Strategies for Structuring Story Generation

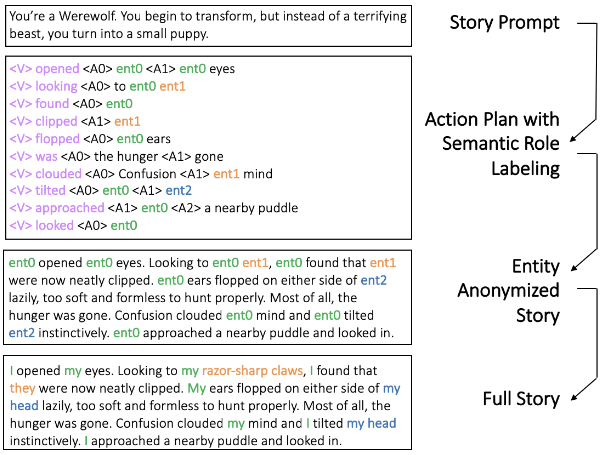
Angela Fan
FAIR, Paris
LORIA, Nancy
angelafan@fb.com

Mike Lewis
FAIR, Seattle
mikelewis@fb.com

Yann Dauphin
Google AI*
ynd@google.com

Abstract

Writers often rely on plans or sketches to write long stories, but most current language models generate word by word from left to right. We explore coarse-to-fine models for creating narrative texts of several hundred words, and introduce new models which decompose stories by abstracting over actions and entities. The model first generates the predicate-argument structure of the text, where different mentions of the same entity are marked with placeholder tokens. It then generates a surface realization of the predicate-argument structure, and finally replaces the entity placeholders with context-sensitive names and references. Human judges prefer the stories from our models to a wide range of previous approaches to hierarchical text generation. Extensive analysis shows that our methods can help improve the diversity and coherence of events and entities in generated stories.



You're a Werewolf. You begin to transform, but instead of a terrifying beast, you turn into a small puppy.

<V> opened <A0> ent0 <A1> ent0 eyes
<V> looking <A0> to ent0 ent1
<V> found <A0> ent0
<V> clipped <A1> ent1
<V> flopped <A0> ent0 ears
<V> was <A0> the hunger <A1> gone
<V> clouded <A0> Confusion <A1> ent1 mind
<V> tilted <A0> ent0 <A1> ent2
<V> approached <A1> ent0 <A2> a nearby puddle
<V> looked <A0> ent0

ent0 opened ent0 eyes. Looking to ent0 ent1, ent0 found that ent1 were now neatly clipped. ent0 ears flopped on either side of ent2 lazily, too soft and formless to hunt properly. Most of all, the hunger was gone. Confusion clouded ent0 mind and ent0 tilted ent2 instinctively. ent0 approached a nearby puddle and looked in.

I opened my eyes. Looking to my razor-sharp claws, I found that they were now neatly clipped. My ears flopped on either side of my head lazily, too soft and formless to hunt properly. Most of all, the hunger was gone. Confusion clouded my mind and I tilted my head instinctively. I approached a nearby puddle and looked in.

Figure 1: Proposed Model. Conditioned upon the prompt, we generate sequences of predicates and arguments. Then, a story is generated with placeholder entities such as *ent0*. Finally we replace the placeholders with specific references.

1 Introduction

Stories exhibit structure at multiple levels. While existing language models can generate stories with good local coherence, they struggle to coalesce individual phrases into coherent plots or even maintain character consistency throughout a story. One reason for this failure is that classical language models generate the whole story at the word level, which makes it difficult to capture the high-level interactions between the plot points.

To address this, we investigate novel decompositions of the story generation process that break down the problem into a series of easier coarse-to-fine generation problems. These decompositions can offer three advantages:

- They allow more abstract representations to be generated first, where challenging long-range dependencies may be more apparent.
- They allow specialized modelling techniques for the different stages, which exploit the structure of the specific sub-problem.
- They are applicable to any textual dataset and require no manual labelling.

Several hierarchical models for story generation have recently been proposed (Xu et al., 2018; Yao et al., 2019), but it is not well understood which properties characterize a good decomposition. We therefore implement and evaluate several representative approaches based on keyword extraction, sentence compression, and summarization.

We build on this understanding to devise the proposed decomposition (Figure 1). Our approach breaks down the generation process in three steps: modelling the action sequence, the story narrative, and lastly entities such as story characters. To model action sequences, we first generate the predicate-argument structure of the story by generating a sequence of verbs and arguments. This

*Work done while at Facebook AI Research representation is more structured than free text, making it easier for the model learn dependencies across events. To model entities, we initially generate a version of the story where different mentions of the same entity are replaced with placeholder tokens. Finally, we re-write these tokens into different references for the entity, based on both its previous mentions and global story context.

The models are trained on a large dataset of 300k stories, and we evaluate quality both in terms of human judgments and using automatic metrics. We find that our novel approach leads to much better story generation. Specifically, we show that generating the action sequence first makes the model less prone to generating generic events, leading to a much greater diversity of verbs. We also find that by using sub-word modelling for the entities, our model can produce novel names for locations and characters that are appropriate given the story context.

2 Model Overview

The crucial challenge of long story generation lies in maintaining coherence across a large number of generated sentences—in terms of both the logical flow of the story and the characters and entities. While there has been much recent progress in left-to-right text generation, particularly using self-attentive architectures (Dai et al., 2018; Liu et al., 2018), we find that models still struggle to maintain coherence to produce interesting stories on par with human writing. We therefore introduce strategies to decompose neural story generation into coarse-to-fine steps to make modelling high-level dependencies easier to learn.

