Narrative Text Generation with a Latent Discrete Plan

Harsh Jhamtani [1] **Taylor Berg-Kirkpatrick** [2]
[1] School of Computer Science, Carnegie Mellon University
[2] Computer Science and Engineering. University of California San Diego
jharsh@cs.cmu.edu, tberg@ucsd.eng.edu

Abstract

Past work on story generation has demonstrated the usefulness of conditioning on a generation plan to generate coherent stories. However, these approaches have used heuristics or off-the-shelf models to first tag training stories with the desired type of plan, and then train generation models in a supervised fashion. In this paper, we propose a deep latent variable model that first samples a sequence of anchor words, one per sentence in the story, as part of its generative process. During training, our model treats the sequence of anchor words as a latent variable and attempts to induce anchoring sequences that help guide generation in an unsupervised fashion. We conduct experiments with several types of sentence decoder distributions – left-to-right and non-monotonic, with different degrees of restriction. Further, since we use amortized variational inference to train our model, we introduce two corresponding types of inference network for predicting the posterior on anchor words. We conduct human evaluations which demonstrate that the stories produced by our model are rated better in comparison with baselines which do not consider story plans, and are similar or better in quality relative to baselines which use external supervision for plans. Additionally, the proposed model gets favorable scores when evaluated on perplexity, diversity, and control of story via discrete plan.

1 Introduction

Maintaining long-term narrative flow and consistency are important concerns when aiming to generate a plausible story (Porteous and Cavazza, 2009; Hou et al., 2019). Prior work on narrative text generation has focused on generating consistent stories via story outlines using keywords or key phrases (Xu et al., 2018; Yao et al., 2019), event-based representations (Riedl and Young, 2010; Martin et al., 2018; Fan et al., 2019), plot graphs (Li et al., 2013) or a sentence representing theme (Chen et al., 2019).

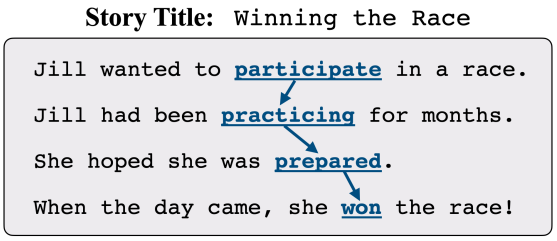

Figure 1: Our aim is to generate a story given a title. We propose models which first generate a high level story plan realized via a sequence of anchor words.

Yao et al. (2019) note that compared to specific event based representations, using keywords to form the outline is more generalizable and widely applicable. In this work, we consider a sequence of anchor words as a means to model story outlines. For example, in Figure 1, given a story title 'Winning the Race', our model first predicts a sequence of anchor words which represents a high level story plan. Thereafter, a decoder conditions on the title and generated sequence of anchor words to generate the final story. We assume an alignment between the anchor words and the story sentences – the i^{th} anchor word corresponds to the i^{th} sentence in the story.

However, stories do not naturally occur with a tagged set of such anchor words or keywords. Many prior works use off the shelf tools to first label stories with plan outlines, thus using external supervision for learning plot structures. For example, Yao et al. (2019) use the RAKE heuristic (Rose et al., 2010) to first identify the most important keyword in each sentence, and then use this to train a model in a supervised fashion. This approach leads to improved coherency and control, but creates a reliance on such heuristics and does not jointly learn anchor words along with the generator.

Findings of the Association for Computational Linguistics: EMNLP 2020, pages 3637–3650

Inspired by prior work indicating that anchor words can effectively capture and control high-level generation structure, we investigate to what extent high-level control can be learned in a fully unsupervised fashion, directly from natural story data. We design a hierarchical latent variable model (Figure 2) that induces sequences of anchor words that explain observed stories, while at the same time learning to generate entire stories by first generating anchor sequences. For training, we use amortized variational learning (Kingma and Welling, 2014), where an inference network is used to approximate the posterior on anchor sequences.

At test time, given a title, we first sample a sequence of anchor words using the prior model conditioned on only the title, and then generate the actual story using the decoder conditioning only on the title and the sampled anchor words.

To induce a useful latent generation plan and to effectively condition on a sampled plan, we propose a constrained story decoder and constrained inference network. Specifically, our constrained decoder begins a story sentence by deterministic *copying* the corresponding anchor word, and then generates words to the left and then to the right (Figure 3). For this decoder, the corresponding true posterior on anchor words is sparse: the anchor word must be chosen from the observed sentence. Thus, we constrain the output vocabulary of the corresponding inference network to the words of the input sentence. We observe that the proposed constrained inference network does not suffer from mode collapse, leading to models which can effectively learn useful anchor words. Further, we also contrast this approach with a model whose decoder is not constrained to use each anchor word in each sentence. The true posterior in this case is over the full vocabulary. We conduct experiments with both constrained and unconstrained decoders and inference networks, and find that the best results are achieved through the combination of an unconstrained decoder with a constrained inference network – indicating, perhaps, that while it is more effective to use flexible models, using a constrained inference network can add a useful inductive bias, leading the model to mimic the constraint of the inference network.

We experiment with two English story datasets, and observe that our best models achieve favorable scores relative to several baselines when evaluated on perplexity, diversity, coherency, and controllable story generation as per various automatic and human evaluations.

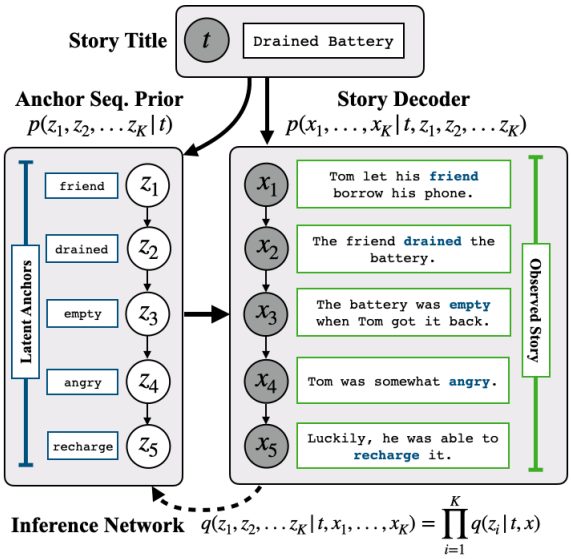

Figure 2: Model Overview: We consider multi-sentence text generation via a latent generation plan realized through a sequence of anchor words with one word per sentence. [We show sequence models with first-order Markov assumption for simplicity, even though all sequence models in our approach are auto-regressive with full context.]

