_{w'})}
\]

- You can apply a *temperature hyperparameter* $\tau$ to the softmax:
  \[
P_t(w) = \frac{\exp(s_w/\tau)}{\sum_{w' \in V} \exp(s_{w'}/\tau)}
\]

- **Raise the temperature $\tau$:** $P_t$ becomes more uniform
  - Thus more diverse output (probability is spread around vocab)
- **Lower the temperature $\tau$:** $P_t$ becomes more spiky
  - Thus less diverse output (probability is concentrated on top words)

**Note:** *softmax temperature is not a decoding algorithm!*
It’s a technique you can apply at test time, in conjunction with a decoding algorithm (such as beam search or sampling)
Decoding algorithms: in summary

- **Greedy decoding** is a simple method; gives low quality output
- **Beam search** (especially with high beam size) searches for high-probability output
  - Delivers better quality than greedy, but if beam size is too high, can return high-probability but unsuitable output (e.g. generic, short)
- **Sampling methods** are a way to get more diversity and randomness
  - Good for open-ended / creative generation (poetry, stories)
  - Top-n sampling allows you to control diversity
- **Softmax temperature** is another way to control diversity
  - It’s not a decoding algorithm! It's a technique that can be applied alongside any decoding algorithm.
Summarization: task definition

Task: given input text $x$, write a summary $y$ which is shorter and contains the main information of $x$.

Summarization can be single-document or multi-document.

- Single-document means we write a summary $y$ of a single document $x$.

- Multi-document means we write a summary $y$ of multiple documents $x_1,\ldots,x_n$.

  Typically $x_1,\ldots,x_n$ have overlapping content: e.g. news articles about the same event.
Summarization: task definition

Within single-document summarization, there are datasets with source documents of different lengths and styles:

- **Gigaword**: first one or two sentences of a news article → headline (aka *sentence compression*)
- **LCSTS** (Chinese microblogging): paragraph → sentence summary
- **NYT, CNN/DailyMail**: news article → (multi)sentence summary
- **Wikihow** (*new!*): full how-to article → summary sentences

*Sentence simplification* is a different but related task: rewrite the source text in a simpler (sometimes shorter) way

- **Simple Wikipedia**: standard Wikipedia sentence → simple version
- **Newsela**: news article → version written for children

List of summarization datasets, papers, and codebases: [https://github.com/mathsyouth/awesome-text-summarization](https://github.com/mathsyouth/awesome-text-summarization)
Summarization: two main strategies

Extractive summarization

Select parts (typically sentences) of the original text to form a summary.

- Easier
- Restrictive (no paraphrasing)

Abstractive summarization

Generate new text using natural language generation techniques.

- More difficult
- More flexible (more human)
Pre-neural summarization systems were mostly extractive.

Like pre-neural MT, they typically had a pipeline:
- **Content selection**: choose some sentences to include
- **Information ordering**: choose an ordering of those sentences
- **Sentence realization**: edit the sequence of sentences (e.g. simplify, remove parts, fix continuity issues)

Diagram credit: *Speech and Language Processing*, Jurafsky and Martin
Pre-neural summarization

Pre-neural **content selection** algorithms:

- **Sentence scoring functions** can be based on:
  - Presence of topic keywords, computed via e.g. tf-idf
  - Features such as where the sentence appears in the document
- **Graph-based algorithms** view the document as a set of sentences (nodes), with edges between each sentence pair
  - Edge weight is proportional to sentence similarity
  - Use graph algorithms to identify sentences which are *central* in the graph

Figure 23.14  The basic architecture of a generic single document summarizer.

Diagram credit: *Speech and Language Processing*, Jurafsky and Martin
Summarization evaluation: ROUGE

ROUGE (Recall-Oriented Understudy for Gisting Evaluation)

Like BLEU, it’s based on **n-gram overlap**. Differences:

- **ROUGE has no brevity penalty**
- **ROUGE is based on recall**, while BLEU is based on **precision**
  - Arguably, precision is more important for MT (then add brevity penalty to fix under-translation), and recall is more important for summarization (assuming you have a max length constraint)
  - However, often a F1 (combination of precision and recall) version of ROUGE is reported anyway.

