The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Generating Sentences Using a Dynamic Canvas

Harshil Shah
University College London

Bowen Zheng
University College London

David Barber
University College London
and Alan Turing Institute

Abstract

We introduce the ***A**ttentive **U**nsupervised **T**ext (W)**r**iter* (AUTR), which is a word level generative model for natural language. It uses a recurrent neural network with a dynamic attention and canvas memory mechanism to iteratively construct sentences. By viewing the state of the memory at intermediate stages and where the model is placing its attention, we gain insight into how it constructs sentences. We demonstrate that AUTR learns a meaningful latent representation for each sentence, and achieves competitive log-likelihood lower bounds whilst being computationally efficient. It is effective at generating and reconstructing sentences, as well as imputing missing words.

1 Introduction

Latent variable models have recently enjoyed significant success when modelling images (Gregor et al. 2015; Rezende et al. 2016; Gulrajani et al. 2017), as well as sequential data such as handwriting and speech (Bayer and Osendorfer 2015; Chung et al. 2015). They specify a conditional distribution of observed data, given a set of hidden (latent) variables. The stochastic gradient variational Bayes (SGVB) algorithm (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014) has made (approximate) maximum likelihood learning possible on a large scale in models where the true posterior distribution of the latent variables is not tractable. Deep neural networks can be used to parametrise the generative and variational distributions, allowing for extremely flexible and powerful model classes.

There has been somewhat less exploration into deep generative models for natural language. Graves (2014) uses a stacked RNN architecture at the character level to generate sentences with long range dependencies. The model sequentially emits characters based on the previously generated ones, however it does not map each sentence to a single latent representation. This means that even though the generated sentences are syntactically coherent and may show local semantic consistency, the model does not encourage the sentences to have long range semantic consistency. Additionally, the model cannot generate sentences conditioned on meaning, style, etc. Bowman et al. (2016) use a word level latent variable model with an RNN and train it using SGVB (we refer to this as Gen-RNN). Samples from the prior produce well-formed, coherent sentences, and the model is effective at imputing missing words. However, the authors find that the KL divergence term of the log-likelihood lower bound reduces to 0, which implies that the model ignores the latent representation and collapses to a standard RNN language model, similar to that of Graves (2014). The authors use word dropout to alleviate this problem, and show that Gen-RNN generates sentences with more varied vocabulary and is better at imputing missing words than the RNN language model. Semeniuta, Severyn, and Barth (2017) and Yang et al. (2017) make use of convolutional layers, which appear to encourage their models to more strongly rely on the latent representation without using word dropout.

In the context of computer vision, DRAW (Gregor et al. 2015) showed that using an attention mechanism to 'paint' locally on a canvas produced images of remarkable quality. A natural question therefore is whether a similar approach could work well for natural language. To this end we introduce the ***A**ttentive **U**nsupervised **T**ext (W)**r**iter* (AUTR), which is a word level generative model for text; AUTR uses an RNN with a dynamic attention mechanism to iteratively update a canvas (analogous to an external memory (Gemici et al. 2017)).

Using an attention mechanism in this way can be very powerful—it allows the model to use the RNN to focus on local parts of the sentence at each time step whilst relying on the latent representation to encode global sentence features. By viewing the canvas at intermediate stages, and where the RNN is placing its attention, we gain insight into how the model constructs a sentence. Additionally, we verify that AUTR attains competitive lower bounds on the log-likelihood whilst being computationally efficient. As well as learning a meaningful latent representation for each sentence, the model generates coherent sentences and successfully imputes missing words. A generative model which is able to, in some sense, 'understand' natural language (which we believe AUTR shows signs of doing) should facilitate much better performance when used as a module for downstream tasks such as translation and question answering.

The remainder of this paper is structured as follows: in section 2 we define the AUTR architecture and generative process, in section 3 we review related work, in section 4

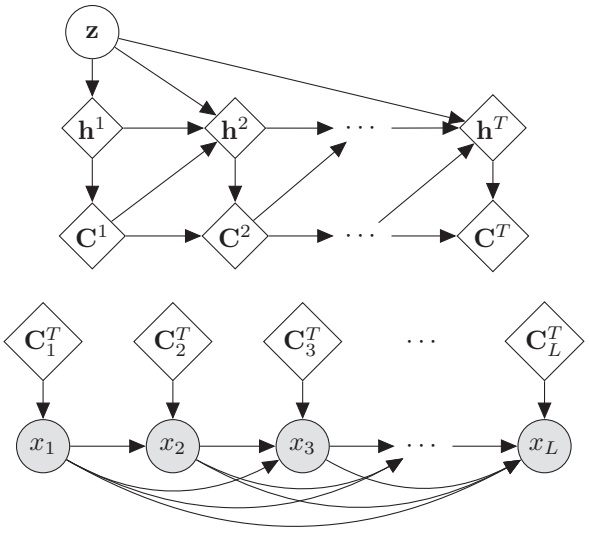

Figure 1: The AUTR generative model. The shaded nodes signify the observed words and non-shaded nodes signify latent random variables; a rhombus node is a deterministic function of its incoming variables. Here $\mathbf{z}$ denotes the latent sentence representation; $\mathbf{C}^t \; \forall \; t \in \{1, \ldots, T\}$ are the states of the canvas; $\mathbf{C}_l^T \; \forall \; l \in \{1, \ldots, L\}$ are the slots of the final canvas corresponding to each word; $x_l \; \forall \; l \in \{1, \ldots, L\}$ are the observed words.

we provide experimental results on the Book Corpus dataset along with examples of generated sentences, and in section 5 we make concluding remarks. Appendix A provides a review of the SGVB algorithm.

