arXiv:2302.00195v2 [cs.LG] 8 Aug 2023

权重预测提升 AdamW 的收敛性

Lei Guan

国防科技大学数学系 guanleimath@163.com

摘要。 本文将权重预测引入 AdamW 优化器，以提升其训练深度神经网络模型时的收敛性。具体而言，在每次小批量训练之前，我们根据 AdamW 的更新规则预测未来权重，然后使用预测的未来权重进行前向传播和反向传播。这样，AdamW 优化器始终利用相对于未来权重而非当前权重计算的梯度来更新深度神经网络参数，从而获得更好的收敛性。我们提出的方法简单直接、易于实现，同时能有效提升 DNN 训练的收敛性。我们在图像分类和语言建模任务上进行了广泛的实验评估，以验证该方法的有效性。实验结果表明，在训练深度神经网络模型时，我们的方法能够提升 AdamW 的收敛性，并达到比 AdamW 更高的准确率。

关键词： 深度学习 · 权重预测 · 收敛 · AdamW。

1 引言

深度神经网络（DNN）模型的优化，旨在借助优化器寻找最优参数。优化器对模型的收敛起着决定性作用，因此会直接影响总训练时间。RMSprop [20], 、AdaGrad [3], 、Adam [7]和 AdamW [11], 等自适应梯度方法，目前在深度学习训练中具有核心的实用价值，因为它们能够快速训练现代深度神经网络（DNN）模型。尤其是 AdamW [11], ，也称为采用解耦权重衰减的 Adam，已被用作训练各种深度神经网络模型的默认优化器 [1, 10, 11, 20, 21]。AdamW 的主要优势在于，它提升了 Adam [7] 的泛化性能，因此在图像分类任务上的表现与动量随机梯度下降 [18]同样出色。

与其他流行的基于梯度的优化方法一样，使用 AdamW 作为优化器时，DNN 训练的每次迭代（即一次小批量训练）通常包含一次前向传播和一次反向传播，反向传播期间会计算所有参数（也称为权重）相对于目标函数的梯度。生成的梯度随后会被用于

AdamW 优化器据此计算所有参数的更新值，最终用于更新权重。使用 AdamW 更新参数的显著特点包括：1）权重的更新是连续的；2）每个小批次都会使用当前可用的权重进行前向传播和反向传播。

鉴于深度神经网络权重以连续方式更新，且 AdamW 计算出的更新值应反映权重更新的“正确”方向，我们将权重预测 [2,4] 引入 DNN 训练，以进一步加快 AdamW 的收敛。具体而言，在每次小批量训练之前，我们首先根据当前可用的权重和 AdamW 的更新规则预测权重。随后，我们使用预测的未来权重而非当前权重进行前向传播和反向传播。最后，AdamW 优化器利用相对于预测权重的梯度来更新深度神经网络参数。我们在两种典型的机器学习任务上进行了实验，包括图像分类和语言建模。实验结果表明，我们的方法在收敛和准确率方面均优于 AdamW。例如，在 CIFAR-10 数据集上训练四种卷积神经网络（CNN）模型时，与 AdamW 相比，我们的方法平均准确率提升了 0.47%（最高达 0.74%）。在宾夕法尼亚树库数据集上训练长短期记忆网络时，与 AdamW 相比，我们的方法平均困惑度降低了 5.52。

本文的贡献可总结如下：
(1) 我们首次构建了在使用 AdamW 作为优化器时，当前权重与经过若干次连续更新后的未来权重之间的数学关系。(2) 我们设计了一种将权重预测融入 AdamW 的有效方法。据我们所知，这是首次使用权重预测策略来加速 AdamW 收敛。我们认为，所提出的权重预测策略也适用于 RMSprop [20], AdaGrad [3],Adam [7],等其他常用优化方法。(3) 我们进行了广泛的实验评估，以验证所提方法的有效性。实验结果表明，在训练深度神经网络模型时，我们的方法能够加速 AdamW 收敛。

2 相关工作

使用基于梯度的优化方法训练深度神经网络模型时，不同优化方法的区别在于使用梯度更新模型参数的方式不同。一般而言，常用的一阶梯度方法可分为两类：加速随机梯度下降（SGD）系列 [15,16,18] 和自适应梯度方法 [7,23,24]。

自适应梯度方法，也称为自适应学习方法，已在以往研究中得到深入探讨，并广泛应用于深度学习训练。与使用统一学习率来更新所有参数的 SGD 方法（例如动量随机梯度下降 [18]）截然不同，自适应梯度方法会为