Finally, we note that our modelling approach for story generation has an interesting connection with work that treats text as a latent variable in deep generative models (Miao and Blunsom, 2016; Wen et al., 2017). We treat a latent sequence of anchor words as a form of hierarchical control over generated outputs, while related work treats the latent sequence itself as sequential text that is the output of the model.

2 Model

Our goal is to generate a story x, consisting of multiple sentences $x_1, x_2, ..x_K$, given a title t. Our model's generative process is depicted in Figure 2 and operates as follows: First, a sequence of anchor words representing a generation plan is sampled from an auto-regressive prior conditioned on the title. Next, for each anchor word, a sentence is generated conditioned on the anchor words and previously generated sentences using a decoder. During training, the sequence of anchor words is unobserved and treated as a latent variable. As described in more detail later, we will explore several choices of decoder – those that treat anchor words as an explicit token in the sentence to be generated,

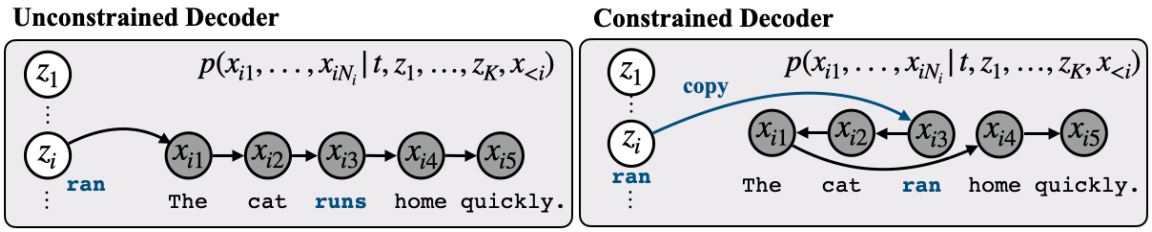

Figure 3: Simplified demonstration of generation of a sentence conditioned on anchor words and preceding sentences for the two types of decoders: (1) Unconstrained decoder is based on the story generation model of (Yao et al., 2019), which may or may not use the corresponding anchor word. (2) Constrained decoder is forced to use anchoring words in corresponding sentences, generating words to the left and then to the right of an anchor word. [Again, we show sequence models with a first-order Markov assumption for simplicity, even though all sequence models are auto-regressive with full context.]

generating surrounding context to the left and right, and those that simply treat the anchor words as conditioning information. In the former case, the posterior must be sparse. In the latter case, our choice of variational learning scheme will bias (but not force) the model to use anchor words in output story sentences. We shall refer to our proposed model as Latent Anchor Plan model (**LAP**).

2.1 Anchor Sequence Prior

We model the sequence of anchor words representing the generation plan via a sequence of discrete random variables $z_1, z_2, .., z_K$. Since our aim is to induce latent plans, we assume z are unobserved. We consider an auto-regressive prior model $p_\phi(z|t) = \prod_i p_\phi(z_i|z_{<i}, t)$ where each anchor word is conditioned on preceding anchor words and the title t.

2.2 Story Decoder

Our decoder $p_\theta(x|t, z)$ generates a story given the title t and anchor words z. As mentioned earlier, z_i is aligned to the sentence x_i. We consider two decoders: (1) an unconstrained decoder which is not bound to use z_i in x_i, and (2) a constrained decoder which assumes z_i is present in x_i, and constructs words to the left and then to the right of the anchor word z_i.

Unconstrained Decoder: Our unconstrained decoder is based on Yao et al. (2019)'s decoder which does not use any explicit alignment of anchor words to corresponding sentences (Figure 3). The decoder is fed the title and the anchor words appended together, and is trained to generate the multi-sentence text. The decoder is not bound to use the anchor word z_i for x_i, but may have incentive to do so depending on the training objective, as discussed later. At the same time, the unconstrained decoder has higher flexibility and can skip using an anchor word if it doesn't fit with the preceding context.

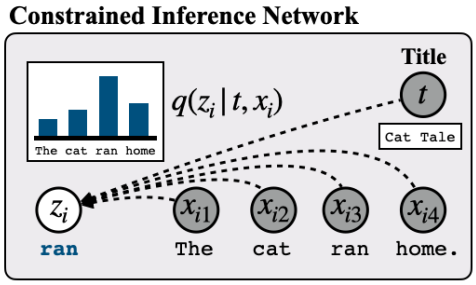

Figure 4: Constrained Inference Network: Proposed model is trained through amortized variational learning using an inference network. One of the proposed models is trained using a constrained inference network which assigns non-zero probability to only the words present in corresponding sentences.

Constrained Decoder: We consider a constrained decoder that always uses z_i while generating x_i. This is achieved by first copying z_i, then generating to the left until the sentence start, and then to the right. Such a decoder is bound to use the corresponding anchor word by design, and will potentially demonstrate higher control of the anchor words on the story.

Our decoder architecture follows from Yao et al. (2019), who use a 3-layer LSTM recurrent model. Our final reported model uses 1000 dimensional hidden layer, with tied input and output word embeddings. Moreover, the prior model shares the underlying LSTM modules with the decoder. Since our goal is to induce a latent discrete plan and compare with keyword tagging based methods, we stick to the same choice of decoder as in prior work.

3 Learning and Inference

Our goal is to maximize the log likelihood of the stories conditioned on the corresponding titles. Since z is unobserved at training, we must marginalize over all possible values of z.

$$\sum_{t,x\in\mathcal{D}} \log p(x|t) = \sum_{t,x\in\mathcal{D}} \log \mathbb{E}_{z\sim p_\phi(z|t)}[p_\theta(x|t,z)]$$

, $\mathcal{D}$ represents the dataset of titles and corresponding stories. Since it is infeasible to compute the exact marginal stated above, we use amortized variational learning by introducing an inference network q_γ, and train the model to maximize the following evidence lower-bound (ELBO):

$$\text{ELBO} = \underbrace{\mathbb{E}_{z\sim q_\gamma(z|x,t)}[\log p_\theta(x|z,t)]}_{\text{Reconstruction}} - \underbrace{\text{KL}(q_\gamma(z|x,t)||p_\phi(z|t))}_{\text{KL-term}}$$

We shall refer to the first term as the reconstruction term and the second term as the KL-term.