2.1 Tractable Decompositions

In general, we can decompose the generation process by converting a story x into a more abstract representation z. The negative log likelihood of the decomposed problem is given by

$$\mathcal{L} = -\log \sum_{z} p(x|z)p(z). \tag{1}$$

We can generate from this model by first sampling from $p(z)$ and then sampling from $p(x|z)$. However, the marginalization over z is in general intractable, except in special cases where every x can only be generated by a single z (for example, if the transformation removed all occurrences of certain tokens). Instead, we minimize a variational upper bound of the loss by constructing a deterministic posterior $q(z|x) = 1_{z=z^*}$, where z^* can be given by running semantic role labeller or coreference resolution system on x. Put together, we optimize the following loss:

$$z^* = \arg\max_{z} p(z|x) \tag{2}$$

$$\mathcal{L} \leq -\log p(x|z^*) - \log p(z^*) \tag{3}$$

This approach allows models $p(z^*)$ and $p(x|z^*)$ to be trained tractably and separately.

2.2 Model Architectures

We build upon the convolutional sequence-to-sequence architecture (Gehring et al., 2017). Deep convolutional networks are used as both the encoder and decoder. The networks are connected with an attention module (Bahdanau et al., 2015) that performs a weighted sum of the encoder output. The decoder uses a gated multi-head self-attention mechanism (Vaswani et al., 2017; Fan et al., 2018) to allow the model to refer to previously generated words and improve the ability to model long-range context.

3 Modelling Action Sequences

To decompose a story into a structured form that emphasizes logical sequences of actions, we use Semantic Role Labeling (SRL). SRL identifies *predicates* and *arguments* in sentences, and assigns each argument a *semantic role*. This representation abstracts over different ways of expressing the same semantic content. For example, *John ate the cake* and *the cake that John ate* would receive identical semantic representations.

Conditioned upon the prompt, we generate an SRL decomposition of the story by concatenating the predicates and arguments identified by a pretrained model (He et al., 2017; Tan et al., 2018)[1] and separating sentences with delimiter tokens. We place the predicate verb first, followed by its arguments in canonical order. To focus on the main narrative, we retain only core arguments.

Verb Attention Mechanism SRL parses are more structured than free text, allowing scope for more structured models. To encourage the

[1] for predicate identification, we use https://github.com/luheng/deep_srl, for SRL given predicates, we use https://github.com/XMUNLP/Tagger

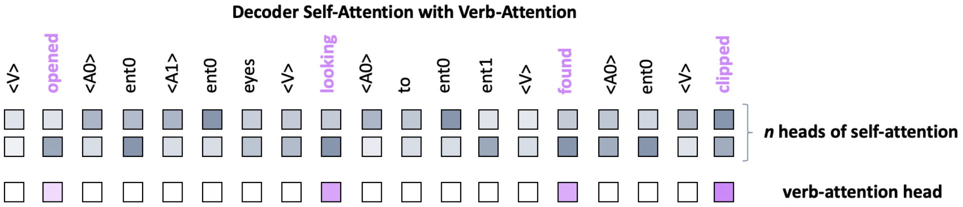


Figure 2: Verb-Attention. To improve the model's ability to condition upon past verbs, one head of the decoder's self-attention mechanism is specialized to only attend to previously generated verbs.

model to consider sequences of verbs, we designate one of the heads of the decoder's multihead self-attention to be a *verb-attention* head (see Figure 2). By masking the self-attention appropriately, this verb-attention head can only attend to previously generated verbs. When the text does not yet have a verb, the model attends to a zero vector. We show that focusing on verbs with a specific attention head generates a more diverse array of verbs and reduces repetition in generation.

4 Modelling Entities

The challenges of modelling characters throughout a story is twofold: first, entities such as character names are rare tokens, which make them hard to model for neural language models. Human stories often feature imaginative, novel character or location names. Second, maintaining the consistency of a specific set of characters is difficult, as the same entity may be referenced by many different strings throughout a story—for example *Bilbo Baggins*, *he*, and *the hobbit* may refer to the same entity. It is challenging for existing language models to track which words refer to which entity purely using a language modelling objective.

We address both problems by first generating a form of the story with different mentions of the same entity replaced by a placeholder token (e.g. *ent0*), similar to Hermann et al. (2015). We then use a sub-word seq2seq model trained to replace each mention with a reference, based on its context. The sub-word model is better equipped to model rare words and the placeholder tokens make maintaining consistency easier.

4.1 Generating Entity Anonymized Stories

We explore two approaches to identifying and clustering entities:

- **NER Entity Anonymization**: We use a named entity recognition (NER) model[2] to identify all people, organizations, and locations. We replace these spans with placeholder tokens (e.g. *ent0*). If any two entity mentions have an identical string, we replace them with the same placeholder. For example, all mentions of *Bilbo Baggins* will be abstracted to the same entity token, but *Bilbo* would be a separate abstract entity.
- **Coreference-based Entity Anonymization**: The above approach cannot detect different mentions of an entity that use different strings. Instead, we use the Coreference Resolution model from Lee et al. (2018)[3] to identify clusters of mentions. All spans in the same cluster are then replaced with the same entity placeholder string. Coreference models do not detect singleton mentions, so we also replace non-coreferent named entities with unique placeholders.

