# OBJECT REPRESENTATIONS AS EQUILIBRIA: TRAINING ITERATIVE INFERENCE ALGORITHMS WITH IMPLICIT DIFFERENTIATION

**Anonymous authors**
Paper under double-blind review

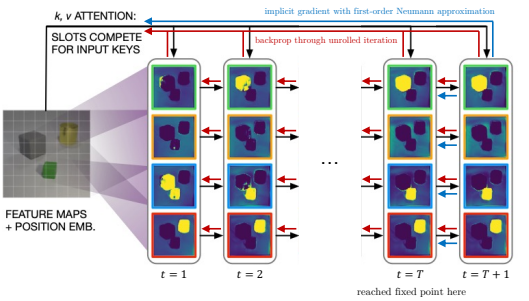
## ABSTRACT

Deep generative models, particularly those that aim to factorize the observations into discrete entities (such as objects), must often use iterative inference procedures that break symmetries among equally plausible explanations for the data. Such inference procedures include variants of the expectation-maximization algorithm and structurally resemble clustering algorithms in a latent space. However, combining such methods with deep neural networks necessitates differentiating through the inference process, which can make optimization exceptionally challenging. In this work, we observe that such iterative inference methods can be made differentiable by means of the implicit function theorem, and develop an implicit differentiation approach that improves the stability and tractability of training such models by decoupling the forward and backward passes. This connection enables us to apply recent advances in optimizing implicit layers to not only improve the stability and optimization of the slot attention module in SLATE, a state-of-the-art method for learning entity representations, but do so with constant space and time complexity in backpropagation and only one additional line of code.

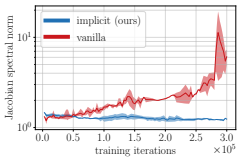
## 1 INTRODUCTION

Many generative models assume each observed datapoint $x$ is generated from a *set* $\boldsymbol{\lambda}$ of latent variables and parameters, where the ordering of these variables does not matter. In mixture models, for example, a single datapoint would have the same probability if the indices of the latent components were simply relabeled with a different permutation. However, this symmetry results in many equally valid explanations of the same datapoint, making the inference problem ill-posed. This phenomenon is especially relevant when learning representations of objects in visual scenes, where treating latent variables symmetrically enables the object representations to be composed in novel ways.

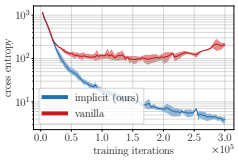
In these models, inference is typically done by breaking symmetry via a random initial guess $\boldsymbol{\lambda}_0$ and then iteratively updating the value of $\boldsymbol{\lambda}$, with variants of the classical expectation maximization (EM) algorithm (Dempster et al., 1977) being prominent examples. Recent works have replaced the analytical updates of EM with a neural network $f$ that directly computes the update as $\boldsymbol{\lambda} = f(\boldsymbol{\lambda}, x)$. These **iterative amortized inference algorithms** (Marino et al., 2018a) have predominantly been applied to learning




(a) Implicit differentiation of slot attention




(b) Jacobian norm




(c) Validation loss

Figure 1: **Overview.** We propose to train the slot attention model (1a), whose figure is adapted from Locatello et al. (2020), with implicit differentiation. Our approach leads to more stable training (1b) and substantially lower validation loss (1c) compared to vanilla slot attention.

object-centric representations of visual scenes (Greff et al., 2017; Van Steenkiste et al., 2018; Greff et al., 2019; 2020; Veerapaneni et al., 2020; Locatello et al., 2020; Kipf et al., 2021; Zoran et al., 2021; Singh et al., 2021). All are trained by differentiating through the unrolled iterations of $f$.

Despite their conceptual elegance, it has been difficult to scale iterative amortized inference methods beyond modeling simple static scenes or short video sequences because differentiating through the unrolled forward iteration makes training unstable. Fig. 5 shows that the spectral norm of the Jacobian of $f$ gradually increases over the course of training, which has empirically been observed to cause training instabilities (Bai et al., 2021). Such instabilities result in sensitivity to hyperparameter choices (e.g., number of inference iterations) and have motivated adding optimization tricks such as gradient clipping, learning rate warm-up, and learning rate decay, all of which make such models more complex and harder to use, restrict the model from optimizing its learning objective fully, and only temporarily delay instabilities that still emerge in later stages of training.

To approach this problem, we observe that previous methods have not taken advantage of the fact that $f$ can be viewed as a fixed point operation. Thus, $f$ can be trained with **implicit differentiation** applied at the fixed point, without backpropagating gradients through the unrolled iterations. We then leverage tools developed for implicit differentiation in neural models for improving the training of these methods. *Our primary contribution is to propose implicit differentiation for training the iterative amortized inference procedures of symmetric generative models, such as those used for learning object representations.* Namely:

1. We show that prior iterative amortized inference methods, including those on object-centric learning, can be cast as fixed point procedures that can be trained with implicit differentiation. We call the resulting class of methods **implicit iterative inference** algorithms.
2. We show on the latest state-of-the-art of these methods, SLATE (Singh et al., 2021), that using the first-order Neumann approximation of the implicit gradient for the slot attention module (Locatello et al., 2020) yields substantial improvement in optimization.

We show across three datasets that, compared to SLATE, our method for training achieves much lower validation loss in training, as well as lower Fréchet inception distance (FID) (Heusel et al., 2017) and mean squared error (MSE) in image reconstruction. Our method also removes the need for gradient clipping, learning rate decay, learning rate warmup, or tuning the number of iterations, while achieving lower space and time complexity in the backward pass, all with one additional line of code.