2 Model

AUTR is a word level generative recurrent neural network (RNN) which iteratively updates a canvas that parametrises the probability distribution over the sentence's text. Using L to denote the number of words in the sentence and E to denote the word embedding size, the canvas $\mathbf{C} \in \mathbb{R}^{L \times E}$ is a 2 dimensional array with L 'slots', each of which represents the model's estimation of the word embedding for that position in the sentence.

We use T to denote the number of time steps in the RNN—at each time step an attention mechanism selects the canvas' slots to be updated; $\mathbf{C}^t$ denotes the state of the canvas at time step t. Note that T is a hyper-parameter of the model.

Figure 1 shows the graphical model for AUTR and figure 2 shows examples of how AUTR iteratively constructs sentences. Like other latent variable models (Bowman et al. 2016; Semeniuta, Severyn, and Barth 2017; Yang et al. 2017), AUTR uses a hidden representation $\mathbf{z}$ to encode each sentence, and can construct new sentences by sampling $\mathbf{z}$ from a prior distribution $p(\mathbf{z})$ and passing this sample through the RNN. A summary of the generative process is given in algorithm 1 and full details follow in section 2.1.

Algorithm 1: AUTR generative process

Sample a latent vector $\mathbf{z}$ from a $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distribution.
Initialise both the hidden state and canvas as zeros, *i.e.* $\mathbf{h}^0 = \mathbf{0}$ and $\mathbf{C}^0 = \mathbf{0}$
for $t = 1, \ldots, T$ **do**
  Compute the hidden state: $\mathbf{h}^t = f(\mathbf{z}, \mathbf{h}^{t-1}, \mathbf{C}^{t-1})$
  Compute the gate: $\mathbf{g}_l^t = \frac{\exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_l\right]\left(1 - \sum_{t'=1}^{t-1} \mathbf{g}_l^{t'}\right)}{\sum_{k=1}^{L} \exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_k\right]\left(1 - \sum_{t'=1}^{t-1} \mathbf{g}_k^{t'}\right)} \left(1 - \sum_{t'=1}^{t-1} \mathbf{g}_l^{t'}\right)$ for $l = 1, \ldots, L$
  Update the canvas: $\mathbf{C}^t = (\mathbf{1} - \mathbf{g}^t) \odot \mathbf{C}^{t-1} + \mathbf{g}^t \odot (\mathbf{W}_u \cdot \mathbf{h}^t)$
end
Sample the sentence according to the distribution: $\mathbf{x} \sim p(\mathbf{x}|\mathbf{z}, \mathbf{C}^T) = p(x_1|\mathbf{z}, \mathbf{C}_1^T) \prod_{l=2}^{L} p(x_l|\mathbf{z}, \mathbf{C}_l^T, x_{1:l-1})$

2.1 Generative process

We first sample the latent representation $\mathbf{z}$ from a $\mathcal{N}(\mathbf{0}, \mathbf{I})$ distribution. Each RNN hidden state is then computed as a function of this latent representation, as well as the previous RNN hidden state and the canvas so far: $\mathbf{h}^t = f(\mathbf{z}, \mathbf{h}^{t-1}, \mathbf{C}^{t-1})$. In our experiments, we use the LSTM for $f(\cdot)$ (Hochreiter and Schmidhuber 1997). Allowing the RNN hidden state to see what has been written to the canvas so far allows the model to maintain the sentence's long range semantic coherence because the hidden state can anticipate the words that will be written at the end of the sentence and adjust the beginning accordingly, and vice versa.

Each hidden state $\mathbf{h}^t$ is then used to determine where to write (or more specifically, how 'strongly' to write to each of the canvas' slots). We denote the gate as $\mathbf{g}_l^t \in [0, 1]$ for $l = 1, \ldots, L$.

Attention mechanism For the gate (or attention), we use a modified softmax attention mechanism. A standard softmax mechanism (Luong, Pham, and Manning 2015) would be:

$$\mathbf{g}_l^t = \frac{\exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_l\right]}{\sum_{k=1}^{L} \exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_k\right]} \tag{1}$$

This would ensure that (at each time step) the attention for each slot is between 0 and 1 and the total attention across all slots is 1. To encourage the model to write to those slots where it hasn't yet written, we multiply the elements of the softmax by $(1 - \sum_{t'=1}^{t-1} \mathbf{g}_l^{t'})$. To ensure that the cumulative attention, over time, applied to any of the slots is no greater than 1, we multiply the softmax itself by $(1 - \sum_{t'=1}^{t-1} \mathbf{g}_l^{t'})$. This results in the following modified attention mechanism:

$$\mathbf{g}_l^t = \frac{\exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_l\right] s_l}{\sum_{k=1}^{L} \left[\exp\left[(\mathbf{W}_g \cdot \mathbf{h}^t)_k\right] s_k\right]} \cdot s_l \tag{2}$$

where $s_l = \left(1 - \sum_{t'=1}^{t-1} \mathbf{g}_l^{t'}\right)$. In our experiments, we found that this modification performed favourably compared to the

standard softmax mechanism. Note that, once a slot has been written to with a cumulative attention of 1 (*i.e.* $\sum_{t'=1}^{t-1} \mathbf{g}_l^{t'} = 1$), it cannot be updated further.

