DipInfo-UniTo at the GEM'24 Data-to-Text Task: Augmenting LLMs with the Split-Generate-Aggregate Pipeline

Michael Oliverio, Pier Felice Balestrucci, Alessandro Mazzei and Valerio Basile
University of Turin - Italy Computer Science Department
michael.oliverio@edu.unito.it
pierfelice.balestrucci@unito.it
alessandro.mazzei@unito.it
valerio.basile@unito.it

Abstract

This paper describes the DipInfo-UniTo system participating to the GEM Shared Task 2024. We participate only to the Data-to-Text (D2T) task. The DipInfo-UniTo system is based on Mistral (Jiang et al., 2023), a recent Large Language Model (LLM). Most LLMs are capable of generating high-quality text for D2T tasks but, crucially, they often fall short in terms of adequacy, and sometimes exhibit "hallucinations". To mitigate this issue, we have implemented a generation pipeline that combines LLMs with techniques from the traditional Natural Language Generation (NLG) pipeline. In particular, we have a three step process *SGA*, consisting in (1) *S*plitting the original set of triples, (2) *G*enerating verbalizations from the resulting split data units, (3) *A*ggregating the verbalizations produced in the previous step.

1 Introduction

In the last few years, LLMs have become the state of the art in natural language generation tasks, as can be seen in the most important conferences and challenges, such as the INLG conference and the WebNLG challenge.[1][2] LLMs enable high performance across various fields of NLP, including RDF-to-Text. The use of such models can occur through prompting techniques or, if there is a suitable dataset available for their task, by fine-tuning the models, which involves further training. This latter approach leaded to improved performances in many tasks related to generation. Systems like LLaMA2 (Touvron et al., 2023) and Mistral are among the most popular open-weights system for text generation. Fortunately, a linguistic resource for fine-tuning these models is provided by the WebNLG challenge. This corpus, originally designed for English and later extended to other languages (German (Ferreira et al., 2018), Russian (Shimorina et al., 2019) and partially Maltese (Cripwell et al., 2023), among others), consists of data units, i.e., sets of RDF triples, composed of subject, predicate, and object, accompanied by their verbalizations, which represent the semantics of the triples. The system employed in the GEM Shared Task (Mille et al., 2024) consists of a three-step pipeline, which we called SGA (split-generate-aggregate). It includes a Data Unit Splitting Algorithm (S) to simplify data units for subsequent steps, an RDF-to-Text System (G) designed to generate verbalizations from obtained data units, and a Sentence Aggregation System (A) to combine the verbalizations produced in the previous steps.

The paper is structured as follows: in Section 2 we provided a brief selection of related work; in Section 3 we give few details about GEM Shared Task; in Section 4 we give some details on WebNLG 3.0, that is our training corpus; in Section 5 we describe the SGA pipeline; in Section 6 we present the official results of the DipInfo-UniTo system and, finally, Section 7 closes the paper by considering future development. The code and submitted outputs of the DipInfo-UniTo system can be found on GitHub.[3]

2 Related Works

Over the years, RDF-to-Text has become an increasingly important task. Several WebNLG challenges have been held (2017, 2020, and 2023) to develop the best RDF-to-Text models based on WebNLG corpora.[4][5][6] A common strategy involves using prompting techniques or fine-tuning to generate verbalizations from given RDF triples (Wang

[1]https://aclanthology.org/venues/inlg/
[2]https://synalp.gitlabpages.inria.fr/webnlg-challenge/
[3]https://github.com/MichaelOliverio/DipInfo-UniTo-GEM24
[4]https://synalp.gitlabpages.inria.fr/webnlg-challenge/challenge_2017/
[5]https://synalp.gitlabpages.inria.fr/webnlg-challenge/challenge_2020/
[6]https://synalp.gitlabpages.inria.fr/webnlg-challenge/challenge_2023/

et al., 2021). In the latest WebNLG challenge, several pipelines emerged to generate more accurate outputs, incorporating techniques such as data splitting to reduce the input data and backtranslation for low-resource languages (Kumar et al., 2023).

