Barkavi Sundararajan

University of Aberdeen
King's College
Aberdeen
United Kingdom

b.sundararajan.21@abdn.ac.uk

1 Research Overview

Neural Language Models have become central to Natural Language Generation (NLG) research and can produce fluent and coherent text (Badaro et al., 2023). However, when models generate text from complex, structured or long-form data such as tables or event logs, they often hallucinate and introduce factual errors (Jacovi et al., 2025; Rebuffel et al., 2021). These hallucinations limit the practical deployment of large language models (LLMs) in applications where factual accuracy is critical. In my research, *factual accuracy* refers to the faithfulness of the generated text to the given input data. My PhD focuses on reducing hallucinations and improving factual accuracy in data-to-text generation, which I address through two core approaches: (i) analysing how input quality and structure improve factual accuracy, and (ii) developing a manual error annotation protocol and extending it into an LLM-as-Judge framework. This work aims to assess when automatic evaluation can complement human annotation and enable larger-scale evaluation.

Past work: In my initial work **(Sundararajan et al., 2022)**, I fine-tuned a T5-base model (Raffel et al., 2023) on the ToTTo dataset (Parikh et al., 2020) using the published configuration and compared its outputs with two GeM benchmark models (Gehrmann et al., 2021), which at the time represented the top-performing ToTTo baselines as ranked by PARENT, BLEURT, and BLEU. Using a manual error annotation protocol adapted from Thomson et al. (2023), extended to include `DATE-DIMENSION` and `NON-ENGLISH` error categories, I analysed more than 3,000 outputs in the politics domain and categorised errors into eight types (`WORD`, `NAME`, `NUMBER`, `DATE-DIMENSION`, `CONTEXT`, `NOTCHECKABLE`, `NON-ENGLISH`, and `OTHER`).

- By capturing factual errors more effectively than surface-level automatic metrics, this protocol *revealed systematic weaknesses such as verb mispredictions, numeric swaps, and hallucinations around dates* and demonstrated limitations that BLEU (Papineni et al., 2002) and similar metrics fail to detect, reflecting broader evaluation challenges discussed in Gehrmann et al. (2022).
- The analysis also highlighted challenges when reasoning over complex tables, especially for tables with more than 20 rows.
- The annotation protocol achieved substantial inter-annotator agreement between two independent annotators (Cohen's Kappa = 0.79) on a 10% subset of the outputs, confirming the reliability of the categories (see Appendix A for the procedure).

Building on this, I investigated whether hallucinations in neural table-to-text models can be traced back to problems in the tabular input in the ToTTo dataset **(Sundararajan et al., 2024)**. I manually annotated 1,837 outputs in the politics domain using a further-adapted protocol that included an `ADDITION` category and excluded `NON-ENGLISH`. On a 6.5% overlapping subset of the outputs (120 in total) annotated independently by three annotators, the annotation achieved substantial inter-annotator agreement (Fleiss' Kappa = 0.622).

- This analysis identified several recurring input problems: *non-atomic cell values, insufficient input, longer table inputs, complex tables*, and *other politics domain-specific* input problems.
- I then systematically corrected the non-standard tabular inputs to a standard form (see Appendix B). Across 210 samples, input correction reduced factual errors by 62% in T5-base and 57% in T5-large.
- I extended this to Llama2 models (Touvron et al., 2023), correcting inputs decreased factual errors by 52% for Llama2–7B and 76% for Llama2–13B.

These results demonstrate that many hallucinations arise from garbage-in, garbage-out effects: poorly structured inputs lead to more factual errors, while corrected inputs substantially improve factual accuracy. This finding is consistent with prior work, which shows that improving inputs in NLG datasets leads to better outputs (Dušek and Kasner, 2020).

Proceedings of the 1st Workshop for Young Researchers in Natural Language Generation, pages 10–16

Recent work: In this work (accepted at INLG 2025), I investigated how different input structures influence the factual accuracy of LLM-generated NBA game summaries from play-by-play data. This is a challenging text generation task with over 450 rows and 13 columns because play-by-play data contains rich information about plays, including entities (teams and players), events/actions (e.g., jump shot, layup, dunk, free throw, rebound), outcomes (missed or made for any attempt), and attributes (score, time, period, distance). To empirically study the impact of input structuring on model output, we experimented with three input data formats: unstructured (control), row-based, and JSON, and generated 180 summaries using Llama3.1-70B and Qwen2.5-72B (Qwen et al., 2025).

