

Control, Generate, Augment: A Scalable Framework for Multi-Attribute Text Generation

Giuseppe Russo¹, Nora Hollenstein¹, Claudiu Musat², Ce Zhang¹

¹ ETH Zurich, {russog,noraho,ce.zhang}@inf.ethz.ch

² Swisscom, claudiu.musat@swisscom.com

Abstract

We introduce CGA, a conditional VAE architecture, to control, generate, and augment text. CGA is able to generate natural English sentences controlling multiple semantic and syntactic attributes by combining adversarial learning with a context-aware loss and a cyclical word dropout routine. We demonstrate the value of the individual model components in an ablation study. The scalability of our approach is ensured through a single discriminator, independently of the number of attributes. We show high quality, diversity and attribute control in the generated sentences through a series of automatic and human assessments. As the main application of our work, we test the potential of this new NLG model in a data augmentation scenario. In a downstream NLP task, the sentences generated by our CGA model show significant improvements over a strong baseline, and a classification performance often comparable to adding same amount of additional real data.

1 Introduction

Recently, natural language generation (NLG) has become a prominent research topic in NLP due to its diverse applications, ranging from machine translation (e.g., Sennrich et al. (2016)) to dialogue systems (e.g., Budzianowski and Vulić (2019)). The common goal of these applications using automatic text generation is the augmentation of datasets used for supervised NLP tasks. To this end, one of the key demands of NLG is *controlled* text generation, more specifically, the ability to systematically control semantic and syntactic aspects of generated text.

Most previous approaches simplify this problem by approximating NLG with the control of one single attribute of the text, such as sentiment or formality (e.g., Li et al. (2018), Fu et al. (2018), and John et al. (2019)). However, the problem of

controlled generation typically relies on multiple components such as lexical, syntactic, semantic and stylistic aspects. Therefore, the simultaneous control of multiple attributes becomes vital to generate natural sentences suitable for downstream tasks. Methods such as the ones presented by Hu et al. (2017) and Subramanian et al. (2018) succeed in simultaneously controlling various attributes. However, these methods depend on the transformation of input reference sentences, or do not scale easily to more than two attributes due to architectural complexities, such as the requirement for separate discriminators for each additional attribute.

In light of these challenges, we propose the Control, Generate, Augment framework (CGA), a powerful model to synthesize additional labeled data sampled from a latent space. The accurate multi-attribute control of our approach offers significant performance gains on downstream NLP tasks. We provide the code and all generated English sentences to facilitate future research¹.

The main contribution of this paper is a scalable model which learns to control **multiple semantic and syntactic attributes** of a sentence. The CGA model requires only a single discriminator for simultaneously controlling multiple attributes. To the best of our knowledge, we are the first to incorporate techniques such as cyclical word-dropout and a context-aware loss, which allow the CGA model to generate natural sentences given a latent representation and an attribute vector, *without requiring an input reference sentence during training*. We present automatic and human assessments to confirm the multi-attribute control and high quality of the generated sentences. Further, we provide a thorough comparison to previous work.

We use CGA as a natural language generation method for **data augmentation**, which boosts the

¹<https://github.com/DS3Lab/control-generate-augment>

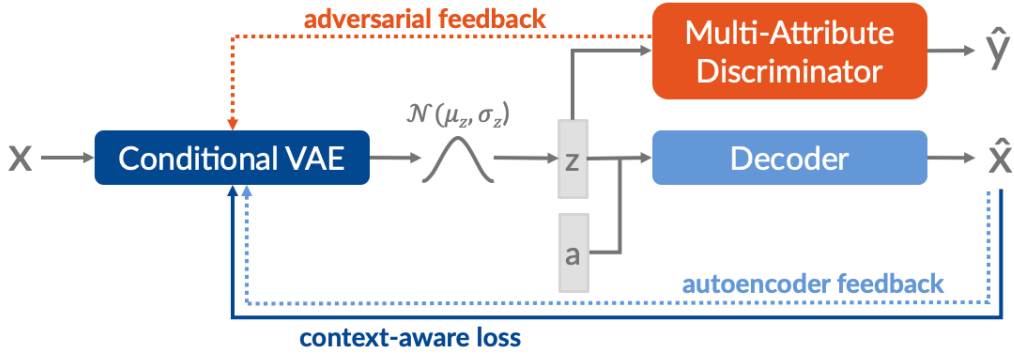


Figure 1: Model architecture depicting the key components of CGA.

performance of downstream tasks. We present data augmentation experiments on various English datasets, where we significantly outperform a strong baseline and achieve a performance often comparable to adding same amount of additional real data.

2 Method

We now present our model for controlled text generation. Our model is based on the Sentence-VAE framework by Bowman et al. (2016). However, we modify this model to allow the generation of sentences conditioned not only on the latent code but also on an attribute vector. We achieve this by disentangling the latent code from the attribute vector, in a similar way as the Fader networks (Lample et al., 2017), originally developed for computer vision tasks. As we will see, this simple adaptation is not sufficient, and we introduce further techniques to improve the multi-attribute sentence generation.

2.1 Model Architecture

We assume access to a corpus of sentences $\mathcal{X} = \{x_i\}_{i=1}^N$ and a set of K categorical attributes of interest. For each sentence x_i , we use an attribute vector a_i to represent these K associated attributes. Example attributes include the sentiment or verb tense of a sentence.

Given a latent representation z , which encodes the context information of the corpus and an attribute vector a , our goal is to construct a ML model which generates a new sentence x containing the attributes of a .

Sentence Variational Autoencoder The main component of our model is a Variational Auto-Encoder (Kingma and Welling, 2013). The encoder network $E_{\theta_{enc}}$, parameterized by a trainable parameter θ_{enc} , takes as input a sentence x and defines a

probabilistic distribution over the latent code z :

$$z \sim E_{\theta_{enc}}(x) := q_E(z|x; \theta_{enc}) \quad (1)$$

The decoder $G_{\theta_{dec}}$, parameterized by a trainable parameter θ_{dec} , tries to reconstruct the input sentence x from a latent code z and its attribute vector a . We assume that the reconstructed sentence \hat{x} has the same number of tokens as the input sentence x :

$$\begin{aligned} \hat{x} &\sim G_{\theta_{dec}}(z, a) := p_G(\hat{x}|z, a; \theta_{dec}) \\ &= \prod_{t=1}^T p_G(\hat{x}_{t+1}|\hat{x}_t, z, a; \theta_{dec}) \end{aligned} \quad (2)$$

where T is the length of the input sentence and \hat{x}_t is the t^{th} token. Here we use p_G to denote both sentence-level probability and word-level conditional probability.

To train the encoder and decoder, we use the following VAE loss:

$$L_{VAE}(\theta_{enc}, \theta_{dec}) := KL(q_E(z|x)||p(z)) - \mathbb{E}_{z \sim q_E(z|x)} \log p_G(x|z, a; \theta_{dec}), \quad (3)$$

where $p(z)$ is a standard Gaussian distribution.

When we try to optimize the loss in Equation 3, the KL term often vanishes. This problem is known in text generation as *posterior collapse* (Bowman et al., 2016). To mitigate this problem we follow Bowman et al. (2016) and add a weight λ_{kl} to the KL term in Equation 3. At the start of training, we set the weight to zero, so that the model learns to encode as much information in z as possible. Then, as training progresses, we gradually increase this weight, as in the standard KL-annealing technique.