We make a mean-field assumption in the posterior approximation on z as follows: $q(z|x,t) = \prod_{i=1}^{K} q(z_i|x_i,t)$. Note that $p(z|t)$ is autoregressive, and thus it is intractable to exactly compute the KL term. We resort to Monte Carlo sampling to approximate the ELBO by drawing samples from inference network; though we will perform this differently for the KL term and the reconstruction term (more details in Section 3.2).

3.1 Inference Network and Posterior Sparsity

Constrained Inference Network With the constrained decoder discussed earlier, the true posterior is sparse – so making the inference net also sparse would help the learning procedure better approximate the true posterior (Figure 4). To leverage this observation, we constrain the inference network's output distribution to have non-zero probabilities only on the tokens present in the corresponding sentence:

$$q(z_i = v|x_i,t) = 0 \text{ if } v \notin x_i$$
$$\propto \exp(s_v) \text{ otherwise}$$

Here, s_v is the logit output for the token v produced by the inference network. Our constrained inference network is a BiLSTM model which generates an encoding h_j for j^{th} token in a story sentence. A linear layer transforms h_j to a score s_j. Finally, for sentence x_i, we compute a softmax over the scores of words in x_i to obtain $q(z_i|x)$.

Unconstrained Inference Network We also consider an unconstrained inference network which does not constrain the inference network's output – i.e. the output distribution is over the entire vocabulary. We use a LSTM model to encode each sentence, obtain the last word hidden state, and then finally employ a linear layer to transform it to the vocabulary size.

When the decoder is not constrained, it may be interesting to compare the choice of inference network. Using the constrained inference net with the unconstrained decoder will bias the decoder to use the anchor words in the aligned sentences – the model is not required to do this, but variational learning will pull the inference network and true model posterior towards each other (i.e. the ELBO objective pressures them to agree). Thus, if the inference net is constrained, but the decoder is not, learning will try to find a weakly constrained decoder to match the approximate posterior.

3.2 Optimization

Reconstruction term: As mentioned earlier, we draw samples from the inference network to approximate the reconstruction term. The decoder parameters θ can be trained directly through back-propagation to minimize the approximate reconstruction loss. However, since z is discrete, we use the REINFORCE (Williams, 1992) algorithm to train the parameters γ of the inference network $q(z|x,t)$. Following prior work (Xu et al., 2015), we use an entropy regularizer term and a moving average baseline to reduce the variance of the resulting gradient estimator for inference network parameters γ.

KL term: Note that the KL term can be simplified as follows:

$$\begin{aligned}\text{KL}(q_\gamma(z)||p_\phi(z)) = \text{KL}(q_\gamma(z_1)||p_\phi(z_1)) + \\ \mathbb{E}_{z_1\sim q_\gamma(z_1)}[\text{KL}(q_\gamma(z_2)||p_\phi(z_2|z_1)) + \\ \mathbb{E}_{z_2\sim q_\gamma(z_2)}[\text{KL}(q_\gamma(z_3)||p_\phi(z_3|z_{<3})] + \ldots]]]\end{aligned}$$

We draw samples of z from $q(z)$ to approximate the KL term.

KL term for the constrained inference network: For the constrained inference network, we have a

sparse approximate posterior. Given the fact that typical sentences in our dataset are 5-20 words in length, it is computationally easy to exactly compute individual $\mathrm{KL}(q(z_i)||p(z_i|z_{<i}))$ terms by summing over the tokens in x_i instead of the whole vocabulary. This is still an approximation to the full KL term since we cannot feasibly sum over the context.

$$\begin{aligned}\mathrm{KL}(q(z_i)||p(z_i|z_{<i}) &= \sum_{z_i \in V} q(z_i) \log q(z_i)/p(z_i) \\ &= \sum_{z_i \in \mathbf{x_i}} q(z_i) \log q(z_i)/p(z_i)\end{aligned}$$

Thus, for the constrained inference network, KL computation now proceeds as follows: we first compute $\mathrm{KL}(q(z_1)||p(z_1))$ as described above. Then we sample $z1 \sim q(z_1)$, and compute $\mathrm{KL}(q(z_2)||p(z_2|z_{<1}))$, and so on – we still need to use samples, but can exactly compute each of the K individual KL terms, one at each of the K steps in the plan, similar to the approach of (Yang et al., 2018). We observe that the constrained inference network leads to lower variance in the KL term approximation, thereby leading to more stable gradients.

Pretraining: Pretraining the inference network in an autoencoder setup has been found useful for VAE training (Li et al., 2019). We pretrain the inference network in an autoencoder setup where the decoder reconstructs the corresponding sentences (rather than whole story). Thereafter, we train the decoder and prior keeping the inference network fixed. Finally we perform the full training with all parameters being updated. We observe that pretraining through this procedure leads to more stable training.

4 Experiments

We evaluate and report generation quality of various models using automatic metrics for fluency and diversity, as well as human evaluations for coherence of story and relevance to title. We also analyze the ability of anchor words to control the generated story, and highlight comparisons between various choices of inference networks and decoders.

4.1 Dataset

We use a subset of the ROC-stories corpus (ROC-DATA) (Mostafazadeh et al., 2016) used earlier by Yao et al. (2019). Yao et al. (2019) had chosen a subset of the original ROC corpus in order to select only those stories which are accompanied by a title. The train, validation and test splits consist of 78529, 9816, and 9816 stories respectively. Most of the data consist of five sentence stories. Additionally, we experiment with the visual story dataset (only the text portion), which we discuss in more detail in Section 4.8.

4.2 Methods

NOPLAN-LM: This baseline does not consider any story generation plan and conditions only on the title. We use the same 3-layer LSTM as in the proposed model.

