Generating Video Game Scripts with Style

Gaetan Lopez Latouche* and **Laurence Marcotte*** and **Ben Swanson**
Ubisoft La Forge
{gaetan.lopez-latouche,ben.swanson2}@ubisoft.com
laurencemarcotte@hotmail.ca

Abstract

While modern language models can generate a scripted scene in the format of a play, movie, or video game cutscene the quality of machine generated text remains behind that of human authors. In this work, we focus on one aspect of this quality gap; generating text in the style of an arbitrary and unseen character. We propose the Style Adaptive Semiparametric Scriptwriter (SASS) which leverages an adaptive weighted style memory to generate dialog lines in accordance with a character's speaking patterns. Using the LIGHT dataset as well as a new corpus of scripts from twenty-three AAA video games, we show that SASS not only outperforms similar models but in some cases can also be used in conjunction with them to yield further improvement.

1 Introduction

As the affordances of large language models (LLMs) continue to reveal themselves, this technology hints at the possibility of transformative changes to narrative media such as scriptwriting, songwriting and journalism. In this work, we focus on scriptwriting for AAA[1] video game dialog, a domain similar to the scripts used in movies, television and theater but with its own unique flavor that often features larger-than-life characters and action-packed dialogs.

Our particular goal is to advance the ability to incorporate a character style or voice in responses generated by LLMs. The importance of this aspect is motivated by the observation that in a AAA game a character's lines will often be written by several scriptwriters asynchronously, meaning that any assistance in maintaining a consistent style is a boon.

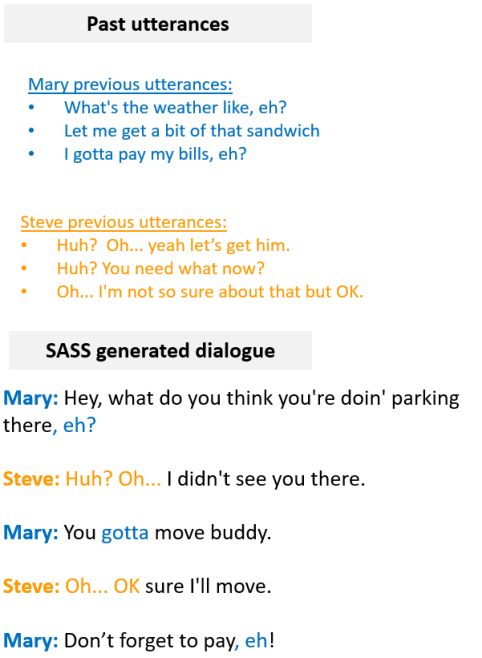

Figure 1: An illustration of our approach. SASS reuses words present in the character's previous conversations to generate character specific stylized responses.

Central to the problem is the representation of a character's style, with recent work using attributes such as target styles (Zhou et al., 2018), character description (Rashkin et al., 2018), previous character utterances (Madotto et al., 2021; Han et al., 2022a) and conversation history (Boyd et al., 2020). The approaches presented can be partitioned into two categories; the first, which we call *explicit style*, consists of a short text sample that explicitly describes the character, their profession, age, interests, and other traits. The second, which we will call *implicit style*, uses a list of previously authored utterances from a character instead.

Considering the ultimate application of these techniques in AAA game development, we propose the use of implicit style provided at inference time as most suitable for several reasons. First, it can be too limiting to summarize the style of a character

*These authors contributed equally to this work.

[1]The term "AAA" refers to multi-million dollar budget productions often with hundreds of highly specialized contributors.

Proceedings of the 5th Workshop on NLP for Conversational AI (NLP4ConvAI 2023), pages 129–139

narrator: Marcus has hacked the final computer:
speakerA: Ladies. Gentlemen. Wrench. You are now talking to the DedSec master.
speakerB: Nice! So how did it end?
speakerA: Well... I signed up with the NSA in exchange for turning over personal data on every DedSec member.
speakerC: Marcus, I am going to hit you.
speakerA: It was a recruitment tool, like I thought. But I did a little extra and erased all traces I was ever there... along with the other two people who had filled the forms. Maybe when the NSA never calls them back, they'll turn to DedSec.
speakerC: Fingers crossed.