## 2 Related Work

The equivalence of maximizing the evidence lower bound (ELBO) (Neal & Hinton, 1998; Bottou & Bengio, 1995) and minimizing variational free energy (Dayan et al., 1995; Friston, 2010) signifies that any method for minimizing an energy function (LeCun et al., 2006) can be cast as a fixed point procedure. Thus, iterative inference procedures can naturally be described as specifying the attractor dynamics of a dynamical system whose stable states correspond to the posterior or model parameters produced as a result of the inference, making these procedures especially useful for dynamically inferring representations of objects (Greff et al., 2020). Thus far, however, the methods that instantiate these iterative inference updates with neural networks (see §3) are difficult to train because they all differentiate through the unrolled dynamics of the fixed point procedure. Implicit differentiation can offer a solution to this difficulty. In deep learning (Duvenaud et al., 2020), it has been applied to embedded optimization layers (Amos & Kolter, 2017; Agrawal et al., 2019), neural ordinary differential equations (Chen et al., 2018), meta-learning (Rajeswaran et al., 2019), implicit neural representations (Huang et al., 2021), declarative layers (Gould et al., 2019), and entire networks (Bai et al., 2019). We focus on the novel application of implicit differentiation to iterative inference algorithms that approximate the optimization of the ELBO.

## 3 Background

Our work builds on prior works on iterative amortized inference and implicit differentiation with deep neural networks. This section reviews recent advances in these two areas and introduces the formalism we use in the rest of the paper.

### 3.1 ITERATIVE AMORTIZED INFERENCE

Consider a generative model with observed variables $X$, local (per-observation) latent variables $Z$, and global (across-observations) parameters $\theta$, defining the joint distribution for a particular sample $(z, x)$ as $p(z, x; \theta) = p(z; \theta^z)p(x \mid z; \theta^{x \mid z})$. Given a datapoint $x$, the goals of statistical inference often involve estimating the parameters $\theta$ or inferring the posterior $p(z \mid x)$. Both can be achieved via variational techniques (Neal & Hinton, 1998; Dayan et al., 1995) that frame inference as a maximization of the evidence lower bound (ELBO) $\mathcal{L}$ with respect to $\theta$ and an approximate posterior $q(z \mid \cdot)$:

$$\mathcal{L}(q, \theta, x) := \mathop{\mathbb{E}}_{z \sim q(z \mid \cdot)} \left[\log p(x, z; \theta) - \log q(z \mid \cdot)\right]. \tag{1}$$

**Classical and amortized inference** Classical approaches for maximizing $\mathcal{L}$ include variants of the expectation maximization (EM) algorithm (Dempster et al., 1977), which alternates between optimizing $\max_q \mathcal{L}(q, \theta, x)$ and $\max_\theta \mathcal{L}(q, \theta, x)$, using incremental (e.g., gradient descent) or analytic approaches. Given a dataset $\{x^n\}_{n=1}^N$, and assuming that $q$ can be parameterized by variational parameters $\phi$, classical iterative methods employ a fixed learning rule (Hoffman et al., 2013), for improving $\phi$ or $\theta$, e.g.

$$\phi_{t+1}^n \leftarrow \phi_t^n + \alpha \nabla_{\phi^n} \mathcal{L}(\phi_t^n, x^n), \quad \forall n \tag{2}$$

$$\theta_{t+1} \leftarrow \theta_t + \beta \sum\nolimits_n \nabla_\theta \mathcal{L}(\theta_t, x^n) \tag{3}$$

which is costly to scale to high-dimensional datasets. Thus, techniques related to the variational autoencoder (Kingma & Welling, 2013; Rezende et al., 2014, VAE) amortize (Gershman & Goodman, 2014) the optimization of $\phi^n$ for each $x^n$ via an encoder network that directly maps $x$ to $\phi$. However, estimating $\phi$ without feedback from an iterative procedure results in decreased modeling performance (Krishnan et al., 2018; Cremer et al., 2018) and cannot break symmetry among exchangeable unobserved variables.

**Iterative amortized inference** Several works have proposed to combine the paradigms of iterative optimization and neural networks by replacing the fixed update rule with a update network $f$ that is trained to optimize the unrolled iterative procedure (Kirsch & Schmidhuber, 2020; Andrychowicz et al., 2016) for improving ELBO. All proposals so far have trained $f$ by backpropagating gradients through the unrolled updates. These can be categorized as performing posterior inference or parameter estimation via a meta-learning algorithm (Thrun & Pratt, 2012; Schmidhuber, 1987), with the former conducted across a single dataset (like Andrychowicz et al. (2016)) and the latter conducted across a dataset of mini-datasets (like Finn et al. (2017)).

For methods that **meta-learn posterior inference** (Marino et al., 2018b;a; 2020; Greff et al., 2019; Veerapaneni et al., 2020), instead of an encoder network that directly maps $x^n$ to $\phi^n$, an update network $f$ improves a (initially random) previous estimate $\phi_t^n$ as $\phi_{t+1}^n \leftarrow f(\phi_t^n, \nabla_{\phi_t^n} \mathcal{L}_t)$ for each datapoint $x^n$. While $\phi^n$ is updated per-datapoint, the model parameters $\theta$ and weights of $f$ are updated across datapoints.