4.2 Generating Entity References in a Story

We train models to replace placeholder entity mentions with the correct surface form, for both NER-based and coreference-based entity anonymised stories. Both our models use a seq2seq architecture that generates an entity reference based on its placeholder and the story. To better model the specific challenges of entity generation, we also make use of a pointer mechanism and sub-word modelling.

Pointer Mechanism Generating multiple consistent mentions of rare entity names is challenging. To make it easier for the model to re-use previous names for an entity, we augment the standard seq2seq decoder with a pointer-copy mechanism (Vinyals et al., 2015). To generate an entity reference, the decoder can either generate a new abstract entity token or choose to copy an already generated abstract entity token, which encourages

[2] Specifically, Spacy: https://spacy.io/api/entityrecognizer, `en_core_web_lg`

[3] https://github.com/kentonl/e2e-coref

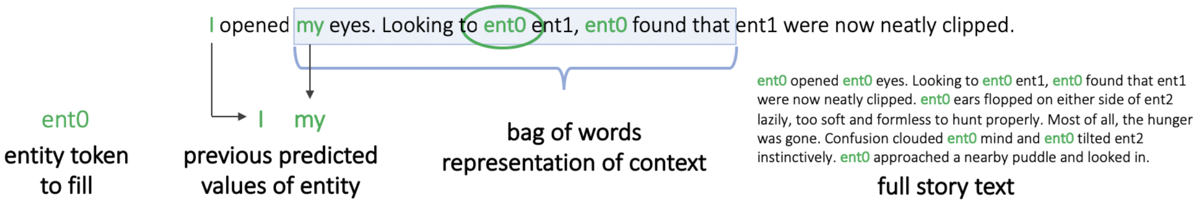

I opened my eyes. Looking to ent0 ent1, ent0 found that ent1 were now neatly clipped.
ent0
entity token to fill
I my
previous predicted values of entity
bag of words representation of context
ent0 opened ent0 eyes. Looking to ent0 ent1, ent0 found that ent1 were now neatly clipped. ent0 ears flopped on either side of ent2 lazily, too soft and formless to hunt properly. Most of all, the hunger was gone. Confusion clouded ent0 mind and ent0 tilted ent2 instinctively. ent0 approached a nearby puddle and looked in.
full story text

Figure 3: Input for Coreferent entity reference generation. The model has a representation of the entity context in a bag of words form, all previous predicted values for the same anonymized entity token, and the full text story. The green circle represents the entity mention the model is attempting to fill.

the model to use consistent naming for the entities.

To train the pointer mechanism, the final hidden state of the model h is used as input to a classifier $p_{\text{copy}}(h) = \sigma(w_{\text{copy}} \cdot h)$. w_{copy} is a fixed dimension parameter vector. When the model classifier predicts to copy, the previously decoded abstract entity token with the maximum attention value is copied. One head of the decoder multi-head self-attention mechanism is used as the pointer copy attention head, to allow the heads to specialize.

Sub-word Modelling Entities are often rare or novel words, so word-based vocabularies can be inadequate. We compare entity generation using word-based, byte-pair encoding (BPE) (Sennrich et al., 2015), and character-level models.

NER-based Entity Reference Generation Here, each placeholder string should map onto one (possibly multiword) surface form—e.g. all occurrences of the placeholder *ent0* should map only a single string, such as *Bilbo Baggins*. We train a simple model that maps a combination placeholder token and story (with anonymized entities) to the surface form of the placeholder. While the placeholder can appear multiple times, we only make one prediction for each placeholder as they all correspond to the same string.

Coreference-based Entity Reference Generation Generating entities based on coreference clusters is more challenging than for our NER entity clusters, because different mentions of the same entity may use different surface forms. We generate a separate reference for each mention by adding the following inputs to the above model:

- A *bag-of-words* context window around the specific entity mention, which allows local context to determine if an entity should be a name, pronoun or nominal reference.
- *Previously generated* references for the same entity placeholder. For example, if the model is filling in the third instance of *ent0*, it receives that the previous two generations for *ent0* were *Bilbo, him*. Providing the previous entities allows the model to maintain greater consistency between generations.

5 Experimental Setup

5.1 Data

We use the WRITINGPROMPTS dataset from (Fan et al., 2018) [4] of 300k story premises paired with long stories. Stories are on average 734 words, making the generation far longer compared to related work on storyline generation. In this work, we focus on the prompt to story generation aspect of this task. We assume models receive a human-written prompt, as shown in Figure 1. We follow the previous preprocessing of limiting stories to 1000 words and fixing the vocabulary size to 19,025 for prompts and 104,960 for stories.

5.2 Baselines

We compare our results to the *Fusion* model from Fan et al. (2018) which generates the full story directly from the prompt. We also implement various decomposition strategies as baselines:

- *Summarization*: We propose a new baseline that generates a summary conditioned upon the prompt and then a story conditioned upon the summary. Story summaries are obtained with a multi-sentence summarization model (Wu et al., 2019) trained on the full-text version of the CNN-Dailymail summarization corpus (Hermann et al., 2015; Nallapati et al., 2016; See et al., 2017)[5] and applied to stories.
- *Keyword Extraction*: We generate a series of keywords conditioned upon the prompt and