3 GEM 2024 RDF-to-Text Task Description

The GEM Shared Task 2024 focuses on text summarization and RDF-to-Text generation. Our participation is limited to the second task, which involves generating verbalizations from a set of RDF triples. These triples, consisting of a subject, predicate, and object. The shared task provides six files containing RDF triples extracted from the web. Three of these files each contain $1,799$ inputs extracted from WebNLG and are classified as "seen" inputs, because these data could have corresponding gold-standard verbalizations that can be used to train potential statistical or neural systems. The other three files each contain $1,800$ inputs extracted from Wikipedia, for which no gold-standard verbalizations are available online. These inputs are therefore classified as "unseen" inputs. These files contain triples extracted directly from WebNLG and Wikidata, altered triples where the subject or object has been changed and triples with entities generated using LLM prompting. The task is designed for multiple languages, including English, Chinese, German, Russian, Spanish, Korean, Hindi, Swahili, and Arabic. In our case, we have chosen to participate in the task using only English.

4 English WebNLG Corpus Description

WebNLG is a corpus containing data units, a set of RDF triples, each paired with one or multiple natural language expressions handwritten by expert annotators, where verbalizations express the semantics of the corresponding data units. For instance:

Data unit:
```
(Ajoblanco country Spain)
(Ajoblanco ingredient Garlic)
```
Verbalization:
Garlic is an ingredient used in Ajoblanco which originates from the country of Spain.

The triples are extracted from 15 different DBpedia categories, including Food, City, and others. The authors selected a wide range of categories to create a resource with a high variety of data (Perez-Beltrachini et al., 2016). The data units contain triples with diverse types of relationships. Among these are chains, where the object of a triple becomes the subject of another triple. There are also siblings, where distinct triples share the same subject. Furthermore, certain data units exhibit mixed relationships, containing both sibling and chain-related triples within them (see Figure 1). The extraction of these triples with varied relationships aimed to capture a wide range of linguistic structures.

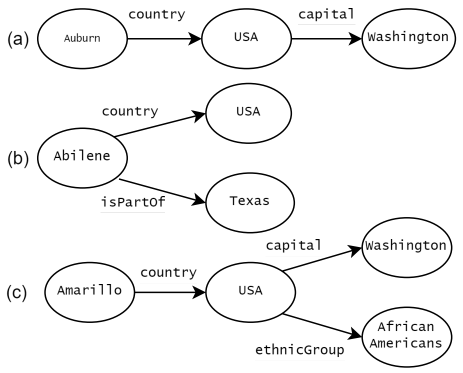

Figure 1: (a) The triples in the data unit are chain-related to each other. USA is the subject of the second triple and the object of the first one. (b) The relation between triples in the data unit is defined as sibling. Abilene is the subject of all the triples. (c) Some triples in the data unit are sibling-related, while others are chain-related, hence they are referred to as triples in a mixed relation

The latest version of English WebNLG is 3.0, released during the WebNLG challenge in 2020. This version contains $18,812$ data units with $47,195$ verbalizations. The corpus has been divided into training, development, and test sets, each consisting of data units containing 1 to 7 RDF triples.

5 The SGA Pipeline

Our work is based on the SGA pipeline, illustrated in Figure 2, which consists of three main steps: Data Unit Splitting (S), RDF-to-Text generation (G), and Sentence Aggregation (A). While the first step is based on a symbolic deterministic algorithm, the second and third steps rely on LLMs. We chose this modular approach to mitigate the "hallucinations" of LLMs' holistic approach. This was done because we hypothesize that as the amount of input data increases, the performance of LLMs in terms of adequacy and fluency decreases. In the shared task, the provided data units contain up to seven triples. To address these issues, we simplified the problem by dividing the data units into separate sets of triples, which were then verbalized through an RDF-to-Text system and unified using a Sentence

