Sequence to Better Sequence: Continuous Revision of Combinatorial Structures

Jonas Mueller [1] **David Gifford** [1] **Tommi Jaakkola** [1]

Abstract

We present a model that, after learning on observations of (sequence, outcome) pairs, can be efficiently used to revise a new sequence in order to improve its associated outcome. Our framework requires neither example improvements, nor additional evaluation of outcomes for proposed revisions. To avoid combinatorial-search over sequence elements, we specify a generative model with continuous latent factors, which is learned via joint approximate inference using a recurrent variational autoencoder (VAE) and an outcome-predicting neural network module. Under this model, gradient methods can be used to efficiently optimize the continuous latent factors with respect to inferred outcomes. By appropriately constraining this optimization and using the VAE decoder to generate a revised sequence, we ensure the revision is fundamentally similar to the original sequence, is associated with better outcomes, and looks natural. These desiderata are proven to hold with high probability under our approach, which is empirically demonstrated for revising natural language sentences.

Introduction

The success of recurrent neural network (RNN) models in complex tasks like machine translation and audio synthesis has inspired immense interest in learning from sequence data (Eck & Schmidhuber, 2002; Graves, 2013; Sutskever et al., 2014; Karpathy, 2015). Comprised of elements $s_t \in \mathcal{S}$, which are typically symbols from a discrete vocabulary, a sequence $x = (s_1, \ldots, s_T) \in \mathcal{X}$ has length T which can vary between different instances. Sentences are a popular example of such data, where each s_j is a word from the language. In many domains, only a tiny fraction of $\mathcal{X}$ (the set of possible sequences over a given vocabulary) represents sequences likely to be found in nature (ie. those which appear realistic). For example: a random sequence of words will almost never form a coherent sentence that reads naturally, and a random amino-acid sequence is highly unlikely to specify a biologically active protein.

In this work, we consider applications where each sequence x is associated with a corresponding outcome $y \in \mathbb{R}$. For example: a news article title or Twitter post can be associated with the number of shares it subsequently received online, or the amino-acid sequence of a synthetic protein can be associated with its clinical efficacy. We operate under the standard supervised learning setting, assuming availability of a dataset $\mathcal{D}_n = \{(x_i, y_i)\}_{i=1}^n \overset{iid}{\sim} p_{XY}$ of sequence-outcome pairs. The marginal distribution p_X is assumed as a generative model of the natural sequences, and may be concentrated in a small subspace of $\mathcal{X}$. Throughout this paper, p denotes both density and distribution functions depending on the referenced variable.

After fitting models to $\mathcal{D}_n$, we are presented a new sequence $x_0 \in \mathcal{X}$ (with unknown outcome), and our goal is to quickly identify a revised version that is expected to have superior outcome. Formally, we seek the revised sequence:

$$x^* = \underset{x \in \mathcal{C}_{x_0}}{\operatorname{argmax}} \ \mathbb{E}[Y \mid X = x] \tag{1}$$

Here, we want the set $\mathcal{C}_{x_0}$ of feasible revisions to ensure that x^* remains natural and is merely a minor revision of x_0. Under a generative modeling perspective, these two goals are formalized as the following desiderata: $p_X(x^*)$ is not too small, and x^* and x_0 share similar underlying latent characteristics. When revising a sentence for example, it is imperative that the revision reads naturally (has reasonable likelihood under the distribution of realistic sentences) and retains the semantics of the original.

This optimization is difficult because the constraint-set and objective may be highly complex and are both unknown (must be learned from data). For many types of sequence such as sentences, standard distance measures applied directly in the space of $\mathcal{X}$ or $\mathcal{S}$ (eg. Levenshtein distance or TF-IDF similarity) are inadequate to capture meaningful similarities, even though these can be faithfully reflected by a simple metric over an appropriately learned space of continuous latent factors (Mueller & Thyagarajan, 2016). In this work, we introduce a generative-modeling framework which transforms (1) into a simpler differentiable optimiza-

[1]MIT Computer Science & Artificial Intelligence Laboratory. Correspondence to: J. Mueller <jonasmueller@csail.mit.edu>.

Proceedings of the 34th International Conference on Machine Learning, Sydney, Australia, PMLR 70, 2017. Copyright 2017 by the author(s).

tion by leveraging continuous-valued latent representations learned using neural networks. After the generative model has been fit, our proposed procedure can efficiently revise any new sequence in a manner that satisfies the aforementioned desiderata (with high probability).

Related Work

Unlike imitation learning, our setting does not require availability of improved versions of a particular sequence. This prevents direct application of a sequence-to-sequence model (Sutskever et al., 2014). Similar to our approach, Gómez-Bombarelli et al. (2016) also utilize latent autoencoder representations in order to propose novel chemical structures via Bayesian optimization. However, unlike sequential bandit/reinforcement-learning settings, our learner sees no outcomes outside of the training data, neither for the new sequence it is asked to revise, nor for any of its proposed revisions of said sequence (Mueller et al., 2017). Our methods only require an easily-assembled dataset of sequence-outcome pairs and are thus widely applicable.

Combinatorial structures are often optimized via complex search heuristics such as genetic programming (Zaefferer et al., 2014). However, search relies on evaluating isolated changes in each iteration, whereas good revisions of a sequence are often made over a larger context (ie. altering a phrase in a sentence). From the vast number of possibilities, such revisions are unlikely to be found by search-procedures, and it is generally observed that such methods are outperformed by gradient-based optimization in high-dimensional continuous settings. Unlike combinatorial search, our framework leverages gradients in order to efficiently find good revisions at test time. Simonyan et al. (2014) and Nguyen et al. (2015) also proposed gradient-based optimization of inputs with respect to neural predictions, but work in this vein has been focused on conditional generation (rather than revision) and is primarily restricted to the continuous image domain (Nguyen et al., 2016).

