Generalization in Generation: A closer look at Exposure Bias

Florian Schmidt
Department of Computer Science
ETH Zürich
florian.schmidt@inf.ethz.ch

Abstract

Exposure bias refers to the train-test discrepancy that seemingly arises when an autoregressive generative model uses only ground-truth contexts at training time but generated ones at test time. We separate the contributions of the model and the learning framework to clarify the debate on consequences and review proposed counter-measures.
In this light, we argue that generalization is the underlying property to address and propose unconditional generation as its fundamental benchmark. Finally, we combine latent variable modeling with a recent formulation of exploration in reinforcement learning to obtain a rigorous handling of true and generated contexts. Results on language modeling and variational sentence auto-encoding confirm the model's generalization capability.

1 Introduction

Autoregressive models span from n-gram models to recurrent neural networks to transformers and have formed the backbone of state-of-the-art machine learning models over the last decade on virtually any generative task in Natural Language Processing. Applications include machine translation (Bahdanau et al., 2015; Vaswani et al., 2017), summarization (Rush et al., 2015; Khandelwal et al., 2019), dialogue (Serban et al., 2016) and sentence compression (Filippova et al., 2015).

The training methodology of such models is rooted in the language modeling task, which is to predict a single word given a context of previous words. It has often been criticized that this setting is not suited for multi-step generation where – at test time – we are interested in generating words given a *generated* context that was potentially not seen during training. The consequences of this train-test discrepancy are summarized as *exposure bias*. Measures to mitigate the problem typically rely on replacing, masking or pertubing ground-truth contexts (Bengio et al., 2015; Bowman et al., 2016; Norouzi et al., 2016; Ranzato et al., 2016). Unfortunately, exposure bias has never been succesfully separated from general test-time log-likelihood assessment and minor improvements on the latter are used as the only signifier of reduced bias. Whenever explicit effects are investigated, no significant findings are made (He et al., 2019).

In this work we argue that the standard training procedure, despite all criticism, is an immediate consequence of combining autoregressive modeling and maximum-likelihood training. As such, the paramount consideration for improving test-time performance is simply regularization for better generalization. In fact, many proposed measures against exposure bias can be seen as exactly that, yet with respect to an usually implicit metric that is not maximum-likelihood.

With this in mind, we discuss regularization for conditional and unconditional generation. We note that in conditional tasks, such as translation, it is usually sufficient to regularize the *mapping* task – here translation – rather than the generative process itself. For unconditional generation, where tradeoffs between accuracy and coverage are key, generalization becomes much more tangible.

The debate on the right training procedure for autoregressive models has recently been amplified by the advent of *latent* generative models (Rezende et al., 2014; Kingma and Welling, 2013). Here, the practice of decoding with true contexts during training conflicts with the hope of obtaining a latent representation that encodes significant information about the sequence (Bowman et al., 2016). Interestingly, the ad hoc tricks to reduce the problem are similar to those proposed to ad-

Proceedings of the 3rd Workshop on Neural Generation and Translation (WNGT 2019), pages 157–167
Hong Kong, China, November 4, 2019.
www.aclweb.org/anthology/D19-56%2d

dress exposure bias in deterministic models.

Very recently, Tan et al. (2017) have presented a reinforcement learning formulation of exploration that allows following the intuition that an autoregressive model should not only be trained on ground-truth contexts. We combine their framework with latent variable modeling and a reward function that leverages modern word-embeddings. The result is a single learning regime for unconditional generation in a deterministic setting (language modeling) and in a latent variable setting (variational sentence autoencoding). Empirical results show that our formulation allows for better generalization than existing methods proposed to address exposure bias. Even more, we find the resulting regularization to also improve generalization under log-likelihood.

We conclude that it is worthwhile exploring reinforcement learning to elegantly extend maximum-likelihood learning where our desired notion of generalization cannot be expressed without violating the underlying principles. As a result, we hope to provide a more unified view on the training methodologies of autoregressive models and exposure bias in particular.

2 Autoregressive Modeling

Modern text generation methods are rooted in models trained on the language modeling task. In essence, a *language model* p is trained to predict a word given its left-side context

$$p(w_t|w_{1:t-1}) \ . \tag{1}$$

With a trained language model at hand, a simple recurrent procedure allows to generate text of arbitrary length. Starting from an initial special symbol $\hat{w}_0$, we iterate $t = 1 \ldots$ and alternate between sampling $\hat{w}_t \sim p(w_t|\hat{w}_{1:t-1})$ and appending $\hat{w}_t$ to the context $\hat{w}_{1:t-1}$. Models of this form are called *autoregressive* as they condition new predictions on old predictions.

Neural Sequence Models Although a large corpus provides an abundance of word-context pairs to train on, the cardinality of the context space makes explicit estimates of (1) infeasible. Therefore, traditional n-gram language models rely on a truncated context and smoothing techniques to generalize well to unseen contexts.

Neural language models lift the context restriction and instead use neural context representations. This can be a hidden state as found in recurrent neural networks (RNNs), i.e. an LSTM (Hochreiter and Schmidhuber, 1997) state, or a set of attention weights, as in a transformer architecture (Vaswani et al., 2017). While the considerations in this work apply to all autoregressive models, we focus on recurrent networks which encode the context in a fixed-sized continuous representation $\mathbf{h}(w_{1:t-1})$. In contrast to transformers, RNNs can be generalized easily to variational autoencoders with a single latent bottleneck (Bowman et al., 2016), a particularly interesting special case of generative models .

