**Guo Attentive Weights Generation for Few Shot Learning via Information Maximization**

会议：CVPR2020

代码：https://github.com/Yiluan/AWGIM

**摘要**

小样本图像分类旨在从有限的标记数据中学习分类器。 归因于其简单性和有效性，生成分类权重已在许多元学习方法中用于小样本图像分类。 在这项工作中，我们将介绍通过信息最大化（AWGIM）进行小样本学习的注意力加权生成，它引入了两个新颖的贡献：i）生成的权重和任务内数据之间的互信息最大化； 这使生成的权重可以保留任务信息和特定的查询样本。 ii）自注意力和交叉注意力路径，对任务和单个查询的上下文进行编码。 在广泛的实验中，这两个贡献都非常有效。 总体而言，AWGIM具有最先进的竞争力。

1. **引言**

虽然深度学习方法在计算机视觉方面取得了巨大的成功[14]，自然语言处理[9]，强化学习[38]，但它们对大量标记数据的渴望限制了仅可用于训练的数据很少的应用场景 。 相反，人类能够从有限的数据中学习，这是深度学习方法所需要的。 因此提出了小样本学习来使深度模型能够从很少的样本中学习[10]。

元学习是解决一些镜头问题的有前途的方法[43，11，39，33，36]。 在元学习方法中，该模型提取了跨不同任务的高级知识，因此它可以使自己快速适应新出现的任务[37，2]。 有几种元学习方法可用于小样本学习，例如基于梯度的[11，33]和基于度量的[39，41]。 在这些不同的方法中，权重生成已显示出简单公式化的有效性[31、32、12、13]。 通常，权重生成方法将学习如何根据有限的标记数据为不同任务生成分类权重。 但是，一项任务中不同查询样本的固定分类权重可能不是最优的。

在这项工作中，我们将介绍通过信息最大化进行小样本学习的注意力加权生成（AWGIM），以解决这一局限性。AWGIM对以整个支持集和单个查询样本为条件的分类权重的概率分布进行建模。在我们的模型中，采用了由注意块组成的两条路径来编码上下文信息和特定于查询的信息。但是，我们在实验中显示，查询样本和支持集之间的交叉注意力不足以生成适用于各种查询数据的分类权重。 特别是，某些特定于查询的信息在权重生成过程中丢失。

为了解决这个问题，我们从InfoGAN [6]中得到启发。 特别是，在训练GAN时，[6]建议通过最大化结构化潜在代码和生成器输出之间的互信息（MI）来学习解缠结表示。MI最大化有助于将结构化潜在代码的信息保留在生成器输出中。 以类似的精神，我们应用MI最大化以将查询/支持样本的信息保留在生成的权重中。我们的贡献是：

我们通过最大化生成的权重与支持/查询数据之间的相互信息，解决了小样本分类的权重生成问题。借助MI最大化，权重生成器能够生成适应各种查询样本的分类权重。

我们建议在两个单独的路径中对任务上下文和单个查询样本进行编码。 在两条路径中都应用了注意力机制来捕获上下文信息。

我们进行了广泛的实验，结果表明AWGIM与最先进的方法相比具有优势。 我们还将进行详细分析，以验证AWGIM中每个组件的贡献。计算开销很小，这是因为小样本问题样本少的特性。 自适应分类权重的产生还导致更快的收敛。

**2.相关工作**

**2.1 小样本学习**

最近，从很少的带标签的训练数据中学习变得越来越受到关注。 现有的大多数成功方法都使用元学习来解决此问题，并且可以分为几类。 在基于梯度的方法中，学习了所有任务的最佳初始化方法[11]。 [33]直接学习了元学习器LSTM来优化给定的少数镜头分类任务。 [40]通过梯度学习了每个层的激活转换，以更好地适应当前任务。 在基于度量的方法中，了解了查询样本和支持样本之间的相似性度量[19、43、39、41、23]。 在一些工作中还考虑了空间信息或局部图像描述符，以计算更丰富的相似性[25、24、45]。