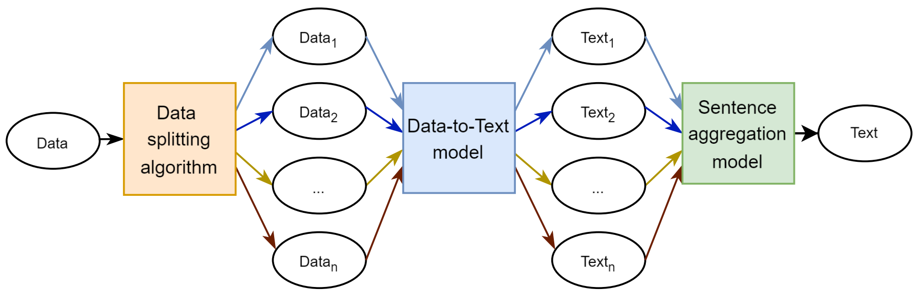

Figure 2: The SGA pipeline we propose begins with structured data, which is divided using a Data Unit Splitting Algorithm. Next, a RDF-to-Text System generates corresponding verbalizations, which are then unified using a Sentence Aggregation System.

Aggregation system.

5.1 Data Unit Splitting Description

Algorithm 1: Data Unit Splitting Algorithm

```
Data: Triples, Max triples per set
Result: Triples sets
subjects_dict = { } objects_dict = { }
foreach t in triples do
    subjects_dict[t.subj].append(t)
    foreach t1 in triples do
        if t.subj == t1.obj then
            objects_dict[t1.obj].append(t1)
        end
    end
end
merged_dict = { }
foreach subj, s_triples in subjects_dict do
    merged_dict[subj] = s_triples
    if subj in objects_dict then
        foreach o_triple in obj_triples[subj]
          do
            if not find(o_triple, merged_dict)
              then
                merged_dict[subj].append(o_triple)
            end
        end
    end
end
return generate_sets(merged_dict,
  max_triples);
```

As described in Section 4, the data units in the test sets provided by the GEM Shared Task could also have different shape types, representing various relationships between them, namely chain, sibling, and mixed type. To reduce the complexity of the data units, i.e., reducing the number of triples in each unit, the main idea is to divide data units into subsets of triples, with a maximum of three triples per set. To achieve this goal, we analyze the shape of each data unit to identify the relationships between triples. Unfortunately, the data units provided by GEM do not contain information about the shape type. Therefore, we created a Splitting algorithm to find the relationship between triples and divide them based on the retrieved information (cf. Algorithm 1). The Splitting algorithm processes triples within a data unit by storing those with identical subjects in `subjects_dict` and those whose objects appear as subjects in other triples in `objects_dict`. It splits each triple into subject, predicate, and object, using the subject as the key in `subjects_dict` and checking if the subject appears as an object in other triples. If so, those triples are added to `objects_dict`. After populating both `subjects_dict` and `objects_dict`, the algorithm merges these dictionaries into a unified structure called `merged_dict`. This involves copying entries from `subjects_dict` into `merged_dict`. For keys that are present in both dictionaries, the algorithm checks if any triple from `subjects_dict` is already listed under that key in `merged_dict`. If a triple is not found in the existing list, it is added to the list, capturing the chain relationships between triples. Once the dictionaries are merged, the algorithm addresses cases where any key in `merged_dict` contains more than three values. It splits these lists into chunks of up to three items each, dividing the triples based on their order. For instance, if there are four triples, the first three are grouped into one subset, while the fourth is placed in a separate subset. This method ensures that no list becomes too large, making the data easier to process and analyze. The choice of three as the chunk size is based on a qualitative analysis of the results from the SGA pipeline. For example, given this data unit with four triples:

```
(Trafford ground Estadio_Hirschi)
(Estadio_Hirschi location Itamarati)
(Trafford league League_One)
(League_One country USA)
```

The dictionaries obtained by the splitting algorithm are:

```
subjects_dict: [
  "Trafford": [
    "Trafford ground Estadio_Hirschi",
    "Trafford league League_One",
  ],
  "Estadio_Hirschi": [
    "Estadio_Hirschi location Itamarati",
  ],
  "League_One": [
    "League_One country USA"
  ]
]
objects_dict: [
  "Estadio_Hirschi": [
    "Trafford ground Estadio_Hirschi",
  ],
  "League_One": [
    "Trafford league League_One"
  ]
]
merged_dict: [
  0: [
    "Trafford ground Estadio_Hirschi",
    "Trafford league League_One",
  ],
  1: [
    "Estadio_Hirschi location Itamarati",
  ],
  2: [
    "League_One country USA"
  ]
]
```

The resulting subsets are:

Set 1:
```
Trafford ground Estadio_Hirschi
Trafford league League_One
```
Set 2:
```
Estadio_Hirschi location Itamarati
```
Set 3:
```
League_One country USA
```

For each set, the corresponding verbalizations will be generated using the approach described in the next section.