2.1 Evaluation and Generalization

Conditional vs. Unconditional
Conditional generation tasks, such as translation or summarization, are attractive from an application perspective. However, for the purpose of studying exposure bias, we argue that unconditional generation is the task of choice for the following reasons.

First, exposure bias addresses conditioning on past words *generated* which becomes less essential when words in a source sentence are available, in particular when attention is used.

Second, the difficulty of the underlying mapping task, say translation, is of no concern for the mechanics of generation. This casts sentence autoencoding as a less demanding, yet more economic task.

Finally, generalization of conditional models is only studied with respect to the underlying mapping and not with respect to the conditional distribution itself. A test-set in translation usually does not contain a source sentence seen during training with a *different* target[1]. Instead, it contains unseen source-target pairs that evaluate the generalization of the mapping. Even more, at test-time most conditional models resort to an arg-max decoding strategy. As a consequence, the entropy of the generative model is zero (given the source) and there is no generalization at all with respect to generation. For these reasons, we address unconditional generation and sentence auto-encoding for the rest of this work.

The big picture Let us briefly characterize output we should expect from a generative model with respect to generalization. Figure 1 shows

[1]Some datasets do provide several targets for a single source. However, those are typically only used for BLEU computation, which is the standard test metric reported.

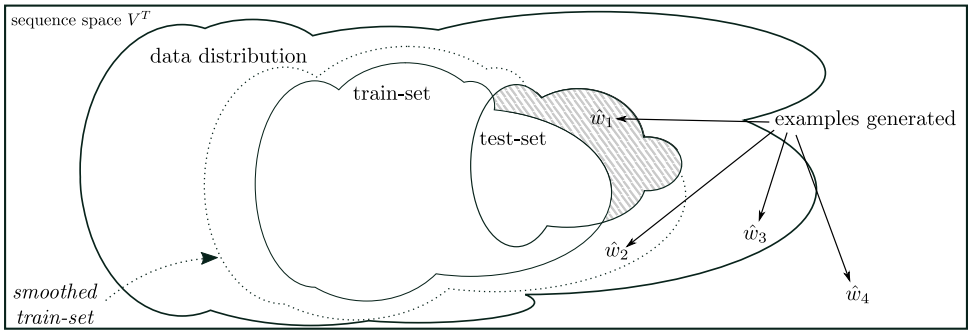

Figure 1: Generalization

an idealized two-dimensional dataspace of (fixed-length) sentences $w \in V^T$. We sketch the support of the unknown underlying generating distribution, the train set and the test set.[2] Let us look at some hypothetical examples $\hat{w}_1, \hat{w}_2, \hat{w}_3, \hat{w}_4$ generated from some well trained model. Samples like $\hat{w}_1$ certify that the model did not overfit to the training data as can be certified by test log-likelihood. In contrast, the remaining samples are indistinguishable under test log-likelihood in the sense that they identically decrease the metric (assuming equal model probability) even though $\hat{w}_2, \hat{w}_3$ have non-zero probability under the true data distribution. Consequently, we cannot identify $\hat{w}_4$ as a malformed example. Holtzman et al. (2019) show that neural generative models – despite their expressiveness – put significant probability on clearly unreasonable repetitive phrases, such as *I dont know. I dont know. I dont know*.[3]

Evaluation under smoothed data distribution The most common approach to evaluating an unconditional probabilistic generative model is training and test log-likelihood. For a latent variable model, the exact log-likelihood (2) is intractable and a lowerbound must be used instead. However, at this point it should be noted that one can always estimate the log-likelihood from an empirical distribution across output generated. That is, one generates a large set of sequences $\mathcal{S}$ and sets $\hat{p}(w)$ to the normalized count of w in $\mathcal{S}$. However, the variance of this estimate is impractical for all but the smallest datasets. Also, even a large test-set cannot capture the flexibility and compositionality found in natural language.

[2]Here we do not discuss *generalization error*, the discrepancy between empirical test error and expected test error. It should also be noted that cross-validation provides another complementary technique to more robust model estimation, which we omit to keep the picture simple.

[3]They report that this also holds for non-grammatical repetitive phrase, which is what we would expect for $\hat{w}_4$.

With aforementioned shortcomings of test log-likelihood in mind, it is worthwhile discussing a recently proposed evaluation technique. Fedus et al. (2018) propose to use n-gram statistics of the underlying data to asses generated output. For example, one can estimate an n-gram language model and report perplexity of the generated data under the n-gram model. Just as BLEU and ROUGE break the sequence reward assignment problem into smaller sub-problems, n-gram language models effectively *smooth* the sequence likelihood assignment which is usually done with respect to the empirical data distribution. Under this metric, some sequences such as $\hat{w}_2$ which are close to sequences in the dataset at hand might receive positive probability.

This raises two questions. First, can we break sequence-level evaluation into local statistics by using modern word embeddings instead of n-grams (as BLEU does)? Second, can we incorporate these measures already during training to obtain better generative models. These considerations will be key when defining a reward function in Section 4.5.

