Story Centaur: Large Language Model Few Shot Learning as a Creative Writing Tool

Ben Swanson, Kory W. Mathewson[2], Ben Pietrzak,
Sherol Chen, Monica Dinalescu
Google, [2]DeepMind
pwnr, korymath, bpietrzak, sherol, noms@google.com

Abstract

Few shot learning with large language models has the potential to give individuals without formal machine learning training the access to a wide range of text to text models. We consider how this applies to creative writers and present STORY CENTAUR, a user interface for prototyping few shot models and a set of recombinable web components that deploy them. STORY CENTAUR's goal is to expose creative writers to few shot learning with a simple but powerful interface that lets them compose their own co-creation tools that further their own unique artistic directions. We build out several examples of such tools, and in the process probe the boundaries and issues surrounding generation with large language models.

1 Introduction

One of the most promising possibilities for large language models (LLMs) is *few-shot learning* (Brown et al., 2020) in which it is possible to create text to text models with little or no training data. Few shot learning with LLMs relies on the ease at which the desired Input/Output (I/O) behavior can be effectively translated into the text continuation problem at which these models excel. In recent years, LLMs have progressed to the level where this translation requires only familiarity with the natural (e.g. English) language on which the model was trained.

We present STORY CENTAUR, a Human-Computer Interface that closes the gap between non-technical users and the power and possibilities of few shot learning, with the intended audience of writers of creative text. It is our intention that by giving writers a tool for building generative text models unique to their process and vision that these artists will experience genuine feelings of co-creation.

STORY CENTAUR consists of a prototyping UI as well as a set of Angular web components that interact via a central pub-sub synchronization mechanism[1]. Due to the non-trivial monetary cost of inference and the requirement of specialized hardware, the LLM that underlies our tool is not included in this release; instead we dependency inject the LLM with a simple (string) $\rightarrow$ string interface to be provided by an arbitrary service.

As the ethical implications of LLMs (Bender et al., 2021) are an important and unsolved problem in NLP, we highlight this design choice to decouple the LLM itself from STORY CENTAUR, a web based user interface that prepares the LLM's inputs and processes its outputs. To put it another way, the text generated is no more or less biased than the LLM and user themselves, as STORY CENTAUR's purpose is not to enhance or change the abilities of a LLM, but instead to democratize its use to non-technical users.

2 Related Work

The observation that simple "fill-in-the-blank" neural network models trained on large quantities of text can be used for problems beyond their primary learning objective dates back to word2vec (Mikolov et al., 2013) in which word embeddings were able to perform some SAT style analogies. While this ability captured many researchers' fascination, these models' dual function as a representation learner from which other models could be initialized and/or fine-tuned took center stage in the following years.

Representation learning techniques made steady advances, expanding to sentence level contextually sensitive word embedding with ELMo (Peters et al., 2018), the introduction of Transformers and MLM objectives with BERT (Devlin et al., 2018),

[1]source code: https://chonger.github.io/centaur/

Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations, pages 244–256
April 19 - 23, 2021.

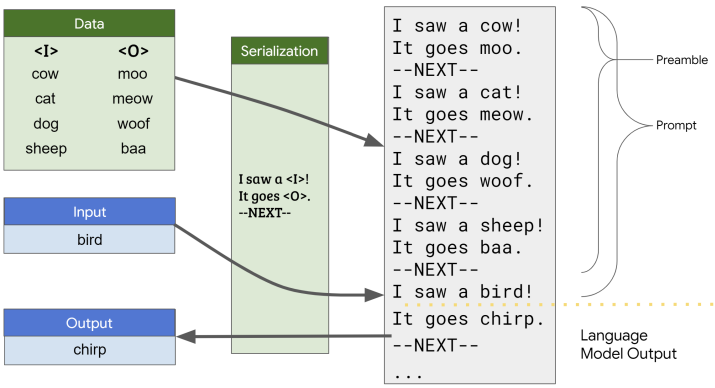


Figure 1: A Few Shot Formula in action. The Data and Serialization are used to create the Prompt, which along with the serialized inference time Input becomes the Preamble. The LM generates a continuation, from which the Serialization extracts the output. The Sentinels used (with newlines omitted) are "I saw a", "! It goes ", and ". –NEXT–".

and the menagerie of similar systems that followed. While most work concerned itself primarily with topping each other's GLUE and SUPERGLUE scores (Wang et al., 2019), work from OpenAI kept the torch lit for investigating the emergent abilities of representation learning (Radford et al., 2017, 2018, 2019; Brown et al., 2020). Their most recent work undeniably shows that sufficiently large LMs enable few shot models that approach and sometimes surpass state of the art performance on a wide range of NLP tasks.

Human + AI co-creation has existed in both practice and theory for several years. To highlight some examples in practice that relate to creative writing, as opposed to music or visual art of which there are many, we refer the reader to browse the Electronic Literature Collection[2] a longstanding community of artist-technologists who have blazed this trail since the days of hypertext. A number of publications of AI co-creation exist as well on a diverse range of artistic applications (Martin et al., 2016; Mathewson and Mirowski, 2017; Oh et al., 2018; Mirowski and Mathewson, 2019; Kreminski et al., 2019; Sloan, 2019; Tsioustas et al., 2020).

For a lighter introduction, Case (2018) gives some examples of AI + Human collaborations, or Centaurs[3], but primarily presents the argument that the HCI that connects the Human and Computer is of paramount importance, a sentiment that is in line with our own work. We also resonate with the opinion of Llano et al. (2020), which argues for explainability as a catalyst for fruitful co-creation, as it is central to our design that the artist be given the tools to create and iterate on NLP models.