Moreover, the posterior collapse problem occurs partially due to the fact that, during training, our decoder $G_{\theta_{dec}}$ predicts each token conditioned on the previous *ground-truth* token. We aim to make the model rely more on z . A natural way to achieve this is to weaken the decoder by removing some

or all of this conditional information during the training process. Previous works (Bowman et al., 2016; Hu et al., 2017) replace a — randomly selected — significant portion of the ground-truth tokens with UNKNOWN tokens. However, this can severely affect the decoder and deteriorate the generative capacity of the model. Therefore, we define a new word-dropout routine, which aims at both accommodating the posterior collapse problem and preserving the decoder capacity. Instead of fixing the word-dropout rate to a large constant value as in Bowman et al. (2016), we use a cyclical word-dropout rate ζ :

$$\zeta(s) = \begin{cases} k_{max} & s \leq \text{warm-up} \\ k_{max} & \left| \cos\left(\frac{2\pi}{\tau}s\right) \right| \geq k_{max} \\ k_{min} & \left| \cos\left(\frac{2\pi}{\tau}s\right) \right| \leq k_{min} \\ \left| \cos\left(\frac{2\pi}{\tau}s\right) \right| & \text{otherwise} \end{cases} \quad (4)$$

where s is the current training iteration, k_{max} and k_{min} are fixed constant values we define as upper and lower thresholds, and τ defines the period of the cyclical word-dropout rate schedule (see Suppl. Section A.2).

Disentangling Latent Code z and Attribute Vector a To be able to generate sentences given any attribute vector a' , we have to disentangle the attribute vector with the latent code. In other words, we seek that z is *attribute-invariant*: A latent code z is *attribute-invariant* if given two sentences x_1 and x_2 , they only differ in their attributes (e.g., two versions of the same review expressing opposite sentiment). Hence, they should result in the same latent representation $z = E_{\theta_{enc}}(x_1) = E_{\theta_{enc}}(x_2)$.

To achieve this, we use a concept from predictability minimization (Schmidhuber, 1992) and adversarial training for domain adaptation (Ganin et al., 2016; Louppe et al., 2017), which was recently applied in the Fader Networks by Lampl et al. (2017). We apply adversarial learning directly on the latent code z of the input sentence x . We set a min-max game and introduce a discriminator $D_{\theta_{disc}}(z)$, that takes as input the latent code and tries to predict the attribute vector a . Specifically, $D_{\theta_{disc}}(z)$ outputs for each attribute k , a probability distribution p_D^k over all its possible values. To train the discriminator, we optimize for the following loss:

$$L_{DISC}(\theta_{disc}) := -\log \prod_k p_D^k(a_k) \quad (5)$$

where a_k is the ground-truth of the k^{th} attribute.

Simultaneously, we hope to learn an encoder and decoder which (1) combined with the attribute vector a , allows the decoder to reconstruct the input sentence x , and (2) does not allow the discriminator to infer the correct attribute vector corresponding to x . We optimize for:

$$L_{ADV} := L_{VAE}(\theta_{enc}, \theta_{dec}) - \lambda_{Disc} L_{DISC}(\theta_{disc}) \quad (6)$$

Context-Aware Loss Equation 6 forces our model to choose which information the latent code z should retain or disregard. However, this approach comes with the risk of deteriorating the quality of the latent code itself. Therefore, inspired by Sanakoyeu et al. (2018), we propose an attribute-aware context loss, which tries to preserve the context information by comparing the sentence latent representation and its back-context representation:

$$L_{CTX} := \|E_{\theta_{enc}}(x) - E_{\theta_{enc}}(G_{\theta_{dec}}(E_{\theta_{enc}}(x)))\|_1 \quad (7)$$

We use a “stop-gradient” procedure, i.e., we compute the gradient w.r.t. $E_{\theta_{enc}}(x)$, which makes the function in Equation 7 differentiable.

The latent vector $z = E_{\theta_{enc}}(x)$ can be seen as a contextual representation of the input sentence x . This latent representation is changing during the training process and hence adapts to the attribute vector. Thus, when measuring the similarity between z and the back-context representation $E_{\theta_{enc}}(G_{\theta_{dec}}(E_{\theta_{enc}}(x)))$, we focus on preserving those aspects which are profoundly relevant for the context representation.

Finally, when training the encoder and decoder (given the current discriminator), we optimize for the following loss:

$$L_{CGA} := L_{VAE}(\theta_{enc}, \theta_{dec}) + \lambda_{CTX} L_{CTX} - \lambda_{Disc} L_{DISC}(\theta_{disc}) \quad (8)$$

3 Evaluation

To assess our newly proposed model for controlled sentence generation, we perform the following evaluations described in this section: An automatic and human evaluation to analyze the quality of the new sentences with multiple controlled attributes; an examination of sentence embedding similarity to assess the diversity of the generated samples; downstream classification experiments with data augmentation on two different datasets to prove the effectiveness of the new sentences in a pertinent

Sentence	Attributes
it was a great time to get the best in town and i loved it.	Past / Positive
it was a great time to get the food and it was delicious.	Past / Positive
it is a must!	Present/Positive
they're very reasonable and they are very friendly and helpful.	Present / Positive
i had agroupon and the service was horrible.	Past / Negative
this place was the worst experience i've ever had.	Past / Negative
it is not worth the money.	Present / Negative
there is no excuse to choose this place.	Present / Negative

Table 1: Examples of generated sentences with two attributes: SENTIMENT and VERB TENSE.

Sentence	Attributes
they have a great selection of beers and shakes.	Present / Positive / Plural
i love this place and i will continue to go here.	Present / Positive / Singular
the mashed potatoes were all delicious!	Past / Positive / Plural
the lady who answered was very friendly and helpful.	Past / Positive / Singular
the people are clueless.	Present / Negative / Plural
i mean i'm disappointed.	Present / Negative / Singular
drinks were cold and not very good.	Past / Negative / Plural
it was a complete disaster.	Past / Negative / Singular

Table 2: Examples of generated sentences with three attributes: SENTIMENT, VERB TENSE, and PERSON NUMBER.

	Sentiment	Tense	Person
YELP	91.1% (0.04)	96.6% (0.03)	95.9% (0.06)
IMDB	90.0% (0.06)	96.1% (0.04)	92.0% (0.05)

Table 3: Mean attribute matching accuracy of the 30K generated sentences (in %); standard deviation reported in brackets.

application scenario; and, finally, a comparison of our results to previous work to specifically contrast our model against other single and multi-attribute models.

Datasets We conduct all experiments on two datasets, YELP and IMDB reviews. Both contain sentiment labels for the reviews. From the YELP business reviews dataset (YELP, 2014), we use reviews only from the category *restaurants*, which results in a dataset of approx. 600'000 sentences. The IMDB movie reviews dataset (Maas et al., 2011) contains approx. 150'000 sentences. For reproducibility purposes, details about training splits and vocabulary sizes can be found in the supplementary materials (A.1.1).

Attributes For our experiments we use three attributes: sentiment as a semantic attribute; verb tense and person number as syntactic attributes.

SENTIMENT: We labeled each review as *positive* or *negative* following Shen et al. (2017).

VERB TENSE: We detect *past* and *present* verb tenses using SpaCy's part-of-speech tagging model². We define a sentence as *present* if it contains more present than past verbs. We provide the specific PoS tags used for the labeling in the supplementary materials (A.1.2).

PERSON NUMBER: We also use SpaCy to detect *singular* or *plural* pronouns and nouns. Consequently, we label a sentence as *singular* if it contains more singular than plural pronouns or nouns, we define it *plural*, in the opposite case, *balanced* otherwise.