Methods

To identify good revisions, we first map our stochastic combinatorial optimization problem into a continuous space where the objective and constraints exhibit a simpler form. We assume the data are generated by the probabilistic graphical model in Figure 1A. Here, latent factors $Z \in \mathbb{R}^d$ specify a (continuous) configuration of the generative process for X, Y (both sequences and outcomes), and we adopt the prior $p_Z = N(0, \mathbf{I})$. Relationships between these variables are summarized by the maps F, E, D which we parameterize using three neural networks $\mathcal{F}$, $\mathcal{E}$, $\mathcal{D}$ trained to enable efficient approximate inference under this model.

The first step of our framework is to fit this model to $\mathcal{D}_n$

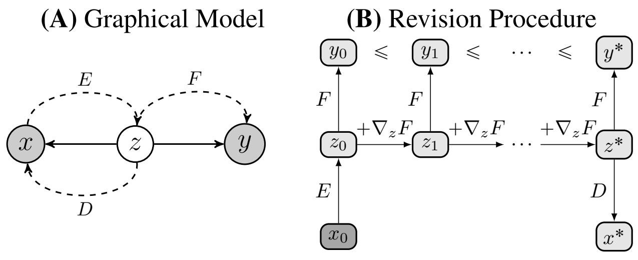

Figure 1. (A) Assumed graphical model (shaded nodes indicate observed variables, dashed arrows are learned neural network mappings). (B) Procedure for revising a given x_0 to produce x^* with superior expected outcome.

by learning the parameters of these inference networks: the *encoder* $\mathcal{E}$, the *decoder* $\mathcal{D}$, and the *outcome-predictor* $\mathcal{F}$. A good model that facilitates high-quality revision under our framework will possess the following properties: (1) Y can efficiently be inferred from Z and this relationship obeys a smooth functional form, (2) the map D produces a realistic sequence x given any z with reasonable prior probability, (3) the distribution of natural sequences is geometrically simple in the latent Z-space. We explicitly encourage (1) by choosing $\mathcal{F}$ as a fairly simple feedforward network, (2) by defining D as the most-likely x given z, and (3) by endowing Z with our simple $N(0, \mathbf{I})$ prior.

Another characteristic desired of our Z-representations is that they encode meaningful sequence-features such that two fundamentally similar sequences are likely to have been generated from neighboring z-values. Applied to image data, VAE models similar to ours have been found to learn latent representations that disentangle salient characteristics such as scale, rotation, and other independent visual concepts (Higgins et al., 2016). The latent representations of recurrent architectures trained on text (similar to the models used here) have also been shown to encode meaningful semantics, with a strong correlation between distances in the latent space and human-judged similarity between texts (Mueller & Thyagarajan, 2016). By exploiting such simplified geometry, a basic shift in the latent vector space may be able to produce higher-quality revisions than attempts to directly manipulate the combinatorial space of sequence elements.

After fitting a model with these desirable qualities, our strategy to revise a given sequence $x_0 \in \mathcal{X}$ is outlined in Figure 1B. First, we compute its latent representation $z_0 = E(x_0)$ using a trained encoding map. As the latent representations z are continuous, we can employ efficient gradient-based optimization to find a nearby local optimum z^* of $F(z)$ (within a simple constraint-set around z_0 defined later on). To z^*, we subsequently apply a simple decoding map D (defined with respect to our learned model) in order to obtain our revised sequence x^*. Under our assumed model, the optimization in latent representation-space attempts to identify a generative configuration which produces large values of Y (as inferred via F). The subsequent decoding step seeks the most likely sequence produced by the optimized setting of the latent factors.

Variational Autoencoder

For approximate inference in the X, Z relationship, we leverage the variational autoencoder (VAE) model of Kingma & Welling (2014). In our VAE, a generative model of sequences is specified by our prior over the latent values z combined with a likelihood function $p_D(x \mid z)$ which our decoder network $\mathcal{D}$ outputs in order to evaluate the likelihood of any sequence x given $z \in \mathbb{R}^d$. Given any sequence x, our *encoder* network $\mathcal{E}$ outputs a variational approximation $q_E(z \mid x)$ of the true posterior over the latent-values $p(z \mid x) \propto p_D(x \mid z) p_Z(z)$. As advocated by Kingma & Welling (2014) and Bowman et al. (2016), we employ the variational family $q_E(z \mid x) = N(\mu_{z|x}, \Sigma_{z|x})$ with diagonal covariance. Our revision methodology employs the encoding procedure $E(x) = \mu_{z|x}$ which maps a sequence to the maximum a posteriori (MAP) configuration of the latent values z (as estimated by the encoder network $\mathcal{E}$).

The parameters of $\mathcal{E}, \mathcal{D}$ are learned using stochastic variational inference to maximize a lower bound for the marginal likelihood of each observation in the training data:

$$\begin{aligned} \log p_X(x) &\geqslant -\left[\mathcal{L}_{\text{rec}}(x) + \mathcal{L}_{\text{pri}}(x)\right] \qquad (2) \\ \mathcal{L}_{\text{rec}}(x) &= -\mathbb{E}_{q_E(z|x)}\left[\log p_D(x \mid z)\right] \\ \mathcal{L}_{\text{pri}}(x) &= \text{KL}(q_E(z \mid x) || \, p_Z) \end{aligned}$$

Defining $\sigma_{z|x} = \text{diag}(\Sigma_{z|x})$, the prior-enforcing Kullback-Leibler divergence has a differentiable closed form expression when q_E, p_Z are diagonal Gaussian distributions. The *reconstruction* term $\mathcal{L}_{\text{rec}}$ (ie. negative log-likelihood under the decoder model) is efficiently approximated using just one Monte-Carlo sample $z \sim q_E(z \mid x)$. To optimize the variational lower bound over our data $\mathcal{D}_n$ with respect to the parameters of neural networks $\mathcal{E}$, $\mathcal{D}$, we use stochastic gradients of (2) obtained via backpropagation and the reparameterization trick of Kingma & Welling (2014).