Table 1: Example Scene from an AAA game in the UBISCENES dataset.

with a few labels or utterances (Han et al., 2022a). Second, due to production constraints of the already complex narrative pipeline in the video game industry, models should be locked and not involve re-training if possible or else they risk becoming a pipeline blocker. Finally as the writing process naturally develops a character through the course of scriptwriting, using inference time implicit style allows the model to adapt as the character's lines manifest in other scenes of the game. It is worth noting that methods which employ small samples of implicit style (Han et al., 2022a; Suzgun et al., 2022; Reif et al., 2021; Boyd et al., 2020) achieve strong performance but do not fully leverage this continuous increase of character's lines.

To harness the signal of implicit style we build on the k-nearest neighbour language model (KNN-LM) (Khandelwal et al., 2019), adapting and improving it for the task of style-controlled generation. Our model, the Style Adaptive Semiparametric Scriptwriter (SASS), provides a drop-in replacement for a traditional language model architecture that scales well with the number of reference character lines supplied as implicit style, does not require added work from the script writers to keep up to date, and can be used orthogonally to other methods of style-controlled dialog generation. An illustration of our method is shown in Figure 1. Our automatic evaluations show that SASS generates responses that are more aligned with a target character style without sacrificing fluency when compared to both KNN-LM and a finetuned LLM baseline.

2 Related Work

We continue a long thread of study in style controlled dialog generation and the closely related topic of style transfer where the term style is heavily overloaded, often treating style as a categorical variable such as emotion, formality, or sentiment (Kong et al., 2021; Dathathri et al., 2019; Prabhumoye et al., 2018). We differentiate this notion of *ephemeral* style from the *character* style which is always present to some degree in a character's speech regardless of situation, of which the latter is our focus.

Recent research into methods incorporating explicit style has been fueled primarily by the PERSONA-CHAT (Zhang et al., 2018) and LIGHT datasets (Urbanek et al., 2019). Examples include Kim et al. (2022) which augments personas during inference and Madotto et al. (2021) which utilizes the persona in LLM prompting. Previous approaches to the use of implicit style vary from concatenation of the references to the conversation history (Boyd et al., 2020), to the construction of artificial prepended dialog (Han et al., 2022b), to approaches more similar to our own which seek to directly capture the frequent words used by a character as a proxy for their style (Fikri et al., 2021; Liu et al., 2020). Another approach to implicit style generation and transfer is learn a mapping to a vector style encoding (Li et al., 2020a; Riley et al., 2020a) which allows for inference time adaptation to arbitrary styles.

Our work also serves as a direct improvement to the k-nearest neighbour language model (Khandelwal et al., 2019) for which some previous attention has been paid to the intersection with style conditioning (Trotta et al., 2022).

3 Dataset

As no public dataset of video game dialog exists, we leverage our privileged access to the back catalog of all UBISOFT games to build one. From a pool of 23 games that are sufficiently narrative

	Train	*Valid*	*Test*
Games	16	3	4
Characters	3,514	477	291
Scenes	16,458	1,727	1,403
Utterances	107,222	14,058	12,170
Vocabulary Size	29,200	13,723	13,555
Utterance Length	15.82	15.11	16.20
Character/Scene	2.99	3.42	3.34
Utterance/Scene	6.51	8.14	8.67

Table 2: UBISCENES dataset statistics after filtering.

heavy, we collect 19,588 well filtered scenes featuring 4,282 characters and 133,450 lines of dialog, and refer to this dataset as UBISCENES. For filtering, we use a combination of thresholds on automatic metrics (the rate at which the same character speaks twice in a row as well as the entropy of the identity of the speaker across the scene) and game specific rules to accommodate the quirks of each production. Table 1 and Appendix A2 show examples of scenes that we collected. We split the dataset by game into training, validation, and test sets to avoid data leakage. Overall statistics of the collected dataset are given in table 2.

We also evaluate on the LIGHT dataset (Urbanek et al., 2019) which is, in our opinion, the most similar academic dataset to video game dialog despite its many differences. Specifically, we note the disparity in terms of writing quality between these two datasets: UBISCENES is composed of professionally authored text while LIGHT is created by crowdworkers. They also differ structurally as many dialogues in UBISCENES have a narrator involved which is not the case in LIGHT or other dialog datasets such as PERSONACHAT (Zhang et al., 2018), WIZ. OF WIKIPEDIA (Dinan et al., 2018), DAILY DIALOG (Li et al., 2017) or HLA-CHAT (Li et al., 2020b).

For both datasets, we replace all script cue names with an added special token <*speaker*> for a simple speaker or <*narrator*> for a narrator, although names are preserved in the dialog text. We evaluate only on dialogues where the target character has spoken strictly less than three times to avoid relying on the previous conversation history as a style indicator. 60% of the scenes are used to supply the indices of implicit style and we study the twenty characters with the highest number of lines in the remaining 40%.

