Structured Object Language Modeling (SoLM): Native Structured Objects Generation Conforming to Complex Schemas with Self-Supervised Denoising

Amir Tavanaei **Kee Kiat Koo** **Hayreddin Ceker**
Shaobai Jiang **Qi Li** **Julien Han** **Karim Bouyarmane**

Amazon, Seattle, USA
https://so-lm.github.io

Abstract

In this paper, we study the problem of generating structured objects that conform to a complex schema, with intricate dependencies between the different components (facets) of the object. The facets of the object (attributes, fields, columns, properties) can be a mix of short, structured, type-constrained facts, or long natural-language descriptions. The object has to be self-consistent between the different facets in the redundant information it carries (relative consistency), while being grounded with respect to world knowledge (absolute consistency). We frame the problem as a Language Modeling problem (Structured Object Language Modeling) and train an LLM to perform the task natively, without requiring instructions or prompt-engineering. We propose a self-supervised denoising method to train the model from an existing dataset of such objects. The input query can be the existing object itself, in which case the model acts as a regenerator, completing, correcting, normalizing the input, or any unstructured blurb to be structured. We show that the self-supervised denoising training provides a strong baseline, and that additional supervised fine-tuning with small amount of human demonstrations leads to further improvement. Experimental results show that the proposed method matches or outperforms prompt-engineered general-purpose state-of-the-art LLMs (Claude 3, Mixtral-8x7B), while being order-of-magnitude more cost-efficient.

1 Introduction

Following natural-language text generation and code generation by the state-of-the-art Large Language Models (LLMs) (Jiang et al., 2024; Reid et al., 2024; Floridi and Chiriatti, 2020; ANTHROP, 2024; Jiang et al., 2023), structured objects generation, also known as JSON (JavaScript Object Notation) generation or key-value pairs object generation, is a challenging problem for existing LLMs (Kitouni et al., 2024). It is one of the most desired behaviour of LLMs when used in production settings beyond traditional chatbot applications. It allows LLMs to be used as autonomous agents that integrate seamlessly with APIs, since JSON is the de-facto communication standard between APIs, and the universal string serialization format of structured objects. It also allows to use LLM outputs directly without any post-processing required, for example writing the output directly to a data store or passing it as input to subsequent functions. Finally, it allows to optimize LLM inference cost and number of calls and ensures self-consistency of the output by generating the entire object in a single LLM call, instead of generating each field of the object independently by an LLM query.

General-purpose instruction-following and human-intent-aligned LLMs (chatbots) can be steered towards generating JSON objects outputs by specifying the requirement in their instruction prompts. Multiple prompting techniques have been tried to ensure that the output is a valid JSON that conforms to a schema, with variable success (Beurer-Kellner et al., 2024; Wang, 2024; Sengottuvelu, 2023). State-of-the-art LLM services such as OpenAI's GPT-4, Anthropic's Claude 3, and Mistral AI's models, to name a few, have also recently introduced a "JSON-mode" that allows the user to steer the model's output towards generating JSON, but without strict guarantee and still requiring the model to be explicitly instructed to output the JSON (OpenAI, 2024). Various wrapper libraries like JSONFormer (Sengottuvelu, 2023) allow to decompose the JSON generation problem into multiple independent value generation queries for each key, then using the generated values to fill the schema of the object in a post-processing re-composition step. The drawbacks of the approaches mentioned above

Correspondence: {atavanae,kiatkoo,hayro,shaobaij,qlimz, hameng,bouykari}@amazon.com are 1) long prompts/instructions and extensive prompt-engineering process, 2) unstable LLM's behaviour in response to prompt changes, 3) prerequisites for prompt preparation such as structured objects schema or "keys" in JSON, and 4) computationally expensive LLMs.