Throughout, our encoder/decoder models $\mathcal{E}$, $\mathcal{D}$ are recurrent neural networks (RNN). RNNs adapt standard feedforward neural networks for sequence data $x = (s_1, \ldots, s_T)$, where at each time-step $t \in \{1, \ldots, T\}$, a fixed size hidden-state vector $h_t \in \mathbb{R}^d$ is updated based on the next element in the input sequence. To produce the approximate posterior for a given x, our encoder network $\mathcal{E}$ appends the following additional layers to the final RNN hidden-state (parameterized by $W_\mu, W_\sigma, W_v, b_\mu, b_\sigma, b_v$):

$$\begin{aligned} &\mu_{z|x} = W_\mu h_T + b_\mu \ \in \mathbb{R}^d \\ &\sigma_{z|x} = \exp(-|W_\sigma v + b_\sigma|), \ v = \text{ReLU}(W_v h_T + b_v) \quad (3) \end{aligned}$$

The (squared) elements of $\sigma_{z|x} \in \mathbb{R}^d$ form the diagonal of our approximate-posterior covariance $\Sigma_{z|x}$. Since $\mathcal{L}_{\text{pri}}$ is minimized at $\sigma_{z|x} = \vec{1}$ and $\mathcal{L}_{\text{rec}}$ is likely to worsen with additional variance in encodings (as our posterior approximation is unimodal), we simply do not consider $\sigma_{z|x}$ values that exceed 1 in our variational family. This restriction results in more stable training and also encourages the encoder and decoder to co-evolve such that the true posterior is likely closer to unimodal with variance $\leqslant 1$.

To evaluate the likelihood of a sequence, RNN $\mathcal{D}$ computes not only its hidden state h_t, but also the additional output:

$$\pi_t = \text{softmax}(W_\pi h_t + b_\pi) \qquad (4)$$

At each position t, π_t estimates $p(s_t \mid s_1, \ldots, s_{t-1})$ by relying on h_t to summarize the sequence history. By the factorization $p(s_1, \ldots, s_T) = \prod_{t=1}^T p(s_t \mid s_{t-1}, \ldots, s_1)$, we have $p_D(x \mid z) = \prod_{t=1}^T \pi_t[s_t]$, which is calculated by specifying an initial hidden-state $h_0 = z$ and feeding $x = (s_1, \ldots, s_T)$ into $\mathcal{D}$. From a given latent configuration z, our revisions are produced by decoding a sequence via the most-likely observation, which we denote as the map:

$$D(z) = \underset{x \in \mathcal{X}}{\text{argmax}} \, p_D(x \mid z) \qquad (5)$$

While the most-likely decoding in (5) is itself a combinatorial problem, beam search can exploit the sequential-factorization of $p(x \mid z)$ to efficiently find a good approximate solution (Wiseman & Rush, 2016; Sutskever et al., 2014). For $x^* = D(z) \in \mathcal{X}$, this decoding strategy seeks to ensure neither $p_X(x^*)$ nor $p(z \mid x^*)$ is too small.

Compositional Prediction of Outcomes

In addition to the VAE component, we fit a compositional outcome-prediction model which uses a standard feed forward neural network $\mathcal{F}$ to implement the map $F : \mathbb{R}^d \to \mathbb{R}$. It is assumed that $F(z) = \mathbb{E}[Y \mid Z = z]$ under our generative model. Rather than integrating over Z to compute $\mathbb{E}[Y \mid X = x] = \int F(z) q_E(z \mid x) \mathrm{d}z$, we employ the first-order Taylor approximation $F(E(x))$, where the approximation-error shrinks the more closely F resembles an affine transformation. To ensure this approximate-inference step accurately estimates the conditional expectation, we jointly train $\mathcal{E}$ and $\mathcal{F}$ with the loss:

$$\mathcal{L}_{\text{mse}}(x, y) = [y - F(E(x))]^2 \qquad (6)$$

If the architecture of networks $\mathcal{E}, \mathcal{F}$ is specified with sufficient capacity to capture the underlying conditional relationship, then we should have $F(E(x)) \approx \mathbb{E}[Y \mid X = x]$ after properly learning the network parameters from a sufficiently large dataset (even F is a nonlinear map).

Enforcing Invariance

In theory, it is possible that some dimensions of z pertain solely to the outcome y and do not have any effect on the decoded sequence $D(z)$. Happening to learn this sort of latent representation would be troubling, since subsequent optimization of the inferred y with respect to z might not actually lead to a superior revised sequence. To mitigate this issue, we carefully ensure the dimensionality d of our latent Z does not significantly exceed the bottleneck capacity needed to produce accurate outcome-predictions and VAE reconstructions (Gupta et al., 2016). We explicitly suppress this undesirable scenario by adding the following loss to guide training of our neural networks:

$$\mathcal{L}_{\text{inv}} = \mathbb{E}_{z \sim p_Z}\big[F(z) - F(E(D(z)))\big]^2 \tag{7}$$

When optimizing neural network parameters with respect to this loss, we treat the parameters of $\mathcal{D}$ and the lefthand $F(z)$ term as fixed, solely backpropagating Monte-Carlo estimated gradients into $\mathcal{E}$, $\mathcal{F}$. Driving $\mathcal{L}_{\text{inv}}$ toward 0 ensures our outcome-predictions remain invariant to variation introduced by the encoding-decoding process (and this term also serves as a practical regularizer to enforce additional smoothness in our learned functions).

Joint Training

The parameters of all components of this model (q_E, p_D, and F) are learned jointly in an end-to-end fashion. Training is done via stochastic gradient descent applied to minimize the following objective over the examples in $\mathcal{D}_n$:

$$\mathcal{L}(x, y) = \mathcal{L}_{\text{rec}} + \lambda_{\text{pri}}\mathcal{L}_{\text{pri}} + \frac{\lambda_{\text{mse}}}{\sigma_Y^2}\mathcal{L}_{\text{mse}} + \frac{\lambda_{\text{inv}}}{\sigma_Y^2}\mathcal{L}_{\text{inv}} \tag{8}$$

where σ_Y^2 denotes the (empirical) variance of the outcomes, and the $\lambda \geqslant 0$ are constants chosen to balance the relative weight of each goal so that the overall framework produces maximally useful revisions. By setting $\lambda_{\text{mse}} = \lambda_{\text{inv}} = 0$ at first, we can optionally leverage a separate large corpus of unlabeled examples to initially train only the VAE component of our architecture, as in the unsupervised pre-training strategy used successfully by Kiros et al. (2015); Erhan et al. (2010).