[4] https://github.com/pytorch/fairseq/tree/master/examples/stories

[5] https://github.com/abisee/cnn-dailymail

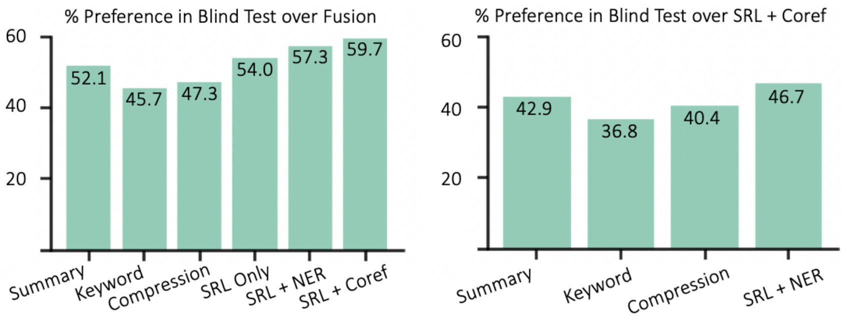


Figure 4: Human evaluations of different decomposed models for story generation. We find that using SRL action plans and coreference-resolution to build entity clusters generates stories that are preferred by human judges.

Decomposition	**Stage 1** $-\log p(z^*)$	**Stage 2** $-\log p(x\|z^*)$
Summary	4.20	5.09
Keyword	6.92	4.23
Compression	5.05	3.64
SRL Action Plan	2.72	3.95
NER Entity Anonymization	3.32	4.75
Coreference Anonymization	3.15	4.55

Table 1: Negative log likelihood of generating stories using different decompositions (lower is easier for the model). Stage 1 is the generation of the intermediate representation z^*, and Stage 2 is the generation of the story x conditioned upon z^*. Entity generation is with a word-based vocabulary to be consistent with the other models.

then a story conditioned upon the keywords, based on Yao et al. (2019). Following Yao et al, we extract keywords with the RAKE algorithm (Rose et al., 2010)[6]. Yao et al. extract one word per sentence, but we find that extracting $n = 10$ keyword phrases per story worked well, as our stories are much longer.

- *Sentence Compression*: Inspired by Xu et al. (2018), we generate a story with compressed sentences conditioned upon the prompt and then a story conditioned upon the compression. We use the same deletion-based compression data as Xu et al., from Filippova and Altun (2013)[7]. We train a seq2seq model to compress all non-dialog story sentences (as the training data does not contain much spoken dialogue). The compressed sentences are concatenated to form the compressed story.

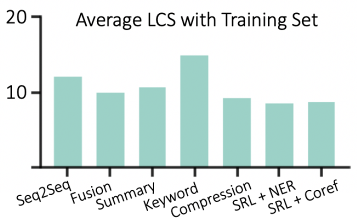


Figure 5: Average Longest Common Subsequence of Generated Stories with human-written stories in the training set.

5.3 Training

We implement models using `fairseq-py` (Ott et al., 2019)[8] in PyTorch and train Fan et al. (2018)'s convolutional architecture. We tune all hyperparameters on validation data.

5.4 Generation

We suppress the generation of unknown tokens to ease human evaluation. For all evaluations, we require stories to be at least 150 words and cut off the story at the nearest sentence for stories longer than 250 words. We generate stories with temperature 0.8 and random top-k sampling method proposed in (Fan et al., 2018), where next words are sampled from the top k candidates rather than the entire vocabulary distribution. We set $k = 10$.

6 Experiments

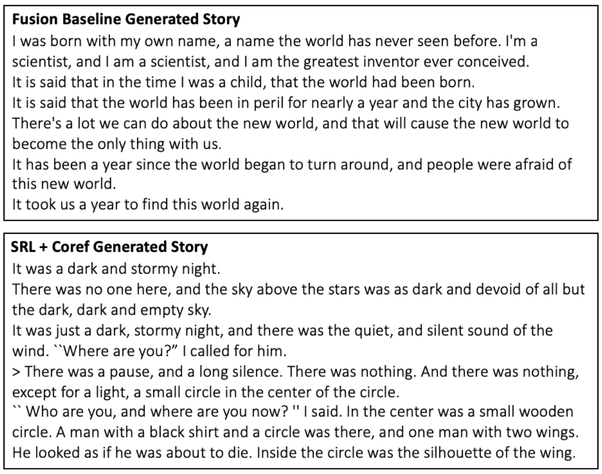
6.1 Comparing Decomposition Strategies

Automated Evaluation We compare the relative difficulty of modelling using each decomposition strategy by measuring the log loss of the different stages in Table 1. We observe that generating the SRL structure has a lower negative log-likelihood and so is much easier than generating

[6] `https://pypi.org/project/rake-nltk/`

[7] `https://github.com/google-research-datasets/sentence-compression`

[8] `https://github.com/pytorch/fairseq/`



Fusion Baseline Generated Story

I was born with my own name, a name the world has never seen before. I'm a scientist, and I am a scientist, and I am the greatest inventor ever conceived.

It is said that in the time I was a child, that the world had been born.

It is said that the world has been in peril for nearly a year and the city has grown.

There's a lot we can do about the new world, and that will cause the new world to become the only thing with us.

It has been a year since the world began to turn around, and people were afraid of this new world.

It took us a year to find this world again.

SRL + Coref Generated Story

It was a dark and stormy night.

There was no one here, and the sky above the stars was as dark and devoid of all but the dark, dark and empty sky.

It was just a dark, stormy night, and there was the quiet, and silent sound of the wind. ``Where are you?” I called for him.

> There was a pause, and a long silence. There was nothing. And there was nothing, except for a light, a small circle in the center of the circle.