We manually annotated factual errors using a further adapted protocol that distinguishes `WORD-OBJECTIVE` (fact-checkable) from `WORD-SUBJECTIVE` (opinion). Error counts were normalised to 'errors per 100 words'.

- A two-way repeated-measures ANOVA on input structure (unstructured, row-based, JSON) revealed significant main effects on error rates ($p < 0.05$).
- JSON inputs cut error rates by 69% for Llama3.1-70B and 65% for Qwen2.5-72B versus unstructured text. Row-based inputs cut errors by 54% for Llama3.1-70B and 51% for Qwen2.5-72B, and switching from row-based to JSON cuts errors by another 32% for Llama3.1-70B and 27% for Qwen2.5-72B.
- Error-type analysis shows Llama3.1-70B produces more numeric mistakes, whereas Qwen2.5-72B makes more naming and subjective-wording errors. Word objective errors are predominant among both models.

These findings demonstrate that structured input representations substantially improve factual accuracy in complex, event-driven summarisation tasks such as sports reporting, although limitations remain in scaling manual annotations and in generalizing to other domains.

Ongoing and future work: The recent study shows that structured inputs improve factual accuracy but also expose the limits of manual evaluation, which is costly and difficult to scale. To address this, I am developing an LLM-as-Judge framework that operationalises my human error annotation protocol into structured prompts and directs judge LLMs to assess atomic factual claims against play-by-play input logs. I will evaluate agreement with human annotations using accuracy metrics and determine when automatic judgments can reliably complement manual ones. As part of this ongoing work, I plan to extend the automatic evaluation framework to other event-driven sports such as ice hockey to examine its generalizability.

2 Biography

Barkavi Sundararajan is a final-year PhD student in Computer Science at the University of Aberdeen. Her research interests lie in reducing hallucinations and improving factual accuracy in data-to-text generation, with a particular focus on developing reliable models and robust evaluation methods for factuality. She is a part-time Data Scientist intern at the UK Ministry of Justice, where she codifies free-text assessments from large-scale datasets into structured data to support projects such as Safer Streets UK. She plans to pursue a career as a research scientist or applied scientist at the intersection of natural language processing and high-stakes decision-making domains (where factual accuracy is critical).

3 Questions for Senior Researchers

1. What role can knowledge graphs play in improving factuality in neural NLG for long-form inputs, compared with other approaches such as retrieval or input restructuring?
2. How can one best approach the transition from a PhD in NLG to Research Scientist or Applied Scientist roles in industry labs?
3. As a PhD student, how can I identify NLG projects that are both publishable on standard benchmarks and meaningful for real-world, high-stakes applications?
4. Which research directions are most likely to advance NLG in the coming years?
5. How can we design evaluation methods for NLG that scale beyond small manual annotations while still capturing factual accuracy and reducing reliance on surface-level metrics?

4 Topics for Roundtable Discussions with Peers

1. The Role of Input Representation and Formatting in Generation Quality
2. Generalizing Factuality Evaluation: Bridging Closed Benchmarks and Real-World Complex Domains (sports, finance, or healthcare)
3. Scaling Evaluation: Designing Protocols that remain reliable while Reducing Cost and Annotation Effort
4. Navigating Career Paths from Academia: Opportunities and Challenges

Acknowledgements

We thank the anonymous reviewers for their constructive and detailed feedback, which helped clarify and improve this paper. Due to space constraints, some of their questions could not be addressed in the main paper; I will discuss them during the workshop and have included further clarifications and analyses in the appendices.

References

Gilbert Badaro, Mohammed Saeed, and Paolo Papotti. 2023. Transformers for tabular data representation: A survey of models and applications. *Transactions of the Association for Computational Linguistics* 11:227–249. https://aclanthology.org/2023.tacl-1.14.

Ondřej Dušek and Zdeněk Kasner. 2020. Evaluating semantic accuracy of data-to-text generation with natural language inference. In Brian Davis, Yvette Graham, John Kelleher, and Yaji Sripada, editors, *Proceedings of the 13th International Conference on Natural Language Generation*. Association for Computational Linguistics, Dublin, Ireland, pages 131–137. https://doi.org/10.18653/v1/2020.inlg-1.19.