3 Teacher Forcing and Exposure Bias

A concern often expressed in the context of autoregressive models is that the recursive sampling procedure for generation presented in Section 1 is never used at training time; hence the model cannot learn to digest its own predictions. The resulting potential train-test discrepancy is referred to as *exposure bias* and is associated with compounding errors that arise when mistakes made early accumulate (Bengio et al., 2015; Ranzato et al., 2016; Goyal et al., 2016; Leblond et al., 2018). In this context, *teacher-forcing* refers to the fact that – seen from the test-time perspective – ground-truth contexts are substituted for model predictions. Although formally teacher forcing and exposure bias

should be seen as cause (if any) and symptom, they are often used exchangeably.

As is sometimes but rarely mentioned, the presence of the ground-truth context is simply a consequence of maximum-likelihood training and the chain rule applied to (1) as in $p(w_{1:T}) = \prod p(w_t|w_{1:t-1})$ (Goodfellow et al., 2016). As such, it is out of question whether generated contexts should be used as long as log-likelihood is the sole criterion we care about. In this work we will furthermore argue the following:

> **Proposition 1** *Exposure bias describes a lack of generalization with respect to an – usually implicit and potentially task and domain dependent – measure other than maximum-likelihood.*

The fact that we are dealing with generalization is obvious, as one can train a model – assuming sufficient capacity – under the criticized methodology to match the training distribution. Approaches that address exposure bias do not make the above notion of generalization explicit, but follow the intuition that training on other contexts than (only) ground-truth contexts should regularize the model and result in – subjectively – better results. Of course, these forms of regularization might still implement some form of log-likelihood regularization, hence improve log-likelihood generalization. Indeed, all of the following methods do report test log-likelihood improvements.

Proposed methods against exposure bias *Scheduled sampling* (Bengio et al., 2015) proposed for conditional generation randomly mixes in predictions form the model, which violates the underlying learning framework (Husz'ar, 2015). *RAML* (Norouzi et al., 2016) proposes to effectively perturbs the ground-truth context according to the exponentated payoff distribution implied by a reward function. Alternatively, adversarial approaches (Goyal et al., 2016) and learning-to-search (Leblond et al., 2018) have been proposed.

VAE Collapse In Section 4.1 we will take a look at *latent* generative models. In that context, the standard maximum-likelihood approach to autoregressive models has been criticized from a second perspective that is worth mentioning. Bowman et al. (2016) show empirically that autoregressive decoders $p(w|\mathbf{z})$ do not rely on the latent code $\mathbf{z}$, but collapse to a language model as in (1).

While some work argues that the problem is rooted in autoregressive decoders being "too powerful" (Shen et al., 2018), the proposed measures often address the autoregressive training regime rather than the models (Bowman et al., 2016) and, in fact, replace ground-truth contexts just as the above methods to mitigate exposure bias.

In addition, a whole body of work has discussed the implications of optimizing only a bound to the log-likelihood (Alemi et al., 2017) and the implications of re-weighting the information-theoretic quantities inside the bound (Higgins et al., 2017; Rainforth et al., 2018).

4 Latent Generation with ERPO

We have discussed exposure bias and how it has been handled by either implicitly or explicitly leaving the maximum-likelihood framework. In this section, we present our reinforcement learning framework for unconditional sequence generation models. The generative story is the same as in a latent variable model:

1. Sample a latent code $\mathbf{z} \sim \mathbb{R}^d$
2. Sample a sequence from a code-conditioned policy $p_\theta(w|\mathbf{z})$.

However, we will rely on reinforcement learning to train the decoder $p(w|\mathbf{z})$. Note that for a constant code $\mathbf{z} = \mathbf{0}$ we obtain a language model as a special case. Let us now briefly review latent sequential models.

4.1 Latent sequential models

Formally, a latent model of sequences $w = w_{1:T}$ is written as a marginal over latent codes

$$p(w) = \int p(w, \mathbf{z}) d\mathbf{z} = \int p(w|\mathbf{z}) p_0(\mathbf{z}) d\mathbf{z} \ . \quad (2)$$

The precise form of $p(w|\mathbf{z})$ and whether $\mathbf{z}$ refers to a single factor or a sequence of factors $\mathbf{z}_{1:T}$ depends on the model of choice.

The main motivation of enhancing p with a latent factor is usually the hope to obtain a meaningful structure in the space of latent codes. How such a structure should be organized has been discussed in the *disentanglement* literature in great detail, for example in Chen et al. (2018), Hu et al. (2017) or Tschannen et al. (2018).

In our context, latent generative models are interesting for two reasons. First, explicitly introducing uncertainty inside the model is often motivated as a regularizing technique in Bayseian machine learning (Murphy, 2012) and has been applied extensively to latent sequence models (M. Ziegler and M. Rush, 2019; Schmidt and Hofmann, 2018; Goyal et al., 2017; Bayer and Osendorfer, 2014). Second, as mentioned in Section 3 (VAE collapse) conditioning on ground-truth contexts has been identified as detrimental to obtaining meaningful latent codes (Bowman et al., 2016) – hence a methodology to training decoders that relaxes this requirement might be of value.