To address these issues, we propose a self-supervised learning model to learn a native JSON Language Model, or **Structured object Language Model (SoLM)**, that natively generates objects that conform to a given structure (schema, class, database model, relational model, API specification, etc). The proposed SoLM acts as an object generator, but also as an object self-regeneration machine. No instructions or prompt-engineering is required for the model, which intelligently and autonomously understands what is the best possible schema and output given the input payload. Our model can also inherently perform multiple enhancement tasks while (re)generating the object. Tasks include 1) creation of the structured object from unstructured noisy input, 2) auto-completion of incomplete structured input, 3) error detection and auto-correction of noisy structured input, 4) auto-normalization of noisy structured input to desired normalization schemes, 5) auto-dependency resolution and auto-enforcement of inter-dependent parts/facets of the object.

In this work, we focus specifically on complex multi-facet objects with intricate dependencies between the different components (facets) of the object. The facets of the object (also known as attributes, fields, columns, properties) can be a mix of short, structured facts, or long, complex, natural-language descriptions. This type of structure naturally occurs in complex production use-case. Examples include product listings in online stores, house listings, job listings, entity records, etc. The object has to be self-consistent between the different facets and redundant information it carries (relative consistency), while being grounded and consistent with respect to a world knowledge about the entity (absolute consistency). We use an online store product catalog as an example application. For these types of e-commerce listings, some parts of the structure (e.g. title, product description, feature bullets, etc) are free-form natural language type of content, while other parts (structured metadata) are short form data-type and enumeration-constrained type of content. The proposed Structured Object Language Model handles the interleaving of these different types of content and ensures

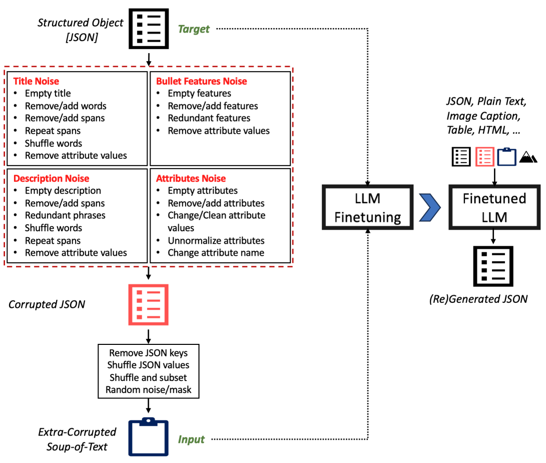


Figure 1: The noising functions applied to structured objects of products in e-commerce.

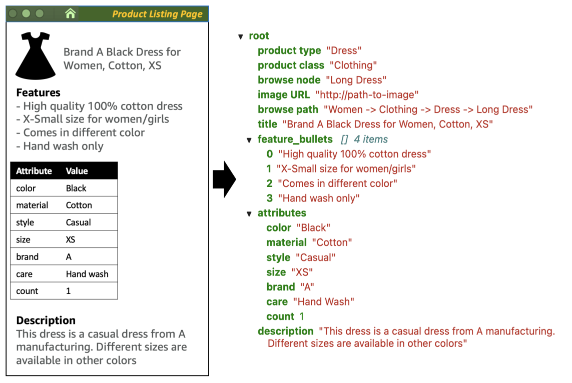
self-consistency between the natural-language portions and the structured content portions.

In this paper, starting from a general purpose 7B parameter pre-trained Language Model, we first train our Structured Object Language Model using novel targeted denoising functions in a self-supervised manner (SoLM Self Supervised). The current model is then further fine tuned based on few human generated high quality human demonstrations to align the LLM to human preferences (SoLM SFT: Supervised Fine Tuning). We compare this approach against prompt-engineering of SOTA LLMs, namely Claude 3.0 Sonnet and Mixtral-8x7B-Instruct, using two different prompt-engineering paradigms (whole object generation versus individual attributes generation). Results show that the proposed SoLM model is able to match the performance of prompt-engineered Claude 3.0 Sonnet while being order of magnitude more cost-effective.

2 Self-Supervised Training

In the following, we use the pre-trained MPT-7B as the backbone transformer architecture. MPT-7B is a decoder-only transformer pre-trained on English text and code including 1 trillion tokens (MosaicML, 2023). However, the proposed approach can be applied to any generative model (encoder-decoder or decoder-only). MPT-7B supports ALiBi position encoding for long text processing and generation regardless of the training text length and Flash Attention (Dao et al., 2022) for less GPU memory usage and a faster attention algorithm.