Sebastian Gehrmann, Tosin Adewumi, Karmanya Aggarwal, Pawan Sasanka Ammanamanchi, Aremu Anuoluwapo, Antoine Bosselut, Khyathi Raghavi Chandu, Miruna Clinciu, Dipanjan Das, Kaustubh D. Dhole, Wanyu Du, Esin Durmus, Ondřej Dušek, Chris Emezue, Varun Gangal, Cristina Garbacea, Tatsunori Hashimoto, Yufang Hou, Yacine Jernite, Harsh Jhamtani, Yangfeng Ji, Shailza Jolly, Mihir Kale, Dhruv Kumar, Faisal Ladhak, Aman Madaan, Mounica Maddela, Khyati Mahajan, Saad Mahamood, Bodhisattwa Prasad Majumder, Pedro Henrique Martins, Angelina McMillan-Major, Simon Mille, Emiel van Miltenburg, Moin Nadeem, Shashi Narayan, Vitaly Nikolaev, Rubungo Andre Niyongabo, Salomey Osei, Ankur Parikh, Laura Perez-Beltrachini, Niranjan Ramesh Rao, Vikas Raunak, Juan Diego Rodriguez, Sashank Santhanam, João Sedoc, Thibault Sellam, Samira Shaikh, Anastasia Shimorina, Marco Antonio Sobrevilla Cabezudo, Hendrik Strobelt, Nishant Subramani, Wei Xu, Diyi Yang, Akhila Yerukola, and Jiawei Zhou. 2021. The gem benchmark: Natural language generation, its evaluation and metrics. https://arxiv.org/abs/2102.01672.

Sebastian Gehrmann, Elizabeth Clark, and Thibault Sellam. 2022. Repairing the cracked foundation: A survey of obstacles in evaluation practices for generated text. https://arxiv.org/abs/2202.06935.

Alon Jacovi, Andrew Wang, Chris Alberti, Connie Tao, Jon Lipovetz, Kate Olszewska, Lukas Haas, Michelle Liu, Nate Keating, Adam Bloniarz, Carl Saroufim, Corey Fry, Dror Marcus, Doron Kukliansky, Gaurav Singh Tomar, James Swirhun, Jinwei Xing, Lily Wang, Madhu Gurumurthy, Michael Aaron, Moran Ambar, Rachana Fellinger, Rui Wang, Zizhao Zhang, Sasha Goldshtein, and Dipanjan Das. 2025. The facts grounding leaderboard: Benchmarking llms' ability to ground responses to long-form input. https://arxiv.org/abs/2501.03200.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In Pierre Isabelle, Eugene Charniak, and Dekang Lin, editors, *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Philadelphia, Pennsylvania, USA, pages 311–318. https://doi.org/10.3115/1073083.1073135.

Ankur Parikh, Xuezhi Wang, Sebastian Gehrmann, Manaal Faruqui, Bhuwan Dhingra, Diyi Yang, and Dipanjan Das. 2020. ToTTo: A controlled table-to-text generation dataset. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics, Online, pages 1173–1186. https://doi.org/10.18653/v1/2020.emnlp-main.89.

Qwen, :, An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, Huan Lin, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jingren Zhou, Junyang Lin, Kai Dang, Keming Lu, Keqin Bao, Kexin Yang, Le Yu, Mei Li, Mingfeng Xue, Pei Zhang, Qin Zhu, Rui Men, Runji Lin, Tianhao Li, Tianyi Tang, Tingyu Xia, Xingzhang Ren, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yu Wan, Yuqiong Liu, Zeyu Cui, Zhenru Zhang, and Zihan Qiu. 2025. Qwen2.5 technical report. https://arxiv.org/abs/2412.15115.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2023. Exploring the limits of transfer learning with a unified text-to-text transformer. https://arxiv.org/abs/1910.10683.

Clément Rebuffel, Marco Roberti, Laure Soulier, Geoffrey Scoutheeten, Rossella Cancelliere, and Patrick Gallinari. 2021. Controlling hallucinations at word level in data-to-text generation. *Data Mining and Knowledge Discovery* 36:318 – 354. https://api.semanticscholar.org/CorpusID:231802211.

Barkavi Sundararajan, Somayajulu Sripada, and Ehud Reiter. 2022. Error analysis of ToTTo table-to-text neural NLG models. In Antoine Bosselut, Khyathi Chandu, Kaustubh Dhole, Varun Gangal, Sebastian Gehrmann, Yacine Jernite, Jekaterina Novikova,

and Laura Perez-Beltrachini, editors, *Proceedings of the Second Workshop on Natural Language Generation, Evaluation, and Metrics (GEM)*. Association for Computational Linguistics, Abu Dhabi, United Arab Emirates (Hybrid), pages 456–470. https://doi.org/10.18653/v1/2022.gem-1.43.