4 Method

Our choice of model arises from our guiding hypothesis that a principal component of character style is simply their preferential choice of words that are either optional or semantically exchangeable with other words in context. SASS consists of two components:

- An autoregressive transformer (Vaswani et al., 2017) language model that encodes the dialog context.
- A non-parametric token retrieval module with access to each character's *style index*: a collection of all of their previously authored lines.

Both components provide a categorical distribution over the token vocabulary of the language model, and a *style adapter* combines these two components. Our model architecture draws on KNN-LM(Khandelwal et al., 2019) and Adaptive Semiparametric Language Models (Yogatama et al., 2021), improving on the gating mechanism of the latter to better choose when to leverage implicit style and when to rely on LLM generation.

4.1 Base Model

The transformer architecture (Vaswani et al., 2017) is our base model, using the GPT-J (Wang and Komatsuzaki, 2021) model from Huggingface Transformers (Wolf et al., 2019) as our initial pretrained language model. This model contains 6 billion parameters with a vocabulary size of 50,400 tokens.

This model is finetuned on the training split of our dataset and provides p_{LM}, the categorical language model probability of the next token of the dialog being generated. Note that at inference time the only representation of the character's style that is available to this model are the previous lines in the dialog.

4.2 Character Style Index

As in Trotta et al. (2022) each character has its own character style index with an average of 425 entries in the UBISCENES dataset and 1,404 in LIGHT. Given a character's implicit style as a list of strings $\mathcal{C}$, we formally define its style index $\mathcal{S}$ as the following set of key-value pairs:

$$\mathcal{S} = \bigcup_{s \in \mathcal{C}} \{(f(w_{i-}), w_i) \forall w_i \in s\}$$

where $f(w_{i-})$ is a vector encoding of the prefix of s at index i, but before the decision to produce w_i has been made.

Previous work has relied on the last layer hidden state of the LLM at index i as a definition of f (Khandelwal et al., 2019). Given the arbitrary relative scale and redundancy of the hidden state's parameters before its transformation into a predictive distribution over tokens, we propose the alternative use of the actual categorical probability distribution given by the language model at the i_{th} position to provide f. As the vocabulary size of GPT-J (Wang and Komatsuzaki, 2021) is high, we reduce the dimension of this probability distribution to a 768 long vector using PCA (Abdi and Williams, 2010).

At inference time we are given the input dialog context c and the style index of the currently speaking character, and we retrieve the k-nearest neighbors of $f(c)$ among the keys of $\mathcal{S}$ using L2 distance. Early qualitative analysis suggested that $k = 10$ gave reasonable results, and we leave the investigation of the impact of k on our method to future work.

As in the k-nearest neighbor language model (Khandelwal et al., 2019) we softmax a vector with a large negative number in all locations except for the retrieved tokens indices which are set to the negative L2 distance obtained during retrieval. This distribution over the LLM vocabulary is returned by the k-nearest neighbor component, which we will refer to as p_{kNN}.

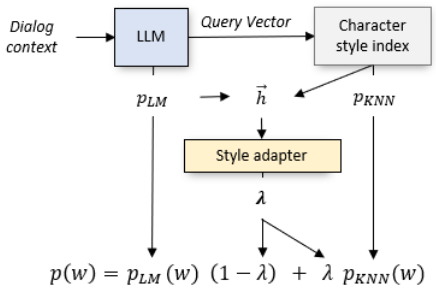

Figure 2: An illustration of SASS. Given a dialog context and a character style index, a query vector is constructed, a set of k nearest neighbors are retrieved and part of $\vec{h}$ is created. We extract from p_{LM} the probability of those k tokens and concatenate it to $\vec{h}$. $\vec{h}$ is then passed to a style adapter and returns the interpolation parameter λ.

