Optimizing Unnormalized Statistical Models Through Compositional Optimization

Wei Jiang, Jiayu Qin, Lingyu Wu, Changyou Chen, Tianbao Yang, *Senior Member, IEEE*, and Lijun Zhang, *Senior Member, IEEE*

***Abstract*—Learning unnormalized statistical models (e.g., energy-based models) is computationally challenging due to the complexity of handling the partition function. To eschew this complexity, noise-contrastive estimation (NCE) has been proposed by formulating the objective as the logistic loss between the real data and the artificial noise. However, previous research indicates that NCE may perform poorly in many tasks due to its flat loss landscape and slow convergence. In this paper, we study a direct approach for optimizing the negative log-likelihood of unnormalized models through the lens of compositional optimization. To tackle the partition function, a noise distribution is introduced such that the log partition function can be expressed as a compositional function whose inner function can be estimated using stochastic samples. Consequently, the objective can be optimized via stochastic compositional optimization algorithms. Despite being a simple method, we demonstrate it is more favorable than NCE by (1) establishing a fast convergence rate and quantifying its dependence on the noise distribution through the variance of stochastic estimators; (2) developing better results in Gaussian mean estimation by showing our method has a much favorable loss landscape and enjoys faster convergence; (3) demonstrating better performance on various applications, including density estimation, out-of-distribution detection, and real image generation.**

***Index Terms*—Noise-contrastive estimation, stochastic compositional optimization, unnormalized statistical models, parameter-free algorithms, max likelihood estimation.**

I. Introduction

THIS paper studies the problem of learning unnormalized statistical models. Suppose we observe a set of training samples $\mathcal{D}_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ from an unknown probability density function (pdf) $p_{\text{data}}(\mathbf{x})$ and estimate this data pdf by

$$p(\mathbf{x};\theta) = \frac{p_0(\mathbf{x};\theta)}{\int p_0(\mathbf{x};\theta)d\mathbf{x}}, \tag{1}$$

where $p_0(\mathbf{x};\theta)$ is defined as an unnormalized model, and θ denotes the parameter that will be learnt to fit the data. The term $\int p_0(\mathbf{x};\theta)d\mathbf{x}$ in (1) is called partition function, which is used to ensure the final model is normalized, i.e., $\int p(\mathbf{x};\theta)d\mathbf{x} = 1$. By introducing the partition function, we can use more flexible structures to represent $p_0(\mathbf{x};\theta)$. Specifically, by setting $p_0(\mathbf{x};\theta) = e^{f_0(\mathbf{x};\theta)}$, the above formulation (1) reduces to the well-known energy-based model (EBM) [1]:

$$p(\mathbf{x};\theta) = \frac{e^{f_0(\mathbf{x};\theta)}}{\int e^{f_0(\mathbf{x};\theta)}d\mathbf{x}},$$

which enjoys wide applications in machine learning [2], [3], [4].

To find the best parameter θ, we can maximize the log-likelihood of the observed training data. This process involves optimizing the objective $\mathcal{L}(\theta)$, defined as:

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[\log p_0(\mathbf{x}_i;\theta)] + \log\int p_0(\mathbf{x};\theta)d\mathbf{x}. \tag{2}$$

However, the challenge is that the log partition function term $\log\int p_0(\mathbf{x};\theta)d\mathbf{x}$ and its gradient are difficult to calculate exactly. To address this issue, prior works [5], [6], [7] resort to Markov chain Monte Carlo (MCMC) sampling methods [8]. Considering the gradient of loss $\mathcal{L}(\theta)$, it can be expressed as:

$$\nabla\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[\frac{\nabla p_0(\mathbf{x}_i;\theta)}{p_0(\mathbf{x}_i;\theta)}\right] + \mathbb{E}_{\mathbf{x}\sim p_\theta}\left[\frac{\nabla p_0(\mathbf{x};\theta)}{p_0(\mathbf{x};\theta)}\right],$$

where p_θ represents the pdf $p(\mathbf{x},\theta)$, i.e., (1). To compute the second term, previous literature applies several MCMC steps to sample from p_θ, and then calculates the estimated gradient. However, this method is slow and unstable during training [9], [10], [11], partly because approximate samples are obtained with only a limited number of steps. As pointed out by [9] and [12], estimating based on finite MCMC steps will produce biased estimations, leading to optimizing a different objective other than the original MLE loss.

To eschew the complexity of estimating the gradient of the log partition function, the Noise-Contrastive Estimation (NCE) method [13] has been proposed, which transforms the original problem into a classification task. Specifically, NCE introduces

Received 21 February 2024; revised 12 September 2025; accepted 5 October 2025. Date of publication 14 October 2025; date of current version 9 January 2026. This work was supported in part by the NSFC under Grant U23A20382, and in part by the Collaborative Innovation Center of Novel Software Technology and Industrialization. Recommended for acceptance by Rishabh Krishnan Iyer. *(Corresponding author: Lijun Zhang.)*

Wei Jiang is with the School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: jw.njust@gmail.com).

Jiayu Qin and Changyou Chen are with the Department of Computer Science and Engineering, University at Buffalo, the State University of New York, Buffalo, NY 14260 USA (e-mail: jiayuqin@buffalo.edu; changyou@buffalo.edu).

Lingyu Wu and Lijun Zhang are with the National Key Laboratory for Novel Software Technology, and School of Artificial Intelligence, Nanjing University, Nanjing 210023, China (e-mail: wuly@lamda.nju.edu.cn; zhanglj@lamda.nju.edu.cn).

Tianbao Yang is with the Department of Computer Science and Engineering, Texas A&M University, College Station, TX 77843 USA (e-mail: tianbao-yang@tamu.edu).

This article has supplementary downloadable material available at https://doi.org/10.1109/TPAMI.2025.3621320, provided by the authors.

Digital Object Identifier 10.1109/TPAMI.2025.3621320

a noise distribution $q(\mathbf{x})$, and then aims to optimize the extended model parameter $\tau = (\theta, \alpha)$ by minimizing the following loss function:

$$\mathcal{J}(\tau) = -\mathbb{E}_{\mathbf{x}\sim p_{\text{data}}} \log h(\mathbf{x}, \tau) - \mathbb{E}_{\mathbf{x}\sim q} \log(1 - h(\mathbf{x}, \tau)),$$

where $h(\mathbf{x}, \tau) = 1/(1 + e^{-(\log p_0(\mathbf{x},\theta) - \log q(\mathbf{x}) - \alpha)})$, and the parameter α is used to estimate the log partition function, i.e., $\alpha = \log \int p_0(\mathbf{x}; \theta) d\mathbf{x}$. This new objective can be interpreted as the logistic loss to distinguish between noise and real data. In practice, the first term is estimated using n observed training examples, and the second term is evaluated with noise samples.

However, as pointed out in many works [10], [14], if the noise distribution q is very different from the data distribution p_{data}, this classification problem would be too easy to solve and the learned model may fail to capture adequate information about the real data distribution. In particular, [15] demonstrates that when q and p_{data} are not close enough, the loss $\mathcal{J}(\tau)$ is extremely flat near its optimum, leading to the slow convergence rate of NCE method.

In this paper, we explore an alternative approach to directly optimizing the MLE objective by transforming it into a stochastic compositional optimization (SCO) problem. To deal with the partition function, we introduce a noise distribution $q(\mathbf{x})$, and then convert the log partition function $\log \int p_0(\mathbf{x}; \theta) d\mathbf{x}$ into $\log \mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$. Then, the objective function (2) becomes:

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n} [\log p_0(\mathbf{x}; \theta)] + \log \mathbb{E}_{\mathbf{x}\sim q}\left[\frac{p_0(\mathbf{x}; \theta)}{q(\mathbf{x})}\right],$$

which can be written as a two-level SCO problem. Since SCO has been studied extensively, state-of-the-art algorithms can be employed to solve the above problem. However, besides its simplicity, a major question remains: *What are the advantages of this approach compared with NCE and MCMC-based methods for learning unnormalized models?* Our main contributions are to demonstrate the following advantages of our method through theoretical analysis and empirical studies:

1) We propose to solve the unnormalized models by Maximum likelihood Estimation via Compositional Optimization (MECO) and prove that our algorithm converges asymptotically to an optimal solution under the Polyak-Łojasiewicz (PL) condition, and establish a fast convergence rate of $\mathcal{O}(1/\epsilon)$ for achieving an ϵ-optimal solution. In contrast, NCE and MCMC-based approaches do not provide such guarantees.
2) We prove through one-dimensional Gaussian mean estimation that the MLE objective enjoys a more favorable loss landscape than the NCE objective, and establish a faster convergence rate of our algorithm than a state-of-the-art NCE-based approach [15] for this task.
3) We illustrate the better performance of our method on different tasks, including density estimation, out-of-distribution detection, and real image generation. For the last task, we show that the choice of the noise distribution has a significant impact on the quality of generated data in line with our theoretical analysis.

A preliminary version of this paper was presented at the 40th International Conference on Machine Learning [16]. In this paper, we have enriched the initial version by incorporating the following significant extensions.

1) The MECO method requires knowing problem-dependent variables, such as smoothness and Lipschitz constant, to set hyperparameters, which are hard to know in practice. To address this limitation, we introduce a Parameter-Free version of MECO (PF-MECO), which eliminates the need to know these variables. Since PF-MECO can adjust hyperparameters automatically, it significantly reduces the requirement for extensive hyperparameter tuning.
2) Although the MECO method provides convergence guarantees for the problem, we are not sure whether this convergence rate is optimal or not. In this paper, we propose the Optimal MECO (OP-MECO) method, which achieves an improved and optimal rate for non-convex functions, reducing the complexity from $\mathcal{O}(\epsilon^{-4})$ to $\mathcal{O}(\epsilon^{-3})$.
3) By using a two-phase design and a weighting strategy, we improve the rate of MECO for PL objectives, and establish the rate for convex functions. Furthermore, we develop a variant of the OP-MECO algorithm to achieve a better and optimal convergence for convex and PL functions.
4) We compare the newly proposed algorithms, PF-MECO and OP-MECO, with existing methods in three different tasks, and numerical results demonstrate that the OP-MECO algorithm performs better in many cases, validating the effectiveness of the newly proposed methods.

A comparison between this paper and its earlier conference version is provided in Table I.

II. Related Work

This section reviews related work on noise-contrastive estimation, other methods for learning unnormalized models, energy-based models, and stochastic compositional optimization.