\[
\text{ROUGE-N} = \frac{\sum_{S \in \{\text{Reference Summaries}\}} \sum_{\text{gram}_n \in S} \text{Count}_{\text{match}}(\text{gram}_n)}{\sum_{S \in \{\text{Reference Summaries}\}} \sum_{\text{gram}_n \in S} \text{Count}(\text{gram}_n)}
\]

(1)

---

Summarization evaluation: ROUGE

- BLEU is reported as a single number, which is combination of the precisions for n=1,2,3,4 n-grams
- ROUGE scores are reported separately for each n-gram

The most commonly-reported ROUGE scores are:
  - ROUGE-1: unigram overlap
  - ROUGE-2: bigram overlap
  - ROUGE-L: longest common subsequence overlap

- There is now a convenient Python implementation of ROUGE!

Python implementation of ROUGE: https://github.com/google-research/google-research/tree/master/rouge
Neural summarization (2015 - present)

- 2015: Rush et al. publish the first seq2seq summarization paper
- Single-document abstractive summarization is a translation task!
- Thus we can apply standard seq2seq + attention NMT methods

Neural summarization (2015 - present)

• Since 2015, there have been lots more developments!
  • Making it easier to copy
    • But also preventing too much copying!
  • Hierarchical / multi-level attention
  • More global / high-level content selection
  • Using Reinforcement Learning to directly maximize ROUGE, or other discrete goals (e.g. length)
  • Resurrecting pre-neural ideas (e.g. graph algorithms for content selection) and working them into neural systems
  • ...

List of summarization datasets, papers, and codebases: https://github.com/mathsyouth/awesome-text-summarization

Neural summarization: copy mechanisms

- Seq2seq+attention systems are **good at writing fluent output**, but **bad at copying over details** (like rare words) correctly
- **Copy mechanisms** use attention to enable a seq2seq system to easily copy words and phrases from the input to the output
  - Clearly this is very useful for summarization
  - Allowing both copying and generating gives us a **hybrid extractive/abstractive approach**
- There are several papers proposing copy mechanism variants:
  - etc
Neural summarization: copy mechanisms

One example of how to do a copying mechanism:

On each decoder step, calculate $p_{gen}$, the probability of generating the next word (rather than copying it). The final distribution is a mixture of the generation (aka “vocabulary”) distribution, and the copying (i.e. attention) distribution:

$$P(w) = p_{gen}P_{vocab}(w) + (1 - p_{gen})\sum_{i:w_i=w}d_i^t$$

Neural summarization: copy mechanisms

• **Big problem** with copying mechanisms:
  • **They copy too much!**
    • Mostly long phrases, sometimes even whole sentences
  • What *should* be an abstractive system collapses to a mostly extractive system.

• Another problem:
  • They’re **bad at overall content selection**, especially if the input document is **long**
  • **No overall strategy** for selecting content
Neural summarization: better content selection

• Recall: pre-neural summarization had separate stages for content selection and surface realization (i.e. text generation)

• In a standard seq2seq+attention summarization system, these two stages are mixed in together
  • On each step of the decoder (i.e. surface realization), we do word-level content selection (attention)
  • This is bad: no global content selection strategy

• One solution: bottom-up summarization
Bottom-up summarization

- Content selection stage: Use a neural sequence-tagging model to tag words as *include* or *don’t-include*
- Bottom-up attention stage: The seq2seq+attention system can’t attend to words tagged *don’t-include* (apply a mask)

**Figure 2:** Overview of the selection and generation processes described throughout Section 4.

*Simple but effective!*

- Better overall content selection strategy
- Less copying of long sequences (i.e. more abstractive output)
Neural summarization via Reinforcement Learning

• In 2017 Paulus et al. published a “deep reinforced” summarization model
• Main idea: Use Reinforcement Learning (RL) to directly optimize ROUGE-L
  • By contrast, standard maximum likelihood (ML) training can’t directly optimize ROUGE-L because it’s a non-differentiable function
• Interesting finding:
  • Using RL instead of ML achieved higher ROUGE scores, but lower human judgment scores

<table>
<thead>
<tr>
<th>Model</th>
<th>ROUGE-1</th>
<th>ROUGE-2</th>
<th>ROUGE-L</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML, no intra-attention</td>
<td>44.26</td>
<td>27.43</td>
<td>40.41</td>
</tr>
<tr>
<td>ML, with intra-attention</td>
<td>43.86</td>
<td>27.10</td>
<td>40.11</td>
</tr>
<tr>
<td>RL, no intra-attention</td>
<td>47.22</td>
<td>30.51</td>
<td>43.27</td>
</tr>
<tr>
<td>ML+RL, no intra-attention</td>
<td>47.03</td>
<td>30.72</td>
<td>43.10</td>
</tr>
</tbody>
</table>

“"We observed that our models with the highest ROUGE scores also generated barely-readable summaries.”

