STOCHASTIC DEEP NETWORKS WITH LINEAR COMPETING UNITS FOR MODEL-AGNOSTIC META-LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

This work addresses meta-learning (ML) by considering deep networks with stochastic local winner-takes-all (LWTA) activations. This type of network units result in sparse representations from each model layer, as the units are organized into blocks where only one unit generates a non-zero output. The main operating principle of the introduced units lies on stochastic arguments, as the network performs posterior sampling over competing units to select the winner. Therefore, the proposed networks are explicitly designed to extract input data representations of sparse stochastic nature, as opposed to the currently standard deterministic representation paradigm. We posit that these modeling arguments, inspired from Bayesian statistics, allow for more robust modeling when uncertainty is high due to the limited availability of task-related training data; this is exactly the case with ML, which is the focus of this work. At training time, we rely on the reparameterization trick for Discrete distributions to perform reliable training via Monte-Carlo sampling. At inference time, we rely on Bayesian Model Averaging, which effectively averages over a number of sampled representations. As we experimentally show, our approach produces state-of-the-art predictive accuracy on standard few-shot image classification benchmarks; this is achieved without compromising computational efficiency.

1 INTRODUCTION

When we train machine learning models on problems with limited amounts of training data, we cannot usually get good predictive performance (Lai, 2019; Sculley et al., 2015). This comes in contrast to the human ability to quickly derive information from a range of different tasks (Kühl et al., 2020; Castro et al., 2008), and then adapt to a new task with limited available new examples. In essence, this capability of the mind of learning how to learn (Black et al., 2006) has inspired researchers to investigate the concept of Meta-Learning (ML) (Lake et al., 2017; Vilalta & Drissi, 2002; Wang et al., 2016; Zoph & Le, 2017; Flennerhag et al., 2020); ML can be defined as the challenge of intelligent adaptation to a large variety of unseen situations (Lake et al., 2015; Hutsebaut-Buysse et al., 2019).

In the context of deep learning, various recent works have attempted to improve the poor performance of standard deep neural networks in few-shot learning environments where limited data is available (Finn et al., 2019; Baik et al., 2020; Andrychowicz et al., 2016). For instance, Finn et al. (2017) presented the *Model-Agnostic Meta-Learning* (MAML) algorithm that enables tuning the parameters of a trained model to quickly learn a new task with only a few gradient updates. To get away with the entailed second-order computations, which are expensive, several researchers proposed appropriate first-order approximations for MAML; such works are the *First-Order MAML* (FOMAML) of Finn et al. (2017), and the *Reptile* algorithm of Nichol et al. (2018).

Despite these advances, the existing works building upon MAML and its approximate variants do not take into consideration an important aspect that underlies ML tasks: The limited availability of training data pertaining to the target task implies a high level of (epistemic) uncertainty in the data representations obtained from the employed deep networks. To overcome this challenge, we posit that we need to devise foundational deep network arguments that intrinsically give rise to trainable stochastic representations. Specifically, we draw inspiration from recent advances at the frontier of

biologically plausible deep network research. In this context, there is an increasing body of evidence from Neuroscience that neurons with similar functions in biological systems are aggregated together in blocks, and local competition takes place therein for their activation (Local-Winner-Takes-All, LWTA, mechanism). Under this scheme, in each block, a given sensory stimulus activates only one neuron at a time, while the rest are inhibited to silence. Crucially, it appears that this mechanism is of stochastic nature, in the sense that the same system may produce different neuron activation patterns when presented with exactly the same stimulus at multiple times (Lansner, 2009).

Recent works in the field have shown that deep networks founded upon Stochastic LWTA activations (Panousis et al., 2019), which are trained via stochastic gradient variational Bayes and are sampled from at inference time, allow for: (i) deploying deep networks that achieve state-of-the-art adversarial robustness, without resorting to adversarial training or to elaborate extraneous defense schemes built on top of them (Panousis et al., 2021); and (ii) developing innovative Transformer-type networks for multimodal (video-to-text) modeling tasks that completely outperform existing Transformer-based networks in a Sign-Language Translation benchmark, both in terms of accuracy and imposed memory footprint (Voskou et al., 2021).

Drawing from this inspiration, in this work we propose a *stochastic Bayesian meta-sampling* framework for MAML-type algorithms that leverages deep networks founded upon Stochastic LWTA activation principles; we dub our approach Stochastic LWTA ML (StochLWTA-ML). The methodological innovations that StochLWTA-ML contributes can be summarized as follows:

- We use LWTA units with stochastic winner sampling within each block, as opposed to conventional ReLU units. This, combined with a posteriori Gaussian-distributed weights over the network synapses, results in sparse inferred representations of stochastic nature. Overall, these innovations constitute a radical departure from the currently prominent network design paradigm in ML.
- We use Bayesian Model Averaging (BMA) to perform inference; we create a population of sampled representations of the observed data, and then perform BMA over the corresponding predictions to derive the final inferential output of our algorithm. This way, our StochLWTA-ML approach reframes model-agnostic ML models in a Bayesian inference context; this gives rise to a predictive posterior distribution that allows for final predictions that are more robust to uncertainty.