We train our model to generate sentences of maximally 20 tokens by controlling one, two or three attributes simultaneously. The sentences are generated by the decoder as described in Equation 2. We chose to set the maximum sentence length to 20 tokens in this work, since (a) it is considerably more than previous approaches (e.g., Hu et al. (2017) presented a max length of 15 tokens), and (b) it covers more than the 99th percentile of the sentence lengths in the datasets used, which is 16.4 tokens per sentence for YELP and 14.0 for IMDB. In

²<https://spacy.io/usage/linguistic-features#pos-tagging>

Attribute	Sentences	Accuracy (κ)
Sentiment	106 / 120	0.88 (0.73)
Verb Tense	117 / 120	0.98 (0.97)
Person Number	114 / 120	0.95 (0.85)
2 Attributes	120 / 120	1.0
3 Attributes	97 / 120	0.80
Coherence	79 / 120	0.66

Table 4: Results of the human evaluation showing accuracy and Cohen’s κ for each attribute.

Table 1 we illustrate some examples of sentences where the model controls two attributes, SENTIMENT and VERB TENSE. Table 2 shows sentences where the model controls three attributes simultaneously. Sentences with single controlled attributes can be found in the supplementary material (A.4).

Experimental Setting The encoder and decoder are single-layer GRUs with a hidden dimension of 256 and maximum sample length of 20. The discriminator is a single-layer LSTM. To avoid a vanishingly small KL term in the VAE (Bowman et al., 2016), we use a KL term weight annealing that increases from 0 to 1 during training according to a logistic scheduling. λ_{disc} increases linearly from 0 to 20. Finally, we set the back-translation weight δ to 0.5. All hyper-parameters are provided in the supplementary material (A.2).

3.1 Quality of Generated Sentences

We quantitatively measure the sentence attribute control of our CGA model by inspecting the accuracy of generating sentences containing the designated attributes by conducting both automatic and human evaluations.

Attribute Matching For this automatic evaluation, we generate sentences given the attribute vector a as described in Section 2. To assign SENTIMENT attribute labels to the newly generated sentences, we apply a pre-trained TextCNN (Kim, 2014). To assign the VERB TENSE and PERSON NUMBER labels we use SpaCy’s part-of-speech tagging. We calculate the *attribute matching accuracy* as the percentage of the predictions of these pre-trained models on the generated sentences that match the attribute labels expected to be generated by our CGA model. Table 3 shows the averaged results over five balanced sets of 6000 sentences generated by CGA models, trained on YELP and IMDB, respectively.

Human Evaluation To further understand the quality of the generated sentences we go beyond the automatic attribute evaluation and perform a human judgement analysis. We provide all generated sentences including the human judgements³. One of our main contributions is the generation of sentences with up to three controlled attributes. Therefore, we randomly select 120 sentences generated by the CGA model trained on YELP, which controls all three attributes. Two human annotators labeled these sentences by marking which of the attributes are included correctly in the sentence.

In addition to the accuracy we report inter-annotator rates with Cohen’s κ . In 80% of the sentences all three attributes are included correctly and in 100% of the sentences at least two of the three attributes are present. Finally, the annotators also judged whether the sentences are grammatically correct, complete and coherent English sentences. Most of the incorrect sentences contain repeated words or incomplete endings. The results are shown in Table 4.

Ablation Study We conduct an ablation study testing the key components of the CGA model, both on the YELP and IMDB datasets. We separately trained four different versions of CGA to assess the impact on the multi-attribute control of the disjoint and joint usage of the *cyclical word-dropout* (Equation 4) and of the *context-aware loss* (Equation 7). We computed the attribute matching score following the same approach described above. As shown in Table 5, both techniques are beneficial for attribute control, especially for SENTIMENT and PERSON NUMBER. When the model is trained using at least one of these techniques it already shows significant improvements in all cases except for VERB TENSE on the IMDB data. Moreover, when the cyclical word-dropout and the context-aware loss are used jointly during training, the model experiences an increase of performance between 1-6% w.r.t. the model trained without using these techniques.

Sentence Embedding Similarity Although generative models have been shown to produce outstanding results, in many circumstances they risk producing extremely repetitive examples (e.g., Zhao et al. (2017)). In this experiment, we qualitatively assess the capacity of our model to gener-

³<https://github.com/DS3Lab/control-generate-augment>

Model	YELP			IMDB		
	Sentiment	Tense	Person	Sentiment	Tense	Person
$L_{ADV} + \text{standard WD}$	88.5%	95.5%	92.7%	84.1%	97.0%	86.3%
$L_{ADV} + \text{cyclical WD}$	89.7%	96.8%	93.1%	88.0%	93.6%	91.4%
$L_{CTX} + \text{standard WD}$	90.5%	95.6%	94.6%	89.2%	95.8%	87.9%
$L_{CTX} + \text{cyclical WD (CGA)}$	91.1%	96.6%	95.4%	90.0%	96.1%	92.0%

Table 5: Ablation Study of the key components, reporting attribute matching scores for three features on the YELP and IMDB datasets. $L_{ADV} + \text{standard WD}$ is trained with the word-dropout of Bowman et al. (2016); $L_{ADV} + \text{cyclical WD}$ is trained with our cyclical word-dropout; $L_{CTX} + \text{standard WD}$ is trained with the standard word-dropout and with context-aware loss; $L_{CTX} + \text{cyclical WD}$ is trained with both cyclical word-dropout and context-aware loss.

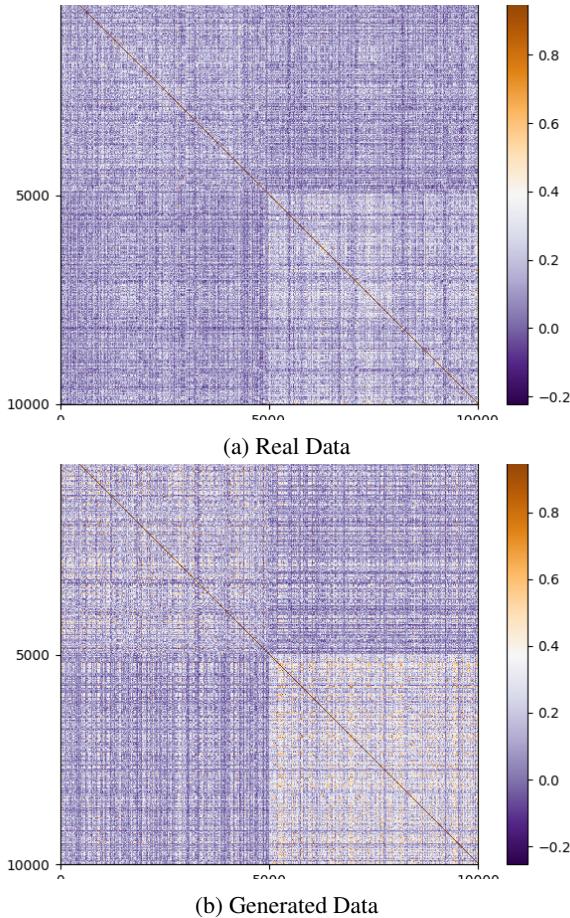


Figure 2: Similarity matrices for real data (M_{real}) and data generated by our CGA model controlling the sentiment attribute (M_{gen}).

ate diversified sentences to further strengthen the results obtained in this work. We sample 10K sentences from YELP (D_{real}) and from our generated sentences (D_{gen}), respectively, both labeled with the SENTIMENT attribute. We retrieve the sentence embedding for each of the sentences in D_{real} and D_{gen} using the Universal Sentence Encoder (Cer et al., 2018). Then, we compute the cosine simi-

ilarity between the embeddings of all sentences of D_{real} and, analogously, between the embeddings of our generated sentences D_{gen} .

