Learning Implicit Text Generation via Feature Matching

Inkit Padhi, Pierre Dognin, Ke Bai[‡], Cicero Nogueira dos Santos[†*]
Vijil Chenthamarakshan, Youssef Mroueh, Payel Das
IBM Research, [‡]Duke University, [†]Amazon AWS AI
inkpad@ibm.com, ke.bai@duke.edu, cicnog@amazon.com
{pdognin,ecvijil,mroueh,daspa}@us.ibm.com

Abstract

Generative feature matching network (GFMN) is an approach for training implicit generative models for images by performing moment matching on features from pre-trained neural networks. In this paper, we present new GFMN formulations that are effective for sequential data. Our experimental results show the effectiveness of the proposed method, SeqGFMN, for three distinct generation tasks in English: unconditional text generation, class-conditional text generation, and unsupervised text style transfer. SeqGFMN is stable to train and outperforms various adversarial approaches for text generation and text style transfer.

1 Introduction

Generative feature matching networks (GFMNs) (dos Santos et al., 2019) has been recently proposed for learning implicit generative models by performing moment matching on features from pre-trained neural networks. This approach demonstrated that GFMN could produce state-of-the-art image generators while avoiding instabilities associated with adversarial learning. Similarly to training generative adversarial networks (GANs) (Goodfellow et al., 2014), GFMN training requires to backpropagate through the generated data to update the model parameters. This backpropagation through the generated data, combined with adversarial learning instabilities, has proven to be a compelling challenge when applying GANs for discrete data such as text. However, it remains unknown if this is also an issue for feature matching networks since the effectiveness of GFMN for sequential discrete data has not yet been studied.

In this work, we investigate the effectiveness of GFMN for different text generation tasks. As a

first contribution, we propose a new formulation of GFMN for unconditional sequence generation, which we name *Sequence-GFMN* or *SeqGFMN* for short, by performing token level feature matching. SeqGFMN has a stable training because it does not concurrently train a discriminator, which in principle could easily learn to distinguish between one-hot and soft one-hot representations. As a result, we can use soft one-hot representations that the generator outputs during training without using the Gumbel softmax or REINFORCE algorithm as needed in GANs for text. Additionally, different from GANs (Zhu et al., 2018), SeqGFMN can produce meaningful text without the need of pre-training the generator with maximum likelihood estimation (MLE). We perform experiments using Bidirectional Encoder Representations from Transformers (BERT), GloVe, and FastText as our feature extractor networks. We use two different corpora, and assess both the quality and diversity of the generated texts with three different quantitative metrics: BLEU, Self-BLEU and Fréchet Infersent Distance (FID). Additionally, we show that the *latent space* induced by SeqGFMN contains semantic and syntactic structure, as evidenced by interpolations in the z space.

Our **second contribution** consists in proposing a new strategy for class-conditional generation with GFMN. The key idea here is to perform class-wise feature matching. We apply SeqGFMN to perform sentiment-based conditional generation using the Yelp Reviews dataset, and assess its performance using classification accuracy, BLEU, and Self-BLEU.

Finally, as a **third contribution**, we demonstrate that the feature matching loss is an effective approach to perform distribution matching in the context of unsupervised text style transfer (UTST). Most previous work on UTST adapts the autoencoder framework by adding an additional

---

*work done prior to joining Amazon

loss term: adversarial loss or back-translation loss. Our method consists in replacing the adversarial and back-translation loss with style-wise feature matching. Our experimental results indicate that the feature matching loss produces better results than the traditionally used losses.

2 Feature Matching Nets for Text

2.1 SeqGFMN

Let G be a sequence generator implemented as a neural network with parameters θ, and let E be a pretrained NLP feature extractor network with L hidden layers, that produces features at token-level for each token in a sequence of length T. The method consists of training G by minimizing the following token-level feature matching loss function:

$$\min_\theta \sum_{t=1}^{T} \sum_{j=1}^{M} ||\mu_{p_{data}}^{j,t} - \mu_{p_G}^{j,t}(\theta)||^2 + ||\sigma_{p_{data}}^{j,t} - \sigma_{p_G}^{j,t}(\theta)||^2 \quad (1)$$

where:

