The Whole Truth and Nothing But the Truth: Faithful and Controllable Dialogue Response Generation with Dataflow Transduction and Constrained Decoding

Hao Fang*, Anusha Balakrishnan*, Harsh Jhamtani*, John Bufe, Jean Crawford, Jayant Krishnamurthy, Adam Pauls, Jason Eisner, Jacob Andreas, Dan Klein

**Two Predominant Paradigms for Dialogue Response Generation**

- rule-based generation
  - easy to control (by modifying rules)
  - safe for production (can only produce responses allowed by hand-written rules)
  - issues:
    - hard to maintain for complex domains.
    - requires extensive domain knowledge, including both low-level details like the grammar and high-level properties like truthfulness.

- dataflow transduction
  - produces fluent, coherent, and diverse responses.
  - can leverage pre-trained large language models (e.g., GPT-3, ChatGPT).
  - issues:
    - suffer from hallucination.
    - struggle in maintaining faithfulness.
    - produce unsafe responses.
    - difficult to control.

**Our Framework: A Hybrid Approach for Response Generation**

- the context-free grammar (CFG) defines the space of all responses allowed for the given computation (dataflow graph).
- responses are faithful but not always grammatical, or natural.
- found 1 event on Thursday. It’s “Show and Tell”.
- found 1 events on Thursday.
- The “Show and Tell” meeting on Thursday starts on Thursday.
- Hybrid generation has a long history in NLP dating back to Knight & Hatzivassiloglou (1995) and Langkilde & Knight (1998).

**Example Dataflow Transduction**

- function definitions
  - `findEventsOnDate` (e.g., `findEventsOnDate(date)`)
  - `nonEmpty` (e.g., `nonEmpty(v)`)

- function application
  - `nonEmpty(v)`
  - `findEventsOnDate(date)`

- variable extraction
  - `v` (e.g., `v`)

**Neural Language Modeling**

- produce fluent, coherent, and diverse responses.
- can leverage pre-trained large language models (e.g., GPT-3, ChatGPT).
- issues:
  - suffer from hallucination.
  - struggle in maintaining faithfulness.
  - produce unsafe responses.
  - difficult to control.

**Dataflow Transduction Rules**

- **Head:** $\delta$
- **Body:** 
  - `match computation:`
  - `case findEventsOnDate(date):`
  - `num = size(computation)`
  - `event = head(computation)`
  - `return "num": num, "event": event, "date": date`

- **Response Template:** 
  - `find([LEX <num> event] PP [date])`. It’s `{EVENT <event>}`

**Data and Human Evaluation**

- SMCalFlow2Text
  - a subset of SMCalFlow examples involving calendar event queries.
  - 8938 training examples, 1041 validation examples, with meta information for executing the dataflow programs.
  - 187 transition rules (written by some of us in a matter of hours) sufficient to cover all gold agent responses.

- Human evaluation
  - grammaticality (‘has the virtual assistant made any grammar errors?’)
  - relevance (‘has the virtual assistant misunderstood the user’s request?’)
  - truthfulness (‘has the virtual assistant provided any incorrect information as judged using the database and timestamp?’)

**Conclusion**

- a hybrid approach for building dialogue response generation systems.
- developers can write transduction rules to faithfully describe computations.
- surface realization decisions are deferred to a flexible language model.
- the proposed approach outperforms constrained conditional language modeling in both automatic and human evaluations, especially on truthfulness.
- several expert hours spent on authoring rules hold almost equivalent value to a large volume of training data.
- code and data: https://github.com/microsoft/dataflow2text

---

**Dataflow Transduction (F, Σ, R, t_start)**

- **Nonterminal Types ($t \in T$)**
  - $S$: the start nonterminal
  - PP, NP, …: syntactic categories
  - EVENT, ...: semantic categories

- **Terminals ($w \in Σ$)**
  - LEX: lexicization

- **Translation Rules ($r \in R$)**
  - Applied to a dataflow node $v$ to create a QCFG production ($v_s \rightarrow \beta_f$), $f$ is fixed.

**Constrained Decoding**

- generate response candidates from a neural LM (pre-trained and fairly fine-tuned), constrained by the QCFG.

**Dataflow Transducer**

- the transducer can extend the graph as needed.

---

**Head:** $\delta$

**Body:**
  - `match computation:`
  - `case findEventsOnDate(date):`
  - `num = size(computation)`
  - `event = head(computation)`
  - `return "num": num, "event": event, "date": date`

**Response Template:**
  - `find([LEX <num> event] PP [date])`. It’s `{EVENT <event>}`

---

**Dataflow Transduction Graph G**

- **EXPANDED DATAFLOW GRAPH G**

- **QCFG PRODUCTIONS**

- **THE RESULTING QCFG COMPACTLY REPRESENTS A COMBINATORIAL SPACE OF POSSIBLE RESPONSES.**

---

**SMCalFlow2Text**

- a subset of SMCalFlow examples involving calendar event queries.

- 8938 training examples, 1041 validation examples, with meta information for executing the dataflow programs.

- 187 transition rules (written by some of us in a matter of hours) sufficient to cover all gold agent responses.

---

**Human Evaluation**

- **Grasstmaticality** (‘has the virtual assistant made any grammar errors?’)
- **Relevance** (‘has the virtual assistant misunderstood the user’s request?’)
- **Truthfulness** (‘has the virtual assistant provided any incorrect information as judged using the database and timestamp?’)

---

**Dataflow Transduction Procedure**

- Each dataflow node is converted to one or more specialized nonterminals, which expand to natural language descriptions of that node.
- Descriptions are nested: they can recurse to descriptions of neighboring nodes.
- Neighboring nodes may be added on demand.
- See a full example at the bottom of the poster.
- Neighboring nodes may be added on demand.
- Descriptions are nested: they can recurse to descriptions of neighboring nodes.

---

**Nonterminal Types ($t \in T$)**

- $S$: the start nonterminal
- PP, NP, …: syntactic categories
- EVENT, ...: semantic categories

**Terminals ($w \in Σ$)**

- LEX: lexicization

**Translation Rules ($r \in R$)**

- Applied to a dataflow node $v$ to create a QCFG production ($v_s \rightarrow \beta_f$), $f$ is fixed.

---

**Conclusion**

- A hybrid approach for building dialogue response generation systems.
- Developers can write transduction rules to faithfully describe computations.
- Surface realization decisions are deferred to a flexible language model.
- The proposed approach outperforms constrained conditional language modeling in both automatic and human evaluations, especially on truthfulness.
- Several expert hours spent on authoring rules hold almost equivalent value to a large volume of training data.
- Code and data: https://github.com/microsoft/dataflow2text