一些作品已经探索了直接生成分类权重。[12]将分类权重生成为基础类和新颖类的权重的线性组合。类似地，[32]和[31]都从受过训练的特征提取器的激活中生成了分类权重。在[13]中使用了图神经网络去噪自动编码器。[29]提出从每个任务的损失梯度中产生“快速权重”。 所有这些方法都没有考虑为不同的查询示例生成不同的权重，也没有考虑最大化MI。

还有一些其他方法可用于小样本分类。生成模型用于生成或幻化更多数据[50，44，7]。 [5]和[21]直接使用了封闭解的方案。[26]在转导图上集成标签传播以预测查询类标签。

**2.2. Attention**

注意机制在计算机视觉[46，30]和自然语言处理[3，42]中显示出巨大的成功。 它可以有效地对来自特定上下文的查询和键值对之间的交互进行建模。 基于关键字和查询是否指向同一实体这一事实，人们将注意力称为自我关注或交叉关注。 在这项工作中，我们同时使用两种注意力来对任务和查询任务信息进行编码。 最相似的工作是“注意神经过程” [17]，它也需要自我注意和交叉注意。但是，我们通过最大化MI来关注小样本图像分类。 与之形成鲜明对比的是，[17]从随机过程的角度研究回归问题，并优化了变异目标。 有一些作品[47，16]利用空间注意力来增强特征，而我们并不依赖于空间线索，而是着重于建模与自我/交叉注意力的相互作用。

**2.3互信息**

给定两个随机变量x和y，互信息I（x， y）衡量了一个已知随机变量时不确定性的降低。 它定义为联合分布p（x，y）与边际分布p（x）⊗p（y）乘积之间的Kullback-Leibler散度，



当x和y独立时，p（x，y）= p（x）⊗p（y），因此I（x，y）= 0，这表明知道x不会揭示有关y的任何信息。当y是x的确定性函数时，I（x，y）达到最大值。MI已被广泛应用于生成对抗网络[6]，自我监督学习[15]，视觉问题生成[20]等。最近，MI在小样本学习中被引入作为记忆问题的正则化[48]。 具体而言，查询标签和支持数据之间的MI最大化或查询标签和元参数之间的MI最小化。 在[22]中，采用封闭形式的解决方案最大化了学习的二进制代码和标签之间的MI。 取而代之的是，我们通过生成具有MI下限变化的精确权重来直接解决小样本分类问题。

**3.拟议方法**

在本节中，我们提供问题的表述。 然后，我们讨论最相关的工作并揭示其局限性。我们从第3.3节中的理论分析得出目标函数。 总体模型在第3.4节中详细介绍。

**3.1问题表述**

遵循针对小样本分类的许多流行的元学习方法，我们在情节式训练范式下提出问题[43，11]。 从未知任务分布P（T）采样的一次N-way-K-shot任务T包括支持集和查询集：



其中，。 支持集S包含NK标记的样本。 查询集Q包括ˆx，我们需要基于S预测ˆx的标签ˆy。在下面的讨论中，我们分别使用和（ˆx，ˆy）表示支持（来自类）和查询集的样本。 在元测试期间，如果标记了S，则在Q上评估元学习方法的性能。元训练和元测试中使用的类是不相交的，因此元学习模型需要学习可转让的跨任务高级知识并快速适应新任务。

我们提出的方法遵循通用框架来生成分类权重[31、32、36、12、13]。在这个框架中，有一个特征提取器来输出图像特征嵌入。 元学习者需要为不同的任务生成分类权重。

**3.2潜在嵌入优化**

潜在嵌入优化（LEO）[36]是与我们的工作最相关的权重生成方法之一。在LEO中，以S为条件的u生成潜码z，记为z = u（S）。 可以使用生成函数v从z解码分类权重w，w = v（z）。 在内部循环中，w用于计算支撑集上的损耗（通常是交叉熵），然后更新z：