<table>
<thead>
<tr>
<th>Model</th>
<th>Readability</th>
<th>Relevance</th>
</tr>
</thead>
<tbody>
<tr>
<td>ML</td>
<td>6.76</td>
<td>7.14</td>
</tr>
<tr>
<td>RL</td>
<td>4.18</td>
<td>6.32</td>
</tr>
<tr>
<td>ML+RL</td>
<td>7.04</td>
<td>7.45</td>
</tr>
</tbody>
</table>

Overall, a hybrid approach does best!

“Dialogue” encompasses a large variety of settings:

- **Task-oriented dialogue**
  - Assistive (e.g. customer service, giving recommendations, question answering, helping user accomplish a task like buying or booking something)
  - Co-operative (two agents solve a task together through dialogue)
  - Adversarial (two agents compete in a task through dialogue)

- **Social dialogue**
  - Chit-chat (for fun or company)
  - Therapy / mental wellbeing
Pre- and post-neural dialogue

• Due to the difficulty of open-ended freeform NLG, pre-neural dialogue systems more often used predefined templates, or retrieve an appropriate response from a corpus of responses.

• As in summarization research, since 2015 there have been many papers applying seq2seq methods to dialogue – thus leading to a renewed interest in open-ended freeform dialogue systems.

• Some early seq2seq dialogue papers include:
  • *A Neural Conversational Model*, Vinyals et al, 2015
  • *Neural Responding Machine for Short-Text Conversation*, Shang et al, 2015
    [https://www.aclweb.org/anthology/P15-1152](https://www.aclweb.org/anthology/P15-1152)

This is a nice overview of recent (mostly neural) conversational conversational AI work: [https://medium.com/gobeyond-ai/a-reading-list-and-mini-survey-of-conversational-ai-32fcee97180](https://medium.com/gobeyond-ai/a-reading-list-and-mini-survey-of-conversational-ai-32fcee97180)
Seq2seq-based dialogue

• However, it quickly became apparent that a naïve application of standard seq2seq+attention methods has serious pervasive deficiencies for (chitchat) dialogue:
  • Genericness / boring responses
  • Irrelevant responses (not sufficiently related to context)
  • Repetition
  • Lack of context (not remembering conversation history)
  • Lack of consistent persona
Irrelevant response problem

- **Problem**: seq2seq often generates response that’s unrelated to user’s utterance
  - Either because it’s generic (e.g. “I don’t know”)
  - Or because changing the subject to something unrelated

- **One solution**: optimize for Maximum Mutual Information (MMI) between input S and response T:

  \[
  \hat{T} = \arg \max_T \left\{ \log \frac{p(S, T)}{p(S)p(T)} \right\}
  \]

  \[
  \hat{T} = \arg \max_T \left\{ \log p(T|S) - \log p(T) \right\}
  \]
Genericness / boring response problem

• Easy test-time fixes:
  • Directly upweight rare words during beam search
  • Use a sampling decoding algorithm rather than beam search

• Conditioning fixes:
  • Condition the decoder on some additional content (e.g. sample some content words and attend to them)
  • Train a retrieve-and-refine model rather than a generate-from-scratch model
    • i.e. sample an utterance from your corpus of human-written utterances, and edit it to fit the current scenario.
    • This usually produces much more diverse / human-like / interesting utterances!

Why are Sequence-to-Sequence Models So Dull?, Jiang et al, 2018
Repetition problem

Simple solution:
• Directly block repeating n-grams during beam search.
  • Usually pretty effective!