3 Few Shot Formulas

The core contribution of this work is a UI for the creation of few shot text generation models (Figure 2). We first define terms for the components of LM based few shot modeling as it is decomposed in our system: The few shot learning system as a whole is represented as a **Formula** which when used with a large LM provides arbitrary text to text I/O behavior.

We note that while generally LMs refer to any probability distribution over a sequence of tokens, in this work we use the term to refer to the subset model class that factorizes the joint probability into conditional $P(w_t|\mathbf{w_{1...t-1}})$ terms. Put simply, we are referring to the "predict the next word given all words so far" variety of LM, which includes all of the GPT models.

A Formula is composed of **Data** and **Serialization**. Each item in the Data consists of lists of string inputs and outputs that exemplify the desired I/O. The Serialization defines a reversible transformation between the Data and the raw text handled by the LM.

STORY CENTAUR uses a Serialization template of fixed text **Sentinels** that interleave the inputs and outputs; a Sentinel is defined to precede each input and output, as well as one that separates inputs and outputs and one that comes after the final output (See Figure 2 for an example). Carefully chosen sentinels are powerful tools for nudging the language model in desired directions (see the

[2]https://collection.eliterature.org/

[3]Case (2018) also explains this term's etymology

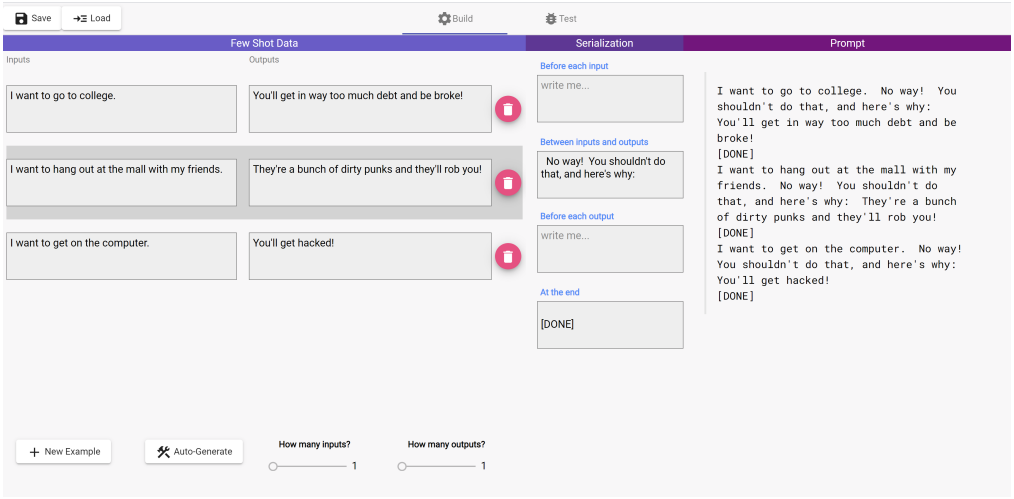


Figure 2: STORY CENTAUR's Formula Development User Interface.

Appendices for examples), but must also be designed so as not to be confused with model input or outputs.

A Formula is used by first invoking the Serialization on the Data, creating the **Preamble**. Then, the new inputs are converted using the Serialization and concatenated to the Preamble, creating the **Prompt**. The LM is asked to continue the text in the Prompt, and the Serialization is used to extract the output(s) from the result. The LM cannot explicitly enforce the Serialization format and as such will often produce non-conformant results, in which case it must be rejected. In practice, if the LM is sufficiently capable and the task well suited then a simple rejection sampler suffices to produce several acceptable options, as decoding is parallelizable.

3.1 Formula Development

STORY CENTAUR's user interface for Formula design is shown in Figure 2, with supplemental screenshots in the appendices. While there is no fixed workflow, we have found the following process to be effective. We assume only that the user is proficient in English and has a strong concept of their desired I/O.

First, the user must enter at least two examples of I/O pairs into the Data panel and take a pass at defining a Serialization, relying on the live updated Preamble panel to preview their progress. With a few examples in place, the Auto-Generate button can then be used to suggest new candidate IO pairs by passing the Preamble to the LM and allowing the user to prune these suggestions. This process can be repeated, quickly converging to several (10 or more) solid examples and clear evidence that the Serialization is being captured by the LM. As a final evaluation technique, we provide a Test mode that takes inference inputs and applies the current Formula, also reporting the rate at which the LM output respects the Serialization.

4 Writing Tool Experiments

We showcase the potential of Formulas that one might create using STORY CENTAUR in several Experiments. These experiments all rely on one or more Formulas that were built using the development tool and workflow described above, and are each motivated by a different artistic scenario. When possible, we present the I/O specifications for each Experiment and invite the reader to view full Experiment screenshots as well as the underlying Formulas' Data and Serialization in the Appendices.

4.1 Magic Word

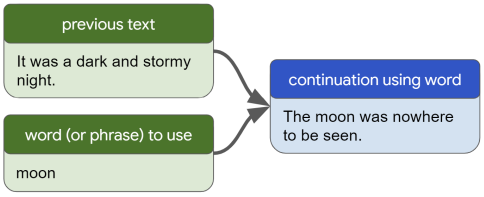


Figure 3: Formula I/O for Magic Word.

Perhaps the most obvious application of generative language models to creative writing is overcoming writer's block. Specifically, we consider the scenario in which the writer has some existing seed

text and wants to be presented with possible continuations.

Generative LMs are ripe for this task as they can reliably continue short text; for the definition of LM used in this work (see Section 3) this is indeed exactly the task they were trained on. In this pure use of the LM the author is only able to provide the seed text, and so in this experiment we use a few shot Formula to provide an additional input of a word or phrase that is required to appear in the generated text.