One of the key computational advantages of AUTR is that it works well with $T < L$, because it writes to multiple slots at each RNN time step. This is shown to work well empirically in section 4.4.

Updating the canvas The content to be written to the canvas is a linear function of the hidden state at that time step: $\mathbf{U}^t = \mathbf{W}_u \cdot \mathbf{h}^t$.

Using $\odot$ to denote element-wise multiplication, the canvas is updated as: $\mathbf{C}^t = (\mathbf{1} - \mathbf{g}^t) \odot \mathbf{C}^{t-1} + \mathbf{g}^t \odot \mathbf{U}^t$.

$\mathbf{g}_l^t = 0$ means that the l^{th} slot from the previous time step carries over exactly to the current time step (*i.e.* no updating takes place), whereas $\mathbf{g}_l^t = 1$ means that the previous values of the l^{th} slot are completely forgotten and new values are entered in their place.

We tested a fixed mechanism instead of one with attention to update the canvas, but found that the RNN didn't use the T computational steps available - at the final time step it simply overwrote everything it had previously written.

Text generation Conditioned on the final canvas, we considered first a model that generated each word independently. However, this was ineffective in our experiments since local consistency between words was lost. Therefore, we sample the sentence's text from a Markov model where the sampled word at position l depends on the l^{th} slot of the final canvas $\mathbf{C}_l^T$ and on all of the $l-1$ words that have been sampled so far. We first compute the 'context' $\tilde{\mathbf{x}}_l$, which is a weighted average of the previously generated words; this is then used to modify the word probabilities that would have been assigned by the canvas alone. Specifically:

$$\tilde{\mathbf{x}}_l = \sum_{l'=1}^{l-1} w_{l'}^l(\mathbf{z}) \mathbf{e}(x_{l'}) \tag{3}$$

where $\mathbf{e}(x)$ is the embedding of word x and:

$$w_{l'}^l(\mathbf{z}) = \frac{\exp[(\mathbf{W}_l \cdot \mathbf{z})_{l'}]}{\sum_{l'=1}^{l-1} \exp[(\mathbf{W}_l \cdot \mathbf{z})_{l'}]} \tag{4}$$

Therefore $\sum_{l'=1}^{l-1} w_{l'}^l(\mathbf{z}) = 1$. Then, denoting $\mathbf{b}_l = \mathbf{C}_l^T + \mathbf{W}_x \cdot \tilde{\mathbf{x}}_l + \mathbf{W}_z \cdot \mathbf{z}$:

$$p(x_l = a | \mathbf{z}, x_{1:l-1}, \mathbf{C}_l^T) = \frac{\exp[\mathbf{e}(a) \cdot \mathbf{b}_l]}{\sum_{v \in \mathcal{V}} \exp[\mathbf{e}(v) \cdot \mathbf{b}_l]} \tag{5}$$

where $\mathcal{V}$ is the entire vocabulary.

2.2 Inference

Due to the intractability of the true posterior $p(\mathbf{z}|\mathbf{x})$, we perform (approximate) maximum likelihood estimation using SGVB, as described in appendix A. The variational distribution is Gaussian: $q_\phi(\mathbf{z}|\mathbf{x}) = \mathcal{N}(\boldsymbol{\mu}_\phi(\mathbf{x}), \text{diag}(\boldsymbol{\sigma}_\phi^2(\mathbf{x})))$; its mean and variance are parametrised by an RNN which takes as input each word embedding in order, one at a time (Bowman et al. 2016). For the hidden states, we use the LSTM and the final hidden state is passed through a feedforward network with two output layers to produce the mean and variance of the variational distribution.

3 Related Work

The use of stochastic gradient variational Bayes (SGVB) (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014) to train latent variable generative models is widespread for images (Gregor et al. 2015; Rezende et al. 2016; Gulrajani et al. 2017).

Whilst latent variable generative models for natural language have been less common, they have recently increased in popularity. Bowman et al. (2016) use an LSTM for the generative model (Gen-RNN), which outputs a single word at each time step. This is in contrast to AUTR, which updates the distribution for every word in the sentence at each RNN time step, using an attention mechanism.

More recently, Miao, Grefenstette, and Blunsom (2017) use an RNN based generative model similar to Gen-RNN in order to perform topic allocation to documents. Yang et al. (2017) use dilated convolutions to replace the recurrent structure in Gen-RNN in order to control the contextual capacity, leading to better performance.

Our canvas based model with its dynamic attention mechanism is largely inspired by DRAW (Gregor et al. 2015), which iteratively updates the canvas that parametrises the final distribution over the observed image. One of the primary differences between DRAW and AUTR is that conditioned on the final canvas DRAW treats each pixel independently—whilst this may not be too constraining in the image domain, this was ineffective in our natural language experiments where the sampled word embedding may differ significantly from the entry in that slot of the canvas. Therefore AUTR conditions on all previously generated words in the sentence when sampling the next word.