The self-supervised learning approach does not require human labeled data and uses denoising techniques on a corpus of existing noisy data and



Brand A Black Dress for Women, Cotton, XS

Features
- High quality 100% cotton dress
- X-Small size for women/girls
- Comes in different color
- Hand wash only

Attribute	Value
color	Black
material	Cotton
style	Casual
size	XS
brand	A
care	Hand wash
count	1

Description
This dress is a casual dress from A manufacturing. Different sizes are available in other colors

```
▼ root
   product type "Dress"
   product class "Clothing"
   browse node "Long Dress"
   image URL "http://path-to-image"
   browse path "Women -> Clothing -> Dress -> Long Dress"
   title "Brand A Black Dress for Women, Cotton, XS"
 ▼ feature_bullets  [] 4 items
     0 "High quality 100% cotton dress"
     1 "X-Small size for women/girls"
     2 "Comes in different color"
     3 "Hand wash only"
 ▼ attributes
     color "Black"
     material "Cotton"
     style "Casual"
     size "XS"
     brand "A"
     care "Hand Wash"
     count 1
   description "This dress is a casual dress from A manufacturing.
     Different sizes are available in other colors"
```

Figure 2: A structured object representing a product listing that includes multiple correlated components.

tasks (Tay et al., 2022; Raffel et al., 2020). The proposed noising functions are designed to minimize hallucination while preserving most of the pre-trained model's world knowledge.

We define two modes of operations: one where the output has to strictly be grounded in data explicitly mentioned in the input payload/context, and one where the model is encouraged to rely on its inductive biases to guess the most plausible values even if not explicitly mentioned in the input payload. Those two modes of operation allow to cover a range of applications of structured objects generation or regeneration. Sec 6 shows use-cases for the two modes.

2.1 Noising Functions

The two main components of the self-supervised learning stage are 1) a dataset of existing objects of interest (possibly noisy) and 2) a set of noising functions to apply on these noisy objects and generate very noisy inputs. The self-supervised model learns to remove the noise from the very noisy objects to recover their less noisy version. The assumption we make is that most of the objects in the dataset naturally carry some minimum amount of quality. We use these objects as target samples, and use the set of engineered noising functions to corrupt these target objects to form artificial input samples. The concatenation of the corrupted input and original target form one learning sample.

Each component (facet) of an object is corrupted based on a subset of noising functions only targeting that component while using other components for noise customization. For instance, a structured object in an e-commerce catalog dataset would include four components: 1) title, 2) free-form bullet points describing the product features, 3) long description, and 4) tabular attributes such as color, material, brand, and size (up to a few hundred attributes per product, with schema depending on the product category). Each component becomes noisy by semi-randomly removing, changing, or adding information to the component's details. The "semi-randomly" here refers to random, controlled noises such that the noisy part of the component should be able to be recovered to the correct/complete format based on information understood from other components. For example, if the noising function removes or changes the color of a product in the tabular attributes, the correct color value should be mentioned (explicitly or implicitly based on the product category) in the other three components. This controlled noising function makes sure that the model does not hallucinate at inference and only generates or changes the texts if there are valid references in the whole structured object. This control can be tuned or turned off depending on the mode of operation (creative generation versus strictly grounded generation).

2.2 Training Data Preparation

For each structured object, a random combination of the noising functions explained above is calculated on-the-fly and applied during training. The noising function for each component is randomly selected from a noising functions pool for that component. The noise intensity for each function (for instance, average number of words removed from the title) is itself randomly chosen from 0 to 100%. At the end, a combination of noising functions on all the components are applied to the structured object to prepare a noisy structured object.

While adequate for the use-case of regeneration of existing structured object for the purpose of cleaning (completing, correcting, normalizing, etc), the combination of the targeted noising functions mentioned above is not sufficient for the use-case of structured object generation from scratch given free-style input contexts or completely unstructured blurb inputs. To make the model more general and able to convert any informative text to the structured object of interest, with ρ probability (e.g. $\rho = 30\%$), we apply additional extreme noising to convert the corrupted JSON to so called "soup-of-words" (including complete structure destruction of the input and random shuffling of the tokens). Fig. 1 shows the noising functions and the final noise combination specified for e-commerce training data preparation. As shown in this figure, the model can get any input (plain text, structured object/JSON, image caption, tabular data, etc.) and generate a structured object/JSON with consistent, correct, and complete components in single pass.