4.3 Style Adapter

Once equipped with both p_{LM} and p_{kNN}, all that remains is to interpolate these distributions to predict the next token. We predict a linear interpolation parameter λ from the concatenation of the transformer's last hidden state, the raw distances (L2 distance between the query vector and the retrieval key) and the probability of the tokens retrieved under p_{LM} which we denote as $\vec{h_i}$ with dimension $d_{\vec{h_i}} = 2 * k + d_{embd}$ where d_{embd} is the dimension of the hidden states of the LLM. This gives the full token prediction probability distribution returned by SASS as

$$\begin{aligned} \lambda &= \sigma(\mathbf{W}\vec{h_i}) \\ p(w_{i+1}|w_{\leq i}) &= \lambda p_{\text{kNN}}(w_{i+1}|w_{\leq i}) \\ &\quad +(1-\lambda)p_{\text{LM}}(w_{i+1}|w_{\leq i}) \end{aligned}$$

where σ is the sigmoid function, and $\mathbf{W}$ is a parameter vector of size $d_{\vec{h_i}}$.

Intuitively, each component of $\vec{h_i}$ provides complementary information to the style adapter: the last hidden state gives a representation of the current dialog context, the raw distances show how confident the model is in the retrieved tokens, and the probability of the tokens retrieved under p_{LM} reveals the appropriateness of each retrieved token in the context. For nearest neighbor retrieval we use the FAISS library (Johnson et al., 2019) and use L2 as a distance metric as in Khandelwal et al. (2019). We train SASS with a learning rate of 2e-4 for the style adapter and 2e-5 for the LLM on the training set for one epoch. An illustration of SASS architecture is depicted in Figure 2.

5 Experiments

5.1 Evaluation

Following previous works on text style transfer (Li et al., 2018; Smith et al., 2020; Riley et al., 2020b) and style-controlled dialog agents (Han et al., 2022a), we train two multi-class classifiers on the utterances of the characters present in the test set (twenty characters per dataset) of UBISCENES and LIGHT(Urbanek et al., 2019) respectively. We denote $StyleAcc$ the classifier accuracy of predicting the target character, where a higher value indicates that generated text is more closely aligned to characters' styles.

To calculate $StyleAcc$ we use all dialog histories where the test characters have 0-2 previous lines. We also report $StyleAcc_0$ and $StyleAcc_1$,

Dataset	Context	5	10	25	50	100
UbiScenes	$knn_{p_{LM}}$	**44.03**	**48.98**	**54.79**	**58.95**	**62.61**
	knn^*	31.31	36.06	42.05	46.66	51.41
LIGHT	$knn_{p_{LM}}$	**38.85**	**45.63**	**53.60**	**59.33**	**64.52**
	knn^*	33.79	41.11	49.30	54.07	57.95

Table 3: Retrieval Recall for different k number of retrieved tokens. Using the language model probability p_{LM} improves recall for all k compared to the same method using the final hidden state state as a retrieval key (knn*).

the accuracy of the classifier when the character has exactly 0 or 1 previous lines in the dialogue history.

To measure how similar the vocabulary used in the generated responses is to its corresponding style index, we compute the n-gram overlap (where n=2) as done in (Han et al., 2022a) who define the n-gram overlap as the percent of n-grams in the generated line that appear anywhere in the style indices.

To check for a degenerate solution that represents style at the cost of fluency, we follow previous work on language models with external memory (Khandelwal et al., 2019; Trotta et al., 2022; Yogatama et al., 2021; Bhardwaj et al., 2022) and report perplexity. To validate the choice of our alternative encoding f of the retrieval key, we measure the quality of the retrieved tokens with recall over the retrieved tokens as in Bhardwaj et al. (2022).

5.2 Baseline Methods

Full-dataset Fine-tuning (SCRIPTWRITER): This straightforward baseline simply finetunes the vanilla GPT-J model on the training set, and is equivalent to fixing the style adapter's interpolation parameter λ to zero and ignoring p_{kNN}.

PDP Random Match: (PDP_r): PDP (Han et al., 2022a) constructs and prepends an artificial dialog before the input dialog context, effectively providing a Scriptwriter style model with a small selection of the style index. In their work, the authors use one of several pseudo-contexts, and present models that select the pseudo-context to be used based on the dialog history. We implement a variation of their Random Match method which selects the pseudo-context at random, with the difference that in our case we use a character's previous scenes directly. While this diverges from their exact approach, our goal of comparison in this case is not to show that one model is better than the other but instead to demonstrate that they can complement each other.

Adapted kNN-LM ($kNN\text{-}LM_r$): Our work is directly inspired by language models with external memory. Our approach is closely related to Khandelwal et al. (2019) with key modifications on the retrieval representation and the dynamic runtime calculation of the interpolation parameter. We use the SCRIPTWRITER as the base LLM required to compute the retrieval keys and tune lambda using the validation set. Comparing to this strong baseline allows us to determine if our style adapter has learned how to interpolate between the style of the character and the LLM more efficiently than using a constant interpolation term.