4 Experiments

We train our model on the Book Corpus dataset (Zhu et al. 2015), which is composed of sentences from 11,038 unpublished books. We report results on language modelling tasks, comparing against Gen-RNN.

4.1 Preprocessing

We restrict the vocabulary size to 20,000 words and we use sentences with a maximum length of 40 words. Of the 53M sentences that meet these criteria, we use 90% for training, and 10% for testing.

4.2 Model architectures

Generative model For both AUTR and Gen-RNN, we use a 50 dimensional latent representation $\mathbf{z}$ and the RNN hidden states have 500 units each. To compute the hidden states, we use the LSTM (Hochreiter and Schmidhuber 1997).

Gen-RNN requires $T = L = 40$, because it outputs one word at each RNN time step. However, as explained in section 2.1, one of the key advantages of AUTR is that it works well with $T < L = 40$. Therefore, for AUTR, we compare results with $T \in \{30, 40\}$.

Variational distribution As per section 2.2, we optimise both AUTR and Gen-RNN using SGVB. In both models, we use the LSTM architecture introduced by Bowman et al.

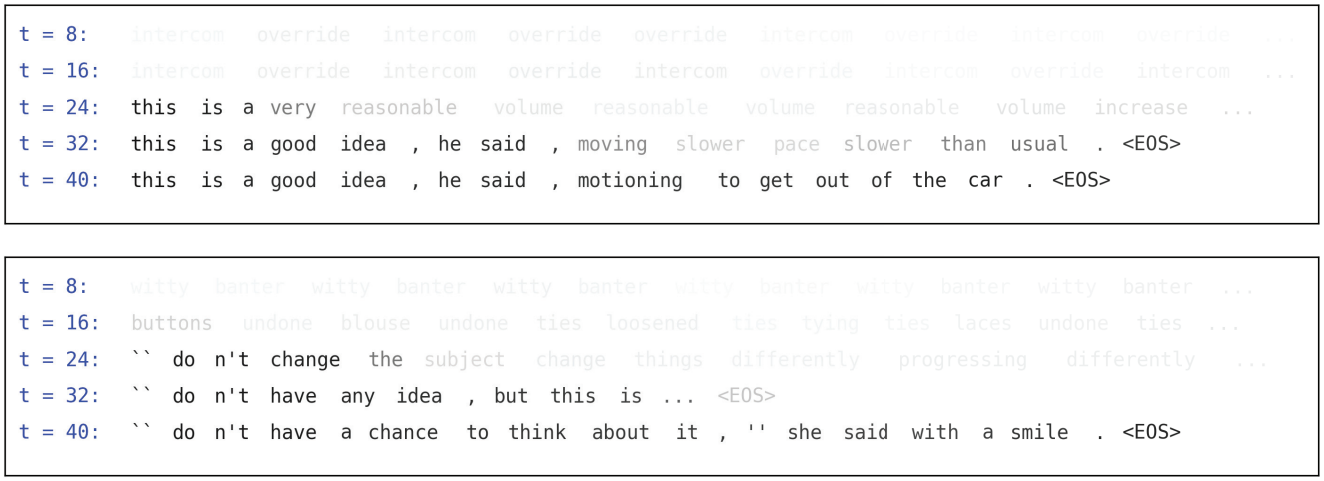

Figure 2: Visualising the sequential construction of sentences generated from the learned model. We sample $\mathbf{z}$ from its prior, *i.e.* $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ and pass the sample through the RNN, visualising the canvas at several points along the way. The darkness indicates the cumulative attention that has been placed on that slot so far.

(2016) to parametrise the mean and variance of the Gaussian variational distribution; the hidden states have 500 units each.

Parameters and speed For both models, we use 300 dimensional word embeddings, which are learned jointly with the generative and variational parameters. AUTR and Gen-RNN have 10.9M and 10.3M parameters respectively. They take, on average, 0.19 and 0.17 seconds per training iteration and 0.06 and 0.05 seconds for sentence generation respectively.[1]

4.3 Training process

We optimise the ELBO, shown in equation (7), using stochastic gradient ascent. We train both models for 1,000,000 iterations, using Adam (Kingma and Ba 2015) with an initial learning rate of 10^{-4} and mini-batches of size 200. To ensure training is fast, we use only a single sample $\mathbf{z}$ per data point from the variational distribution at each iteration. We implement both models in Python, using the Theano (Theano Development Team 2016) and Lasagne (Dieleman et al. 2015) libraries.

KL divergence annealing The ELBO can be expressed as: $\mathcal{L}(\mathbf{x}) = \mathbb{E}_{q_\phi(\mathbf{z}|\mathbf{x})}\left[\log p(\mathbf{x}|\mathbf{z})\right] - D_{\mathrm{KL}}\left[q_\phi(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z})\right]$. We multiply the KL divergence term by a constant weight, which we linearly anneal from 0 to 1 over the first 20,000 iterations of training (Bowman et al. 2016; Sønderby et al. 2016).

Word dropout To encourage Gen-RNN to make better use of the latent representation, Bowman et al. (2016) randomly drop out a proportion of the words when training the generative RNN - without this, they find that their model collapses to a simple RNN language model which ignores the latent representation. Following this, when training Gen-RNN, we randomly drop out 30% of the words. However, to show that (unlike Gen-RNN) AUTR does not need the dropout mechanism to avoid the KL divergence term $D_{\mathrm{KL}}\left[q_\phi(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z})\right]$ from collapsing to 0, we train it both with 30% dropout, and without any dropout.