The Magic Word formula takes two inputs: the seed text that must be continued by the model and the "Magic Word" that must be used in the continuation. In this Experiment, the generated outputs are not only discarded if they do not conform to the Serialization but also if the Magic Word does not appear as a substring. The UI allows editing of both the magic word and seed text, and on generation the user is given a maximum of three sentences that they can click to append to the editable main text box.

From an academic perspective, it is worth noting that this I/O paradigm has been explored in several examples of previous work, often with the same motivation as a writer's aid (Ippolito et al., 2019).

4.2 Say It Again

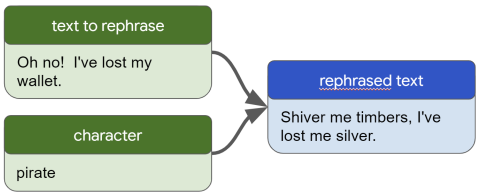


Figure 4: Formula I/O for Say It Again's CUSTOM Formula.

Many literary characters have their own peculiar way of speaking; Yoda, Tolkein's dwarves, Treasure Island's Pirates. In this Experiment we address the scenario where a writer has a clear idea of *what* they want the character to say and want suggestions as to *how* their character might actually say it.

We phrase this problem as a Formula with one input and one output in which the input is in neutral style and the output is a paraphrase with the desired style applied. This works nicely with few shot learning, as it is relatively easy to invent (or generate) a simple unstyled statement and then to imagine how a character might say it. We showcase several such Formulas in this experiment, selectable in a menu. For unstyled source text, there are three editable areas for text to rephrase that can be restyled individually or all at once.

We provide one additional Formula that might be considered zero shot style transfer, although it is still performed using a few shot Formula. When the style "CUSTOM" is selected, an input box appears where the user can enter any raw text they wish. This text is then used in a Formula with two inputs, the text to be restyled and the name or description of the character whose style to use. The surprising result is that this is often possible with no examples of the requested style itself, only the proper Serialization and a few example of the full I/O shown in Figure 4 with other custom characters. As this information most likely comes from patterns and associations encoded into the LMs parameters during training, this method works best with fictional characters from major movies or celebrities.

We encourage the further examination of large LMs for style transfer, as we were anecdotally impressed with the output of this experiment in particular. As some recently successful work in style transfer (Riley et al., 2020) already follows a label free approach that might itself be considered few shot in nature, interesting experimental comparisons are likely possible.

4.3 Story Spine

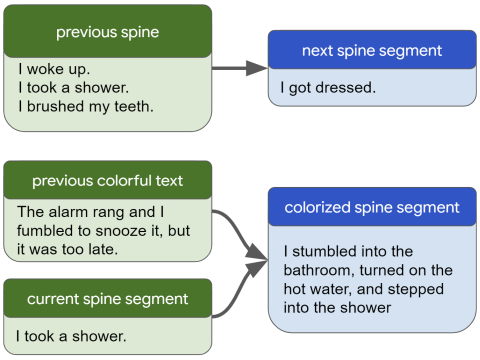


Figure 5: Story Spine's two formulas for spine continuation (top) and spine colorizing (bottom).

Modern LMs excel at producing text that is coherent, grammatical, and at times interesting, and frequently amusing. However, cracks begin to show in coherence as generations grow longer (Cho et al., 2018). A common mitigating technique has been to construct hierarchical generation systems in which a high level representation that is focused on common sense story structure which is then

transformed into narrative text (Fan et al., 2018; Ammanabrolu et al.). This trend inspires this experiment, whose goal is the co-creation of a short story that is both coherent and detailed.

One ubiquitous quality of such hierarchical systems is that the high level representation is a structural and/or semantic abstraction chosen to be amenable to plot coherence modeling. This experiment poses the question: what if the high level representation was itself natural language? To explain our setup we make the distinction between *simple* text and *colorful* text, where the former is a grammatically bare bones statement of fact and the latter is more linguistically interesting, as a sentence might actually appear.

We use two Formulas to accomplish this goal, shown in Figure 5. The first takes a simple short plot point sentence as input and returns a plausible following simple plot point as output; this is used in a loop to generate the *spine*. The second is a context conditional paraphrasing formula with two inputs; the first is a simple plot point and the second is the "story so far" which is written in colorful text. The Formula's output is a paraphrase of the simple plot point, *colorized* to both respect the factual information of the plot point and the context of the story so far.

The user is presented with an interface that lets them write and edit custom spine plot points as well as use the first Formula to generate up to five candidates for plot points to continue the story. Each spine plot point is connected to its colorized paraphrase, which appear as a whole on the right side. In order to maintain a model of the mapping between the spine and colorized text, the colorized text is not editable.

4.4 Character Maker

Super-Objective	to write a novel
Sub-Objective	to invent a main character
Scene Objective	to find inspiration
Internal Obstacles	is bored and tired
External Obstacles	can't find a pen
Tactics	trying tech tools
Physical Attributes	frizzy hair
Backstory	from Sandusky, Ohio

Figure 6: The eight character creation points used in Character Maker, with examples.

Interesting characters are at the heart of much creative writing, and various template filling exercises exist to create them. Often this comes down to filling out a template containing fields that flesh out the character, as shown in Figure 6. In this experiment, the user is presented with an editable template containing each of these fields, with the option to edit or clear any of the fields' values. Once a value has been cleared, it can be filled in by the LM conditioned on all current non-empty fields.

We take this experiment in a direction that goes beyond our own Formula development tools to define a flexible Few Shot model for data completion. Our generalized problem statement is as follows: given a set of fields of which an arbitrary subset are blank, for one such blank field generate a plausible value conditioned on the non-blank fields. We build a dynamic Formula creation system that fulfills this generalized contract, and apply it to the filling of character creation exercise forms.