Training via Variational Inference Variational inference (Zhang et al., 2018) allows to optimize a lower-bound instead of the intractable marginal likelihood and has become the standard methodology to training latent variable models. Introducing an inference model q and applying Jensen's inequality to (2), we obtain

$$\log p(w) = \mathbb{E}_{q(\mathbf{z}|w)}\left[\log \frac{p_0(\mathbf{z})}{q(\mathbf{z}|w)} + \log P(w|\mathbf{z})\right]$$
$$\geq D^{\text{KL}}(q(\mathbf{z}|w)||p_0(\mathbf{z})) + \mathbb{E}_{q(\mathbf{z}|w)}\left[\log P(w|\mathbf{z})\right] \quad (3)$$

Neural inference networks (Rezende et al., 2014; Kingma and Welling, 2013) have proven as effective amortized approximate inference models.

Let us now discuss how reinforcement learning can help training our model.

4.2 Generation as Reinforcement Learning

Text generation can easily be formulated as a reinforcement learning (RL) problem if words are taken as actions (Bahdanau et al., 2016). Formally, p_θ is a parameterized policy that factorizes autoregressively $p_\theta(w) = \prod p_\theta(w_t|\mathbf{h}(w_{1:t-1}))$ and $\mathbf{h}$ is is a deterministic mapping from past predictions to a continuous state, typically a recurrent neural network (RNN). The goal is then to find policy parameters θ that maximize the expected reward

$$J(\theta) = \mathbb{E}_{p_\theta(w)}[R(w, w^\star)] \quad (4)$$

where $R(w, w^\star)$ is a task-specific, not necessarily differentiable metric.

Policy gradient optimization The REINFORCE (Williams, 1992) training algorithm is a common strategy to optimize (4) using a gradient estimate via the log-derivative

$$\nabla_\theta J(\theta) = \mathbb{E}_{p_\theta(w)}[R(w, w^\star) \log p_\theta(w)] \quad (5)$$

Since samples from the policy $\hat{w} \sim p_\theta$ often yield low or zeros reward, the estimator (5) is known for its notorious variance and much of the literature is focused on reducing this variance via baselines or control-derivative (Rennie et al., 2016).

4.3 Reinforcement Learning as Inference

Recently, a new family of policy gradient methods has been proposed that draws inspiration from inference problems in probablistic models. The underlying idea is to pull the reward in (5) into a new *implicit* distribution $\tilde{p}$ that allows to draw samples $\hat{w}$ with much lower variance as it is informed about reward.

We follow Tan et al. (2017) who optimize an entropy-regularized version of (4), a common strategy to foster exploration. They cast the reinforcement learning problem as

$$J(\theta, \tilde{p}) = \mathbb{E}_{\tilde{p}}[R(w, w^\star)] + \alpha D^{\text{KL}}(\tilde{p}(w)||p_\theta(w)) + \beta H(\tilde{p}) \quad (6)$$

where α, β are hyper-parameters and $\tilde{p}$ is the new non-parametric, *variational* distribution[4] across sequences. They show that (6) can be optimized using the following EM updates

$$\text{E-step: } \tilde{p}^{n+1} \propto \exp\left(\frac{\alpha p_\theta{}^n(w) + R(w, w^\star)}{\alpha + \beta}\right) \quad (7)$$

$$\text{M-step: } \theta^{n+1} = \arg\max_\theta \mathbb{E}_{\tilde{p}^{n+1}}[\log p_\theta(w)] \quad (8)$$

As Tan et al. 2018 have shown, for $\alpha \to 0$, $\beta = 1$ and a specific reward, the framework recovers maximum-likelihood training.[5] It is explicitly not our goal to claim text generation with end-to-end reinforcement learning but to show that it is beneficial to operate in an RL regime relatively close to maximum-likelihood.

4.4 Optimization with Variational Inference

In conditional generation, a policy is conditioned on a source sentence, which guides generation towards sequences that obtain significant reward. Often, several epochs of MLE pretraining (Rennie et al., 2016; Bahdanau et al., 2016) are necessary to make this guidance effective.

[4] In (Tan et al., 2018) $\tilde{p}$ is written as q, which resembles variational distributions in approximate Bayesian inference. However, here $\tilde{p}$ is not defined over variables but datapoints.

[5] Refer to their work for more special cases, including MIXER (Ranzato et al., 2016)

In our unconditional setting, where a source is not available, we employ the latent code $\mathbf{z}$ to provide guidance. We cast the policy p_θ as a code-conditioned policy $p_\theta(w|\mathbf{z})$ which is trained to maximize a marginal version of the reward (6):

$$J(\theta) = \mathbb{E}_{p_0(\mathbf{z})}\mathbb{E}_{p_\theta(w|\mathbf{z})}[R(w, w^\star)]] \,. \quad (9)$$

Similar formulations of expected reward have recently been proposed as *goal-conditioned* policies (Ghosh et al., 2018). However, here it is our explicit goal to also learn the representation of the goal, our latent code. We follow Equation (3) and optimize a lower-bound instead of the intractable marginalization (9). Following (Bowman et al., 2015; Fraccaro et al., 2016) we use a deep RNN inference network for q to optimize the bound. The reparametrization-trick (Kingma and Welling, 2013) allows us to compute gradients with respect to q. Algorithm 1 shows the outline of the training procedure.