We evaluate our approach considering a number of standard benchmarks in the field, namely Omniglot (Lake et al., 2017) and Mini-Imagenet (Vinyals et al., 2016). We show that our approach offers a variety of advantages over the current state-of-the-art methods, namely: (i) incurring, on average, half the predictive error rate compared to the currently state-of-the-art methods in the field; (ii) obtaining this performance with networks that comprise less trainable parameters, and therefore give rise to better computational efficiency and imposed memory footprint.

The remainder of this paper is organized as follows: In Section 2, we briefly provide some background theory information on MAML, its variants and the LWTA mechanism. Section 3 introduces our approach and provides the related training and inference algorithms. In Section 4, we perform a thorough experimental evaluation of StochLWTA-ML, and compare our findings to the current state-of-the-art. In the final Section 5, we end up with the conclusions of our work, and suggest lines of further research.

2 Preliminaries

2.1 Model-Agnostic Meta-Learning

ML (Schmidhuber, 1987), also referred to as *learning to learn* (Thrun & Pratt, 1998), has the goal of obtaining machine learning models that can learn new tasks with limited availability of data, and via only a few gradient steps, by capitalizing upon information stemming from previously learned tasks. In this context, Finn et al. (2017) suggested a *model-agnostic* algorithm for ML, that can be applied to any model trained via gradient descent. The introduced MAML algorithm initializes model parameters in a way that can be quickly adapted to several types of new tasks.

Let us consider a model with parameters $\boldsymbol{\theta}$ and a parametric form $f_{\boldsymbol{\theta}}$. When the model is adapting to an unseen task T_i, sampled from the distribution over tasks $P(T)$, MAML initially runs few steps of *inner-loop* gradient descent that yields the task-specific parameter set

$$\boldsymbol{\theta}'_i = \boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L_{T_i}(f_{\boldsymbol{\theta}}) \tag{1}$$

where α is the step size hyperparameter and L_{T_i} denotes the loss on the task T_i.

Subsequently, training proceeds to optimize the function $f_{\boldsymbol{\theta}'_i}$ with respect to the model parameters $\boldsymbol{\theta}$. Assuming that the batch of tasks has size M, we can define the targeted *meta-objective* as:

$$L_{meta}(\boldsymbol{\theta}) = \sum_{i=1}^{M} L_{T_i}(f_{\boldsymbol{\theta}'_i}) = \sum_{i=1}^{M} L_{T_i}(f_{\boldsymbol{\theta} - \alpha \nabla_{\boldsymbol{\theta}} L_{T_i}(f_{\boldsymbol{\theta}})}). \tag{2}$$

Optimization of this meta-objective over $\boldsymbol{\theta}$ yields the *outer-loop* update:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \nabla_{\boldsymbol{\theta}} \sum_{i=1}^{M} L_{T_i}(f_{\boldsymbol{\theta}'_i}) \tag{3}$$

where β stands for the *outer-loop* learning rate.

2.1.1 First-Order Meta-Learning

The *MAML* process involves expensive computations stemming from the second-order updates that the *outer-loop* entails (Eqs. (2) and (3)). To mitigate this drawback, Finn et al. (2017) developed a first-order approximation that reduces the *outer-loop* update (3) to:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} - \beta \sum_{i=1}^{M} \nabla_{\boldsymbol{\theta}'_i} L_{T_i}(f_{\boldsymbol{\theta}'_i}) \tag{4}$$

In other words, *FOMAML* computes the gradients with respect to the updated parameter values $\boldsymbol{\theta}'_i$, but omits the gradients of $\boldsymbol{\theta}'_i$ with respect to $\boldsymbol{\theta}$.

In a different fashion, the *Reptile* algorithm of Nichol et al. (2018) disposes the costly second-order updates of MAML by: (i) applying some *inner-loop* gradient descent steps using (1); and (ii) suggesting a new *outer-loop* that simply subtracts the parameters, $\boldsymbol{\theta}$, from the updates $\boldsymbol{\theta}'_i$ (instead of computing derivatives):

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \beta(\boldsymbol{\theta}'_i - \boldsymbol{\theta}). \tag{5}$$

2.2 Local Winner-Takes-All Units

LWTA is a scheme that gives rise to units that are organized into blocks; the units within a block compete between each other, and only one produces non-zero activation given some network input. Specifically, a layer of an LWTA-based network consists of R blocks. Each block $r \in \{1, \ldots, R\}$ includes J competing linear neurons. It produces an output vector $\boldsymbol{y}_r$, which contains only a single non-zero entry; this corresponds to the activation of the winner unit within the block.

Denoting as $h_r^j, j = 1, \ldots, J$, the linear activation function of the j-th unit in the r-th block, we have

$$h_r^j = \boldsymbol{w}_{r,j}^T \boldsymbol{x} \tag{6}$$

where $\boldsymbol{w}_{r,j}$ is the weight vector of unit j in r-th block and $\boldsymbol{x}$ is the input vector from the previous layer.