每个参数分别计算特定的学习率 [24]。2011 年，Duchi 等人 [3]提出了 AdaGrad，它能够根据先前迭代的历史梯度动态调整学习率，并利用所有先前梯度的平方和更新模型参数。Zeiler [23] 提出了 AdaDelta，旨在缓解 AdaGrad 学习率持续衰减的问题。AdaDelta 无需手动调整学习率，并且对含噪梯度信息具有鲁棒性。Tieleman 和 Hinton [20] 改进了 AdaGrad，并提出 RMSprop。与 AdaGrad 一样，RMSprop 通过逐元素计算调整学习率，然后更新变量。RMSprop 的一大特点是能够避免学习率衰减过快。为了兼顾 AdaGrad 和 RMSprop 的优点，Kingma 和 Ba [7]提出了另一种著名的自适应梯度方法 Adam，它已成为深度学习训练中极为重要的选择。Loshchilov 和 Hutter [11] 发现，Adam 泛化性能不佳的主要原因在于，其 L_2 正则化效果不如竞争方法动量随机梯度下降。为此，他们为 Adam 提出了解耦权重衰减正则化，也称为 AdamW。实验结果表明，AdamW 显著提升了 Adam 的泛化性能，并且在处理图像分类任务时展现出与动量随机梯度下降 [18] 相当的竞争力。为了同时实现快速收敛和良好的泛化性能，Zhuang 等人 [24]提出了另一种名为 AdaBelief 的自适应梯度方法，它根据对当前梯度方向的“信念”来调整步长。其他自适应梯度方法还包括 AdaBound [12],RAdam [9], Yogi [22],等。值得注意的是，所有这些自适应梯度方法都有一个共同特点：权重连续更新，每次小批量训练都会使用当前可用的权重执行前向传播和反向传播。

此前，权重预测用于解决异步流水线并行中的权重不一致问题。Chen 等人 [2] 使用平滑梯度替代真实梯度，以便在使用动量随机梯度下降 [18] 作为优化器时预测未来权重。Guan 等人 [4] 提出使用 Adam [7] 的更新值进行权重预测。然而，这两种方法都使用权重预测来确保流水线训练中的权重一致性，而没有考虑权重预测对优化器本身的影响。

3 种方法

在任意第 t 次（$t \geq 1$）迭代之前，我们假设当前可用的深度神经网络权重为 $\boldsymbol{\theta}_{t-1}$。给定初始学习率 $\gamma \in \mathbb{R}$、动量因子 $\beta_1 \in \mathbb{R}$和 $\beta_2 \in \mathbb{R}$，以及权重衰减值 $\lambda \in \mathbb{R}$，我们将 AdamW 的更新重新表述为

AdamW [11]为

$$\begin{aligned}
&\boldsymbol{\theta}_t = (1-\gamma\lambda)\boldsymbol{\theta}_{t-1} - \frac{\gamma\hat{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{v}}_t}+\epsilon}, \\
&\text{s.t.} \begin{cases}
\mathbf{g}_t = \nabla_t f(\boldsymbol{\theta}_{t-1}), \\
\mathbf{m}_t = \beta_1 \cdot \mathbf{m}_{t-1} + (1-\beta_1)\cdot \mathbf{g}_t, \\
\mathbf{v}_t = \beta_2 \cdot \mathbf{v}_{t-1} + (1-\beta_2)\cdot \mathbf{g}_t^2, \\
\hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \\
\hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t}.
\end{cases}
\end{aligned} \tag{1}$$

在 (1) 中，$\mathbf{m}_t$ 和 $\mathbf{v}_t$ 分别表示一阶矩向量和二阶矩向量，ϵ 是可防止除以零的平滑项。

令 $\boldsymbol{\theta}_0$ 表示深度神经网络模型的初始权重，则在接下来的s 次连续小批量训练中，深度神经网络权重通过以下方式更新：

$$\begin{aligned}
\boldsymbol{\theta}_1 &= (1-\gamma\lambda)\boldsymbol{\theta}_0 - \frac{\gamma\hat{\mathbf{m}}_1}{\sqrt{\hat{\mathbf{v}}_1}+\epsilon}, \\
\boldsymbol{\theta}_2 &= (1-\gamma\lambda)\boldsymbol{\theta}_1 - \frac{\gamma\hat{\mathbf{m}}_2}{\sqrt{\hat{\mathbf{v}}_2}+\epsilon}, \\
&\cdots \\
\boldsymbol{\theta}_s &= (1-\gamma\lambda)\boldsymbol{\theta}_{s-1} - \frac{\gamma\hat{\mathbf{m}}_s}{\sqrt{\hat{\mathbf{v}}_s}+\epsilon},
\end{aligned} \tag{2}$$