In contrast, methods that **meta-learn parameter estimation** (Greff et al., 2017; Van Steenkiste et al., 2018; Zoran et al., 2021; Locatello et al., 2020; Singh et al., 2021) treats each datapoint $x^n$ as *itself* a mini-dataset of $M$ measurements $x^{n,m}$ (e.g. $x^n$ is an image and $x^{n,m}$ is a pixel or feature of $x^n$). Each datapoint $x^n$ is generated from per-datapoint model parameters $\theta^n$ with per-measurement latents $z^{n,m}$, thus defining a per-datapoint ELBO $\mathcal{L}^n$. The role of the update network $f$ in this setting is to improve the (initially random) model parameters $\theta^n$ as $\theta_{t+1}^n \leftarrow f(\theta_t^n, \nabla_{\theta_t^n} \mathcal{L}_t^n)$, which generally also involves improving the per-measurement variational parameters $\phi^{n,m}$.

**Object-centric learning** A concrete application of iterative amortized inference has been in so-called **object-centric learning**, a research area that seeks to decompose observations $x$ into a set of independent representations of entities without supervision on how to decompose. Each datapoint $x^n$ (e.g. image or sensorimotor sequence) is a set of independent sensor measurements $x^{n,m}$ (e.g. pixels) which are generally posited as having been generated from a mixture model whose components represent the entities. Under a clustering lens, the problem reduces to finding the $K$ groups of cluster parameters $\boldsymbol{\theta}^n := \{\theta^{n,k}\}_{k=1}^K$ and cluster assignments $\boldsymbol{\phi}^{n,m} := \{\phi^{n,m,k}\}_{k=1}^K$ that were

responsible for the measurements $x^{n,m}$ of the datapoint $x^n$. Modeling entities as cluster components encodes the assumption that entities are a priori *independent* and *symmetric*, thereby requiring a symmetry-breaking mechanism during inference.

To solve this problem, an update network $f$ breaks symmetry among components by alternately updating $\boldsymbol{\theta}^n$ and $\boldsymbol{\phi}^{n,m}$ starting from independent randomly initialized $\theta^{n,k}$'s. The state-of-the-art slot attention module (Locatello et al., 2020), e.g., computes $\boldsymbol{\theta}_{t+1}^n \leftarrow f\left(\boldsymbol{\theta}_t^n, x^n\right)$, where $\boldsymbol{\phi}^{n,m}$ is updated as an intermediate step inside $f$. The $\boldsymbol{\theta^n}$, called *slots*, serve as input to a downstream objective, e.g. image reconstruction, whose gradients are backpropagated through the unrolling of $f$. Earlier works applied this meta-learned parameter estimation approach to binary images (Greff et al., 2017) and videos (Van Steenkiste et al., 2018). Methods via meta-learned posterior inference have also been developed for images (Greff et al., 2017) and model-based planning (Veerapaneni et al., 2020) using mixture-density networks (Bishop, 1994) for the generative model.

### 3.2 Implicit differentiation

Implicit differentiation is a technique for computing the gradients of a function defined in terms of satisfying a joint condition of the input and output. For example, a fixed point operation $f$ is defined to satisfy "find $\lambda$ such that $\lambda = f(x, \lambda)$," rather than through an explicit parameterization of $f$. This fixed point $\lambda_*$ can be computed by simply repeatedly applying $f$ or by using a black-box root-finding solver. Letting $f_{\mathbf{w}}$ be parameterized by weights $\mathbf{w}$, with input $x$ and fixed point $\lambda_*$, the implicit function theorem (Cauchy, 1831) enables us to directly compute the gradient of the loss $\ell$ with respect to $\mathbf{w}$, using only the output $\lambda_*$:

$$\frac{\partial \ell}{\partial \mathbf{w}} = \frac{\partial \ell}{\partial \lambda_*} \left(I - J_{f_{\mathbf{w}}}\left(\lambda_*\right)\right)^{-1} \frac{\partial f_{\mathbf{w}}\left(\lambda_*, x\right)}{\partial \mathbf{w}}, \tag{4}$$

where $J_{f_{\mathbf{w}}}\left(\lambda_*\right)$ is the Jacobian matrix of $f_{\mathbf{w}}$ evaluated at $\lambda_*$. Compared to backpropagating through the unrolled iteration of $f$, which is just one of many choices of the solver, implicit differentiation via Eq. 4 removes the memory cost of storing any intermediate results from the unrolled iteration.

Much effort has been put into approximating the inverse-Jacobian term $\left(I - J_{f_{\mathbf{w}}}\left(\lambda_*\right)\right)^{-1}$ which has $\mathcal{O}(n^3)$ complexity to compute. Geng et al. (2021); Fung et al. (2021); Huang et al. (2021); Shaban et al. (2019) propose instead to approximate $\left(I - J_{f_{\mathbf{w}}}\left(\lambda_*\right)\right)^{-1}$ with its Neumann series expansion:

$$\left(I - J_{f_{\mathbf{w}}}\left(\lambda_*\right)\right)^{-1} = \lim_{T \to \infty} \sum_{i=0}^{T} J_{f_{\mathbf{w}}}\left(\lambda_*\right)^i. \tag{5}$$

The first-order approximation ($T = 1$) amounts to applying $f$ once to the fixed point $\lambda_*$ and differentiating through the resulting computation graph. This is not only cheap to compute and easy to implement, but has also been shown empirically (Geng et al., 2021) to have a regularizing effect on the spectral norm of $J_{f_{\mathbf{w}}}$ without sacrificing performance.

## 4 Inference As A Fixed Point Iteration

To explain why it makes sense to train iterative amortized inference algorithms with implicit differentiation, we explicitly unify iterative amortized inference methods (§ 3.1) as solving a particular nested optimization problem whose inner optimization is that of maximizing the ELBO, thereby allowing them to be understood as fixed point procedures. We describe the abstracted bi-level optimization problem (one-level of nesting), then show that meta-learned posterior inference and meta-learned parameter estimation instantiate this problem with one and two levels of nesting respectively.