SUPERVPLAN: This baseline is based on the work of (Yao et al., 2019) which utilizes RAKE-tagged keywords as observed anchor words. The model is trained to predict the the observed anchor words and the story given the title. We can view this baseline as a latent variable model that was trained using RAKE keywords as the output of a deterministic inference network.

LAP: (1) We will refer to our model with the constrained inference network and unconstrained decoder as **LAP-CINF-UDEC**. (2) **LAP-UINF-UDEC** uses the unconstrained inference network and unconstrained decoder. (3) **LAP-CINF-CDEC** uses the constrained inference network with the constrained decoder. We found that the model with constrained decoder and unconstrained encoder performed poorly during training, and so we do not include it in experiments.

Decoding procedure: For all the methods, we generate samples with top-p sampling (Holtzman et al., 2020) with $p = 0.6$ at the time of story generation. Unless otherwise stated, the same decoding procedure is followed for the evaluations of diversity, story quality, and controllable generation discussed below. Later in the analysis we discuss the effect of changing the parameter p on some of the evaluation metrics.

4.3 Perplexity

For the models with latent generation plans, we use importance weighting (IW) (Burda et al., 2016) (with 20 samples) to estimate perplexity scores since (IW) has been shown to provide a tighter bound than ELBO for evaluation purposes (Li et al.,

Method	Inference N/W	Decoder	PPL↓	NLL↓		DIV↑		DIV-B↓
			test	test	dev	plan	story	story
No Plan								
ROC-DATA	NA	NA	NA	NA	NA	NA	9.01	0.23
NOPLAN-LM	NA	Unconstrained	**17.3**	154.0	160.7	NA	7.70	0.50
With Plan								
SUPERVPLAN	NA [1]	Unconstrained	≤28.3	≤180.3	≤187.6	8.71	7.74	0.49
LAP-CINF-UDEC	Constrained	Unconstrained	≤**21.3**	≤168.9	≤176.5	9.24	**7.93**	**0.45**
LAP other variants:								
LAP-CINF-CDEC	Constrained	Constrained	≤**20.9**	≤166.9	≤174.1	9.24	**7.98**	**0.44**
LAP-UINF-UDEC	Unconstrained	Unconstrained	≤17.5	≤154.2	≤160.9	0.01	7.67	0.52

Table 1: Automated metrics: We report Negative Log Likelihood (NLL), perplexity (PPL) (computed using importance weighted samples for models with latent variables), and diversity (DIV and DIV-B). LAP-CINF-UDEC performs better than SUPERVPLAN on perplexity as well as diversity. We also experiment with two other variants for LAP. LAP-UINF-UDEC, which does not constrain the inference network, suffers from posterior collapse. LAP-CINF-CDEC, which uses the constrained decoder, achieves perplexity and diversity results that are comparable to LAP-CINF-UDEC.

2019). For the baseline, SUPERVPLAN, we also evaluate its marginal likelihood for comparison with our model. To do this, we separately train an inference network (with the same architecture as that used by the LAP-CINF-UDEC model) to approximate the posterior on anchor words for the trained SUPERVPLAN (by keeping the trained model parameters fixed). This approximate posterior is then used to compute an upper bound on NLL and perplexity.

The proposed model LAP-CINF-UDEC performs better than the baseline SUPERVPLAN, which uses separately tagged generation plans (Table 1). However, the proposed method's perplexity is close to that of NOPLAN-LM, which does not consider any generation plan. This is not uncommon for deep latent variable models – since their held-out likelihood is intractable, and most approximations yield upper bounds on perplexity, their reported perplexity is always pessimistic. Among LAP variants, we observe that LAP-UINF-UDEC suffers from posterior collapses, and behaves similarly to NOPLAN-LM since the latent variables z are not informative or useful. Finally, LAP-CINF-CDEC performs similar on likelihood evaluations compared to the LAP-CINF-UDEC model with an unconstrained decoder .

4.4 Diversity

We generate story samples for all the titles in the test split. We employ two evaluations to report diversity in the generated outputs:

DIV We compute the geometric mean of empirical unigram, bigram, and trigram distribution entropy from the generated set of stories (Jhamtani et al., 2018). For methods which use generation plans, we also compute this diversity metric on anchor word sequences. Table 1 shows the results for various models. LAP-CINF-UDEC performs better than SUPERVPLAN, achieving higher diversity for both story and plans. Among the LAP variants, using the non-constrained inference network (LAP-UINF-UDEC) leads to worse results on story diversity, and fares poorly in plan diversity (due to posterior collapse). LAP-CINF-CDEC again performs similarly to LAP-CINF-UDEC.

DIV-B We also report inter-story BLEU4 scores (Zhu et al., 2018). We compute samples from various methods for 1000 titles in the test split. For each generated story, the remaining 999 are treated as references. Thus, lower values indicate higher diversity in the generated stories. Table 1 shows the results. LAP-CINF-UDEC performs better than SUPERVPLAN, though is still far from the values for human written stories in the ROC dataset itself.

4.5 Human Evaluations

We conduct human evaluations on Amazon Mechanical Turk to evaluate the quality of generated stories given the title. We evaluate the story samples with respect to: (1) coherence, which measures the logical and coherent narrative flow in a story, and (2) fidelity to title, which measures the degree to which the story is relevant to the given title. Given two stories from two different methods, we request human annotators to provide their preference (or mark as tie).

LAP-CINF-UDEC vs Method M	Coherence win-tie-loss	Title-Fidelity win-tie-loss
M=SUPERVPLAN	0.31 0.37 0.32	**0.39** 0.27 0.34
M=NOPLAN-LM	**0.36** 0.35 0.29 †	0.33 0.37 0.30
M=ROC-DATA	0.12 0.08 0.80 †	0.08 0.15 0.77 †

Table 2: Human preference evaluations when pitting various methods against LAP-CINF-UDEC (i.e. preference for LAP-CINF-UDEC is reported under *win*). Compared to SUPERVPLAN, LAP-CINF-UDEC performs better on fidelity to title and similar on coherence. Loss vs win judgements marked with † are statistically significant under bootstrap test ($p < 0.05$) considering 1000 subsets each of size 400.