A. Noise-Contrastive Estimation

Noise-contrastive estimation (NCE) is first proposed by [13], [17], and gains its popularity in various machine learning applications quickly [18], [19], [20], [21], [22]. The basic idea is to introduce a noise distribution, and then distinguish it from the data distribution by a logistic loss. As pointed out in previous works [10], [14], [15], the selection of an appropriate noise is crucial to the success of NCE, and [23] showed that the commonly used Gaussian noise is problematic in practice. As a result, many methods have been proposed to tune the noise automatically. For example, [24] utilizes a learned adversarial distribution as the noise, and [25] generates noise samples with the help of the observed data. Afterward, [10] proposes Flow Contrastive Estimation, using a flow model to learn the noise distribution by joint training. However, these methods usually introduce extra computation, and the adversarial training of the noise is slow and complex. Instead of designing more complicated noise, [15] recently proposed a new objective named eNCE, which replaces the log loss in NCE with an exponential loss. Yet, eNCE still suffers from the ill-behaved loss landscape,

TABLE I
SUMMARY OF DIFFERENCES BETWEEN THIS PAPER AND THE CONFERENCE VERSION

Method	Parameter-free	Non-convex	Convex	PL-condition
MECO	✗	$\mathcal{O}\left(\epsilon^{-4}\right)$		$\mathcal{O}\left(\max\left\{\frac{1}{\mu\sqrt{\epsilon}}, \mu^{-2}\epsilon^{-1}\right\}\right)$
MECO-v2 **(new)**	✗		$\mathcal{O}\left(\epsilon^{-3}\right)$	$\mathcal{O}\left(\max\left\{\frac{1}{\mu}\ln\frac{1}{\epsilon}, \mu^{-2}\epsilon^{-1}\right\}\right)$
PF-MECO **(new)**	✓	$\mathcal{O}\left(\epsilon^{-4}\right)$		
OP-MECO **(new)**	✓	$\mathcal{O}\left(\epsilon^{-3}\right)$	$\mathcal{O}\left(\epsilon^{-2}\right)$	$\mathcal{O}\left(\mu^{-1}\epsilon^{-1}\right)$

which is extremely flat near the optimum. A more recent and comprehensive understanding can be found in the paper [26], which offers a unified perspective on various methods for learning unnormalized distributions through NCE.

B. Other Methods for Learning Unnormalized Models

Except for NCE and its variants, there are several alternative approaches to solve unnormalized models. To bypass the partition function, score matching [27] is proposed to optimize the squared distance between the gradient of the model's log density and that of the observed data, thereby excluding the partition function from calculations. However, the output dimension of score matching is the same as the input, and training such a model is expensive and unstable, especially for high-dimensional data [28], [29]. Besides that, optimizing the MLE objective with Markov chain Monte Carlo (MCMC) sampling is also widely used. One well-known method is Contrastive Divergence (CD) introduced by [5], which employs MCMC sampling for a fixed number of steps. To sample more efficiently, [6] initializes the MCMC in each training step with the outcome of the previous sampling and names this method as persistent CD. Other variants of CD have also been proposed later, such as modified CD [30], adversarial CD [31], etc.

C. Energy-Based Models

Energy-Based Models (EBMs) are a typical class of unnormalized probabilistic models, which have been successfully applied in various domains, including image generation [2], [32], discriminative learning [4], natural language processing [33], density estimation [34] and reinforcement learning [35]. Recent advancements have focused on addressing challenges in training EBMs, particularly the difficulty of estimating the partition function. Techniques such as contrastive divergence [5], score matching [36], [37], and noise contrastive estimation [38] have been developed to facilitate effective training. Additionally, methods like Langevin dynamics [39] and Hamiltonian Monte Carlo sampling [40] have been employed to improve sampling efficiency. On top of that, [41] and [42] employ a generator as a fast sampler to help EBM training. However, the main drawback of most methods is that the sampling step is usually time-consuming and unstable during training [9], [10], [11].

D. Stochastic Compositional Optimization

Stochastic Compositional Optimization (SCO) has been investigated extensively in the literature. The objective of a two-level SCO is defined as $\mathbb{E}_{\xi_1}[f_{\xi_1}(\mathbb{E}_{\xi_2}[g_{\xi_2}(\theta)])]$, where ξ_1 and ξ_2 are random variables. To solve this problem, [43] develops stochastic compositional gradient descent, which achieves a complexity of $\mathcal{O}(\epsilon^{-7})$, $\mathcal{O}(\epsilon^{-3.5})$, and $\mathcal{O}(\mu^{-14/4}\epsilon^{-5/4})$ for non-convex, convex, and μ-strongly convex functions, respectively. These complexities are further improved to $\mathcal{O}(\epsilon^{-4.5})$, $\mathcal{O}(\epsilon^{-2})$ and $\mathcal{O}(\epsilon^{-1})$ in a subsequent work [44]. To obtain a better rate, [45] propose the NASA algorithm, which incorporates a momentum-update technique to estimate the inner function and the gradient, and attains a complexity of $\mathcal{O}(\epsilon^{-4})$ for non-convex objectives. Recently, with the popularity of variance reduction methods such as SARAH [46], SPIDER [47] and STORM [48], this complexity is further improved to $\mathcal{O}(\epsilon^{-3})$, by estimating the inner function and the gradient with variance-reduction techniques [49], [50], [51], [52], [53], [54].

III. MAXIMUM LIKELIHOOD ESTIMATION VIA COMPOSITIONAL OPTIMIZATION

We first present our proposed algorithm for solving unnormalized models, and then discuss the advantages of the MLE formulation. Finally, we analyze the convergence rate of our method, as well as its relationship with the noise distribution.

A. The Proposed Method

In this subsection, we propose to learn unnormalized models by Maximum Likelihood Estimation via Compositional Optimization (MECO). First, with a noise distribution $q(\mathbf{x})$, the MLE objective function (2) can be reformulated as:

$$\mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[\log p_0(\mathbf{x}_i;\theta)] + \log \mathbb{E}_{\mathbf{x}\sim q}\left[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}\right].$$

However, optimizing this objective is still challenging due to the nested structure of the second term. Specifically, we can not acquire its unbiased estimation by sampling from $q(\mathbf{x})$, since the expectation cannot be moved out of the log function, i.e., $\mathbb{E}_{\mathbf{x}\sim q}[\log \frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}] \neq \log \mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$. Due to similar reasons, we also can not obtain an unbiased estimation of its gradient, which is the main obstacle to deriving an algorithm with convergence guarantees.

To address this challenge, we treat $\log \mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$ as a compositional function. In this formulation, $\log(\cdot)$ is the outer function, and $\mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$ serves as the inner function, which can be unbiased-estimated by sampling from $q(\mathbf{x})$. Inspired by the NASA method [45] for solving stochastic compositional optimization (SCO), we employ variance-reduced estimators to approximate both the inner function and the gradient, so that the estimation errors can be reduced over time. Specifically,

Algorithm 1: MECO.

1: **Input:** Initial points $(\theta_1, \mathbf{u}_1, \mathbf{v}_1)$, sequence $\{\eta_t, \gamma_t, \beta_t\}$
2: **for** time step $t = 1$ **to** T **do**
3: Sampling $\mathbf{z}_t$ from $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $\widetilde{\mathbf{z}}_t$ from $q(\mathbf{x})$
4: Update estimator $\mathbf{u}_t$ according to (3)
5: Update estimator $\mathbf{v}_t$ according to (4)
6: Update the weight: $\theta_{t+1} = \theta_t - \eta_t \mathbf{v}_t$
7: **end for**
8: Choose τ uniformly at random from $\{1, \ldots, T\}$
9: Return θ_τ

considering the gradient of the objective, we have:

$$\nabla \mathcal{L}(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\left[\frac{\nabla p_0(\mathbf{x};\theta)}{p_0(\mathbf{x};\theta)}\right] + \frac{\mathbb{E}_{\mathbf{x}\sim q}\left[\frac{\nabla p_0(\mathbf{x};\theta)}{q(\mathbf{x})}\right]}{\mathbb{E}_{\mathbf{x}\sim q}\left[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}\right]}.$$

To estimate this gradient, at each step t, we first draw sample $\mathbf{z}_t$ from the training set $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$, and sample $\widetilde{\mathbf{z}}_t$ from the noise distribution $q(\mathbf{x})$. Then, we approximate $\mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$ using a function estimator $\mathbf{u}_t$ in the style of momentum update:

$$\mathbf{u}_t = (1-\gamma_t)\mathbf{u}_{t-1} + \gamma_t \frac{p_0(\widetilde{\mathbf{z}}_t;\theta_t)}{q(\widetilde{\mathbf{z}}_t)}. \tag{3}$$

When evaluating the gradient, we use $\mathbf{u}_t$ to approximate the term $\mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$ in the denominator. Similarly, we employ a gradient estimator $\mathbf{v}_t$ to track the overall gradient $\nabla\mathcal{L}(\theta)$ by another momentum update:

$$\mathbf{v}_t = (1-\beta_t)\mathbf{v}_{t-1} + \beta_t\left(-\frac{\nabla p_0(\mathbf{z}_t;\theta_t)}{p_0(\mathbf{z}_t;\theta_t)} + \frac{1}{\mathbf{u}_t}\frac{\nabla p_0(\widetilde{\mathbf{z}}_t;\theta_t)}{q(\widetilde{\mathbf{z}}_t)}\right). \tag{4}$$

Although the estimators $\mathbf{u}_t$ and $\mathbf{v}_t$ are still biased, it can be proved that their estimation error is reduced over time. After obtaining the gradient estimator $\mathbf{v}_t$, we use it to update the parameter θ_t in the style of SGD. The whole algorithm is presented in Algorithm 1. Note that in the first iteration, we can initiate the estimators as $\mathbf{u}_1 = \frac{p_0(\widetilde{\mathbf{z}}_1;\theta_1)}{q(\widetilde{\mathbf{z}}_1)}$ and $\mathbf{v}_1 = -\frac{\nabla p_0(\mathbf{z}_1;\theta_1)}{p_0(\mathbf{z}_1;\theta_1)} + \frac{1}{\mathbf{u}_1}\frac{\nabla p_0(\widetilde{\mathbf{z}}_1;\theta_1)}{q(\widetilde{\mathbf{z}}_1)}$.

Difference from NASA: The optimization method we used is a modified version of NASA for SCO problems [45]. Compared with the original NASA applied to constrained SCO, our method does not need to project the variable θ onto the feasible set and is thus simpler. Additionally, we introduce a parameter-free version of MECO later, which is more practical to use. Another difference from NASA lies in the improved rate we will derive for an objective satisfying the PL condition or convexity, which is not covered in their original work.