### 4.1 The nested optimization problem

Consider the following bi-level optimization problem over a generic dataset $\{x^n\}_{n=1}^N$ with datapoints $x^n$. Define the parameters $\lambda^n$ as optimized *per*-datapoint, and the parameters $\mathbf{w}$ as optimized *across* datapoints. With the ELBO $\mathcal{L}$ as the inner objective and a task objective $\mathcal{J}$ as the outer objective, we

express the bi-level optimization problem as

$$\begin{aligned} \min_{\mathbf{w}} \quad & \sum_n \mathcal{J}\left(x^n, \mathbf{w}, \lambda_*^n\right) \\ \text{s.t.} \quad & \lambda_*^n = \arg\max_{\lambda^n} \mathcal{L}\left(\tilde{x}^n, \lambda^n\right). \end{aligned} \tag{6}$$

When the inner optimization is conducted via a fixed update rule, the solution of the inner problem can be embedded as a differentiable optimization layer (Amos & Kolter, 2017) within a neural network with weights $\mathbf{w}$. Here, we partition $\mathbf{w}$ as $\mathbf{w} = [\mathbf{w}_e, \mathbf{w}_d]$, where $\mathbf{w}_e$ are weights of an encoder that processes $x^n$ into $\tilde{x}^n$, and $\mathbf{w}_d$ are weights of a decoder that computes the outer objective $\mathcal{J}$ with the fixed point $\lambda_*^n$ as an input. Special cases include the case where $\mathbf{w}_e$ is the identity (i.e., no pre-processing of $x^n$) and the case where $\mathbf{w}_d$ is the identity (i.e. no post-processing of $\lambda_*^n$).

Using a trainable network $f_{\mathbf{w}}$ as the update rule instead, e.g. $\lambda_{t+1}^n \leftarrow f_{\mathbf{w}}(\lambda_t^n, x^n)$, implicitly parameterizes a constraint set $\mathcal{C}_{\mathbf{w}}\left(x^n\right)$. The weights $\mathbf{w}$ from Eq. 6 now include the weights $\mathbf{w}_u$ of the learnable update rule $f_{\mathbf{w}}$, yielding:

$$\begin{aligned} \min_{\mathbf{w}} \quad & \sum_n \mathcal{J}\left(x^n, \mathbf{w}, \lambda_*^n\right) \\ \text{s.t.} \quad & \lambda_*^n = \arg\max_{\lambda^n \in \mathcal{C}_{\mathbf{w}}(x^n)} \mathcal{L}\left(\tilde{x}^n, \lambda^n\right). \end{aligned} \tag{7}$$

The constraint set $\mathcal{C}_{\mathbf{w}}\left(x^n\right)$ implicitly depends on $\mathbf{w}_e$, which pre-processes $x^n$, and $\mathbf{w}_u$, which updates $\lambda^n$. Any update rule that monotonically improves upon $\mathcal{L}$ is thus a fixed point operation whose fixed point locally maximizes $\mathcal{L}$ (Neal & Hinton, 1998; Wu, 1983). It is in this sense that we can understand $f_{\mathbf{w}}$ as trained to perform a fixed point operation.

### 4.2 Posterior inference and parameter estimation

Now we show that iterative amortized inference for posterior inference and parameter estimation implement fixed point procedures that solve the aforementioned nested optimization problem. This is to our knowledge the first unification of both approaches under the same problem statement.

**Meta-learned posterior inference** Methods for meta-learned posterior inference (§3.1) train a VAE decoder as the generative model with parameters $\theta$ and an update network $f_{\mathbf{w}}$ that updates $\phi_{t+1}^n \leftarrow f_{\mathbf{w}}(\phi_t^n, \nabla_{\phi_t^n}\mathcal{L}_t)$ for each datapoint $x^n$. We recover the problem formulation in Eq. 7 by substituting the negative ELBO for the outer objective $\mathcal{J}$, the per-datapoint variational parameters $\phi^n$ for $\lambda^n$, and the model parameters $\theta$ for the subset of $\mathbf{w}_d$ that compute the $p(x \mid z)$ term of the negative ELBO. Then the update network implements the fixed point operation $\phi_{t+1}^n \leftarrow f_{\mathbf{w}}(\phi_t^n, x^n)$ that computes $\nabla_{\phi_t^n}\mathcal{L}_t$ from $\phi_t^n$ and $x^n$ as an initial pre-processing step.

**Meta-learned parameter estimation** Methods for meta-learned parameter estimation (§3.1) treat each datapoint $x^n$ as a mini-dataset of measurements $x^{n,m}$. We recover the problem formulation in Eq. 7 by substituting a per-datapoint ELBO $\mathcal{L}^n$ for the inner objective and let $\lambda^n := \left(\theta^n, \{\phi^{n,m}\}_{m=1}^M\right)$, meaning that the inner optimization jointly optimizes the per-datapoint model parameters $\theta^n$ and all per-measurement variational parameters $\phi^{n,m}$. Existing approaches implement this by using $f_{\mathbf{w}}$ to compute an EM (Greff et al., 2017; Van Steenkiste et al., 2018) or modified soft K-means (Locatello et al., 2020; Zoran et al., 2021; Singh et al., 2021) step. Since the variational inference problem (Eq. 1) is itself a bi-level optimization over $\theta$ and $\phi$, meta-learned parameter estimation is actually a tri-level optimization, optimizing $\mathbf{w}$ across datapoints $x^n$ at the outer level, $\theta$ across measurements $x^{n,m}$ but per-datapoint at the middle level, and $\phi$ per-measurement at the inner level. The inner two objectives are the ELBO $\mathcal{L}^n$ defined for each datapoint and the outer objective $\mathcal{J}$ is a task objective specified for the dataset, such as image reconstruction or attribute classification (Locatello et al., 2020).