More complex solutions:
• Train a coverage mechanism – in seq2seq, this is an objective that prevents the attention mechanism from attending to the same words multiple times.
• Define a training objective to discourage repetition
  • If this is a non-differentiable function of the generated output, then will need some technique like e.g. RL to train

*SGD vs RL
Lack of consistent persona problem

- In 2016, Li et al proposed a seq2seq dialogue model that learns to encode both conversation partners’ personas as embeddings
  - The generated utterances are conditioned on the embeddings

- More recently, there is now a chitchat dataset called PersonaChat, which includes personas (collections of 5 sentences describing personal traits) for every conversation.
  - This provides a light type of grounding, allowing researchers to build persona-conditional dialogue agents

Negotiation dialogue

In 2017, Lewis et al collected a negotiation dialogue dataset

- Two agents negotiate (via natural language) how to divide a set of items.
- The agents have different value functions for the items.
- The agents talk until they reach an agreement.

Negotiation dialogue

• They find that training seq2seq systems for the standard maximum likelihood (ML) objective yields fluent but strategically poor dialogue agents.

• Like the Paulus et al summarization paper, they use Reinforcement Learning to optimize for a discrete reward (with the agents playing against themselves during training)

• The RL goal-based objective is combined with the ML objective

• Potential pitfall: If the agents just optimize just the RL goal while playing against each other, they might diverge from English*

At test time, the model chooses between possible responses by computing *rollouts*: simulations of the rest of the conversation and the expected reward.
Negotiation dialogue

• In 2018, Yarats et al proposed another dialogue model for the negotiation task, that separates the strategic aspect from the NLG aspect.

• Each utterance $x_t$ has a corresponding discrete latent variable $z_t$.

• $z_t$ is learnt to be a good predictor of future events in the dialogue (future messages, ultimate strategic outcome), but not a predictor of $x_t$ itself.

• This means that “$z_t$ learns to represent $x_t$’s effect on the dialogue, but not the words of $x_t$.”

• Thus $z_t$ separates the strategic aspect of the task from the NLG aspect. This is useful for controllability, interpretability, easier to learn strategy, etc.
Negotiation dialogue

Figure 3: We pre-train a model to learn a discrete encoder for sentences, which bottlenecks the message $x_t$ through discrete representation $z_t$ (Figure 3a; §5). This architecture forces $z_t$ to capture the most relevant aspects of $x_t$ for predicting future messages and actions. We then extract the learned discrete representations $z_t$ (marked by orange ellipses) and train our full model (Figure 3b): $p_x$ is trained to translate representations $z_t^*$ into messages $x_t$ (§6.1), and $\hat{p}_z$ is trained to predict a distribution over $z_t$ given the dialogue history (§6.2).
Conversational question answering: CoQA

- A new dataset from Stanford NLP!
- **Task:** answer questions about a piece of text *within the context of a conversation*
- Answers must be written abstractively (i.e. not copied)
- Both a QA / reading-comprehension task, and a dialogue task

---

Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80. Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie’s husband Josh were coming as well. Jessica had . . .

**Q₁:** Who had a birthday?
**A₁:** Jessica
**R₁:** Jessica went to sit in her rocking chair. Today was her birthday and she was turning 80.

**Q₂:** How old would she be?
**A₂:** 80
**R₂:** She was turning 80

**Q₃:** Did she plan to have any visitors?
**A₃:** Yes
**R₃:** Her granddaughter Annie was coming over

**Q₄:** How many?
**A₄:** Three
**R₄:** Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie’s husband Josh were coming as well.

**Q₅:** Who?
**A₅:** Annie, Melanie and Josh
**R₅:** Her granddaughter Annie was coming over in the afternoon and Jessica was very excited to see her. Her daughter Melanie and Melanie’s husband Josh were coming as well.

---

Figure 1: A conversation from the CoQA dataset. Each turn contains a question (Qᵢ), an answer (Aᵢ) and a rationale (Rᵢ) that supports the answer.

---

Storytelling

• Most neural storytelling work uses some kind of prompt
  • Generate a story-like paragraph given an image
  • Generate a story given a brief writing prompt
  • Generate the next sentence of a story, given the story so far (story continuation)
    • This is different to the previous two, because we are not concerned with the system’s performance over several generated sentences

• Neural storytelling is taking off!
  • The first Storytelling Workshop was held in 2018
  • It held a competition (generate a story to accompany a sequence of 5 images)
What’s interesting here is that this isn’t straightforward supervised image-captioning. There was no paired data to learn from.
Generating a story from an image

• **Question:** How to get around the lack of parallel data?
• **Answer:** Use a common sentence-encoding space

• **Skip-thought vectors** are a type of general-purpose sentence embedding method
  • The idea is similar to how we learn an embedding for a word by trying to predict the words around it

• Using COCO (an image captioning dataset), learn a mapping from images to the skip-thought encodings of their captions

• Using the target style corpus (Taylor Swift lyrics), train a RNN-LM to decode a skip-thought vector to the original text

• Put the two together


Generating a story from a writing prompt

- In 2018, Fan et al released a new story generation dataset collected from Reddit’s WritingPrompts subreddit.
- Each story has an associated brief writing prompt.