Method	CTRL
SUPERVPLAN	38.8%
LAP-CINF-UDEC	**72.9**%
LAP variants:	
LAP-CINF-CDEC	100.0%
LAP-UINF-UDEC	0.0%

Table 3: We evaluate models for the extent to which the story follows the generation plan by evaluating the fraction of anchor words used in corresponding sentences (CTRL). LAP-CINF-UDEC demonstrates better control compared to SUPERVPLAN. Model with LAP-UINF-UDEC inference network collapses, while LAP-CINF-CDEC demonstrates perfect control due to the nature of the decoder.

In order to ensure the quality of human evaluations, we restrict the annotation task to annotators from Anglophone countries, and limited to workers with more than 90% HIT (Human Intelligence Task) acceptance rates. We randomize the order of presented stories to avoid positional bias effects. Additionally, we added two 'check' data points with each HIT. More specifically, to construct a 'check', we pick a random story from the development set, and then prepare a 'decoy' story by replacing three lines of the story with that of another randomly chosen story. The HITs where annotators marked the 'decoy' as the preferred story relative to the unaltered story with respect to either coherence or fidelity for either of the two check data points are skipped. These skipped HITs are then re-annotated.

Based on the automated metrics and manual qualitative inspection, we pick LAP-CINF-UDEC as the best configuration among all the variants of our model for human evaluation. We randomly selected 200 titles from the test split, generate samples from all the methods under consideration, and evaluate each method against LAP-CINF-UDEC. Each comparison is rated by three different annotators leading to a total of 600 judgements per pair. Table 2 shows the results. We observe that on average, annotators found LAP-CINF-UDEC outputs similar or better on coherence and fidelity compared to the baselines. LAP-CINF-UDEC is judged better than NOPLAN-LM on coherence, perhaps because having a plan provides a rough sketch of the story leading to more coherent outputs. Compared to SUPERVPLAN, outputs from the proposed method LAP-CINF-UDEC are judged similar in quality in terms of coherence but better in terms of fidelity to title, perhaps because the ELBO objective encourages the inference network to pick anchor words which can be more easily predicted from the title by the prior model, leading to better title fidelity. We show example generated samples from LAP-CINF-UDEC in Table 4. More examples and qualitative analysis can be found in the Appendix.

We found LAP-CINF-CDEC outputs to be slightly worse than LAP-CINF-UDEC and SUPERVPLAN outputs on coherency. Compared to LAP-CINF-UDEC, the constrained decoder achieves slightly better scores for perplexity and diversity (Table 1) and control (next subsection), but suffers on overall coherency. This behavior is likely due to the reduced flexibility of the model architecture (an example output is provided in Table 5). In contrast, the non-constrained decoder achieves a favorable balance between control and coherency. This highlights an interesting balance between the generation plan and the degree to which the decoder must follow the plan.

4.6 Controllable Generation

We evaluate models for the extent to which the story follows the generation plan. To evaluate this, we draw one story sample per title in the test split, and report the fraction of anchor words which were used in corresponding sentences (**CTRL**). LAP-CINF-UDEC (73%) fares much better than SUPERVPLAN (39%) (Table 3). We note that in some outputs from LAP-CINF-UDEC, even though the exact anchor word was not used, we observe semantically equivalent concepts being used – for example, for the sampled anchor word 'dismay', the generated story sentence was: 'She then realized she wasn't able to attempt it'.

We also analyze CTRL and DIV-B values when sampling with different values of parameter p in

[1]We retrofit an inference network to a trained SUPERVPLAN to approximate PPL and NLL for evaluation purposes only. Training the SUPERVPLAN model does not involve any inference network.

TITLE:	the exam
ANCHOR WORDS:	midterm knew nervous performed passed
STORY:	I had a big geometry exam today. I knew that i would have to do it. I was nervous. I had not performed since i was a little girl. I passed out.
TITLE:	the new bed
ANCHOR WORDS:	alex new store amazing glad
STORY:	Alex was trying to find a new bed. She needed a new one. She went to the store to get one. She found a amazing one. She was glad she bought it.
TITLE:	picnic
ANCHOR WORDS:	goes fancy least eating leave
STORY:	Last week i visited my friends to the park. It was at the fancy park. They had to eat the food and water. I had a great time eating. I had to leave.

Table 4: Generated samples from the proposed method LAP-CINF-UDEC. We observe that samples from the proposed method demonstrate fidelity to the title, better follow the sampled plan of anchor word sequences, and are in aggregate more coherent than baselines which do not consider a generation plan.

LAP-CINF-CDEC	**TITLE:**	the exam
	ANCHOR WORDS:	failing nervous tried test shocked
	STORY:	Jessica was failing her math class. She was nervous to try to take the test. She tried to help. She took the test. She was shocked and confident
LAP-UINF-UDEC	**TITLE:**	the new bed
	ANCHOR WORDS:	forms forms forms forms forms
	STORY:	Jane was about to get a new bed. She had been trying to catch a few new sheets. She decided to get a new bed. She looked at the new sheets. It was the right choice.

Table 5: Generated samples from LAP-CINF-CDEC and LAP-UINF-UDEC variants of the proposed model class. We observe that when using the constrained decoder variant, story outputs lack coherence more often than when using the unconstrained decoder, though they demonstrate better control by design. The LAP-UINF-UDEC variant suffers from posterior collapse, leading to a generic anchor word sequence, and often produces stories that lack overall structure.

p	LAP-CINF-UDEC		LAP-CINF-CDEC		SUPERVPLAN	
	CTRL	DIV-B	CTRL	DIV-B	CTRL	DIV-B
0.5	80%	0.48	100%	0.48	43%	0.54
0.6	73%	0.45	100%	0.44	39%	0.48
0.7	67%	0.41	100%	0.40	34%	0.43
0.8	59%	0.35	100%	0.34	29%	0.38

Table 6: Using higher p in top-p sampling leads to lower control of story via plan and higher diversity.

top-p sampling. As we increase p, we observe higher diversity in samples, along with lower scores for CTRL for LAP-CINF-UDEC as well as SUPERVPLAN (Table 6). This further shows an interesting trade-off between control and diversity.