Difference from energy-based model (EBM) training: Derived from SCO, our algorithm fundamentally differs from standard EBM training. By setting $p_0(\mathbf{x};\theta) = e^{f_0(\mathbf{x};\theta)}$, the unnormalized model converts to an EBM, and our gradient estimator $\mathbf{v}_t$ is represented as:

$$(1-\beta_t)\mathbf{v}_{t-1} + \beta_t\left(-\nabla f(\mathbf{z}_t;\theta_t) + \frac{e^{f(\widetilde{\mathbf{z}}_t;\theta_t)}}{q(\widetilde{\mathbf{z}}_t)\mathbf{u}_t}\nabla f(\widetilde{\mathbf{z}}_t;\theta_t)\right).$$

In contrast, EBM usually optimizes the objective function $\mathcal{L}'(\theta) = -\mathbb{E}_{\mathbf{z}\sim p_{\text{data}}}[f(\mathbf{z};\theta)] + \mathbb{E}_{\widetilde{\mathbf{z}}\sim q}[f(\widetilde{\mathbf{z}};\theta)]$, with the stochastic gradient expressed as $\nabla\mathcal{L}'(\theta_t) = -\nabla f(\mathbf{z}_t;\theta_t) + \nabla f(\widetilde{\mathbf{z}}_t;\theta_t)$. In this sense, our method can be viewed as introducing an adaptive weight $\frac{e^{f(\widetilde{\mathbf{z}}_t;\theta_t)}}{q(\widetilde{\mathbf{z}}_t)\mathbf{u}_t}$ to the second term and applying SGD with momentum to update the model.

B. Advantages of the MLE Formulation

Since we use MLE to optimize unnormalized models, our objective inherits several nice properties. Here, we analyze the behavior of the estimator $\widehat{\theta} = \arg\min_\theta \mathcal{L}(\theta)$ for large sample sizes and assume that there exists an optimal solution θ^* such that $p(\mathbf{x},\theta^*) = p_{\text{data}}(\mathbf{x})$. To facilitate the comparison of different estimators, we introduce the concept of asymptotic relative efficiency (ARE) [55] below.

Definition 1: For any two estimators R and S with

$$\sqrt{n}\,(R-\theta^*) \rightsquigarrow N\left(0, r^2\right),\ \sqrt{n}\,(S-\theta^*) \rightsquigarrow N\left(0, s^2\right),$$

the ARE of S to R is defined as $ARE(S,R) = r^2/s^2$.

Remark: From the definition, we know that $ARE(S,R) < 1$ indicates estimator R is more efficient than estimator S.

Theorem 1: According to the property of MLE [56], the estimator $\widehat{\theta}$ enjoys the following guarantees:

1) *(Consistency):* The estimator $\widehat{\theta}$ converges in probability to θ^*, i.e., $\widehat{\theta} \xrightarrow{P} \theta^*$.
2) *(Asymptotically Normality):* $\sqrt{n}(\widehat{\theta} - \theta^*) \rightsquigarrow N(0, \widehat{\text{se}}^2)$, where $\widehat{\text{se}}$ can be computed analytically.
3) *(Asymptotically Optimality):* Denote that $\widetilde{\theta}$ is the output of any other estimator, then $ARE(\widetilde{\theta}, \widehat{\theta}) \leq 1$.

Remark: The last property implies that the MLE objective has the smallest variance, indicating it is more efficient than optimizing other objectives, e.g., the NCE objective.

C. The Convergence of the Proposed Method

Then, we examine the convergence of Algorithm 1. First, we define the sample complexity to measure the convergence rate, which is commonly used in stochastic optimization.

Definition 2: The sample complexity refers to the number of samples needed to find a point satisfying $\mathbb{E}[\|\nabla\mathcal{L}(\theta)\|] \leq \epsilon$ (ϵ-stationary), or $\mathbb{E}[\mathcal{L}(\theta) - \inf_\theta \mathcal{L}(\theta)] \leq \epsilon$ (ϵ-optimal).

For notation simplicity, we denote $g(\theta) = \mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$, $h(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[\log p_0(\mathbf{x}_i;\theta)]$, $f(\cdot) = \log(\cdot)$, estimator $g(\theta;\widetilde{\mathbf{z}}) = \frac{p_0(\widetilde{\mathbf{z}};\theta)}{q(\widetilde{\mathbf{z}})}$ and $h(\theta;\mathbf{z}) = -\log p_0(\mathbf{z};\theta)$, where $\widetilde{\mathbf{z}}$ and $\mathbf{z}$ are samples drawn from q and $\mathcal{D}_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$. Next, we make the following assumptions, which are commonly used in SCO problems [49], [57], [58], [59], [60].

Assumption 1: We assume that (i) the functions $f(\cdot)$, $g(\cdot)$, $h(\cdot)$ are Lipschitz continuous and smooth with respect to their inputs; (ii) $\mathcal{L}$ is lower bounded by $\mathcal{L}^*$.

Assumption 2: There exist constant $\sigma_g, \zeta_g, \zeta_h$ such that

$$\mathbb{E}_{\widetilde{\mathbf{z}}\sim q(\widetilde{\mathbf{z}})}\left[\left\|g(\theta;\widetilde{\mathbf{z}}) - g(\boldsymbol{\theta})\right\|^2\right] \leq \sigma_g^2,$$

$$\mathbb{E}_{\widetilde{\mathbf{z}}\sim q(\widetilde{\mathbf{z}})}\left[\left\|\nabla g(\theta;\widetilde{\mathbf{z}}) - \nabla g(\boldsymbol{\theta})\right\|^2\right] \leq \zeta_g^2,$$

$$\mathbb{E}_{\mathbf{z}\sim\mathcal{D}_n}\left[\left\|\nabla h(\theta;\mathbf{z}) - \nabla h(\theta)\right\|^2\right] \leq \zeta_h^2.$$

Remark: In Assumption 1, $f(\cdot)$ is Lipschitz and smooth in terms of its input when $p_0(\widetilde{\mathbf{z}};\theta)/q(\widetilde{\mathbf{z}}) \geq c$, where c is a positive constant. Assumption 2 assumes that the variances of the estimation for $g(\theta)$, $\nabla g(\theta)$ and $h(\theta)$ by sampling from the corresponding distributions are bounded. As a result, although these assumptions seem strong at first glance, they can be satisfied for many practical scenarios. Also, these assumptions may be relaxed to weaker generalized Lipschitz and affine variance [61], [62], [63] with more careful analysis.

We first derive an asymptotic convergence, demonstrating that Algorithm 1 can converge to a stationary point of $\mathcal{L}(\theta)$.

Theorem 2: Assume that the sequence of stepsizes satisfies $\sum_{t=1}^{\infty}\eta_t = +\infty, \sum_{t=1}^{\infty}\eta_t^2 < \infty$. Then with probability 1, accumulation point $(\theta', \mathbf{u}', \mathbf{v}')$ of the sequence $\{\theta_t, \mathbf{u}_t, \mathbf{v}_t\}$ generated by Algorithm 1 satisfies the following conditions:

$$\nabla\mathcal{L}(\theta') = 0, \quad u' = g(\theta'), \quad \mathbf{v}' = \nabla\mathcal{L}(\theta').$$

Next, we present the non-asymptotic convergence result.

Theorem 3: For non-convex function $\mathcal{L}(\cdot)$, by setting $\eta_t = \mathcal{O}(\epsilon^2), \beta_t = \gamma_t = \mathcal{O}(\epsilon^2)$, Algorithm 1 finds an ϵ-stationary point in $T = \mathcal{O}(\max\{\frac{1}{\epsilon^2}, \frac{(\sigma_g^2+\zeta_g^2+\zeta_h^2)}{\epsilon^4}\})$ iterations.

Notably, in the above analysis of Theorem 3, we need to know values of Lipschitz and smoothness constant, as well as variances $\sigma_g, \zeta_g, \zeta_h$, to appropriately set hyperparameters η_t, β_t, and γ_t in the algorithm. However, these values are typically difficult or even impossible to know in practice, making the proposed method impractical to use or rely on extensive hyperparameter tuning. To overcome this limitation, we develop a parameter-free algorithm that autonomously sets the learning rates and momentum parameters, eliminating the need for any problem-dependent variables. Specifically, we still use the framework of MECO (Algorithm 1), but design the momentum parameters β_t and γ_t to decrease over time in the style of

$$\beta_t = \gamma_t = t^{-1/2}. \tag{5}$$

The learning rate is then determined based on the momentum parameter and the historical values of the gradient estimator:

$$\eta_t = \left(\Sigma_{i=1}^{t}\left(\left\|\mathbf{v}_i\right\|^2/\beta_{i+1}^A\right)\right)^{-B}, \tag{6}$$

where we set $AB + B = 1$ and $B < 1/2$. In practice, we can simply set $B = 1/3$ and $A = 2$ to avoid further tuning. We call this method Parameter-Free MECO (PF-MECO). With such a design, the hyperparameter values are independent of the smoothness constant or variance, and PF-MECO maintains a similar convergence as that of the original MECO method.

Theorem 4: For non-convex function $\mathcal{L}(\cdot)$, PF-MECO finds an ϵ-stationary point in $T = \mathcal{O}(\frac{1+\sigma_g^2+\zeta_g^2+\zeta_h^2}{\epsilon^4})$ iterations.

Algorithm 2: MECO-v2.

Replace Step 8 of Algorithm 1 with the following:
1: Sample τ from $\{1, \ldots, T\}$ based on weights $\{w_t\}_{t\in[T]}$

D. Improved Convergence Under Stronger Conditions

Another significant contribution of our analysis is to demonstrate that by properly setting hyperparameters, Algorithm 1 can achieve a faster convergence rate when the objective satisfies the Polyak-Łojasiewicz (PL) condition [64] or convexity. We present the definition of the PL condition as follows:

Definition 3: $\mathcal{L}(\theta)$ satisfies the μ-PL condition if there exists $\mu > 0$ such that $2\mu(\mathcal{L}(\theta) - \mathcal{L}^*) \leq \|\nabla\mathcal{L}(\theta)\|^2$.

Remark: The PL condition has been shown to be satisfied for deep learning under over-parameterized networks by several prior works [65], [66]. Under this condition, a stationary point θ' becomes a global optimal solution for the objective $\mathcal{L}$. Consequently, our algorithm is guaranteed to asymptotically converge to an optimal solution $\widehat{\theta} = \arg\min\mathcal{L}(\theta)$ according to Theorem 2. Next, we show the improved complexity under the PL condition as stated below.

Theorem 5: When the objective satisfies μ-PL condition, by setting that $\gamma_{t+1} = \beta_{t+1} = \mathcal{O}(\max\{1, \mu\}\eta_t)$ and $1 - \mu\eta_t = \eta_t^2/\eta_{t-1}^2$, Algorithm 1 can find an ϵ-optimal solution in $T = \mathcal{O}(\max\{\frac{1}{\mu\sqrt{\epsilon}}, \frac{(\sigma_g^2+\zeta_g^2+\zeta_h^2)}{\mu\epsilon}, \frac{(\sigma_g^2+\zeta_g^2+\zeta_h^2)}{\mu^2\epsilon}\})$ iterations.

