Text Generation with MCMC

Shlok Mishra

April 7, 2024

Introduction to Language Models

- **What are Language Models (LMs)?**
  Predict the likelihood of a sequence of words, enabling machines to generate and understand human languages.
- **Applications:**
  - Text generation
  - Sentiment analysis
  - Conversation AI
  - And more...
- **Why are they important?**
  Key to advancing AI's ability to interact in human-like ways, driving innovations in automation, content creation, and beyond.

Examples of Advanced Language Models

ChatGPT by OpenAI

Gemini by Google

LLaMA by Meta

- ChatGPT, Gemini, and LLaMA represent the cutting-edge in language model technology, each contributing uniquely to the field.
- Their development by leading tech companies underscores the significant investment and interest in AI and NLP research.

Introduction to Neural Text Generation

What is Neural Text Generation?

Neural Text Generation is the process of using neural networks to model the distribution of text sequences. This technology allows for the automatic creation of text that mimics human language patterns.

Importance

It's a foundational technology for a range of applications, from chatbots and automated story generation to translation services and beyond.

The Mathematics Behind Text Generation

Modeling Sequence Probability

The probability of a text sequence is decomposed into per-token conditional probabilities, parameterized by neural networks like transformers:

$$p_\theta = \prod_{t=1}^{T} p_\theta(y_t | y_{<t}), \tag{1}$$

where:

- $y = (y_1, \ldots, y_T)$ is a sequence of tokens, where a token is the smallest unit of text that holds meaning in the dataset. Depending on the preprocessing, a token can be a word, part of a word, a character, or a punctuation mark.
- $y_{<t}$ represents all tokens before the t^{th} token, providing a context for predicting the next token.
- θ denotes the model parameters, which are learned from data to map sequences of tokens to probabilities.

Introduction to Constrained Text Generation

Defining Constrained Text Generation

Constrained Text Generation focuses on producing text sequences that satisfy specific predefined constraints. These constraints can be categorized into:

- **Hard constraints:** Must-include tokens or phrases.
- **Soft constraints:** Desired semantic similarities or thematic consistencies.

Conditioning on Constraints and Prompts

Incorporating Constraints

Constraints are integrated directly into the generation model, allowing for text production that aligns with the set conditions, including conditioning on a specific prompt x.

$$p_\theta(y|C,x) = \prod_{t=1}^{T} p_\theta(y_t|y_{<t}, C, x), \tag{2}$$

where C denotes the constraints, and x is the given prompt guiding the generation process.

Goal

The ultimate goal is to generate text that is not only contextually and semantically coherent but also adheres closely to the predefined constraints and prompt x.

Lexically Constrained Text Generation

Key Elements

- **Fluency:** Ensures the text flows logically and naturally.
- **Keywords:** Predetermined words that must appear in the generated text.

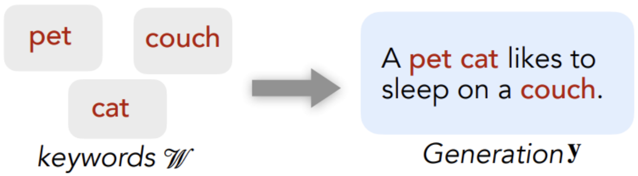
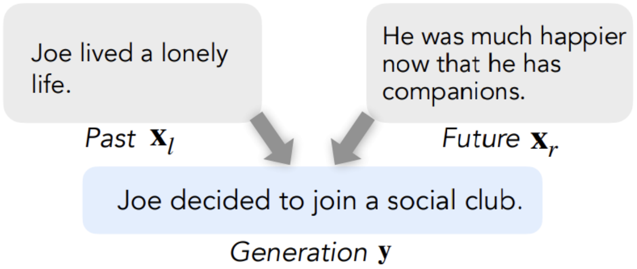


Figure: Example of lexically constrained text generation.

Abductive Reasoning in Text Generation

Key Constraints

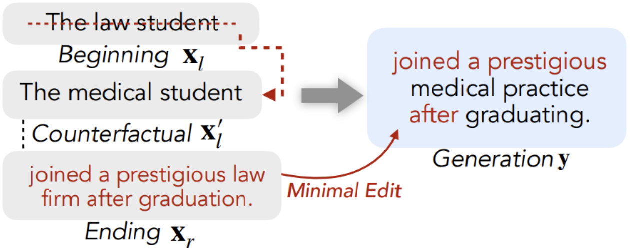
- **Fluency:** The generated text must read smoothly and naturally, adhering to grammatical norms.
- **Past Coherence:** The text must logically follow from the provided past context, creating a seamless narrative transition.
- **Future Coherence:** The text should lead logically into the given future scenario, ensuring a coherent and believable progression of events.