其中，对于任意 $i \in \{1, 2, \cdots, s\}$，有

$$\begin{cases}
\mathbf{g}_i = \nabla_i f(\boldsymbol{\theta}_{i-1}), \\
\mathbf{m}_i = \beta_1 \cdot \mathbf{m}_{i-1} + (1-\beta_1)\cdot \mathbf{g}_i, \\
\mathbf{v}_i = \beta_2 \cdot \mathbf{v}_{i-1} + (1-\beta_2)\cdot \mathbf{g}_i^2, \\
\hat{\mathbf{m}}_i = \frac{\mathbf{m}_i}{1-\beta_1^i}, \\
\hat{\mathbf{v}}_i = \frac{\mathbf{v}_i}{1-\beta_2^i}.
\end{cases} \tag{3}$$

将 (2) 中的所有权重更新方程相加，可得

$$\begin{aligned}
&\boldsymbol{\theta}_s = \boldsymbol{\theta}_0 - \gamma\lambda\sum_{i=0}^{t-1}(\boldsymbol{\theta}_0 + \boldsymbol{\theta}_1 + \cdots + \boldsymbol{\theta}_{t-1}) - \sum_{i=1}^{t}\frac{\gamma\hat{\mathbf{m}}_i}{\sqrt{\hat{\mathbf{v}}_i}+\epsilon}, \\
&\text{s.t.} \begin{cases}
\mathbf{g}_i = \nabla_i f(\boldsymbol{\theta}_{i-1}), \\
\mathbf{m}_i = \beta_1 \cdot \mathbf{m}_{i-1} + (1-\beta_1)\cdot \mathbf{g}_i, \\
\mathbf{v}_i = \beta_2 \cdot \mathbf{v}_{i-1} + (1-\beta_2)\cdot \mathbf{g}_i^2, \\
\hat{\mathbf{m}}_i = \frac{\mathbf{m}_i}{1-\beta_1^i}, \\
\hat{\mathbf{v}}_i = \frac{\mathbf{v}_i}{1-\beta_2^i}.
\end{cases}
\end{aligned} \tag{4}$$

众所周知，权重衰减值 λ 通常设得极小（例如 $5e^{-4}$），而学习率 γ 通常设为小于 1 的值（例如 0.01）。因此，$\gamma\lambda$非常接近于零，所以可以忽略公式 (4) 右侧的第二项。由此可得

以下方程：

$$\boldsymbol{\theta}_s \approx \boldsymbol{\theta}_0 - \sum_{i=1}^{s} \frac{\gamma \hat{\mathbf{m}}_i}{\sqrt{\hat{\mathbf{v}}_i} + \epsilon}. \tag{5}$$

(5) 表明，给定初始权重 $\boldsymbol{\theta}_0$，经过 s 次连续更新后的权重可以近似计算。相应地，给定 $\boldsymbol{\theta}_t$，经过 s 次连续更新后的权重可以通过以下方式近似计算：

$$\boldsymbol{\theta}_{t+s} \approx \boldsymbol{\theta}_t - \sum_{i=t+1}^{t+s} \frac{\gamma \hat{\mathbf{m}}_i}{\sqrt{\hat{\mathbf{v}}_i} + \epsilon}. \tag{6}$$

由 (6) 可知，给定初始权重 $\boldsymbol{\theta}_t$，通过用 $\boldsymbol{\theta}_t$ 减去权重连续相对变化量 s 的总和，可以近似计算出 $\boldsymbol{\theta}_{t+s}$。请注意，每次迭代中权重的相对增量应^反映该次迭代中的权重更新趋势。在 (6) 中，$\sqrt{\gamma}\ {}^{i}\mathbf{m}\hat{}\mathbf{v}_i+\epsilon$ 应反映更新权重 $\boldsymbol{\theta}_t$ 的“正确”方向，因为它由 AdamW 计算得出；权重以连续方式沿着惯性方向更新。

