OSU CompLing at the GEM'24 Data-to-Text task

Alyssa Allen and **Ash Lewis** and **Yi-Chien Lin** and **Tomiris Kaumenova** and **Mike White**
{allen.2334, lewis.2799, lin.4434, kaumenova.1, white.1240} @osu.edu

Abstract

This paper details experiments conducted for completing the GEM 2024 Data-to-Text task for a WebNLG dataset (Gardent et al., 2017). We show that model performance varies greatly across English, Spanish, Chinese, and Russian. Data filtering was done with automatic model judgments via error detection, which performs differently per language. We report English and Spanish dev set results for a data filtering and knowledge distillation approach to generating natural language outputs for sets of triples across a variety of domains. Specifically, we compare three generation conditions: 1) few-shot prompting with ChatGPT (GPT4), 2) fine-tuning Llama2 on the unfiltered dataset, and 3) fine-tuning Llama2 on a filtered version of the dataset. Russian and Chinese efforts did not result in submissions due to inconsistent or incoherent translations being produced in either the data synthesis or final generation stages. We provide details on these shortcomings but largely focus on Spanish and English efforts that align with our task submissions. We ultimately submitted outputs in English and Spanish that were generated using a version of Llama2 fine-tuned on a filtered dataset.

1 Introduction

In the WebNLG 2020 Challenge, the OSU system (Li et al., 2020) and others achieved nearly flawless English performance by carefully fine-tuning a pretrained model on the training set, though performance in Russian was remarkably poor by comparison. Since then, large language models (LLMs) like OpenAI's ChatGPT, Anthropic's Claude, and Google's Gemini have exhibited increasingly remarkable performance on a wide variety of tasks in multiple languages using in-context learning (i.e., few-shot generation), potentially obviating the need for human annotated training data. In this work, we first examine ChatGPT's few-shot performance on this data-to-text task — converting WebNLG-style logical triples to text — in a variety of languages and find several limitations in cases with lower-resource languages. Coupling these limitations with the high computational and financial costs, as well as lack of consistent behavior across time, makes ChatGPT difficult to rely upon for such tasks across languages. In this work, we leverage the robust capabilities of ChatGPT but attempt to offset the aforementioned downsides via knowledge distillation with smaller, more economical open source models.

Following the example of Lewis and White (2023), we leverage few-shot prompting to first generate training data using the ChatGPT API, then to detect errors in the generated data and filter them out. The filtered and unfiltered datasets are then used to fine-tune a Llama-2 (Touvron et al., 2023) model for the data-to-text task. This strategy was employed for English, Spanish, Russian, and Chinese data but we ultimately found that it was not effective for the latter two.

Specifically, Chinese fine-tuning Llama2 (pretrained on Chinese) on the initial filtered and unfiltered datasets yielded inconsistent outputs. This paper therefore describe alternative data filtering approaches and initial dev set results.

On the other hand, Russian experimentation was the least fruitful as there were numerous errors in the data synthesis and filtering stages. This paper describes some of the obstacles faces for this language and details our efforts to resolve these issues.

This work shows that filtering synthesized training data (for Spanish and English conditions) is a promising strategy for generating fluent, informative, and concise outputs. Initial dev set results indicate that improvements to our filtration methods would ultimately maximize performance. By employing a knowledge distillation approach, we capitalize on ChatGPT's expertise while lowering overall cost of generation by using an open-source

Proceedings of the 17th International Natural Language Generation Conference: Generation Challenges, pages 100–111
September 23 –27, 2024.

model like Llama2.

Our work also describes shortcomings of LLMs in completing this task for Chinese and Russian. By describing various data filtering efforts, we show potential paths forward in generating fluent and grammatically correct outputs across languages.

2 Related Work

The strategy of knowledge distillation in which we use ChatGPT to generate synthetic data, apply automatic data filtering methods, and use the filtered data to train a downstream model is most closely related to the work of Lewis and White (2023) in which they use the same general approach to create a virtual tour guide for a museum. Kim et al. (2023) use GPT-3 to construct a conversational dataset for distilling a smaller T5 conversational model in a similar fashion, focusing on social conversation filtered for safety. Lewis and White's (2023) setting involves retrieval augmented generation (RAG) and both of these works involve chat functions, while our work here applies knowledge distillation and filtering in a data-to-text task.

Schneider et al. (2024) explore the viability of using LLMs, both with few-shot prompting alone and fine-tuning, for the task of semantic parsing in a conversational setting and find that the recent models are able to perform reasonably well on the task, though they investigate English only. Our work also explores performance on other languages.

Madaan et al. (2023) use a self-refine approach to refine synthetic data, which, while not immediately relevant to this work, aligns with our efforts to refine synthetic data to improve the overall training set. Other works that have utilized knowledge distillation for seq2seq models include Tang et al. (2019) and Chen et al. (2020), who both explore distilling BERT's bidirectional encoder knowledge into a seq2seq model for generation. Our method can also be viewed as the first step in an approach combining knowledge distillation with self-training, not unlike Heidari et al. (2021), who implement self-training using an acceptability classifier (Batra et al., 2021) and ultimately distill fine-tuned BART (Lewis et al., 2019) seq2seq models. More recent approaches also explores using LLMs in self-training with various refinements to how acceptability classifiers are incorporated (Gulcehre et al., 2023; Yuan et al., 2024).