Algorithm 1 Latent ERPO Training

for do $w^\star \in$ DATASET	
Sample a latent code	$\mathbf{z} \sim q(\mathbf{z}\|w^\star)$
Sample a datapoint	$\tilde{w} \sim \tilde{p}(w\|\mathbf{z})$
Perform a gradient step	$\nabla_\theta \log p_\theta(\tilde{w}\|\mathbf{z})$

Note that exploration (sampling $\tilde{w}$) and the gradient step are both conditioned on the latent code, hence stochasticity due to sampling a single $\mathbf{z}$ is coupled in both. Also, no gradient needs to be propagated into $\tilde{p}$.

So far, we have not discussed how to efficiently sample from the implicit distribution $\tilde{p}$. In the remainder of this section we present our reward function and discuss implications on the tractability of sampling.

4.5 Reward

Defining a meaningful reward function is central to the success of reinforcement learning. The usual RL forumlations in NLP require a measure of sentence-sentence similarity as reward. Common choices include BLEU (Papineni et al., 2002), ROUGE (Lin, 2004), CIDEr (Banerjee and Lavie, 2005) or SPICE (Anderson et al., 2016). These are essentially n-gram metrics, partly augmented with synonym resolution or re-weighting schemes.

Word-movers distance (WMD) (Kusner et al., 2015) provides an interesting alternative based on the optimal-transport problem. In essence, WMD computes the minimum accumulated distance that the word vectors of one sentence need to "travel" to coincide with the word vectors of the other sentence. In contrast to n-gram metrics, WMD can leverage powerful neural word representations. Unfortunately, the complexity of computing WMD is roughly $\mathcal{O}(T^3 \log T)$.

4.6 A Reward for Tractable Sampling

Tan et al. (2018) show that thanks to the factorization of p_θ the globally-normalized inference distribution $\tilde{p}$ in (7) can be written as a locally-normalized distribution at the word-level

$$\tilde{p}(w_t|w_{1:t-1}) \propto \exp\left(\frac{\alpha p_\theta(w_t|w_{1:t-1}) + R_t(w, w^\star)}{\alpha + \beta}\right) \quad (10)$$

when the reward is written as incremental reward R_t defined via $R_t(w, w^\star) = R(w_{1:t}, w^\star) - R(w_{1:t-1}, w^\star)$. Sampling form (10) is still hard, if R_t hides dynamic programming routines or other complex time-dependencies. With this in mind, we choose a particularly simple reward

$$R(w, w^\star) = \sum_{t=1}^{T} \phi(w_t)^\top \phi(w_t^\star) \quad (11)$$

where ϕ is a lookup into a length-normalized pre-trained but fixed word2vec (Mikolov et al., 2013) embedding. This casts our reward as an efficient, yet drastic approximation to WMD, which assumes identical length and one-to-one word correspondences. Putting (10) and (11) together, we sample sequentially from

$$\tilde{p}(w_t|w_{1:t-1}) \propto \exp\left(\frac{\alpha p_\theta(w_t|w_{1:t-1}) + \phi(w_t)^\top \phi(w_t^\star)}{\alpha + \beta}\right) \quad (12)$$

with the complexity $\mathcal{O}(dV)$ of a standard softmax. Compared to standard VAE training, Algorithm 1 only needs one additional forward pass (with identical complexity) to sample $\tilde{w}$ form $\tilde{p}$.

Equation (12) gives a simple interpretation of our proposed training methodology. We locally correct predictions made by the model proportionally to the distance to the ground-truth in the embeddings space. Hence, we consider the ground-truth and the model prediction for exploration.

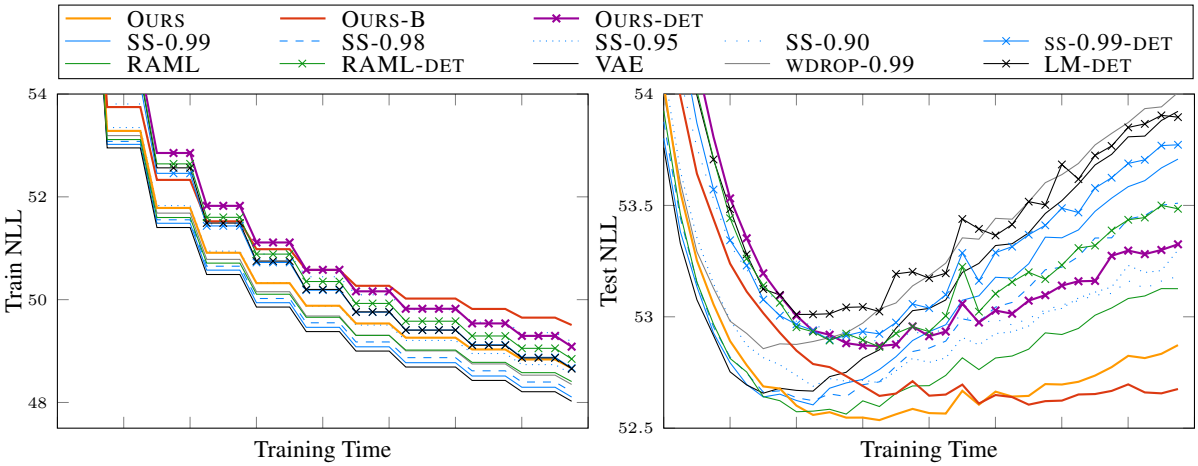

Figure 2: Generalization performance in terms of sequence NLL across latent and deterministic methods

5 Related Work

Our discussion of exposure bias complements recent work that summarizes modern generative models, for example Caccia et al. (2018) and Lu et al. (2018). Shortcomings of maximum-likelihood training for sequence generation have often been discussed (Ding and Soricut, 2017; Leblond et al., 2018; Ranzato et al., 2016), but without pointing to generalization as the key aspect. An overview of recent deep reinforcement learning methods for conditional generation can be found in (Keneshloo et al., 2018).