$$\mu_{p_{data}}^{j,t} = \mathbb{E}_{x \sim p_{data}} E_{j,t}(x) \in \mathbb{R}^{d_j},$$
$$\mu_{p_G}^{j,t}(\theta) = \mathbb{E}_{z \sim \mathcal{N}(0,I_{n_z})} E_{j,t}(G(z;\theta)) \in \mathbb{R}^{d_j},$$
$$\sigma_{p_{data},\ell}^{j,t} = \mathbb{E}_{x \sim p_{data}} E_{j,\ell,t}(x)^2 - [\mu_{p_{data}}^{j,\ell,t}]^2,$$
$$\sigma_{p_G,\ell}^{j,t}(\theta) = \mathbb{E}_{z \sim \mathcal{N}(0,I_{n_z})} E_{j,\ell,t}(G(z;\theta))^2 - [\mu_{p_G}^{j,\ell,t}]^2,$$
$$\ell = 1 \ldots d_j,$$

where $||.||^2$ is the L_2 loss; x is a real data point sampled from the data distribution p_{data}; $z \in \mathbb{R}^{n_z}$ is a noise vector sampled from the normal distribution $\mathcal{N}(0, I_{n_z})$; $E_{j,t}(x)$ denotes the token-level t feature map at a hidden layer j from E; $M \leq L$ is the number of hidden layers used to perform feature matching; T is the maximum sequence length; and $\sigma_{p_{data}}^2$ and $\sigma_{p_G}^2$ are the variances of the features for real data and generated data respectively. Note that this loss function is quite different from both the MLE loss used in regular language models and the adversarial loss used in GANs.

In order to train G, we first precompute $\mu_{p_{data}}^{j,t}$ and $\sigma_{p_{data},\ell}^{j,t}$ on the entire training data. During training, we generate a minibatch of *fake* data by passing the Gaussian noise vector through the generator. The fixed feature extractor E is used to extract features on the output of the generator at a per-token level. The loss is then computed, as mentioned in Eq. 1. The parameters θ of the generator G are optimized using stochastic gradient descent. Note that the network E is used for feature extraction only and is kept fixed during the training of G. Similar to (dos Santos et al., 2019), we use ADAM moving average, which allows us to use small minibatch sizes. Fig. 1 illustrates SeqGFMN training; note that we use mean matching only for brevity, in practice we match both mean and diagonal covariance.

In our SeqGFMN framework, the output of the generator G is a sequence $\tilde{x}$ of *soft one-hot representations*, $\{\tilde{w}_1, \tilde{w}_2, ..., \tilde{w}_T\}$, where each element $\tilde{w}_i$ consists in the output of the softmax function at token i. In the feature extractor E, these soft one-hot representations are multiplied by an embedding matrix to generate *soft embeddings*, which are then fed to the following layers of E.

2.2 Class-Conditional SeqGFMN

Conditional generation is motivated by the assumption that if the training data can be clustered into distinct and meaningful classes, knowledge of such classes at training time would improve the overall performance of the model. For class-based text generation, some datasets provide such opportunity by labeling the training data with relevant classes (e.g., positive/negative sentiment for Yelp Reviews dataset), information that can be leveraged by our model to condition the generation.

For this to be effective, the extracted features used for SeqGFMN need to be sufficiently representative of the text generated yet still be different between classes. To account for the knowledge of latent classes, we extend the loss from Eq.1 for the case of two distinct classes:

$$\min_\theta \sum_{t=1}^{T} \sum_{j=1}^{M} ||\delta_{c=0}^{j,t}||^2 + ||\Delta_{c=0}^{j,t}||^2 + $$
$$||\delta_{c=1}^{j,t}||^2 + ||\Delta_{c=1}^{j,t}||^2 \quad (2)$$

where $\delta_c^{j,t} = \mu_{p_{data}^c}^{j,t} - \mu_{p_G^c}^{j,t}(\theta)$ and $\Delta_c^{j,t} = \sigma_{p_{data}^c}^{j,t} - \sigma_{p_G^c}^{j,t}(\theta)$ follows the same definition for means and variances as Eq.1, with the exception that they are now class-dependent. Given a class c, we allow for conditional generation by conditioning the noise vector z on c. Indeed, if $z \sim \mathcal{N}(0, I_{n_z})$, applying a class dependent linear transformation $z_c = A_c z + b_c$ will change the noise distribution such that $z_c \sim \mathcal{N}(b_c, A_c^\top A_c)$. A_c and b_c are learned at training time so to minimize our loss. This enables the model to effectively sample