In practice, we found the following training strategy to work well, in which numerous mini-batch stochastic gradient updates (typically 10-30 epochs) are applied within every one of these steps:

Step 1: Begin with $\lambda_{\text{inv}} = \lambda_{\text{pri}} = 0$, so $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{mse}}$ are the only training objectives. We found that regardless of the precise value specified for λ_{mse}, both $\mathcal{L}_{\text{rec}}$ and $\mathcal{L}_{\text{mse}}$ were often driven to their lowest possible values during this joint optimization (verified by training individually against each objective).

Step 2: Grow λ_{pri} from 0 to 1 following the sigmoid annealing schedule proposed by Bowman et al. (2016), which is needed to ensure the variational sequence to sequence model does not simply ignore the encodings z (note that the formal variational lower bound is attained at $\lambda_{\text{pri}} = 1$).

Step 3: Gradually increase λ_{inv} linearly until $\mathcal{L}_{\text{inv}}$ becomes small on average across our Monte-Carlo samples $z \sim p_Z$. Here, p_D is treated as constant with respect to $\mathcal{L}_{\text{inv}}$, and each mini-batch used in stochastic gradient descent is chosen to contain the same number of Monte-Carlo samples for estimating $\mathcal{L}_{\text{inv}}$ as (sequence, outcome) pairs.

Proposing Revisions

While the aforementioned training procedure is computationally intensive, once learned, our neural networks can be leveraged for efficient inference. Given user-specified constant $\alpha > 0$ and a to-be-revised sequence x_0, we propose the revision x^* output by the following procedure.

```
REVISE Algorithm
Input: sequence $x_0 \in \mathcal{X}$, constant $\alpha \in (0, |2\pi\Sigma_{z|x_0}|^{-\frac{1}{2}})$
Output: revised sequence $x^* \in \mathcal{X}$
1) Use $\mathcal{E}$ to compute $q_E(z \mid x_0)$
2) Define $\mathcal{C}_{x_0} = \{z \in \mathbb{R}^d : q_E(z \mid x_0) \geqslant \alpha\}$
3) Find $z^* = \operatorname{argmax}_{z \in \mathcal{C}_{x_0}} F(z)$        (gradient ascent)
4) Return $x^* = D(z^*)$        (beam search)
```

Intuitively, the level-set constraint $\mathcal{C}_{x_0} \subseteq \mathbb{R}^d$ ensures that z^*, the latent configuration from which we decode x^*, is likely similar to the latent characteristics responsible for the generation of x_0. Assuming x_0 and x^* share similar latent factors implies these sequences are fundamentally similar according to the generative model. Note that $z^* = E(x_0)$ is always a feasible solution of the latent-factor optimization over $z \in \mathcal{C}_{x_0}$ (for any allowed value of α). Furthermore, this constrained optimization is easy under our Gaussian approximate-posterior, since $\mathcal{C}_{x_0}$ forms a simple ellipsoid centered around $E(x_0)$.

To find z^* in Step 3 of the REVISE procedure, we use gradient ascent initialized at $z = E(x_0)$, which can quickly reach a local maximum if F is parameterized by a simple feedforward network. Starting the search at $E(x_0)$ makes most sense for unimodal posterior approximations like our Gaussian q_E. To ensure all iterates remain in the feasible region $\mathcal{C}_{x_0}$, we instead take gradient steps with respect to a penalized objective $F(z) + \mu \cdot J(z)$ where:

$$J(z) = \log\Big[K - (z - E(x_0))^T \Sigma_{z|x_0}^{-1}(z - E(x_0))\Big]$$
$$K = -2\log[(2\pi)^{d/2}|\Sigma_{z|x}|^{1/2}\alpha] \tag{9}$$

and $0 < \mu \ll 1$ is gradually decreased toward 0 to ensure the optimization can approach the boundary of $\mathcal{C}_{x_0}$. In terms of resulting revision quality, we found this log barrier method outperformed other standard first-order techniques for constrained optimization such as the projected gradient and Franke-Wolfe algorithms.

In principle, our revision method can operate on the latent representations of a traditional deterministic autoencoder for sequences, such as the seq2seq models of Sutskever et al. (2014) and Cho et al. (2014). However, the VAE offers numerous practical advantages, some of which are highlighted by Bowman et al. (2016) in the context of generating more-coherent sentences. The posterior uncertainty of the VAE encourages the network to smoothly spread the training examples across the support of the latent distribution. In contrast, central regions of the latent space under a traditional autoencoder can contain holes (to which no examples are mapped), and it is not straightforward to avoid these in our optimization of z^*. Furthermore, we introduce an adaptive variant of our decoder in §S1 which is designed to avoid poor revisions in cases where the initial sequence is already not reconstructed properly: $D(E(x_0)) \neq x_0$.

Theoretical Properties of Revision

Here, we theoretically characterize properties of revisions obtained via our REVISE procedure (all proofs are relegated to §S3 in the Supplementary Material). Our results imply that in an ideal setting where our neural network inference approximations are exact, the revisions proposed by our method are guaranteed to satisfy our previously stated desiderata: x^* is associated with an expected outcome-increase, x^* appears natural (has nontrivial probability under p_X whenever x_0 is a natural sequence), and x^* is likely to share similar latent characteristics as x_0 (since x^* is the most likely observation generated from z^* and $q_E(z^* \mid x_0) \geqslant \alpha$ by design). Although exact approximations are unrealistic in practice, our theory precisely quantifies the expected degradation in the quality of proposed revisions that accompanies a decline in either the accuracy of our approximate inference techniques or the marginal likelihood of the original sequence to revise.

Theorems 1 and 2 below ensure that for an initial sequence x_0 drawn from the natural distribution, the likelihood of the revised sequence x^* output by our REVISE procedure under p_X has lower bound determined by the user-parameter α and the probability of the original sequence $p_X(x_0)$. Thus, when revising a sequence x_0 which looks natural (has substantial probability under p_X), our procedure is highly likely to produce a revised sequence x^* which also looks natural. The strength of this guarantee can be precisely controlled by choosing α appropriately large in applications where this property is critical.