Our proposed approach follows work by Ding et al. (2017) and Tan et al. (2018) by employing both, policy and reward for exploration. In contrast to them, we do not use n-gram based reward. Compared to RAML (Norouzi et al., 2016), we do not perturb the ground-truth context, but correct the policy predictions. Scheduled sampling (Bengio et al., 2015) and word-dropout (Bowman et al., 2016) also apply a correction, yet one that only affects the probability of the ground-truth. Chen et al. (2017) propose Bridge modules that similarly to Ding et al. (2017) can incorporate arbitrary ground-truth perturbations, yet in an objective motivated by an auxiliary KL-divergence.

Merity et al. (2017) have shown that generalization is crucial to language modeling, but their focus is regularizing parameters and activations. Word-embeddings to measure deviations from the ground-truth have also been used by Inan et al. (2016), yet under log-likelihood. Concurrently to our work, Li et al. (2019) employ embeddings to design reward functions in abstractive summarization.

6 Experiments

Parametrization The policies of all our models and all baselines use the same RNN. We use a 256 dimensional GRU (Cho et al., 2014) and 100-dimensional pre-trained word2vec input embeddings. Optimization is preformed by Adam (Kingma and Ba, 2014) with an initial learning rate of 0.001 for all models. For all methods, including scheduled sampling, we do not anneal hyper-parameters such as the keep-probability for the following reasons. First, in an unconditional setting, using *only* the model's prediction is not a promissing setting, so it is unclear what value to anneal to. Second, the continous search-space of schedules makes it sufficiently harder to compare different methods. For the same reason, we do not investigate annealing the KL term or the α, β-parametrization of the models. We use the inference network parametrization of (Bowman et al., 2016) which employs a diagonal Gaussian for q.

We found the training regime to be very sensitive to the α, β-parametrization. In particular, it is easy to pick a set of parameters that does not truly incorporate exploration, but reduces to maximum likelihood training with only ground truth contexts (see also the discussion of Figure 3 in Section 6.2). After performing a grid-search (as done also for RAML) we choose[6] $\alpha = 0.006, \beta = 0.067$ for OURS, the method proposed. In addition, we report for an alternative model OURS-B with $\alpha = 0.01, \beta = 0.07$.

[6]The scale of α is relatively small as the log-probabilities in (12) have significantly larger magnitude than the inner products, which are in $[0, 1]$ due to the normalization.

Data For our experiments, we use a one million sentences subset of the BooksCorpus (Kiros et al., 2015; Zhu et al., 2015) with a 90-10 train-test split and a 40K words vocabulary. The corpus size is chosen to challenge the above policy with both scenarios, overfitting and underfitting.

6.1 Baselines

As baselines we use a standard VAE and a VAE with RAML decoding that uses identical reward as our method (see Tan et al.(2018) for details on RAML as a special case). Furthermore, we use two regularizations of the standard VAE, scheduled sampling SS-P and word-dropout WDROP-P as proposed by Bowman et al. (2016), both with fixed probability p of using the ground-truth.

In addition, we report as special cases with $\mathbf{z} = \mathbf{0}$ results for our model (OURS-DET), RAML (RAML-DET), scheduled sampling (SS-P-DET), and the VAE (LM, a language model).

6.2 Results

Figure 2 shows training and test negative sequence log-likelihood evaluated during training and Table 1 shows the best performance obtained. All figures and tables are averaged across three runs.

Model	Train NLL	Test NLL
OURS	48.52	**52.54**
OURS-B	49.51	52.61
OURS-DET	48.06	52.87
SS-0.99	48.11	52.60
SS-0.98	48.21	52.62
SS-0.95	48.38	52.69
SS-0.90	49.02	52.89
SS-0.99-DET	48.08	52.90
RAML	48.26	52.56
RAML-DET	48.26	52.86
WDROP-0.99	48.19	52.86
LM	**47.65**	53.01
VAE	47.86	52.66
WDROP-0.9	50.86	54.65

Table 1: Training and test performance

We observe that all latent models outperform their deterministic counterparts (crossed curves) in terms of both, generalization and overall test performance. This is not surprising as regularization is one of the benefits of modeling uncertainty through latent variables. Scheduled sampling does improve generalization for $p \approx 1$ with diminishing returns at $p = 0.95$ and in general performed better than word dropout. Our proposed models outperform all others in terms of generalization and test performance. Note that the performance difference over RAML, the second best method, is solely due to incorporating also model-predicted contexts during training.