[1]These values are for AUTR with $T = 40$.

4.4 Results

We report test set results on the Book Corpus dataset in table 1. We evaluate the ELBO on the test set by drawing 1,000 samples of the latent vector $\mathbf{z}$ per data point. We see that AUTR, both with $T = 30$ and $T = 40$, trained with or without dropout, achieves a higher ELBO and lower perplexity than Gen-RNN. Importantly, AUTR (trained with and without dropout) relies more heavily on the latent representation than Gen-RNN, as is shown by the larger contribution to the ELBO from the KL divergence term. Note that if a model isn't taking advantage of the latent vector $\mathbf{z}$, the loss function drives it to set $q(\mathbf{z}|\mathbf{x})$ equal to the prior on $\mathbf{z}$ (disregarding $\mathbf{x}$), which yields a KL divergence of zero.

Model		ELBO	KL	PPL
Gen-RNN (30% dropout)		-52.3	7.1	41.9
AUTR (no dropout)	T = 30	-50.7	8.0	37.4
	T = 40	-50.7	7.8	37.4
AUTR (30% dropout)	T = 30	-51.6	14.0	39.9
	T = 40	-51.5	13.8	39.6

Table 1: Test set results on the Book Corpus dataset. We report the ELBO, the contribution to the ELBO from the KL divergence term ($D_{\mathrm{KL}}\left[q_\phi(\mathbf{z}|\mathbf{x}) \,||\, p(\mathbf{z})\right]$), and the perplexity (PPL) on the test set. For the ELBO, higher is better, and for the perplexity, lower is better.

4.5 Observing the generation process

Conditioned on a sampled $\mathbf{z}$, we would like to know the most likely sentence, *i.e.* $\arg\max_{\mathbf{x}} \log p_\theta(\mathbf{x}|\mathbf{z})$. However, because each word depends on all of the words generated before it, this optimisation has a computationally intractable memory requirement. We therefore perform this maximisation approximately by considering only the K 'best' trajectories for each position in the sentence - this is known as beam search with beam size K (Wiseman and Rush 2016).

In figure 2, we show examples of how the canvas changes as the RNN makes updates to it. At each time step we take the state of the canvas and plot the sequence of words found using beam search with a beam size of 15. In figure 3, we plot the cumulative attention that has been placed on each of the canvas' slots at each RNN time step.

In figure 3, the model's attention appears to spend the first 15 time steps to decide how long the sentence will be (19 words in this case), and then spends the remaining 25 time steps filling in the words from left to right (even though it is not restricted to do so). It is notable that the model is able to dynamically adjust the length of the sentences by moving the end-of-sentence token and either inserting or deleting words as it sees fit. The model is also able to notice sentence features such as the open quotation marks at $t = 32$ in the second example of figure 2, which it subsequently closes at $t = 40$.

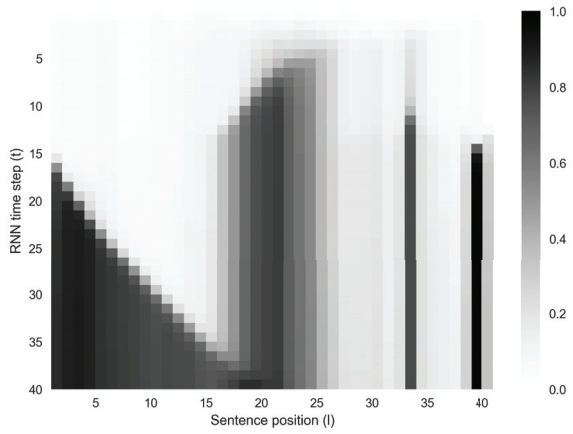

Figure 3: Visualising the cumulative attention on each of the sentence's slots for the second example of figure 2. The darker the shade, the more attention has been placed on that position.

4.6 Sampled sentences

In tables 2 and 3, we show samples of text generated from the prior and posterior distributions, for both AUTR and Gen-RNN. Once again, we show the sequence of words found using beam search with a beam size of 15. In both models, sampling from the prior often produces syntactically coherent sentences. However, the AUTR samples appear to better represent the global semantics of the sentences. This is likely due to the canvas feedback mechanism when computing the RNN hidden states, and the ability of the model to place attention on the entire sentence at each time step.

When attempting to reconstruct a sentence, it appears that both models' latent representations capture information about meaning and length. For example, in the second sentence of table 3, both models are able to recognise that there is a question. The evidence of the AUTR latent representation learning meaning better than Gen-RNN is evident in several of the examples in table 3, and this is quantitatively verified by the larger contribution to AUTR's ELBO from the KL divergence term.

4.7 Imputing missing words

AUTR's latent sentence representation makes it particularly effective at imputing missing words. To impute missing words, we use an iterative procedure inspired by the EM algorithm (Neal and Hinton 1998). We increase a lower bound on the log-likelihood of the visible and missing data, *i.e.* $\log p_\theta(\mathbf{x}_{\text{vis}}, \mathbf{x}_{\text{miss}})$, by iterating between an E-like and M-like step, as described in algorithm 2. The M-like step treats the missing words as model parameters, and hence (approximately, using beam search) maximises the lower bound with respect to them.