3 Methodology

We used the WebNLG dataset (Gardent et al., 2017) to synthesize a training set, including factual, counterfactual, and fictional examples. We then experimented with few-shot prompting, data filtering, and knowledge distillation via fine-tuning.

Target languages were English, Spanish, Chinese, and Russian. This section provides an overview of methods that apply to all target languages. Section 4 details language-specific efforts that vary depending on problem cases. Section 4 for Spanish and English also provides automatic evaluations on dev set results and initial insights based on those findings.

3.1 Dataset Conditions

This describes how each training data for each task condition is generated. Factual examples include sets of triples given in the WebNLG dataset (Gardent et al., 2017) across a variety of domains. For fictional and counterfactual data synthesis, we followed processes and definition outlined by the GEM task description. Our tactics differ from Axelsson and Skantze's (2023) knowledge-graph approach in that we chose to leverage the same ideas of entity relationships as is present in the aforementioned knowledge-graph approach, but manually assigned relationship (which we'll call predicate types) in order to more quickly generate our training data for this task.[1]

Fictional and Counterfactual examples were created by first manually assigning a type for each unique predicate (or second element) in a factual data set triple. After each unique triple across domains in the dataset were identified, a sample triple was selected randomly for that predicate. We then used our best judgment to create general categories that reflect the type of element needed to accompany the predicate. For example, if the predicate is `birthDate`, the author doing the annotations would see a triple such as `Alan_Bean | birthDate | 1932-03-15`. The annotator then assigns predicate `birthDate` with type `Person, Location`.

A predicate type consists of a Subject and Object label, indicating the relationship between the first and third element within a triple with its predicate. This predicate type indicates that the first element

[1]Had the organizers made synthetic input triples available for development, or code to generate such triples, it would have made it possible to spend more of the limited development time on the text generation portion of the task.

(i.e., subject) needs to be a person and the third element (i.e., object) should be a location.

After each unique predicate (372 predicates) is assigned a type, a set of fictional elements for each subject and object is made. For each unique subject and object label, we prompt ChatGPT (gpt-3.5-turbo). From the example above, the prompts would be 'Generate 20 realistic sounding People' and 'Generate 20 realistic sounding Locations'. Using the subject and object labels for each generated list, fictional elements can be swapped into the original factual triple sets. See Appendix A for more details.

Counterfactual examples were created by using a similar method, except instead of prompting ChatGPT, the list of potential element replacements were created by keeping track of first elements with the same subject or object label. Any first element of a given type could be swapped with another first element of that type. The same was done for elements in the third position in the triple.

Due to time constraints, type assignments did not change across domains per predicate. Therefore, some counterfactual and fictional swaps yielded less intuitive triple sets. For example, `creator` is a predicate in two domains: 'Food' and 'ComicsCharacter'. If creator is given the predicate type `Food, Person` in the labeling process, the swap could result in one triple relating to food and the rest if the triple set referring to a show. This disconnect makes the model less likely to generate concise and fluent outputs. Ultimately, these examples were not noticeably evaluated differently by the model than more logical triple sets. These annotations were also used to gain a rough sense of auto-evaluation accuracy. Further refinement of our swapping processes could help elevate the quality of our training data.

3.2 Data Synthesis and Knowledge Distillation

Using the training data, we prompt ChatGPT (3.5-turbo or 4 depending on target language) to generate natural language sentences per triple set. Instructions were given in the target language (see Appendix B). Dataset triples were all in English. Table 1 shows training set sizes for each language. Unfiltered, synthesized training sets for English, Chinese, and Russian include 1500 examples. Spanish training set size was 8,643 examples (or 20% of all possible examples provided in the WebNLG dataset). The Spanish data is more robust than the other languages because we found initial success generating coherent outputs, but wanted to test how more data could improve model performance for Spanish. Early experimentation with Russian and Chinese was less successful and therefore more time needed to be spent on process versus data quantity. We acknowledge that OpenAI does not provide a way to produce outputs in a reproducible manner. To increase transparency in our process that uses ChatGPT outputs as training data, we made the synthetic data generated by ChatGPT publicly available on GitHub.[2]

Filtering should exclude only examples with poor quality. However, the automatic filtering method described in Section 3.3 is noisy, cutting out a lot of good examples along with bad ones. Thus, one would expect that the larger the size of the training set, the less it matters to mistakenly cut out some good items, as there remain plenty of others, leaving more room to see a benefit of using cleaner data with relatively fewer bad items remaining. Based on the data set sizes in Table 1, we expect filtering to work best on Spanish since it is nearly 6x larger than Chinese and English data sets.

We fine-tuned Llama2 on this data set to provide us with a baseline ahead of filtering (see Section 3.3). As discussed in Section 4, each language leveraged various pretrained versions of Llama2.

3.3 Data Filtering via Error Detection

Following data synthesis, we experimented with filtering the training data discussed in Section 3.2. We used ChatGPT (GPT 4) as an error detector where we asked the model to determine if a generated text faithfully and fluently corresponded to a given triple set (see Appendix C for full prompts).