Consequently, we obtain two similarity matrices M_{real} and M_{gen} (see Figure 2). Both matrices show a four cluster structure: **top-left** – similarity scores between negative reviews (C_{nn}); **top-right or bottom-left** – similarity scores between negative and positive reviews (C_{np}); and **bottom-right** – similarity scores between positive reviews (C_{pp}).

Further, for each sample of D_{real} and D_{gen} we compute a similarity score as follows:

$$\text{sim}(s_{i,c}) = \frac{1}{K} \sum_{x \in \mathcal{N}_{K,c}} \text{score}(s_i, x) \quad (9)$$

where $c \in \{C_{nn}, C_{np}, C_{pp}\}$. s_i is the i -th sample of D_{real} or D_{gen} and c is the cluster to which s_i belongs. $\mathcal{N}_{K,c}$ is the set of the k -most similar neighbours of s_i in cluster c , and $k=50$.

To gain a qualitative understanding of the generation capacities of our model, we assume that an ideal generative model should produce samples that have comparable similarity scores to the ones of the real data. Figure 3 contrasts the similarity scores of D_{real} and D_{gen} , computed on each cluster separately.

Although our generated sentences are clearly more similar between themselves than to the original ones, our model is able to produce samples clustered according to their labels. This highlights the good attribute control abilities of our CGA model and shows that it is able to generate diverse sentences which robustly mimic the structure of the original dataset. Hence, the generated sentences are good candidates for augmenting existing datasets.

We generalized this experiment for the multi-attribute case. The similarity matrices and the his-

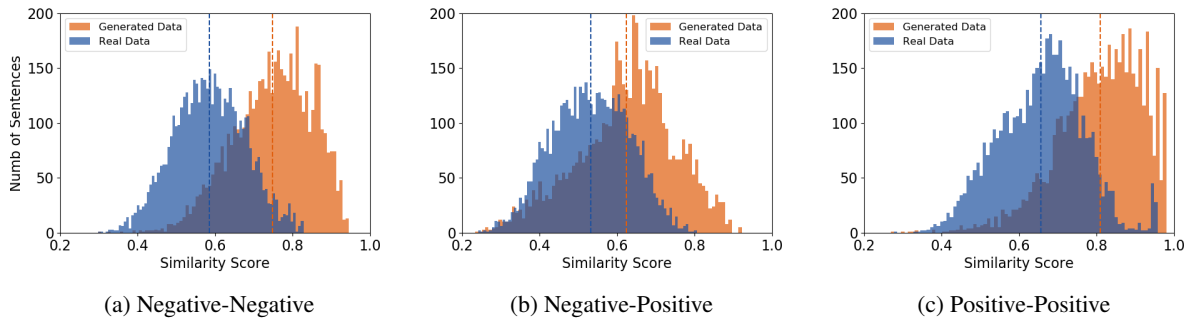


Figure 3: Sentence similarity scores computed for real data and data generated by our CGA model on the three sentiment clusters (Negative-Negative, Negative-Positive, Positive-Positive).

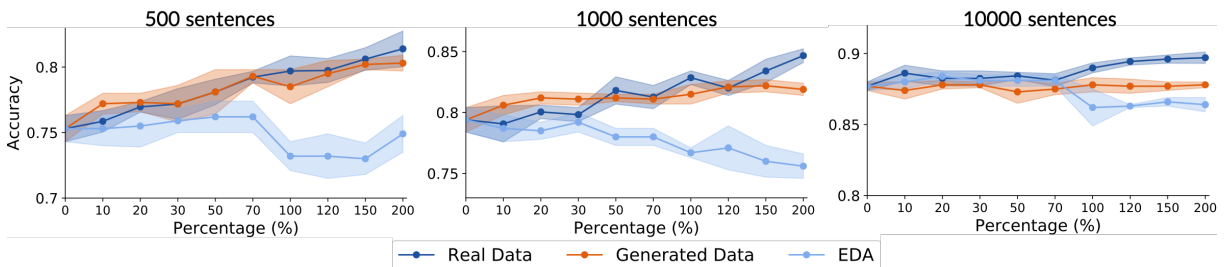


Figure 4: Data augmentation results for the YELP dataset.

tograms for these additional experiments are provided in the supplementary material (A.3.1).

3.2 Data Augmentation

The main application of our work is to generate sentences for data augmentation purposes. Simultaneously, the data augmentation experiments presented in this section reinforce the high quality of the sentences generated by our model.

As described, we conduct all experiments on two datasets, YELP and IMDB reviews. We train an LSTM sentiment classifier on both datasets, each with three different training set sizes. We run all experiments for training sets of 500, 1000 and 10000 sentences. These training sets are then augmented with different percentages of generated sentences (10, 20, 30, 50, 70, 100, 120, 150 and 200%). This allows us to analyze the effect of data augmentation on varying original training set sizes, as well as varying increments of additionally generated data. In all experiments we average the results over 5 random seeds and we report the corresponding standard deviation.

To evaluate how beneficial our generated sentences are for the performance of downstream tasks, we compare training sets augmented with sentences generated from our CGA model to (a) real sentences from the original datasets, and (b) sentences

generated with the Easy Data Augmentation (EDA) method by Wei and Zou (2019). EDA applies a transformation (e.g., synonym replacement or random deletion) to a given sentence of the training set and provides a strong baseline.

The results are presented in Figures 4 and 5, for YELP and IMDB, respectively. They show the performance of the classifiers augmented with sentences from our CGA model, from EDA and from the original datasets. Our augmentation method proved to be favorable in all six scenarios. Our model clearly outperforms EDA in all the possible scenarios, especially with larger augmentation percentages. The performance of the classifiers augmented with CGA sentences is equal to real data, and only begins to diverge when augmenting the training set with more than 100% of generated data.

In Table 6, we report the best average test accuracy as well as the percentage of data increment of real data, EDA and our CGA model for all three training set sizes and both datasets. Numerical results for all augmentation percentages including validation performance can be found in the supplementary materials (A.3.2).

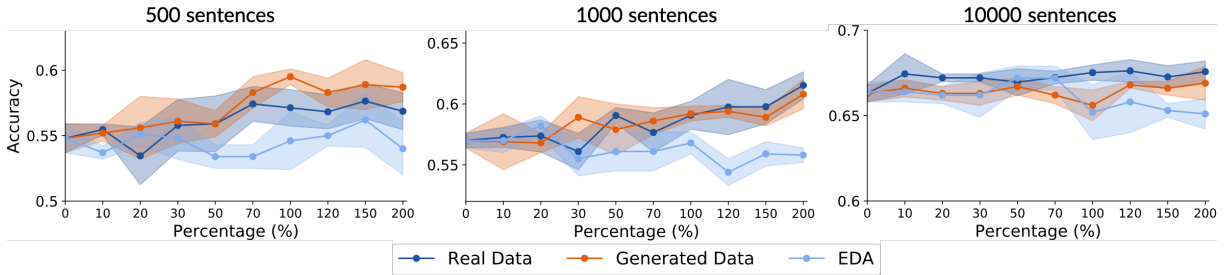


Figure 5: Data augmentation results for the IMDB dataset.

	Training Size					
	500 sentences		1000 sentences		10000 sentences	
Model	acc. (std)	%	acc. (std)	%	10000	%
Real Data YELP	0.75 (0.01)	0	0.79 (0.01)	0	0.87 (0.03)	0
YELP + EDA	0.77 (0.02)	70	0.80 (0.08)	30	0.88 (0.02)	70
YELP + CGA (Ours)	0.80 (0.02)	150	0.82 (0.03)	120	0.88 (0.04)	100
Real Data IMDB	0.54 (0.01)	0	0.57 (0.06)	0	0.66 (0.05)	0
IMDB + EDA	0.56 (0.02)	150	0.58 (0.07)	70	0.67 (0.02)	100
IMDB + CGA (Ours)	0.60 (0.01)	120	0.61 (0.01)	200	0.67 (0.03)	120

Table 6: Best performance for each method independent of the augmentation percentage used. For each method we report accuracy, standard deviation, and augmentation percentage.