We drop 30% of the words from each of the test set sentences, and run algorithm 2 with 50 different initialisations for the missing words, and select the resulting imputation with the highest bound on the log-likelihood. AUTR successfully imputes 34.1% of the missing words, whilst Gen-RNN achieves 31.9%. Sampled missing word imputations for AUTR and Gen-RNN are shown in table 4.

Algorithm 2: Missing data imputation

1 Make an initial (random) 'guess' for the missing words.
2 **while** *not converged* **do**
3 **E-like step**: Sample $\mathbf{z}$ from its variational distribution $q_\phi(\mathbf{z}|\mathbf{x})$, where $\mathbf{x}$ is the latest setting of the sentence.
4 **M-like step**: Choose the missing words in $\mathbf{x}$ to maximise $\frac{1}{S}\sum_{s=1}^{S} \log p_\theta(\mathbf{x}_{\text{vis}}, \mathbf{x}_{\text{miss}}, |\mathbf{z}^{(s)})$. This is done approximately, using beam search.
5 **end**

4.8 Exploring the latent space

Finding similar sentences To compare the quality of the latent representations under each model, we take a sentence $\tilde{\mathbf{x}}$ from the test set and compute the mean of its posterior distribution, $\boldsymbol{\mu}_\phi(\tilde{\mathbf{x}})$. We then find the 'best matching' sentence $\mathbf{x}^*$ in the remainder of the test set, which satisfies: $\mathbf{x}^* = \arg\max_{\mathbf{x}\neq\tilde{\mathbf{x}}} \log p_\theta(\mathbf{x}|\mathbf{z} = \boldsymbol{\mu}_\phi(\tilde{\mathbf{x}}))$. If the latent representation does indeed capture the sentence's meaning, $\mathbf{x}^*$ ought to appear qualitatively similar to $\tilde{\mathbf{x}}$.

We show some examples of the best matching sentences using AUTR and Gen-RNN in table 5; we see that the AUTR latent representations are generally successful at capturing

AUTR	Gen-RNN
"do you have any idea how much i love you when i'm with you?"	but i didn't want to think of any other way to get it.
"if he didn't want to kill me," he said , but he was trying to keep distance.	when i reach the top of the stairs , i feel of sight of my back into the doors open the door swings .
hundreds of thousands of stars rose above, reflecting the sky above the horizon.	i had no idea what i was going to do, but i was wrong.
"that sounds like a good idea," he said, as his voice trailed off.	"you're going to look at least one of course, but i have been in the most of course," he said.

Table 2: Sentences sampled from the prior: $\mathbf{z}$ is drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I})$ and passed through the generative model $p_\theta(\mathbf{x}|\mathbf{z})$ to produce a sentence $\mathbf{x}$.

INPUT ($\mathbf{x}$)	RECONSTRUCTION (AUTR)	RECONSTRUCTION (Gen-RNN)
unable to stop herself, she briefly, gently, touched his hand.	unable to stop herself, she leaned forward, and touched his eyes.	unable to help her , and her back and her into my way.
why didn't you tell me?	why didn't you tell me?	why didn't you tell me?"
a strange glow of sunlight shines down from above, paper white and blinding, with no heat.	the light of the sun was shining through the window, illuminating the room.	a tiny light on the door, and a few inches from behind him out of the door.
he handed her the slip of paper.	he handed her a piece of paper.	he took a sip of his drink.

Table 3: Sentences sampled from the posterior: conditioned on a test set input sentence $\mathbf{x}$, $\mathbf{z}$ is drawn from its variational distribution, $q_\phi(\mathbf{z}|\mathbf{x})$, and passed through the generative model $p_\theta(\mathbf{x}'|\mathbf{z})$ to produce a reconstruction $\mathbf{x}'$.

the sentences' meanings and are able to learn sentence features such as tense and gender as well.

Interpolating between latent representations To further understand how AUTR uses its latent representation, we randomly sample two latent representations from the prior distribution, and linearly interpolate between them. That is, we sample $\mathbf{z}^{(1)}$ and $\mathbf{z}^{(2)}$ from the $\mathcal{N}(\mathbf{0}, \mathbf{I})$ prior, and then for $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$, we take:

$$\mathbf{z}^{(\alpha)} = \alpha \mathbf{z}^{(1)} + (1 - \alpha)\mathbf{z}^{(2)} \tag{6}$$

Then, using beam search with a beam size of 15, we evaluate the sentence produced by $\mathbf{z}^{(\alpha)}$. Three examples are shown in table 6; it is clear that in all of these examples, AUTR maintains the sentence's syntactic structure throughout the interpolations. It is also notable that the sentence topics and meanings remain consistent as α increases.

5 Conclusion

We introduce the ***A**ttentive **U**nsupervised **T**ext (W)**r**iter* (AUTR), a latent variable model which uses an external memory and a dynamically updated attention mechanism to write natural language sentences. We visualise this external memory at intermediate stages to understand the sentence generation process. We find that the model achieves a higher ELBO on the Book Corpus dataset, and relies more heavily on the latent representation compared to a purely RNN-based generative model. AUTR is also computationally efficient, requiring fewer RNN time steps than the sentence length. In addition, we verify that it is able to generate coherent sentences, as well as impute missing words effectively.