Counterfactual Reasoning in Text Generation

Core Constraints

- **Fluency:** Generated text must be smooth and grammatically correct.
- **Coherence:** The alternate narrative should logically follow from the original context, maintaining a coherent story or argument.
- **Minimal Edit:** Changes to the original text should be minimal, altering the narrative without extensive rewrites.


~~The law student~~
Beginning $\mathbf{x}_l$
The medical student
Counterfactual $\mathbf{x}'_l$
joined a prestigious law firm after graduation.
Ending $\mathbf{x}_r$
Minimal Edit
joined a prestigious medical practice after graduating.
Generation $\mathbf{y}$

Introduction to Energy-Based Models (EBMs)

What is an Energy Function?

In the realm of text generation, an **energy function** $E(y)$ assesses the compatibility of a text sequence y with specific constraints. Lower energy values signify a higher degree of adherence to these constraints, guiding the generation towards more desirable outcomes.

Defining EBMs

EBMs offer a robust framework for utilizing energy functions to shape the distribution of generated text sequences. This approach is encapsulated in the Boltzmann distribution:

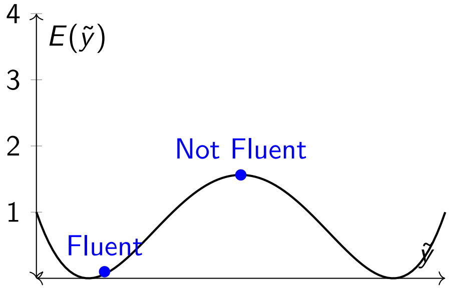
$$p(y) = \frac{\exp(-E(y))}{Z}, \tag{3}$$

where $E(y) \in \mathbb{R}$ represents the sequence's energy, and Z, the normalizing factor, ensures probabilities sum to 1.

Leveraging Energy Functions in EBMs

- Energy functions $E(y)$ central to embedding multiple constraints in models.
  - Encodes fluency, coherence, and keyword inclusion directly into $E(y)$.
  - Sequences fulfilling constraints have lower energy.
- EBMs' versatility key to their effectiveness.
  - Provide a unified system for incorporating diverse constraints.
  - Facilitate generation of contextually aligned and condition-compliant text.
  - Enhance text quality and broad applicability.

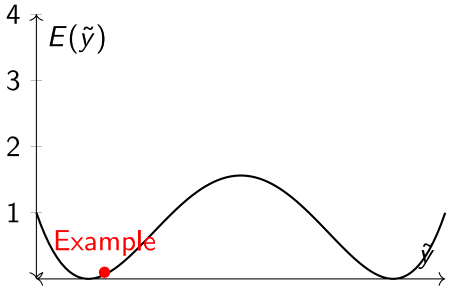
Energy Function: Fluent Generation



$$f_{\overrightarrow{\mathrm{LM}}}(\tilde{\mathbf{y}}) = \sum_{t=1}^{T} \sum_{v \in V} p_{\overrightarrow{\mathrm{LM}}}(v|\tilde{\mathbf{y}}_{<t}) \log \operatorname{softmax}(\tilde{\mathbf{y}}_t(v))$$

- **Not Fluent Example:** “I has a dog where is curious.”
- **Fluent Example:** “I have a dog that is very curious.”

Energy Function: Keyword Emphasis



$$f_{\text{sim}}(\tilde{\mathbf{y}}; \mathbf{y}^*) = \text{ngram-match}(\tilde{\mathbf{y}}, \mathbf{y}^*), \tag{4}$$

- **Example with Keywords:** "My dog liked to sleep on a couch."
- **Keywords** ($\mathbf{y}^*$): {sleep, couch}

Coherence Constraints: Foundation

Coherence Constraint Equation

$$f_{pred}(\tilde{\mathbf{y}}; \mathbf{x_r}) = \sum_{k=1}^{K} \log p_{\overrightarrow{LM}}(x_{r,k} | \tilde{\mathbf{y}}, \mathbf{x}_{r,<k}), \quad (5)$$

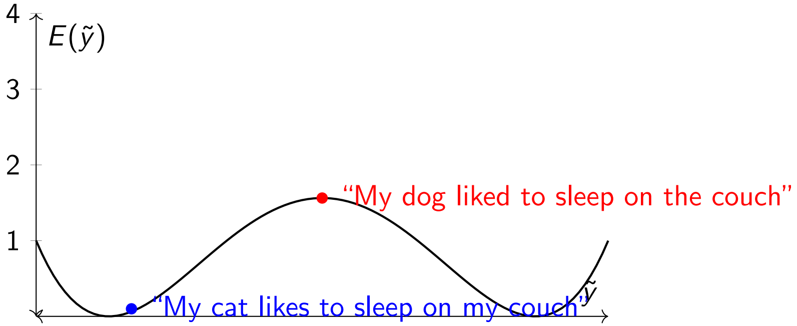
Example Setup for Coherence

Illustrating the function of coherence constraints using a fraction-like representation:

- Text Generation Objective (?): Generate text $\tilde{\mathbf{y}}$ that logically precedes or follows the context.
- Contextual Right Side ($\mathbf{x_r}$): "I love my cat."