因此，我们可以将 (6) 中的 $\sum_{i=t+1}^{t+s} \frac{\gamma}{\hat{}\mathbf{m}\sqrt{}}\ {}^{i}\hat{}\mathbf{v}_i+\epsilon$ 替换为 $s\ \frac{\gamma}{\hat{}\mathbf{m}\sqrt{}}\ {}^{t+1}\hat{\mathbf{v}}_{t+1}+\epsilon$，以便在仅有 $\boldsymbol{\theta}_t$、$\mathbf{g}_t$ 和权重预测步数 s 可用时，近似预测 $\boldsymbol{\theta}_{t+s}$。请注意，在任意第 t 次迭代中，随机目标函数的梯度，即 $i.e.$，$\mathbf{g}_t = \nabla_t(\boldsymbol{\theta}_{t-1})$，可在反向传播完成后计算。令 $\boldsymbol{\theta}_{t+s}$ 表示针对 $\boldsymbol{\theta}_{t+s}$ 近似预测的权重，我们可以构建 $\boldsymbol{\theta}_t$ 与 ^$\boldsymbol{\theta}_{t+s}$ 之间的数学关系：

$$\begin{aligned}
&\hat{\boldsymbol{\theta}}_{t+s} = \boldsymbol{\theta}_t - s\frac{\gamma \hat{\mathbf{m}}_{t+1}}{\sqrt{\hat{\mathbf{v}}_{t+1}} + \epsilon}, \\
&\text{s.t.} \begin{cases} \mathbf{g}_t = \nabla_t f(\boldsymbol{\theta}_{t-1}), \\ \mathbf{m}_t = \beta_1 \cdot \mathbf{m}_{t-1} + (1-\beta_1) \cdot \mathbf{g}_t, \\ \mathbf{v}_t = \beta_2 \cdot \mathbf{v}_{t-1} + (1-\beta_2) \cdot \mathbf{g}_t^2, \\ \hat{\mathbf{m}}_t = \frac{\mathbf{m}_t}{1-\beta_1^t}, \\ \hat{\mathbf{v}}_t = \frac{\mathbf{v}_t}{1-\beta_2^t}. \end{cases}
\end{aligned} \tag{7}$$

下面我们展示使用 AdamW [11] 作为优化器时，如何将权重预测融入 DNN 训练。算法 1 展示了详细信息。在 DNN 训练开始前，需要确定权重预测步长 s 及其他超参数。每次迭代时，在前向传播开始前，应缓存当前可用的权重 $\boldsymbol{\theta}_t$（第 4 行）。然后使用公式（7）执行权重预测，并生成预测权重 ^ ^ $\boldsymbol{\theta}_{t+s}$（第 5 行）。随后，使用预测权重 $\boldsymbol{\theta}_{t+s}$ 进行前向传播和反向传播（第 6 行和第 7 行）。最后，恢复缓存的权重 $\boldsymbol{\theta}_t$，并使用 AdamW 优化器对其进行更新（第 8 行和第 9 行）。

算法 1 AdamW 的权重预测

要求： 权重预测步长 s，其他超参数，例如 γ、 β_1、 β_2、 γ、 ϵ。
1：初始化或加载深度神经网络权重 $\boldsymbol{\theta}_0$。
2： $t \leftarrow 1$。
3：**当** 未满足停止准则时 **执行**
4： 缓存当前权重 $\boldsymbol{\theta}_t$。
5： 使用 (7) 计算 $\hat{} \boldsymbol{\theta}_{t+s}$ 。
6： 使用 $\hat{} \boldsymbol{\theta}_{t+s}$ 进行前向传播。
7： 使用 $\hat{} \boldsymbol{\theta}_{t+s}$ 进行反向传播。
8： 恢复缓存的权重 $\boldsymbol{\theta}_t$。
9： 使用 AdamW 优化器更新权重 $\boldsymbol{\theta}_t$ 。
10： $t \leftarrow t + 1$。
11: **结束 while**