In each high probability statement, our bounds assume the initial to-be-revised sequence x_0 stems from the natural distribution p_X, and each result holds for any fixed constant $\delta > 0$. We first introduce the following assumptions:

(A1) For $\delta > 0, \alpha > 0$, there exists $0 < \gamma \leqslant 1$ such that:

i. With probability $\geqslant 1 - \delta/2$ (over $x \sim p_X$):

$$p(z \mid x) \geqslant \gamma \cdot q_E(z \mid x) \quad \text{whenever} \quad q_E(z \mid x) \geqslant \alpha$$

ii. $\Pr(Z \notin B_{R/2}(0)) \geqslant \gamma \cdot \Pr(\widetilde{Z} \notin B_{R/2}(0))$

where $Z \sim N(0, \mathbf{I})$, and $\widetilde{Z} \sim q_Z$, the *average encoding distribution* defined by Hoffman & Johnson (2016) as:

$$q_Z(z) = \mathbb{E}_{x \sim p_X}[q_E(z \mid x)] \tag{10}$$

$B_R(0) = \{z \in \mathbb{R}^d : ||z|| \leqslant R\}$ denotes the Euclidean ball centered around 0 with radius R defined here as:

$$R = \max\{R_1, R_2\} \tag{11}$$

with $R_1 = \sqrt{-8 \log[\alpha \cdot (2\pi)^{d/2}]}$

$$R_2 = \max\{\widetilde{R}_2, 2\}, \quad \widetilde{R}_2 = \sqrt{8 - \tfrac{1}{4d} \log\left(\tfrac{\gamma\delta}{8}\right)}$$

(A2) There exists $\eta > 0$ (depends on δ) such that with probability $\geqslant 1 - \delta/2$ (over $x_0 \sim p_X$): $\quad p(z^* \mid x^*) \leqslant \eta$

This means the latent posterior is bounded at x^*, z^* (as defined in REVISE), where both depend upon the initial to-be-revised sequence x_0.

Theorem 1. *For any $\delta > 0$, (A1) and (A2) imply:*

$$p_X(x^*) \geqslant \frac{\alpha\gamma}{\eta} \cdot p_X(x_0)$$

with probability $\geqslant 1 - \delta$ (over $x_0 \sim p_X$).

Condition (A1) forms a generalization of absolute continuity, and is required since little can be guaranteed about our inference procedures if the variational posterior is too inaccurate. Equality holds in (A1) with probability 1 if the variational distributions q_E exactly represent the true posterior ($\gamma \to 1$ as the variational approximations become more accurate over the measure p_X). In practice, minimization of the *reverse* KL divergence ($\mathcal{L}_{\text{pri}}$) used in our VAE formulation ensures that $q_E(z \mid x)$ is small wherever the true posterior $p(z \mid x)$ takes small values (Blei et al., 2017).

While the bound in Theorem 1 has particularly simple form, this result hinges on assumption (A2). One can show for example that the inequality in (A2) is satisfied if the posteriors $p(z \mid x^*)$ are Lipschitz continuous functions of z at z^* (sharing one Lipschitz constant over all possible x^*). In general however, (A2) heavily depends on both the data distribution p_X and decoder model p_D. Therefore, we provide a similar lower bound guarantee on the likelihood of our revision x^* under p_X, which instead only relies on weaker assumption (A3) below.

(A3) There exists $L > 0$ such that for each $x \in \mathcal{X}$: $p_D(x \mid z)$ is a L-Lipschitz function of z over $B_{R+1}(0)$.

Here, L depends on δ (through R), and we assume $L \geqslant 1$ without loss of generality. (A3) is guaranteed to hold in the setting where we only consider sequences of finite length $\leqslant T$. This is because the probability output by our decoder model, $p_D(x \mid z)$, is differentiable with bounded gradients over all $z \in B_R(0)$ under any sequence-to-sequence RNN architecture which can be properly trained using gradient methods. Since $B_{R+1}(0) \subset \mathbb{R}^d$ is a closed interval, $p_D(x \mid z)$ must be Lipschitz continuous over this set, for a given value of x. We can simply define L to be the largest Lipschitz constant over the $|\mathcal{S}|^T$ possible choices of $x \in \mathcal{X}$ ($|\mathcal{S}|$ = size of the vocabulary). In the next theorem below, user-specified constant $\alpha > 0$ is defined in REVISE, and L, γ, R all depend on δ.

Theorem 2. *For any $\delta > 0$, if (A1) and (A3) hold, then with probability $\geqslant 1 - \delta$ (over $x_0 \sim p_X$):*

$$p_X(x^*) \geqslant \frac{Ce^{-R}}{L^d} \cdot \left[\gamma \cdot \alpha \cdot p_X(x_0)\right]^{d+1}$$

where constant $C = \dfrac{\pi^{d/2}}{\Gamma(\frac{d}{2}+1)} \cdot \dfrac{(d+1)^d}{(d+2)^{d+1}}$

Our final result, Theorem 3, ensures that our optimization of z^* with respect to F is tied to the expected outcomes at $x^* = D(z^*)$, so that large improvements in the optimization objective: $F(z^*) - F(E(x_0))$ imply that our revision procedure likely produces large expected improvements in the outcome: $\mathbb{E}[Y \mid X = x^*] - \mathbb{E}[Y \mid X = x_0]$. For this result, we make the following assumptions:

(A4) For any $\delta > 0$, there exists $\kappa > 0$ such that $\Pr(X \in \mathcal{K}) \geqslant 1 - \delta/2$, where we define:

$$\mathcal{K} = \{x \in \mathcal{X} : x_0 = x \implies p_X(x^*) \geqslant \kappa\} \tag{12}$$

as the subset of sequences whose improved versions produced by our REVISE procedure remain natural with likelihood $\geqslant \kappa$. Note that either Theorem 1 or 2 (with the corresponding assumptions) ensures that one can suitably define κ such that (A4) is satisfied (by considering a sufficiently large finite subset of $\mathcal{X}$).