This setup aims to generate a coherent sequence that aligns with the given context, $\mathbf{x_r}$.

Coherence Constraints: Visualization



Interpreting the Points:

- The blue point represents a sentence coherent with the right context "I love my cat," hence lower energy.
- The red point represents a less coherent sentence, not perfectly aligning with the context, thus higher energy.

Total Energy and Induced Distribution in Text Generation

Energy Function and Induced Distribution

- The set of constraints induces a distribution over text, formulated in an energy-based form as:

$$p(y) = \frac{\exp\left(\sum_i \lambda_i f_i(y)\right)}{Z},$$

  where $\lambda_i \geq 0$ is the weight of the i^{th} constraint, and Z is the normalizing factor.

- Here, the energy function $E(y)$ is defined as:

$$E(y) := -\sum_i \lambda_i f_i(y),$$

  which aggregates the weighted sum of various energy components, each reflecting a specific constraint.

Challenges in Constrained Generation: Energy Functions

Complex High-Dimensional Energy Functions

The energy function $E(y)$ in constrained text generation presents significant challenges:

- The **normalizing factor** Z is intractable due to summation over all possible states, complicating direct sampling efforts.
- Text sequences y are **inherently discrete**, making traditional Markov Chain Monte Carlo (MCMC) methods like Gibbs sampling less effective due to slow convergence in high-dimensional spaces.

Solutions: Continuous Approximation and Langevin Dynamics

Addressing Sampling Challenges

To navigate the complexity of sampling from high-dimensional energy functions:

- **Continuous Approximation:** Transforming the discrete sequence y into a continuous space allows for the application of more sophisticated sampling techniques.
- **Langevin Dynamics:** A gradient-based MCMC method that exploits the continuous approximation to perform efficient sampling by utilizing the gradient information of $E(y)$.

Advantages

These strategies enable effective and efficient exploration of the energy landscape, facilitating the generation of text that adheres to the specified constraints while overcoming the limitations of discrete sequence spaces.

Gradient-Based MCMC for Efficient Sampling

Langevin Dynamics

The Langevin dynamics update rule for sampling is given by:

$$\tilde{\boldsymbol{y}}^{(n+1)} \leftarrow \tilde{\boldsymbol{y}}^{(n)} - \eta \nabla_{\tilde{\boldsymbol{y}}} E(\tilde{\boldsymbol{y}}^{(n)}) + \epsilon^{(n)}, \tag{6}$$

where $\epsilon^{(n)} \sim \mathcal{N}(0, \sigma)$ is the noise term at iteration n, and η is the step size.

Efficient Sampling

Gradient-based MCMC methods like Langevin dynamics allow for more efficient sampling from complex distributions by utilizing the gradient of the energy function $\nabla_{\tilde{\boldsymbol{y}}} E(\tilde{\boldsymbol{y}})$. This approach leverages gradient information to navigate the energy landscape and generate samples that adhere to the desired constraints.

From Discrete Tokens to Continuous Soft Sequences

Soft Sequence Representation

Rather than operating on discrete tokens, we construct an energy function over a soft sequence of continuous vectors:

$$\tilde{\boldsymbol{y}} = (\tilde{\boldsymbol{y}}_1, \ldots, \tilde{\boldsymbol{y}}_T),$$

where each $\tilde{\boldsymbol{y}}_t \in \mathbb{R}^V$ represents a position in the sequence with V being the vocabulary size.

Example: Soft Sequence in Natural Language Processing

Model's Logits at Time Step t

Given a small vocabulary, the neural network model produces logits for each word at time step t:

Word	**Logit**	**Probability**
“the”	2	0.12
“cat”	5	0.79
“sat”	1	0.07
“mat”	-1	0.02

Table: Logits and probability distribution for each word in the vocabulary.