4 实验

4.1 实验设置

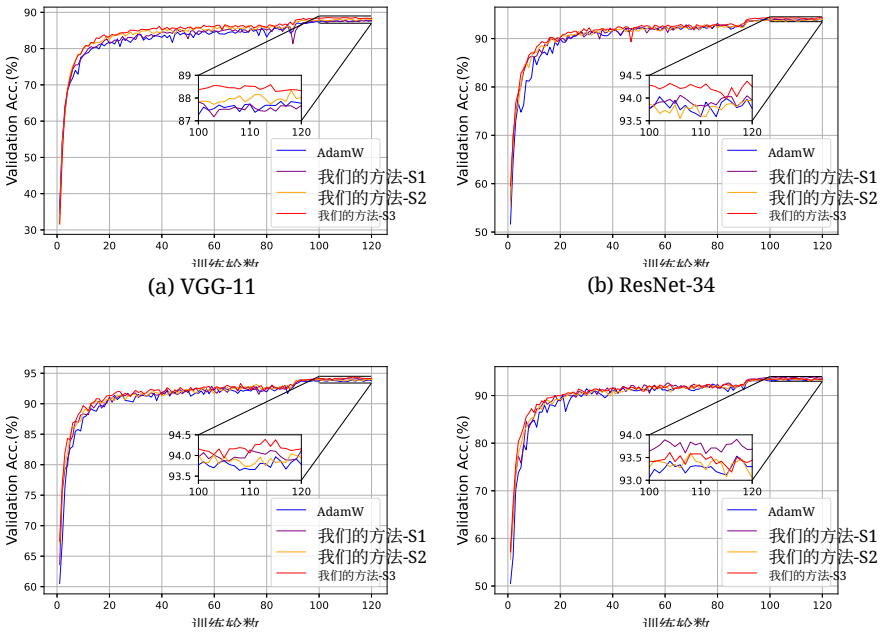
本节主要将我们提出的方法与 AdamW [11]进行比较。我们使用三种不同的权重预测步长（即， $s = 1$、 $s = 2$和$s = 3$）评估了我们提出的方法，为方便起见，分别称为我们的方法-S1、我们的方法-S2和我们的方法-S3。我们在两种不同的机器学习任务上进行了实验评估：使用四种卷积神经网络模型在 CIFAR-10 [8] 数据集上进行图像分类，以及使用两种长短期记忆网络 [13] 模型在宾州树库 [14]数据集上进行语言建模。所有实验均在一台多 GPU 平台上进行，该平台配备四块 NVIDIA Tesla P100 GPU，每块显存为 16GB。该平台的中央处理器为 Intel Xeon E5-2680，配备 128GB DDR4-2400 片外主存。

CIFAR-10 数据集共包含 60k 张图像，其中 50k 张用于训练，10k 张用于验证。宾夕法尼亚树库数据集包含 929k 个训练词、73k 个验证词以及 82k 个测试词。在图像分类任务中，使用的 CNN 模型包括 VGG-11 [17], ResNet-34 [5],DenseNet-121 [6], 和 Inception-V3 [19]。在语言建模任务中，我们训练了两种规模的长短期记忆网络模型：单层长短期记忆网络和双层长短期记忆网络。每一层配置了 650 个单元，并在非循环连接上应用了 50% 的丢弃。

我们对所有卷积神经网络模型进行了 120 个训练轮数的训练，每个小批次的大小为 128。学习率初始化为 $1e^{-4}$，并在第 90 个训练轮数时降为原来的十分之一。对于训练单层和双层长短期记忆网络模型，我们将每个小批次的大小设为 20。两个长短期记忆网络模型均训练 100 个训练轮数，初始学习率为 0.01，并在第 60 和第 80 个训练轮数时将学习率降低为原来的十分之一。对于 AdamW [11] 和我们的方法，我们始终使用默认参数进行评估，即 $\beta_1 = 0.9$、 $\beta_2 = 0.999$ 和 $\epsilon = 10^{-8}$。两种方法的权重衰减均设为 $\lambda = 5e^{-4}$。

4.2 CIFAR-10 上的卷积神经网络

本节报告了在 CIFAR-10 数据集上训练四种卷积神经网络模型时的实验结果。表1总结了最高验证 top-1 准确率，表 2 则展示了最小验证损失。图1分别描绘了使用 AdamW、我们的方法-S1、我们的方法-S2 和我们的方法-S3 训练卷积神经网络时，验证准确率随训练轮数变化的学习曲线。图 2 展示了验证损失随训练轮数变化的学习曲线。