### 4.3 Entities as independently initialized fixed points

Having established the above formalism, the object-centric learning problem as studied so far represents a subset of instances of the nested optimization problem described in §4.1, where the

inner optimization is of a *set* of independently initialized parameters $\boldsymbol{\lambda}^n := \{\lambda^{n,k}\}_{k=1}^K$ that are symmetrically updated by $f_{\mathbf{w}}$. What this paper contributes is the explicit formulation of these entity representations $\boldsymbol{\lambda}^n$ as fast weights that converge towards a set of fixed points during execution. This gives us a unifying tangible problem statement for focusing object-centric learning research and enables us to improve the training of such methods with implicit differentiation, as we discuss in the next section.

## 5 Implicit Iterative Inference

Having shown how iterative amortized inference methods can be expressed as implementing learnable fixed point operations of the form $\lambda^n \leftarrow f_{\mathbf{w}}(\lambda^n, x^n)$, it is simple to just substitute $\mathcal{J}$ for $\ell$ in Eq. 4 to get the implicit gradient of $\mathcal{J}$ with respect to the weights $\mathbf{w}$. Because implicit differentiation decouples the forward and backward passes, any black box solver for computing the fixed point and black box gradient estimator for computing the implicit gradient can be used. We refer to this family of algorithms for solving Eq. 7 with implicit differentiation as **implicit iterative inference** algorithms.

```
def step(slots, k, v):
    # compute assignments given slots
    q = project_q(norm_slots(slots))
    k = k * (slot_size ** (-0.5))
    attn = F.softmax(torch.einsum('bkd,bqd->bkq', k, q), dim=-1)
    attn = attn / torch.sum(attn + epsilon, dim=-2, keepdim=True)
    # update slots given assignments
    updates = torch.einsum('bvq,bvd->bqd', attn, v)
    slots = gru(updates, slots)
    slots = slots + mlp(norm_mlp(slots))
    return slots

def iterate(f, x, num_iters):
    for _ in range(num_iters):
        x = f(x)
    return x

def forward(inputs, slots):
    inputs = norm_inputs(inputs)
    k, v  = project_k(inputs), project_v(inputs)
    slots = iterate(lambda z: step(z, k, v), slots, num_iterations)
    slots = step(slots.detach(), k, v)
    return slots
```

Figure 2: **Code.** The first order Neumann approximation to the implicit gradient adds only one additional line of Pytorch code (Paszke et al., 2019) to the original forward function of slot attention, but yields substantial improvement of optimization. `attn` and `slots` correspond to $\boldsymbol{\phi}$ and $\boldsymbol{\theta}$ in the text respectively.

**Implicit object-centric learning** To illustrate an example of such implicit inference algorithms, we propose **implicit slot attention**: a method for training the state-of-the-art slot attention module (Locatello et al., 2020), which performs meta-learned parameter estimation, with the simplest and most effective method that we have empirically found for approximating the implicit gradient, which is its first-order Neumann approximation (Eq. 5). It can be implemented by simply differentiating the computation graph of applying the slot attention update *once* to the fixed point $\boldsymbol{\theta}_*^n$, where $\boldsymbol{\theta}_*^n$ is computed by simply iterating the slot attention module forward as usual, but *without* the gradient tape. The time and space complexity of backpropagation for our method compared to vanilla slot attention as a function of the number of slot attention iterations $n$, is shown below:

| | vanilla slot attention | ours |
|---|---|---|
| time (forward) | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| space (forward) | $\mathcal{O}(n)$ | $\mathcal{O}(n)$ |
| time (backward) | $\mathcal{O}(n)$ | $\mathcal{O}(1)$ |
| space (backward) | $\mathcal{O}(n)$ | $\mathcal{O}(1)$. |

Our method is not only more efficient but also requires only one additional line of code (Fig. 2).

## 6 Experiments

The main hypothesis behind this paper is that implicit differentiation can improve the training of iterative amortized inference methods for object-centric learning. We test this hypothesis by replacing the backward pass of the slot attention module in SLATE (Singh et al., 2021) with the first-order Neumann approximation of the implicit gradient, and measuring optimization performance.

For the task of image reconstruction, SLATE uses a discrete VAE (Ramesh et al., 2021) to compress an input image into a grid of discrete tokens. These tokens index into a codebook of latent code-vectors, which, after applying a learned position encoding, serve as the input to the slot attention module. An Image GPT decoder (Chen et al., 2020) is trained with a cross-entropy loss to autoregressively reconstruct the latent code-vectors, using the outputted slots from slot attention as queries and the latent code-vectors as keys/values. Gradients are blocked from flowing in and out of the discrete VAE