To evaluate the error detection capabilities of ChatGPT, the first author manually annotated a small set of 99 examples across categories (factual, counterfactual, and fictional) for English and Spanish. 8 errors were found in Spanish and 9 errors were found in English.

Out of 99 Spanish examples, ChatGPT's (GPT4) performance as an error detector yielded 50% recall and 36% precision (see Table 2). Out of 99 English examples, error detection yielded 44% recall and 31% precision. These numbers exceed the actual error rate of about 10% in both languages which means the error detector performance is far from ideal but well above chance.

[2] https://github.com/nicalin/2024-GEM-data

Language	Unfilt. Size	Filt. Size	Filt. FA	Filt. CFA	Filt. FI
English	1,500	1,315	500	378	437
Chinese	1,500	1,172	463	324	385
Spanish	8,643	7,607	2,735	2,286	2,586

Table 1: Size of unfiltered and filtered training data sets. For filtered data, this table also shows how many examples were included across languages and per category — factual (FA), counterfactual (CFA), fictional (FI).

	Agreement (%)				Error Detection		
Language	Overall	FA	CFA	FI	Recall	Precision	F-Score
English	87	90	80	90	0.44	0.31	0.36
Spanish	88	91	76	97	0.50	0.36	0.42

Table 2: Agreement and error detection results for training data sets across languages and triple set categories — factual (FA), counterfactual (CFA), fictional (FI) — for model-based judgments.

As seen in Table 2, counterfactual cases had the lowest accuracy in both English and Spanish. Qualitatively, the model labeled correct counterfactual cases as incorrect when it wanted to correct the facts or if there were conflicting dates in the triple set. Consider the example in Figure 1.

In the above example, the generated text is faithful to the information in the triples. The evaluator is unable to ignore real-world knowledge. This output is deemed as incorrect because of factual contradictions.

Performance could likely be improved with more nuanced evaluation prompts or a refined process for creating the counterfactual triples. Time was a constraint in making these adjustments.

Fictional cases had higher agreement than expected due to fewer errors being in the fictional category compared to factual or counterfactual. Therefore, the agreement between model and author was primarily for correct cases instead of on errors.

Cases that passed the filter are added to a validated dataset. This version of the training data is then used to fine-tune Llama2. Russian experimentation did not successfully complete the filtering stage (discussed in Section 4.4). Outputs from Llama2 (fine-tuned on the filtered data) for English and Spanish were ultimately submitted.

4 Experiments

This section details language-specific efforts and offers further insights as to how model performance differed per language.

4.1 English

Experimentation with English on data synthesis and filtering followed the methods described in Section 3. This section describes dev set results for English. The dev set consists of 1,998 examples (666 per category). Our English submission for this task was generated using Llama2 fine-tuned on the filtered dataset.

4.1.1 Automatic Evaluation

Since our dev set did not have gold standard outputs, we could not use reference-based methods for automatic evaluation. We instead employed the same error detection method as we used for filtering (Section 3.3) in order to compare experimental methods within a target language. Few-shot prompting with Llama2 yielded inconsistent outputs. For example, common output errors were regurgitating the triple set and printing '\n' repeatedly. These errors were not present in the fine-tuning conditions. Table 3 shows the percentage of 'good' outputs per category. As shown in Table 3, dev set results are nearly equal for unfiltered and filtered Llama2 fine-tuning conditions with 77% and 75% (respectively) of the outputs being judged as faithful to their corresponding triple sets.

4.1.2 Analysis

To quickly check efforts during the task, we performed an automatic evaluation on a small dev set of about 120 items that led us to believe a filtering approach would yield higher quality results. Results shown in Table 3 were conducted after the task deadline. Factual cases across conditions outperformed counterfactual and fictional categories. Counterfactual cases across conditions showed the

TRIPLES: `France | country | Jinnah_International_Airport`
GENERATED TEXT: Francia es el país donde se encuentra ubicado el Aeropuerto Internacional Jinnah.
ENG TRANSLATION: France is the country where Jinnah International Airport is located.
CHATGPT JUDGMENT: Bad. The triple suggests that Jinnah International Airport is in France, which is incorrect. Jinnah International Airport is in Pakistan, not in France.
ANNOTATOR JUDGMENT: Good.

Figure 1: Example of a generated output in Spanish that is considered by the authors to be a 'good' example, but the evaluator deemed it 'bad'. As seen in the CHATGPT JUDGMENT, the evaluator labels the output as incorrect due to the counterfactual nature of the triple.

	'good' Judgments (%)			
Condition	**Overall**	**FA**	**CFA**	**FI**
Eng_Unfiltered_LL	77	91	65	74
Eng_Filtered_LL	75	88	65	72
Eng_FewShot_GPT	84	97	72	83

Table 3: Automatic evaluation English dev set results for filtered (Llama2), unfiltered (Llama2), and few shot (GPT4) conditions. Percentages represent how many generated outputs were judged as faithful to its triple set input. Results are shown overall and across categories Factual (FA), Counterfactual (CFA), and Fictional (FI).