Remark: We emphasize that the above convergence for a single-loop algorithm is novel in SCO. Existing methods for SCO usually have to employ two-loop methods to obtain similar results under the PL condition [49], [52], [67].

We note that in Theorem 5, when the variance is zero, i.e., $\sigma_g^2 + \zeta_g^2 + \zeta_h^2 = 0$, the complexity would reduce to $\mathcal{O}(\frac{1}{\sqrt{\epsilon}})$. However, the optimal rate is known to be $\mathcal{O}(\ln(1/\epsilon))$ for this case, implying the rate we obtained is still improvable. To achieve this, we develop MECO-v2 method in Algorithm 2. The first difference lies in the setup of the learning rate η_t, which is divided into two phases. In the first phase, we use a constant learning rate $\eta_t = \eta_0$, and in the second phase, we use a decreasing learning rate on the order of $\mathcal{O}(1/(\mu t))$. Specifically, our new learning rate is written as

$$\eta_t = \begin{cases} \eta_0 & \text{if } t \leq T/2 \\ \frac{4}{\mu(t+k-T/2)} & \text{else} \end{cases},$$

where k is some positive constant. Furthermore, in the original MECO method, we choose τ uniformly at random from $\{1, \ldots, T\}$ and return θ_τ in the final step (Step 8 and 9in Algorithm 1), which means that $\mathbb{E}[\theta_\tau] = \frac{1}{T}\sum_{t=1}^{T}\theta_t$. Here, instead of choosing the decision variable uniformly, we assign weight w_t for each θ_t, so that $\mathbb{E}[\theta_\tau] = \sum_{t=1}^{T}\frac{w_t}{\sum_{t=1}^{T}w_t}\theta_t$. In our method, we only use the decision variable θ_t in the second phase, and the weight is inversely proportional to the learning rate, so that the latest solution would get more weight. The whole weighting strategy is designed as

$$w_t = \begin{cases} 0 & \text{if } t \leq T/2 \\ \frac{1}{\eta_t} & \text{else} \end{cases}.$$

Algorithm 3: OP-MECO.

1: **Input:** Initial points $(\theta_1, \mathbf{u}_1, \mathbf{v}_1)$, sequence $\{\eta_t, \gamma_t, \beta_t\}$
2: **for** time step $t = 1$ **to** T **do**
3: Sampling $\mathbf{z}_t$ from $\{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ and $\widetilde{\mathbf{z}}_t$ from $q(\mathbf{x})$
4: Update estimator $\mathbf{u}_t$ according to (7)
5: Update estimator $\mathbf{v}_t$ according to (8)
6: Update the weight: $\theta_{t+1} = \theta_t - \eta_t \mathbf{v}_t$
7: **end for**
8: Choose τ uniformly at random from $\{1, \ldots, T\}$
9: Return θ_τ

By such a design, we can obtain a better complexity, which is stated as follows.

Theorem 6: Our MECO-v2 method can find an ϵ-optimal solution within $\mathcal{O}(\max\{\frac{1}{\mu}\ln\frac{1}{\epsilon}, \frac{(\sigma_g^2+\zeta_g^2+\zeta_h^2)}{\mu\epsilon}, \frac{(\sigma_g^2+\zeta_g^2+\zeta_h^2)}{\mu^2\epsilon}\})$ iterations, when the objective satisfies μ-PL condition,.

Remark: In this case, we can achieve the optimal $\mathcal{O}(\ln(1/\epsilon))$ sample complexity when the variance vanishes.

Next, we also present the convergence for general convex objective functions, as stated below.

Theorem 7: When the objective is convex, our method can find an ϵ-optimal solution in $T = \mathcal{O}(\epsilon^{-3})$ iterations.

IV. Improved and Optimal Convergence for MECO

In the previous section, we show that our MECO algorithm can achieve a complexity of $\mathcal{O}(\epsilon^{-4})$, $\mathcal{O}(\epsilon^{-3})$, $\mathcal{O}(\mu^{-2}\epsilon^{-1})$ for non-convex, convex, and μ-PL objective functions, respectively. However, these rates are not optimal since they fail to match the corresponding lower bounds of the problem [68], [69]. In this section, we improve these complexities to their optimal levels by incorporating an additional error correction term. The proposed method is presented in Algorithm 3, called Optimal MECO (OP-MECO). The basic framework is similar to the MECO method, and the main difference is the STORM-based estimators [48] we used.

Specifically, we evaluate $\mathbb{E}_{\mathbf{x}\sim q}[\frac{p_0(\mathbf{x};\theta)}{q(\mathbf{x})}]$ by the estimator $\mathbf{u}_t$:

$$\mathbf{u}_t = (1-\gamma_t)\mathbf{u}_{t-1} + \gamma_t \frac{p_0(\widetilde{\mathbf{z}}_t;\theta_t)}{q(\widetilde{\mathbf{z}}_t)} + (1-\gamma_t)\left(\frac{p_0(\widetilde{\mathbf{z}}_t;\theta_t)}{q(\widetilde{\mathbf{z}}_t)} - \frac{p_0(\widetilde{\mathbf{z}}_t;\theta_{t-1})}{q(\widetilde{\mathbf{z}}_t)}\right), \quad (7)$$

where the last term can be viewed as an error correction term. This adjustment ensures that the error of our estimation would be reduced more quickly. When evaluating the overall gradient $\nabla\mathcal{L}(\theta)$, we employ a similar estimator $\mathbf{v}_t$ as:

$$\mathbf{v}_t = (1-\beta_t)\mathbf{v}_{t-1} - \frac{\nabla p_0(\mathbf{z}_t;\theta_t)}{p_0(\mathbf{z}_t;\theta_t)} + \frac{1}{\mathbf{u}_t}\frac{\nabla p_0(\widetilde{\mathbf{z}}_t;\theta_t)}{q(\widetilde{\mathbf{z}}_t)} + (1-\beta_t)\left(-\frac{\nabla p_0(\mathbf{z}_t;\theta_{t-1})}{p_0(\mathbf{z}_t;\theta_{t-1})} + \frac{1}{\mathbf{u}_{t-1}}\frac{\nabla p_0(\widetilde{\mathbf{z}}_t;\theta_{t-1})}{q(\widetilde{\mathbf{z}}_t)}\right). \quad (8)$$

In the first iteration, we set the estimator as $\mathbf{u}_1 = \frac{p_0(\widetilde{\mathbf{z}}_1;\theta_1)}{q(\widetilde{\mathbf{z}}_1)}$ and $\mathbf{v}_1 = -\frac{\nabla p_0(\mathbf{z}_1;\theta_1)}{p_0(\mathbf{z}_1;\theta_1)} + \frac{1}{\mathbf{u}_1}\frac{\nabla p_0(\widetilde{\mathbf{z}}_1;\theta_1)}{q(\widetilde{\mathbf{z}}_1)}$. In order to achieve a parameter-free design, we adopt a similar technique of PF-MECO, but with a smaller momentum parameter set as

$$\beta_t = \gamma_t = t^{-2/3}. \quad (9)$$

We still define the learning rate as

$$\eta_t = \left(\Sigma_{i=1}^t \left(\|\mathbf{v}_i\|^2/\beta_{i+1}^A\right)\right)^{-B}, \quad (10)$$

but with $2AB + B = 1$ and $B < 1/2$ this time. In practice, we can simply set $B = 1/3$ and $A = 1$. Next, we present the convergence of OP-MECO as stated below.

Theorem 8: For non-convex function $\mathcal{L}(\cdot)$, OP-MECO reaches an ϵ-stationary point in $T = \mathcal{O}(\epsilon^{-3})$ iterations.

Remark: The $\mathcal{O}(\epsilon^{-3})$ complexity is much better than the $\mathcal{O}(\epsilon^{-4})$ complexity of the original MECO algorithm, aligning with the lower bound for non-convex functions [69].

Additionally, with the error correction term, the OP-MECO method can be utilized to achieve improved convergence rates for convex functions or objectives that satisfy the PL condition.

Theorem 9: When the objective function satisfies the μ-PL condition, our OP-MECO algorithm can be used to attain an ϵ-optimal solution with a sample complexity of $\mathcal{O}(\mu^{-1}\epsilon^{-1})$.

Theorem 10: For convex functions, OP-MECO can be applied to find an ϵ-optimal solution with a complexity of $\mathcal{O}(\epsilon^{-2})$.

Remark: Notably, our results demonstrate optimal complexities when the objective is convex or satisfies the PL condition. For smooth and convex functions, our approach matches the $\mathcal{O}(\epsilon^{-2})$ lower bound for this problem [68]. In the context of the PL condition, there exists $\mathcal{O}(\mu^{-1}\epsilon^{-1})$ lower bound for the μ-strongly convex setting [68], which is a special case of the PL condition, thus proving the optimality of our method.

V. Choosing the Noise Distribution

Similar to Noise-Contrastive Estimation (NCE), our approach also relies on a noise distribution $q(\widetilde{\mathbf{z}})$. The difference is that the noise distribution only affects the convergence rate of our method, not the optimal solution. According to Theorem 2, our algorithm is guaranteed to converge to a stationary point or a globally optimal solution (under PL condition) of the MLE objective, as long as $q(\widetilde{\mathbf{z}})$ is positive whenever $p_0(\widetilde{\mathbf{z}};\theta)$ is positive, no matter what the noise distribution is. In contrast, for NCE, the noise distribution directly affects its objective function. It has been noted that if the noise distribution is too different from the data distribution, the classification problem becomes too easy, which prevents the model from learning much about the data structure [13]. In our method, the impact of $q(\widetilde{\mathbf{z}})$ on the convergence rate is through the variances of $g(\theta;\widetilde{\mathbf{z}})$ and $\nabla g(\theta;\widetilde{\mathbf{z}})$. From our theoretical results (e.g., Theorems 3 and 5), we can see that if the variance is zero, then the noise distribution has no impact on the convergence rate. We show the condition to achieve zero variance below.

Lemma 1: If $q(\widetilde{\mathbf{z}}) = p(\widetilde{\mathbf{z}};\theta)$, then $\sigma_g^2 = 0$ and $\zeta_g^2 = 0$.

Remark: However, the above choice is impractical, as sampling directly from $p(\widetilde{\mathbf{z}};\theta)$ is not easy, and the dependence of $q(\widetilde{\mathbf{z}})$ on θ would also make our convergence analysis fail. In practice, we can only hope $q(\widetilde{\mathbf{z}})$ is close to $p(\widetilde{\mathbf{z}};\theta)$.