2.3 Denoising Training

The fine-tuning in the decoder-only model is conducted by feeding the prepared input followed by the target structured object to the LLM for CLM (causal language modeling) training. The training sample template is

```
<BOS><input text><\n><target JSON><EOS>
```

At inference, the model requires the noisy (original) structured object or text followed by "\n" to generate the corresponding structured object.

3 Supervised Fine Tuning

Through the use of denoising functions, the self-supervised denoising trains the LLM to generate structured objects that conform to the target schema. Although it is effective in adapting a pretrained LLM to the desired domain, as shown in Section 6, the resulting model performance is limited as it is not explicitly trained to generate human preferred objects, especially on the subjective parts of the object (e.g. long description and free-form bullets).

Supervised Fine Tuning (SFT) is commonly used in the literature to align an LLM to desirable user responses (Ouyang et al., 2022). Given a structured object, there exist a notion of human desirable or preferred responses. The desired output is one that contains all relevant and factual information representing the data. The key to SFT lies in a demonstration dataset generated by human experts.

4 Data Corpus

We validate our approach in the domain of e-commerce structured product catalog data.

4.1 Self-Supervised Denoising Dataset

We used a sample dataset from an established e-commerce online store containing 30 million product listings across thousands of product categories, filtered with simple heuristics to ensure a minimum data quality bar. The components representing a product are correlated and have different data types like image, free-form text, structured attributes, and class names. These product listings are used as the target text generated by our model and the input is created by the proposed targeted noising functions explained in the previous section. The self-supervision stage maximizes for training data quantity over quality to ensure that the model is able to adapt to different products in the universe. Fig 2 shows an example of the structured object (JSON) representing a product in our dataset.

4.2 SFT Dataset

A naive approach to SFT would be to collect a small amount set of supervised training data for all product categories. However given the number of categories present in the data-set, this naive approach is both expensive and impractical to implement at scale. Instead, we propose a training pipeline similar to a funnel, where as it progress down the funnel the quality of the training data improves but with lower quantity:

- **SFT Stage 1 - Existing High Quality Structured Objects**: We used an ad-hoc model trained to predict product quality to select existing structured objects from the noisy corpus which are of high quality for SFT training. This product quality model is trained based on existing business definition of product listing quality. The model is able to identify a sample subset of around 200K existing products which are deemed to be high quality according to this definition, filtered down from the original 30M self-supervision dataset.

- **SFT Stage 2 - Human Labels**: Due to the aggressive nature of SFT Stage 1, many product categories remain under represented in the SFT training data set. Samples from these under represented products are then sent to human experts for labelling. Each human regenerated product is cross checked by another expert. This dataset contains around 3K product listings (structured objects).

5 Model Training and Evaluation Metrics

5.1 Training

The 7B SoLM model is trained on 5xAWS P4 instances, each with 8x 40GB A100 GPUs. We run ablations on a few backbone architectures including FLAN-T5 (XL and XXL) (Chung et al., 2022), MPT-7B (MosaicML, 2023) and Mistral-7B (Jiang et al., 2023) to find the best pre-trained base model for the rest of the developments. See Appendix B.

5.2 Evaluation Metrics

The evaluation metrics we use in this paper fit the multi-facet structured objects (such as JSON data of e-commerce product listings). The evaluation metrics are divided into two categories: 1) metrics for free-form texts (title, feature bullets, and description) and 2) metrics for tabular attributes.

Generated Free-form Facets - The free-form texts are evaluated using subjective and objective methods. The objective evaluation is formulated by Rouge scores (Lin, 2004) on the reference texts for synthetically noised inputs. A more reliable evaluation is the subjective evaluation on the generated texts given original, real inputs. A text is labeled as "correct" if it 1) uses fluent language, 2) includes necessary attributes (e.g. color for shirt), 3) has no hallucination and false claims, and 4) represents the product based on the input information.