**Prompt:** The Mage, the Warrior, and the Priest

**Story:** A light breeze swept the ground, and carried with it still the distant scents of dust and time-worn stone. The Warrior led the way, heaving her mass of armour and muscle over the uneven terrain. She soon crested the last of the low embankments, which still bore the unmistakable fingerprints of haste and fear. She lifted herself up onto the top the rise, and looked out at the scene before her. [...]
Generating a story from a writing prompt

Fan et al also proposed a complex seq2seq prompt-to-story model:

• It’s convolutional-based
  • This makes it faster than RNN-based seq2seq

• Gated multi-head multi-scale self-attention
  • The self-attention is important for capturing long-range context
  • The gates allow the attention mechanism to be more selective
  • The different attention heads attend at different scales – this means there are different attention mechanisms dedicated to retrieving fine-grained information and coarse-grained information

• Model fusion:
  • Pretrain one seq2seq model, then train a second seq2seq model that has access to the hidden states of the first
  • The idea is that the first seq2seq model learns general LM and the second learns to condition on the prompt

*conv seq2seq with decoder multi-head self-attention
Generating a story from a writing prompt

The results are impressive!

- **Related** to prompt
- **Diverse; non-generic**
- **Stylistically dramatic**

**Example Prompt 1:** Aliens start abducting humans

It has been two weeks, and the last of my kind has gone. It is only a matter of time until there will be nothing left. I’m not sure what the hell is going on... I can’t think. I can hear a distant scream. I think of a strange, alien sound. I try to speak, but am interrupted by something, something that feels like a drum, I can not tell. I mean I’m just a little bit older than an average human. But I can, and I can feel the vibrations. I hear the sound of a ship approaching. The ground quakes at the force of the impact, and a loud explosion shatters the silence.

**However:**

- Mostly atmospheric/descriptive/scene-setting; **less events/plot**
- When generating for longer, mostly **stays on the same idea** without moving forward to new ideas – coherence issues

Challenges in storytelling

Stories generated by neural LMs can sound fluent... but are meandering, nonsensical, with no coherent plot

What’s missing?

LMs model sequences of words. Stories are sequences of events.

- To tell a story, we need to understand and model:
  - Events and the causality structure between them
  - Characters, their personalities, motivations, histories, and relationships to other characters
  - State of the world (who and what is where and why)
  - Narrative structure (e.g. exposition → conflict → resolution)
  - Good storytelling principles (don’t introduce a story element then never use it)
Challenges in storytelling

Stories generated by neural LMs can sound fluent... but are meandering, nonsensical, with no coherent plot

What’s missing?
LMs model *sequences of words*. Stories are *sequences of events*.

- To tell a story, we need to understand and model:
  - Events and the causality structure between stories, and relationships to other stories, and
  - Characters, their personalities, motivations, histories, and
  - State of the world (who and what is where and why)

- Good storytelling principles (don’t introduce a story element then never use it)

THIS IS INCREDIBLY DIFFICULT
Event2event Story Generation

![Diagram](https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17046/15769)

*Figure 1: Our automated story generation pipeline. Dashed boxes and arrows represent future work.*