	'good' Judgments (%)			
Condition	**Overall**	**FA**	**CFA**	**FI**
Esp_Unfiltered_LL	79	92	71	75
Esp_Filtered_LL	81	92	72	77
Esp_FewShot_GPT	89	98	82	88

Table 4: Automatic evaluation for Spanish dev set for filtered (Llama2), unfiltered (Llama2), and few shot (GPT4) conditions. Percentages represent how many generated outputs were judged as faithful to its triple set input. Results are shown overall and across categories Factual (FA), Counterfactual (CFA), and Fictional (FI).

worst performance. Both Llama2 conditions performed worse than ChatGPT.

Further experimentation with filtering is needed to improve over an unfiltered fine-tuning approach. Overall, filtering the data slightly decreased 'good' judgments of English outputs, from 77% to 75%. As mentioned in Section 3.2, the smaller the training dataset is, the more a noisy filtering method will likely hinder performance results. Therefore, the filtering method was not as effective on the smaller English dataset as it was on the larger Spanish dataset (see Table 1).

We expect that a more carefully orchestrated filtration methodology could yield better results and further work explore better ways to utilize LLMs for this task via prompting or perhaps by other fine-tuning strategies. Based on results in Section 4.2.1, where the filtering condition improved upon the unfiltered condition, we expect that increasing the training data size for English would improve results for the filtered condition.

Furthermore, we find that using ChatGPT as an evaluator is helpful for obtaining rapid judgments but is not fully reliable. Table 3.3 shows that agreement with an annotator (the first author) is high but imperfect, particularly in the counterfactual cases. Because we used ChatGPT in a similar way for our filtration methodology, it follows that it too is imperfect. Figure 2 shows an example of disagreement between annotator and model judgments for a counterfactual case.

Because we found lower agreement with author judgments for the CFA items (see Table 2), we expect the error rate to be somewhat inflated in these automatic evaluation results shown in Table 4. That said, we still expect the CFA items to be the most difficult, and cursory inspection reveals plenty of real errors in the text, as seen in Figure 2. Further experimentation with explicitly prompting the system to ignore real-world facts could lead to improved CFA case results.

4.2 Spanish

Experimentation with Spanish followed the methods described in Section 3. We used a version of Llama2 pretrained on Spanish[3] for all Spanish ex-

[3] https://huggingface.co/clibrain/Llama-2-7b-ft-instruct-es

TRIPLE: Chicago | isPartOf | Linn_County,_Oregon
TEXT: Chicago is part of Linn County, Oregon.
CHATGPT JUDGMENT: Bad. There is a geographical inaccuracy as Chicago is not part of Linn County, Oregon; it is in fact, a city in Illinois. Therefore, the information provided in the text contradicts established facts and cannot be corrected without changing the original triples.
ANNOTATOR JUDGMENT: Good.

Figure 2: Example of a generated output in English that is considered by the authors to be a 'good' example, but the evaluator deemed it 'bad'. As seen in the CHATGPT JUDGMENT, the evaluator labels the output as incorrect due to the counterfactual nature of the triple.

periments. Our Spanish submission for this task was generated using Llama2 fine-tuned on the filtered dataset.

This section describes the Spanish dev set results. As described in Section 3.1, factual inputs are compiled from the WebNLG dev set (Gardent et al., 2017) and we synthesized examples for fictional and counterfactual categories. Our dev set consists of 1,998 examples (666 per category).

4.2.1 Automatic Evaluation

Error detection (described in Section 3.3) was used for data filtering. Few-shot prompting with Llama2 yielded inconsistent outputs. As seen in Table 4, fine-tuning with our full training set yielded 79% 'good' judgments during the automatic evaluation. Fine-tuning on a filtered dataset yielded 81% 'good' judgments. Using ChatGPT (GPT4) to generate outputs yielded 89% 'good' judgments.

Unlike the English case, this method resulted in some improvement for counterfactual and fictional conditions, thereby increasing the overall accuracy in Spanish.

4.2.2 Analysis

Similar to the process mentioned in Section 4.1.2, this evaluation was conducted on a substantial dev set after the deadline for this task. Within the scope of the task, we conducted an initial automatic evaluation on a small dev set of about 90 examples, where results were comparable for the two fine-tuning conditions.

Factual cases yielded the highest percentage of 'good' cases across conditions. Counterfactual cases were also the worst performing across conditions. This trend is consistent with trends found in English results, see Table 4.

Results for the GPT condition shown in Table 3 and Table 4 are surprising given English's presumed greater prevalence in ChatGPT's training data. The reason for Spanish results outperforming English results is unknown.

Of note, as mentioned in Section 3.2, the Spanish training dataset was significantly larger than the training set used for any other language (see Table 1). This increase in training data could have led to overall improvement in filtered (79%) vs. non-filtered (81%) conditions for Spanish, but not English. To further improve filtered condition performance, further refinement of filtration methodology is needed. More training data could also improve results for both filtered and unfiltered model conditions.

4.3 Chinese

For Chinese, we used the same data synthesis methods as described in Section 3. We ultimately attempted to fine-tune a version of Llama2 trained on Chinese (Zefeng Du, 2023) with filtered Chinese data, but were unsuccessful in generating consistent, high-quality results. One recurring issue with dev set results was that Llama2 would provide additional narration to the outputs such as: '好的，让我来给您介绍一下... (English translation: Okay, let me introduce you to...)'.