Despite some slightly improved performance, all latent models except for OURS-B have a KL-term relatively close to zero. OURS-B is α-β-parametrized to incorporte slightly more model predictions at higher temperatur and manages to achieve a KL-term of about 1 to 1.5 bits. These findings are similar to what (Bowman et al., 2016) report *with* annealing but still significantly behind work that addresses this specific problem (Yang et al., 2017; Shen et al., 2018). Appendix A illustrates how our models can obtain larger KL-terms – yet at degraded performance – by controlling exploration. We conclude that improved autoregressive modeling inside the ERPO framework cannot alone overcome VAE-collapse.

We have discussed many approaches that deviate from training exclusively on ground-truth contexts. Therefore, an interesting quantity to monitor across methods is the fraction of words that correspond to the ground-truth. Figure 3 shows these fractions during training for the configurations that gave the best results. Interestingly, in the latent setting our method relies by far the least on ground-truth contexts whereas in the deterministic setting the difference is small.

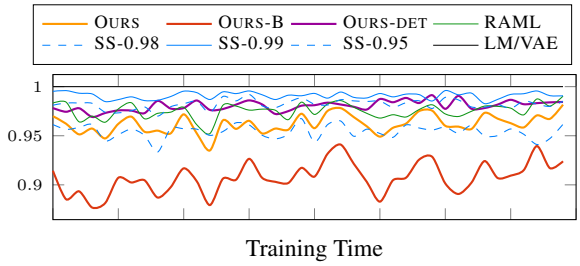

Figure 3: Fraction of correct words during training. Numbers include *forced* and *correctly predicted* words.

7 Conclusion

We have argued that exposure bias does not point to a problem with the standard methodology of training autoregressive sequence model. Instead, it refers to a notion of generalization to unseen sequences that does not manifest in log-likelihood training and testing, yet might be desirable in order to capture the flexibility of natural language.

To rigorously incorporate the desired generalization behavior, we have proposed to follow

the reinforcement learning formulation of Tan et al. (2018). Combined with an embedding-based reward function, we have shown excellent generalization performance compared to the unregularized model and better generalization than existing techniques on language modeling and sentence autoencoding.

Future work We have shown that the simple reward function proposed here leads to a form of regularization that fosters generalization when evaluated inside the maximum-likelihood framework. In the future, we hope to conduct a human evaluation to assess the generalization capabilities of models trained under maximum-likelihood and reinforcement learning more rigorously. Only such a framework-independent evaluation can reveal the true gains of carefully designing reward functions compared to simply performing maximum-likelihood training.

References

Alexander A. Alemi, Ben Poole, Ian Fischer, Joshua V. Dillon, Rif A. Saurous, and Kevin Murphy. 2017. An information-theoretic analysis of deep latent-variable models. *CoRR*, abs/1711.00464.

Peter Anderson, Basura Fernando, Mark Johnson, and Stephen Gould. 2016. SPICE: semantic propositional image caption evaluation. *CoRR*, abs/1607.08822.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron C. Courville, and Yoshua Bengio. 2016. An actor-critic algorithm for sequence prediction. *CoRR*, abs/1607.07086.

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

Satanjeev Banerjee and Alon Lavie. 2005. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, pages 65–72, Ann Arbor, Michigan. Association for Computational Linguistics.

Justin Bayer and Christian Osendorfer. 2014. Learning stochastic recurrent networks. *arXiv preprint arXiv:1411.7610*. ArXiv.

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *NIPS*.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Samuel R. Bowman, Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Józefowicz, and Samy Bengio. 2016. Generating sentences from a continuous space. In *ACL*.

Massimo Caccia, Lucas Caccia, William Fedus, Hugo Larochelle, Joelle Pineau, and Laurent Charlin. 2018. Language gans falling short. *CoRR*, abs/1811.02549.

Tian Qi Chen, Xuechen Li, Roger B. Grosse, and David Duvenaud. 2018. Isolating sources of disentanglement in variational autoencoders. *CoRR*, abs/1802.04942.

Wenhu Chen, Guanlin Li, Shujie Liu, Zhirui Zhang, Mu Li, and Ming Zhou. 2017. Neural sequence prediction by coaching. *CoRR*, abs/1706.09152.

Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *EMNLP*, pages 1724–1734.

Nan Ding and Radu Soricut. 2017. Cold-start reinforcement learning with softmax policy gradients. *CoRR*, abs/1709.09346.

William Fedus, Ian J. Goodfellow, and Andrew M. Dai. 2018. Maskgan: Better text generation via filling in the _____. In *ICLR*.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *EMNLP 2015*, pages 360–368.

Marco Fraccaro, Søren Kaae Sønderby, Ulrich Paquet, and Ole Winther. 2016. Sequential neural models with stochastic layers. pages 2199–2207. NIPS.

Dibya Ghosh, Abhishek Gupta, and Sergey Levine. 2018. Learning actionable representations with goal-conditioned policies. *CoRR*, abs/1811.07819.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep Learning*. MIT Press.

Anirudh Goyal, Alex Lamb, Ying Zhang, Saizheng Zhang, Aaron C. Courville, and Yoshua Bengio. 2016. Professor forcing: A new algorithm for training recurrent networks. In *NIPS*.

Anirudh Goyal, Alessandro Sordoni, Marc-Alexandre Côté, Nan Rosemary Ke, and Yoshua Bengio. 2017. Z-forcing: Training stochastic recurrent networks. In *NIPS*.