其中表示仅在S上评估损失。 更新后的潜在码z′用于用v解码新的分类权重w′。在外循环中对查询集Q采用w′，然后LEO的目标函数可以写为



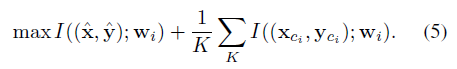
其中代表u和v的参数。LEO通过学习一个低维的潜在空间来避免在内部循环中更新高维w，从中可以使用采样的z来生成w。

LEO和AWGIM之间有两个重要区别。 首先，LEO依靠内部更新（公式3）来指导v生成适合输入任务的权重。取而代之的是，AWGIM是经过培训的前馈网络，可最大限度地提高MI，使其完全适合不同的任务。 其次，AWGIM学习为每个查询样本生成最佳分类权重，而LEO则根据一个任务的支持集生成固定权重。

**3.3信息最大化以生成权重**

我们的目标是为很少有标签训练数据的一个采样任务生成分类权重。 换句话说，我们要为一个任务T定义模型p（w | T）。 注意，在LEO中生成的分类权重对不同的查询样本不敏感，查询样本是任务T的一部分。 为了解决这个问题，我们可以在权重生成过程中对查询特定的信息进行编码，然后学习模型p（w | ˆx，S）。 但是，在实验过程中观察到的有关ˆx的信息可能会在生成过程中被忽略。

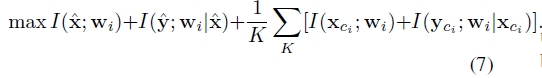
为了解决此限制，我们建议最大化生成的权重w与查询以及支持数据之间的MI。 在不失一般性的前提下，我们在以下讨论中考虑ci类的分类权重wi。 目标函数可以描述为



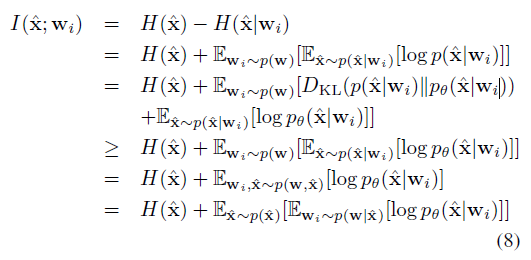
根据MI的链式规则，我们有



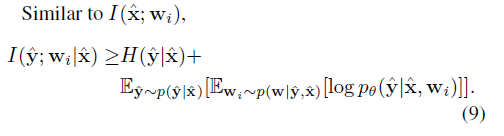
等式6在5中对两个项都成立。因此，目标函数可以写为



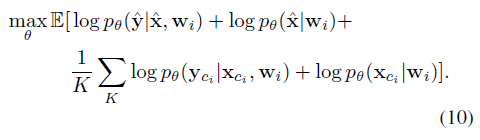
在方程7中直接计算MI是很困难的，因为像这样的真实后验分布仍然未知。 因此，我们使用变分信息最大化[4，6]来计算等式5的下界。我们使用p（ˆx | wi）近似真实的后验分布，其中代表模型参数。 结果，我们有



H（·）是随机变量的熵。 H（ˆx）是给定数据的常数。 我们可以将此下限最大化，作为真实MI的代理。



我们可以对类ci的支持数据（进行相同的推导。 将下界放回方程式7。为清楚起见，省略常数熵项和期望下标，我们有了新的目标函数为



给定生成的分类权重，第一项和第三项将使用网络参数分别最大化支持和查询数据的标签对数似然性。 这等效于最小化预测和地面真相之间的交叉熵。 此外，我们假设和是高斯分布。因此，可以通过最小化L2重建损失来实现对数似然最大化。 总的来说，应用MI最大化，我们得到具有交叉熵损失和重构损失的目标函数，这将在3.5节中讨论。

**3.4注意权重生成**

我们提出的方法的框架如图1所示。假设我们有一个特征提取器，它可以是简单的4层Convnet或更深的Resnet。 包括在采样任务T中的所有图像都由该特征提取器处理，并且随后被表示为d维向量，即，。 有两种路径分别对任务上下文和单个查询样本进行编码，分别称为上下文路径和关注路径。 两条路径的输出被串联在一起，作为输入到生成器的分类权重。 生成的分类权重不仅用于预测ˆx的标签，而且还可以最大化MI的下限。