<table>
<thead>
<tr>
<th>Experiment</th>
<th>Input</th>
<th>Extracted Event(s)</th>
<th>Generated Next Event(s)</th>
<th>Generated Next Sentence</th>
</tr>
</thead>
<tbody>
<tr>
<td>All Generalized Events &amp;</td>
<td>He reaches out to Remus Lupin, a Defence Against the Dark Arts teacher who is eventually revealed to be a werewolf.</td>
<td>(male.n.02, get-13.5.1, ∅, &lt;CHAR&gt;0) (ORGANIZATION, say-37.7-1, monster.n.01, ∅)</td>
<td>⟨monster.n.01, amuse-31.1, sarge, ∅⟩ ⟨monster.n.01, amuse-31.1, realize, ∅⟩ ⟨monster.n.01, conjecture-29.5-1, ∅, ∅⟩ ⟨male.n.02, conduit.n.01, entity.n.01, ∅⟩ ⟨male.n.02, free-80-1, ∅, penal_institution.n.01⟩</td>
<td>When monster.n.01 nemesis.n.01 describes who finally realizes male.n.02 can not, dangerous entity.n.01 male.n.02 is released from penal_institution.n.01.</td>
</tr>
<tr>
<td>Generalized Sentence</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*Event Representations for Automated Story Generation with Deep Neural Nets, Martin et al, 2018*  
Structured Story Generation

You’re a Werewolf. You begin to transform, but instead of a terrifying beast, you turn into a small puppy.

| <V> opened | <AO> | ent0 | <A1> | ent0 eyes |
| <V> looking | <AO> | to | ent0 | ent1 |
| <V> found | <AO> | ent0 |
| <V> clipped | <A1> | ent1 |
| <V> flopped | <AO> | ent0 ears |
| <V> was | <AO> | the hunger | <A1> | gone |
| <V> clouded | <AO> | Confusion | <A1> | ent1 mind |
| <V> tilted | <AO> | ent0 | <A1> | ent2 |
| <V> approached | <A1> | ent0 | <A2> | a nearby puddle |
| <V> looked | <AO> | ent0 |

ent0 opened ent0 eyes. Looking to ent0 ent1, ent0 found that ent1 were now neatly clipped. ent0 ears flopped on either side of ent2 lazily, too soft and formless to hunt properly. Most of all, the hunger was gone. Confusion clouded ent0 mind and ent0 tilted ent2 instinctively. ent0 approached a nearby puddle and looked in.

I opened my eyes. Looking to my razor-sharp claws, I found that they were now neatly clipped. My ears flopped on either side of my head lazily, too soft and formless to hunt properly. Most of all, the hunger was gone. Confusion clouded my mind and I tilted my head instinctively. I approached a nearby puddle and looked in.

Figure 1: Proposed Model. Conditioned upon the prompt, we generate sequences of predicates and arguments. Then, a story is generated with placeholder entities such as ent0. Finally we replace the placeholders with specific references.

*CNN encoder–decoder with multi-head self attention

*seq2seq encoder–decoder with pointer copy

Tracking events, entities, state, etc.

- **Sidenote**: there’s been lots of work on tracking events/entities/state in neural NLU (natural language understanding)
  - For example, Yejin Choi’s group* does lots of work in this area

- **Applying these methods to NLG is even more difficult**
  - It’s more manageable if you narrow the scope:
  - Instead of generating open-domain natural language stories while tracking state...
  - generate a recipe (given the ingredients) while tracking the state of the ingredients!

*Yejin Choi research group: [https://homes.cs.washington.edu/~yejin/](https://homes.cs.washington.edu/~yejin/)
Tracking world state while generating a recipe

- **Neural Process Network**: generates recipe instructions, given the ingredients
- **Explicitly tracks the state** of all the ingredients, and uses this to decide what action to take next.
Poetry generation: Hafez

- **Hafez**: a poetry generation system by Ghazvininejad et al
- **Main idea**: Use a Finite State Acceptor (FSA) to define all possible sequences that obey the desired rhythm constraints. Then use the FSA to constrain the output of a RNN-LM.

**For example:**

- A Shakespearean sonnet is 14 lines of iambic pentameter

```
010 1 0 10 101
Attending on his golden pilgrimage
```

- So the Shakespearean sonnet FSA is \((01)^5)^{14}\)
- During beam search decoding, only explore hypotheses that fall within the FSA.
Poetry generation: Hafez

- Full system:
- User provides topic word
- Get a set of words related to topic
- Identify rhyming topical words. These will be the ends of each line
- Generate the poem using RNN-LM constrained by FSA
- The RNN-LM is *backwards* (right-to-left). This is necessary because last word of each line is fixed.

Poetry generation: Hafez

In a follow-up paper, the authors made the system interactive and user-controllable.

The control method is simple: during beam search, upweight the scores of words that have the desired features.
Non-autoregressive generation for NMT

• In 2018, Gu et al published a “Non-autoregressive Neural Machine Translation” model
  • Meaning: it does not generate the translation left-to-right, with each word depending on the ones before.