Then, the block output $\boldsymbol{y}_r$ becomes $\boldsymbol{y}_r = [y_r^j]_{j=1}^J$, where

$$y_r^j = g(h_r^1, h_r^2, ..., h_r^J) \tag{7}$$

and $g(\cdot)$ is the employed unit competition function. Typical LWTA networks consider a hard competition function $g(\cdot)$ that reads:

$$y_r^j = \begin{cases} h_r^j, & \text{if } h_r^j \geq h_r^i, \ \forall i = 1, \ldots, J, \ \ i \neq j \\ 0, & \text{otherwise} \end{cases} \tag{8}$$

Under this rationale, the neuron with the *highest* activation within a block *turns-off* the activations of all other units in the block, given some network input. In addition, back-propagation takes place only through the winning part of the network for a given training data point.

In contrast, in our work we employ a stochastic rationale in the competition mechanism among the units of LWTA blocks. We posit that this novel stochastic paradigm will give rise to more potent learned representations that generalize better in the context of ML tasks.

3 Proposed Approach

3.1 Deep Networks built of Stochastic LWTA Units

The proposed approach attempts to obtain more robust generalization properties over existing gradient-based networks. To this end, we consider a paradigm that yields sparse input representations of stochastic nature. Specifically, drawing some inspiration from Voskou et al. (2021) and Panousis et al. (2021), we introduce a meta-agnostic ML algorithm that relies upon the sparse and stochastic representations obtained from trained deep networks composed of stochastic LWTA layers.

A conventional deep neural network can be viewed as a mapping function h of an observation to a decision, where $h : \boldsymbol{x} \rightarrow \boldsymbol{y}$. Let us assume that the observation $\boldsymbol{x} \in R^{A_1}$ is presented to a hidden layer composed of nonlinear units with corresponding weights matrix $\boldsymbol{W} \in R^{A_1 \times A_2}$. The output of the layer is the vector $\boldsymbol{y}$ that reads $\boldsymbol{y} = \sigma(\boldsymbol{W} \cdot \boldsymbol{x})$, where $\boldsymbol{y} \in R^{A_2}$, and $\sigma(\cdot)$ is a non-linear activation function, e.g. ReLU.

In our work, stochastic LWTA blocks replace the nonlinear units. Let us consider a layer comprising R blocks of J competing linear units per block. The input $\boldsymbol{x}$ is now presented to each block through different weights that are organized into a three-dimensional matrix $\boldsymbol{W} \in R^{A_1 \times R \times J}$. Then, the j-th competing unit within r-th block computes the sum $\sum_{i=1}^{A_1}(w_{i,r,j}) \cdot x_i$ (as shown in Fig. 1). On the basis of this intermediate outcome, we proceed to establish a stochastic unit competition mechanism, under which each block determines a winning unit within the block by application of a random sampling process (described next). The stochastically selected winner unit presents its intermediate outcome to the layer output, $\boldsymbol{y} \in R^{R \times J}$, while the remainder block units are zeroed out.

Specifically, let us introduce the discrete latent indicator vector $\boldsymbol{\xi} \in \text{one_hot}(J)^R$ to denote the set of sampled winner units in the R blocks that constitute a considered stochastic LWTA layer. This vector comprises R component subvectors, where each component entails one non-zero value at the index position that corresponds to the winner unit of the respective LWTA block. On this basis, the output $\boldsymbol{y} \in R^{R \times J}$ of the stochastic LWTA layer is defined as $\boldsymbol{y} = [y_{r,j}]$, with:

$$[\boldsymbol{y}]_{r,j} = [\boldsymbol{\xi}]_{r,j} \sum_{i=1}^{A_1}(w_{i,r,j}) \cdot x_i \in R. \tag{9}$$

In Eq. (9), we postulate that the latent winner indicator variables are drawn from a Discrete distribution which is proportional to the intermediate linear computation that each unit performs. Therefore, the stronger the linearity the higher the chance of the unit winning the stochastic competition within its block. In detail, we stipulate that, a posteriori, the winner distributions yield:

$$q([\boldsymbol{\xi}]_r) = \text{Discrete}\left([\boldsymbol{\xi}]_r \middle| \text{softmax}(\sum_{i=1}^{A_1}[w_{i,r,j}]_{j=1}^{J} \cdot x_i)\right) \tag{10}$$

where $[w_{i,r,j}]_{j=1}^{J}$ denotes the vector concatenation of the set $\{w_{i,r,j}\}_{j=1}^{J}$.

Based on these arguments, we proceed to derive a novel model-agnostic ML algorithm. Specifically, as a result of the stochastic nature of the yielded sparse input representations, we are now dealing with a training objective function that has to take into consideration the involved posterior distributions. This takes the form of an evidence lower-bound (ELBO) objective, as we describe next. Capitalizing upon this Bayesian inferential machinery, we take one step further, also electing to impose a posterior density over the network weights $\boldsymbol{W}$. This results in a second source of stochasticity