4 Comparison to Previous Work

As a final analysis, we compare our results with previous state-of-the-art models for both single-attribute and multi-attribute control.

4.1 Single-Attribute Control

Li et al. (2018) model style control in the *Delete, Retrieve, Generate* framework, which deletes words related to a specific attribute and then inserts new words which belongs to the vocabulary of the target style (e.g., sentiment). Sudhakar et al. (2019) improve this framework by combining it with a transformer architecture (Vaswani et al., 2017). However, these approaches are susceptible to error, due to the difficulty of accurately selecting only the style-containing words.

Other approaches on text generation have leveraged adversarial learning. John et al. (2019) use a VAE with multi-task loss to learn a content and style representation that allows to elegantly control the sentiment of the generated sentences while preserving the content. Shen et al. (2017) train a cross-alignment auto-encoder (CAAE) with shared content and separate style distribution. Fu et al. (2018) suggested a multi-head decoder to generate sentences with different styles. As in our work, Shen et al. (2017) and Fu et al. (2018) do not en-

force content preservation.

These models are specifically designed to control this single attribute by approximating the style of a sentence with its sentiment. Shen et al. (2017) reported a sentiment matching accuracy of 83.5%. Both Fu et al. (2018) and John et al. (2019) achieve better sentiment matching accuracy (96% and 93.4%, respectively) in the automatic evaluation than our CGA model trained for a single attribute (93.1%). However, our CGA model obtains 96.3% in human evaluation, which is comparable with these works. Moreover, CGA offers a strong competitive advantage because it guarantees high sentiment matching accuracy while controlling additional attributes and, thus, offers major control over multiple stylistic aspects of a sentence.

4.2 Multi-Attribute Control

Few works have succeed in designing an adequate model for text generation and controlling multiple attributes. Hu et al. (2017) use a VAE with controllable attributes to generate short sentences with max. 15 tokens. Our CGA model improves upon this by generating sentences of high quality with a max. length of 20 tokens. This restricted sentence length is still one of the major limitations in NLG.

Subramanian et al. (2018) and Logeswaran et al. (2018) apply a back-translation technique from un-

supervised machine translation for style transfer tasks. Lai et al. (2019) follow the approach of the CAAE with a two-phase training procedure. Unlike our CGA model, these works enforce content preservation and require input reference sentences. Hence, it is not straight-forward to directly compare the results. However, their reported attribute matching accuracies for the SENTIMENT and VERB TENSE attributes are considerably lower than ours (91.1% and 96.6%, respectively). CGA also yields significantly better performance in the human evaluation. Recently, Wang et al. (2019) proposed an architecture for multi-attribute control. However, they focus merely on sentiment aspect attributes, while our CGA model is able to control both semantic and syntactic attributes.

These previous works reported content preservation as an additional evaluation metric. It is important to note that this metric is of no interest for our work, since, differently from these previous models, CGA generates sentences directly from an arbitrary hidden representations and it does not need a reference input sentence. Moreover, our CGA model is scalable to more attributes, while the previous architectures require multiple discriminators for controlling the attributes. Although we provide extensive evaluation analyses, it is still an open research question to define an appropriate evaluation metric for text generation to allow for neutral comparisons.

5 Conclusion

To the best of our knowledge, we propose the first framework for controlled natural language generation which (1) generates coherent sentences sampling from a smooth latent space, with multiple semantic and syntactic attributes; (2) works within a lean and scalable architecture, and (3) improves downstream tasks by synthesizing additional labeled data.

To sum up, our CGA model, which combines a context-aware loss function with a cyclical word-dropout routine, achieves state-of-the-art results with improved accuracy on sentiment, verb tense and person number attributes in automatic and human evaluations. Moreover, our experiments show that our CGA model can be used effectively as a data augmentation framework to boost the performance of downstream classifiers.

A thorough investigation of the quality of the attribute-invariant representation in terms of inde-

pendence between the context and the attribute vector will provide further insights. Additionally, a benchmark study of the maximum possible length of the generated sentences and the number of controllable attributes will deepen our understanding of the capabilities and limitations of CGA.

Acknowledgements

CZ and the DS3Lab gratefully acknowledge the support from the Swiss National Science Foundation (Project Number 200021_184628), Swiss Data Science Center, Alibaba, Cisco, eBay, Google Focused Research Awards, Oracle Labs, Swisscom, Zurich Insurance, Chinese Scholarship Council, and the Department of Computer Science at ETH Zurich.

References

- Samuel Bowman, Luke Vilnis, Oriol Vinyals, Andrew Dai, Rafal Jozefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*, pages 10–21.
- Paweł Budzianowski and Ivan Vulić. 2019. Hello, it’s gpt-2-how can i help you? towards the use of pre-trained language models for task-oriented dialogue systems. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*, pages 15–22.
- Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. 2018. Universal sentence encoder for English. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 169–174.
- Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Thirty-Second AAAI Conference on Artificial Intelligence*.
- Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. 2016. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030.
- Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 1587–1596. JMLR.
- Vineet John, Lili Mou, Hareesh Bahuleyan, and Olga Vechtomova. 2019. Disentangled representation

- learning for non-parallel text style transfer. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 424–434.
- Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1746–1751.
- Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
- Diederik P. Kingma and Max Welling. 2013. Auto-encoding variational Bayes. *arXiv preprint arXiv:1312.6114*.
- Chih-Te Lai, Yi-Te Hong, Hong-You Chen, Chi-Jen Lu, and Shou-De Lin. 2019. Multiple text style transfer by using word-level conditional generative adversarial network with two-phase training. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3570–3575.
- Guillaume Lample, Neil Zeghidour, Nicolas Usunier, Antoine Bordes, Ludovic Denoyer, and Marc’Aurelio Ranzato. 2017. Fader networks: Manipulating images by sliding attributes. In *Advances in Neural Information Processing Systems*, pages 5967–5976.
- Juncen Li, Robin Jia, He He, and Percy Liang. 2018. Delete, retrieve, generate: A simple approach to sentiment and style transfer. In *Proceedings of NAACL-HLT*, pages 1865–1874.
- Lajanugen Logeswaran, Honglak Lee, and Samy Bengio. 2018. Content preserving text generation with attribute controls. In *Advances in Neural Information Processing Systems*, pages 5103–5113.
- Gilles Louppe, Michael Kagan, and Kyle Cranmer. 2017. Learning to pivot with adversarial networks. In *Advances in neural information processing systems*, pages 981–990.
- Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. 2011. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150.
- Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.
- Artsiom Sanakoyeu, Dmytro Kotovenko, Sabine Lang, and Bjorn Ommer. 2018. A style-aware content loss for real-time HD style transfer. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 698–714.
- Jürgen Schmidhuber. 1992. Learning factorial codes by predictability minimization. *Neural Computation*, 4(6):863–879.
- Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40.
- Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in neural information processing systems*, pages 6830–6841.
- Sandeep Subramanian, Guillaume Lample, Eric Michael Smith, Ludovic Denoyer, Marc’Aurelio Ranzato, and Y-Lan Boureau. 2018. Multiple-attribute text style transfer. *arXiv preprint arXiv:1811.00552*.
- Akhilesh Sudhakar, Bhargav Upadhyay, and Arjun Maheswaran. 2019. “Transforming” Delete, Retrieve, Generate approach for controlled text style transfer. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3260–3270.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.
- Ke Wang, Hang Hua, and Xiaojun Wan. 2019. Controllable unsupervised text attribute transfer via editing entangled latent representation. In *Advances in Neural Information Processing Systems*, pages 11034–11044.
- Jason Wei and Kai Zou. 2019. EDA: Easy data augmentation techniques for boosting performance on text classification tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 6383–6389.
- Dataset YELP [online]. 2014. [[link](#)].
- Shengjia Zhao, Jiaming Song, and Stefano Ermon. 2017. InfoVAE: Information maximizing variational autoencoders. *arXiv preprint arXiv:1706.02262*.