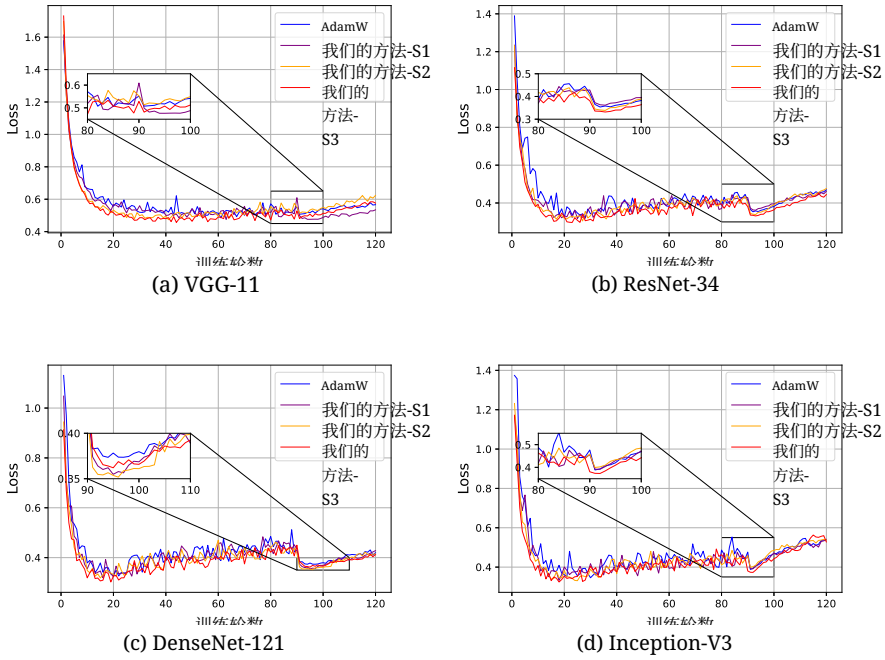
(a) VGG-11 (b) ResNet-34

(c) DenseNet-121 (d) Inception-V3

图1. VGG-11、ResNet-34、DenseNet-121 和 Inception-V3 在 CIFAR-10 上的验证准确率与训练轮次的关系。

表1. CIFAR-10 上的最高验证 top-1 准确率。**越高** 越好。

模型	AdamW	我们的方法-S1	我们的方法-S2	我们的方法-S3
VGG-11	87.85%	87.83%	88.34%	**88.59**%
ResNet-34	94.03%	94.06%	93.95%	**94.37**%
DenseNet-121	93.97%	94.13%	94.04%	**94.39**%
Inception-V3	93.53%	**93.90**%	93.60%	93.61%



(a) VGG-11 (b) ResNet-34

(c) DenseNet-121 (d) Inception-V3

图 2. VGG-11、ResNet-34、DenseNet-121 和 Inception-V3 在 CIFAR-10 上训练轮次与验证损失的关系。

表 2. CIFAR-10 上的最小验证损失。**越低** 越好。

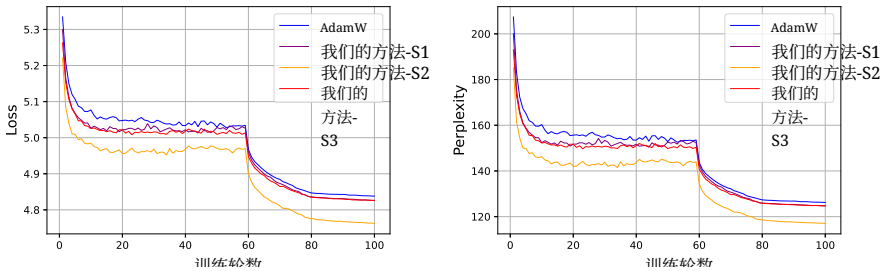
模型	AdamW	我们的方法-S1	我们的方法-S2	我们的方法-S3
VGG-11	0.485	0.475	0.474	**0.456**
ResNet-34	0.323	0.318	0.305	**0.297**
DenseNet-121	0.327	0.319	0.319	**0.302**
Inception-V3	0.346	0.333	0.331	**0.324**

根据对表1和图1的观察，我们可以立即得出以下结论。首先，图1显示，我们的方法在采用不同权重预测步数时，其学习曲线与 AdamW 的学习曲线高度吻合，但收敛速度更快，尤其是在训练初期。图1(a)、1(b)、1(c)和1(d)中的学习曲线还表明，在训练结束时，我们的方法通常能取得比 AdamW 更高的验证准确率。其次，表1显示，就获得的最高验证准确率而言，我们的方法在所有评估的卷积神经网络模型上均优于 AdamW。