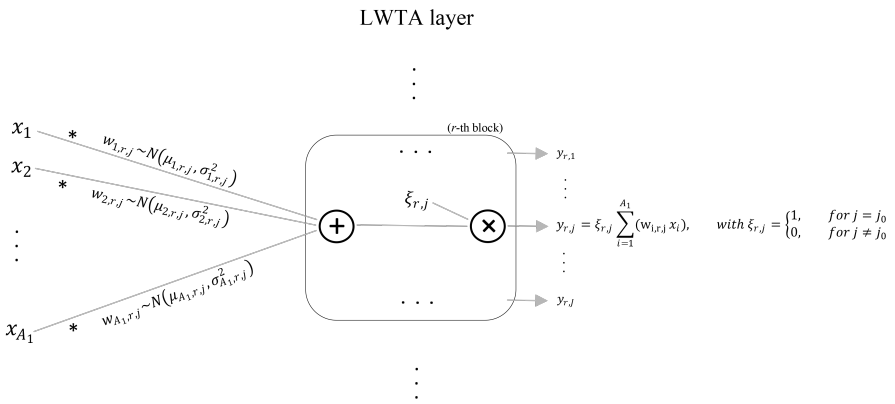


Figure 1: A zoomed-in graphical illustration of the r-th block of a stochastic LWTA layer. Input $\boldsymbol{x} = [x_1, x_2, \ldots, x_{A_1}]$ is presented to each unit in the block. Assume that the index of the winner unit is j_0. Then, the output of the layer is a vector with a single non-zero value at index j_0.

for our approach, which may further increase its generalization capacity under uncertain conditions arising from limited training data availability. We postulate:

$$q(\text{vec}(\boldsymbol{W})) = N(\text{vec}(\boldsymbol{W})|\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2)) \tag{11}$$

where $\boldsymbol{\psi} \triangleq \{\boldsymbol{\mu}, \boldsymbol{\sigma}^2\}$ are the means and variances of the Gaussian weight posteriors, respectively.

This concludes the formulation of the proposed model-agnostic ML approach.

Algorithm 1: Model training with StochLWTA-ML

Require: $P(T)$: distribution over tasks
Initialize $\boldsymbol{\psi} := \{\boldsymbol{\mu}, \boldsymbol{\sigma}^2\}$
Define outer-step size β and inner learning rate α
for *i = 1,2, . . .* **do**
  Inner training loop:
    Sample task $T_i \sim P(T)$, where T_i contains data $D_i = (X_i, Y_i) \subset D$
    Compute $L_{T_i}(\boldsymbol{\psi})$ using Eqs. (12) - (14)
    Compute adapted parameters with SGD: $\boldsymbol{\psi}'_i = \boldsymbol{\psi} - \alpha \nabla_{\boldsymbol{\psi}} L_{T_i}(f_{\boldsymbol{\psi}})$
  Outer training loop:
    Derive $\boldsymbol{\psi} \leftarrow \boldsymbol{\psi} + \beta(\boldsymbol{\psi}'_i - \boldsymbol{\psi})$
end

3.2 Model Training

Let us define an ML problem where we are given a training dataset D of tasks, T, that are governed by a distribution $P(T)$. We consider an N-way, K-shot learning setting, where each task contains K labelled examples for each of N available classes.

Our approach entails parameter sets $\boldsymbol{\theta}$ which coincide with the weights $\boldsymbol{W} \in R^{A_1 \times R \times J}$ of all network layers. These parameter sets radically depart from the current state of affairs in the literature of model-agnostic ML algorithms. Indeed, the existing approaches need to obtain point-estimates for the sets $\boldsymbol{\theta}$. In contrast, Eq. (11) introduces stochastic parameter sets $\boldsymbol{\theta}$ over which we can only obtain appropriate Gaussian posterior densities. These entail the posterior density hyperparameter sets $\boldsymbol{\psi} = \{\boldsymbol{\mu}, \boldsymbol{\sigma}^2\}$, which our training algorithm must target.

Therefore, to perform model training, we have to first initialize the trainable parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}^2$ across layers. To this end, one can appropriately exploit popular initialization schemes, such as Glorot Uniform (Glorot & Bengio, 2010). Then, our approach entails an inner-outer loop scheme, in a vein similar to existing approaches, but with some crucial differences:

1. The stochastic nature of the postulated weights, $\boldsymbol{W}$, results in the updates taking place over the posterior means, $\boldsymbol{\mu}$ and variances, $\boldsymbol{\sigma}^2$.
2. The stochastic nature of both the inferred representations and the network weights themselves implies that proper training must rely on optimization of the ELBO functional of the network. Let us consider an inner-loop dealing with task $T_i \sim P(T)$, with data $D_i = (X_i, Y_i) \subset D$. Let $\log p(D_i|\boldsymbol{W}, \boldsymbol{\xi})$ be the negative categorical cross-entropy of the postulated deep network composed of the introduced stochastic LWTA layers. Then, considering independence among network layers (mean-field approximation), we yield:
$$L_{T_i}(\boldsymbol{\psi}) = \mathbb{E}_{q(\boldsymbol{\xi}),q(\boldsymbol{W})}[\log p(D_i|\boldsymbol{W}, \boldsymbol{\xi})] - KL[q(\boldsymbol{\xi})\,||\,p(\boldsymbol{\xi})] - KL[q(\boldsymbol{W})\,||\,p(\boldsymbol{W})] \quad (12)$$
for task T_i with dataset D_i which is dealt with on the i-th training iteration. Here, for simplicity and without harming generality, we consider that the weights prior $p(\text{vec}(\boldsymbol{W}))$ is a Gaussian distribution $N(\boldsymbol{0}, \boldsymbol{I})$ and the latent variables prior $p(\boldsymbol{\xi})$ is a symmetric Discrete distribution Discrete$(1/J)$.