Barkavi Sundararajan, Yaji Sripada, and Ehud Reiter. 2024. Improving factual accuracy of neural table-to-text output by addressing input problems in ToTTo. In Kevin Duh, Helena Gomez, and Steven Bethard, editors, *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*. Association for Computational Linguistics, Mexico City, Mexico, pages 7350–7376. https://doi.org/10.18653/v1/2024.naacl-long.408.

Craig Thomson, Ehud Reiter, and Barkavi Sundararajan. 2023. Evaluating factual accuracy in complex data-to-text. *Computer Speech & Language* 80:101482. https://doi.org/https://doi.org/10.1016/j.csl.2023.101482.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288* .

Appendices

A Inter-Annotation Procedure for Participants

1. **Guidelines.** For the *ToTTo* and *NBA play-by-play summaries* annotations, I provided a detailed document defining the error categories with examples.

2. **Recruitment.** I initially recruited four annotators for the ToTTo task (Sundararajan et al., 2024); two did not proceed beyond the pilot stage after a quality check. Two trained annotators and I completed the final *ToTTo* annotations. For the NBA task, two qualified annotators started; one of them withdrew due to time constraints. The other annotator and I completed the annotations for *NBA game summaries*.

3. **Clarification.** Before the main task, I held short clarification sessions (meetings and emails) to ensure a shared understanding of the guidelines.

4. **Annotation.** Annotators completed their assigned samples independently and uploaded their annotation files via a Microsoft form.

5. **Compensation.** For the NBA task, the external annotator received a minimum payment of €35.

B Input Data Corrections and Output Text

We followed a systematic procedure to apply manual fixes identified in specific input problems in the ToTTo politics-domain dataset, as detailed in Sundararajan et al. (2024). A simple pseudocode is presented, where the Algorithm 1 takes tabular data and title (metadata) from ToTTo as input parameters to execute six functions to gather insights and make corrections to each of the input problems. The first two functions, DATASIZEMANAGEABLEFORLONGTABLE and IDENTIFYLEADERNAMEORDER do not correct the tabular input but provide insights on the input problems leading to factual errors in the outputs. The other four functions, CORRECTNONATOMICCELLS, UPDATEHEADERS, ADDRESSMISSINGVALUES, and REPLACESYMBOLS correct the input problems as presented in the paper.

Function summaries.

DATASIZEMANAGEABLEFORLONGTABLE(tabularData)
checks size thresholds (default: $\leq$ 20 rows, $\leq$ 10 columns); returns a boolean.

IDENTIFYLEADERNAMEORDER(tabularData, title)
extracts the leader name from the title and scans rows to record the name sequence; returns $(\mathit{leaderNameFromTitle}, \mathit{recordedData})$.

CORRECTNONATOMICCELLS(tabularData)
splits multi-valued cells into atomic columns; returns whether any corrections were made.

UPDATEHEADERS(tabularData)
fixes nested/misaligned headers; returns whether any corrections were made.

ADDRESSMISSINGVALUES(tabularData)
fills missing cells from local context; returns whether any corrections were made.

REPLACESYMBOLS(tabularData)
normalises domain-specific symbols and party abbreviations; returns whether any corrections were made.

Algorithm 1 Manual Correction Procedure for ToTTo Tabular Input

function MAINCORRECTIONPROCEDURE($tabularData, title$)
  if not DATASIZEMANAGEABLEFORLONGTABLE($tabularData$) **then**
    return "Please simplify the tabular data with fewer records", null
  end if
  $leaderName, recordedData \leftarrow$ IDENTIFYLEADERNAMEORDER($tabularData, title$)
  $correctionsMade \leftarrow$ **false**
  $correctionsMade \leftarrow$ CORRECTNONATOMICCELLS($tabularData$) **or** $correctionsMade$
  $correctionsMade \leftarrow$ UPDATEHEADERS($tabularData$) **or** $correctionsMade$
  $correctionsMade \leftarrow$ ADDRESSMISSINGVALUES($tabularData$) **or** $correctionsMade$
  $correctionsMade \leftarrow$ REPLACESYMBOLS($tabularData$) **or** $correctionsMade$
  if $correctionsMade$ **then**
    return $tabularData$, "Leader Data: " + $recordedData$ ▷ "Returns corrected input"
  else
    return $tabularData$, "Leader Data: " + $recordedData$ ▷ "Returns original input, if no issues"
  end if
end function