4.7 Inference Network

The latent plan model with no constraint on the inference network, LAP-UINF-UDEC, suffers from severe mode collapse and essentially ignores the plan. This demonstrates that constraining the inference network was useful in mitigating the posterior collapse issue. In preliminary experiments, we also observed that using a bag-of-words inference network instead of the BiLSTM leads to worse performance on perplexity, diversity and control, which indicates that the learned posteriors for the BiLSTM network are in fact considering words in context rather than just identifying topical words in the vocabulary.

On analyzing the argmax outputs from the inference network of the trained LAP-CINF-UDEC model, we find that 42% of the predicted anchor words are nouns, 39% of them are verbs, and 11% are adjectives, compared to 58%, 33% and 6% respectively for the RAKE extracted keywords for the ROC data. Thus, the inference network learned along-with the LAP-CINF-UDEC model has higher preference for verbs and adjectives compared to the RAKE algorithm.

4.8 Visual Storytelling Dataset

We also conduct experiments with the text portion of a visual story dataset (Huang et al., 2016). The dataset consists of 40155, 4990, and 5055 stories in train, dev, and test splits. Compared to the ROC data, there are no titles associated with stories, and we learn unconditional anchor word sequence $p(z)$. We train the best model configuration LAP-CINF-UDEC (with constrained inference network and

Model	PPL↓		DIV↑	
	dev	test	plan	story
No Plan				
VIZSTORYDATA	NA	NA	NA	8.9
NOPLAN-LM	**38.5**	40.0	NA	6.3
With Plan				
SUPERVPLAN	≤41.5	≤42.2	6.5	6.5
LAP-CINF-UDEC	≤**39.9**	≤**40.8**	8.0	6.6

Table 7: Experiments with a second story dataset. We experiment with the text portion of the Visual Story Dataset. We observe that LAP-CINF-UDEC is able to perform better than SUPERVPLAN on perplexity and diversity.

unconstrained decoder). To train the baseline SUPERVPLAN, we run the RAKE algorithm to tag the data with the anchor words. We observe that LAP-CINF-UDEC performs better in terms of diversity of generated stories and plans, as well as perplexity relative to SUPERVPLAN (Table 7). Diversity computations are performed with 200 generated samples. We provide further example generations from various methods in the Appendix.

5 Related Work

Prior work on story generation has largely focused on plot outline via keywords or key phrases (Yao et al., 2019; Xu et al., 2018), event-based representations (Martin et al., 2018; Fan et al., 2019), or a sentence theme (Chen et al., 2019). Liu et al. (2020) propose a method to generate a story conditioned on a character description. Prior work on narrative text generation with plans has mostly relied on external resources or tools to extract outlines (Zhou et al., 2018; Fan et al., 2019), and then training in a supervised manner. For example, using VADER (Hutto and Gilbert, 2014) to tag sentiment polarity (Luo et al.).

Much prior work has used manually defined objectives to encourage coherence in generated text. In this context, reinforcement learning has been used to encourage stories to follow certain manually defined goals such as being locally coherent (Tambwekar et al., 2018; Xu et al., 2018). Prior work on visual story generation aim to learn topically coherent visual story generation (Huang et al., 2019; Wang et al., 2019). Compared to topics, keywords provide more fine-grained plan, and thus are more likely to provide fine-grained control over generated outputs.

In this work we have proposed a constrained inference network and a constrained decoder for story generation. Pointer networks (Vinyals et al., 2015) have been used for amortized inference in prior work on summarization (Miao and Blunsom, 2016), though in a semi-supervised context. Non-monotonic sequence generation has been explored in past for tasks such as machine translation (Welleck et al., 2019).

In the proposed model, the generation plan can be used to control the story via the anchor words. Hard and soft constraints for incorporating keywords into generation have been explored in Kiddon et al. (2016); Miao et al. (2019). Controllable text generation has been explored in other tasks as well, such as summarization (Fan et al., 2018), paraphrasing (Goyal and Durrett, 2020), style transfer (Keskar et al., 2019), and data-to-text generation (Shen et al., 2019).

6 Conclusion

In this work we have proposed a deep latent variable model which induces a discrete sequence of anchor words as a high-level plan to guide story generation.[2] We train the models though variational learning using a constrained inference network, and compare constrained and unconstrained versions of the decoder. The proposed model performs similarly or better than baselines on various automated and human evaluations. Related approaches might be used more broadly for a variety of language generation tasks, or even for related domains like music generation. Other modeling extensions might explore richer structure in latent plans – for example, generalizing beyond isolated words. Finally, in this work we trained decoders from scratch, though it would be interesting to explore the incorporation of large pretrained models such as GPT2 (Radford et al., 2018) to increase fluency.

Acknowledgements

We thank Nikita Duseja, Aashi Jain, Prakhar Gupta, Bodhisattwa Majumder, and the anonymous conference reviewers for providing valuable feedback. This project is funded in part by the NSF under grants 1618044 and 1936155, and by the NEH under grant HAA256044-17. The first author is supported in part by an Adobe Research Fellowship. Findings and observations do not necessarily

[2] https://github.com/harsh19/Latent-Anchor-Plan

reflect the views of funding agencies.

References

Yuri Burda, Roger B. Grosse, and Ruslan Salakhutdinov. 2016. Importance weighted autoencoders.

Gang Chen, Yang Liu, Huanbo Luan, Meng Zhang, Qun Liu, and Maosong Sun. 2019. Learning to predict explainable plots for neural story generation. *arXiv preprint arXiv:1912.02395*.

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *Proceedings of the 2nd Workshop on Neural Machine Translation and Generation, NMT@ACL 2018,*.

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *ACL*.

Tanya Goyal and Greg Durrett. 2020. Neural syntactic preordering for controlled paraphrase generation. In *Annual Meeting of the Association for Computational Linguistics*.

Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. 2020. The curious case of neural text degeneration. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net.

Chenglong Hou, Chensong Zhou, Kun Zhou, Jinan Sun, and Sisi Xuanyuanj. 2019. A survey of deep learning applied to story generation. In *International Conference on Smart Computing and Communication*. Springer.

Qiuyuan Huang, Zhe Gan, Asli Celikyilmaz, Dapeng Wu, Jianfeng Wang, and Xiaodong He. 2019. Hierarchically structured reinforcement learning for topically coherent visual story generation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33.