5.2 RDF-to-Text Description

Addressing the challenge of converting RDF data into natural language, and following the state-of-the-art, we built an RDF-to-Text system by fine-tuning a LLM, using the English version of WebNLG 3.0 for training. Before fine-tuning, we preprocessed the corpus by removing vertical bars between triple elements, sorting the triples alphabetically by predicate, and then concatenating them. For instance:

Data unit:
```
(Trafford | league | League_One)
(Trafford | nickname | Steve_Bright)
```
Pre-processed data unit:
```
(Trafford league League_One
Trafford nickname Steve_Bright)
```

We chose to fine-tune two different LLMs, specifically LLaMA-2 and Mistral, to evaluate their performance and selected the best-performing model for the GEM Shared Task. We opted for the 7 billion parameter versions of LLaMA-2 and Mistral models.[7][8] Moreover, we used a QLoRA quantization technique (Dettmers et al., 2023) to simplify the fine-tuning process and reduce the computational impact, using the following parameters: the LoRA attention dimension ($lora\_r$) was set to 64, the alpha parameter for LoRA scaling ($lora\_alpha$) was set to 16, and the dropout probability for LoRA layers ($lora\_dropout$) was set to 0.1. Additionally, we fine-tuned the models using only 20% of the dataset. Our training set comprised $7,085$ examples, and the development set included 893 instances. In both models, a single training epoch and a batch size of 4 were used. Furthermore, the following hyperparameters were employed: the maximum gradient norm for gradient clipping was set to 0.3, the initial learning rate for the AdamW optimizer was set to 2×10^{-4}, the weight decay applied to all layers except bias/LayerNorm weights was set to 0.001, and the optimizer used was "paged_adamw_32bit". The learning rate schedule followed a cosine pattern, with the number of training steps set to -1, and a linear warmup ratio of 0.03.

5.3 Sentence Aggregation Description

In this phase, we show how we aggregated the sentences generated in the previous step (RDF-to-Text) to achieve the final verbalization. This was accomplished using an LLM zero-shot prompting technique. After a qualitative assessment of the output with different prompts, the chosen one was:

> *"Instruction="You have to aggregate and paraphrase together the following sentences. You have to generate the result in Italian."*
> *Input: Text1: "...", Text2: "...", ...*
> *Output:""'*

We filled this prompt with the texts generated in the previous step, where verbalizations for each

[7] https://huggingface.co/meta-llama/Llama-2-7b

[8] https://huggingface.co/mistralai/Mistral-7B-v0.1

subset of triples were created. For the GEM Shared Task, after a qualitative evaluation of the performance of Mistral-7B and LLaMA-2-7B in the SGA pipeline, we chose to use the former model for both the RDF-to-Text and sentence aggregation steps.

6 Results

In this section, we present the results provided by the organizers for the various systems participating in the GEM Shared Task, evaluated on different subsets of data. The performance of these systems are compared using metrics such as BLEU, METEOR, chrF++, and Bert F1 for the English tasks. The evaluation was conducted using 180 selected data points, each associated with a single reference text. It is important to note that the use of only one reference per data point might lead to lower scores compared to evaluations with multiple references or a larger number of data points.

D2T-1-FA The D2T-1-FA subtask consists of data units directly extracted from the WebNLG test set. In this task, the DipInfo-UniTo system demonstrated excellent performance with a BLEU score of 32.31, making it the top system in this metric. Additionally, it achieved great results across other metrics, ranking among the best systems for this task (see Table 1).

D2T-1-CFA This subtask involves switching entities in the data units extracted from WebNLG (e.g., replacing a person entity with another person entity, a date with another date, etc.). DipInfo-UniTo achieved the highest scores across all metrics, surpassing the other participants by a significant margin, making it the best system for this task (see Table 2).