具体而言，我们的方法在验证 top-1 准确率上始终高于 AdamW。与 AdamW 相比，在训练 VGG-11、ResNet-34、DenseNet-121 和 Inception-V3 时，我们的方法分别提升了 0.74%、0.34%、0.42% 和 0.37%。平均而言，我们的方法相比 AdamW 提高了 0.47% 的 top-1 准确率（最高可达 0.74% ）。第三，对比我们的方法 -S1 、我们的方法 -S2 和我们的方法 -S3 的实验结果可以看出，我们的方法采用不同权重预测步数时均能稳定取得良好结果，这表明我们的方法的性能不受权重预测步长设置的影响。具体来说，实验结果显示，我们的方法-S3 在 VGG-11、ResNet-34 和 DenseNet-121 上表现最佳，而我们的方法-S1 在 Inception-V3 上表现最佳。观察表 2 和图 2 也可以得出类似结论。我们的方法始终比 AdamW 获得更低的验证损失，这再次验证了权重预测能够加快 AdamW 在训练深度神经网络模型时的收敛。

4.3 在 PennTreeBank 上的长短期记忆网络

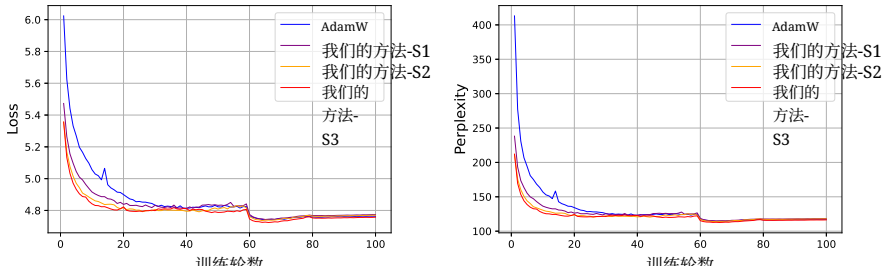
本节报告在宾夕法尼亚树库数据集 [14] 上训练单层和双层长短期记忆网络模型的实验结果。图3和图4展示了学习曲线。表3列出了所得的最小困惑度（越低越好），表4汇总了所得的最小验证损失（越低越好）。



(a) 损失 (b) 困惑度

图3. 在宾州树库上训练单层长短期记忆网络。左：损失与训练轮数的关系；右：困惑度与训练轮数的关系。

我们可以从实验结果中得出以下结论。首先，如表3所示，对于单层和双层长短期记忆网络模型，我们的方法均比 AdamW 获得更低的困惑度和验证损失，这验证了我们的方法具有快速收敛和良好准确率。其次，对于单层长短期记忆网络，我们的方法在 $s = 2$ 下比 AdamW 的困惑度低 9.22。对于双层长短期记忆网络，我们的方法在 $s = 3$ 下比 AdamW 的困惑度低 2.02。平均而言，我们的方法



(a) 损失 (b) 困惑度

图 4. 在宾州树库上训练双层长短期记忆网络。左：损失随训练轮数的变化；右：困惑度随训练轮数的变化。

表 3. 宾州树库上的最低困惑度。**越低** 越好。

模型	AdamW	我们的方法-S1	我们的方法-S2	我们的方法-S3
单层长短期记忆网络	126.21	124.75	**116.99**	124.71
双层长短期记忆网络	114.68	114.80	113.99	**112.64**

表4. 宾州树库上的最小验证损失。**越低** 越好。

模型	AdamW	我们的方法-S1	我们的方法-S2	我们的方法-S3
单层长短期记忆网络	4.838	4.826	**4.762**	4.826
双层长短期记忆网络	4.742	4.743	4.736	**4.724**

比 AdamW 的困惑度低 5.52。其次，根据图3(a)和图4(a)以及表4中的损失随训练轮数变化的学习曲线，也可以得出类似结论。我们的方法始终比 AdamW 获得更低的验证损失，这再次验证了权重预测可以加快 AdamW 的收敛。

5 结论

为进一步提升 AdamW 的收敛速度，本文将权重预测引入 DNN 训练。我们提出的方法有一个显著特点：根据 AdamW 的更新预测未来权重，并使用这些未来权重进行前向传播和反向传播。具体而言，我们构建了当前权重与未来权重之间的数学关系，并设计了一种将权重预测融入 DNN 训练的有效方法。该方法易于实现，且能有效提升 DNN 训练的收敛速度。在图像分类和语言建模任务上的实验结果验证了该方法的有效性。

在训练深度神经网络模型时，权重预测也应能很好地应用于 RMSprop [20], AdaGrad [3], 和 Adam [7]等其他自适应优化方法。未来，我们希望将权重预测应用于这些常用的优化方法。