5.3 Additional Studies

In addition to the evaluation metrics presented above, we perform some extra experiments to validate our model. First, we shuffle the indices of each character to ensure that we observe a decrease in performance; if the model is simply leveraging game specific proper nouns as a proxy for character style then it would not be effected by using the wrong character index.

Second, we also perform data ablation on the size of the style indices to investigate at which point in the writing process our method can achieve improvements over simple finetuning.

6 Results & Discussion

Our results demonstrate that both our use of the PCA probability distribution as a retrieval key as well as a dynamic style adapter lead to improvements over the strong baseline of $kNN\text{-}LM_r$. We also show that SASS can be used in combination with other methods such as PDP (Han et al., 2022a) and explore the effects of size of the style index on performance.

Considering first the use of the PCA probability distribution as an alternative to the LLM hidden state used in k-nearest neighbor language models as a retrieval key, Table 3 demonstrates improved recall on both datasets.

	PPL	$StyleAcc$	$StyleAcc_0$	$StyleAcc_1$	$N-gram$
REAL	X	.6868	.701	.683	X
SCRIPTWRITER	35.873	.316	.232	.352	.186
$kNN\text{-}LM_r$	27.016	.409	.359	.427	.212
SASS	**23.055**	**.458**	**.413**	**.483**	**.233**
SASS SHUFFLED	38.700	.233	.160	.255	.177
PDP_r	32.786	.424	.364	.455	**.255**
$PDP_r + kNN\text{-}LM_r$	29.889	.473	.433	.493	.230
$PDP_r + SASS$	**27.394**	**.510**	**.467**	**.539**	.248

Table 4: Results for our automatic evaluation on UBISCENES. Best results are highlighted in bold for each metric. SASS generally outperforms its baseline method on all studied metrics.

UBISCENES: The results on our new dataset of AAA video game dialogs is shown in Table 4 and demonstrate that SASS outperforms all other baselines on all metrics except for N-Gram overlap where it is second best to PDP_r. Additionally, adding SASS or $kNN\text{-}LM_r$ to PDP_r(Han et al., 2022a) leads to better perplexity and generally superior style specific metrics compared to PDP_r alone.

The central result that deserves highlighting is that SASS outperforms the closely related $kNN\text{-}LM_r$ on all metrics, showing that our dynamic style adapter can effectively learn when to give importance to the style index of the character over the language model or vice versa. It is also worth noting the gap on the style metrics between all the models under experiment and the real scripts, hinting at the large amount of improvement still required to approach human authored quality. Example outputs of SASS can be found in Appendix A1.

LIGHT: Results for the LIGHT dataset are shown in Table 5. We first note that the performance of the classifier on the gold dialogs is considerably lower on this dataset compared to UBISCENES with an accuracy of 30.86% compared to 68.68%. We take this as evidence of our own qualitative assessment that characters in LIGHT do not have a strong style and as such it is difficult for the classifier to guess which utterance was written by whom. The smaller relative performance improvement of SASS and $kNN\text{-}LM_r$ over SCRIPTWRITER validate this hypothesis as does the fact that shuffling the character style index also has a low impact on style aware metrics. Overall, this demonstrates that LIGHT is not ideal for research in style based dialog and highlights the potential of professionally authored datasets such as UBISCENES.

6.1 Additional study results

Our data ablation study is especially important given our stated domain of AAA video game scriptwriting as our non-parametric design instantly integrates any lines spoken by a character as they are written, and data ablation viewed in reverse simulates this writing process. Figure 4 shows the results of this study. We also investigate the effect of randomly shuffling the characters' indices, expecting to see a drop in performance as long as our gains in style are not due to game level word frequencies like proper nouns but instead to actual character speaking patterns. Results of SASS SHUFFLED in table 4 reveals the outcome of this study.

Decreasing the number of entries in style index reduces performance, but, even at 10% of the original 60% of game data that was held out to create the style index SASS yields better perplexity and style specific scores compared to the SCRIPTWRITER baseline. This suggests that SASS has value even at the beginning of the writing process, which is perhaps when it can be most helpful to writers.

As hoped, replacing the character style index of our characters by the one of another character (SASS SHUFFLED) significantly decreases the performance in terms of both perplexity and style aware metrics. This demonstrates that not only do the characters in UBISCENES have their own style but also that SASS can leverage these styles effectively.