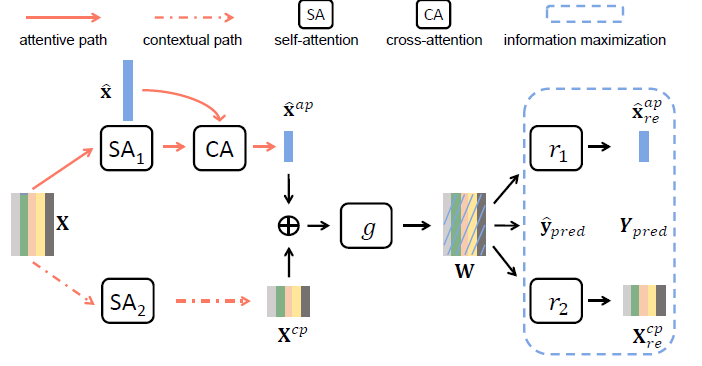
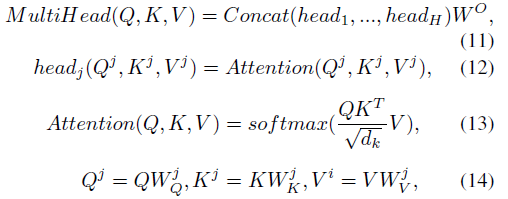


图1.我们建议的AWGIM概述。 输入任务为5-way-1-shot，其中X作为支持集，而作为一个查询示例。支持集中数据的不同颜色表示不同的类别。 上下文路径中的编码过程会生成上下文感知的支持表示。 同样，注意路径使查询样本具备任务知识。 这两种途径都是通过注意力机制实现的。 重复执行以与连接。 权重生成器g将这些级联表示作为输入，以生成特定于的分类权重W，用带有斜杠的彩色矩阵表示。W可以用来预测和X的类别标签。W也可以用来通过两个网络r1和r2重构生成器g的输入。 以此方式，互信息的下限被最大化，并且g被强制生成对不同查询样本敏感的分类权重。 在元测试期间，r1和r2被丢弃。

**3.4.1上下文和注意路径**

我们使用多头注意力网络进行编码。注意机制的使用是将一项任务中的样本之间的交互/关系建模为任务特定的属性。 先前的工作为此目的应用了关系网络[36]。 我们使用更高级的多头注意力，因为它在建模来自不同表示子空间的交互中具有优势[42]。 H头的多头注意力可以描述为



其中Q，K，V是查询，键，值矩阵。 是第j个头的权重矩阵。是输出的权重矩阵。dk是键的尺寸。将原始Q添加到公式11的输出中，以将训练稳定化为残差学习。

编码过程包括两个路径，即上下文路径和注意路径。 上下文路径旨在仅学习由 [42]参数化的多头自注意网络支持集的表示。cp，sa分别代表上下文路径，自我注意。



上下文路径的输出因此包含有关任务的更丰富的信息，以后可用于权重生成。 dh <d是隐藏维。

现有的权重生成方法仅根据支持集生成分类权重，等效于使用上下文路径。 但是，以这种方式生成的分类权重可能不是最佳的，缺乏对不同查询样本的必要适应性。 我们通过引入一条注意力路径来解决此问题，在该路径中，单个查询示例将关注任务上下文，然后将其用于生成分类权重。 因此，分类权重适用于不同的查询样本，并且也了解任务上下文。 换句话说，我们的上下文/注意路径分别对全局/局部任务结构进行建模。