(A5) For any $\kappa > 0$, there exists $\epsilon_{\text{mse}} > 0$ such that $\Pr(X \in \mathcal{E}_{\text{mse}}) > 1 - \kappa$, where we define:

$$\mathcal{E}_{\text{mse}} = \{x \in \mathcal{X} : |F(E(x)) - \mathbb{E}[Y|X = x]| \leqslant \epsilon_{\text{mse}}\} \tag{13}$$

(A6) For any $\delta > 0$, there exists $\epsilon_{\text{inv}} > 0$ such that:

$$|F(z) - F(E(D(z)))| \leqslant \epsilon_{\text{inv}} \quad \text{for all } z \in B_R(0) \subset \mathbb{R}^d$$

where R is defined in (11) and depends on δ.

Here, ϵ_{mse} and ϵ_{inv} quantify the approximation error of our neural networks for predicting expected outcomes and ensuring encoding-decoding invariance with respect to F. Standard learning theory implies both $\epsilon_{\text{mse}}, \epsilon_{\text{inv}}$ will be driven toward 0 if we use neural networks with sufficient capacity to substantially reduce $\mathcal{L}_{\text{mse}}$ and $\mathcal{L}_{\text{inv}}$ over a large training set.

Theorem 3. *For any $\delta > 0$, if conditions (A1), (A4), (A5), and (A6) hold, then with probability $\geqslant 1 - \delta - \kappa$:*

$$\Delta_{z*} - \epsilon \leqslant F(z^*) - F(E(x_0)) \leqslant \Delta_{z*} + \epsilon \tag{14}$$

where $\Delta_{z*} = \mathbb{E}[Y \mid X = x^*] - \mathbb{E}[Y \mid X = x_0]$
$\epsilon = \epsilon_{inv} + 2\epsilon_{mse}$

Here, $\kappa, \epsilon_{\text{inv}}$ are defined in terms of δ as specified in (A4), (A6), and ϵ_{mse} is defined in terms of κ as specified in (A5).

Experiments

All of our RNNs employ the Gated Recurrent Unit (GRU) of Cho et al. (2014), which contains a simple gating mechanism to effectively learn long-range dependencies across a sequence. Throughout, $\mathcal{F}$ is a simple feedforward network with 1 hidden layer and tanh activations (note that the popular ReLU activation is inappropriate for $\mathcal{F}$ since it has zero gradient over half its domain). Decoding with respect to p_D is simply done entirely greedily (ie. a beam-search of size 1) to demonstrate our approach is not reliant on search heuristics. §S2 contains additional details for each analysis.

Simulation Study

To study our methods in a setting where all aspects of performance can be quantified, we construct a natural distribution p_X over sequences of lengths 10-20 whose elements stem from the vocabulary $\mathcal{S} = \{A, B, \ldots, I, J\}$. Each sequence is generated via the probabilistic grammar of Table S1. For each sequence, the associated outcome y is simply the number of times A appears in the sequence (a completely deterministic relationship). Since A often follows C and is almost always followed by B under p_X, a procedure to generate natural revisions cannot simply insert/substitute A symbols at random positions.

Table 1 compares various methods for proposing revisions. Letting σ_Y denote the standard deviation of outcomes in $\mathcal{D}_n$, we evaluate each proposed x^* using a rescaled version of the actual underlying outcome-improvement: $\Delta_Y(x^*) = \sigma_Y^{-1}(\mathbb{E}[Y \mid X = x^*] - \mathbb{E}[Y \mid X = x_0])$. Except where sample size is explicitly listed, all models were trained using $n = 10,000$ (sequence, outcome) pairs sampled from the generative grammar. Wherever appropriate, the different methods all make use of the same neural network components with latent dimension $d = 128$. Other than α, all hyperparameters of each revision method described below were chosen so that over 1000 revisions, the Levenshtein (edit) distance $d(x^*, x_0) \approx 3.3$ on average.

Model	$\Delta_Y(x^*)$	$-\log p_X(x^*)$	$d(x^*, x_0)$
$\log\alpha = -10000$	**0.51** ±0.55	29.0 ±9.3	3.3 ±3.4
$n = 1000$	0.15 ±0.44	32.0 ±9.4	2.8 ±3.4
$n = 100$	0.02 ±0.30	37.0 ±9.7	4.2 ±4.0
$\log\alpha = -1$	0.20 ±0.39	**28.2** ±7.6	1.4 ±2.2
ADAPTIVE	0.47 ±0.49	28.8 ±9.0	3.1 ±3.4
$\lambda_{\text{inv}} = \lambda_{\text{pri}} = 0$	0.05 ±0.68	30.4 ±8.4	3.3 ±3.5
SEARCH	0.45 ±0.51	29.0 ±9.4	3.2 ±1.4

Table 1. Results for revisions x^* produced by different methods in our simulation study (averaged over the same test set of 1000 starting sequences $x_0 \sim p_X$, with ± 1 standard deviation shown and the best results in bold).

All three results above the line in Table 1 are based on the full model described in our joint training procedure, with new sequences proposed via our REVISE algorithm (using the setting $\log\alpha = -10000$). In the latter two results, this model was only trained on a smaller subset of the data. We also generated revisions via this same procedure with the more conservative choice $\log\alpha = -1$. ADAPTIVE denotes the same approach (with $\log\alpha = -10000$), this time using the adaptive decoding D_{x_0} introduced in §S1, which is intended to slightly bias revisions toward x_0. The model with $\lambda_{\text{inv}} = \lambda_{\text{pri}} = 0$ is a similar method using a deterministic sequence-to-sequence autoencoder rather than our probabilistic VAE formulation (no variational posterior approximation or invariance-enforcing) where the latent encodings are still jointly trained to predict outcomes via F. Under this model, a revision is proposed by starting at $E(x_0)$ in the latent space, taking 1000 (unconstrained) gradient steps with respect to F, and finally applying D to the resulting z.