Ting-Hao (Kenneth) Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross B. Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual storytelling. In *NAACL 2016*.

Clayton J. Hutto and Eric Gilbert. 2014. VADER: A parsimonious rule-based model for sentiment analysis of social media text. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014*.

Harsh Jhamtani, Varun Gangal, Eduard Hovy, Graham Neubig, and Taylor Berg-Kirkpatrick. 2018. Learning to generate move-by-move commentary for chess games from large-scale social forum data. In *ACL 2018*.

Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. 2019. Ctrl: A conditional transformer language model for controllable generation. *arXiv preprint arXiv:1909.05858*.

Chloé Kiddon, Luke Zettlemoyer, and Yejin Choi. 2016. Globally coherent text generation with neural checklist models. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP*.

Diederik P. Kingma and Max Welling. 2014. Auto-encoding variational bayes.

Bohan Li, Junxian He, Graham Neubig, Taylor Berg-Kirkpatrick, and Yiming Yang. 2019. A surprisingly effective fix for deep latent variable modeling of text. In *EMNLP-IJCNLP*.

Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. Story generation with crowdsourced plot graphs. In *Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, July 14-18, 2013, Bellevue, Washington, USA*. AAAI Press.

Danyang Liu, Juntao Li, Meng-Hsuan Yu, Ziming Huang, Gongshen Liu, Dongyan Zhao, and Rui Yan. 2020. A character-centric neural model for automated story generation. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020*. AAAI Press.

Fuli Luo, Damai Dai, Pengcheng Yang, Tianyu Liu, Baobao Chang, Zhifang Sui, and Xu Sun. Learning to control the fine-grained sentiment for story ending generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*.

Lara J Martin, Prithviraj Ammanabrolu, Xinyu Wang, William Hancock, Shruti Singh, Brent Harrison, and Mark O Riedl. 2018. Event representations for automated story generation with deep neural nets. In *AAAI*.

Ning Miao, Hao Zhou, Lili Mou, Rui Yan, and Lei Li. 2019. Cgmh: Constrained sentence generation by metropolis-hastings sampling. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33.

Yishu Miao and Phil Blunsom. 2016. Language as a latent variable: Discrete generative models for sentence compression. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696*.

Tom Pelsmaeker and Wilker Aziz. 2020. Effective estimation of deep generative language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Online. Association for Computational Linguistics.

Julie Porteous and Marc Cavazza. 2009. Controlling narrative generation with planning trajectories: The role of constraints. In *Interactive Storytelling, Second Joint International Conference on Interactive Digital Storytelling, ICIDS*, Lecture Notes in Computer Science.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2018. Language models are unsupervised multitask learners.

Mark O. Riedl and Robert Michael Young. 2010. Narrative planning: Balancing plot and character. *J. Artif. Intell. Res.*, 39.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text mining: applications and theory*, 1.

Xiaoyu Shen, Jun Suzuki, Kentaro Inui, Hui Su, Dietrich Klakow, and Satoshi Sekine. 2019. Select and attend: Towards controllable content selection in text generation. In *EMNLP-IJCNLP*.

Pradyumna Tambwekar, Murtaza Dhuliawala, Lara J Martin, Animesh Mehta, Brent Harrison, and Mark O Riedl. 2018. Controllable neural story plot generation via reinforcement learning. *arXiv preprint arXiv:1809.10736*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in neural information processing systems*.

Ruize Wang, Zhongyu Wei, Piji Li, Haijun Shan, Ji Zhang, Qi Zhang, and Xuanjing Huang. 2019. Keep it consistent: Topic-aware storytelling from an image stream via iterative multi-agent communication. *arXiv preprint arXiv:1911.04192*.

Sean Welleck, Kianté Brantley, Hal Daumé III, and Kyunghyun Cho. 2019. Non-monotonic sequential text generation. In *Proceedings of the 36th International Conference on Machine Learning, ICML 2019*.

Tsung-Hsien Wen, Yishu Miao, Phil Blunsom, and Steve Young. 2017. Latent intention dialogue models. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4).

Jingjing Xu, Xuancheng Ren, Yi Zhang, Qi Zeng, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. In *EMNLP*.

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhudinov, Rich Zemel, and Yoshua Bengio. 2015. Show, attend and tell: Neural image caption generation with visual attention. In *International conference on machine learning*.

Zichao Yang, Zhiting Hu, Chris Dyer, Eric P Xing, and Taylor Berg-Kirkpatrick. 2018. Unsupervised text style transfer using language models as discriminators. In *Advances in Neural Information Processing Systems*.

Lili Yao, Nanyun Peng, Ralph Weischedel, Kevin Knight, Dongyan Zhao, and Rui Yan. 2019. Plan-and-write: Towards better automatic storytelling. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Deyu Zhou, Linsen Guo, and Yulan He. 2018. Neural storyline extraction model for storyline generation from news articles. In *NAACL-HLT*.

Yaoming Zhu, Sidi Lu, Lei Zheng, Jiaxian Guo, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Texygen: A benchmarking platform for text generation models. *CoRR*, abs/1802.01886.

APPENDIX

A. Additional Implementation Details

Additional Training Details: We found it useful to add certain regularizers. Following Yao et al. (2019), we add a temporal L2 penalty on successive hidden state representations of LSTM. Additionally, we block stopwords from being sampled from the posterior since we are more interested in inducing generation plans. We use NLTK's English stop-words list for this purpose. During model training (after pretraining inference network), we also use KL thresholding / free-bits (Pelsmaeker and Aziz, 2020) which thresholds each component of the KL term to help prevent posterior collapse.

Hyperparameters We perform model selection based on best dev split performance as per NLL. (In case of latent variable models, we use the upper bound on NLL). The final model and training configuration for LAP-CINF-UDEC is as follows: batch size of 20, temporal regularization weight of 1.0, smoothing factor for moving average baseline for reinforce reward is 0.1, dimension of hidden embedding is 1000, input and output token embeddings are tied. A summary of the decoder and inference network for the final configuration of LAP-CINF-UDEC model is shown in Figure 5.