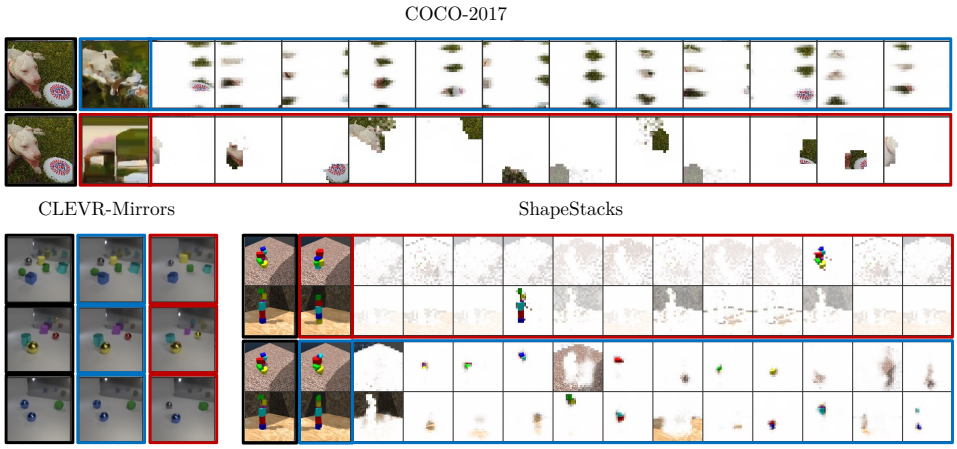



Figure 3: **Qualitative results.** Across three datasets, optimizing SLATE with implicit differentiation leads to improved image reconstructions through the slot bottleneck. Black borders indicate the ground truth image, blue border indicate our method, and red borders indicate vanilla SLATE. Other panels visualize attention masks.

to the rest of the network (i.e. the slot attention module and the Image GPT decoder), but the entire system is trained simultaneously.

We consider three datasets: CLEVR-Mirror (Singh et al., 2021), Shapestacks (Groth et al., 2018), and COCO-2017 (Lin et al., 2014). We obtained CLEVR-Mirror directly from the SLATE authors and used a 70-15-15 split for training, validation, and testing. We pooled all the data variants of Shapestacks together as Singh et al. (2021) did and used the original train-validation-test splits. The COCO-2017 dataset was downloaded from FiftyOne and used the original train-validation-test splits.

### 6.1 Does implicit differentiation stabilize the training of slot attention?

Using the two primary metrics used in Singh et al. (2021), images generated by SLATE trained with implicit differentiation achieve both lower pixel-wise mean-squared error and FID score (Heusel et al., 2017). The FID score was computed with the PyTorch-Ignite (Fomin et al., 2020) library using the inception network from the PyTorch port of the FID official implementation. All methods were trained for 250k gradient steps. Table 1 compares the FID and MSE scores of the images that result from compressing the SLATE encoder's set of discrete tokens through the slot attention bottleneck, using Image-GPT to autoregressively re-generate these image tokens one by one, and using the discrete VAE decoder to render the generated image tokens. Implicit differentiation significantly improves the quantitative image reconstruction metrics of SLATE across the test sets of CLEVR-Mirrors, Shapestacks, and COCO. In the case of MSE for CLEVR, this is almost a 7x improvement.

Table 1: Quantitative metrics for image reconstruction through the slot bottleneck.

| Data | Ours | Vanilla |
|---|---|---|
| CLEVR (FID) | **22.19** | 25.89 |
| CLEVR (MSE) | **10.66** | 67.04 |
| COCO (FID) | **127.79** | 147.48 |
| COCO (MSE) | **1659.15** | 1821.75 |
| ShapeStacks (FID) | **34.2** | 34.76 |
| ShapeStacks (MSE) | **108.67** | 312.14 |

The higher quantitiatve metrics also translate into better quality reconstructions on the test set, as shown in Figure 3. For CLEVR-Mirrors, vanilla SLATE sometimes drops or changes the appearance of objects, even simple scenes with three objects. In contrast, the reconstructions produced from training with implicit differentiation match the ground truth very closely. For Shapestacks, our method consistently segements the scene into constituent objects. This is sometimes the case with vanilla SLATE on the training and validation set as well, but we observed for both of the seeds we ran that vanilla SLATE produced degenerated attention maps where one slot captures the entire foreground, and the background is divided among the other slots. The visual complexity of the COCO dataset is much higher than either CLEVR-Mirrors and Shapestacks, and the reconstructions on the COCO dataset are quite poor, for both SLATE's discrete VAE and consequently for the

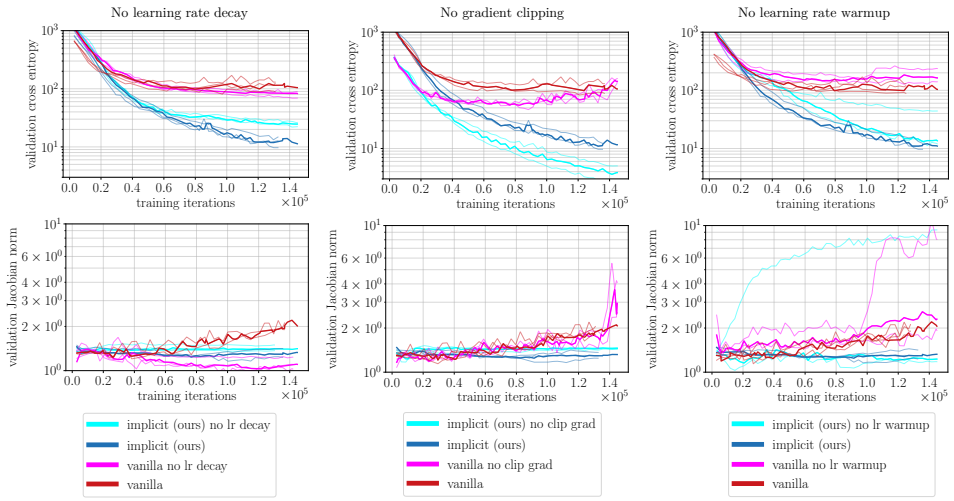



Figure 4: **Implicit differentiation removes the need for many optimization tricks.** We ablate three heuristically-motivated optimization tricks from both vanilla SLATE and our method. Whereas removing gradient clipping and learning rate warmup causes vanilla SLATE's training to become unstable, as indicated by the growth of the Jacobian norm of the slot attention cell, our method trains significantly more stably and can take advantage of the larger gradient steps.

reconstruction through the slot bottleneck. This may be expected because we did not attempt to tune SLATE's hyperparameters to COCO, but it does highlight the gap that still exists between using the state-of-the-art in object-centric learning out-of-the-box and what the community may want these methods to do. The attention masks for both the vanilla SLATE and our method furthermore do not appear to correspond consistently to coherent objects in COCO but rather patches on the image that do not immediately seem to match with our human intuition of what constitutes a visual entity.