$$\mathcal{L} = \sum_{t=1}^{T} \sum_{j=1}^{M} \|\boldsymbol{\mu}_{\boldsymbol{p_{data}}}^{\boldsymbol{j,t}} - \frac{1}{N} \sum_{i=1}^{N} E_{j,t}(\tilde{x}_i)\|^2$$
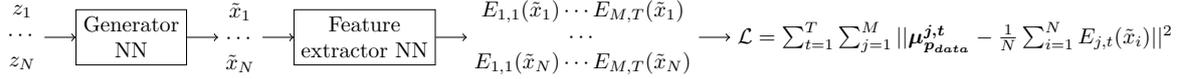
Figure 1: For each training iteration, Generator (G) outputs N sentences from noise signals $z_1 \cdots z_N$. A fixed feature extractor is used to extract token level features ($E_{j,t}$) for the generated data. $\mathcal{L}$ is the L_2-norm of the difference between extracted features means of generated and real data $\boldsymbol{\mu}_{\boldsymbol{p_{data}}}^{\boldsymbol{j,t}}$, which is then backpropagted to update the parameters of G. The same strategy is used for variance terms in $\mathcal{L}$ (here ignored for brevity).

a new input noise from distinct distributions, conditioned on the class c. Since the model can update the linear transformation parameters A_c and b_c to minimize its loss, the model can learn transformations that separate or disentangle between the different classes c naturally. For example, conditioning on sentiment where $c = 0$ is the negative sentiment class and $c = 1$ the positive class, amounts simply to learning two transformations (A_0, b_0) and (A_1, b_1). This approach can be extended beyond learning linear transformations to allow for deep neural network to be employed. During training, a minibatch is composed of input noise samples conditioned on class c. Within our generator, we use a conditional batch normalization (condBN) from (Dumoulin et al., 2016). The conditional BN is a 2-stage process: First, we perform a standard BN of a minibatch regardless of c where $y_i = \text{BN}_{\gamma,\beta}(x_i)$, using notations from (Ioffe and Szegedy, 2015). Then y_i enters a second stage where $w_i = \gamma_c y_i + \beta_c$ brings class dependency on c as proposed in (Dumoulin et al., 2016). This allows for the influence of class conditioning to carry over the whole model where conditional BN is used. Our models can have three distinct configurations: conditional input noise, conditional BN, or both conditional input noise and conditional BN.

2.3 Unsupervised Text Style Transfer (UTST) with SeqGFMN

Text style transfer consists of rewriting a sentence from a given style s_i (e.g., informal) into a different style s_j (e.g., formal) while maintaining the content and keeping the sentence fluent. The major challenge for this task is the lack of parallel data, and many recent approaches adapt the encoder-decoder framework to work with non-parallel data (Shen et al., 2017; Fu et al., 2018). This adaptation normally consists in using: (1) the reconstruction loss in an autoencoding fashion, which is intended to learn a conditional language model (decoder D) while providing content preservation; together with (2) a classification loss produced by a style

classifier C, which is intended to guarantee the correct transfer. Balancing these two losses while generating good quality sentences is difficult, and several approaches such as adversarial discriminators (Shen et al., 2017) and cycle-consistency loss (Melnyk et al., 2017) have been employed in recent works. Here, we use feature matching as a way to alleviate this problem. Essentially, our unsupervised text style transfer approach is an encoder-decoder trained with the following three losses:

Reconstruction loss: Given an input sentence x^{s_i} from set X and its decoded sentence $\hat{x}^{s_i} = D(E(x^{s_i}), s_i)$ (decoded in the same input style s_i), the reconstruction loss measures how well the decoder D is able to reconstruct it:

$$\mathcal{L}_{\text{rec}} = \mathbb{E}_{x^{s_i} \sim X} \left[- \log p_D(x^{s_i} | E(x^{s_i}), s_i) \right]. \quad (3)$$