Due to the time constraints, we were not able to investigate the cause of this issue or otherwise improve our fine-tuned model. While we did not submit Chinese results to this task, this section details experimentation with data filtering for Chinese.

4.3.1 Chinese Data Synthesis and Filtering

As mentioned in Section 3, we used ChatGPT (GPT 4) for data synthesis and filtering via few-shot prompting. In addition to generating one Chinese output per triple set (One-to-One), we experimented with generating five Chinese outputs per triple set (One-to-Many), with the aim of increasing the size of the filtered dataset (see Section 3.3 for error detection and filtering methods).

We also experimented with different data filtering methods using GPT 4 for One-to-One and One-to-Many datasets. For One-to-One, we experimented with two data filtering methods: (1) error detection and (2) reconstruction. For One-to-Many,

we experimented with likelihood rankings.

Filtering via Error Detection

Filtering via error detection leverages methods outlines in Section 3.3. We provided ChatGPT with a triple set and a Chinese output. We then prompted ChatGPT to judge outputs as 'good' or 'bad' (See Appendix C for full prompt). In addition to asking ChatGPT to provide judgments, we also experimented with two different settings: (1) asking ChatGPT to only provide the judgment (i.e., 'good' or 'bad') and (2) asking ChatGPT to provide a correction for any output labeled 'bad'.

Filtering via Reconstruction

For filtering via reconstruction, we employed ChatGPT as a reverse model to reconstruct the English triple set from a given Chinese synthesized output. In principle, if the reconstructed English triple sets are overly different (see Section 4.3.2) from the corresponding original triple sets, the synthesized outputs are likely to have poor quality and should be discarded.

Filtering via Likelihoods

For the One-to-Many dataset, we used the ChatGPT `logprobs` parameter to assign a pair likelihood scores to each output per triple set. Specifically, we treated this process as a binary classification task. Each candidate output per triple set received two log probability scores (likelihood that output is labeled 'good' and likelihood that output is labeled 'bad'). The output with the highest 'good' score compared to other candidate outputs for a given triple set was kept as the output for that triple set. An output was selected at random if multiple candidates received the same score.

4.3.2 Automatic Evaluation

Automatic evaluation for each filtering method discussed in Section 4.3.1 follows error detection methods as described in Section 3.3. In order to justify automatic evaluation results, we compare model judgments with author judgments. For each data filtering method, we manually annotated 30 examples to compare filtering method performance. The sets of annotated examples differ per filtering method. Therefore, the comparisons in Table 5 are not directly comparable.

We excluded the odd cases in the 30 examples, resulting in 29, 30, 29, and 30 annotated examples respectively for model judgment with correction (Judge/Corr), model judgment without correction (Only_Judge), reconstruction, and likelihood. Table 5 shows precision, recall, and agreement resulting from author judgment compared to ChatGPT's judgment.

Judge/Corr and Only_Judge model outputs are synonymous with filtering method judgments. For reconstruction, we did not reach a stage of automatic filtering based on similarity of the reconstructed triple set to the original triple set. Therefore, the annotator manually judged each reconstructed triple set. Reconstructed triple sets deemed as faithful to the original triple set are expected to yield faithful natural language outputs as well. These judgments on the reconstructed triples (i.e., acting as pseudo–model judgments) are compared to the original author annotations.

For filtering via likelihood, an output is labeled 'bad' if it has a higher likelihood for 'bad' instead of 'good'. Similarly, the output is labeled 'good' if it has a higher likelihood for 'good' instead of 'bad'. These judgments serve as the model judgments to compare with the author annotations.

Table 5 shows results of model agreement with the manual annotations. As shown in Table 5, filtering with reconstruction yielded the best precision and recall, albeit with manual similarity judgment of triple sets.

4.3.3 Analysis

In the Judge/Corr condition, outputs which were incorrectly judged as 'bad' were either counterfactual or fictional. Cases judged incorrectly as 'good' were most often due to missing information in the triple sets, hallucination, or incorrectly representing the relation.

In the Only_Judge condition, the outputs incorrectly judged as 'bad' by ChatGPT were counterfactual and were often due to the model fact-checking the output. Outputs with incorrect 'good' judgments were mostly due to incorrect or disfluent translations of the relations, which were not detected by ChatGPT.

In the reconstruction condition, outputs which were incorrectly judged as 'bad' were across all three categories, mainly due to the reconstructed triple set missing information. The reconstruction condition also incorrectly judged some outputs as 'good', mainly because the reconstructed triple sets were not able to show disfluency and incorrect translation in the synthesized sentences.

In order to automate the reconstruction filtering

Filtering Method	Precision	Recall	Agreement (%)
Judge/Corr	0.33	0.50	79
Only_Judge	0.50	0.40	83
Reconstruction	0.63	0.56	76
Likelihood	0.33	0.25	83

Table 5: Precision, recall, and agreement of the different data filtering methods for the Chinese data. Precision and recall are for error detection. Judge/Corr, Only_Judge, and Reconstruction methods are One-to-One. Likelihood method is One-to-Many.

process, we would need a process for comparing the reconstructed triple set to the original triple set. Additionally, an added complication of Chinese is that some Chinese terms can have multiple English translations. Therefore, translating from English to Chinese and then back to English leaves increasing room for error. Using LLMs to automating the process of comparison could be a potential direction to explore in future work.