The above methods form an ablation study of the various components in our framework. SEARCH is a different combinatorial approach where we randomly generate 100 revisions by performing 4 random edits in x_0 (each individual edit is randomly selected as one of: substitution, insertion, deletion, or no change). In this approach, we separately learn a *language-model* RNN L on our training sequences (Mikolov et al., 2010). Sharing the same GRU architecture as our decoder model, L directly estimates the likelihood of any given sequence under p_X. Of the randomly generated revisions, we only retain those sequences x for which $L(x) \geqslant \frac{1}{|\mathcal{S}|} L(x_0)$ (in this case, those which are not estimated to be $<$ 10 times less likely than the original sequence x_0 under p_X). Finally, we score each remaining candidate (including x_0) using the outcome-prediction model $F(E(x))$, and the best is chosen as x^*.

Table 1 shows that our probabilistic VAE formulation outperforms the alternative approaches, both in terms of outcome-improvement achieved as well as ensuring revisions follow p_X. For comparison, $-\log p_X(x_0)$ had an average value of 26.8 (over these 1000 starting sequences), and changing one randomly-selected symbol in each sequence to A results in an average negative log-probability of 32.8. Thus, all of our revision methods clearly account for p_X to some degree. We find that all components used in our REVISION procedure are useful in achieving superior revisions. While individual standard deviations seem large, nearly all average differences in Δ_Y or $-\log p_X$ values produced by different methods are statistically significant considering they are over 1000 revisions.

Model	$\Delta_Y(x^*)$	$\Delta_L(x^*)$	$d(x^*, x_0)$
$\log\alpha = -10000$	**0.52** ±0.77	-8.8 ±6.5	2.6 ±3.3
$\log\alpha = -1$	0.31 ±0.50	**-7.6** ±5.8	1.7 ±2.6
ADAPTIVE	0.52 ±0.72	-8.7 ±6.4	2.5 ±3.3
$\lambda_{\text{inv}} = \lambda_{\text{pri}} = 0$	0.22 ±1.03	-10.2 ±7.0	3.3 ±3.4
SEARCH	0.19 ±0.56	-7.7 ±4.2	3.0 ±1.2

Table 2. Results for revised beer-review sentences x^* produced by different methods (average $\pm$ standard deviation reported over the same held-out set of 1000 initial sentences x_0). The third column employs the definition $\Delta_L(x^*) = \log L(x^*) - \log L(x_0)$.

From Supplementary Figure S1, it is clear that α controls how conservative the changes proposed by our REVISE procedure tend to be, in terms of both $-\log p_X(x^*)$ and the edit distance $d(x_0, x^*)$. The red curve in Figure S1A suggests that our theoretical lower bounds for $p_X(x^*)$ are overly stringent in practice (although only the average-case is depicted in the figure). The relationship between $\log p_X(x_0)$ and $\log p_X(x^*)$ (see Figure S1B) is best-fit by a line of slope 1.2, indicating that the linear dependence on $p_X(x_0)$ in the Theorem 1 bound for $p_X(x^*)$ is reasonably accurate. Figure S1C shows that the magnitude of changes in the latent space (arising from z-optimization during our REVISE procedure) only exhibits a weak correlation with the edit distance between the resulting revision and the original sequence. This implies that a fixed shift in different directions in the latent space can produce drastically different degrees of change in the sequence space. To ensure a high-quality revision, it is thus crucial to carefully treat the (variational) posterior landscape when performing manipulations of Z.

Improving Sentence Positivity

Next, we apply our model to $\sim$1M reviews from BeerAdvocate (McAuley et al., 2012). Each beer review is parsed into separate sentences, and each sentence is treated as an individual sequence of words. In order to evaluate methods using an outcome that can be obtained for any proposed revision, we choose $y \in [0, 1]$ as the VADER sentiment compound score of a given sentence (Hutto & Gilbert,

Model	Sentence	$\Delta_Y(x^*)$	$\Delta_L(x^*)$	$d(x^*, x_0)$
x_0	**this smells pretty bad.**	-	-	-
$\log \alpha = -10000$	smells pretty delightful!	+2.8	-0.5	3
ADAPTIVE	smells pretty delightful!	+2.8	-0.5	3
$\log \alpha = -1$	i liked this smells pretty.	+2.5	-2.8	3
$\lambda_{\text{inv}} = \lambda_{\text{pri}} = 0$	pretty this smells bad!	-0.2	-3.1	3
SEARCH	wow this smells pretty bad.	+1.9	-4.6	1

Table 3. Example of a held-out beer review x_0 (in bold) revised to improve the VADER sentiment. Underneath the original sentence, we show the revision produced by each different method along with the true (rescaled) outcome improvement Δ_Y, change in estimated marginal likelihood Δ_L, and edit distance $d(x^*, x_0)$. Table S2 contains additional examples.

# Steps	Decoded Sentence
x_0	**where are you, henry??**
100	where are you, henry??
1000	where are you, royal??
5000	where art thou now?
10000	which cannot come, you of thee?
x^*	where art thou, keeper??
x_0	**you are both the same size.**
100	you are both the same.
1000	you are both wretched.
5000	you are both the king.
10000	you are both these are very.
x^*	you are both wretched men.

Table 4. Decoding from latent Z configurations encountered at the indicated number of (unconstrained) gradient steps from $E(x_0)$, for the model trained to distinguish sentences from Shakespeare vs. contemporary authors. Shown first and last are x_0 and the x^* returned by our REVISION procedure (constrained with $\log \alpha = -10000$). Table S3 contains additional examples.

2014). VADER is a complex rule-based sentiment analysis tool which jointly estimates polarity and intensity of English text, and larger VADER scores correspond to text that humans find more positive with high fidelity.

We applied all aforementioned approaches to produce revisions for a held-out set of 1000 test sentences. As p_X underlying these sentences is unknown, we report estimates thereof obtained from a RNN language-model L learned on the sentences in $\mathcal{D}_n$. Table 2 demonstrates that our VAE approach achieves the greatest outcome-improvement. Moreover, Tables 3 and S2 show that our probabilistically-constrained VAE revision approach produces much more coherent sentences than the other strategies.

Revising Modern Text in the Language of Shakespeare

For our final application, we assemble a dataset of $\sim$100K short sentences which are either from Shakespeare or a more contemporary source (details in §S2.3). In this training data, each sentence is labeled with outcome $y = 0.9$ if it was authored by Shakespeare and $y = 0.1$ otherwise (these values are chosen to avoid the flat region of the sigmoid output layer used in network $\mathcal{F}$). When applied in this domain, our REVISE procedure thus attempts to alter a sentence so that the author is increasingly expected to be Shakespeare rather than a more contemporary source.