A Supplementary Material

A.1 Data

A.1.1 Structure

We use YELP and IMDB for the training, validation and testing of our CGA models. The label distributions for all attributes are described in Table 7.

From the YELP business reviews dataset (YELP, 2014)⁴, we use reviews only from the category *restaurants*. We use the same splits for training, validation and testing as John et al. (2019), which contain 444101, 63483 and 126670, respectively. The vocabulary contains 9304 words.

We further evaluate our models on the IMDB dataset of movie reviews (Maas et al., 2011)⁵. We use reviews with less than 20 sentences and we select only sentences with less than 20 tokens. Our final dataset contains 122345, 12732, 21224 sentences for train validation and test, respectively. The vocabulary size is 15362 words.

A.1.2 Attribute Labeling

In this work we simultaneously control three attributes: SENTIMENT, VERB TENSE and PERSON NUMBER.

We use SpaCy’s Part-of-Speech tagging to assign the VERB TENSE labels. Specifically, we use the tags VBP and VBZ to identify present verbs, and the tag VBD to identify past verbs.

Analogously, we use the SpaCy’s PoS tags and the personal pronouns to assign PERSON NUMBER labels. In particular, we use the tag NN, which identifies singular nouns, and the following list of pronouns $\{i, he, she, it, myself\}$ to identify a singular sentence. We use NNS and the list of pronouns $\{we, they, themselves, ourselves\}$ to identify a plural sentence.

A.2 Training Details

All hyper-parameters were manually tuned. We report the tested ranges in square brackets.

VAE architecture Our VAE has one GRU encoder and one GRU decoder. The encoder has a hidden layer of 256 dimensions [64, 128, 256, 512], linearly transferred to the content vector of 32 dimensions (for one or two attributes), or 50

⁴Retrieved from <https://github.com/shentianxiao/language-style-transfer>

⁵Retrieved from <https://www.kaggle.com/lakshmi25npathi/imdb-dataset-of-50k-movie-reviews>

dimensions (for three attributes) [32, 50, 64, 128]. For training the decoder we set the initial hidden state as $h = \text{Linear}(z \oplus a)$. Moreover, we use teacher-forcing combined with the cyclical word-dropout described in Equation 4.

Discriminator The discriminator is used to create the attribute-invariant content vectors. We experimented with two architectures for the discriminator which held similar results. We tried a two-layer (64 dimensions each) fully-connected architecture with batch normalization; and a single-layer LSTM with 50 dimensions (for one or two attributes), or 64 dimensions (for three attributes).

KL-Annealing One of the challenges during the training process was the posterior collapse of the KL term. Similar to Bowman et al. (2016), we used a logistic KL annealing:

$$\lambda_{kl} = \frac{1}{1 + \exp(-K(x - x_0))} \quad (10)$$

where x is the current training step. x_0 indicates how many training steps are needed to set $\lambda_{kl} = 1$. K is a constant value given by:

$$K = -\frac{\log(-1 + \frac{1}{1-\epsilon})}{0.5 * \text{steps}} \quad (11)$$

We set $x_0 = 1000$ for YELP and $x_0 = 5000$ for IMDB. ϵ is a constant we set to 10^{-4} .

Discriminator Weight The interaction between the VAE and the Discriminator is a crucial factor for our model. Thus, we decide to linearly increase the discriminator weight λ_{disc} during the training process:

$$\lambda_{disc}(x) = \begin{cases} 0 & x \leq k_1 \\ \min(t, (t/(x_0)) * (x - k_1)) & \text{otherwise} \end{cases} \quad (12)$$

where t is the maximum value that λ_{disc} can have. x_0 indicates after how many training steps $\lambda_{disc} = t$. x is the current training step. k_1 is the warm-up value and it indicates after how many training steps the L_{disc} is included in L_{CGA} . We set $t = 20$, $x_0 = 6K$ and $k_1 = 12K$ for YELP or $x_0 = 3K$ and $k_1 = 5K$.

Word-Dropout We use Equation 4 with the following parameters: $\tau = 500$, $k_{min} = 0.3$, $k_{max} = 0.7$ and *warm-up threshold* = 2000.

	Sentiment		Tense		Person Number		
Dataset	Positive	Negative	Present	Past	Singular	Plural	Balanced
Yelp	393,237	241,017	304,441	329,813	190,276	317,127	126,851
IMDB	71,676	64,625	86,965	69,336	53,468	54,046	28,787

Table 7: YELP and IMDB dataset details showing the exact number of sentences used and the class distribution.

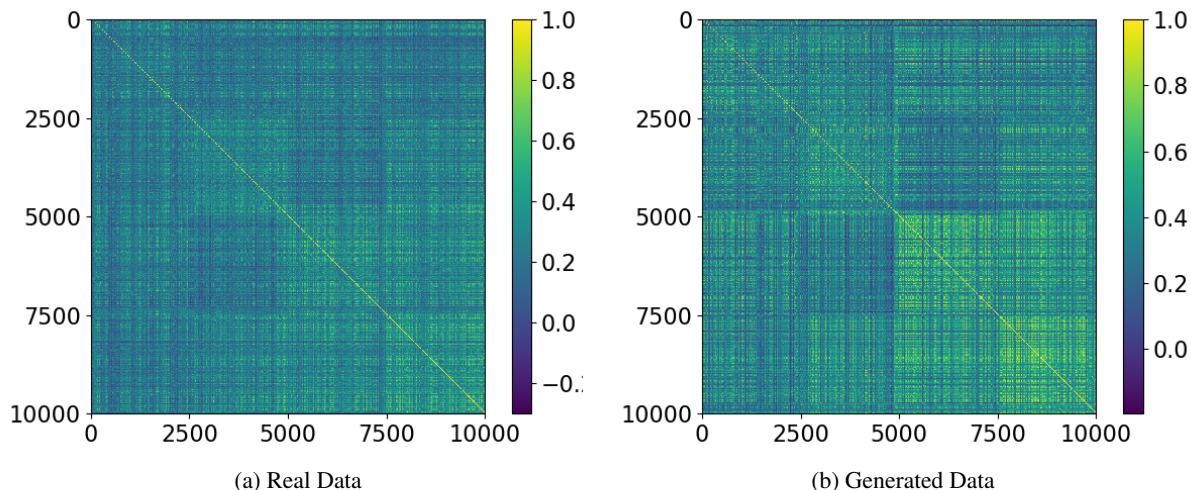


Figure 6: Similarity matrices for real data and data generated by our CGA model controlling the SENTIMENT and VERB TENSE attributes.

Optimizer The Adam optimizer, with initial learning rates of 10^{-3} [10^{-5} , 10^{-4} , 10^{-3} , 10^{-2} , 10^{-1}], was used for both the VAE and the discriminator (Kingma and Ba, 2014).