Classification loss: This loss is formulated as :

$$\begin{aligned} \mathcal{L}_{\text{class}} = &\mathbb{E}_{x^{s_i} \sim X} \left[- \log p_C(s_i | x^{s_i}) \right] + \\ &\mathbb{E}_{\hat{x}^{s_i \to s_j} \sim \hat{X}} \left[- \log p_C(s_j | \hat{x}^{s_i \to s_j}) \right]. \quad (4) \end{aligned}$$

where $\hat{X}$ is the set of style transferred sentences generated by the current model. For the classifier, the first term provides supervised signal regarding style classification and the second term gives additional training signal from the transferred data, enabling the classifier to be trained in a semi-supervised regime. For the encoder-decoder the second term gives feedback on the current generator's effectiveness on transferring sentences to a different style.

Feature Matching loss: It is computed in a similar way as the class-conditional loss (Eq. 2). This loss consists of matching statistics of the features for each style separately. This means that when transferring from style s_i to s_j, we match the features of the resulting sentence with the features of real data that are from the target style s_j.

3 Related work

(Zhang et al., 2017a) proposes Adversarial Feature Matching for Text Generation by adding a reconstruction feature loss to the GAN objective. This

is different from our setup, as our discriminator is not learned, and our feature matching is per token and not on a global sentence level. Sequence GAN (SeqGAN) (Yu et al., 2017), MaliGAN (Che et al., 2017), and RankGAN (Lin et al., 2017) use a pre-trained generator with MLE loss with a per token reward discriminator that is trained with reinforcement learning. SeqGFMN is similar to SeqGAN in the sense that it has a per token reward (per token feature matching loss). Still, it alleviates the need for pre-training the generator and the cumbersome training of a discriminator by relying on a fixed, state-of-the-art, text feature extractor such as BERT. Due to the discrete nature of the problem, training implicit models is tricky (de Masson d'Autume et al., 2019), which is addressed by using REIN-FORCE, actor-critic methods (Fedus et al., 2018), and Gumbel softmax trick(Kusner and Hernández-Lobato, 2016).

For unsupervised text style transfer, different adaptations of the encoder-decoder framework have been proposed recently. (Shen et al., 2017; Fu et al., 2018) uses adversarial classifiers to decode to a different style/language. (Melnyk et al., 2017),(Nogueira dos Santos et al., 2018) proposed a method that combines a collaborative classifier with the back-transfer loss. (Prabhumoye et al., 2018) presented an approach that trains different encoders, one per style, by combining the encoder of a pre-trained NMT and style classifiers. The main difference between our approach and these previous work consists in the fact that we use the feature matching loss to perform distribution matching.

4 Experiments and Results

Datasets: We evaluate our proposed approach on three different english datasets: MSCOCO (Lin et al., 2014), EMNLP 2017 WMT News dataset (Bojar et al., 2017), and Yelp Reviews Dataset (Shen et al., 2017). Both COCO and WMT News datasets are used for unconditional models, while Yelp Reviews is employed to evaluate class-conditional generation and unsupervised text style transfer.

Feature Extractors for Textual Data: We experiment with different feature extractors that generate token-level representations. We use word embeddings from GloVe (Pennington et al., 2014) and FastText (Bojanowski et al., 2017) as representatives of shallow (cheap-to-train) architectures. As

a representative of large, deep feature extractor we use BERT (Devlin et al., 2018). Devlin et al. (2018) demonstrated that the features extracted by BERT can boost the performance of diverse NLP tasks. Our hypothesis is that BERT features are informative enough to allow the training of (cross-domain) text generators with the help of feature matching.

Metrics: In order to evaluate the diversity and quality of texts of the unconditional generators we use three metrics *BLEU* (Papineni et al., 2002), *Self-BLEU*(Zhu et al., 2018) and *Fréchet Infersent Distance, FID*(Heusel et al., 2017). Additionally, for class-conditional generation and unsupervised text style transfer, we report accuracy scores from a CNN sentiment classifier trained on the Yelp.

4.1 Experimental Results

Unconditional Text Generation: In Tab. 1, we show quantitative results for SeqGFMN trained on COCO and WMT News using different feature extractors. As expected, BERT as a feature extractor gives better performance because of a more significant and richer features used.