Our few shot solution naturally relies on a small number of fully filled out and plausibly consistent fields (e.g. complete character descriptions). At inference time, we extract the subset of non-blank fields in the inference item from each of these few shot examples and stitch together a Formula on the spot with precisely these inputs and the single output of the desired inference output field. This dynamic creation of Formulas requires a flexible Serialization that can accommodate any field name and value in any order, which for this experiment we simple simply use "name : value".

4.5 Improv Prompts

In improvisational acting (improv) one of the primary pleasures is to see actors bring a set of constraints provided by the audience to life in a coherent story. We see the potential for the sometimes wildly creative suggestions of large language models to supply these constraints, either as a tool for practitioners to hone their craft or as a way to spice up (or speed up) a live performance itself.

Improv constraints must be both open ended and subject to specific categories; for example the popular "Party Quirks" game requires a personal quirk for each actor attending a dinner party. We build Formulas and UIs for several improv games, and note their distinction from the other Formulas in this work in that they require no user input at all.

In constructing such zero input few shot learning

models it became apparent that beyond controlling the grammatical form and semantic intent of the outputs we could also control their tone, as it would mimic the tone of the Formula's Data. Crucially, this allows easy adaptation of these tools to different audiences (children versus adults, for example) and an implicit nudge towards whimsical outputs.

5 Discussion

While the experiments presented above demonstrate how few shot learning can be used to create interesting tools for writers, the real power of STORY CENTAUR is its unlocking of rapid experimentation. Not only were we able to probe the boundary of what "works" efficiently, but also to engage individuals regardless of formal machine learning training to help us to do so. Needless to say, in the course of this work many attempted Formulas did not produce compelling results.

Perhaps our most interesting failure was to build a Formula that would produce the second half of a rhyming couplet given the first half, a task that would require understanding of both phonetics and meter as well as linguistic coherence. This was disappointing given the compelling examples of GPT-3 poetry available online[4]. One possible explanation is that while general poetry and specifically rhyming couplets are in our minds connected closely with a subset relationship rooted in human culture, the hard constraint of rhyme and meter in fact divides them into very different problems for an LM. It is certainly the case that recent successful work in rhyming metered poetry generation has needed to resort to fixed rhyme words and syllable counting(Ghazvininejad et al., 2017).

In terms of larger themes, we found that constructing Formulas in which any of the inputs or outputs were much longer than a few sentences were hard to construct. We speculate that it is more difficult for the models to latch on to the Serialization in this case, as the observed symptom was often that no generated text passed the de-serialization filter. On the positive side we observed that few shot tasks that rely on paraphrasing (such as those used Say It Again) were surprisingly easy to construct successfully.

It is a common and intuitively plausible observation that the design of the Serialization is crucial to the performance of few shot learning with large LMs. Our Formulas can only be evaluated qualitatively and so we leave to future work the human studies that would be necessary to investigate this hypothesis. Our Magic Word experiment does offer the promise of a good test bed for Serialization design; even after considerable iteration we found the rate at which outputs pass both the de-Serialization filter to be surprisingly low given the relative simplicity of the task and the model's innate ability to generate coherent text continuation.

Finally we note that our true goal is to empower artists with no technical training to imagine a Formula, construct it in our development mode, and then produce experiments as we have. In our current process, such an artist could indeed construct their Formula, but would at some point require a programmer to build it into an experiment, requiring e.g. a WYSIWG editor. While this was beyond the scope of our work, we did construct our system using Angular, a modern web development framework whose core premise is modularity, dependency injection, and reuse of components. Not only do our experiments make use of a small set of these reusable components for functionality like editable text fields and clickable suggestion lists, but also all text and Formulas are synchronized by a global pub-sub service with simple string keys.

6 Conclusion

We present STORY CENTAUR, a tool for the creation and tuning of text based few shot learning Formulas powered by large language models and several experiments using Formulas built with our tool that are focused around the topic of creative writing.

The emergence of large language models has shaped the course of NLP research in the late 2010's but the question remains as to what, if any, is a viable use case for these models in their raw, un-finetuned, form. Additionally, while some claim that scaling these models is a viable path to Artificial General Intelligence, others disagree (Bender and Koller, 2020), and learning what is easy, hard, and impossible for them is crucial this debate. The answers to these questions will undoubtedly reveal themselves in the coming years and we are particularly excited to see their impact on the fine arts. In particular, we see great potential in tools built with this technology when there is a human, in this case an artist, in the loop to complement the natural deficiencies of a simple but powerful text generator that lacks editorial control and responsibility.

[4] https://www.gwern.net/GPT-3#transformer-poetry

References

Prithviraj Ammanabrolu, Ethan Tien, Wesley Cheung, Zhaochen Luo, William Ma, Lara J Martin, and Mark O Riedl. Story realization: Expanding plot events into sentences.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big. In *Proceedings of the 2020 Conference on Fairness, Accountability, and Transparency; Association for Computing Machinery: New York, NY, USA*.

Emily M. Bender and Alexander Koller. 2020. Climbing towards NLU: On meaning, form, and understanding in the age of data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5185–5198, Online. Association for Computational Linguistics.

Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Nicky Case. 2018. How to become a centaur. *Journal of Design and Science*.

Woon Sang Cho, Pengchuan Zhang, Yizhe Zhang, Xiujun Li, Michel Galley, Chris Brockett, Mengdi Wang, and Jianfeng Gao. 2018. Towards coherent and cohesive long-form text generation. *arXiv preprint arXiv:1811.00511*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. *arXiv preprint arXiv:1805.04833*.

Marjan Ghazvininejad, Xing Shi, Jay Priyadarshi, and Kevin Knight. 2017. Hafez: an interactive poetry generation system. In *Proceedings of ACL 2017, System Demonstrations*, pages 43–48.