The posterior expectations $q(\boldsymbol{\xi}), q(\boldsymbol{W})$ in the first term of Eq. (12) cannot be computed analytically, due to the nonlinear nature of $\log p(D_i|\boldsymbol{W}, \boldsymbol{\xi})$. Thus, to render back-propagation feasible, we draw Monte-Carlo (MC) samples from the entailed Gaussian posteriors, and we resort to a continuous relaxation of the discrete latent variables $\boldsymbol{\xi}$. Specifically, to ensure low-variance gradients, we employ the Gumbel-Softmax relaxation trick for the latent variables $\boldsymbol{\xi}$ (Jang et al., 2017; Maddison et al., 2017), and the Normal reparameterization trick, $\text{vec}(\hat{\boldsymbol{W}}) = \boldsymbol{\mu} + \boldsymbol{\sigma}\boldsymbol{\epsilon}$ for the weights $\boldsymbol{W}$, with $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$. Concerning the Gumbel-Softmax trick, let $\bar{\boldsymbol{\xi}}$ be the unnormalized probabilities of the Discrete distribution $q(\boldsymbol{\xi})$ (Eq. 10), and $\tau \in (0, \infty)$ be a temperature factor that controls how closely the discrete distribution is approximated by the relaxed Gumbel-Softmax distribution. The sampled instances of $\boldsymbol{\xi}$, $\hat{\xi}_{r,j}$ become then differentiable, being expressed as:
$$\hat{\xi}_{r,j} = \text{Softmax}(\log \bar{\xi}_{r,j} + g_{r,j}),\ \forall r = 1, \ldots, R,\ j = 1, \ldots, J \quad (13)$$
where $g_{r,j} = -\log(-\log U_{r,j})$, $U_{r,j} \sim \text{Uniform}(0, 1)$.

Thus, we eventually yield:
$$\mathbb{E}_{q(\boldsymbol{\xi}),q(\boldsymbol{W})}[\log p(D_i|\boldsymbol{W}, \boldsymbol{\xi})] \approx \log p(D_i|\hat{\boldsymbol{W}}, \hat{\boldsymbol{\xi}}) \quad (14)$$
The analytical terms of KL divergences in (12) can be found in the Supplementary material. We summarize our model training algorithm in Alg. 1.

3.3 Inference Algorithm

At inference time, we draw a set of B samples of the Gaussian connection weights, using the trained posteriors $\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\sigma}^2)$. Then, we select the winning units in each block of the network by similarly sampling from the posteriors $q(\boldsymbol{\xi})$. This results in a set of B output logits of the network, which we average in a BMA fashion to obtain the final predictive outcome. We emphasize that, in this inference process, we do not employ any sort of reparameterization trick, as this is useful only for low-variance gradient-based training.

Algorithm 2: Inference with StochLWTA-ML

Require: Learned parameters $\boldsymbol{\psi} = \{\boldsymbol{\mu}, \boldsymbol{\sigma}^2\}$, input data X'
  Sample $\boldsymbol{\psi} \sim q(\boldsymbol{\psi}) = N(\boldsymbol{\psi}|\boldsymbol{\mu}, \text{diag}(\boldsymbol{\sigma}^2))$
  Sample $\boldsymbol{\xi} \sim q(\boldsymbol{\xi})$ defined in Eq. (10), for $\boldsymbol{w} = \boldsymbol{\psi}$ and $(x_i = x'_i) \in X'$
  Compute output logits, given the sampled values $\boldsymbol{\xi}$ and $\boldsymbol{\psi}$
Repeat B times
Perform BMA over the resulting set of B logits to derive the final prediction.

4 Experiments

4.1 Experimental Setup

We evaluate StochLWTA-ML on Omniglot (Lake et al., 2017) and Mini-Imagenet (Vinyals et al., 2016), which are popular few-shot image classification datasets, and compare its performance to

state-of-the-art prior results. After thorough exploration on the number of LWTA layers as well as the number of blocks for each layer and the competing units per block, we end up with using networks comprising 2 layers with 16 blocks and 2 competing units per block on the former layer, and 8 blocks with 2 neurons per block on the latter. The penultimate network layer is a Softmax. Weight mean initialization, as well as point-estimate initialization for our competitors, is performed via Glorot Uniform. Weight log-variance initialization is performed via Glorot Normal, by sampling from $N(0.0005, 0.01)$. The Gumbel-Softmax relaxation temperature is set to $\tau = 0.67$.

In the inner-loop updates, we use the Stochastic Gradient Descent (SGD) (Robbins, 2007) optimizer with a learning rate of 0.003. For the outer-loop, we use SGD with a linear annealed outer step size equal to 0, and an initial value of 0.25. Additionally, all the experiments were ran with task batch size of 25 for both training and testing mode. Inference is carried out using $B = 4$ candidate models in the context of BMA.