Tables 4 and S3 show revisions (of held-out sentences) proposed by our REVISE procedure with adaptive decoding (see §S1), together with sentences generated by applying the adaptive decoder at various points along an unconstrained gradient-ascent path in latent Z space (following gradients of F). Since the data lack similar versions of a sentence written in both contemporary and Shakespearean language, this revision task is an ambitious application of our ideas. Without observing a continuous spectrum of outcomes or leveraging specially-designed style transfer features (Gatys et al., 2016), our REVISE procedure has to alter the underlying semantics in order to nontrivially increase the expected outcome of the revised sentence under F. Nevertheless, we find that many of the revised sentences look realistic and resemble text written by Shakespeare. Furthermore, these examples demonstrate how the probabilistic constraint in our REVISE optimization prevents the revision-generating latent Z configurations from straying into regions where decodings begin to look very unnatural.

Discussion

This paper presents an efficient method for optimizing discrete sequences when both the objective and constraints are stochastically estimated. Leveraging a latent-variable generative model, our procedure does not require any examples of revisions in order to propose natural-looking sequences with improved outcomes. These characteristics are proven to hold with high probability in a theoretical analysis of VAE behavior under our controlled latent-variable manipulations. However, ensuring semantic similarity in text-revisions remains difficult for this approach, and might be improved via superior VAE models or utilizing additional similarity labels to shape the latent geometry.

References

Blei, D. M., Kucukelbir, A., and McAuliffe, J. D. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 2017.

Bowman, S. R., Vilnis, L., Vinyals, O., Dai, A. M., Jozefowicz, R., and Bengio, S. Generating sentences from a continuous space. *Conference on Computational Natural Language Learning*, 2016.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., and Bengio, Y. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *Empirical Methods on Natural Language Processing*, 2014.

Eck, D. and Schmidhuber, J. A first look at music composition using lstm recurrent neural networks. *IDSIA Technical Report*, 2002.

Erhan, D., Bengio, Y., Courville, A., Manzagol, P., Vincent, P., and Bengio, S. Why does unsupervised pre-training help deep learning? *Journal of Machine Learning Research*, 11:625–660, 2010.

Gatys, L. A., Ecker, A. S., and Bethge, M. Image style transfer using convolutional neural networks. *Computer Vision and Pattern Recognition*, 2016.

Gómez-Bombarelli, R., Duvenaud, D., Hernández-Lobato, J. M., Aguilera-Iparraguirre, J., , Hirzel, T., Adams, R. P., and Aspuru-Guzik, A. Automatic chemical design using a data-driven continuous representation of molecules. *arXiv:1610.02415*, 2016.

Graves, A. Generating sequences with recurrent neural networks. *arXiv:1308.0850*, 2013.

Gupta, P., Banchs, R. E., and Rosso, P. Squeezing bottlenecks: Exploring the limits of autoencoder semantic representation capabilities. *Neurocomputing*, 175:1001–1008, 2016.

Higgins, I., Matthey, L., Glorot, X., Pal, A., Uria, B., Blundell, C., Mohamed, S., and Lerchner, A. Early visual concept learning with unsupervised deep learning. *arXiv:1606.05579*, 2016.

Hoffman, M. D. and Johnson, M. J. Elbo surgery: yet another way to carve up the variational evidence lower bound. *NIPS Workshop on Advances in Approximate Bayesian Inference*, 2016.

Hutto, C.J. and Gilbert, E. Vader: A parsimonious rule-based model for sentiment analysis of social media text. *Eighth International Conference on Weblogs and Social Media*, 2014.

Karpathy, A. The unreasonable effectiveness of recurrent neural networks. *Andrej Karpathy blog*, 2015. URL `karpathy.github.io`.

Kingma, D. P. and Welling, M. Auto-encoding variational bayes. *International Conference on Learning Representations*, 2014.

Kiros, R., Zhu, Y., Salakhutdinov, R., Zemel, R. S., Torralba, A., Urtasun, R., and Fidler, S. Skip-thought vectors. *Advances in Neural Information Processing Systems*, 2015.

McAuley, J., Leskovec, J., and Jurafsky, D. Learning attitudes and attributes from multi-aspect reviews. *IEEE International Conference on Data Mining*, 2012.

Mikolov, T., Karafiat, M., Burget, L., Cernocky, J., and Khudanpur, S. Recurrent neural network based language model. *Interspeech*, 2010.

Mueller, J. and Thyagarajan, A. Siamese recurrent architectures for learning sentence similarity. *Proc. AAAI Conference on Artificial Intelligence*, 2016.

Mueller, J., Reshef, D. N., Du, G., and Jaakkola, T. Learning optimal interventions. *Artificial Intelligence and Statistics*, 2017.

Nguyen, A., Yosinski, J., and Clune, J. Deep neural networks are easily fooled: High confidence predictions for unrecognizable images. *Computer Vision and Pattern Recognition*, 2015.

Nguyen, A., Dosovitskiy, A., Yosinski, J., Brox, T., and Clune, J. Synthesizing the preferred inputs for neurons in neural networks via deep generator networks. *Advances in Neural Information Processing Systems*, 2016.

Simonyan, K., Vedaldi, A., and Zisserman, A. Deep inside convolutional networks: Visualising image classification models and saliency maps. *ICLR Workshop Proceedings*, 2014.

Sutskever, I., Vinyals, O., and Le, Q.V. Sequence to sequence learning with neural networks. *Advances in Neural Information Processing Systems*, 2014.

Wiseman, S. and Rush, A. M. Sequence-to-sequence learning as beam-search optimization. *Empirical Methods in Natural Language Processing*, 2016.

Zaefferer, M., Stork, J., Friese, M., Fischbach, A., Naujoks, B., and Bartz-Beielstein, T. Efficient global optimization for combinatorial problems. *Genetic and Evolutionary Computation Conference*, 2014.