Generated Tabular Attributes - The aim of the proposed model is to generate correct, complete, and normalized objects, we define the two metrics:

- **Correctness Rate / Precision**: Measures the number of correctly generated attributes divided by the number of generated attributes
- **Completeness Rate / Recall**: Measures the number of generated attributes divided by the number of required attributes

6 Experiments and Results

6.1 Offline Evaluations

We performed initial experiments to assess the performance of the self-supervised model in comparison with a SOTA instruction-tuned LLM (Mixtral-8x7B-Instruct) with JSON-mode in zero-shot. The self-supervised model outperforms zero-shot Mixtral significantly. For example on title generation, the self-supervised SoLM outperformed Mixtral by 40.38 percentage points on Rouge-L F1 Score. Details are available in Appendix **A**.

As a proof of concept, Table 1 shows the results of the self-supervision model on the synthetic task of improving and regenerating the whole structured object in one pass after applying a combination of synthetic noises to all the object's components.

Table 1: Performance of our self-supervised Structured-Objects Language Model (Self Supervised SoLM) in denoising synthetically noised product listings.

Object's Facet *(eval. metric)*	**Noised Inputs**	**Regen. Outputs**
Title *(Rouge-L F1)*	52.09	**69.58**
Feature Bullets *(Rouge-L F1)*	64.67	**73.36**
Description *(Rouge-L F1)*	54.45	**67.66**
Tabular Attributes *(Accuracy)*	82.07	**90.32**

6.2 Real Test Cases

The real case benchmark consists of a sample of around 5K product listings randomly sampled from an e-commerce catalog, with the task of improving their quality and fixing any issue with the listing. The original (input) structured objects and the regenerated ones were all human labeled to measure the baseline versus the regenerated quality.

Table 2 shows the product listings quality, versus the quality of regenerated ones by the proposed model (natively) and by SOTA LLM extensively prompt-engineered for the task. Claude 3.0 Sonnet was prompt-engineered for generating the object in one LLM call (single prompt). We also compare against an alternative prompting strategy consisting in running multiple independent LLM calls for each component/attribute of the object, generating the structured object one piece at a time by executing the prompt for each attribute separately. This strategy requires >100 LLM runs per object followed by post processing for recomposing the object. Due to its high throughput requirement (100X throughput), we could not use Claude 3.0 for this approach, therefore we used a SOTA self-hosted open-source LLM, namely Mixtral-8x7B-Instruct. Note that both prompt-engineering approaches require product categories and corresponding product schema to be given as input. This requires running an upstream product category classification model and connecting the prompt with a product-category to product-schema mapping table. The precision and recall in Table 2 represent the correctness and completeness scores of the generated attributes (Sec. 5.2). The title quality is a composite score of human scores of overall quality, and automatic quality check of title length and restricted characters/phrases in the title. The feature bullets quality is assessed by heuristics rules.

As our SoLM model is trained in 2 stages (self-supervised training followed by SFT), we report results for both stages. As reported in Table 2, the self-supervised model shows high precision (correctness) for structured attribute generation. General self supervised denoising increases the hallucination rate as the model is trying to fill out the missing parts as much as possible which drops precision (correctness). However, in our proposed targeted denoising, we define specific control variables in the noising functions (as explained earlier) to minimize the hallucination, resulting in high precision/correctness. The SFT model shows sig-

Table 2: Product listing regeneration applied on 5K real case test. Best in bold, second-best underlined. TQ: title quality. FBQ: feature bullets quality.