### 6.2 Can we simplify the need for optimization tricks?

To further understand the benefits of implicit differentiation, we then ask whether it stabilizes the training of slot attention without the need for optimization tricks like learning rate decay, gradient clipping, and learning warmup. Fig. 4 shows that these tricks generally help regularize spectral norm of the Jacobian of vanilla slot attention but are not required by our method. Decaying the learning rate regularizes the Jacobian norm from exploding, but it also hurts optimization performance for both our method and vanilla SLATE, as expected. When we remove gradient clipping, the Jacobian norm of vanilla SLATE explodes, whereas it stays stable for our method. Lastly, removing learning rate warmup also consistently makes vanilla SLATE's training unstable, whereas it only affects the stability of our method for one out of three seeds.

## 7 Discussion and Limitations

Our results show clear signal that implicit differentiation can offer a significant optimization improvement over backpropagating through the unrolled iteration of slot attention, and potentially any iterative inference algorithm, with lower space and time complexity and only one additional line of code. Despite our work pushing the optimization performance for a state-of-the-art model in object-centric learning, the discrepancy between the quantitative improvement in optimization and evaluation metrics on the one hand and the less intuitive qualitative attention masks on real world observations like COCO (Fig. 3) on the other hand still suggests a gap between what we optimize these methods to do and what we actually want them to do. This paper proposes a novel conceptualization of object representations as fast weights that converge towards a set of fixed points during execution. Because it is so simple to apply implicit differentiation to any fixed point algorithm, we hope this work inspires future work to leverage tools developed for implicit differentiation for improving object-centric learning and methods for learning latent structure more broadly.

## REFERENCES


Akshay Agrawal, Brandon Amos, Shane Barratt, Stephen Boyd, Steven Diamond, and Zico Kolter. Differentiable convex optimization layers. *arXiv preprint arXiv:1910.12430*, 2019.

Brandon Amos and J Zico Kolter. Optnet: Differentiable optimization as a layer in neural networks. In *International Conference on Machine Learning*, pp. 136–145. PMLR, 2017.

Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. In *Advances in neural information processing systems*, pp. 3981–3989, 2016.

Shaojie Bai, J Zico Kolter, and Vladlen Koltun. Deep equilibrium models. *arXiv preprint arXiv:1909.01377*, 2019.

Shaojie Bai, Vladlen Koltun, and J Zico Kolter. Stabilizing equilibrium models by jacobian regularization. *arXiv preprint arXiv:2106.14342*, 2021.

Christopher M Bishop. Mixture density networks. 1994.

Leon Bottou and Yoshua Bengio. Convergence properties of the k-means algorithms. In *Advances in neural information processing systems*, pp. 585–592, 1995.

Augustin-Louis Cauchy. Résumé d'un mémoir sur la mécanique céleste et sur un nouveau calcul appelé calcul des limites. *Oeuvres Complétes d'Augustun Cauchy*, 12(48-112):3, 1831.

Mark Chen, Alec Radford, Rewon Child, Jeffrey Wu, Heewoo Jun, David Luan, and Ilya Sutskever. Generative pretraining from pixels. In *International Conference on Machine Learning*, pp. 1691–1703. PMLR, 2020.

Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. *arXiv preprint arXiv:1806.07366*, 2018.

Chris Cremer, Xuechen Li, and David Duvenaud. Inference suboptimality in variational autoencoders. In *International Conference on Machine Learning*, pp. 1078–1086. PMLR, 2018.

Peter Dayan, Geoffrey E Hinton, Radford M Neal, and Richard S Zemel. The helmholtz machine. *Neural computation*, 7(5):889–904, 1995.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1): 1–22, 1977.

David Duvenaud, J. Zico Kolter, and Matthew Johnson. Deep implicit layers tutorial - neural ODEs, deep equilibirum models, and beyond. *Neural Information Processing Systems Tutorial*, 2020.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International Conference on Machine Learning*, pp. 1126–1135. PMLR, 2017.

V. Fomin, J. Anmol, S. Desroziers, J. Kriss, and A. Tejani. High-level library to help with training neural networks in pytorch. `https://github.com/pytorch/ignite`, 2020.

Karl Friston. The free-energy principle: a unified brain theory? *Nature reviews neuroscience*, 11(2): 127–138, 2010.

Samy Wu Fung, Howard Heaton, Qiuwei Li, Daniel McKenzie, Stanley Osher, and Wotao Yin. Fixed point networks: Implicit depth models with jacobian-free backprop. *arXiv preprint arXiv:2103.12803*, 2021.

Zhengyang Geng, Xin-Yu Zhang, Shaojie Bai, Yisen Wang, and Zhouchen Lin. On training implicit models. *Advances in Neural Information Processing Systems*, 34, 2021.

Samuel Gershman and Noah Goodman. Amortized inference in probabilistic reasoning. In *Proceedings of the annual meeting of the cognitive science society*, volume 36, 2014.

Stephen Gould, Richard Hartley, and Dylan Campbell. Deep declarative networks: A new hope. *arXiv preprint arXiv:1909.04866*, 2019.

Klaus Greff, Sjoerd Van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. *arXiv preprint arXiv:1708.03498*, 2017.