TABLE II
RUNNING TIME OF EACH METHOD (UNIT: SECONDS)

NCE	NCE (NGD)	eNCE (NGD)	MCMC	Ours
176	181	169	1757	168

Besides, to ensure the partition function $\int p_0(\widetilde{\mathbf{z}};\theta)d\widetilde{\mathbf{z}}$ can be written as $\mathbb{E}_{\widetilde{\mathbf{z}}\sim q}[\frac{p_0(\widetilde{\mathbf{z}};\theta)}{q(\widetilde{\mathbf{z}})}]$, we should guarantee that $q(\widetilde{\mathbf{z}})>0$ whenever $p_0(\widetilde{\mathbf{z}};\theta)>0$. This is also required in NCE, which is not difficult to satisfy, since many continuous probability distributions can ensure their probability density functions are always positive, e.g., Gaussian, Laplace, and Cauchy distributions. Thus, our analysis suggests the following guidelines:

1) Choose a noise distribution $q(\widetilde{\mathbf{z}})$ that can be easily sampled and computed.
2) Ensure $q(\widetilde{\mathbf{z}})>0$ whenever $p_0(\widetilde{\mathbf{z}};\theta)>0$.
3) Select a noise distribution that is similar to $p(\widetilde{\mathbf{z}};\theta)$ or the data distribution $p_{\text{data}}(\widetilde{\mathbf{z}})$.

To satisfy the last two properties, we can sample the real data and add some noise to it. Specifically, to generate samples $\widetilde{\mathbf{z}}\sim q(\widetilde{\mathbf{z}})$, we first draw a sample $\mathbf{x}'\sim\mathcal{D}_n$, $\mathbf{z}'\sim\mathcal{N}(\boldsymbol{\mu},\Sigma)$, and then define $\widetilde{\mathbf{z}}=\mathbf{x}'+\mathbf{z}'$. Considering $\mathcal{D}_n=\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}$ and denoting the probability density function of $\mathcal{N}(\cdot;\boldsymbol{\mu},\Sigma)$ as $\kappa(\cdot)$, the noise distribution would be $q(\widetilde{\mathbf{z}})=\frac{1}{n}\sum_{i=1}^{n}\kappa(\mathbf{x}-\mathbf{x}_i)$, which can be effectively approximated within the mini-batch in practice. To ensure that the noise is similar to the data, we can fit the parameters μ and Σ by a subset of the training data. This can be readily accomplished using existing numpy functions, such as numpy.mean() and numpy.cov(). In our empirical studies, this strategy is beneficial for high-dimensional problems like image generation. For simpler tasks, such as density estimation and out-of-distribution detection, we can simply set the noise as a multivariate Gaussian distribution, which is very fast and easy to sample from. Since more complex noise would introduce additional computations, it is a trade-off in practice.

Finally, it is worth noting that the influence of the noise distribution can be mitigated by increasing the mini-batch size used for estimating $g(\theta)$ and $\nabla g(\theta)$. If the mini-batch size for noisy samples is B, then the variance σ_g^2 and ζ_g^2 will be scaled by B. Hence, the larger the batch size, the less impact of the noise distribution on the convergence rate.

VI. CASE STUDY: GAUSSIAN MEAN ESTIMATION

As pointed out by [15], the reason that NCE may fail to learn a good parameter is due to its flat loss landscape. To verify this claim, they employ one-dimensional Gaussian mean estimation as an example and show the slow convergence of NCE for this simple task. To demonstrate the effectiveness of our method, we use the same task to show the behavior of our loss and the proposed optimization method. Following their experimental setup, both the data and noise distributions are modeled as Gaussian distributions with mean θ^* and θ_q respectively, and the variances are both fixed as 1, i.e.,

$$p_{\text{data}}(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\theta^*)^2}{2}},\quad q(x)=\frac{1}{\sqrt{2\pi}}e^{-\frac{(x-\theta_q)^2}{2}}.$$

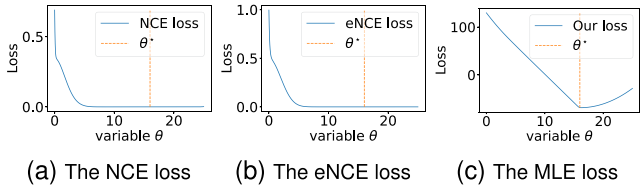

Fig. 1. The loss landscape of three objectives (the optimal parameter $\theta^*=16$).

Then, assume that the model is another Gaussian distribution with mean θ and variance 1, which is equivalent to setting $p_0(x;\theta)=e^{\theta x-\frac{1}{2}x^2}$. The goal is to learn the parameter θ, or $\tau(\theta)=(\theta,\frac{\theta^2}{2}+\log\sqrt{2\pi})$ for NCE.

First, we show the flatness of NCE loss below.

Proposition 1. (Proposition 4.2 in [15]): Denote $R=|\theta^*-\theta_q|$ and $\mathcal{J}(\cdot)$ is the NCE loss. Then, it holds that $\mathcal{J}(\tau)-\mathcal{J}(\tau^*)\leq R\exp(-R^2/8)\|\tau-\tau^*\|^2$, where τ^* is the optimal solution.

Remark: If θ^* and θ_q are not close enough, the difference between $J(\tau)$ and $J(\tau^*)$ will be extremely small when τ approaches τ^*, implying a flat landscape.

In contrast, it is not a problem for the MLE objective $\mathcal{L}(\cdot)$ that we optimize, since we have the following proposition.

Proposition 2: For 1-d Gaussian mean estimation, the MLE objective satisfies that $\mathcal{L}(\theta)-\mathcal{L}(\theta^*)=\frac{1}{2}\|\theta-\theta^*\|^2$.

Remark: Compared with NCE, the MLE objective does not have the extremely small factor $R\exp(-R^2/8)$, and thus has a much favorable loss landscape near the optimum.

To illustrate this difference more vividly, we plot the loss landscape of the NCE objective and the MLE objective, as well as the newly proposed eNCE objective [15]. Following the same setup as [15], we set $\theta_q=0$ and $\theta^*=16$. The results are depicted in Fig. 1. As can be seen, both the NCE and eNCE objectives are very flat near the optimal solution, while the MLE objective presents a much sharper landscape near the optimum. Moreover, we present the convergence rate of our method for this task, as stated below.

Proposition 3: For 1-d Gaussian mean estimation, Algorithm 1 ensures $\mathcal{L}(\theta)-\mathcal{L}^*\leq\epsilon$ after $T=\mathcal{O}(\epsilon^{-1})$ iterations.

Remark: Although [15] demonstrates that $\|\tau-\tau^*\|\leq\epsilon$ can be achieved within $\mathcal{O}(\epsilon^{-2})$ iterations, by using normalized gradient descent (NGD) for the NCE or eNCE objective, their assumptions are very strong. They assume the algorithm can access to the **exact** gradient, which is impossible in practice. In contrast, our analysis only relies on stochastic gradients. Also note that NCE optimized with standard gradient descent requires an exponential number of steps to find a reasonable parameter, according to [15].

Finally, we conduct experiments to compare different methods. We choose the mean square error (MSE) $\|\theta-\theta^*\|^2$ as the criterion and compare our method with NCE trained by SGD and NGD, MCMC training [2], and the eNCE objective trained with NGD. The results are presented in Fig. 2, which clearly illustrate that our method converges more quickly than other methods, and NCE is the slowest due to its flat loss landscape. We run each method for 100 steps, and we report

TABLE III
RESULTS ON SYNTHETIC DATA IN TERMS OF MMD (LOWER IS BETTER)

Method	*2spirals*	*8gaussians*	*checkerboard*	*circles*	*moons*	*swissroll*
NCE	3.253 ± 0.284	0.153 ± 0.095	1.956 ± 0.469	1.223 ± 0.154	5.178 ± 0.341	2.715 ± 0.249
NCE (NGD)	3.445 ± 0.287	0.177 ± 0.102	1.963 ± 0.488	1.270 ± 0.292	5.010 ± 0.349	2.585 ± 0.201
eNCE (NGD)	3.328 ± 0.332	0.257 ± 0.144	1.810 ± 0.218	1.183 ± 0.188	4.728 ± 0.399	2.975 ± 0.398
MCMC	3.060 ± 0.780	0.150 ± 0.035	1.654 ± 0.217	1.154 ± 0.294	4.722 ± 0.633	2.764 ± 0.670
Score Matching	3.268 ± 0.846	0.250 ± 0.076	2.167 ± 0.703	1.302 ± 0.272	4.826 ± 0.153	2.660 ± 0.513
Contrastive Divergence	3.245 ± 0.426	0.182 ± 0.085	1.987 ± 0.470	1.161 ± 0.410	4.716 ± 0.658	2.623 ± 0.203
Normalizing Flows	3.182 ± 0.664	0.136 ± 0.188	1.756 ± 0.462	1.163 ± 0.168	4.774 ± 0.237	2.691 ± 0.773
MECO	3.040 ± 0.199	0.132 ± 0.098	1.645 ± 0.251	1.075 ± 0.169	4.673 ± 0.519	2.566 ± 0.274
PF-MECO	3.065 ± 0.396	0.142 ± 0.068	1.797 ± 0.285	1.128 ± 0.063	**4.615 ± 0.490**	2.638 ± 0.251
OP-MECO	**3.027 ± 0.553**	**0.106 ± 0.040**	**1.623 ± 0.322**	**1.065 ± 0.199**	4.647 ± 0.349	**2.518 ± 0.669**

TABLE IV
RESULTS ON SYNTHETIC DATA IN TERMS OF FID (LOWER IS BETTER)

Method	*2spirals*	*8gaussians*	*checkerboard*	*circles*	*moons*	*swissroll*
NCE	0.103 ± 0.038	0.118 ± 0.026	0.178 ± 0.026	0.096 ± 0.018	0.114 ± 0.020	0.181 ± 0.041
NCE (NGD)	0.078 ± 0.024	0.143 ± 0.048	0.169 ± 0.023	0.103 ± 0.024	0.099 ± 0.029	0.171 ± 0.023
eNCE (NGD)	0.083 ± 0.040	0.128 ± 0.033	0.112 ± 0.029	0.085 ± 0.045	0.096 ± 0.015	0.212 ± 0.040
MCMC	0.072 ± 0.052	0.115 ± 0.038	0.125 ± 0.036	0.075 ± 0.042	0.109 ± 0.042	0.174 ± 0.028
Score Matching	0.109 ± 0.044	0.178 ± 0.086	0.166 ± 0.069	0.099 ± 0.026	0.117 ± 0.024	0.178 ± 0.062
Contrastive Divergence	0.089 ± 0.037	0.142 ± 0.051	0.121 ± 0.029	0.105 ± 0.039	0.110 ± 0.017	0.180 ± 0.025
Normalizing Flows	0.063 ± 0.029	0.130 ± 0.086	0.118 ± 0.078	0.085 ± 0.036	0.103 ± 0.045	0.175 ± 0.046
MECO	0.061 ± 0.032	0.104 ± 0.025	0.111 ± 0.024	0.065 ± 0.030	**0.094 ± 0.131**	0.163 ± 0.039
PF-MECO	0.071 ± 0.024	0.116 ± 0.060	0.123 ± 0.059	0.071 ± 0.034	0.105 ± 0.024	0.159 ± 0.056
OP-MECO	**0.057 ± 0.018**	**0.094 ± 0.049**	**0.099 ± 0.036**	**0.063 ± 0.020**	0.098 ± 0.019	**0.152 ± 0.038**

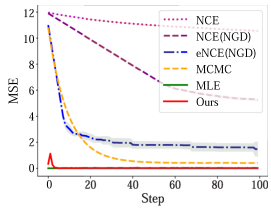

Fig. 2. Results for 1-d Gaussian mean estimation.

the running time of each method in Table II (unit: seconds), where the experiments are conducted on a personal laptop equipped with an Intel i7-10750H CPU and an NVIDIA 1650Ti GPU. It can be seen that the running time of NCE and our method is very similar, while the MCMC method is much slower.