• It generates the translation in parallel!

• This has obvious efficiency advantages, but is also intriguing from a text generation point of view.

• The architecture is Transformer-based; the big difference is that the decoder can run in parallel at test time.

Non-autoregressive generation for NMT

Figure 2: The architecture of the NAT, where the black solid arrows represent differentiable connections and the purple dashed arrows are non-differentiable operations. Each sublayer inside the encoder and decoder stacks also includes layer normalization and a residual connection.
Automatic evaluation metrics for NLG

**Word overlap based metrics** (BLEU, ROUGE, METEOR, F1, etc.)

- We know that they’re **not ideal for machine translation**

- They’re **even worse for summarization**, which is more open-ended than machine translation
  - Unfortunately, ROUGE also rewards extractive summarization systems more than abstractive systems

- And they’re **even worse for dialogue**, which is more open-ended that summarization.
  - Similarly for e.g. story generation
Word overlap metrics are not good for dialogue

Figure 1: Scatter plots showing the correlation between metrics and human judgements on the Twitter corpus (a) and Ubuntu Dialogue Corpus (b). The plots represent BLEU-2 (left), embedding average (center), and correlation between two randomly selected halves of human respondents (right).

Word overlap metrics are not good for dialogue

*word based metric

*SIM: semantic similarity

*INF, NAT, QUA human evaluation

Automatic evaluation metrics for NLG

• What about perplexity?
  • Captures how powerful your LM is, but doesn’t tell you anything about generation (e.g. if your decoding algorithm is bad, perplexity is unaffected)

• Word embedding based metrics?
  • Main idea: compare the similarity of the word embeddings (or average of word embeddings), not just the overlap of the words themselves. Captures semantics in a more flexible way.
  • Unfortunately, still doesn’t correlate well with human judgments for open-ended tasks like dialogue.
Automatic evaluation metrics for NLG

• We have no automatic metrics to adequately capture overall quality (i.e. a proxy for human quality judgment).

• But we can define more focused automatic metrics to capture particular aspects of generated text:
  • Fluency (compute probability w.r.t. well-trained LM)
  • Correct style (prob w.r.t. LM trained on target corpus)
  • Diversity (rare word usage, uniqueness of n-grams)
  • Relevance to input (semantic similarity measures)
  • Simple things like length and repetition
  • Task-specific metrics e.g. compression rate for summarization

• Though these don’t measure overall quality, they can help us track some important qualities that we care about.
Human evaluation

• Human judgments are regarded as the gold standard
• Of course, we know that human eval is slow and expensive
• ...but are those the only problems?
• Supposing you do have access to human evaluation: Does human evaluation solve all of your problems?

• No!
• Conducting human evaluation effectively is very difficult
• Humans: • are inconsistent
  • can be illogical
  • lose concentration
  • misinterpret your question
  • can’t always explain why they feel the way they do
Detailed human eval of controllable chatbots

- **experience** working on a chatbot project (PersonaChat):

- We investigated **controllability** (in particular, controlling aspects of the generated utterances such as **repetition**, **specificity**, **response-relatedness**, and **question-asking**).

---

**Controlling specificity**

Input: Yes, I'm studying law at the moment  
**Baseline Response:** That sounds like a lot of fun!

<table>
<thead>
<tr>
<th>NIDF</th>
<th>Weighted Decoding Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.6%</td>
<td>Oh.....................................</td>
</tr>
<tr>
<td>17.1%</td>
<td>That sounds like a lot of fun!</td>
</tr>
<tr>
<td>18.3%</td>
<td>That sounds like a lot of fun. How long have you been studying?</td>
</tr>
<tr>
<td>38.5%</td>
<td>I majored in practising my spiritual full time philosophy test</td>
</tr>
<tr>
<td>71.9%</td>
<td>Oh wow! Merna jean isa paino yi hao hui bu acara sya gila [...]</td>
</tr>
</tbody>
</table>

**Controlling response-relatedness**

Input: Do you go get coffee often  
**Baseline Response:** I do, when I am not playing the piano.