To ensure statistical significance, the results presented in Sections 4.2 and 4.3 stand for the average performance over three runs with different random seeds. Note that each experiment consists of 60000 iterations for training. The code was implemented in Tensorflow (Abadi et al., 2016).

4.2 Results

In Table 1, we show how StochLWTA-ML performs on Omniglot 20-way and Mini-Imagenet 5-way few-shot settings. Apart from citing the reported outcomes of our main competitors, we have also locally reproduced the original architectures with the same hyperparameters and data preprocessing as in MAML, FOMAML (Finn et al., 2017) and Reptile (Nichol et al., 2018), dubbed "original" and "local", respectively, in Table 1. For completeness sake, we also compare our findings to other state-of-the-art ML models as reported in Finn et al. (2017), including Matching Nets (Santoro et al., 2016) and LSTM Meta-Learner (Ravi & Larochelle, 2017). As we observe, our method improves upon the existing state-of-the-art in both the 1-shot and 5-shot settings. Specifically, it is apparent that the harder the dataset the more the benefits our approach offers. Indeed, in the case of Mini-Imagenet 5-way 1-shot experiments, we achieve a more than 10% increase in the predictive accuracy over the best alternatives. This gain reduces in the easiest considered case of the Omniglot 20-way 5-shot dataset; however, our approach still outperforms the existing state-of-the-art.

Table 1: N-way K-shot accuracy classification accuracies on Omniglot and Mini-Imagenet

	Omniglot 20-way		**Mini-Imagenet 5-way**	
Algorithm	1-shot	5-shot	1-shot	5-shot
Matching Nets	93.8%	98.50%	43.56%	55.31%
LSTM Meta-Learner	-	-	43.44%	60.60%
MAML (original)	95.80%	98.90%	48.70%	63.11%
MAML (local)	95.47%	98.60%	48.57%	62.87%
FOMAML (original)	-	-	48.07%	63.15%
FOMAML (local)	94.93%	98.11%	47.79%	63.08%
Reptile (original)	88.14%	96.65%	47.07%	62.74%
Reptile (local)	87.98%	96.35%	46.83%	62.40%
StochLWTA-ML	**97.78%**	**98.97%**	**52.20%**	**65.91%**

4.3 Ablation study

4.3.1 How stochastic network weights and/or Stochastic LWTA layers contribute to classification accuracy?

To gain more insights into the inferential capacity of our method, we evaluate various modifications of MAML and FOMAML. Specifically, we locally reproduce the original architectures of Finn et al. (2017), and gradually introduce the innovative components of our approach, namely the Stochastic LWTA layers and the Gaussian-distributed weights. Due to space limitations, we report here only results on the Mini-Imagenet 5-way dataset. More results can be found in the Supplementary.

First, we train conventional MAML and FOMAML networks where we impose Gaussian posteriors over the network weights, in a fashion similar to the proposed approach; we dub this setup "ReLU

(Gaussians)" in Table 2. Then, we postulate MAML and FOMAML networks that abandon ReLU layers, by employing the proposed Stochastic LWTA layers, and either train point-estimates of the layer weights or fit Gaussian posteriors; we dub these setups "LWTA (point-estimates)" and "LWTA (Gaussians)" in Table 2, respectively. Note that, to implement the "LWTA (point-estimates)" variant, we adopt the setup of Section 3, but elect to train only the Gaussian weights means, $\boldsymbol{\mu}$, while setting the sampled weight values always equal to $\boldsymbol{\mu}$. This way, we end up with a simplified ELBO expression, that reads:

$$L_{T_i}(\boldsymbol{\mu}) = \mathbb{E}_{q(\boldsymbol{\xi})}[\log p(D_i|\boldsymbol{\xi})] - KL[q(\boldsymbol{\xi})\,||\,p(\boldsymbol{\xi})] \tag{15}$$

Finally, at inference time we follow a process identical to Section 3.3.

As we see in Table 2, the full-fledged StochLWTA-ML approach yields clearly higher performance than any considered alternative, including both conventional networks postulating Gaussian weights, as well as LWTA-based networks that either fit Gaussian posteriors or point-estimates for the weights.

Table 2: Mini-Imagenet 5-way Few-Shot ablation study

Algorithm	**1-shot**	**5-shot**
MAML + ReLU (point estimates)	48.57%	62.87%
MAML + ReLU (Gaussians)	48.69 %	62.95%
MAML + LWTA (point estimates)	48.88%	63.15%
MAML + LWTA (Gaussians)	49.60%	63.92%
FOMAML + ReLU (point estimates)	47.79%	63.08%
FOMAML + ReLU (Gaussians)	47.91%	63.12%
FOMAML + LWTA (point estimates)	48.11%	63.24%
FOMAML + LWTA (Gaussians)	49.23%	64.55%
StochLWTA-ML	**52.20%**	**65.91%**

4.3.2 How does the number of samples at inference time affect accuracy?

We scrutinize the effect of the number of drawn samples, B, on inferential accuracy. To this end, we repeat our experiments using $B = 10$ candidate logits sets. In Table 3, we provide the comparative outcomes concerning the two sample size configurations of $B = 4$ and $B = 10$. As we observe, an increase in inference sample size, B, yields a slight accuracy increase, as one would normally expect. However, the aforementioned increase in sample size imposes some computational overhead, which we elaborate upon in the following Section. We argue that this overhead might not be worth it for the slight performance increase reported in Table 3.