A.3 Evaluation

A.3.1 Sentence Embedding Similarity

Following the approach described in the main paper, we report the results of the sentence embedding similarities for the multi-attribute case (SENTIMENT and VERB TENSE). Similarly to the similarity matrices for the single-attribute case, in Figure 6 we recognize the clustered structure of the similarities. These matrices can be divide into the following clusters:

- **Intra-class Clusters:** These are the clusters which are placed over the diagonal of the matrices and show a high cosine similarity scores. They contain similarity scores between the embeddings of samples with the same labels.
- **Cross-Class Clusters:** These are the clusters located above the intra-class clusters. They contains the similarity scores between embeddings of samples with different labels. Indeed, they show lower similarity scores.

To gain a qualitative understanding of the generation capacities of our model, we assume that an ideal generative model should produce samples that have comparable similarity scores to the ones of the real data. We contrast the similarity scores computed on each cluster separately in the histograms in Figures 7 and 8.

A.3.2 Data Augmentation

For the data augmentation experiments we use a bidirectional LSTM with input size 300 and hidden size 256 [64, 128, 256, 512]. The input size is given by the 300 dimensional pre-trained GloVe embeddings (Pennington et al., 2014). We set dropout to 0.8 [0.5, 0.6, 0.7, 0.8]. For the training we use early stopping, specifically we stop the training process after 8 epochs without improving the validation loss.

Tables 8 and 9 show the detailed results for the data augmentation experiments on IMDB and YELP, respectively. The standard deviation of all results over 5 random seeds was always <0.025 .

Table 10 shows the corresponding performance on the validation set for the results on the test set presented in Table 6.

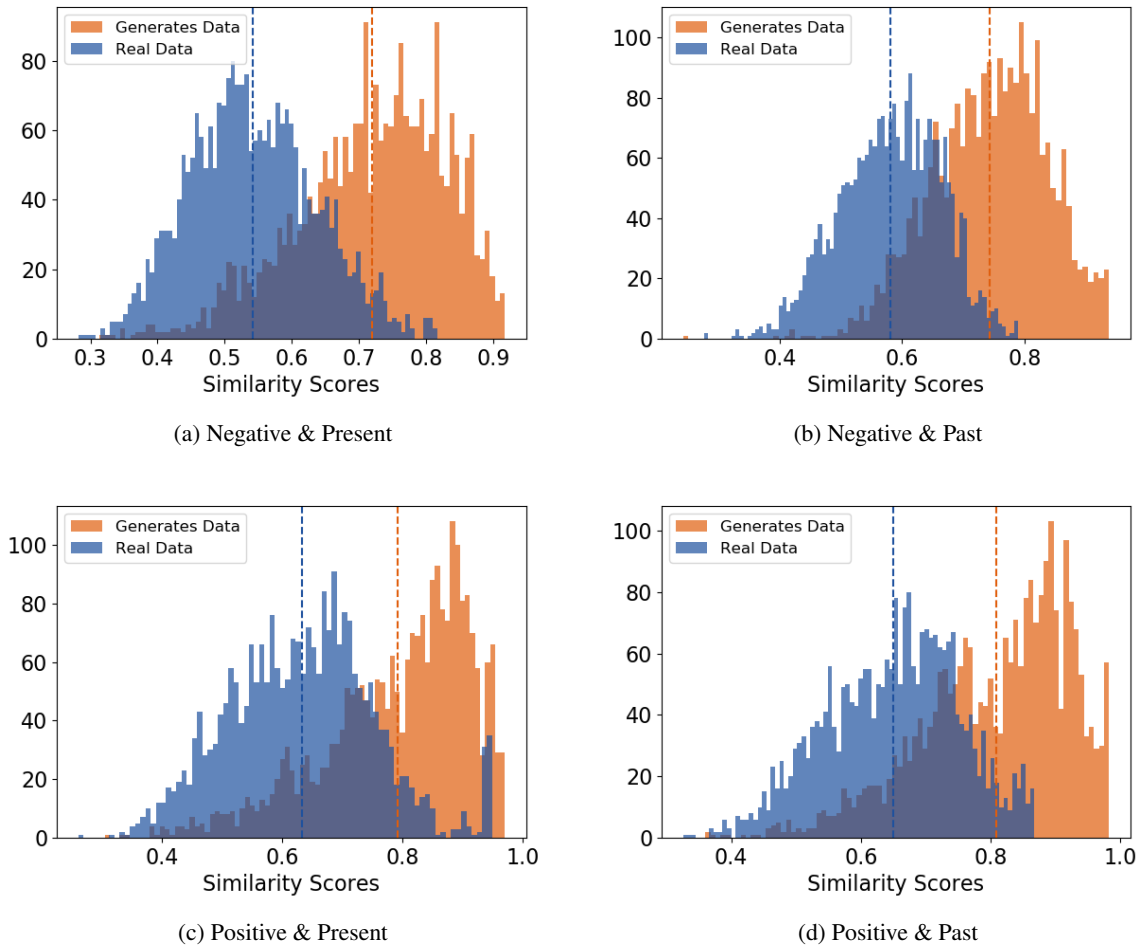


Figure 7: Sentence similarity scores computed for real data and data generated by our CGA model on the SENTIMENT and VERB TENSE clusters.

A.3.3 TextCNN

For the Attribute Matching results presented in Section 3 we use the pre-trained TextCNN (Kim, 2014). This network standardly uses 100 dimensional Glove word embeddings (Pennington et al., 2014), 3 convolutional layers with 100 filters each. The dropout rate is set to 0.5 during the training process.

A.4 Generated Sentences

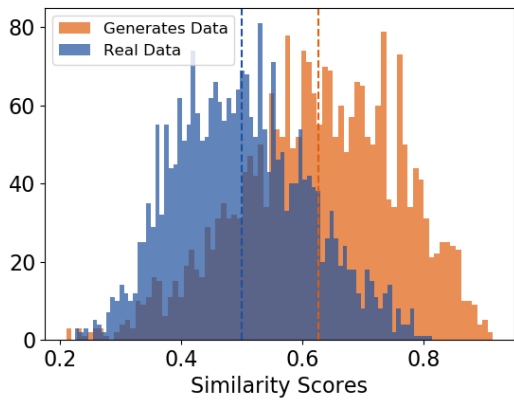
Tables 11 to 13 provide example sentences generated by the CGA model for the three individual attributes. Moreover, the code repository provides all generated sentences⁶.

A.5 Computing Infrastructure

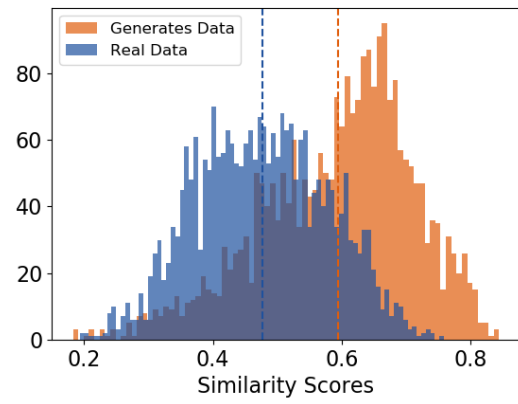
All models presented in this work were implemented in PyTorch, and trained and tested on single Titan XP GPUs with 12GB memory.

The average runtime was 07:26:14 for the model trained on YELP. The average runtime was 04:09:54 for the model trained on IMDB.