Lastly, for the likelihood condition, the final candidates incorrectly detected as errors are either counterfactual or fictional. The main error found in this case was incorrect translations and missing information.

Due to time constraints, data filtering via error detection (without correction) was the filtering method ultimately used to create the filtered Chinese training data for fine-tuning. However, given more time to refine processes, filtering via reconstruction could potentially improve model performance.

4.4 Russian

We employed few-shot prompting to synthesize Russian outputs for the training data triple sets. This method involved providing GPT-4 with 5 examples (1 factual, 2 counterfactual and 2 fictional) in Russian. The model generated sentences often exhibited unnatural phrasing, incorrect case endings, as well as inconsistencies and inaccuracies in translating proper names (see Appendix D).

Grammatical issues persisted after attempting to repair errors via automatic judgment and error correction. We provided ChatGPT with a few-shot prompt where the examples included faulty outputs, suggested corrections, and then the corrected version of the outputs. We also prompted ChatGPT to make judgments without repairing the errors. Instructions were given in Russian. The model failed to consistently detect and correct errors in the synthesized data. Low model performance indicates the need for more sophisticated techniques and training datasets to improve the quality of Russian text generation in future work.

4.4.1 Analysis

Our primary approach for synthesizing Russian data was few-shot prompting which did not yield satisfactory results. The attempt to use model-generated judgments to filter and correct outputs did not sufficiently mitigate the issue of fluency. In future work, advanced translation models could assist in synthesizing Russian data by translating English outputs. Additionally, further experimentation with fine-tuning approaches using a small manually constructed set of Russian examples could improve the quality of generated sentences.

5 Preliminary English Results

The task organizers provided preliminary results for the English data set (Mille et al., 2024). Results are for factual, counterfactual, and fictional cases for the WebNLG test set (i.e., the 'seen' subtask) and the WikiData test set (i.e., the 'unseen' subtask).

As expected, our team's best performing category was WebNLG FA with a BLEU score of 30.03. WebNLG FI performed worse than WebNLG CFA with BLEU scores of 21.44 and 24.45, respectively.

The seen results for FA and FI results were higher than the unseen ones in the same categories with BLEU scores of 24.97 for unseen FA and 16.9 for unseen FI cases.

Given aforementioned challenges with our CFA training data in terms of potentially nonsensical triple sets, it is not surprising that CFA unseen BLEU score (27.06) outperformed the seen BLEU score (24.45).

Fictional cases were our lowest performing for both subtasks. This may be due to the system's desire to adhere to real-world facts. As mentioned in Section 4.1.2, more explicit prompting could have led to improved performance.

Compared to other systems that completed this data-to-text task, our system ranked in the mid-

dle across categories. A difference that potentially led other systems to outperform our system is that we opted out of using existing supervised training data. Instead, we chose to use a limited amount of few-shot synthetic data generated by ChatGPT. Because our efforts expanded across English, Spanish, Chinese, and Russian, we avoided tactics that may give English an advantage over the other languages of interest (for which the same supervised training data was not available). We could have also experimented with generating more few-shot synthetic data and using self-training methods to improve our system's performance but did not due to expense and lack of time.

While we did not have access to Spanish results at the time of publishing, we expect overall improved performance due to increased training data set size compared to the English training set.

6 Discussion and Conclusion

For all languages, newer state of the art models such as Llama3 (AI@Meta, 2024) could have improved output performance — specifically in the cases of Russian and Chinese where incorrect translations were a large portion of errors qualitatively found.

We also chose to not mention in the evaluation prompts how to handle non-factual triple sets and corresponding output being tested. The evaluations for counterfactual and fictional cases could potentially have been improved if the prompt included explicit instructions to ignore any non-factual information and focus on the representation of the triple sets in the output.

A data synthesis and knowledge distillation approach yielded promising results in English and Spanish. ChatGPT was successful in synthesizing training data, but was less-than-ideal in acting as an evaluator — particularly for Chinese. ChatGPT was less successful in generating usable synthetic data for Russian.

Further work could focus on refining our approach to data filtering and experiment with self-training, which could potentially yield results comparable to or even exceeding few-shot prompting with ChatGPT using a cheaper and more reliable open source model. Based on the experiments presented in this paper, we see that LLMs perform better on data-to-text tasks for higher resource languages (English and Spanish) and struggle with others (Chinese and Russian).

References

AI@Meta. 2024. Llama 3 model card.

Agnes Axelsson and Gabriel Skantze. 2023. Using large language models for zero-shot natural language generation from knowledge graphs. *Preprint*, arXiv:2307.07312.