We also present a comparison with other implicit generative models for text generation from scratch. We compare SeqGFMN with five different GAN approaches: SeqGAN (Yu et al., 2017), MaliGAN (Che et al., 2017), RankGAN (Lin et al., 2017), TextGAN (Zhang et al., 2017a) and Rel-GAN (Weili Nie and Patel, 2019). We do not use generator pre-training for any of the models. As reported in Tab. 1, SeqGFMN outperforms all GAN models in terms of BLEU and FID. The combination of low BLEU and low Self-BLEU for the different GANs indicates that the learned models generate random n-grams that do not appear in the test set. All GANs fail to learn reasonable models due to the challenges of learning a discrete data generator from scratch under the min-max game. Whereas, SeqGFMN can learn suitable generators without the need of generator pre-training.

Class-conditional Generation: Conditional generation experiments were conducted on Yelp Reviews dataset with sentiment labels (178K negative, 268K positive). For this experiment, we first pre-trained the Generator using a conditional denoising AE where class labels are provided only to the decoder D. The architecture of the encoder is the same as in (Zhang et al., 2017b) with three strided convolutional layers. Once pre-trained, D is used as initialization for our Generator G. The training is similar

	Model	BLEU-2	BLEU-3	BLEU-4	BLEU-5	Self-BLEU	FID
	Real Data	0.721	0.494	0.308	0.194	0.487	3.559
	SeqGAN	0.044	0.019	0.012	0.010	0.026	13.167
	MaliGAN	0.042	0.017	0.011	0.008	0.032	15.855
COCO	RankGAN	0.039	0.016	0.010	0.008	**0.023**	15.502
	TextGAN	0.034	0.015	0.010	0.008	0.624	17.275
	RelGAN	0.230	0.055	0.026	0.017	0.811	13.948
	SeqGFMN (FastText)	0.389	0.153	0.089	0.059	0.644	6.371
	SeqGFMN (Glove)	0.403	0.139	0.077	0.053	0.655	6.218
	SeqGFMN (BERT)	**0.695**	**0.476**	**0.277**	**0.186**	0.802	**5.610**
	Real Data	0.852	0.596	0.356	0.199	0.289	0.365
	SeqGAN	0.008	0.004	0.003	0.003	0.088	8.731
	MaliGAN	0.070	0.021	0.012	0.008	**0.018**	9.057
WMT News	RankGAN	0.188	0.055	0.024	0.015	0.973	12.306
	TextGAN	0.053	0.018	0.010	0.008	0.644	9.945
	RelGAN	0.076	0.026	0.015	0.012	0.451	8.809
	SeqGFMN (FastText)	0.364	0.102	0.045	0.028	0.787	3.761
	SeqGFMN (Glove)	0.385	0.106	0.047	0.029	0.735	4.033
	SeqGFMN (BERT)	**0.760**	**0.464**	**0.204**	**0.096**	0.888	**3.530**

Table 1: Quantitative results for different implicit generators trained from scratch.

to the previous section except now sentiment class labels are passed to G, and class-dependent statistics of BERT features are used, as described in 2.2.

Model	Accu.	Class	BLEU3	Self-BLEU3
Baseline	-	-	**0.415**	0.509
Conditional	**0.746**	0	**0.473**	0.498
Noise+BN		1	**0.413**	0.472
Cond. BN	0.745	0	0.423	0.473
		1	0.395	0.505
Cond. Noise	0.495	0	0.413	**0.458**
		1	0.412	**0.470**

Table 2: Comparison between Sentiment-dependent and class-agnostic (unconditional) SeqGFMN models.

Tab. 2 presents results for our regular model (baseline) and the three conditional generators: Cond. Noise, Cond. Batch Normalization (BN), Cond. Noise+BN. We use 10K generated sentences for each sentiment class to compute classification accuracy. In terms of accuracy and BLEU-3 score, the Cond. Noise+BN model provides the best generator as it is able to capture and leverage the class information.