	PPL	$StyleAcc$	$StyleAcc_0$	$StyleAcc_1$	$N-gram$
REAL	X	.309	.330	.310	X
SCRIPTWRITER	29.567	.141	.113	.150	.370
$kNN\text{-}LM_r$	30.730	**.172**	**.150**	**.183**	.376
SASS	**26.759**	.157	.132	.166	**.390**
SASS SHUFFLED	27.474	.135	.105	.149	.379

Table 5: Results for our automatic evaluation on LIGHT. Best results are highlighted in bold for each metric. SASS and $kNN\text{-}LM_r$ generally outperforms the baseline method on style specific metrics.

6.2 Style adapter Analysis

We conduct some exploratory visualizations of the style adapter whose purpose is to adjust the importance of the non-parametric retrieved style at each generation step. Figure 3 shows histograms over the validation set of the distribution of values returned by our style adapter for λ when decoded with teacher forcing on the gold outputs.

We observe that in both LIGHT and UBISCENES lambda settles into a bimodal distribution with one peak near zero, which corresponds to ignoring the style index and relying on the language model instead. This Figure also reinforces the difference in performance gain of SASS on the two datasets; it is clear that with UBISCENES the values of λ are more varied which is evidence that the style adapter has found a signal with which to give a more nuanced prediction.

Finally we note that in both datasets λ is rarely much greater than .5, indicating that the style adapter is reluctant to fully disengage the language model. While this may indeed be optimal behavior, we suspect that this is a side effect of our architecture and potential avenue of improvement for this model class. To see the dilemma, consider that the gradient of lambda on a single prediction will only be positive if the gold token is actually retrieved regardless of the quality of the actual retrieved tokens which may be perfectly appropriate.

6.3 Discussion and Future Work

Our quantitative results demonstrate that SASS not only outperforms the strong baseline $kNN\text{-}LM_r$ but also can be used complementary to prompt editing based style control such as PDP_r. We performed qualitative pairwise comparison experiments with earlier versions of the model but did not achieve acceptable inter-annotator agreement. We attribute this to the subjectivity of choosing the better of two possible dialog lines once both lines are grammatically correct and coherent with the scene as are most outputs from all our models including the baseline. Furthermore, to provide raters with a rubric on which to base a choice of the more stylish output we must necessarily boil the character's style down into a short description or a few sample lines, which is a lossy and imprecise operation.

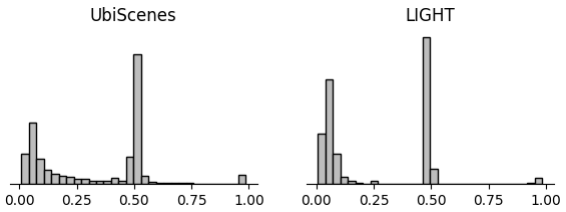

Figure 3: Distributions of values of λ on UBISCENES (left) and LIGHT (right) on the validation sets.

Our opinion is that the output of all of these models is "good enough" to be used as a writing aid, either to provide starter text for editing or simply to spur forward the creative process through inspiration. None of our models can be used as a substitute for actual professional scriptwriters, as is evidenced by the remaining gap in our automatic style metrics between SASS and the human authored lines. Nevertheless, we see clear qualitative evidence that SASS is making use of characters' speaking patterns without too much impact on fluency and coherence.

Our reliance on perplexity as a proxy for fluency is an area for improvement in our methodology, and there exist methods for formal quantization of coherence, topic and fluency in the literature (Aksitov et al., 2023). Although SASS leads to perplexity improvement, qualitative evaluations have shown that it could sometimes lead to a small decrease in fluency. We also note opportunities for further experimentation in the optimal choice of the number of retrieved neighbors k, as this could easily be

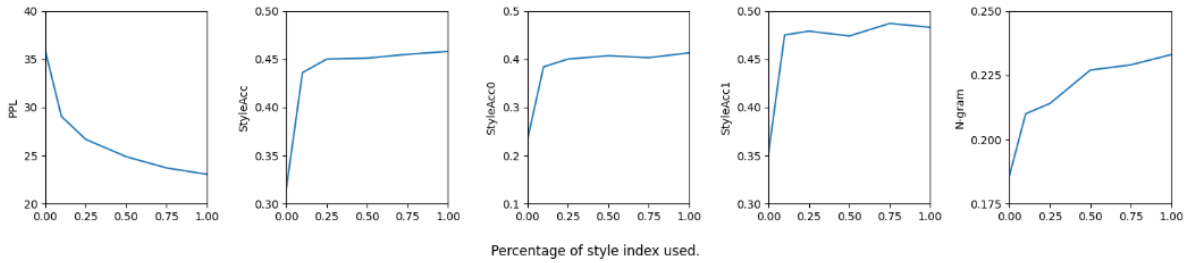

Figure 4: Effect of SASS character style index size on our studied metrics.

increased without sacrificing significant additional latency.