在关注路径上，支持集上使用了一个新的多头自我关注网络对全局任务信息进行编码，



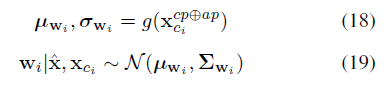
与上下文路径中的不同，因为上下文路径中的自注意力网络强调生成分类权重。 相反，在以下交叉注意中，起到了提供Value上下文的作用，不同的查询样本将为其提供这些上下文。 共享相同的自我注意网络可能会限制两条路径中学习的表示的表达能力。 遵循交叉关注网络应用于每个查询样本和任务感知支持集，以产生，



**3.4.2权重生成器**

上下文路径的输出与串联。 然后我们有。 是特定于一个查询以生成分类权重的。

被输入到权重生成器g中：，由g参数化。 我们假设分类权重遵循具有对角协方差的高斯分布。g输出分布参数，我们使用重新参数化技巧[18]从学习的分布中采样权重，如公式18和19所示。

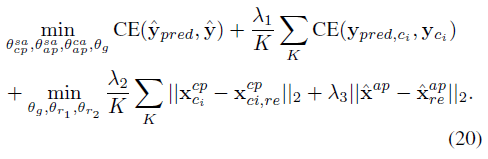


是协方差矩阵，其中是对角线条目。采样的分类权重表示为。 为了降低复杂度，我们计算每个类别的K个分类权重的平均值，使其具有。 可以通过计算查询数据的预测。 支持数据的预测也可以计算为。

除了权重生成器g外，我们还有另外两个解码器和，分别由进行参数设置。 它们都采用生成的权重W作为输入，用于学习等式10中的。换句话说，由于是直接输入，因此r1和r2分别学习重构。 r1和r2的输出表示为。

**3.5训练与推论**

在元训练中使用的目标函数10等效于



CE在这里代表交叉熵。 由于我们将等式10中的对数似然转换为等式20中的均方误差或交叉熵以进行优化，因此等式20中每个项的值与等式10中的相应对数似然相差一个常数乘数。 因此，我们必须确定超参数，以权衡不同的项。 重建损耗用于更新r1，r2用于重建，g用于权重生成。 这是因为我们希望两条途径中的注意力模块专注于对表达表示进行编码以进行后续分类。 在最后三个item的帮助下，生成的分类权重被迫使携带有关支持数据和特定查询样本的信息。 应该注意的是，该损失函数是针对一项任务中的一个查询示例计算的。 在元训练期间，一个任务中有一定数量的查询样本，而一批中有多个任务，这些样本被用于计算等式20的平均值。在元测试中，被用作ci类的分类权重而无需采样。

**3.6 复杂度分析**

上下文路径中的编码过程由于自我关注而导致计算复杂度。同样，关注路径的计算复杂度为。 总的来说，复杂度为O。 但是，由于小样本学习问题的特性，因此的值通常可以忽略不计。 | Q |的值 它是与任务相关的，可以通过矩阵乘法并行实现交叉注意。此外，AWGIM在不影响性能的情况下避免了内部更新，从而进一步缩短了推理时间。 因此，当| Q | 如果不是非常大，则所引起的计算开销可以忽略不计。

**4.实验**

**4.1数据集和协议**

我们在miniImageNet [43]和tieredImageNet [34]这两个常用的基准数据集上进行实验，以与其他方法进行比较并分析我们的模型。 这两个数据集都是ILSVRC-12数据集的子集[35]。 miniImageNet包含100个随机采样的类，每个类600个图像。 我们遵循[33]中的训练/测试划分，其中64个类别用于元训练，16个类别用于元验证，20个类别用于元测试。 tieredImageNet是一个更大的数据集，共有608个类和779,165张图像。 它们是从ImageNet [8]层次结构中的34个更高级别的节点中选择的。 来自20个高级节点的351个类用于元训练，来自6个节点的97个用于元验证，来自8个节点的160个用于元测试。

为了公平地比较，我们使用作者(https://github.com/deepmind/leo)提供的LEO [36]中的相同图像功能。他们在元训练集上训练了28层宽残差网络[49]。然后，每个图像都由640维向量表示，该向量用作我们网络的输入