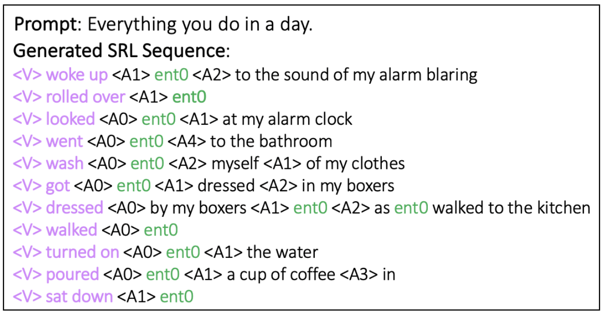
`` Who are you, and where are you now? '' I said. In the center was a small wooden circle. A man with a black shirt and a circle was there, and one man with two wings. He looked as if he was about to die. Inside the circle was the silhouette of the wing.

Figure 6: Our decomposition can generate more coherent stories than previous work.

either summaries, keywords, or compressed sentences — a benefit of its more structured form. We find keyword generation is especially difficult as the identified keywords are often the more salient, rare words appearing in the story, which are challenging for neural seq2seq models. This suggests that rare words should appear mostly at the last levels of the decomposition. Further, we compare models with entity-anonymized stories as an intermediate representation, either with NER-based or coreference-based entity anonymization. Entity references are then filled using a word-based model[9]. The entity fill is the more difficult stage.

Finally, we analyze the similarity of the generated stories with the stories in the training set. We quantify this by measuring the maximum and average longest common subsequence of tokens of a generated story with all human-written stories from the training set. High LCS values would indicate models are copying large subparts from existing stories rather than creatively writing new stories. Results shown in Figure **??** indicate that our proposed decomposition copies slightly less long sequences from the training set compared to the baselines — by separating verb and entity generation into distinct parts, we generate fewer long sequences already present in the training set.

Human Evaluation To compare overall story quality using various decomposition strategies, we conduct human evaluation using a crowdworking platform. Judges are shown two different stories that were generated based on the same human-written prompt (but do not see the prompt). Evaluators are asked to mark which story they prefer. 100 stories are evaluated for each model by 3 different judges. To reduce variance, stories from all models are trimmed to 200 words.



Prompt: Everything you do in a day.

Generated SRL Sequence:

<V> woke up <A1> ent0 <A2> to the sound of my alarm blaring

<V> rolled over <A1> ent0

<V> looked <A0> ent0 <A1> at my alarm clock

<V> went <A0> ent0 <A4> to the bathroom

<V> wash <A0> ent0 <A2> myself <A1> of my clothes

<V> got <A0> ent0 <A1> dressed <A2> in my boxers

<V> dressed <A0> by my boxers <A1> ent0 <A2> as ent0 walked to the kitchen

<V> walked <A0> ent0

<V> turned on <A0> ent0 <A1> the water

<V> poured <A0> ent0 <A1> a cup of coffee <A3> in

<V> sat down <A1> ent0

Figure 7: Example *generated* action plan for the SRL + NER Entity Anonymization model. It shows a plausible sequence of actions for a character.

Figure 6 shows that human evaluators prefer our novel decompositions over a carefully tuned Fusion model from Fan et al. (2018) by about 60% in a blind comparison. We see additive gains from modelling actions and entities. In a second study, evaluators compared various baselines against stories generated by our strongest model, which uses SRL-based action plans and coreference-based entity anonymization. In all cases, our full decomposition is preferred.

6.2 Effect of SRL Decomposition

Human-written stories feature a wide variety of events, while neural models are plagued by generic generations and repetition. Table 2 quantifies model performance on two metrics to assess action diversity: (1) the number of unique verbs generated, averaged across all stories (2) the percentage of diverse verbs, measured by the percent of all verbs generated in the test set that are not one of the top 5 most frequent verbs. A higher percentage indicates more diverse events.[10]

Our decomposition using the SRL predicate-argument structure improves the model's ability to generate diverse verbs. Adding verb attention leads to further improvement. Qualitatively, the model can often outline clear action sequences, as shown in Figure 7. However, all models remain far from matching the diversity of human stories.

[9] To make likelihoods are comparable across models.

[10] We identify verbs using Spacy: https://spacy.io/

Model	# Unique Verbs	% Diverse Verbs
Human Stories	34.0	76.5
Fusion	10.3	61.1
Summary	12.4	60.6
Keyword	9.1	58.2
Compression	10.3	54.3
SRL	14.4	62.5
+ verb-attention	15.9	64.9

Table 2: Action Generation. Generating the SRL structure improves verb diversity and reduces repetition.

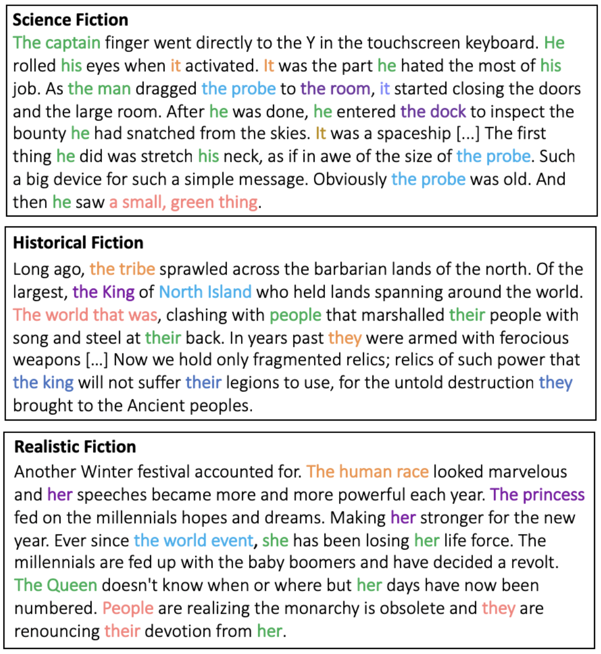
6.3 Comparing Entity Reference Models

We explored a variety of different ways to generate the full text of abstracted entities—using different amounts of context and different granularities of subword generation. To compare these models, we calculated their accuracy at predicting the correct reference in Table 3. Each model evaluates $n = 10, 50, 100$ different entities in the test set, 1 real and $n-1$ randomly sampled distractors. Models must give the true mention the highest likelihood. We analyze accuracy on the first mention of an entity, an assessment of novelty, and subsequent references, an assessment of consistency.