Table 3: Effect of sample size B at inference time

	Omniglot 20-way		**Mini-Imagenet 5-way**	
Number of samples	1-shot	5-shot	1-shot	5-shot
$B = 4$	97.78%	98.97%	52.20%	65.91%
$B = 10$	96.90%	99.23%	53.08%	66.39%

4.3.3 Is there a computational time trade-off for the increased accuracy?

It is also important to investigate whether our approach represents a trade-off between accuracy and computational time compared to our competitors. To facilitate this investigation, in Table 4 we provide training iteration wall-clock times for our approach and the existing state-of-the-art (networks locally reproduced). It appears that our methodology takes on average 11% *less* training time compared to the state-of-the-art. This happens because our approach yields the reported state-of-the-art performance by employing a network architecture (that is, number of LWTA layers, as well as number of blocks and block size on each layer) that result in a total number of trainable parameters that is *much less* than the best-performing configuration of the currently state-of-the-art methods. The exact comparative figures are depicted in Table 5. The situation changes when it comes to inference: our approach imposes a slight computational time overhead compared to

the current state-of-the-art, which is rather negligible for a low number of drawn samples, B, but increases as we increase B. The provided results suggest that a selection of $B = 4$ represents a favorable accuracy/inference wall-clock time for our approach.

Table 4: Average wall-clock time comparison (in msecs) over the considered datasets of Table 1

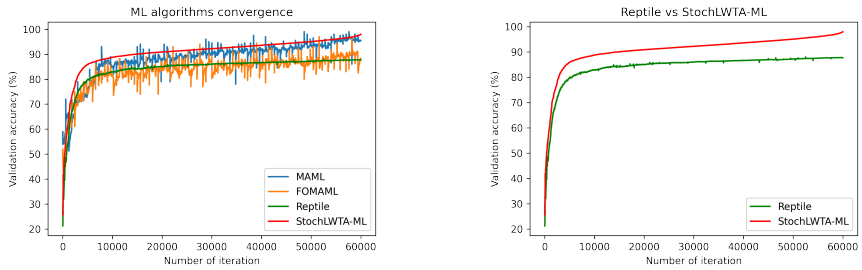
Algorithm	Training iteration	Inference (batch of 25 test examples)
MAML (local)	294.31	108.08
FOMAML (local)	292.68	108.12
Reptile (local)	324.67	109.65
StochLWTA-ML	**290.91**	119.31 ($B = 4$) 128.63 ($B = 10$)

Table 5: Number of parameters for Omniglot (D_A) 20-way and Mini-Imagenet (D_B) 5-way

	Reptile D_A	MAML D_A, FOMAML D_A
ReLU	113221	112005
LWTA	**54549**	**53259**

	Reptile D_B	MAML D_B, FOMAML D_B
ReLU	124613	123079
LWTA	**60112**	**58581**

Finally, for completeness sake we provide an example of how training for our approach converges, and how this compares to the alternatives. In Fig. 2(a), we illustrate our outcomes on the Omniglot 20-way 1-shot benchmark; similar outcomes have been observed in the rest of the considered datasets. It becomes apparent that our approach converges equally fast to the best of our competitors. In addition, it is characteristic that First-Order ML algorithms, including Reptile and StochLWTA-ML, are more stable than the Second-Order ones, that is MAML and FOMAML. Further, Fig. 2(b) compares in more detail Reptile with StochLWTA-ML. It appears that convergence of our approach is even smoother; we argue that this outcome implies an inherently stronger generalization capacity for our approach, which stems from the stochastic nature of the main network components.



(a) (b)

Figure 2: Training algorithm convergence

5 Conclusion

In this paper, we proposed a sparse and stochastic network paradigm for ML, which constitutes a radical departure over network configurations currently used in model-agnostic ML. We introduced *stochastic LWTA activations* in the context of a *variational Bayesian treatment* that gave rise to a doubly-stochastic ML framework, bearing the promise of stronger generalization capacity. We evaluated our approach using standard benchmarks in the field, and showed that it outperformed the state-of-the-art in terms of both predictive accuracy and computational costs. The results have provided strong empirical evidence supporting our claims. In the future, we plan to study the effect of StochLWTA-ML in areas related to ML, such as Continual Learning (Javed & White, 2019) and Reinforcement Learning (Zhu et al., 2020).

REFERENCES

Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S Corrado, Andy Davis, Jeffrey Dean, and Matthieu Devin. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. pp. 265–283. In *Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation*, 2016.

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 *Neural Information Processing Systems*, 2016.

Sungyong Baik, Myungsub Choi, Janghoon Choi, Heewon Kim, and Kyoung Mu Lee. Meta-learning with adaptive hyperparameters. In *Neural Information Processing Systems*, 2020.