Daphne Ippolito, David Grangier, Chris Callison-Burch, and Douglas Eck. 2019. Unsupervised hierarchical story infilling. In *Proceedings of the First Workshop on Narrative Understanding*, pages 37–43.

Max Kreminski, Devi Acharya, Nick Junius, Elisabeth Oliver, Kate Compton, Melanie Dickinson, Cyril Focht, Stacey Mason, Stella Mazeika, and Noah Wardrip-Fruin. 2019. Cozy mystery construction kit: Prototyping toward an ai-assisted collaborative storytelling mystery game. In *Proceedings of the 14th International Conference on the Foundations of Digital Games*, pages 1–9.

Maria Teresa Llano, Mark d'Inverno, Matthew Yee-King, Jon McCormack, Alon Ilsar, Alison Pease, and Simon Colton. 2020. Explainable computational creativity. In *Proc. ICCC*.

Lara J Martin, Brent Harrison, and Mark O Riedl. 2016. Improvisational computational storytelling in open worlds. In *International Conference on Interactive Digital Storytelling*, pages 73–84. Springer.

Kory Mathewson and Piotr Mirowski. 2017. Improvised theatre alongside artificial intelligences. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, volume 13.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26:3111–3119.

Piotr Mirowski and Kory Wallace Mathewson. 2019. Human improvised theatre augmented with artificial intelligence. In *Proceedings of the 2019 on Creativity and Cognition*, pages 527–530.

Changhoon Oh, Jungwoo Song, Jinhan Choi, Seonghyeon Kim, Sungwoo Lee, and Bongwon Suh. 2018. I lead, you help but only with enough details: Understanding user experience of co-creation with artificial intelligence. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pages 1–13.

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Alec Radford, Rafal Jozefowicz, and Ilya Sutskever. 2017. Learning to generate reviews and discovering sentiment. *arXiv preprint arXiv:1704.01444*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Parker Riley, Noah Constant, Mandy Guo, Girish Kumar, David Uthus, and Zarana Parekh. 2020. Textsettr: Label-free text style extraction and tunable targeted restyling. *arXiv preprint arXiv:2010.03802*.

Robin Sloan. 2019. Writing with the machine: Gpt-2 and text generation. In *Roguelike Celebration*.

Charalampos Tsioustas, Daphne Petratou, Maximos Kaliakatsos-Papakostas, Vassilis Katsouros, Apostolos Kastritsis, Konstantinos Christantonis, Konstantinos Diamantaras, and Michael Loupis. 2020.

Innovative applications of natural language processing and digital media in theatre and performing arts. In *Proceedings of the ENTRENOVA-ENTerprise REsearch InNOVAtion Conference*, volume 6, pages 84–96.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. Superglue: A stickier benchmark for general-purpose language understanding systems. *CoRR*, abs/1905.00537.

A Appendices

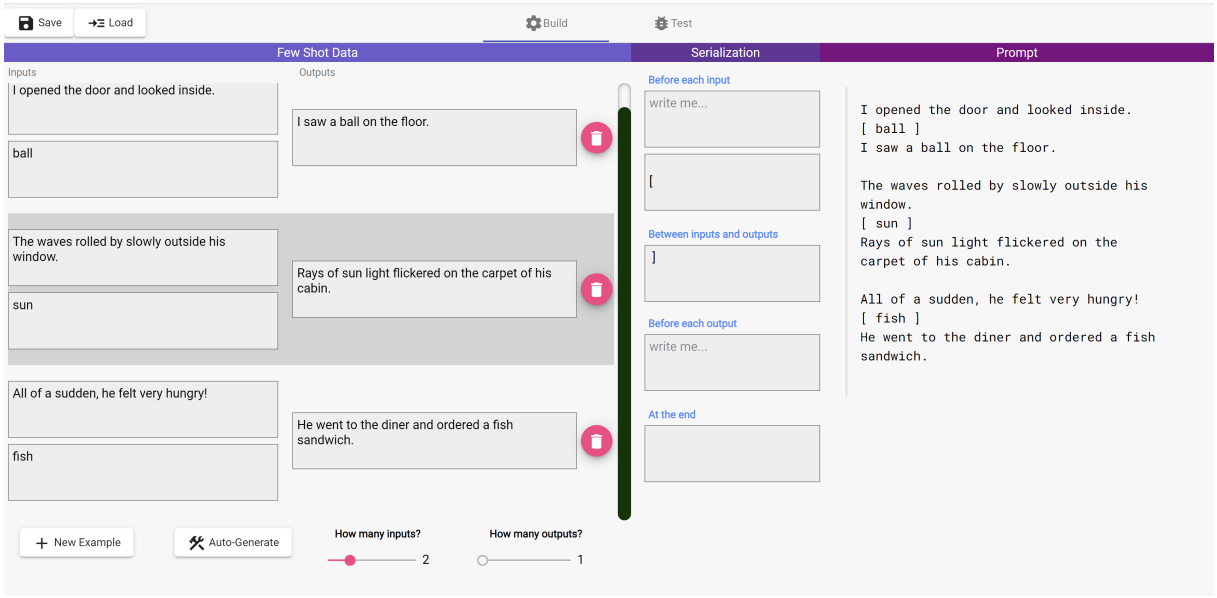
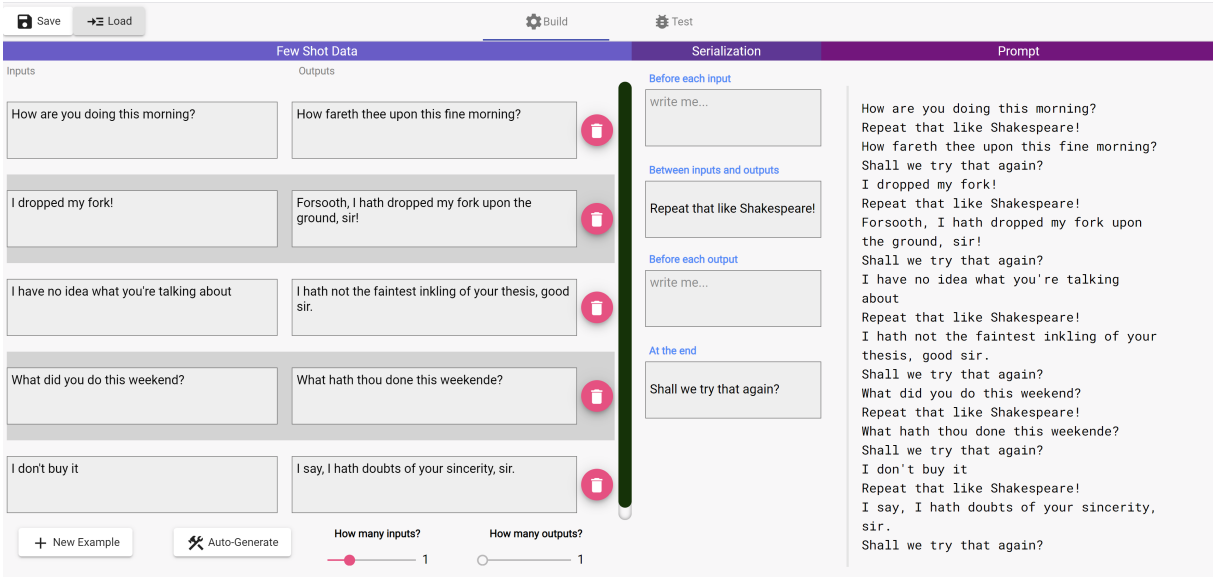