Effect of Sub-word Modelling Table 3 shows that modelling a character-level vocabulary for entity generation outperforms BPE and word-based models, because of the diversity of entity names. This result highlights a key advantage of multi-stage modelling: the usage of specialized modelling techniques for each sub-task.

Effect of Additional Context Entity references should be contextual. Firstly, names must be appropriate for the story setting—*Bilbo Baggins* might be more appropriate for a fantasy novel. Subsequent references to the character may be briefer, depending on context—for example, he is more likely to be referred to as *he* or *Bilbo* than his full name in the next sentence.

We compare three models ability to fill entities based on context (using coreference-anonymization): a model that does not receive the story, a model that uses only leftward context (as in Clark et al. (2018)), and a model with access to the full story. We show in Table 3 that having access to the full story provides the best performance. Having no access to any of the story decreases ranking accuracy, even though the model still receives the local context window of the entity as input. The left story context model performs better, but looking at the complete story provides additional gains. We note that full-story context can only be provided in a multi-stage generation approach.



Science Fiction

The captain finger went directly to the Y in the touchscreen keyboard. He rolled his eyes when it activated. It was the part he hated the most of his job. As the man dragged the probe to the room, it started closing the doors and the large room. After he was done, he entered the dock to inspect the bounty he had snatched from the skies. It was a spaceship [...] The first thing he did was stretch his neck, as if in awe of the size of the probe. Such a big device for such a simple message. Obviously the probe was old. And then he saw a small, green thing.

Historical Fiction

Long ago, the tribe sprawled across the barbarian lands of the north. Of the largest, the King of North Island who held lands spanning around the world. The world that was, clashing with people that marshalled their people with song and steel at their back. In years past they were armed with ferocious weapons [...] Now we hold only fragmented relics; relics of such power that the king will not suffer their legions to use, for the untold destruction they brought to the Ancient peoples.

Realistic Fiction

Another Winter festival accounted for. The human race looked marvelous and her speeches became more and more powerful each year. The princess fed on the millennials hopes and dreams. Making her stronger for the new year. Ever since the world event, she has been losing her life force. The millennials are fed up with the baby boomers and have decided a revolt. The Queen doesn't know when or where but her days have now been numbered. People are realizing the monarchy is obsolete and they are renouncing their devotion from her.

Figure 8: Generating entity references for different genres, using entity-anonymized human written stories. Models use the story context to fill in relevant entities. Color indicates coreferent clusters.

Qualitative Examples Figure 8 shows examples of entity naming in three stories of different genres. We evaluate different genres to examine if generated entities adapt to the style of the story. We show that models can adapt to the context—for example generating *The princess* and *The Queen* when the context includes *monarchy*.

6.4 Effect of Entity Anonymization

To understand the effectiveness of the entity generation models, we examine their performance by analyzing generation diversity.

Diversity of Entity Names Human-written stories often contain many diverse, novel names for people and places. However, these tokens are rare and subsequently difficult for standard neural models to generate. Table 4 shows that the fusion model and baseline decomposition strategies generate very few unique entities in each story. Generated entities are often generic names such as *John*.

Our proposed decompositions generate substantially more unique entities than strong baselines.

	First Mentions			Subsequent Mentions		
Model	**Rank 10**	**Rank 50**	**Rank 100**	**Rank 10**	**Rank 50**	**Rank 100**
Word-Based	42.3	25.4	17.2	48.1	38.4	28.8
BPE	48.1	20.3	25.5	52.5	50.7	48.8
Character-level	64.2	51.0	35.6	66.1	55.0	51.2
No story	50.3	40.0	26.7	54.7	51.3	30.4
Left story context	59.1	49.6	33.3	62.9	53.2	49.4
Full story	64.2	51.0	35.6	66.1	55.0	51.2

Table 3: Accuracy at choosing the correct reference string for a mention, discriminating against 10, 50 and 100 random distractors. We break out results for the first mention of an entity (requiring novelty to produce an appropriate name in the context) and subsequent references (typically pronouns, nominal references, or shorter forms of names). We compare the effect of sub-word modelling and providing longer contexts.

Model	**# Unique Entities**
Human Stories	2.99
Fusion	0.47
Summary	0.67
Keyword	0.81
Compression	0.21
SRL + NER Entity Anonymization	2.16
SRL + Coreference Anonymization	1.59

Table 4: Diversity of entity names. Baseline models generate few unique entities per story. Our decompositions generate more, but still fewer than human stories. Using coreference resolution to build entity clusters reduces diversity here—partly due to re-using existing names more, and partly due to greater use of pronouns.