Paul Black, Robert McCormick, Mary James, and David Pedderd. Learning how to learn and assessment for learning: a theoretical inquiry. *Research Papers in Education*, 21:119–132, 2006.

Rui Castro, Charles Kalish, Robert Nowak, Ruichen Qian, Tim Rogers, and Jerry Zhu. Human active learning. In *Neural Information Processing Systems*, 2008.

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

Chelsea Finn, Aravind Rajeswaran, Sham Kakade, and Sergey Levine. Online meta-learning. In *International Conference on Machine Learning*, 2019.

Sebastian Flennerhag, Andrei A. Rusu, Razvan Pascanu, Francesco Visin, Hujun Yin, and Raia Hadsell. Meta-learning with warped gradient descent. In *International Conference on Learning Representations*, 2020.

Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. *Journal of Machine Learning Research 9*, pp. 249–256, 2010.

Matthias Hutsebaut-Buysse, Kevin Mets, and Steven Latré. Fast task-adaptation for tasks labeled using natural language in reinforcement learning. 2019. URL https://arxiv.org/abs/1910.04040.

Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. In *International Conference on Learning Representations*, 2017.

K. Javed and M White. Meta-learning representations for continual learning. pp. 1818–1828. In *Neural Information Processing Systems*, 2019.

Niklas Kühl, Marc Goutier, Lucas Baier, Clemens Wolff, and Dominik Martin. Human vs. supervised machine learning: Who learns patterns faster? 2020. URL https://arxiv.org/abs/2012.03661.

Yunfei Lai. A comparison of traditional machine learning and deep learning in image recognition. *Journal of Physics: Conference Series*, 1314, 2019.

Brenden M Lake, Ruslan Salakhutdinov, and Joshua B Tenenbaum. Human-level concept learning through probabilistic program induction. *Science*, 350(6266):1332–1338, 2015.

Brenden M Lake, Tomer D Ullman, Joshua B Tenenbaum, and Samuel J Gershman. Building machines that learn and think like people. *Behavioral and Brain Sciences*, 40, 2017.

Anders Lansner. Associative memory models: from the cell-assembly theory to biophysically detailed cortex simulations. *Trends in neurosciences*, 32(3):178–186, 2009.

Chris J. Maddison, Andriy Mnih, and Yee Whye Teh. The concrete distribution: A continuous relaxation of discrete random variables. In *International Conference on Learning Representations*, 2017.

Alex Nichol, Joshua Achiam, and John Schulman. On first-order meta-learning algorithms. 2018. URL http://arxiv.org/abs/1803.02999.

Konstantinos Panousis, Sotirios Chatzis, Antonios Alexos, and Sergios Theodoridis. Local competition and stochasticity for adversarial robustness in deep learning. In *International Conference on Artificial Intelligence and Statistics*, 2021.

Konstantinos P. Panousis, Sotirios Chatzis, and Sergios Theodoridis. Nonparametric bayesian deep networks with local competition. In *International Conference on Machine Learning*, 2019.

Sachin Ravi and Hugo Larochelle. Optimization as a model for few-shot learning. In *International Conference on Learning Representations*, 2017.

H. Robbins. A stochastic approximation method. *Annals of Mathematical Statistics*, 2007.

Adam Santoro, Sergey Bartunov, Matthew Botvinick, Daan Wierstra, and Timothy Lillicrap. Meta-learning with memory-augmented neural networks. In *International Conference on Machine Learning*, 2016.

Jurgen Schmidhuber. On learning how to learn: The meta-meta-... hook., evolutionary principles in self-referential learning. Master's thesis, Institut f. Informatik, Tech. Univ. Munich, 1987.

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, and Todd Phillips. Hidden technical debt in machine learning systems. In *Neural Information Processing Systems*, 2015.

S. Thrun and L. (eds.). Pratt. *Learning to Learn.* Kluwer Academic Publishers, USA, 1998. ISBN 0792380479.

Ricardo Vilalta and Youssef Drissi. A perspective view and survey of meta-learning. *Artificial Intelligence Review*, 18(2):77–95, 2002.

Oriol Vinyals, Charles Blundell, Timothy Lillicrap, Koray Kavukcuoglu, and Daan Wierstra. Matching networks for one shot learning. In *Neural Information Processing Systems*, 2016.

Andreas Voskou, Konstantinos Panousis, Dimitrios Kosmopoulos, Dimitris Metaxas, and Sotirios Chatzis. Stochastic transformer networks with linear competing units: Application to end-to-end sl translation. In *International Conference on Computer Vision*, 2021.

J. X Wang, Z. Kurth-Nelson, D. Tirumala, H. Soyer, J. Z Leibo, R. Munos, C. Blundell, D. Kumaran, and M. Botvinick. Learning to reinforcement learn. In *International Conference on Machine Learning*, 2016.

Henry Zhu, Justin Yu, Abhishek Gupta, Dhruv Shah, Kristian Hartikainen, Avi Singh, Vikash Kumar, and Sergey Levine. The ingredients of real-world robotic reinforcement learning. In *International Conference on Learning Representations*, 2020.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *International Conference on Machine Learning*, 2017.