VII. EXPERIMENTS

In this section, we conduct experiments on three different tasks, and compare our methods with NCE, MCMC training, Normalizing Flows [70], NCE and eNCE optimized by NGD, etc. For our method, we set the hyperparameters $\gamma = 0.1$ and $\beta = 0.9$. Although we do not tune γ and β, we show the results for different hyperparameters in the Appendix, available online (β within range $\{0.8, 0.9, 0.99\}$ and γ within the range $\{0.01, 0.1, 0.2\}$), which indicates that our method is not very sensitive to the choice of β and γ within a certain range. For MCMC training, the number of sampling steps is searched from the set $\{20, 50, 100\}$ and we use Langevin dynamics [71] as the sampling approach. For all tasks, we tune the initial learning rates from $\{1e{-}1, 1e{-}2, 1e{-}3, 1e{-}4\}$ and pick the best one. In this section, all methods are trained with the same training time. Experiments in Sections VII-A and VII-B are conducted on a personal laptop with an Intel i7-10750H CPU and an NVIDIA 1650Ti GPU, and experiments in Section VII-C are trained on four NVIDIA Tesla V100 GPUs. All the results in the experiment part are averaged over 20 runs.

A. Density Estimation on Synthetic Data

First, we focus on density estimation on synthetic data. Following the experimental setup of the previous literature [9], [72], we sample a set of 2D data points as the training set, according to some data distribution $p_{\text{data}}(\mathbf{x})$, visualized at the top of Fig. 3. Then we train unnormalized models $p_0(\mathbf{x};\theta) = e^{f_0(\mathbf{x};\theta)}$ to learn this distribution, where $f_0(\mathbf{x};\theta)$ is a multi-layer perceptron

TABLE V
OOD DETECTION RESULTS (AUROC↑, AUPRC↑ AND FRP80↓) FOR MODELS TRAINED ON CIFAR-10

Dataset		AUROC↑	AUPRC↑	FRP80↓
CIFAR10-Interp	NCE	0.6468 ± 0.0122	0.5455 ± 0.0053	0.4929 ± 0.0301
	NCE (NGD)	0.6748 ± 0.0187	0.6265 ± 0.1565	0.5392 ± 0.0271
	eNCE (NGD)	0.7451 ± 0.0296	0.7052 ± 0.0310	0.4052 ± 0.0586
	MCMC	0.7077 ± 0.0154	0.6787 ± 0.0119	0.5137 ± 0.0253
	Score Matching	0.5329 ± 0.0024	0.5304 ± 0.0029	0.7706 ± 0.0047
	Contrastive Divergence	0.7735 ± 0.0161	0.7371 ± 0.0106	0.3733 ± 0.0382
	Normalizing Flows	0.7689 ± 0.0340	0.7461 ± 0.0236	0.3652 ± 0.0648
	MECO	0.8019 ± 0.0323	0.7679 ± 0.0402	0.3185 ± 0.0621
	PF-MECO	0.8108 ± 0.0086	0.7414 ± 0.0071	0.2785 ± 0.0216
	OP-MECO	**0.8265 ± 0.0071**	**0.8778 ± 0.0061**	**0.2666 ± 0.0236**
SVHN	NCE	0.6425 ± 0.0107	0.3436 ± 0.0052	0.5592 ± 0.0286
	NCE (NGD)	0.6593 ± 0.0035	0.4361 ± 0.0083	0.7284 ± 0.0040
	eNCE (NGD)	0.6612 ± 0.0075	0.4633 ± 0.0109	0.6910 ± 0.0134
	MCMC	0.5359 ± 0.0078	0.2674 ± 0.0048	0.6327 ± 0.0037
	Score Matching	0.5601 ± 0.0068	0.3237 ± 0.0010	0.8346 ± 0.0060
	Contrastive Divergence	0.6103 ± 0.0040	0.3057 ± 0.0018	0.3758 ± 0.0025
	Normalizing Flows	0.6312 ± 0.0564	0.4831 ± 0.0125	0.6835 ± 0.0446
	MECO	0.7843 ± 0.0057	0.5134 ± 0.0076	0.3255 ± 0.2081
	PF-MECO	0.7740 ± 0.0048	0.5122 ± 0.0081	0.3554 ± 0.0039
	OP-MECO	**0.8829 ± 0.0047**	**0.7585 ± 0.0136**	**0.0995 ± 0.0111**
CIFAR-100	NCE	0.5323 ± 0.0199	0.5129 ± 0.0095	0.7122 ± 0.0216
	NCE (NGD)	0.5513 ± 0.0218	0.5458 ± 0.0160	0.7832 ± 0.0306
	eNCE (NGD)	0.5034 ± 0.0069	0.5248 ± 0.0158	0.8263 ± 0.0095
	MCMC	0.5669 ± 0.0059	0.5604 ± 0.0138	0.7217 ± 0.0052
	Score Matching	0.5732 ± 0.0014	0.5343 ± 0.0070	0.6943 ± 0.0059
	Contrastive Divergence	0.5276 ± 0.0098	0.5136 ± 0.0120	0.6642 ± 0.0111
	Normalizing Flows	0.5355 ± 0.0396	0.5364 ± 0.0206	0.7981 ± 0.0735
	MECO	0.6044 ± 0.0049	0.6262 ± 0.0079	0.5795 ± 0.0113
	PF-MECO	0.5202 ± 0.0065	0.5242 ± 0.0029	0.8034 ± 0.0092
	OP-MECO	**0.7368 ± 0.0167**	**0.7019 ± 0.0237**	**0.4108 ± 0.0343**
LSUN-C	NCE	0.5356 ± 0.0006	0.4995 ± 0.0005	0.6876 ± 0.0028
	NCE (NGD)	0.6049 ± 0.0189	0.5817 ± 0.0083	0.7119 ± 0.0408
	eNCE (NGD)	0.5470 ± 0.0039	0.6840 ± 0.0042	0.9450 ± 0.0028
	MCMC	0.5359 ± 0.0153	0.5107 ± 0.0157	0.7436 ± 0.0131
	Score Matching	0.5419 ± 0.0057	0.5221 ± 0.0030	0.7372 ± 0.0097
	Contrastive Divergence	0.5044 ± 0.0054	0.4980 ± 0.0060	0.6248 ± 0.0022
	Normalizing Flows	0.5769 ± 0.0457	**0.6950 ± 0.0389**	0.7340 ± 0.0650
	MECO	0.6944 ± 0.0061	0.6267 ± 0.0046	0.5198 ± 0.0126
	PF-MECO	0.6489 ± 0.0087	0.5932 ± 0.0059	0.5511 ± 0.0236
	OP-MECO	**0.7199 ± 0.0192**	0.6726 ± 0.0265	**0.4489 ± 0.0555**

TABLE VI
AVERAGE NEGATIVE LOG-LIKELIHOOD (SMALLER IS BETTER)

NCE	NCE (NGD)	eNCE (NGD)	MCMC-OT	CF-EBM	Normalizing Flows	MECO	PF-MECO	OP-MECO
1.457 ± 0.003	1.618 ± 0.010	1.600 ± 0.044	1.441 ± 0.012	1.864 ± 0.004	1.463 ± 0.025	1.394 ± 0.007	1.422 ± 0.012	**1.391 ± 0.005**

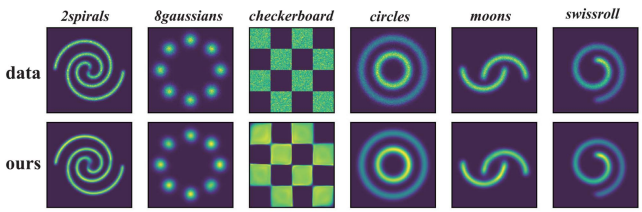

Fig. 3. Visualization of density on synthetic datasets.

(MLP) with 3 hidden layers and 300 units per layer. For NCE, eNCE, and our method, the noise distribution is selected as a multivariate Gaussian distribution, whose mean and variance are fitted on the training set.

To quantify the performance of different methods, we adopt the maximum mean discrepancy (MMD) [73] as the criterion. The MMD metric is widely used to compare different distributions, and a lower MMD value indicates that the two distributions are more similar. We sample 10,000 points from the data distribution and the learned model, and the computed MMD metric is shown in Table III. As can be seen, our OP-MECO method enjoys the lowest MMD in all methods for most cases, indicating the superiority of the proposed method. Also, we report the Frechet Inception Distance (FID) score of each method, which is another widely used measure in comparing distributions. The results are presented in Table IV, and our OP-MECO method enjoys better FID scores than other algorithms (smaller is better). Besides, we compare the estimated density and the ground-truth in Fig. 3, showing that our method learns the density accurately in most cases. Finally, we investigate the behavior of our MECO algorithm with different values of γ and β in Appendix N, available online.

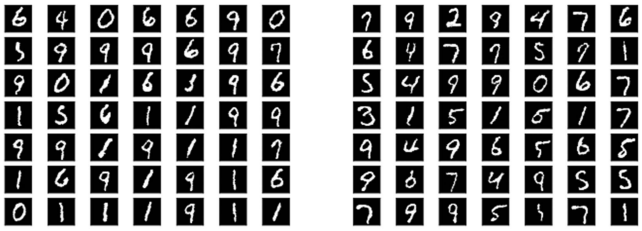

Fig. 4. Generated digits using MECO (left) and OP-MECO (right) via MCMC sampling.