Model	**# Coref Chains**	**Unique Names per Chain**
Human Stories	4.77	3.41
Fusion	2.89	2.42
Summary	3.37	2.08
Keyword	2.34	1.65
Compression	2.84	2.09
SRL + NER Entity Anonymization	4.09	2.49
SRL + Coreference Anonymization	4.27	3.15

Table 5: Analysis of non-singleton coreference clusters. Baseline models generate very few different coreference chains, and repetitive mentions within clusters. Our models generate larger and more diverse clusters.

Interestingly, we found that using coreference resolution for entity anonymization led to fewer unique entity names than generating the names independently. This result can be explained by the coreference-based model re-using previous names more frequently, as well as using more pronouns.

Coherence of Entity Clusters Well structured stories will refer back to previously mentioned characters and events in a consistent manner. To evaluate if the generated stories have these characteristics, we examine the coreference properties in Table 5. We quantify the average number of coreference clusters and the diversity of entities within each cluster (e.g. the cluster *Bilbo, he, the hobbit* is more diverse than the cluster *he, he, he*).

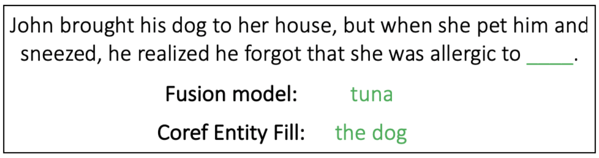
Our full model produces more non-singleton coreference chains, suggesting greater coherence, and also gives different mentions of the same entity more diverse names. However, both numbers are still lower than for human generated stories, indicating potential for future work.

Qualitative Example Figure 9 displays a sentence constructed to require the generation of an entity as the final word. The fusion model does not perform any implicit coreference to associate the *allergy* with *his dog*. In contrast, coreference entity fill produces a high quality completion.

7 Related Work

Decomposing natural language generation into several steps has been extensively explored (Reiter and Dale, 2000; Gatt and Krahmer, 2018). In classical approaches to text generation, various stages were used to produce final written text. For example, algorithms were developed to determine content and discourse at an abstract level, then sentence aggregation and lexicalization, and finally steps to resolve referring expressions (Hovy, 1990; Dalianis and Hovy, 1993; Wahlster et al., 1993; Ratnaparkhi, 2000; Malouf, 2000). Our work builds upon these approaches.

Story Generation with Planning Story generation using a plan has been explored using many different techniques. Traditional approaches organized sequences of character actions with hand crafted models (Riedl and Young, 2010; Porteous and Cavazza, 2009). Recent work extended this to modelling story events (Martin et al., 2017; Mostafazadeh et al., 2016), plot graphs (Li et al., 2013), plot summaries (Appling and Riedl, 2009),



John brought his dog to her house, but when she pet him and sneezed, he realized he forgot that she was allergic to ____.

Fusion model: tuna

Coref Entity Fill: the dog

Figure 9: Constructed sentence where the last word refers to an entity. The coreference model is able to track the entities, whereas the fusion model relies heavily on local context to generate the next words.

story fragments or vignettes (Riedl, 2010), or used sequences of images (Huang et al., 2016) or descriptions (Jain et al., 2017).

We build on previous work that decomposes generation. Xu et al. (2018) learn a skeleton extraction model and a generative model conditioned upon the skeleton, using reinforcement learning to train jointly. Zhou et al. (2018) train a storyline extraction model for news articles, but require supervision from manually annotated storylines. Yao et al. (2019) use RAKE (Rose et al., 2010) to extract storylines, and condition upon the storyline to write the story using dynamic and static schemas that govern if the storyline can change.

Entity Language Models An outstanding challenge in text generation is modelling and tracking entities. Centering (Grosz et al., 1995) gives a theoretical account of how referring expressions for entities are chosen in discourse context. Named entity recognition has been incorporated into language models since at least Gotoh et al. (1999), and can improve domain adaptation (Liu and Liu, 2007). Language models have been extended to model entities based on information such as entity type (Parvez et al., 2018). Recent work has incorporated learning representations of entities and other unknown words (Kobayashi et al., 2017), as well as explicitly model entities by dynamically updating these representations to track changes over time and context (Ji et al., 2017). Dynamic updates to entity representations are used in other story generation models (Clark et al., 2018).

Non-Autoregressive Generation Our method proposes decomposing left-to-right generation into multiple steps. Recent work has explored non-autoregressive generation for more efficient language modeling and machine translation. Ford et al. (2018) developed two-pass language models, generating templates then filling in words. The partially filled templates could be seen as an intermediary representation similar to generating a compressed story. Other models allow arbitrary order generation using insertion operations (Gu et al., 2019; Stern et al., 2019) and Gu et al. (2017) explored parallel decoding for machine translation. In contrast, we focus on decomposing generation to focus on planning, rather than efficient decoding at inference time.

8 Conclusion

We proposed an effective method for writing short stories by separating the generation of actions and entities. We show through human evaluation and automated metrics that our novel decomposition improves story quality.

References

D Scott Appling and Mark O Riedl. 2009. Representations for learning to summarize plots.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *Proc. of ICLR*.

Elizabeth Clark, Yangfeng Ji, and Noah A Smith. 2018. Neural text generation in stories using entity representations as context. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 2250–2260.

Zihang Dai, Zhilin Yang, Yiming Yang, William W Cohen, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2018. Transformer-xl: Language modeling with longer-term dependency.

Hercules Dalianis and Eduard Hovy. 1993. Aggregation in natural language generation. In *European Workshop on Trends in Natural Language Generation*, pages 88–105. Springer.