<table>
<thead>
<tr>
<th>Wt</th>
<th>Sim</th>
<th>Weighted Decoding Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>-10.0</td>
<td>-0.05</td>
<td>I am a musician.</td>
</tr>
<tr>
<td>-5.0</td>
<td>-0.02</td>
<td>I do, when I am not playing the piano.</td>
</tr>
<tr>
<td>0.0</td>
<td>-0.02</td>
<td>I do, when I am not playing the piano.</td>
</tr>
<tr>
<td>5.0</td>
<td>0.40</td>
<td>I do, usually at starbucks.</td>
</tr>
<tr>
<td>8.0</td>
<td>0.59</td>
<td>Not usually, especially when you drink latte.</td>
</tr>
<tr>
<td>11.0</td>
<td>0.72</td>
<td>Not often, usually with drinks, espresso, latte, tea, etc.</td>
</tr>
</tbody>
</table>

---

*Normalized IDF

---

Detailed human eval of controllable chatbots

- How to ask for human quality judgments?
- We tried simple overall quality (multiple-choice) questions like:
  - *How well did this conversation go?*
  - *How engaging was this user?*
  - *Which of these users gave a better response?*
  - *Would you want to talk to this user again?*
  - *Do you think this user is a human or a bot?*

- **Major problems:**
  - Necessarily very subjective
  - Respondents have different expectations; this affects their judgments
  - Catastrophic misunderstanding of the question (e.g. “the chatbot was very engaging because it always wrote back”)
  - Overall quality depends on many underlying factors; how should they be weighed and/or compared?

---

*What makes a good conversation? How controllable attributes affect human judgments,*

Ultimately, we designed a detailed human evaluation system that separates out the important factors that contribute to overall chatbot quality:

![Diagram of human evaluation system]

*not for task dialog system

Figure 1: We manipulate four low-level attributes and measure their effect on human judgments of individual conversational aspects, as well as overall quality.

Detailed human eval of controllable chatbots

Findings:

• Controlling repetition is extremely important for all human judgments
• Asking more questions improves engagingness
• Controlling specificity (less generic utterances) improves engagingness, interestingness and perceived listening ability of the chatbot.
  • However, human evaluators have a low tolerance for the risks (e.g. nonsensical or non-fluent output) associated with the less generic bot
• The overall metric “engagingness” (i.e. enjoyment) is easy to maximize – our bots reached near-human performance
• The overall metric “humanness” (i.e. Turing test) is not at all easy to maximize – all bots are far below human performance

• Humanness is not the same as conversational quality!
• Humans are suboptimal conversationalists: they scored poorly on interestingness, fluency, listening, and asked too few questions.

Possible new avenues for NLG eval?

• Corpus-level metrics
  • Should an eval metric be applied to each example in the test set independently, or a function of the whole corpus?
  • e.g. if a dialogue model always gives the same generic answer to every example in the test set, it should be penalized
• Eval metrics that measure the diversity-safety tradeoff
• Human eval for free
  • Gamification: make the task (e.g. talking to a chatbot) fun, so humans provide supervision and implicit evaluation for free
• Adversarial discriminator as an evaluation metric
  • Test whether the NLG system can fool a discriminator which is trained to distinguish human text from artificially generated text
Exciting current trends in NLG

- Incorporating discrete latent variables into NLG
  - May help with modeling structure in tasks that really need it, like storytelling, task-oriented dialogue, etc

- Alternatives to strict left-to-right generation
  - Parallel generation, iterative refinement, top-down generation for longer pieces of text

- Alternative to maximum likelihood training with teacher forcing
  - More holistic sentence-level (rather than word-level) objectives
NLG research: Where are we? Where are we going?

• ~5 years ago, NLP + Deep Learning research was a wild west

• Now (2019), it’s a lot less wild

• ...but NLG seems like one of the wildest parts remaining

Image credit: mstodt on Pixabay
Neural NLG community is rapidly maturing

- During the **early years** of NLP + Deep Learning, community was mostly **transferring successful NMT methods to NLG tasks**.
- Now, increasingly more **inventive NLG techniques emerging**, specific to **non-NMT generation settings**.
- Increasingly more (neural) **NLG workshops and competitions**, especially focusing on open-ended NLG:
  - NeuralGen workshop
  - Storytelling workshop
  - Alexa challenge
  - ConvAI2 NeurIPS challenge
- **These are particularly useful to organize the community, increase reproducibility, standardize eval, etc.**
- **The biggest roadblock for progress is eval**