对于N-way-K-shot实验，我们随机采样来自元训练集的N个类，每个类包含K个样本作为支持集和15个样本作为查询集。 与其他作品类似，我们训练5-way-1-shot和5-shot模型。在元测试期间，如最近的工作[11、39、36]所述，从元测试集中采样了600个N-way K-shot任务，并以95％的置信区间报告了查询集的平均准确性。

**4.2实施细节**

我们使用TensorFlow [1]来实现我们的方法。 d = 640是要素嵌入的尺寸。 dh设置为128。关注模块中的磁头H数量设置为4。g，r1和r2是具有256个隐藏单元的2层MLP。 通过元验证性能，我们使用。

ADAMW [27]用于优化权重衰减为1×10-6的网络。 初始学习速率对于5-way-1-shot设置为0.0002，对于5-way-5-shot设置为0.001，每15,000次迭代将其递减0.2。 我们训练该模型进行50,000次迭代。 5-way-1-shot批量为64，5-way-5-shot批量为32。 类似于LEO [36]，我们首先在元训练集上训练模型，然后通过验证结果选择最佳超参数。 然后我们一起在元训练和元验证集上训练模型使用固定的超参数。

**4.3与其他方法的比较**

我们将我们的方法AWGIM在两个数据集上的性能与近年来提出的几种最新方法进行了比较。 [26]评估了MAML，原型网，关系网在tieredImageNet上的结果。 在[13]中报告了使用WRN-28-10作为特征提取器的miniImageNet上Dynamic的结果。其他结果报告在相应的原始论文中。 我们还包括特征提取器的骨干网络，以供参考。 miniImageNet和tieredImageNet的结果分别显示在表1和2中。

表1.与miniImageNet上其他方法的准确性比较。 前3个结果突出显示。 我们注意到，我们的AWGIM经过固定图像特征训练，以便与LEO进行合理比较[36]。 相反，MetaOptNet [21]是通过端到端的特征提取器进行训练的。 tieredImageNet上的结果相同。

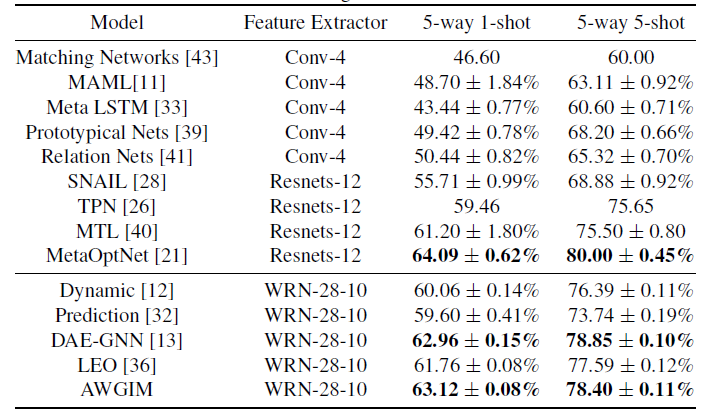


表2.与tieredImageNet上其他方法的准确性比较。 前3个结果突出显示。

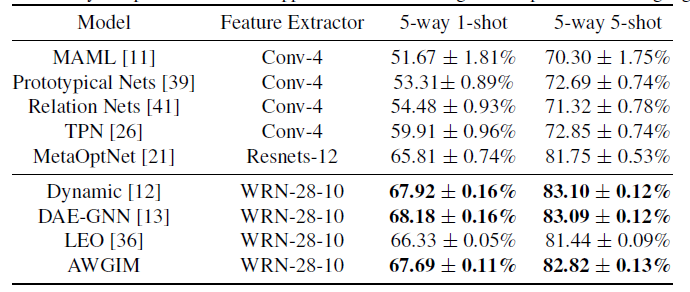


表1和表2的上半部分显示了属于不同元学习类别的方法，例如基于度量的（匹配网络[43]，原型网络[39]），基于梯度的（MAML [11]，MTL [40] ]），基于图（TPN [26]）。 底部显示了分类权重生成方法，包括动态[12]，预测[32]，DAE-GNN [13]，LEO [36]。