Model	Relative Cost	Precision	Recall	TQ	FBQ
Input dataset (baseline)	–	85.31	46.69	42.24	55.39
SoLM Self-Supervised	**1X (1 run)**	83.30	60.20	57.93	98.62
SoLM SFT	**1X (1 run)**	82.30	65.70	**72.99**	98.62
Claude 3.0 Sonnet (single prompt)	6.8X (1 run)	**83.90**	**67.40**	66.47	**99.67**
Mixtral-8x7B-Instruct (1 run per attribute)	2X (M runs)	81.80	58.4	58.37	76.62

Table 3: Comparison between the different approaches

Model	Size	Cost	Prompt tokens	Need schema	Need prod. category	Nb. of runs
SoLM (7B)	**7B**	**1X**	**None**	**No**	**No**	**1**
Claude 3.0 Sonnet	>100B	7X	+2k	Yes	Yes	**1**
Mixtral-8x7B-Instruct	8x7B	2X	+2k	Yes	Yes	>100

nificant improvement in the free-form facet of title, which carries a strong element of human preference.

We ran extensive iterations on Claude prompts, experimenting with multiple prompting approaches and following the model provider's best practices. We worked with expert Claude prompt-engineers to craft the prompts. The final prompt is a long prompt (>2000 tokens) explaining all the requirements, listing the product category, the corresponding schema for each category, and additional control instructions. The average performance of the SoLM SFT model is comparable to the best resulting Claude performance while requiring approximately 7 times less computations without any pre-processing requirement (product category classification, attribute list by product category, etc.).

In another experiment, around 2K product listings with various arbitrary schemas–different from our dataset schema–were selected to be converted to our target schema. As shown in Table 4, Claude performed best, while our model performed closely without being explicitly trained on this task.

Table 4: Arbitrary schema to target schema conversion (new product listing generation). Free form texts quality is reported by **F**eatures **B**ullet **Q**uality, title **Rel**evance, title **Corr**ectness, and title **Cons**istency.

Model	Precision	Recall	FBQ	Rel.	Corr.	Cons.
SoLM (7B)	75.8	44.2	98.2	96.8	75.0	97.0
Claude 3.0 Sonnet	**76.1**	**57.0**	**98.8**	**97.3**	**76.8**	**98.7**
Mixtral-8x7B-Instruct	62.6	57.6	88.0	92.7	64.5	96.5

The last real case experiment involves plain unstructured blurb text as the only source of input information. The user provides a short text and/or image(s) (that can be converted to text by captioning) and the model generates the product listing in JSON format. We used a real dataset 70 samples evaluated by human auditors to assess the generated texts relevancy, correctness, and consistency like above. See Table 5 for results, that are consistent with the other experiments results.

Table 5: Unstructured blurb to structured object. Our model is not trained explicitly for this task

Model	Relevance	Correctness	Consistency
SoLM (7B)	**99.5**	66.0	96.4
Claude 3.0 Sonnet	**99.5**	**71.3**	**100**
Mixtral-8x7B-Instruct	98.4	56.5	97.8

6.3 Online A/B Tests

Structured product data are mostly self-reported by individual retailers when listing on e-commerce product websites. Studies have shown that these self-reported data can be sparse and contain noisy facts (Cheng et al., 2023). In this paper we use the proposed LLM to improve the product titles that is provided by retailers. Specifically given all relevant information provided by retailers when listing a product, our goal is to enhance the initial retailer provided product title in a manner in which will improve our buyers experience in discovering products relevant to their intent. Improving our buyer's shopping experience consequently will also meaningfully improve our retailers products exposure.

We use the enhanced title - as output by our LLM - to run an online A/B test against the existing retailer provided version in an English Language Store over a period of 2 weeks. Evaluation results show that our customers (buyers) prefer the title generated by our LLM compared to the existing retailer provided version overall. Particularly the revised titles improved revenue (p-value=0.059) and increased the total units purchased (p-value=0.034).

7 Conclusion

This paper proposes a new approach to generate structured objects in a single pass without needing any prompt nor objects' schema. The Structured Object Language Model (SoLM) is trained using a novel self-supervised training method incorporating a combination of targeted noising functions to help create or improve structured objects with complete, correct, and normalized components. The self-supervised model is further fine-tuned on human labeled data to improve the quality of the free-form text components (SoLM SFT).

In future work, we will extend the training stages by incorporating reinforcement learning from human feedback (RLHF) and Preference Optimization (DPO) to better capture human preference.

References

ANTHROP. 2024. Introducing claude models, https://www.anthropic.com.

Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. 2024. Guiding llms the right way: Fast, non-invasive constrained generation. *arXiv preprint arXiv:2403.06988*.

Kewei Cheng, Xian Li, Zhengyang Wang, Chenwei Zhang, Binxuan Huang, Yifan Ethan Xu, Xin Luna Dong, and Yizhou Sun. 2023. Tab-cleaner: Weakly supervised tabular data cleaning via pre-training for E-commerce catalog. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 5: Industry Track)*, pages 172–185, Toronto, Canada. Association for Computational Linguistics.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, Albert Webson, Shixiang Shane Gu, Zhuyun Dai, Mirac Suzgun, Xinyun Chen, Aakanksha Chowdhery, Alex Castro-Ros, Marie Pellat, Kevin Robinson, Dasha Valter, Sharan Narang, Gaurav Mishra, Adams Yu, Vincent Zhao, Yanping Huang, Andrew Dai, Hongkun Yu, Slav Petrov, Ed H. Chi, Jeff Dean, Jacob Devlin, Adam Roberts, Denny Zhou, Quoc V. Le, and Jason Wei. 2022. Scaling instruction-finetuned language models. *Preprint*, arXiv:2210.11416.

Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. 2022. Flashattention: Fast and memory-efficient exact attention with io-awareness. *Preprint*, arXiv:2205.14135.

Luciano Floridi and Massimo Chiriatti. 2020. Gpt-3: Its nature, scope, limits, and consequences. *Minds and Machines*, 30:681–694.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. 2023. Mistral 7b. *arXiv preprint arXiv:2310.06825*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Ouail Kitouni, Niklas Nolte, James Hensman, and Bhaskar Mitra. 2024. Disk: A diffusion model for structured knowledge. *Preprint*, arXiv:2312.05253.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pages 74–81.

MosaicML. 2023. Introducing mpt-7b: A new standard for open-source, commercially usable llms. Accessed: 2023-05-05.

OpenAI. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. *Preprint*, arXiv:2203.02155.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67.

Machel Reid, Nikolay Savinov, Denis Teplyashin, Dmitry Lepikhin, Timothy Lillicrap, Jean-baptiste Alayrac, Radu Soricut, Angeliki Lazaridou, Orhan Firat, Julian Schrittwieser, et al. 2024. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*.

Rahul Sengottuvelu. 2023. Jsonformer: A bulletproof way to generate structured json from language models.

Yi Tay, Mostafa Dehghani, Vinh Q Tran, Xavier Garcia, Jason Wei, Xuezhi Wang, Hyung Won Chung, Siamak Shakeri, Dara Bahri, Tal Schuster, et al. 2022. Ul2: Unifying language learning paradigms. *arXiv preprint arXiv:2205.05131*.

Jiaye Wang. 2024. Constraining large language model for generating computer-parsable content. *arXiv preprint arXiv:2404.05499*.

Yanli Zhao, Andrew Gu, Rohan Varma, Liang Luo, Chien-Chin Huang, Min Xu, Less Wright, Hamid Shojanazeri, Myle Ott, Sam Shleifer, et al. 2023. Pytorch fsdp: experiences on scaling fully sharded data parallel. *arXiv preprint arXiv:2304.11277*.

A Initial Results: Proof of Concept

For the initial experiments, we assess the performance of the self-supervised Structured Object Language Model (SoLM - 7B parameters) in comparison with the zero-shot Mixtral-8x7B-Instruct (8×7B parameters). The initial test dataset includes around 1K high quality product listings (structured objects) each including title, feature bullets, description, and a number of tabular attributes. To