Soumya Batra, Shashank Jain, Peyman Heidari, Ankit Arun, Catharine Youngs, Xintong Li, Pinar Donmez, Shawn Mei, Shiunzu Kuo, Vikas Bhardwaj, Anuj Kumar, and Michael White. 2021. Building adaptive acceptability classifiers for neural NLG. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 682–697, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Yen-Chun Chen, Zhe Gan, Yu Cheng, Jingzhou Liu, and Jingjing Liu. 2020. Distilling knowledge learned in BERT for text generation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7893–7905, Online. Association for Computational Linguistics.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for nlg micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 179–188. Association for Computational Linguistics.

Caglar Gulcehre, Tom Le Paine, Srivatsan Srinivasan, Ksenia Konyushkova, Lotte Weerts, Abhishek Sharma, Aditya Siddhant, Alex Ahern, Miaosen Wang, Chenjie Gu, Wolfgang Macherey, Arnaud Doucet, Orhan Firat, and Nando de Freitas. 2023. Reinforced self-training (rest) for language modeling. *Preprint*, arXiv:2308.08998.

Peyman Heidari, Arash Einolghozati, Shashank Jain, Soumya Batra, Lee Callender, Ankit Arun, Shawn Mei, Sonal Gupta, Pinar Donmez, Vikas Bhardwaj, Anuj Kumar, and Michael White. 2021. Getting to production with few-shot natural language generation models. In *Proceedings of the 22nd Annual Meeting of the Special Interest Group on Discourse and Dialogue*, pages 66–76, Singapore and Online. Association for Computational Linguistics.

Hyunwoo Kim, Jack Hessel, Liwei Jiang, Peter West, Ximing Lu, Youngjae Yu, Pei Zhou, Ronan Le Bras, Malihe Alikhani, Gunhee Kim, Maarten Sap, and Yejin Choi. 2023. Soda: Million-scale dialogue distillation with social commonsense contextualization. *Preprint*, arXiv:2212.10465.

Ashley Lewis and Michael White. 2023. Mitigating harms of LLMs via knowledge distillation for a virtual museum tour guide. In *Proceedings of the 1st Workshop on Taming Large Language Models: Controllability in the era of Interactive Assistants!*, pages 31–45, Prague, Czech Republic. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Xintong Li, Aleksandre Maskharashvili, Symon Jory Stevens-Guille, and Michael White. 2020. Leveraging large pretrained models for WebNLG 2020. In *Proceedings of the 3rd International Workshop on Natural Language Generation from the Semantic Web (WebNLG+)*, pages 117–124, Dublin, Ireland (Virtual). Association for Computational Linguistics.

Aman Madaan, Niket Tandon, Prakhar Gupta, Skyler Hallinan, Luyu Gao, Sarah Wiegreffe, Uri Alon, Nouha Dziri, Shrimai Prabhumoye, Yiming Yang, Shashank Gupta, Bodhisattwa Prasad Majumder, Katherine Hermann, Sean Welleck, Amir Yazdanbakhsh, and Peter Clark. 2023. Self-refine: Iterative refinement with self-feedback. *Preprint*, arXiv:2303.17651.

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.

Phillip Schneider, Manuel Klettner, Kristiina Jokinen, Elena Simperl, and Florian Matthes. 2024. Evaluating large language models in semantic parsing for conversational question answering over knowledge graphs. *Preprint*, arXiv:2401.01711.

Raphael Tang, Yao Lu, and Jimmy Lin. 2019. Natural language generation for effective knowledge distillation. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 202–208, Hong Kong, China. Association for Computational Linguistics.

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. *Preprint*, arXiv:2307.09288.

Weizhe Yuan, Richard Yuanzhe Pang, Kyunghyun Cho, Xian Li, Sainbayar Sukhbaatar, Jing Xu, and Jason Weston. 2024. Self-rewarding language models. *Preprint*, arXiv:2401.10020.

Longyue Wang Zefeng Du, Minghao Wu. 2023. Chinese-llama-2. `https://github.com/longyuewangdcu/Chinese-Llama-2`.

7 Appendices

A Fictional Element Prompts

The following prompts showcase the process of creating fictional examples.

Example Predicate types

Examples are of the form - predicate: [Subject label, Object label] foundingDate: [Company, Date] author: [Book, Author] starring: [Film, Person]

Example Prompt

Each label is then substituted into the following prompt 'Generate 20 realistic-sounding...'.

Creating Fictional Examples

After prompting ChatGPT, each subject or object label has a list of options. For each example in the factual dataset, the appropriate labels are found per predicate and an item from the fictional element lists for that label is selected at random. Substitions are then made accordingly. For example:

Factual Triple

1. `Siomay | ingredient | Peanut sauce`
2. `Batagor | dishVariation | Siomay`

Predicate Types

1. ingredient: `Dish, Ingredient`
2. dishVariation: `Dish, Dish`

Fictional Triple

1. `Beef enchiladas with cilantro lime rice | ingredient | Almond flour`
2. `Penne pasta in creamy tomato sauce | dishVariation | Mediterranean Chickpea Salad`

B Generation Prompts

The following prompts were used for each language of interest. Native speakers wrote the English, Chinese, and Russian prompts. Spanish prompt was written by an L2 Spanish speaker.