Angela Fan, Mike Lewis, and Yann Dauphin. 2018. Hierarchical neural story generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Katja Filippova and Yasemin Altun. 2013. Overcoming the lack of parallel data in sentence compression. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*.

Nicolas Ford, Daniel Duckworth, Mohammad Norouzi, and George E Dahl. 2018. The importance of generation order in language modeling. *arXiv preprint arXiv:1808.07910*.

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research*, 61:65–170.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proc. of ICML*.

Yoshihiko Gotoh, Steve Renals, and Gethin Williams. 1999. Named entity tagged language models. In *Acoustics, Speech, and Signal Processing, 1999. Proceedings., 1999 IEEE International Conference on*, volume 1, pages 513–516. IEEE.

Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics*, 21(2):203–225.

Jiatao Gu, James Bradbury, Caiming Xiong, Victor OK Li, and Richard Socher. 2017. Non-autoregressive neural machine translation. *arXiv preprint arXiv:1711.02281*.

Jiatao Gu, Qi Liu, and Kyunghyun Cho. 2019. Insertion-based decoding with automatically inferred generation order. *arXiv preprint arXiv:1902.01370*.

Luheng He, Kenton Lee, Mike Lewis, and Luke Zettlemoyer. 2017. Deep semantic role labeling: What works and whats next. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in neural information processing systems*, pages 1693–1701.

Eduard H Hovy. 1990. Pragmatics and natural language generation. *Artificial Intelligence*, 43(2):153–197.

Ting-Hao Kenneth Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, et al. 2016. Visual storytelling. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1233–1239.

Parag Jain, Priyanka Agrawal, Abhijit Mishra, Mohak Sukhwani, Anirban Laha, and Karthik Sankaranarayanan. 2017. Story generation from sequence of independent short descriptions. *arXiv preprint arXiv:1707.05501*.

Yangfeng Ji, Chenhao Tan, Sebastian Martschat, Yejin Choi, and Noah A Smith. 2017. Dynamic entity representations in neural language models. *arXiv preprint arXiv:1708.00781*.

Sosuke Kobayashi, Naoaki Okazaki, and Kentaro Inui. 2017. A neural language model for dynamically representing the meanings of unknown words and entities in a discourse. *arXiv preprint arXiv:1709.01679*.

Kenton Lee, Luheng He, and Luke Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 687–692.

Boyang Li, Stephen Lee-Urban, George Johnston, and Mark Riedl. 2013. Story generation with crowdsourced plot graphs.

Feifan Liu and Yang Liu. 2007. Unsupervised language model adaptation incorporating named entity information. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 672–679.

Peter J Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. *arXiv preprint arXiv:1801.10198*.

Robert Malouf. 2000. The order of prenominal adjectives in natural language generation. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*.

Lara J Martin, Prithviraj Ammanabrolu, William Hancock, Shruti Singh, Brent Harrison, and Mark O Riedl. 2017. Event representations for automated story generation with deep neural nets. *arXiv preprint arXiv:1706.01331*.

Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. 2016. Caters: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, pages 51–61.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, Bing Xiang, et al. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. *arXiv preprint arXiv:1602.06023*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT 2019: Demonstrations*.

Md Rizwan Parvez, Saikat Chakraborty, Baishakhi Ray, and Kai-Wei Chang. 2018. Building language models for text with named entities. *arXiv preprint arXiv:1805.04836*.

Julie Porteous and Marc Cavazza. 2009. Controlling narrative generation with planning trajectories: the role of constraints. In *Joint International Conference on Interactive Digital Storytelling*, pages 234–245. Springer.

Adwait Ratnaparkhi. 2000. Trainable methods for surface natural language generation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 194–201. Association for Computational Linguistics.

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press.

Mark O Riedl. 2010. Story planning: Creativity through exploration, retrieval, and analogical transformation. *Minds and Machines*, 20(4):589–614.

Mark O Riedl and Robert Michael Young. 2010. Narrative planning: Balancing plot and character. *Journal of Artificial Intelligence Research*, 39:217–268.

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory*.

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization with pointer-generator networks. *arXiv preprint arXiv:1704.04368*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. *arXiv preprint arXiv:1508.07909*.

Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. Insertion transformer: Flexible sequence generation via insertion operations. *arXiv preprint arXiv:1902.03249*.

Zhixing Tan, Mingxuan Wang, Jun Xie, Yidong Chen, and Xiaodong Shi. 2018. Deep semantic role labeling with self-attention. In *AAAI Conference on Artificial Intelligence*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention Is All You Need. In *Proc. of NIPS*.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2692–2700.

Wolfgang Wahlster, Elisabeth André, Wolfgang Finkler, Hans-Jürgen Profitlich, and Thomas Rist. 1993. Plan-based integration of natural language and graphics generation. *Artificial intelligence*, 63(1-2):387–427.

Felix Wu, Angela Fan, Alexei Baevski, Yann Dauphin, and Michael Auli. 2019. Pay less attention with lightweight and dynamic convolutions. In *International Conference on Learning Representations*.

Jingjing Xu, Yi Zhang, Qi Zeng, Xuancheng Ren, Xiaoyan Cai, and Xu Sun. 2018. A skeleton-based model for promoting coherence among sentences in narrative story generation. *arXiv preprint arXiv:1808.06945*.

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

Deyu Zhou, Linsen Guo, and Yulan He. 2018. Neural storyline extraction model for storyline generation from news articles. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, volume 1, pages 1727–1736.