D2T-1-FI In the D2T-1-FI subtask, which is the most challenging of all the D2T-1 dataset, data units were first extracted from WebNLG and then modified with entities generated by an LLM. The DipInfo-UniTo system achieved the highest scores across all metrics, maintaining a significant lead over the other systems, like in the previous subtask (see Table 3).

D2T-2-FA This subtask involves data units directly extracted from Wikidata. The DipInfo-UniTo system achieved the highest score on the Bert F1 metric (0.937) and ranked as the second-best system in the other metrics, just behind SaarLST (see Table 4)

D2T-2-CFA The D2T-2-CFA subtask features data units extracted from Wikidata with swapped entities. The DipInfo-UniTo system achieved a BLEU score of 32.01, the highest among all systems. It was also the second-best in the other metrics, showcasing its strong performance. Specifically, SaarLST outperformed DipInfo-UniTo in METEOR and chrF++, while DCU-NLG-PBN excelled in the Bert F1 metric (see Table 5).

D2T-2-FI Finally, the D2T-2-FI subtask involves data units from Wikidata with entities generated by an LLM. The DipInfo-UniTo system demonstrated strong performance, achieving a BLEU score of 21.26, the highest among all participants. It also ranked second in the other metrics for this task, being outperformed by SaarLST in METEOR and chrF++, and by DCU-NLG-PBN in the Bert F1 metric (see Table 6)

In conclusion, the DipInfo-UniTo system has proven to be highly competitive across all tasks, frequently achieving the highest scores among participants and only falling slightly short in other cases, demonstrating excellent generalization ability across various datasets.

7 Conclusion

The main objective of this work was to enhance the performance of LLMs in RDF-to-Text generation. To achieve this, we employed NLG techniques with LLMs to develop the SGA pipeline designed to simplify the task and improve the quality of the outputs. To demonstrate the effectiveness of this technique, we compared the performance of LLaMA-2 and Mistral models both with fine-tuning and within the SGA pipeline. The results show that our approach improves performance on the RDF-to-Text task. The developed system demonstrated strong competitiveness across all tasks in the GEM 2024, achieving the highest scores in some cases while narrowly missing out in others. This performance underscores its good ability to generalize across various datasets. Future work could involve refining this technique by fine-tuning the models to better specialize in sentence aggregation, developing a more sophisticated data splitting algorithm, and integrating additional NLG techniques to produce more fluent and accurate text.

8 Limitations

The main limitation of our work was the limited computational resources available. To achieve better results, it would be necessary to use the entire WebNLG 3.0 corpus for fine-tuning the models, employ larger LLMs, and analyze performance by adjusting training hyperparameters to identify the configurations that yield the best performance.

References

Liam Cripwell, Anya Belz, Claire Gardent, Albert Gatt, Claudia Borg, Marthese Borg, John Judge, Michela Lorandi, Anna Nikiforovskaya, and William Soto Martinez. 2023. The 2023 webnlg shared task on low resource languages. overview and evaluation results (webnlg 2023). In *Association for Computational Linguistics*, page 55–66.

Tim Dettmers, Artidoro Pagnoni, Ari Holtzman, and Luke Zettlemoyer. 2023. Qlora: Efficient finetuning of quantized llms. *Preprint*, arXiv:2305.14314.

Thiago Castro Ferreira, Diego Moussallem, Emiel Krahmer, and Sander Wubben. 2018. Enriching the webnlg corpus. In *Association for Computational Linguistics*, page 171–176.

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. jiang 7b. *Preprint*, arXiv:2310.06825.

Nalin Kumar, Saad Obaid Ul Islam, and Ondrej Dusek. 2023. Better translation + split and generate for multilingual RDF-to-text (WebNLG 2023). In *Proceedings of the Workshop on Multimodal, Multilingual Natural Language Generation and Multilingual WebNLG Challenge (MM-NLG 2023)*, pages 73–79, Prague, Czech Republic. Association for Computational Linguistics.

Simon Mille, João Sedoc, Yixin Liu, Elizabeth Clark, Agnes Axelsson, Miruna-Adriana Clinciu, Yufang Hou, Saad Mahamood, Ishmael Obonyo, and Lining Zhang. 2024. The 2024 GEM shared task on multilingual data-to-text generation and summarization: Overview and preliminary results. In *Proceedings of the 17th International Conference on Natural Language Generation: Generation Challenges*, Tokyo, Japan. Association for Computational Linguistics.