B. Out-of-Distribution Detection

Next, we experiment on out-of-distribution (OOD) detection, which is an important measure of the density estimation quality. For this task, we choose CIFAR-10 [74] as the in-distribution data. We use the energy-based model as our unnormalized model, by setting $p_0(\mathbf{x};\theta) = e^{f_0(\mathbf{x};\theta)}$, where $f_0(\mathbf{x};\theta)$ is a 40-layer WideResNet [75]. The noise distribution for NCE, eNCE, and our method is selected as the multivariate Gaussian distribution which is fitted on the training set, and we use Adam to optimize. For the OOD test dataset, we use four common benchmarks: CIFAR-10 Interp, SVHN [76], CIFAR-100 [74], and LSUN [77]. We decide whether a test sample $\mathbf{x}$ is anomalous or not by computing the density $p_0(\mathbf{x};\theta)$, where a higher value indicates the test sample is more likely to be a normal sample.

To measure the performance of different methods, we follow previous works [11], [78], [79] to choose three metrics to compare: (1) area under the receiver operating characteristic curve (AUROC↑); (2) area under the precision-recall curve (AUPRC↑); and (3) false positive rate at 80% true positive rate (FPR80↓), where the arrow indicates the direction of improvement of the metrics. These three metrics are commonly used for evaluating OOD detection methods, and the results are reported in Table V. As can be seen, OP-MECO performs the best under three different criteria in all cases, implying the effectiveness of the proposed method.

C. Learning on Real Image Dataset

Finally, we test our method on two real image datasets, MNIST [80] and CIFAR-10 [74]. We describe the setup and the results of MNIST in this subsection, and the results of CIFAR-10 can be found in the Appendix, available online. For the MNIST task, following the same setup in TRE [81], the model takes the form of $p_0(\mathbf{x};\theta) = e^{f_0(\mathbf{x};\theta)}$, where $f_0(\mathbf{x};\theta)$ is the 18-layer ResNet [82]. For NCE, eNCE, and our method, we try three different noise distributions: 1) the empirical distribution $\mathcal{D}_n = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$; 2) multivariate Gaussian distribution, with mean and covariance fitted by the training data; 3) mixture of distribution $\mathcal{D}_n$ and a multivariate Gaussian distribution. We find the last one is the best choice for all methods. We use Adam to optimize, and generate new samples via MCMC sampling after training. The generated samples of our method with different noises are shown in the Appendix, available online, and the result with the third noise distribution is presented in Fig. 4.

We also evaluate the learned model via estimated average negative log-likelihood (bits per dimension) on the testing sets in Table VI, which is computed by the annealed importance sampling (AIS) [83]. We also compare our method with some other methods, including MCMC-OT [84] and CF-EBM [85]. As can be seen, our method attains a better likelihood than other methods, indicating that the proposed method performs better in this task.

VIII. Conclusion

In this paper, we investigate the problem of learning unnormalized models by maximum likelihood estimation. By introducing a noise distribution, we cast the problem as compositional optimization and utilize stochastic algorithms to solve it. Specifically, we first introduce the MECO algorithm that achieves sub-optimal convergence rates for non-convex, convex, and PL functions. Then, we design a parameter-free variant of MECO, reducing the burden on hyperparameter tuning. Next, we develop an optimal version of the MECO algorithm, obtaining optimal convergence for different types of functions. By analyzing the relationship between convergence rate and the choice of noise distribution, we give some suggestions for selecting an appropriate noise distribution. Besides, we employ one-dimensional Gaussian mean estimation as a case study to show the better loss landscape of our loss compared with the NCE loss, and the fast convergence of the proposed method. Finally, experiments on practical problems demonstrate the effectiveness of the proposed method.

References

[1] Y. LeCun, S. Chopra, R. Hadsell, A. Ranzato, and F. J. Huang, "A tutorial on energy-based learning," in *Predicting Structured Data*, Cambridge, MA, USA: MIT Press, 2006.

[2] Y. Du and I. Mordatch, "Implicit generation and modeling with energy based models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 3603–3613.

[3] L. Yu, Y. Song, J. Song, and S. Ermon, "Training deep energy-based models with f-divergence minimization," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 10957–10967.

[4] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, M. Norouzi, and K. Swersky, "Your classifier is secretly an energy based model and you should treat it like one," in *Proc. Int. Conf. Learn. Representations*, 2020.

[5] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[6] T. Tieleman, "Training restricted boltzmann machines using approximations to the likelihood gradient," in *Proc. 25th Int. Conf. Mach. Learn.*, 2008, pp. 1064–1071.

[7] E. Nijkamp, M. Hill, T. Han, S.-C. Zhu, and Y. N. Wu, "On the anatomy of MCMC-based maximum likelihood learning of energy-based models," in *Proc. 33th AAAI Conf. Artif. Intell.*, 2019, pp. 5272–5280.

[8] G. C. Christian and P. Robert, *Monte Carlo Statistical Methods*. Berlin, Germany: Springer, 2004.

[9] W. Grathwohl, K.-C. Wang, J.-H. Jacobsen, D. Duvenaud, and R. Zemel, "Learning the stein discrepancy for training and evaluating energy-based models without sampling," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 3732–3747.

[10] G. Ruiqi, E. Nijkamp, D. Kingma, Z. Xu, A. Dai, and Y. Wu, "Flow contrastive estimation of energy-based models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 7518–7528.

[11] C. Geng, J. Wang, Z. Gao, J. Frellsen, and S. Hauberg, "Bounds all around: Training energy-based models with bidirectional bounds," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 19808–19821.

[12] E. Nijkamp, M. Hill, S.-C. Zhu, and Y. N. Wu, "Learning non-convergent non-persistent short-run MCMC toward energy-based model," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 5233–5243.
[13] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics," *J. Mach. Learn. Res.*, vol. 13, no. 11, pp. 307–361, 2012.
[14] I. Goodfellow et al., "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[15] B. Liu, E. Rosenfeld, P. K. Ravikumar, and A. Risteski, "Analyzing and improving the optimization landscape of noise-contrastive estimation," in *Proc. Int. Conf. Learn. Representations*, 2022.
[16] W. Jiang, J. Qin, L. Wu, C. Chen, T. Yang, and L. Zhang, "Learning unnormalized statistical models via compositional optimization," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 15105–15124.
[17] M. U. Gutmann and A. Hyvärinen, "Noise-contrastive estimation: A new estimation principle for unnormalized statistical models," in *Proc. 13th Int. Conf. Artif. Intell. Statist.*, 2010, pp. 297–304.
[18] A. Mnih and K. Kavukcuoglu, "Learning word embeddings efficiently with noise-contrastive estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2265–2273.
[19] R. D. Hjelm et al., "Learning deep representations by mutual information estimation and maximization," in *Proc. Int. Conf. Learn. Representations*, 2019.
[20] O. Henaff, "Data-efficient image recognition with contrastive predictive coding," in *Proc. 37th Int. Conf. Mach. Learn.*, 2020, pp. 4182–4192.
[21] Y. Tian, D. Krishnan, and P. Isola, *Contrastive Multiview Coding*. Berlin, Germany: Springer, 2020.
[22] L. Kong, C. de Masson d'Autume, L. Yu, W. Ling, Z. Dai, and D. Yogatama, "A mutual information maximization perspective of language representation learning," in *Proc. Int. Conf. Learn. Representations*, 2020.
[23] H. Lee, C. Pabbaraju, A. P. Sevekari, and A. Risteski, "Pitfalls of gaussians as a noise distribution in NCE," in *Proc. Int. Conf. Learn. Representations*, 2023.
[24] A. J. Bose, H. Ling, and Y. Cao, "Adversarial contrastive estimation," in *Proc. 56th Annu. Meeting Assoc. Comput. Linguistics*, 2018, pp. 1021–1032.
[25] C. Ceylan and M. U. Gutmann, "Conditional noise-contrastive estimation of unnormalised models," in *Proc. 35th Int. Conf. Mach. Learn.*, 2018, pp. 726–734.
[26] J. J. Ryu, A. Shah, and G. W. Wornell, "A unified view on learning unnormalized distributions via noise-contrastive estimation," in *Proc. 42nd Int. Conf. Mach. Learn.*, 2025, pp. 52444–52474.
[27] A. Hyvärinen, "Estimation of non-normalized statistical models by score matching," *J. Mach. Learn. Res.*, vol. 6, no. 24, pp. 695–709, 2005.
[28] Y. Song and S. Ermon, "Generative modeling by estimating gradients of the data distribution," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 11895–11907.
[29] T. Pang, K. Xu, C. Li, Y. Song, S. Ermon, and J. Zhu, "Efficient learning of generative models via finite-difference score matching," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 19175–19188.
[30] R. Gao, Y. Lu, J. Zhou, S.-C. Zhu, and Y. N. Wu, "Learning generative convnets via multi-grid modeling and sampling," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 9155–9164.
[31] T. Han, E. Nijkamp, X. Fang, M. Hill, S. Zhu, and Y. N. Wu, "Divergence triangle for joint training of generator model, energy-based model, and inferential model," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2019, pp. 8670–8679.
[32] J. Xie, Y. Lu, S.-C. Zhu, and Y. N. Wu, "A theory of generative convnet," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, 2016, pp. 2635–2644.
[33] A. Bakhtin, Y. Deng, S. Gross, M. Ott, M. Ranzato, and A. Szlam, "Residual energy-based models for text," *J. Mach. Learn. Res.*, vol. 22, no. 40, pp. 1–41, 2021.
[34] Y. Song, S. Garg, J. Shi, and S. Ermon, "Sliced score matching: A scalable approach to density and score estimation," in *Proc. 35th Conf. Uncertainty Artif. Intell.*, 2019, pp. 574–584.
[35] T. Haarnoja, H. Tang, P. Abbeel, and S. Levine, "Reinforcement learning with deep energy-based policies," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 1352–1361.
[36] D. P. Kingma and Y. Cun, "Regularized estimation of image statistics by score matching," in *Proc. Adv. Neural Inf. Process. Syst.*, 2010, pp. 1126–1134.
[37] P. Vincent, "A connection between score matching and denoising autoencoders," *Neural Comput.*, vol. 23, no. 7, pp. 1661–1674, 2011.
[38] A. Mnih and Y. W. Teh, "A fast and simple algorithm for training neural probabilistic language models," in *Proc. 29th Int. Conf. Int. Conf. Mach. Learn.*, 2012, pp. 419–426.
[39] U. Grenander and M. I. Miller, "Representations of knowledge in complex systems," *J. Roy. Stat. Society: Ser. B*, vol. 56, no. 4, pp. 549–581, 2018.
[40] R. Neal, "MCMC using hamiltonian dynamics," in *Handbook of Markov Chain Monte Carlo*, London, U.K.: Chapman and Hall/CRC, 2012.
[41] J. Xie, Y. Lu, R. Gao, and Y. N. Wu, "Cooperative learning of energy-based model and latent variable model via MCMC teaching," in *Proc. AAAI Conf. Artif. Intell.*, 2018, pp. 4292–4301.
[42] X. Jianwen, Y. Lu, S. Zhu, and Y. Wu, "Cooperative training of descriptor and generator networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 42, no. 1, pp. 27–45, Jan. 2020.
[43] M. Wang, E. X. Fang, and H. Liu, "Stochastic compositional gradient descent: Algorithms for minimizing compositions of expected-value functions," *Math. Program.*, vol. 161, no. 1/2, pp. 419–449, 2017.
[44] M. Wang, J. Liu, and E. X. Fang, "Accelerating stochastic composition optimization," *J. Mach. Learn. Res.*, vol. 18, pp. 105:1–105:23, 2017.
[45] S. Ghadimi, A. Ruszczynski, and M. Wang, "A single timescale stochastic approximation method for nested stochastic optimization," *SIAM J. Optim.*, vol. 30, no. 1, pp. 960–979, 2020.
[46] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takac, "SARAH: A novel method for machine learning problems using stochastic recursive gradient," in *Proc. 34th Int. Conf. Mach. Learn.*, 2017, pp. 2613–2621.
[47] C. Fang, C. J. Li, Z. Lin, and T. Zhang, "Spider: Near-optimal non-convex optimization via stochastic path integrated differential estimator," 2018, arXiv: *1807.01695*.
[48] A. Cutkosky and F. Orabona, "Momentum-based variance reduction in non-convex SGD," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 15210–15219.
[49] J. Zhang and L. Xiao, "A stochastic composite gradient method with incremental variance reduction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 9075–9085.
[50] T. Chen, Y. Sun, and W. Yin, "Solving stochastic compositional optimization is nearly as easy as solving stochastic optimization," *IEEE Trans. Signal Process.*, vol. 69, pp. 4937–4948, 2021.
[51] Q. Qi, Z. Guo, Y. Xu, R. Jin, and T. Yang, "An online method for a class of distributionally robust optimization with non-convex objectives," 2021, arXiv: *2006.10138*.
[52] W. Jiang, B. Wang, Y. Wang, L. Zhang, and T. Yang, "Optimal algorithms for stochastic multi-level compositional optimization," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 10195–10216.
[53] W. Jiang, S. Yang, W. Yang, Y. Wang, Y. Wan, and L. Zhang, "Projection-free variance reduction methods for stochastic constrained multi-level compositional optimization," in *Proc. 41st Int. Conf. Mach. Learn.*, 2024, pp. 21962–21987.
[54] W. Jiang, S. Yang, Y. Wang, T. Yang, and L. Zhang, "Revisiting stochastic multi-level compositional optimization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 47, no. 7, pp. 5613–5624, Jul. 2025.
[55] A. W. V. D. Vaart, *Asymptotic Statistics*. Cambridge, U.K.: Cambridge Univ. Press, 1998.
[56] L. Wasserman, *All of Statistics*. Berlin, Germany: Springer, 2004.
[57] M. Wang, J. Liu, and E. Fang, "Accelerating stochastic composition optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 1714–1722.
[58] J. Zhang and L. Xiao, "Multilevel composite stochastic optimization via nested variance reduction," *SIAM J. Optim.*, vol. 31, no. 2, pp. 1131–1157, 2021.
[59] B. Wang and T. Yang, "Finite-sum coupled compositional stochastic optimization: Theory and applications," in *Proc. 39th Int. Conf. Mach. Learn.*, 2022, pp. 23292–23317.
[60] W. Jiang, S. Yang, Y. Wang, and L. Zhang, "Adaptive variance reduction for stochastic optimization under weaker assumptions," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 22047–22080.
[61] M. Faw, I. Tziotis, C. Caramanis, A. Mokhtari, S. Shakkottai, and R. Ward, "The power of adaptivity in SGD: Self-tuning step sizes with unbounded gradients and affine variance," in *Proc. 35th Annu. Conf. Learn. Theory*, 2022, pp. 313–355.
[62] Z. Chen, Y. Zhou, Y. Liang, and Z. Lu, "Generalized-smooth nonconvex optimization is as efficient as smooth nonconvex optimization," in *Proc. 40th Int. Conf. Mach. Learn.*, 2023, pp. 5396–5427.
[63] W. Jiang, S. Yang, W. Yang, and L. Zhang, "Efficient sign-based optimization: Accelerating convergence via variance reduction," in *Proc. Adv. Neural Inf. Process. Syst.*, 2024, pp. 33891–33932.