7 Conclusion

We present the Style Adaptive Semiparametric Scriptwriter (SASS), drawing inspiration from the closely related k nearest neighbor and adaptive semiparametric language models. In particular we propose new formulations of the encoder for retrieval as well as the determination of the style interpolation parameter and demonstrate that they lead to improved performance. Ablation studies reveal that the benefits of our approach manifest even with small amounts of reference style index material, and it is our intention to integrate this in our internal writing assistance tools in the near future.

We perform experiments on two datasets, the LIGHT dataset as well as a first of its kind dataset of video game script dialogs, UBISCENES, and demonstrate that the difference in style between professionally authored and crowdsourced text is a crucial consideration for style controlled generation research. We regret that we cannot release UBISCENES publicly due to concerns of its use in products that do not respect the intellectual property of their data sources. However, we are open to speak with academic collaborators that are interested in working with this data for targeted projects and invite them to reach out to the authors.

Limitations

The main limitation of our proposed method relies on the additional cost of retrieval. Even if the size of our character style indexes is small it still adds latency to our overall pipeline as retrieval must occur once per token. We expect that incorporating the recent work of He et al. (2021) on improving the efficiency of nearest neighbor language models should decrease this latency significantly.

As in most NLG work, another important limitation is in quality evaluation. We found qualitative evaluations to be too imprecise for appropriate inter-annotator agreement, and the quantitative evaluations that we present in this paper are all proxies that cannot be said to capture character style or fluency in full.

Another limitation of our work is the exclusion of models that are only accessible by calling or finetuning powerful external language model APIs due to the excessive monetary cost involved. It is almost certain that these larger models would outperform the 6B parameter model we use, and this may also change the relative performance of the techniques that we present. While we feel that this constraint is appropriate at this moment in history and that our position as major AAA developer gives us the authority to make such a claim, shifts in third party model availability and pricing could change the landscape.

Our work deals with data of a singular domain, video game scripts in English, but represents a wide variety of nationalities and ethnicities over the span of a large catalog of games.

References

Hervé Abdi and Lynne J Williams. 2010. Principal component analysis. *Wiley interdisciplinary reviews: computational statistics*, 2(4):433–459.

Renat Aksitov, Chung-Ching Chang, David Reitter, Siamak Shakeri, and Yunhsuan Sung. 2023. Characterizing attribution and fluency tradeoffs for retrieval-augmented large language models. *arXiv preprint arXiv:2302.05578*.

Rishabh Bhardwaj, George Polovets, and Monica Sunkara. 2022. Adaptation approaches for nearest neighbor language models. *arXiv preprint arXiv:2211.07828*.

Alex Boyd, Raul Puri, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Large scale

multi-actor generative dialog modeling. *arXiv preprint arXiv:2005.06114*.

Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. 2019. Plug and play language models: A simple approach to controlled text generation. *arXiv preprint arXiv:1912.02164*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Abdurrisyad Fikri, Hiroya Takamura, and Manabu Okumura. 2021. Stylistically user-specific response generation. *Journal of Natural Language Processing*, 28(4):1116–1140.

Seungju Han, Beomsu Kim, Jin Yong Yoo, Seokjun Seo, Sangbum Kim, Enkhbayar Erdenee, and Buru Chang. 2022a. Meet your favorite character: Open-domain chatbot mimicking fictional characters with only a few utterances. *arXiv preprint arXiv:2204.10825*.

Seungju Han, Beomsu Kim, Jin Yong Yoo, Seokjun Seo, Sangbum Kim, Enkhbayar Erdenee, and Buru Chang. 2022b. Meet your favorite character: Open-domain chatbot mimicking fictional characters with only a few utterances.

Junxian He, Graham Neubig, and Taylor Berg-Kirkpatrick. 2021. Efficient nearest neighbor language models. *arXiv preprint arXiv:2109.04212*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2019. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547.

Urvashi Khandelwal, Omer Levy, Dan Jurafsky, Luke Zettlemoyer, and Mike Lewis. 2019. Generalization through memorization: Nearest neighbor language models. *arXiv preprint arXiv:1911.00172*.