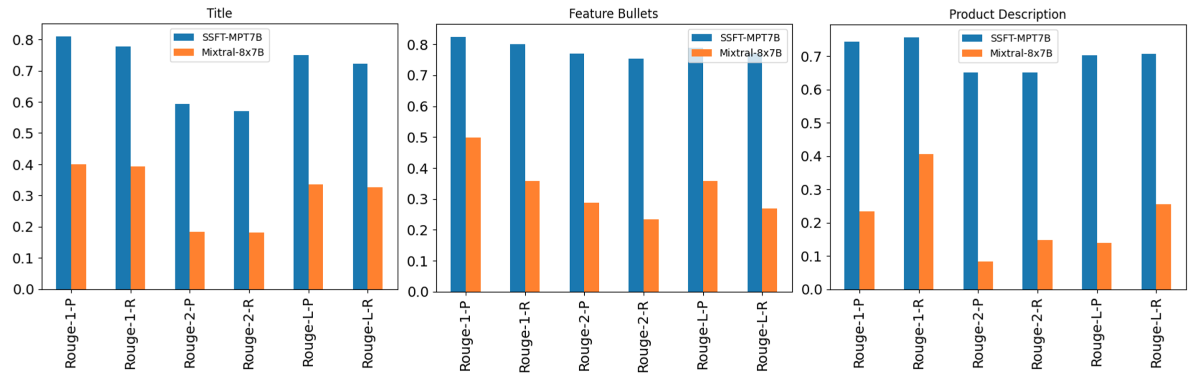


Figure 3: Zero-shot Mixtral with short prompt versus the self-supervised SoLM-7B for free-form text generation representing Title, Feature Bullets, and Product Description components of product listings in e-commerce.

prepare the input, we applied diverse noises (that are also different from the noising functions used in training) on each component and expected the LLMs to improve and correct the noisy component. For free-form texts we used Rouge scores to compare the generated texts with the reference texts in the original structured objects. Since the self-supervised model is specifically trained on the structured objects, it's expected to perform better. Also, the zero-shot Mixtral does not have any information about the structured objects schema. Thus, only the Self Supervised SoLM is able to generate the list of tabular attributes that are missing in the structured object.

Fig. 3 shows the Rouge scores of the zero-shot Mixtral versus SSFT for unstructured text generation. Table 6 compares the Self-Supervised SoLM and zero-shot Mixtral models in terms of 1) Rouge-L-F1 score, 2) subjective evaluation scores, and 3) tabular attributes accuracy. The subjective evaluation is conducted by assigning quality scores (0-100) to 150 generated texts by 2 human evaluators in which the highest scores is given to a text that is human readable and provides correct information about the structured object as explained in Section 5.2. The tabular attribute accuracy reports the correctness of the generated attribute values in comparison with the target attribute values in the original structured objects using fuzzy string matching.

B Training Details

The FLAN-T5s are encoder-decoder models while others are decoder-only models. The current T5 architecture does not support existing Flash Attention algorithms (Dao et al., 2022). Thus, the training and inference time is expected to be high and the maximum input length for training must be less than 1024 tokens (due to memory and time complexity of the original self-attention on our EC2 instances). On the other hand, MPT and Mistral support Flash Attention which requires less memory and time and support very long input/output texts (>8k tokens).

The models greater than 2B parameters cannot fit on a single GPU with $\leq$ 40 GB memory for training of long input/output texts (>1000 tokens). Thus, all the training codes are implemented by sharding the model using Pytorch-FSDP (Zhao et al., 2023) to divide the computations cross multiple GPUs-Instances. The input/output maximum length were set to 1000 tokens for FLAN-T5-XL, 600 tokens for FLAN-T5-XXL and 8000 tokens for MPT and Mistral. Our catalog dataset including the components mentioned above has between 400 to 2500 tokens where make the T5-based models unable to learn long product listings. However, MPT and Mistral models can cover all the product listings (2500 input + 2500 output $\ll$ 8000) in our training dataset.

Table 6: Initial comparison between the Self-Supervised Stuructured Object Language Model (SoLM) and zero-shot, short-prompted Mixtral-7x8B.

Component Generation Eval	**SoLM-7B**	**Mixtral-8x7B**
Title (Rouge-L F1)	**73.53**	33.15
Feature Bullets (Rouge-L F1)	**77.90**	30.68
Description (Rouge-L F1)	**70.50**	18.10
Title (Subjective Score)	**90.95**	70.68
Feature Bullets (Subjective Score)	**91.76**	87.03
Description (Subjective Score)	**87.03**	85.27
Tabular Attributes Accuracy	**90.41**	NA