We have shown that the idea of using a canvas-based mechanism to generate text is very promising and presents plenty of avenues for future research. We would like to investigate alternatives to zero-padding and the use of fixed size canvases, as well as making the number of RNN time steps dependent on the sentence's latent representation. It may also be worthwhile to consider using convolutional layers, as presented by Semeniuta, Severyn, and Barth (2017) and Yang et al. (2017). A particularly interesting avenue of interest is to use an RNN with a similar attention mechanism to parametrise the variational distribution; this would also facilitate extending the model to other natural language tasks such as document classification and translation.

6 Acknowledgements

This work was supported by the Alan Turing Institute under the EPSRC grant EP/N510129/1.

References

Bayer, J., and Osendorfer, C. 2015. Learning Stochastic Recurrent Networks. *arXiv preprint* arXiv:1411.7610.

Bowman, S. R.; Vilnis, L.; Vinyals, O.; Dai, A. M.; Jozefowicz, R.; and Bengio, S. 2016. Generating Sentences from a Continuous Space. Conference on Computational Natural Language Learning (CoNLL).

Chung, J.; Kastner, K.; Dinh, L.; Goel, K.; Courville, A.; and Bengio, Y. 2015. A Recurrent Latent Variable Model for Sequential Data. In *Advances in Neural Information Processing Systems 28*, 2980–2988.

TRUTH	IMPUTATION (AUTR)	IMPUTATION (Gen-RNN)
"i want to draw you again," he says. "i want to ____ you _____,_ he ____.	"i want to see you again," he said.	"i want to see you again," he said.
i believe the lie, and so i survive another day. _ believe the lie_ and so _ survive another ___.	i believe the lie, and so will survive another day.	do believe the lie too and so will survive another day.
he was inside a house made of cheese. __ ___ inside a house made __ cheese_	he was inside a house made of cheese.	it is inside a house made a cheese.
i could have saved more of them if we had realized back then. i _____ have saved more __ them __ we had realized back ____.	i would have saved more of them if we had realized back there.	i should have saved more of them than we had realized back then.

Table 4: Imputing missing words in test set sentences, using the procedure described in algorithm 2. Those words replaced with underscores (_) are considered as missing.

$\tilde{\mathbf{x}}$	$\mathbf{x}^*_{\mathrm{AUTR}}$	$\mathbf{x}^*_{\mathrm{Gen-RNN}}$
he wasn't ready to face the prospect of losing her when he'd only just gotten her back.	he was never going to see her again, and that was the way it had to be.	she didn't want to make any promises, no matter how much she wanted to be with him again.
i can't help but glare at her.	i can't help but smile at her.	i couldn't help but smile at him.
so i stood in the doorway of the chapel, watching it happen.	as i sat on the bench outside the hospital, i looked up.	when he reached the bottom of the hill, he slowed his pace.
dina lets a breath out on the other side of the line.	there is a long pause on the other end of the line.	he reached into his pocket and pulled out a piece of paper.

Table 5: Finding the 'best matching' sentence using the latent representation.

α	Example 1	Example 2	Example 3
0	maybe it wasn't going to happen.	"oh, thank you."	one of them looked at me, and smiled.
0.25	it would be nice to have to go.	"oh, it's nice to meet you."	the moment i met her eyes, she smiled at me.
0.5	he wasn't sure what it was about.	"it's nice to meet you," he said.	the moment i stared at him, he looked down at me.
0.75	i wasn't sure whether or not to talk about it.	"i had no idea what happened," i told him.	instead, he looked at me, whilst i stared at the ceiling.
1	i had no idea how much time to talk about the phone calls.	he couldn't believe what i was talking about when i told him.	"finally," he said, standing up in front of me.

Table 6: Linearly interpolating between latent representations and evaluating the intermediate sentences.

Dieleman, S.; Schlüter, J.; Raffel, C.; Olson, E.; Sønderby, S. K.; et al. 2015. Lasagne: First release.

Gemici, M.; Hung, C.; Santoro, A.; Wayne, G.; Mohamed, S.; Rezende, D. J.; Amos, D.; and Lillicrap, T. P. 2017. Generative Temporal Models with Memory. *arXiv preprint* arXiv:1702.04649.

Graves, A. 2014. Generating Sequences With Recurrent Neural Networks. *arXiv preprint* arXiv:1308.0850.

Gregor, K.; Danihelka, I.; Graves, A.; Rezende, D. J.; and Wierstra, D. 2015. DRAW: A Recurrent Neural Network For Image Generation. In *Proceedings of the 32nd International Conference on Machine Learning*, 1462–1471.

Gulrajani, I.; Kumar, K.; Ahmed, F.; Taiga, A. A.; Visin, F.; Vazquez, D.; and Courville, A. 2017. PixelVAE: A Latent Variable Model for Natural Images. In *International Conference on Learning Representations*.

Hochreiter, S., and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation* 9(8):1735–1780.

Kingma, D. P., and Ba, J. 2015. Adam: A Method for Stochastic Optimization. In *International Conference on*

Learning Representations.

Kingma, D. P., and Welling, M. 2014. Auto-Encoding Variational Bayes. In *International Conference on Learning Representations*.