AWGIM在两个数据集上均达到前三名的最高准确度，并且与最佳结果相提并论。 特别是，AWGIM在所有设置下都可以胜过LEO。 应该注意的是，AWGIM训练有从WRN-28-10提取的固定图像特征，以便与LEO进行合理的比较。 相反，MetaOptNet [21]是通过端到端的特征提取器进行训练的。 从MetaOptNet论文[21]中的表4中，我们可以看到，大部分性能提升源于强大的特征提取器。包括数据增强，重量衰减，下落阻滞等在内的几种训练技术可显着提高性能。

**4.4分析**

我们对AWGIM进行了详细分析，如表3所示。我们将LEO [36]的结果包括在内以供参考。

表3.对我们提议的AWGIM的分析。 在上半部分，删除了注意路径以与LEO进行比较。 在底部，提供了针对不同组件的消融分析。 我们还随机调整所生成的分类权重，以表明它们对于不同的查询样本而言确实是最佳的。

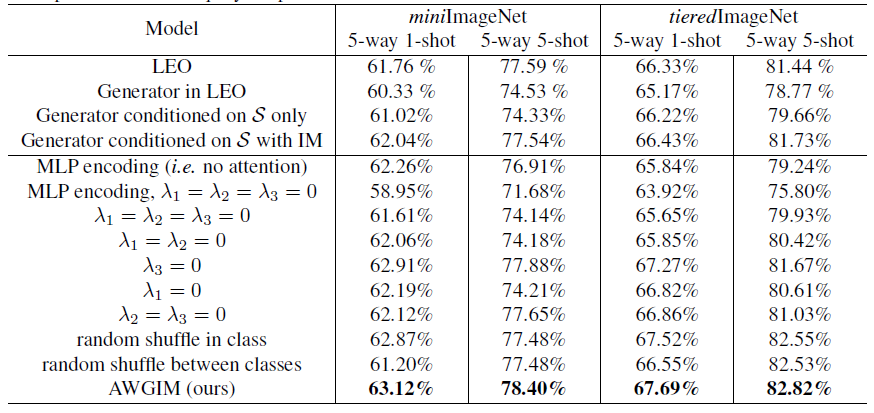
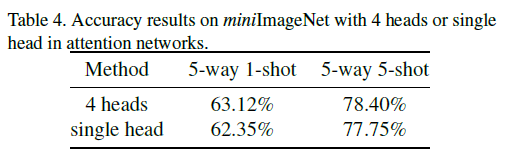


表3的上部显示了注意路径的影响。“ LEO中的生成器”表示LEO中没有内部更新。 在编码过程中，我们实现了两个生成器，仅包括上下文路径。 使用查询集上的交叉熵训练“仅基于S的生成器”，这类似于“ LEO中的生成器”，而没有内部更新。 与“ LEO中的生成器”相比，它可以实现相似或稍好的结果，这意味着对支持集的自我关注不比LEO中用于建模任务上下文的关系网络差。 “生成器以IM为条件，以S为条件”表示我们为支持集添加了交叉熵损失和重构损失。 通过最大化信息，我们的生成器可以获得比LEO更好的性能。

通过将注意模块替换为2层MLP（表示为“ MLP编码”）来研究注意的效果。 更具体地说，上下文路径中的一个MLP用于支持集，而细心路径中的另一个MLP用于查询样本。 我们可以看到，即使不注意对任务上下文信息进行编码，为了信息最大化，“ MLP编码”也可以达到接近LEO的精度。 但是，如果对于MLP编码，让，则性能会明显下降，这表明了最大化信息的重要性。

表4进一步阐明了多头注意力的贡献。我们用单头注意力代替了两条路径中的多头注意力，并在miniImageNet数据集上进行了5-way-1-shot和5-way-5-shot实验。 我们可以清楚地看到，多头注意力可以提高性能。 特别是对于单发实验，单头注意力的结果接近于MLP编码，这表明当标记的支持数据极为匮乏时，单头注意力难以解决。