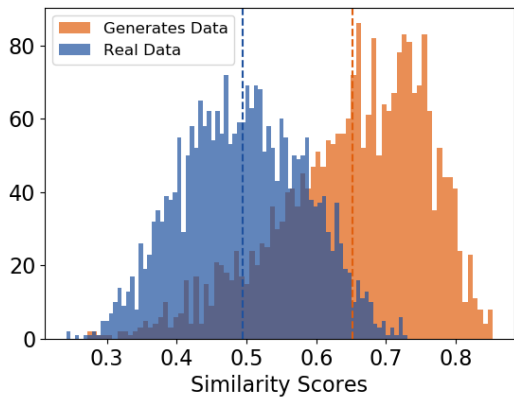
⁶Link omitted for review



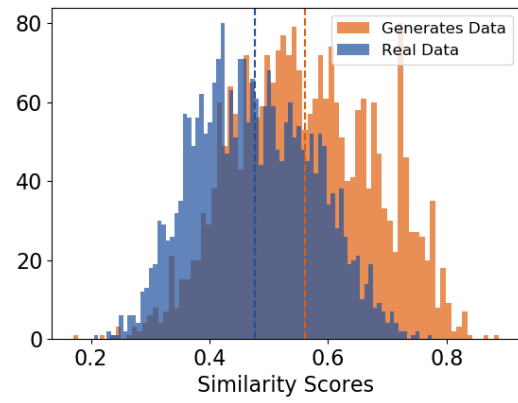
(a) Negative&Present-Positive&Present



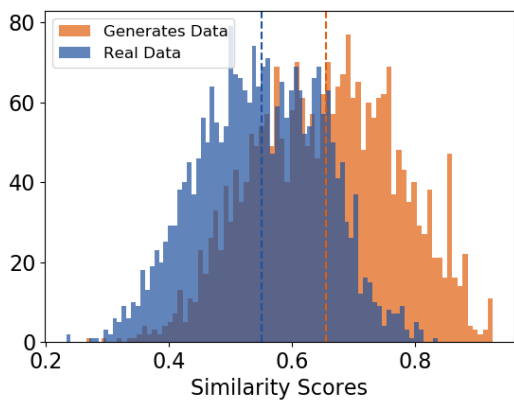
(b) Negative&Present-Positive&Past



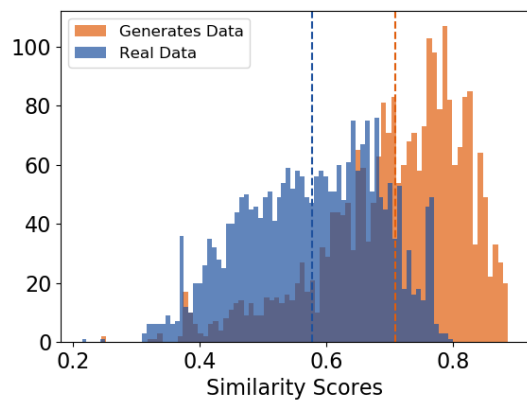
(c) Negative&Present-Negative&Past



(d) Negative&Past-Positive&Present



(e) Negative&Past-Positive&Past



(f) Positive&Present-Positive&Past

Figure 8: Sentence similarity scores computed for real data and data generated by our CGA model on the six multi-attribute clusters.

Augmentation (%)	0	10	20	30	50	70	100	120	150	200
500 sentences										
Real Data	0.753	0.759	0.770	0.772	0.781	0.792	0.797	0.797	0.806	0.814
Generated Data	-	0.772	0.773	0.772	0.781	0.793	0.785	0.795	0.802	0.803
EDA	-	0.753	0.755	0.759	0.762	0.762	0.732	0.732	0.730	0.749
1000 sentences										
Real Data	0.794	0.791	0.801	0.798	0.818	0.813	0.829	0.820	0.834	0.847
Generated Data	-	0.806	0.812	0.811	0.812	0.811	0.815	0.821	0.822	0.819
EDA	-	0.787	0.785	0.792	0.780	0.780	0.767	0.771	0.760	0.756
10000 sentences										
Real Data	0.877	0.886	0.883	0.883	0.884	0.881	0.890	0.894	0.896	0.897
Generated Data	-	0.874	0.878	0.878	0.873	0.875	0.878	0.877	0.877	0.878
EDA	-	0.880	0.884	0.881	0.881	0.881	0.862	0.863	0.866	0.864

Table 8: Detailed accuracy numbers for YELP data augmentation results presented in Figure 4.

Augmentation (%)	0	10	20	30	50	70	100	120	150	200
500 sentences										
Real Data	0.548	0.555	0.535	0.558	0.559	0.574	0.571	0.568	0.576	0.569
Generated Data	-	0.552	0.556	0.561	0.559	0.583	0.595	0.583	0.589	0.587
EDA	-	0.537	0.551	0.548	0.534	0.534	0.546	0.550	0.562	0.540
1000 sentences										
Real Data	0.570	0.573	0.574	0.561	0.591	0.577	0.591	0.598	0.598	0.615
Generated Data	-	0.569	0.568	0.589	0.579	0.586	0.592	0.594	0.589	0.608
EDA	-	0.570	0.582	0.555	0.561	0.561	0.568	0.544	0.559	0.558
10000 sentences										
Real Data	0.663	0.674	0.672	0.672	0.670	0.672	0.675	0.676	0.673	0.676
Generated Data	-	0.666	0.663	0.663	0.667	0.662	0.656	0.668	0.666	0.669
EDA	-	0.664	0.662	0.662	0.672	0.672	0.652	0.658	0.653	0.651

Table 9: Detailed accuracy numbers for IMDB data augmentation results presented in Figure 5.

	Training Size					
	500 sentences		1000 sentences		10000 sentences	
Model	acc. (std)	%	acc. (std)	%	acc. (std)	%
Real Data YELP	0.79 (0.05)	0	0.77 (0.04)	0	0.86 (0.04)	0
YELP + EDA	0.77 (0.09)	70	0.82 (0.04)	70	0.87 (0.03)	70
YELP + CGA (Ours)	0.81 (0.03)	150	0.84 (0.03)	120	0.85 (0.07)	100
Real Data IMDB	0.55 (0.03)	0	0.57 (0.04)	0	0.65 (0.09)	0
IMDB + EDA	0.56 (0.04)	150	0.58 (0.06)	70	0.69 (0.05)	100
IMDB + CGA (Ours)	0.62 (0.05)	120	0.59 (0.03)	200	0.68 (0.02)	120

Table 10: Performance on the validation set for the data augmentation results reported in Table 6.

Sentence	Sentiment
but i'm very impressed with the food and the service is great.	Positive
i love this place for the best sushi!	Positive
it is a great place to get a quick bite and a great price.	Positive
it's fresh and the food was good and reasonably priced.	Positive
not even a good deal.	Negative
so i ordered the chicken and it was very disappointing.	Negative
by far the worst hotel i have ever had in the life.	Negative
the staff was very rude and unorganized.	Negative

Table 11: Examples of generated sentences controlling the SENTIMENT attribute.

Sentence	Tense
i love the fact that they have a great selection of wines.	Present
they also have the best desserts ever.	Present
the food is good , but it's not worth the wait for it.	Present
management is rude and doesn't care about their patients.	Present
my family and i had a great time.	Past
when i walked in the door , i was robbed.	Past
had the best burger i've ever had.	Past
my husband and i enjoyed the food.	Past

Table 12: Examples of generated sentences controlling the VERB TENSE attribute.

Sentence	Person
it was a little pricey but i ordered the chicken teriyaki.	Singular
she was a great stylist and she was a sweetheart.	Singular
worst customer service i've ever been to.	Singular
this is a nice guy who cares about the customer service.	Singular
they were very friendly and eager to help.	Plural
these guys are awesome!	Plural
the people working there were so friendly and we were very nice.	Plural
we stayed here for NUM nights and we will definitely be back.	Plural

Table 13: Examples of generated sentences controlling the PERSON NUMBER attribute.