Unsupervised Text Style Transfer (UTST): In Table 3, we report BLEU and accuracy scores for SeqGFMN and six baselines: BackTranslation (Prabhumoye et al., 2018), which uses back-transfer loss; CrossAligned (Shen et al., 2017), MultiDecoder (Fu et al., 2018), and StyleEmbedding (Fu et al., 2018), which use adversarial loss; and TemplateBased (Li et al., 2018) and Del-Retrieval (Li et al., 2018), which uses rule-based methods. The BLEU score is computed between the transferred

sentences and the human-annotated transferred references, similar to (Li et al., 2018). And, the accuracy is based on our pre-trained classifier. Compared to the other models, SeqGFMN produces the best balance between BLEU and accuracy. Additionally, if we use back-transfer loss together with feature matching loss (*SeqGFMN + BT*) our model gets a significant improvement on both metrics.

Model	BLEU	Accuracy
BackTranslation	2.5	95.7
CrossAligned	9.1	74.1
MultiDecoder	14.6	50.1
StyleEmbedding	21.1	9.2
TemplateBased	22.6	81.1
Del-Retrieval	16.0	88.2
SeqGFMN	23.7	92.9
SeqGFMN + BT	**24.5**	**96.4**

Table 3: Comparison between SeqGFMN and other models for unsupervised text style transfer.

5 Conclusion

We presented new implicit generative models based on feature matching loss that are suitable for unconditional and conditional text generation. Our results demonstrated that backpropagating through discrete data is not an issue for the training via matching distributions at the token level. SeqGFMN can be trained from scratch without the need for RL or Gumbel Softmax. This approach has allowed us to create effective models for unconditional generation, class-conditional generation, and unsupervised text style transfer. We believe this work opens a new competitive avenue in the area of implicit generative models for sequential data.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Ondrej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, Christof Monz, Matteo Negri, Matt Post, Raphael Rubino, Lucia Specia, and Marco Turchi. 2017. Findings of the 2017 conference on machine translation (WMT17). In *Proceedings of the Second Conference on Machine Translation, WMT 2017, Copenhagen, Denmark, September 7-8, 2017*, pages 169–214.

Tong Che, Yanran Li, Ruixiang Zhang, R. Devon Hjelm, Wenjie Li, Yangqiu Song, and Yoshua Bengio. 2017. Maximum-likelihood augmented discrete generative adversarial networks. *CoRR*, abs/1702.07983.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. 2016. A learned representation for artistic style. *CoRR*, abs/1610.07629.

William Fedus, Ian Goodfellow, and Andrew M. Dai. 2018. MaskGAN: Better text generation via filling in the _____. In *International Conference on Learning Representations*.

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. In *Proceedings of AAAI*.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Proc. of NIPS*, page 2672.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, Günter Klambauer, and Sepp Hochreiter. 2017. Gans trained by a two time-scale update rule converge to a nash equilibrium. *CoRR*, abs/1706.08500.

Sergey Ioffe and Christian Szegedy. 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *CoRR*, abs/1502.03167.

Matt J. Kusner and José Miguel Hernández-Lobato. 2016. GANS for sequences of discrete elements with the gumbel-softmax distribution. *CoRR*, abs/1611.04051.

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

Kevin Lin, Dianqi Li, Xiaodong He, Zhengyou Zhang, and Ming-ting Sun. 2017. Adversarial ranking for language generation. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 3155–3165. Curran Associates, Inc.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. 2014. Microsoft coco: Common objects in context. In *Computer Vision – ECCV 2014*, pages 740–755, Cham. Springer International Publishing.

Cyprien de Masson d'Autume, Shakir Mohamed, Mihaela Rosca, and Jack Rae. 2019. Training language gans from scratch. In *Advances in Neural Information Processing Systems*, pages 4302–4313.

Igor Melnyk, Cicero Nogueira dos Santos, Kahini Wadhawan, Inkit Padhi, and Abhishek Kumar. 2017. Improved neural text attribute transfer with non-parallel data. In *NIPS Workshop on Learning Disentangled Representations: from Perception to Control*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 311–318, Stroudsburg, PA, USA. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher 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, Doha, Qatar. Association for Computational Linguistics.

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

Cicero Nogueira dos Santos, Igor Melnyk, and Inkit Padhi. 2018. Fighting offensive language on social media with unsupervised text style transfer. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 189–194, Melbourne, Australia. Association for Computational Linguistics.

Cicero Nogueira dos Santos, Youssef Mroueh, Inkit Padhi, and Pierre Dognin. 2019. Learning implicit generative models by matching perceptual features. In *International Conference on Computer Vision (ICCV)*.