Laura Perez-Beltrachini, Rania Sayed, and Claire Gardent. 2016. Building rdf content for data-to-text generation. In *The COLING 2016 Organizing Committee*, pages 1493–1502.

Anastasia Shimorina, Elena Khasanova, and Claire Gardent. 2019. Creating a corpus for russian data-to-text generation using neural machine translation and post-editing. In *Association for Computational Linguistics*, pages 44–49.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023. Llama 2: Open foundation and fine-tuned chat models.

Qingyun Wang, Semih Yavuz, Xi Victoria Lin, Heng Ji, and Nazneen Rajani. 2021. Stage-wise fine-tuning for graph-to-text generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 16–22, Online. Association for Computational Linguistics.

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-ADAPT-modPB	30.78	0.332	0.555	**0.935**
DCU-NLG-PBN	29.08	0.33	0.555	0.933
DCU-NLG-Small	27.0	0.314	0.537	0.93
DipInfo-UniTo	**32.31**	0.346	0.58	0.933
OSU-CompLing	30.03	0.335	0.566	0.932
RDFpyrealb	26.37	0.331	0.551	0.928
SaarLST	29.7	**0.347**	**0.581**	0.931

Table 1: Metrics scores on the D2T-1-FA English task (1 reference text per data point).

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-ADAPT-modPB	26.98	0.299	0.515	0.924
DCU-NLG-PBN	25.2	0.297	0.513	0.923
DCU-NLG-Small	22.98	0.279	0.488	0.918
DipInfo-UniTo	**29.01**	**0.315**	**0.543**	**0.926**
OSU-CompLing	24.45	0.293	0.514	0.92
RDFpyrealb	21.67	0.291	0.495	0.918
SaarLST	23.48	0.307	0.524	0.921

Table 2: Metrics scores on the D2T-1-CFA English task (1 reference text per data point).

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-ADAPT-modPB	26.54	0.318	0.539	0.921
DCU-NLG-PBN	26.02	0.322	0.549	0.92
DCU-NLG-Small	20.85	0.292	0.507	0.914
DipInfo-UniTo	**28.24**	**0.342**	**0.587**	**0.924**
OSU-CompLing	21.44	0.306	0.537	0.915
RDFpyrealb	21.97	0.31	0.527	0.917
SaarLST	20.76	0.331	0.557	0.917

Table 3: Metrics scores on the D2T-1-FI English task (1 reference text per data point).

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-NLG-PBN	23.96	0.295	0.49	0.936
DCU-NLG-Small	19.48	0.26	0.438	0.925
DipInfo-UniTo	27.22	0.304	0.512	**0.937**
OSU-CompLing	24.97	0.295	0.496	0.934
RDFpyrealb	19.97	0.287	0.479	0.921
SaarLST	**28.25**	**0.32**	**0.538**	0.934

Table 4: Metrics scores on the D2T-2-FA English task (1 reference text per data point).

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-NLG-PBN	30.34	0.348	0.581	**0.937**
DCU-NLG-Small	24.9	0.3	0.51	0.923
DipInfo-UniTo	**32.01**	0.354	0.592	0.936
OSU-CompLing	27.06	0.334	0.567	0.93
RDFpyrealb	25.05	0.335	0.561	0.923
SaarLST	26.47	**0.359**	**0.597**	0.929

Table 5: Metrics scores on the D2T-2-CFA English task (1 reference text per data point).

System ID	BLEU	METEOR	chrF++	Bert F1
DCU-NLG-PBN	20.46	0.3	0.49	**0.924**
DCU-NLG-Small	16.88	0.267	0.442	0.914
DipInfo-UniTo	**21.26**	0.307	0.502	0.923
OSU-CompLing	16.9	0.282	0.475	0.917
RDFpyrealb	16.28	0.286	0.472	0.916
SaarLST	20.16	**0.315**	**0.518**	0.919

Table 6: Metrics scores on the D2T-2-FI English task (1 reference text per data point).