Soft Sequence at Time Step t

The continuous representation of the model's prediction at time t is a vector indicating the probability of each word:

$$[0.12, 0.79, 0.07, 0.02]$$

With this representation, “cat” is the most probable next word

Discretization of Soft Sequence After MCMC

Utilizing the Language Model as a Guardian

- Post-MCMC, we discretize the continuous soft sequence $\tilde{y}$ into discrete tokens y.
- The language model (e.g., GPT2-XL) guides the selection of discrete tokens from the soft sequence.

Top-k Filtering Method

At each position t, the language model suggests the top-k most likely candidate tokens given the preceding tokens:

- Denote the top-k candidate set as ν_k^t.
- The discrete token y_t is chosen as the one with the highest logit from $\tilde{y}_t$ within the top-k set:

$$y_t = \arg\max_{v \in \nu_k^t} \tilde{y}_t(v). \tag{7}$$

Challenges in Implementation of COLD Decoding

- Publicly available implementation: [1]
- Extensive codebase: Tens of Thousands of lines to navigate across multiple files.
- Resource-intensive: Requires high-memory GPUs beyond personal computing capabilities.
- Acknowledgment: Gratitude to Prof. Dootika Vats for providing necessary resources.
- Learning curve: Complex use of PyTorch and advanced NLP concepts.
- Time constraint: Difficulty in mastering sophisticated codebase rapidly.

[1] `https://github.com/qkaren/COLD_decoding`

Langevin MCMC in High-Dimensional Text Generation

Langevin MCMC Fundamentals

- Inspired by particle dynamics in physics.
- Balances deterministic drift with stochastic exploration.

Update Formula

The Langevin MCMC update rule is given by:

$$x_{t+1} = x_t + \frac{\epsilon^2}{2} \nabla \log p(x_t) + \epsilon Z_t, \tag{8}$$

Efficiency and Applications

- Effective in high-dimensional sampling.
- Widely applicable in machine learning and physics.
- Utilizes gradient information for enhanced convergence.

Issue with Langevin Dynamics in COLD Decoding

Discrepancy in Implementation

- The COLD decoding paper claims to use Langevin dynamics for text generation.
- However, the actual implementation utilizes the Adam optimizer.
- Adam optimizer inherently adjusts learning rates adaptively.
- This contradicts the requirements of Langevin MCMC where a fixed ratio between the learning rate and noise is crucial.

Paper's Claim vs. Reality

The title suggests a methodology based on Langevin dynamics, but the use of Adam calls into question the fidelity of the Langevin process in their approach.

Barker Proposal MCMC for Enhanced Sampling

Acceptance Probability

For perturbation z from $z \sim \mathcal{N}(0, \sigma^2)$, the acceptance probability α is:

$$\alpha = \frac{1}{1 + e^{-z \cdot \nabla_y E(y_k)}}, \tag{9}$$

balancing exploration and exploitation based on the energy gradient.

Update Rule

The state update is probabilistically determined:

$$y_{k+1} = \begin{cases} y_k + z & \text{with probability } \alpha, \\ y_k - z & \text{with probability } 1 - \alpha. \end{cases} \tag{10}$$

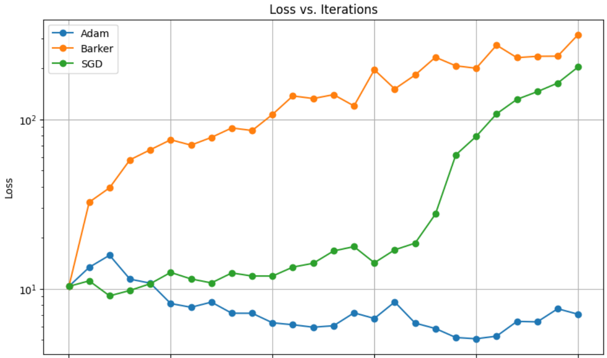
This rule adapts to both the energy landscape and stochastic perturbations.

Results - Performance of SGD and Barker MCMC

Observations

- Direct implementations of SGD and Barker Proposal MCMC demonstrated poor performance in our tests.
- Contrastingly, Adam optimizer showed more stable loss trajectories.

Figure: Loss Trajectories Comparison



Future Directions in Sampling Methodologies

- Thorough investigation into why Adam optimizer outperforms in certain contexts.
- Identify specific challenges with Langevin and Barker sampling techniques.
- Collaborative research with Professor Vats to deepen our understanding of these issues.
- Strategy: Construct and test models from scratch on a manageable scale.
- Focus on experimenting with different sampling techniques to identify effective strategies.
- Engage deeply with NLP aspects of the research to uncover foundational insights.
- Goal: Achieve a nuanced understanding of text generation challenges using MCMC methods.

Thank You

Thank You for Your Attention!

Questions?