Klaus Greff, Raphaël Lopez Kaufman, Rishabh Kabra, Nick Watters, Christopher Burgess, Daniel Zoran, Loic Matthey, Matthew Botvinick, and Alexander Lerchner. Multi-object representation learning with iterative variational inference. In *International Conference on Machine Learning*, pp. 2424–2433. PMLR, 2019.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. On the binding problem in artificial neural networks. *arXiv preprint arXiv:2012.05208*, 2020.

Oliver Groth, Fabian B Fuchs, Ingmar Posner, and Andrea Vedaldi. Shapestacks: Learning vision-based physical intuition for generalised object stacking. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 702–717, 2018.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

Matthew D Hoffman, David M Blei, Chong Wang, and John Paisley. Stochastic variational inference. *Journal of Machine Learning Research*, 14(5), 2013.

Zhichun Huang, Shaojie Bai, and J Zico Kolter. Implicit$^2$: Implicit layers for implicit representations. *Advances in Neural Information Processing Systems*, 34, 2021.

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

Thomas Kipf, Gamaleldin F Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional object-centric learning from video. *arXiv preprint arXiv:2111.12594*, 2021.

Louis Kirsch and Jürgen Schmidhuber. Meta learning backpropagation and improving it. *arXiv preprint arXiv:2012.14905*, 2020.

Rahul Krishnan, Dawen Liang, and Matthew Hoffman. On the challenges of learning with inference networks on sparse, high-dimensional data. In *International Conference on Artificial Intelligence and Statistics*, pp. 143–151. PMLR, 2018.

Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, and F Huang. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pp. 740–755. Springer, 2014.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *arXiv preprint arXiv:2006.15055*, 2020.

Joe Marino, Yisong Yue, and Stephan Mandt. Iterative amortized inference. In *International Conference on Machine Learning*, pp. 3403–3412. PMLR, 2018a.

Joseph Marino, Milan Cvitkovic, and Yisong Yue. A general method for amortizing variational filtering. *arXiv preprint arXiv:1811.05090*, 2018b.

Joseph Marino, Alexandre Piché, Alessandro Davide Ialongo, and Yisong Yue. Iterative amortized policy optimization. *arXiv preprint arXiv:2010.10670*, 2020.

Radford M Neal and Geoffrey E Hinton. A view of the em algorithm that justifies incremental, sparse, and other variants. In *Learning in graphical models*, pp. 355–368. Springer, 1998.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32: 8026–8037, 2019.

Aravind Rajeswaran, Chelsea Finn, Sham Kakade, and Sergey Levine. Meta-learning with implicit gradients. 2019.

Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092*, 2021.

Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *International conference on machine learning*, pp. 1278–1286. PMLR, 2014.

Jürgen Schmidhuber. *Evolutionary principles in self-referential learning, or on learning how to learn: the meta-meta-... hook*. PhD thesis, Technische Universität München, 1987.

Amirreza Shaban, Ching-An Cheng, Nathan Hatch, and Byron Boots. Truncated back-propagation for bilevel optimization. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pp. 1723–1732. PMLR, 2019.

Gautam Singh, Fei Deng, and Sungjin Ahn. Illiterate dall·e learns to compose. *arXiv preprint arXiv:2110.11405*, 2021.

Sebastian Thrun and Lorien Pratt. *Learning to learn*. Springer Science & Business Media, 2012.

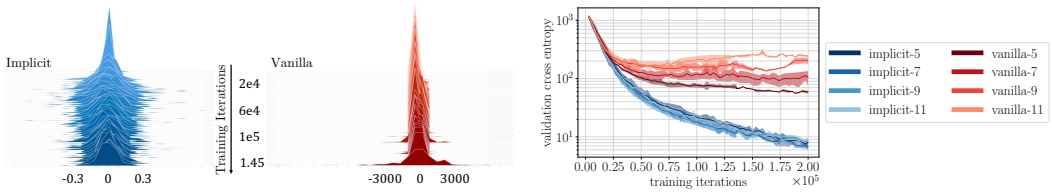
Sjoerd Van Steenkiste, Michael Chang, Klaus Greff, and Jürgen Schmidhuber. Relational neural expectation maximization: Unsupervised discovery of objects and their interactions. *arXiv preprint arXiv:1802.10353*, 2018.

Rishi Veerapaneni, John D Co-Reyes, Michael Chang, Michael Janner, Chelsea Finn, Jiajun Wu, Joshua Tenenbaum, and Sergey Levine. Entity abstraction in visual model-based reinforcement learning. In *Conference on Robot Learning*, pp. 1439–1456. PMLR, 2020.

C. F. Jeff Wu. On the Convergence Properties of the EM Algorithm. *The Annals of Statistics*, 11(1): 95 – 103, 1983. doi: 10.1214/aos/1176346060. URL `https://doi.org/10.1214/aos/1176346060`.

Daniel Zoran, Rishabh Kabra, Alexander Lerchner, and Danilo J Rezende. Parts: Unsupervised segmentation with slots, attention and independence maximization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 10439–10447, 2021.

# A APPENDIX




(a) No need for gradient clipping.

(b) No sensitivity to number of iterations.

Figure 5: **Stability.** (5a) Without gradient clipping, our implicit differentiation technique keeps gradients small while backpropagating through the unrolled iterations causes gradients to explode. (5b) Training with implicit differentiation also is not sensitive to the number of iterations with which to iterate the slot attention module.

When we remove gradient clipping, the gradients of vanilla SLATE explodes, whereas they stays stable for our method. Fig. 5b shows that our method is not sensitive to the number of iterations with which to iterate the slot attention cell, whereas vanilla slot attention is, with more iterations being harder to train.