[64] H. Karimi, J. Nutini, and M. Schmidt, "Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition," in *Proc. Mach. Learn. Knowl. Discov. Databases*, 2016, pp. 795–811.
[65] Z. Allen-Zhu, Y. Li, and Z. Song, "A convergence theory for deep learning via over-parameterization," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 242–252.
[66] S. S. Du, X. Zhai, B. Poczos, and A. Singh, "Gradient descent provably optimizes over-parameterized neural networks," in *Proc. Int. Conf. Learn. Representations*, 2019.
[67] Q. Qi, Y. Luo, Z. Xu, S. Ji, and T. Yang, "Stochastic optimization of areas under precision-recall curves with provable convergence," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 1752–1765.
[68] A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright, "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 3235–3249, May 2012.
[69] Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, and B. E. Woodworth, "Lower bounds for non-convex stochastic optimization," 2019, arXiv: *1912.02365*.
[70] L. Dinh, D. Krueger, and Y. Bengio, "Nice: Non-linear independent components estimation," in *Proc. Int. Conf. Learn. Representations*, 2015.
[71] M. Welling and Y. W. Teh, "Bayesian learning via stochastic gradient Langevin dynamics," in *Proc. 28th Int. Conf. Mach. Learn.*, 2011, pp. 681–688.
[72] M. Liu, H. Liu, and S. Ji, "Gradient-guided importance sampling for learning binary energy-based models," 2022, arXiv: *2210.05782*.
[73] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola, "A kernel two-sample test," *J. Mach. Learn. Res.*, vol. 13, no. 25, pp. 723–773, 2012.
[74] A. Krizhevsky, "Learning multiple layers of features from tiny images," Masters Thesis, Deptartment of Computer Science, University of Toronto, 2009.
[75] S. Zagoruyko and N. Komodakis, "Wide residual networks," in *Proc. Brit. Mach. Vis. Conf.*, 2016, pp. 87.1–87.12.
[76] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng, "Reading digits in natural images with unsupervised feature learning," in *Proc. Adv. Neural Inf. Process. Syst. Workshop Deep Learn. Unsupervised Feature Learn.*, 2011.
[77] F. Yu, A. Seff, Y. Zhang, S. Song, T. Funkhouser, and J. Xiao, "LSUN: Construction of a large-scale image dataset using deep learning with humans in the loop," 2015, arXiv: *1506.03365*.
[78] J. Ren et al., "Likelihood ratios for out-of-distribution detection," in *Proc. Adv. Neural Inf. Process. Syst.*, 2019, pp. 14680–14691.
[79] J. D. Havtorn, J. Frellsen, S. Hauberg, and L. Maaløe, "Hierarchical VAEs know what they don't know," in *Proc. 38th Int. Conf. Mach. Learn.*, 2021, pp. 4117–4128.
[80] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," in *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.
[81] B. Rhodes, K. Xu, and M. U. Gutmann, "Telescoping density-ratio estimation," in *Proc. Adv. Neural Inf. Process. Syst.*, 2020, pp. 4905–4916.
[82] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
[83] C. Nash and C. Durkan, "Autoregressive energy machines," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 1735–1744.
[84] D. An, J. Xie, and P. Li, "Learning deep latent variable models by short-run MCMC inference with optimal transport correction," in *IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2021, pp. 15410–15419.
[85] Y. Zhao, J. Xie, and P. Li, "Learning energy-based generative models via coarse-to-fine expanding and sampling," in *Proc. Int. Conf. Learn. Representations*, 2021.
[86] Z. Guo, Y. Xu, W. Yin, R. Jin, and T. Yang, "On stochastic moving-average estimators for non-convex optimization," 2021, arXiv: *2104.14840*.
[87] K. Y. Levy, A. Kavis, and V. Cevher, "STORM : Fully adaptive SGD with recursive momentum for nonconvex optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2021, pp. 20571–20582.
[88] W. Jiang, G. Li, Y. Wang, L. Zhang, and T. Yang, "Multi-block-single-probe variance reduced estimator for coupled compositional optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2022, pp. 32499–32511.
[89] W. Grathwohl, "Joint energy models," 2020. [Online]. Available: https://github.com/wgrathwohl/JEM
[90] M. Seitzer, "PyTorch-FID: FID Score for PyTorch," 2020. [Online]. Available: https://github.com/mseitzer/pytorch-fid

Wei Jiang received the BE degree from the Department of Computer Science and Technology, Xi'an Jiaotong University, in 2020, and the PhD degree from the Department of Computer Science and Technology, Nanjing University, China, in 2025. He is currently a professor with the School of Computer Science and Engineering, Nanjing University of Science and Technology, China. His research interests include machine learning and stochastic optimization.

Jiayu Qin received the BS degree from Fudan University, China. He is currently working toward the PhD degree in computer science and engineering department with the University at Buffalo, State University of New York, under the supervision of Prof. Changyou Chen. His research interests include machine learning and representation learning.

Lingyu Wu received the BE degree from the School of Artificial Intelligence, Nanjing University, China, in 2023. He is currently working toward the MS degree with the School of Artificial Intelligence, Nanjing University, China. His research interests include machine learning and continual learning.

Changyou Chen received the BS and MS degrees from Fudan University and the PhD degree from Australian National University. He is currently an associate professor with the CSE Department of the University at Buffalo, State University of New York. Previously, he was an assistant professor with the University at Buffalo, a research assistant professor and a postdoctoral associate with the Department of Electrical and Computer Engineering at Duke University. His research interests include generative models, and large-scale Bayesian sampling and inference.

Tianbao Yang (Senior Member, IEEE) is a professor and Herbert H. Richardson faculty fellow with the CSE Department of Texas A&M University, where he directs the lab of Optimization for Machine learning and AI (OptMAI Lab). His research interests include optimization, machine learning and AI with applications in computer vision, NLP, trustworthy AI and medicine. Before joining TAMU, he was an assistant professor and then tenured Dean's Excellence associate professor with the Computer Science Department of the University of Iowa from 2014 to 2022.

Lijun Zhang (Senior Member, IEEE) received the BE and PhD degrees in software engineering and computer science from Zhejiang University, China, in 2007 and 2012, respectively. He is currently a professor with the School of Artificial Intelligence, Nanjing University, China. Prior to joining Nanjing University, he was a postdoctoral researcher with the Department of Computer Science and Engineering, Michigan State University, USA. His research interests include machine learning and optimization.