Luong, M.; Pham, H.; and Manning, C. D. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, 1412–1421.

Miao, Y.; Grefenstette, E.; and Blunsom, P. 2017. Discovering Discrete Latent Topics with Neural Variational Inference. In *Proceedings of the 34th International Conference on Machine Learning*, 2410–2419.

Neal, R. M., and Hinton, G. E. 1998. A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants. In *Learning in Graphical Models*. Springer Netherlands. 355–368.

Rezende, D. J.; Eslami, S. M. A.; Mohamed, S.; Battaglia, P.; Jaderberg, M.; and Heess, N. 2016. Unsupervised Learning of 3D Structure from Images. In *Advances in Neural Information Processing Systems 29*, 4996–5004.

Rezende, D. J.; Mohamed, S.; and Wierstra, D. 2014. Stochastic Backpropagation and Approximate Inference in Deep Generative Models. In *Proceedings of the 31st International Conference on Machine Learning*, 1278–1286.

Semeniuta, S.; Severyn, A.; and Barth, E. 2017. A Hybrid Convolutional Variational Autoencoder for Text Generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 638–648.

Sønderby, C. K.; Raiko, T.; Maaløe, L.; Sønderby, S. K.; and Winther, O. 2016. Ladder Variational Autoencoders. In *Advances in Neural Information Processing Systems 29*, 3738–3746.

Theano Development Team. 2016. Theano: A Python framework for fast computation of mathematical expressions. *arXiv preprint* arXiv:1605.02688.

Wiseman, S., and Rush, A. M. 2016. Sequence-to-Sequence Learning as Beam-Search Optimization. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, 1296–1306.

Yang, Z.; Hu, Z.; Salakhutdinov, R.; and Berg-Kirkpatrick, T. 2017. Improved Variational Autoencoders for Text Modeling using Dilated Convolutions. In *Proceedings of the 34th International Conference on Machine Learning*, 3881–3890.

Zhu, Y.; Kiros, R.; Zemel, R.; Salakhutdinov, R.; Urtasun, R.; Torralba, A.; and Fidler, S. 2015. Aligning Books and Movies: Towards Story-like Visual Explanations by Watching Movies and Reading Books. *arXiv preprint* arXiv:1506.06724.

Appendix A SGVB

Stochastic Gradient Variational Bayes (SGVB) (Kingma and Welling 2014; Rezende, Mohamed, and Wierstra 2014) is a method for learning generative models with a joint density factorised as $p_\theta(\mathbf{x}, \mathbf{z}) = p_\theta(\mathbf{x}|\mathbf{z})p(\mathbf{z})$, where $\mathbf{x}$ is the vector of observations, $\mathbf{z}$ is a latent vector, and θ are the generative model parameters. The task is to learn the values of the generative parameters θ that maximise the log-likelihood of the observed data, *i.e.* $\max_\theta \log p_\theta(\mathbf{x})$.

For any density $q(\mathbf{z}|\mathbf{x})$, the evidence lower bound (ELBO) can be formed using Jensen's inequality:

$$\log p_\theta(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{z}|\mathbf{x})}\left[\log \frac{p_\theta(\mathbf{z}, \mathbf{x})}{q(\mathbf{z}|\mathbf{x})}\right] \equiv \mathcal{L}(\mathbf{x}) \tag{7}$$

The optimal setting for $q(\mathbf{z}|\mathbf{x})$ would be the true posterior $p(\mathbf{z}|\mathbf{x})$, however this is usually intractable. SGVB introduces the variational parameters ϕ which parametrise the distribution $q_\phi(\mathbf{z}|\mathbf{x})$. Monte Carlo sampling is used to approximate the expectation, and gradient steps are taken in the generative parameters θ and the variational parameters ϕ in order to optimise the bound.

Under certain mild conditions (Kingma and Welling 2014), the latent vector $\mathbf{z} \sim q_\phi(\mathbf{z}|\mathbf{x})$ can be reparametrised using a differentiable transformation $g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$, for some variable $\boldsymbol{\epsilon}$ such that $\mathbf{z} = g_\phi(\boldsymbol{\epsilon}, \mathbf{x})$ where $\boldsymbol{\epsilon} \sim p(\boldsymbol{\epsilon})$. The derivatives with respect to the parameters are then computed as follows, where $\boldsymbol{\epsilon}^{(s)} \sim p(\boldsymbol{\epsilon})$:

$$\nabla_{\theta,\phi}\mathcal{L}(\mathbf{x}) = \mathbb{E}_{p(\boldsymbol{\epsilon})}\left[\nabla_{\theta,\phi} \log \frac{p_\theta(g_\phi(\boldsymbol{\epsilon}, \mathbf{x}), \mathbf{x})}{q_\phi(g_\phi(\boldsymbol{\epsilon}, \mathbf{x}))}\right] \tag{8}$$

$$\simeq \frac{1}{S}\sum_{s=1}^{S} \nabla_{\theta,\phi} \log \frac{p_\theta(g_\phi(\boldsymbol{\epsilon}^{(s)}, \mathbf{x}), \mathbf{x})}{q_\phi(g_\phi(\boldsymbol{\epsilon}^{(s)}, \mathbf{x}))} \tag{9}$$