Figure 7: A Screenshot of the Magic Word Formula and Experiment.



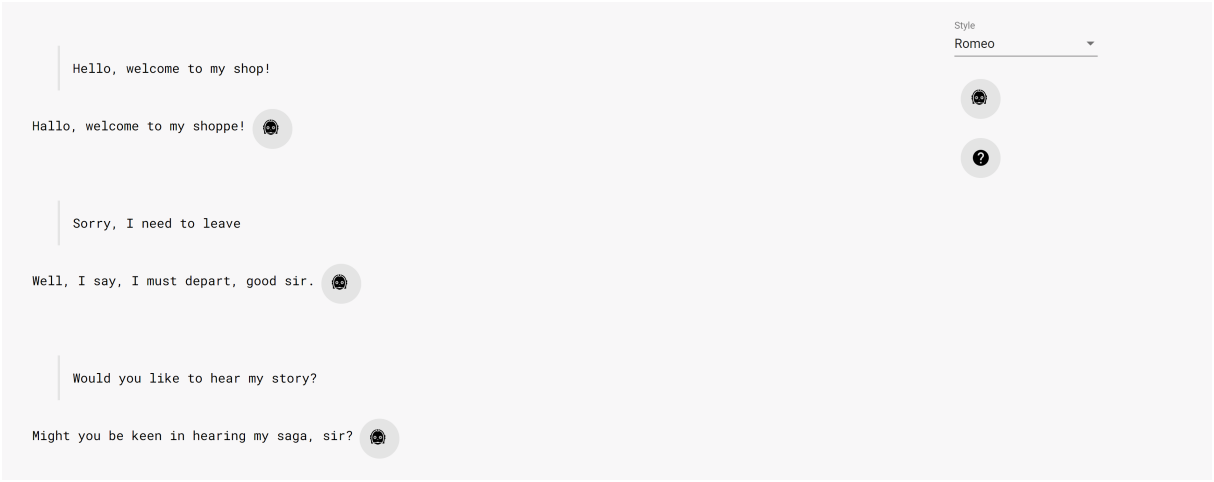


Figure 8: Say It Again in the style of a Shakespearean character using 5 Few Shot Examples.