Datasets: We use ROC data [3] splits from (Yao et al., 2019) [4]. We also used Visual Story Dataset [5]

B. Generated Samples and Qualitative Analysis

Some additional generated samples from various models are shown in Table 8. We note that LAP-CINF-UDEC plans often exhibits good control over the generated story. For example, samples S3 and S4 samples in Table 8 for the same title by-and-large follow the generated plan. We do observe a certain degree of repetition in some samples e.g in sample S2, the first and third sentences both discuss mowing the lawn.

Sample S6 further demonstrates the generation order for LAP-CINF-CDEC. Each sentence begins

```
Decoder:
  (lockdrop): LockedDropout()
  (idrop): Dropout(p=0.4)
  (hdrop): Dropout(p=0.25)
  (drop): Dropout(p=0.4)
  (token_encoder): Embedding(37905, 1000)
  (rnns): ModuleList(
    (0): WeightDrop(
      (module): LSTM(1000, 1000)
    )
    (1): WeightDrop(
      (module): LSTM(1000, 1000)
    )
    (2): WeightDrop(
      (module): LSTM(1000, 1000)
    )
  )
  (token_decoder): Linear(in_features=1000,
    out_features=37905, bias=True)

InferenceNW
    (token_encoder): Embedding(37905, 1000)
    (contextualizer): LSTM(1000, 1000,
      bidirectional=True)
    (scorer): Linear(in_features=2000,
      out_features=1, bias=True)
    (softmax): Softmax()
```

Figure 5: Summary of model architecture.

by copying the corresponding anchor word, generating words to the left and then to the right of it. Thus LAP-CINF-CDEC model commits to using the corresponding anchor words. In contrast LAP-CINF-UDEC has more flexibility in using the anchor words, and sometimes anchor words get dropped or get used in a paraphrased manner. e.g. for sample S2, the model did not use the word 'tired' in the last story sentence, though the second last sentence evokes similar understanding. However, this also means the control of the story via the plan is somewhat weaker. e.g. for S4, the story does not use the anchor word 'boiled'. Finally, compared to outputs from LAP-CINF-UDEC and LAP-CINF-CDEC, we observe that SUPERVPLAN outputs demonstrate much weaker control of the plan on the story, and generate plans less relevant to the title.

Table 9 shows samples from various models trained on (text part of) visual story data.

.

[3] https://cs.rochester.edu/nlp/rocstories/

[4] https://bitbucket.org/VioletPeng/language-model/src/master/

[5] http://visionandlanguage.net/VIST/

S.no.	Method	Title	Story
1	LAP-CINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	race participate high challenged sure won Jill wanted to participate in a race. She was in high school and had no idea how to race. Jill challenged her friend to a race. She wasn't sure if she would win. Ultimately she won the race.
2	LAP-CINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	lawn mowing lawn mowed every tired I was mowing my lawn one day. I was on the lawn by myself. I mowed the lawn. I worked on it for hours. Eventually I was able to go back to work.
3	LAP-CINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	the dryer load cold wash everything long A girl got a load of laundry. She put everything in the dryer. She wanted to wash it off. She did everything but the washing machine was not so good. She had to take a long time to clean it up.
4	LAP-CINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	the dryer cooking boiled dried put cooking There was a pot of beans in the microwave. The pot was piled on the table. They dried it and cooked it. They put the noodles in the oven. The kids were all cooking dinner.
5	LAP-CINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	lost phone bar opened someone left found Tim was at the bar. He opened the door to find a phone. It was of a person. Tim left for a couple of minutes. He took it home and it turned out that he lost it.
6	LAP-CINF-CDEC	**TITLE:** **ANCHOR WORDS:** **STORY:** **STORY (reordered):**	race entered win losing trained win entered tim <start> a local gym. win to wanted he <start>. losing up wound he <start> the race. trained tim <start> to get an extra. win to decided tim <start> the race. Tim entered a local gym. He wanted to win. He wound up losing the race. Tim trained to get an extra. Tim decided to win the race.
7	LAP-UINF-UDEC	**TITLE:** **ANCHOR WORDS:** **STORY:**	race forms forms forms forms forms Cathy and his friends decided to go to the fair. Cathy was worried about the show that she was having to be on the track team. Cathy was worried that the race would have to be over. Cathy was nervous to win by a point of winning the race. Cathy caught the race.
8	SUPERVPLAN	**TITLE:** **ANCHOR WORDS:** **STORY:**	race today good day race race I was the fastest runner on my high school track team. It was very hard to win, but my friend convinced me to do it. I practiced with the runners and practiced every day. I practiced every day after school. I can go to the race every day.
9	SUPERVPLAN	**TITLE:** **ANCHOR WORDS:** **STORY:**	lost phone package wanted found battery missed Tom was getting a new phone. He wanted to buy a new one. When he found it it was still a broken battery. Tom was sad because it was broken. Tom tried to save it but it wasn't enough.

Table 8: Samples from different methods (with minor edits such as captalization)

LAP-CINF-UDEC	**TITLE:**	n/a
	ANCHOR WORDS:	reunion together way together members
	STORY:	We got together to have a reunion. We went to a small town. We had a lot of fun. We took many pictures. Then we all got together to talk to some families.
LAP-CINF-UDEC	**TITLE:**	n/a
	ANCHOR WORDS:	wedding wedding people food beautiful
	STORY:	I went to the wedding last weekend. There were a ton of people at the wedding. There were a lot of people there. The food was delicious. The cake was beautiful .
NOPLAN-LM	**TITLE:**	n/a
	ANCHOR WORDS:	n/a
	STORY:	This is the first day of my vacation. I like to have a trip to location. I have never been to location. Here is my apartment. I've been hoping to be there. I'm so excited.
SUPERVPLAN	**TITLE:**	n/a
	ANCHOR WORDS:	big man none sat showed
	STORY:	The big day. And the man was playing. And the flowers were laid out. Then they sat on the floor. They were all very happy
SUPERVPLAN	**TITLE:**	n/a
	ANCHOR WORDS:	big wedding table party people
	STORY:	It was a big day for a wedding. The wedding party all gathered around the table. The tables were set and ready to be served. People arrived and chatted with each other. The table was set.

Table 9: Samples from different methods for visual story data