Tianxing He, Jingzhao Zhang, Zhiming Zhou, and James R. Glass. 2019. Quantifying exposure bias for neural language generation. *CoRR*, abs/1905.10617.

Irina Higgins, Loïc Matthey, Arka Pal, Christopher Burgess, Xavier Glorot, Matthew M Botvinick, Shakir Mohamed, and Alexander Lerchner. 2017. beta-vae: Learning basic visual concepts with a constrained variational framework. In *ICLR*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *CoRR*, abs/1904.09751.

Z. Hu, Z. Yang, Liang X., R. Salakhutdinov, and E. R. Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning (ICML)*.

Ferenc Husz'ar. 2015. How (not) to train your generative model: Scheduled sampling, likelihood, adversary?

Hakan Inan, Khashayar Khosravi, and Richard Socher. 2016. Tying word vectors and word classifiers: A loss framework for language modeling. *ArXiv*, abs/1611.01462.

Yaser Keneshloo, Tian Shi, Naren Ramakrishnan, and Chandan K. Reddy. 2018. Deep reinforcement learning for sequence to sequence models. *CoRR*, abs/1805.09461.

Urvashi Khandelwal, Kevin Clark, Dan Jurafsky, and Lukasz Kaiser. 2019. Sample efficient text summarization using a single pre-trained transformer. *CoRR*, abs/1905.08836.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. In *ICLR*.

Diederik P Kingma and Max Welling. 2013. Auto-encoding variational bayes. *arXiv preprint arXiv:1312.6114*.

Ryan Kiros, Yukun Zhu, Ruslan Salakhutdinov, Richard S Zemel, Antonio Torralba, Raquel Urtasun, and Sanja Fidler. 2015. Skip-thought vectors. *arXiv preprint arXiv:1506.06726*.

Matt J. Kusner, Yu Sun, Nicholas I. Kolkin, and Kilian Q. Weinberger. 2015. From word embeddings to document distances. In *ICML*, ICML'15, pages 957–966. JMLR.org.

Rémi Leblond, Jean-Baptiste Alayrac, Anton Osokin, and Simon Lacoste-Julien. 2018. SEARNN: training rnns with global-local losses. In *ICLR*.

Siyao Li, Deren Lei, Pengda Qin, and William Wang. 2019. Deep reinforcement learning with distributional semantic rewards for abstractive summarization.

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. page 10.

Sidi Lu, Yaoming Zhu, Weinan Zhang, Jun Wang, and Yong Yu. 2018. Neural text generation: Past, present and beyond. *CoRR*, abs/1803.07133.

Zachary M. Ziegler and Alexander M. Rush. 2019. Latent normalizing flows for discrete sequences. *arXiv preprint arXiv:1901.10548*.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and optimizing LSTM language models. *CoRR*, abs/1708.02182.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *NIPS*.

Kevin P. Murphy. 2012. *Machine Learning: A Probabilistic Perspective*. The MIT Press.

Mohammad Norouzi, Samy Bengio, Zhifeng Chen, Navdeep Jaitly, Mike Schuster, Yonghui Wu, and Dale Schuurmans. 2016. Reward augmented maximum likelihood for neural structured prediction. *CoRR*, abs/1609.00150.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation.

Tom Rainforth, Adam R. Kosiorek, Tuan Anh Le, Chris J. Maddison, Maximilian Igl, Frank Wood, and Yee Whye Teh. 2018. Tighter variational bounds are not necessarily better. In *ICML*.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. 2016. Sequence level training with recurrent neural networks. In *ICLR*.

Steven J. Rennie, Etienne Marcheret, Youssef Mroueh, Jerret Ross, and Vaibhava Goel. 2016. Self-critical sequence training for image captioning. *CoRR*, abs/1612.00563.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. 2014. Stochastic back-propagation and variational inference in deep latent gaussian models. In *ICML*.

Alexander M. Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. In *EMNLP*.

Florian Schmidt and Thomas Hofmann. 2018. Deep state space models for unconditional word generation. In *NeurIPS*.

Iulian Vlad Serban, Alessandro Sordoni, Yoshua Bengio, Aaron C Courville, and Joelle Pineau. 2016. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*.

Xiaoyu Shen, Hui Su, Shuzi Niu, and Vera Demberg. 2018. Improving variational encoder-decoders in dialogue generation. *CoRR*, abs/1802.02032.

Bowen Tan, Zhiting Hu, Zichao Yang, Ruslan Salakhutdinov, and Eric P. Xing. 2018. Connecting the dots between MLE and RL for sequence generation. *CoRR*, abs/1811.09740.

Michael Tschannen, Olivier Bachem, and Mario Lucic. 2018. Recent advances in autoencoder-based representation learning. *CoRR*, abs/1812.05069.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NIPS*.

Ronald J. Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.*, 8(3-4):229–256.

Zichao Yang, Zhiting Hu, Ruslan Salakhutdinov, and Taylor Berg-Kirkpatrick. 2017. Improved variational autoencoders for text modeling using dilated convolutions. *CoRR*, abs/1702.08139.

Cheng Zhang, Judith Butepage, Hedvig Kjellstrom, and Stephan Mandt. 2018. Advances in variational inference. *IEEE transactions on pattern analysis and machine intelligence*.

Yukun Zhu, Ryan Kiros, Richard Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *arXiv preprint arXiv:1506.06724*.