Additionally, Chinese prompt includes both instructions and few-shot examples. For the Chinese prompt, only instructions are included here. For Russian, since our Russian prompt is a few-shot prompt consisting of 5 translated examples without explicit instructions, we do not list the prompt here.

English

Write a text version of the info in the input. The text should include all of the information in the triples, and only that information. The output should be fluent, grammatically correct, and concise.

Spanish

Analiza los siguientes ejemplos de tríos y textos. El texto debe incluir toda la información de los tríos y solo esa información. Además, el texto debe ser fluido, gramaticalmente correcto y conciso. Escribe el texto para el caso de prueba.

Chinese

请把以下每组三词词组转换成中文文字。请不要省略任何信息或是增加任何不必要的信息。有些三词词组可能不符合事实，但请还是把提供的三词词组转换成中文。

C Error Detection Prompts

The following instructions were used with ChatGPT to evaluate the generated texts. Prompts for English and Spanish evaluation were in English. Results were used to determine which cases pass the error detection filter.

English and Spanish

Examine the following triples and [Spanish] text. The [Spanish] text should include all the information in the triples without adding unnecessary information. The [Spanish] text should be fluent, grammatically correct, and as concise as possible. If the [Spanish] text is all correct, respond with 'Good.' If not, respond with 'Bad.' and explain the error and then rewrite the answer.

Chinese

For Chinese, there are four different prompts each filtering method (i.e., automatic judgment (with correction), automatic judgment (without correction), likelihood, and reconstruction). The instructions for the first three filtering methods are in Chinese while instruction for reconstruction is in English.

Error Detection with Correction

检查以下的内容以及正文。管有些容及正文可能不符合事，正文是应该包含所有内容里的资讯，并且正文不能包含不必要的资讯。正文必须流畅且文法正确。如果判断正文为完全正确，回覆'好'。如果有任何错误，回答'坏'，并且解释错误以及修改正文。

Error Detection without Correction

检查以下的内容以及正文。管有些容及正文可能不符合事，正文是应该包含所有内容里的资讯，并且正文不能包含不必要的资讯。正文必须流畅且文法正确。如果判断正文为完全正确，回覆'好'。如果有任何错误，回答'坏'。

Likelihood

请根据以下的内容检查每个句子。尽管有些内容及句子可能不符合事实，正文是应该包含所有内容里的资讯，并且正文不能包含不必要的资讯。正文必须流畅且文法正确。如果判断正文为完全正确，将句子归类到'好'的类别。如果有任何错误，将句子归类到'坏'的类别。

Reconstruction

Convert the following Chinese text into a set of English triples. The Chinese text might be counterfactual. Please still try to convert the Chinese text into English triples.

D Russian Errors

Russian

1. As shown in the example, the model exhibited unnatural phrasing.

   GENERATED TEXT: Эллиот Си был летчиком-испытателем и был выбран NASA в 1962 году. Он родился 23 июля 1927 года и умер 28 февраля 1966 года в Сент-Луисе. Сейчас его статус - умерший.
   ENG TRANSLATION: Elliot See was a test pilot and was selected by NASA in 1962. He was born on July 23, 1927 and died on February 28, 1966 in St. Louis. Currently his status is dead.
   CORRECTED TEXT: Эллиот Си был выбран в NASA

летчиком-испытателем в 1962 году. Он родился 23 июля 1927 года и умер 28 февраля 1966 года в Сент-Луисе.

2. Some words, especially proper nouns, were not properly translated.

   GENERATED TEXT: Эллиот Си, родившийся 23 июля 1927 года в Далласе, был гражданином США. Он занимался профессией тестового пилота. Си учился в Университете Техаса в Остине, который состоит в Большой 12 конференции. Эллиот Си скончался в Сент-Луисе.

   ENG TRANSLATION: Elliot See born on July 23, 1927 in Dallas was an American citizen. He worked as a test pilot. See studied at the University of Texas at Austin which is a part of Big 12 conference. Elliot See died in St. Louis.

   CORRECTED TEXT: Американец Эллиот Си, родившийся 23 июля 1927 года в Далласе, был летчиком-испытателем. Си учился в Университете Техаса в Остине, который входит в состав конференции Big 12. Эллиот Си скончался в Сент-Луисе.

3. Case endings were often generated incorrectly.

   GENERATED TEXT: Техническим кампусом института технологий Ачарьи был признан Всеиндийским советом по техническому образованию (AICTE), расположенным в Мумбаи. Институт находится в городе Бангалор, штат Карнатака, который граничит с Аравийским морем на западе. Институт Ачарьи аффилирован с Вишвесваройским технологическим университетом.

   ENG TRANSLATION: The Acharya Institute of Technology was recognized as the 'Technical Campus' by the All India Council for Technical Education in Mumbai. The institute is located in Bangalore, Karnataka, which borders the Arabian Sea to its west. The Acharya Institute is affiliated with the Visvesvaraya Technological University.

   CORRECTED TEXT: Институт технологий Ачарьи получил статус технического кампуса от Всеиндийского совета по техническому образованию (AICTE), расположенного в Мумбаи. Институт находится в городе Бангалор, штат Карнатака, который граничит с Аравийским морем на западе. Институт Ачарьи является филиалом Вишвесваройского технологического университета.