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 30*, pages 6830–6841.

Nina Narodytska Weili Nie and Ankit Patel. 2019. Relgan: Relational generative adversarial networks for text generation. In *ICLR*.

Lantao Yu, Weinan Zhang, Jun Wang, and Yong Yu. 2017. Seqgan: Sequence generative adversarial nets with policy gradient. In *AAAI*, pages 2852–2858.

Yizhe Zhang, Zhe Gan, Kai Fan, Zhi Chen, Ricardo Henao, Dinghan Shen, and Lawrence Carin. 2017a. Adversarial feature matching for text generation. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 4006–4015.

Yizhe Zhang, Dinghan Shen, Guoyin Wang, Zhe Gan, Ricardo Henao, and Lawrence Carin. 2017b. Deconvolutional paragraph representation learning. In *NIPS*, pages 4172–4182.

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

Appendices

A   Experimental Setup

SeqGFMN Generator: We use a deconvolutional generator that extends the decoder architecture proposed in (Zhang et al., 2017). It consists of three strided deconvolutional layers followed by cosine similarity between the *generated* token embeddings and an embedding matrix. Our adaptations are as follows: (1) we added two convolutional layers after the second deconvolution; (2) we added a self-attention layer before the last deconvolutional layer; (3) we added a convolutional layer after the last deconvolutional layer; (4) after the final convolution, we multiply the resulting token embeddings by the embedding matrix and apply the softmax function to generate a probability distribution over the vocabulary. We use the embedding matrix from BERT model and this matrix is not updated during the training of seqGFMN. The number of convolutional filters used is 400 with kernel size of 5.

SeqGFMN Training: SeqGFMNs are trained with an ADAM optimizer for which most hyper-parameters are kept fixed across datasets. We use $n_z = 100$ and minibatch size of 128. We use learning rates of 10^{-4} and 10^{-3} for updating G, and ADAM Moving Averages (AMA), respectively. The generator is trained for about 100K iterations.

Feature Extractor Details: In the experiments with GloVe and FastText, we used their default 300 dimension vectors pre-trained on 6 billion tokens from Wikipedia 2014 & Gigaword 5, and English Wikipedia, respectively. In the experiments with BERT, we use $BERT_{BASE}$ model, which contains 12 layers and produces 768 features per token per layer. When using a maximum sequence of 32, that leads to a total 294,912 features.

B   Unconditional Text Generation

An interesting comparison would be between SeqGFMN and GANs that use BERT as a pre-trained discriminator. However, GANs fail to train when a very deep network is used as the discriminator Moreover, SeqGFMN also outperforms GAN generators even when shallow word embeddings (Glove / FastText) are used to perform feature matching. Pretrained word embeddings are normally used in GANs for text.

In Tab. 4, we present randomly selected samples that were generated by SeqGFMN and RelGAN. These samples corroborate the quantitative results

and show that SeqGFMN can generate good text when trained from scratch. At the same time, the state-of-the-art method RelGAN is unable to generate reasonable text without pretraining.

C   Class-Conditional Generation

In Tab. 5, we present cherry-picked examples of generated text. Interestingly, since our input noise z is transformed according to sentiment c, we implicitly have a pairing between z_0 and z_1. Text generated from z_0 and z_1 are related to the same z. The effect of this implicit pairing can be seen in the examples where sentences seem somehow related, but of the opposite sentiment. Qualitatively, conditional SeqGFMN models can leverage class information to improve generation.

In Table 6, we present samples of original and sentiment transferred sentences. For each original sentence, we show the reference transferred sentence from the test set (done by a human) and the sentence that was transferred by SeqGFMN. Similar to other recently proposed UTST methods, the most successful cases of sentiment transfer are the ones where the transfer can be done by removing and replacing a few words of the sentence. In Table 6, the last example of each block are cases where SeqGFMN does not do a good job when significant changes in the original sentence are required to perform a more fluent sentiment transfer.

D   Unsupervised Text Style Transfer

The baselines are calculated with the data collected by (Luo et al., 2019) [1] and using Unsupervised NMT methods (Zhang et al., 2018).