对分别进行消融分析，以研究信息最大化的效果。

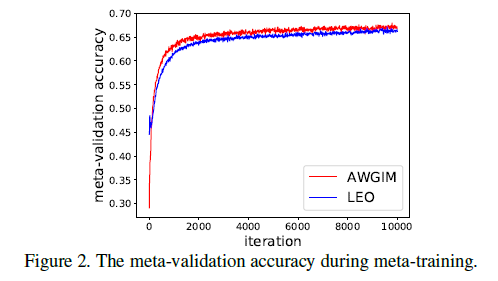
首先，将都设置为0。在这种情况下，精度类似于“仅以S为条件的发电机”，这表明生成的分类权重不适合当前任务，即使使用注意路径。 还可以观察到，最大化权重和支持之间的MI更为关键，因为会大大降低准确性，而。我们进一步研究了分类对支持和重建的相对重要性 。会显着影响性能，这意味着支持标签预测对于信息最大化至关重要。

通过改组分类权重来研究分类权重是否适合于不同的查询样本。 特别是，我们在相同类别内的查询样本之间以及不同类别之间的分类权重进行了混洗。 假设一个任务中每个类有T个查询样本。 可以重塑为。然后，我们沿着第一轴和第二轴随机调整此权重张量。 结果在表3中显示为“类别之间的随机洗牌”和“类别上的随机洗牌”。对于5次1次射击实验，类别之间的随机洗牌会显着降低准确性，而类别上的随机洗牌也不会产生太大影响 许多。 这表明，当支持数据非常有限时，针对同一类别的查询样本所生成的权重彼此非常相似，而对于不同类别而言却有所不同。

当支持集中有更多标记数据时，两种随机混洗在5-way-5-shot实验中显示的结果非常接近甚至相同，均比原始结果差。 这意味着在5-way-5-shot设置中，每个查询样本的生成分类权重更加多样化和特定。 可能的原因是，更大的支持集为每个查询示例提供了更多知识来估计最佳分类权重。

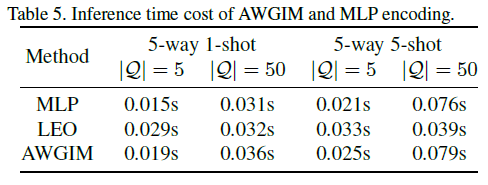
**4.5收敛**

在收敛速度方面，我们将AWGIM与LEO进行了比较。 两种方法的批次大小均设置为16。 我们使用作者调整的超参数来训练LEO。绘制了在5路1拍miniImageNet上进行元训练时设定的元验证的准确性，如图2所示。我们可以清楚地看到，除了前几次迭代外，AWGIM的收敛速度比LEO快，并且优于LEO。



**4.6推理时间成本**

我们测量AWGIM的推理时间，以表明它引起的计算开销最小。 一个MLP路径和LEO均与AWGIM进行比较，后者的时间复杂度为O（NK + | Q |）和O（（NK）2）。 由于AWGIM和MLP的时间复杂度取决于| Q |，因此我们测试了不同数量的查询样本。 我们在miniImageNet上使用了两个设置，并且批次大小设置为64。处理100个批次，并报告一个批次的平均消耗时间。 所有这些实验都是在同一计算设备上进行的。 结果示于表5。可以观察到| Q |时。 很小，由于避免了内部更新，AWGIM比LEO更快。 当| Q | 很大，MLP和AWGIM的执行速度都较慢。 值得注意的是，与MLP编码相比，在AWGIM中使用自我注意和交叉注意会产生可忽略的开销。



**5.结论**

在这项工作中，我们介绍了通过信息最大化进行注意力加权生成的几种镜头图像分类方法。AWGIM学习通过两个编码路径为任务中的每个查询样本生成最佳分类权重。为此，在生成的权重与查询，支持数据之间的相互信息的下限被最大化。AWGIM的有效性通过在两个基准数据集上的竞争表现和广泛的分析得以证明。