参考文献

1. Bai, Y., Mei, J., Yuille, A.L., Xie, C.：Transformer 是否比卷积神经网络更具鲁棒性？神经信息处理系统进展 **34**，26831–26843（2021） 2. Chen, C.C., Yang, C.L., Cheng, H.Y.：通过在多 GPU 平台上采用模型并行，实现高效且稳健的深度神经网络并行训练。arXiv 预印本 arXiv:1809.02839（2018） 3. Duchi, J., Hazan, E., Singer, Y.：用于在线学习和随机优化的自适应次梯度方法。机器学习研究期刊 **12(7)**（2011） 4. Guan, L., Yin, W., Li, D., Lu, X.：Xpipe：面向多 GPU 深度神经网络训练的高效流水线模型并行。arXiv 预印本 arXiv:1911.04610（2019） 5. He, K., Zhang, X., Ren, S., Sun, J.：用于图像识别的深度残差学习。载于：IEEE 计算机视觉与模式识别会议论文集。第 770–778 页（2016） 6. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.：密集连接卷积网络。载于：IEEE 计算机视觉与模式识别会议论文集。第 4700–4708 页（2017） 7. Kingma, D.P., Ba, J.：Adam：一种随机优化方法。arXiv 预印本 arXiv:1412.6980（2014） 8. Krizhevsky, A., Hinton, G., 等：从微型图像中学习多层特征（2009） 9. Liu, L., Jiang, H., He, P., Chen, W., Liu, X., Gao, J., Han, J.：关于自适应学习率的方差及其他问题。arXiv 预印本 arXiv:1908.03265（2019） 10. Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., Guo, B.：Swin Transformer：使用移位窗口的分层视觉 Transformer。载于：IEEE/CVF 国际计算机视觉会议论文集。第 10012–10022 页（2021） 11. Loshchilov, I., Hutter, F.：解耦权重衰减正则化。arXiv 预印本 arXiv:1711.05101（2017） 12. Luo, L., Xiong, Y., Liu, Y., Sun, X.：具有学习率动态边界的自适应梯度方法。arXiv 预印本 arXiv:1902.09843（2019） 13. Ma, X., Tao, Z., Wang, Y., Yu, H., Wang, Y.：利用远程微波传感器数据进行交通速度预测的长短期记忆神经网络。《交通研究 C 部分：新兴技术》**54**，187–197（2015） 14. Marcinkiewicz, M.A.：构建大型英语标注语料库：宾州树库。《使用大型语料库》**273**（1994） 15. Nesterov, Y.：一种求解凸规划问题的方法，其收敛速率为 mathcal { } 1 ^ {2} 27 O (/k)。载于：《苏联数学文献》第 . 卷。Polyak, B.T.：加速迭代方法收敛的一些方法。《苏联计算数学与数学物理》**4(5)**，1–17（1964） 17. Simonyan, K., Zisserman, A.：用于大规模图像识别的超深卷积网络。arXiv 预印本 arXiv:1409.1556（2014） 18. Sutskever, I., Martens, J., Dahl, G., Hinton, G.：初始化和动量在深度学习中的重要性。载于：国际机器学习会议。第 1139–1147 页。PMLR（2013）

19. Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J., Wojna, Z.：重新思考用于计算机视觉的 Inception 架构。载于：IEEE 计算机视觉与模式识别会议论文集。第 2818–2826 页（2016） 20. Tieleman, T., Hinton, G.：第 6.5 讲——RMSprop，Coursera：机器学习神经网络。多伦多大学技术报告 **6** （2012） 21. Wang, P., Wang, X., Luo, H., Zhou, J., Zhou, Z., Wang, F., Li, H., Jin, R.：缩放 ReLU 对视觉 Transformer 训练至关重要。载于：AAAI 人工智能会议论文集。第 36 卷，第 2495–2503 页（2022） 22. Zaheer, M., Reddi, S., Sachan, D., Kale, S., Kumar, S.：用于非凸优化的自适应方法。神经信息处理系统进展 **31**（2018） 23. Zeiler, M.D.：AdaDelta：一种自适应学习率方法。arXiv 预印本 arXiv:1212.5701（2012） 24. Zhuang, J., Tang, T., Ding, Y., Tatikonda, S.C., Dvornek, N., Papademetris, X., Duncan, J.：Adabelief 优化器：根据对观测梯度的信念调整步长。神经信息处理系统进展 **33**，18795–18806（2020）