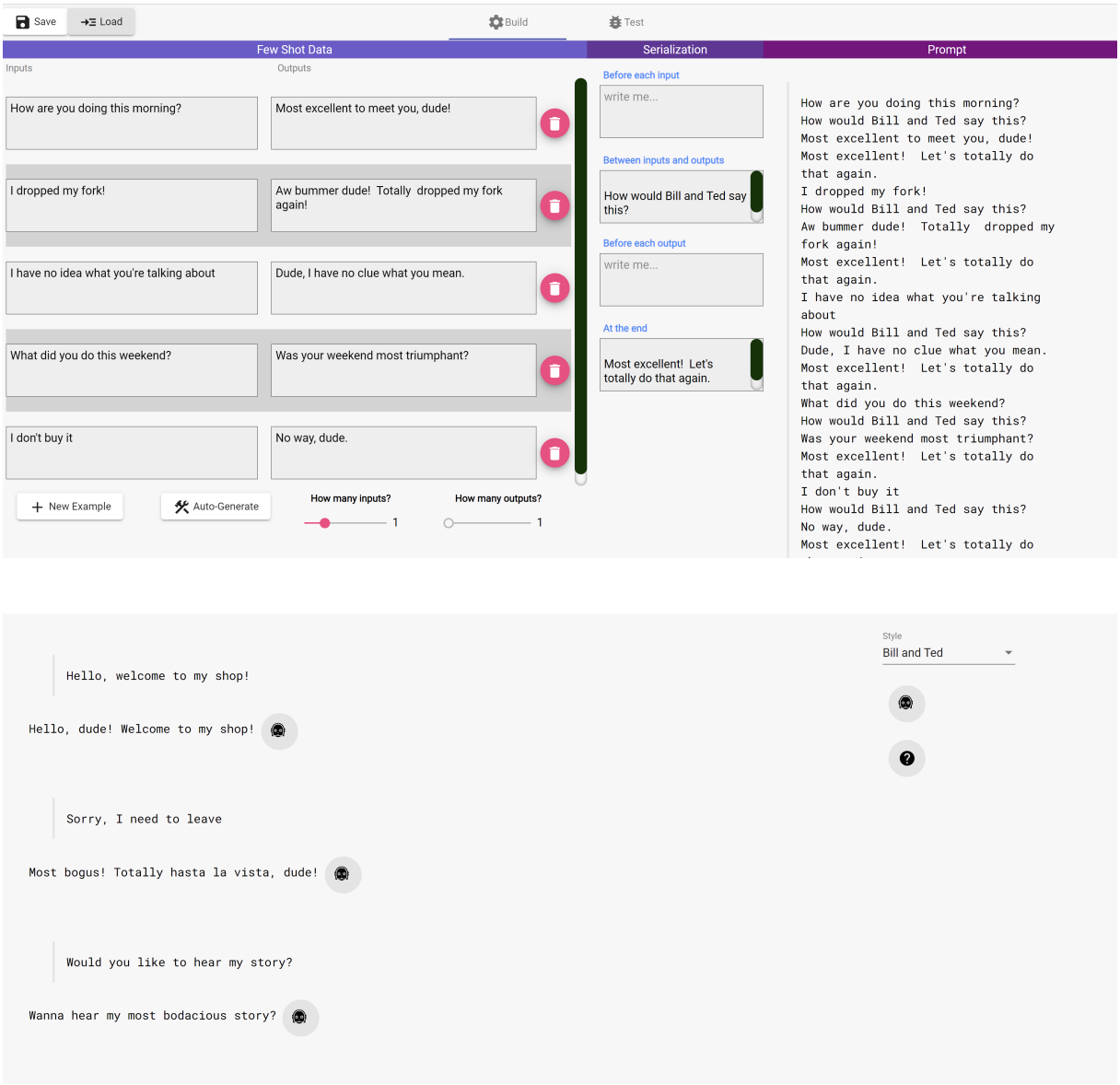


Figure 9: Say It Again in the style of Bill and Ted using 5 Few Shot Examples.

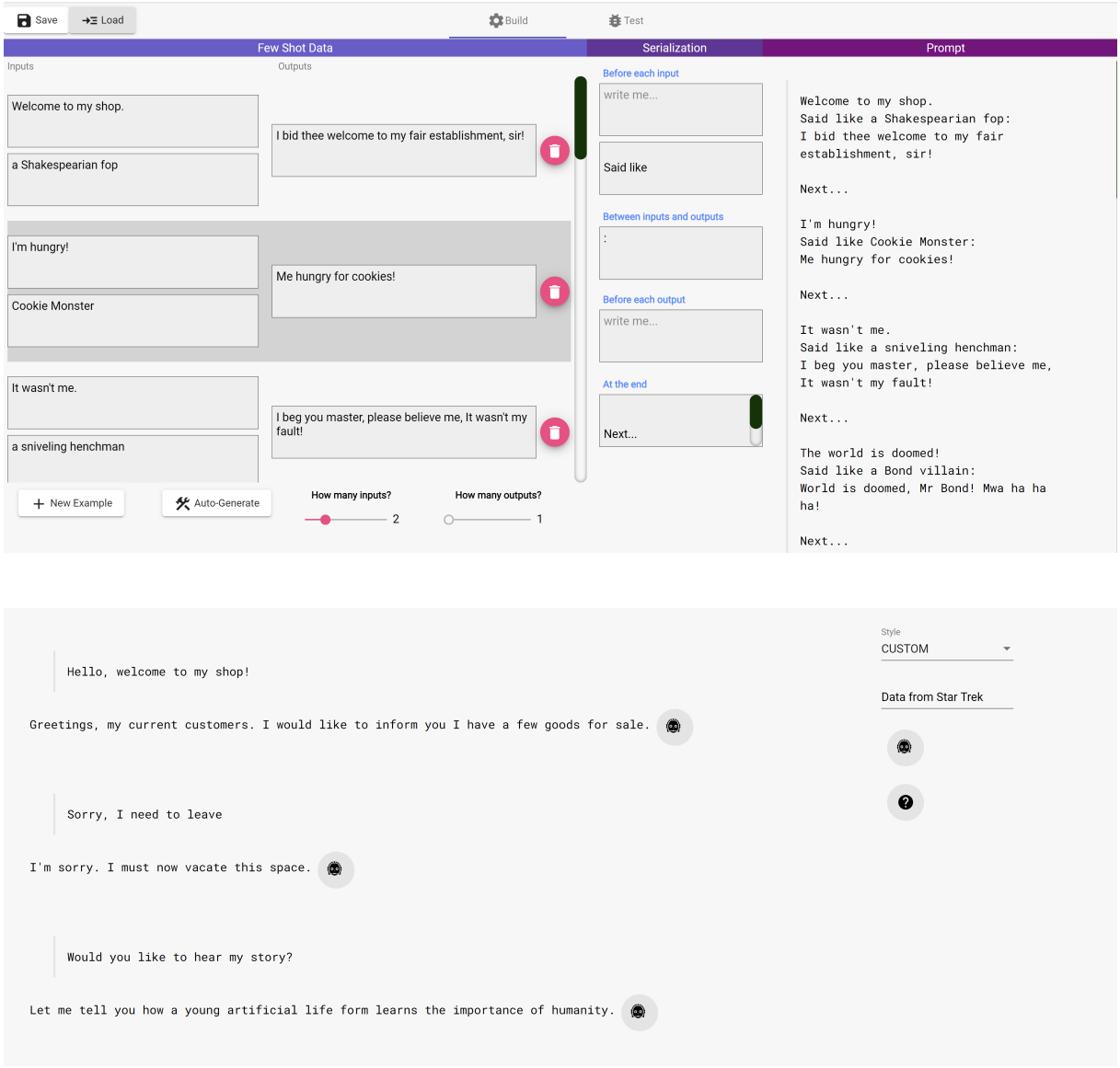


Figure 10: Say It Again in the style of Data from Star Trek using 5 Few Shot Examples of style transfer, but no examples of Data's actual style.