Minju Kim, Beong-woo Kwak, Youngwook Kim, Hongin Lee, Seung-won Hwang, and Jinyoung Yeo. 2022. Dual task framework for improving persona-grounded dialogue dataset. *CoRR*.

Xiangzhe Kong, Jialiang Huang, Ziquan Tung, Jian Guan, and Minlie Huang. 2021. Stylized story generation with style-guided planning. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 2430–2436, Online. Association for Computational Linguistics.

Aaron W Li, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020a. Aloha: Artificial learning of human attributes for dialogue agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8155–8163.

Aaron W Li, Veronica Jiang, Steven Y Feng, Julia Sprague, Wei Zhou, and Jesse Hoey. 2020b. Aloha: Artificial learning of human attributes for dialogue agents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8155–8163.

Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: a simple approach to sentiment and style transfer. *arXiv preprint arXiv:1804.06437*.

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957*.

Siyi Liu, Ziang Leng, and Derry Wijaya. 2020. Learning to mirror speaking styles incrementally. *arXiv preprint arXiv:2003.04993*.

Andrea Madotto, Zhaojiang Lin, Genta Indra Winata, and Pascale Fung. 2021. Few-shot bot: Prompt-based learning for dialogue systems. *arXiv preprint arXiv:2110.08118*.

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W Black. 2018. Style transfer through back-translation. *arXiv preprint arXiv:1804.09000*.

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2018. Towards empathetic open-domain conversation models: A new benchmark and dataset. *arXiv preprint arXiv:1811.00207*.

Emily Reif, Daphne Ippolito, Ann Yuan, Andy Coenen, Chris Callison-Burch, and Jason Wei. 2021. A recipe for arbitrary text style transfer with large language models. *arXiv preprint arXiv:2109.03910*.

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

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

Eric Michael Smith, Diana Gonzalez-Rico, Emily Dinan, and Y-Lan Boureau. 2020. Controlling style in generated dialogue. *arXiv preprint arXiv:2009.10855*.

Mirac Suzgun, Luke Melas-Kyriazi, and Dan Jurafsky. 2022. Prompt-and-rerank: A method for zero-shot and few-shot arbitrary textual style transfer with small language models. *arXiv preprint arXiv:2205.11503*.

Severino Trotta, Lucie Flek, and Charles Welch. 2022. Nearest neighbor language models for stylistic controllable generation. *arXiv preprint arXiv:2210.15762*.

Jack Urbanek, Angela Fan, Siddharth Karamcheti, Saachi Jain, Samuel Humeau, Emily Dinan, Tim Rocktäschel, Douwe Kiela, Arthur Szlam, and Jason Weston. 2019. Learning to speak and act in a fantasy text adventure game. *arXiv preprint arXiv:1903.03094*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Ben Wang and Aran Komatsuzaki. 2021. Gpt-j-6b: A 6 billion parameter autoregressive language model.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Dani Yogatama, Cyprien de Masson d'Autume, and Lingpeng Kong. 2021. Adaptive semiparametric language models. *Transactions of the Association for Computational Linguistics*, 9:362–373.

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243*.

Hao Zhou, Minlie Huang, Tianyang Zhang, Xiaoyan Zhu, and Bing Liu. 2018. Emotional chatting machine: Emotional conversation generation with internal and external memory. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32.

Appendix

Human Input: Join us, the show is about to start.
Eh character: Eh, it's a little soon for all that, we need to set up!
Huh character: I'm sorry I'm not in the mood to talk huh.
Speaker from an AAA game: Just don't get us killed, OK? This is not the place to play.
Speaker from an AAA game: Oh, thank God!
Narrator from an AAA game: Edward enters and walks in.
Narrator from an AAA game: ext. UNDERWORLD of THEATERS.

Table A1: Example outputs from SASS for different well known video game characters and two example characters (one usually starting its sentences by "Eh" and the other finishing with "huh").

narrator: This is the first line that plays during the dialog for big battles occurring at sea. It is followed by an accept/decline hub.
narrator: This greeting plays when Athens is on the offensive, and when the player is at a medium to high level in game.
speakerA: The mighty Eagle Bearer. Rumor has it you command one of the fiercest ships at sea. Maybe you'd be interested in making some drachmae off it?
speakerB: Depends how.
speakerA: Join Athens as we set sail to destroy the Spartan navy... that's all.

Table A2: Example Scene from an AAA game in the UBISCENES dataset.