Algorithm 2 Verify the number of rows and columns in Longer Tabular Input

function DATASIZEMANAGEABLEFORLONGTABLE($tabularData$)
  $maxRows \leftarrow 20$ ▷ Maximum allowable number of rows for the chosen models
  $maxCols \leftarrow 10$ ▷ Maximum allowable number of columns for the chosen models
  return (length(*tabularData*) $\leq maxRows$) **and** (length(*tabularData*[0]) $\leq maxCols$) ▷ Both conditions must be true for returning true.
end function

Algorithm 3 Identify Leader Name Order in Title and Rows

function IDENTIFYLEADERNAMEORDER(*tabularData*, *title*)
  $leaderNameFromTitle \leftarrow$ ExtractLeaderNameFromTitle(*title*)
  $leaderNamesFromRows \leftarrow [\,]$
  for each row **in** *tabularData* **do**
    for each cell **in** current row **do**
      if leader_name is found in cell **then**
        Add leader_name to $leaderNamesFromRows$ ▷ Preliminary step: Record leader's names from rows
      end if
    end for
  end for
  if leaderNameFromTitle is not None **then** ▷ **Scenario 1:** Leader's name found in title
    $recordedData \leftarrow [\,]$ ▷ Initialize empty list to store sequential order of leader names
    Add $leaderNameFromTitle$ to $recordedData$
    for each leader_name **in** *leaderNamesFromRows* **do**
      Add leader_name to $recordedData$
    end for
    return $leaderNameFromTitle, recordedData$ ▷ for example, this returns "Leader b", ["Leader b", "Leader a", "Leader b", "Leader c"]
  else ▷ **Scenario 2:** Leader name not found in title
    if length of $leaderNamesFromRows > 0$ **then** ▷ when leader's name is found in at least one or more rows
      $recordedData \leftarrow$ "Leader name not in title"
      for each leader_name **in** *leaderNamesFromRows* **do**
        Add leader_name to $recordedData$
      end for
      return $leaderNameFromTitle, recordedData$ ▷ for example, this returns None, ["Leader name not in title", "Leader a", "Leader b", "Leader c"]
    else ▷ **Scenario 3:** Leader name not found in title or rows
      return $leaderNameFromTitle$, "Leader name not found"
    end if
  end if
end function

Algorithm 4 Correct Non-Atomic Cells

function CORRECTNONATOMICCELLS($tabularData$)
  $correctionsMade \leftarrow$ **false**
  for all $cell$ **in** $tabularData$ **do**
    if CellIsNonAtomic($cell$) **then**
      CorrectCell($cell$) ▷ Separate multiple atomic values into individual columns
      $correctionsMade \leftarrow$ **true**
    end if
  end for
  return $correctionsMade$
end function

Algorithm 5 Update Column Headers to Atomic cells

```
function UPDATEHEADERS(tabularData)
    correctionsMade ← false
    for all column in tabularData do
        if HeaderIsIncorrect(column) then
            UpdateHeader(column)                    ▷ Fix nested column and row headers
            correctionsMade ← true
        end if
    end for
    return correctionsMade
end function
```

Algorithm 6 Address Missing cell Values

```
function ADDRESSMISSINGVALUES(tabularData)
    correctionsMade ← false
    for all row in tabularData do
        if RowHasMissingValues(row) then
            FillCellValues(row)                     ▷ Pass the right missing cells from the table
            correctionsMade ← true
        end if
    end for
    return correctionsMade
end function
```

Algorithm 7 Replace Politics Domain-Specific Symbols and Abbreviate Party Names with correct semantics

```
function REPLACESYMBOLS(tabularData)
    correctionsMade ← false
    for all cell in tabularData do
        if CellContainsDomainSpecificSymbols(cell) then
            SubstituteSymbolWithEquivalent(cell)    ▷ Pass the right semantic replacing symbols in the input
        else if CellContainsPartyAbbreviations(cell) then
            AbbreviatePartyNames(cell)              ▷ Replace party name abbreviations
            correctionsMade ← true
        end if
    end for
    return correctionsMade
end function
```