E   Interpolation

We interpolate in the latent space of SeqGFMN z and check whether the sentences generated by the interpolation are syntactically and/or semantically related. In detail, we sample two vectors z_0 and z_1 from the prior distribution p_z and build intermediate points $z_\lambda = \lambda z_1 + (1 - \lambda)z_0$. In Tab. 7, we show samples from two interpolations, on models trained on COCO and WMT news dataset. In both these cases, we notice that there exists some syntactic and/or semantic relationship between the sentences along the interpolating path. This is supporting evidence that the latent space induced by SeqGFMN is meaningful, and related sentences are close together in this latent space.

---

[1] https://github.com/luofuli/DualRL/

Model	COCO
SeqGFMN	a 747 aircraft plane flying on a runway . a kitchen with a kitchen sink and a microwave on the counters . a bike flag showcasing a person sitting near a street sign . a bathroom with a toilet on the counter .
RelGAN	fry up on a nuts cargo black tonic rocks kept cruising basket adorable graveyard . border itl washer table a an green with bmw suit heater down . his pushed docked sofas wave messy nursing , triple black school a continue plane siking bbq pickup . quadruple several lots a loft buckets vines a bullhorn the appliances sidewalk sidewalk . uniforms
Model	**WMT News**
SeqGFMN	the ban did nothing but say voters were illegally investing their time at college and to take on your calls at [CONT.] court , " ross . announced . in addition , 32 typical economies in this period are reportedly pledged to have trillion pledged in another [CONT.] time , typically , tens to millions in million in feed .
RelGAN	should should children about about about states . inquiry matthew his s a about am . . appeal only over a ve about found .

Table 4: Randomly sampled sentences from generators trained from scratch on COCO and WMT News datasets.

Positive Sentiment generated z_1	Negative Sentiment generated from z_0
full of good food	everything is bad food
love this place	avoid this place
good job	horrible !
just perfect because my entire menu was fabulous	completely upset with the salon
everything is good !	disgusting
the service staff is extremely welcoming - and my mom loved it	the salon itself is very poor , and my mom admitted it

Table 5: Sentences generated using conditional SeqGFMN trained on Yelp Reviews dataset.

Positive Sentiment (Original)	Negative Sentiment (Transferred)
place was clean and well kept , drinks were reasonably priced .	place was dirty and drinks were expensive and watered down . (GT) place was dirty and horribly kept , drinks were horribly priced . (SeqGFMN)
food is very fresh and amazing !	food was old and stale . (GT) food was ridiculous , too . (SeqGFMN)
this place reminds me of home !	this place reminds me why i want to go home . (GT) this jerk reminds me of trash . (SeqGFMN)
Negative Sentiment (Original)	**Positive Sentiment (Transferred)**
the decor was seriously lacking .	the decor was nice . (GT) the decor was superb . (SeqGFMN)
now the food : not horrible , but below average .	now the food : not bad , above average . (GT) now the food is fantastic ! (SeqGFMN)
i wish i could give less than one star .	i wish there were more stars to give . (GT) i love getting them ! (SeqGFMN)

Table 6: Examples of sentiment transferred texts using SeqGFMN. **(GT)** = ground truth produced by a human.

COCO
a group of people sleeps in the street
a group of people standing in the street
a toy of people warming a street sidewalk
an automobile car lies on an short parking road
an automobile car lies on an green parking road
an automobile car lies on an green bike field
the automobile car lies on an green parking field
the automobile car is on an green parking field
WMT News
"although that might do nothing -i admit it- and i've invested time time at work," i tend to say it doesn do nothing.
"although the odds do it -i get it- and ross hasn always conceded his chance at it," i tend to say our odds are there.
reportedly upon the call to court, i get it, while romney has promised that his ban did nothing but say voters had better announce...
reportedly upon the call at court and i get it, while voters didn ##rem realize the ban was there.
the said pledge would take on one another day, sexually claiming to top the worst in your period at the academy.
the us has to feed two-thirds in one month, typically in the best ##quest best ##gist at the in & in millions in.
this will cover two-thirds billion trillion in this period, possibly two-thirds - 63 0 in one months.
in addition, regulators selected millions in one years, potentially billions in another decade, possibly the bottom-profile economies ...

Table 7: Interpolation in the latent space z of SeqGFMN models trained on COCO Captions and WMT News.