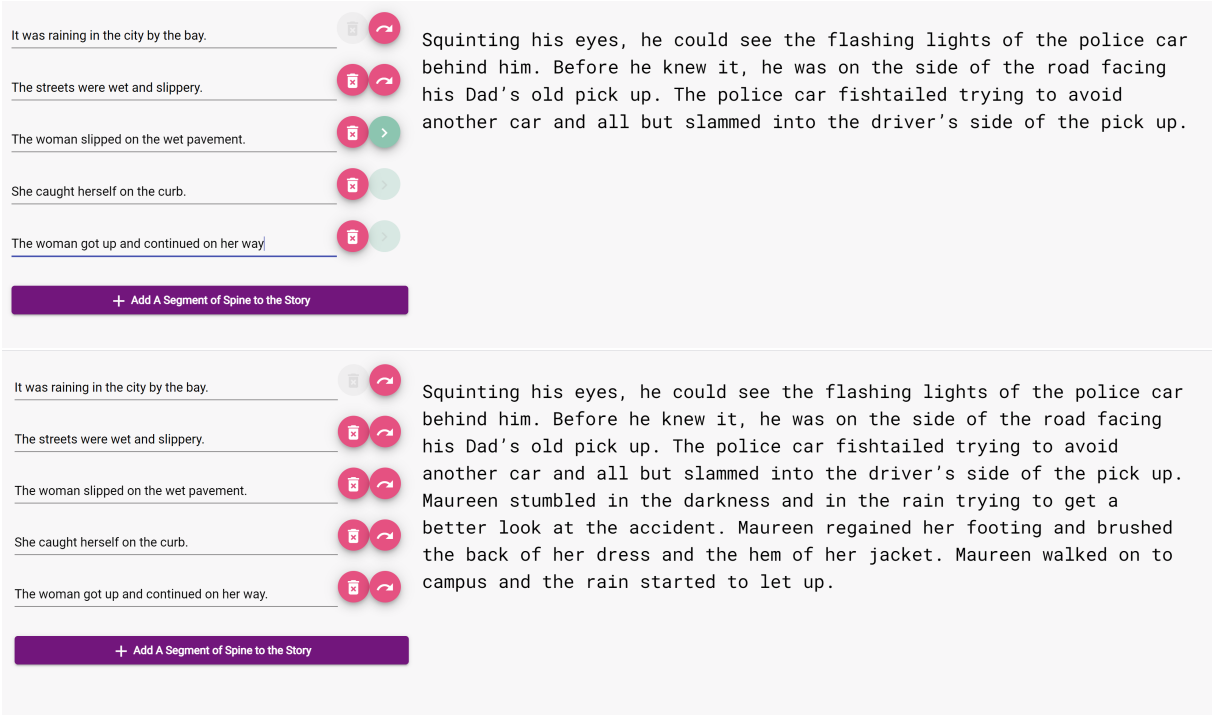


Figure 11: Story Spine screenshots. In the top screenshot, a five segment spine has been constructed, but only two spine segments have been colorized. The bottom image shows the result of colorizing the final three segments.

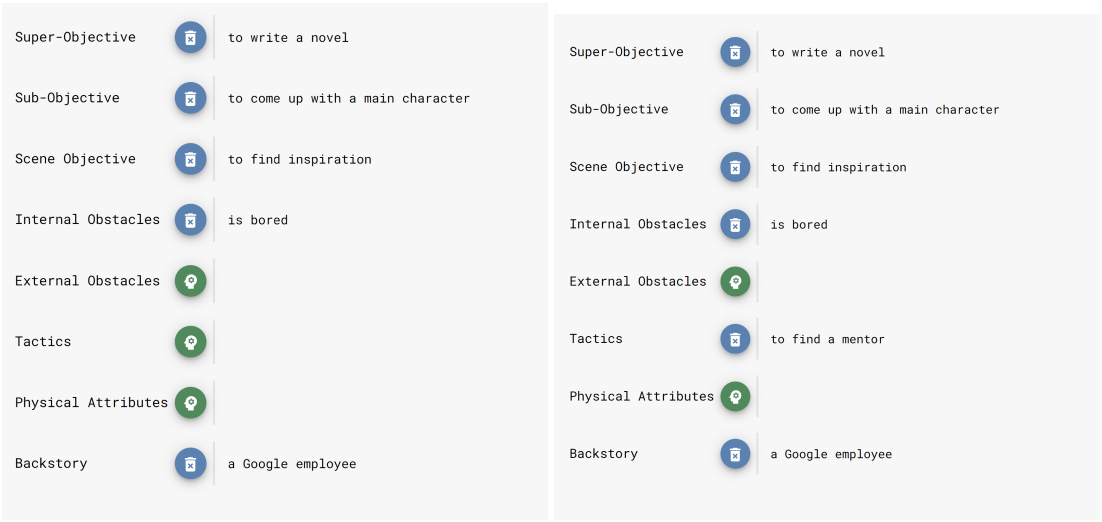


Figure 12: Character Maker, before and after generation of the Tactics field. In this case, generation was performed by constructing a few shot Formula dynamically with Super-Objective, Sub-Objective, Scene Objective, and Backstory as input, and Tactics as output, using 4 examples of full templates to create the Data.