BERT 能否避免在序列任务中遗忘？一项探测研究

Mingxu Tao[1,2], Yansong Feng[1,3], Dongyan Zhao[1,2][l]北京大学王选计算机技术研究所，中国[2]北京大学数据科学中心，中国[3]北京大学教育部计算语言学重点实验室，中国{thomastao, fengyansong, zhaody}@pku.edu.cn

arXiv:2303.01081v1 [cs.CL] 2 Mar 2023

摘要

大型预训练语言模型有助于在各种自然语言处理（NLP）任务上达到当前最先进水平，但在按序列增量学习一系列任务时，仍然会遭受遗忘。为缓解这一问题，近期研究通过稀疏经验回放和局部适应来增强现有模型，取得了令人满意的性能。然而，本文发现，诸如 BERT 的预训练语言模型具有按序列学习的潜在能力，即使不进行任何稀疏记忆回放也能实现这一点。为了验证 BERT 保持旧知识的能力，我们在固定 BERT 参数的情况下，采用并重新微调单层探针网络。我们在两类 NLP 任务上研究这些模型：文本分类和抽取式问答。实验表明，在极其稀疏的回放甚至完全没有回放的情况下，BERT 实际上能够长期为先前学习的任务生成高质量的表示。我们进一步提出了一系列新方法，用于解释遗忘机制以及记忆复习如何在任务增量学习中发挥重要作用，从而弥合这一新发现与以往关于灾难性遗忘的研究之间的差距[1]。

1 引言

持续学习旨在从随时间推移而不断到来的数据流中获取知识（Ring，1994年；Thrun，1998年；Chen 与 Liu，2018年）。作为持续学习中的一个热门领域，任务增量学习要求模型学习一系列任务，同时不遗忘先前学到的知识。在实际场景中，模型可以依次在任务流上进行训练，从而在新任务到来后避免每次都对所有现有数据进行全面重新训练。在自然语言处理中，尽管许多大规模预训练语言模型（预训练语言模型）不断在各种基准测试集上取得新的最佳成绩，但它们无法直接应用于任务增量学习场景。这些模型在学习新任务时，往往会在先前见过的任务上表现不佳。例如，一个依次在文本分类任务上训练的 $\text{BERT}_{\text{BASE}}$模型，在学习新任务后，可能无法对第一个任务做出任何正确预测，准确率几乎为零（d'Autume 等人，2019年）。这一现象被称为 灾难性遗忘 （McCloskey 与 Cohen，1989年；French，1999年；Rosenstein 等，2005年）。许多现有工作设计了新型架构或组件，以缓解模型在增量学习过程中的遗忘问题（Kirkpatrick 等，2017年；Zenke 等，2017年；Rebuffi 等，2017年；Mallya 与 Lazebnik，2018年；d'Autume 等人，2019年；Pfeiffer 等，2020年；Sun 等，2020年；Geng 等，2021年；Jin 等，2022年；Qin 等，2022年）。其中，d'Autume 等人（2019年）发现，通过稀疏记忆回放增强的自然语言处理模型能够在很大程度上避免遗忘。他们的方法在每学习 10,000 个未见实例后，便从旧任务中随机采样 100 个实例进行回放。考虑到该方法仅凭 100 个实例、经过 4 个步骤[2]，就能恢复处理先前任务的能力，我们不禁思考：像 *BERT* 这样的预训练语言模型在学习一系列任务时，是否真的会遭受遗忘？

[1]代码将发布于 https://github.com/kobayashikanna01/plms_是_终身_学习者[2]使用大小为 32 的训练批次，采样 100 个实例意味着只需 4 个步骤即可恢复遗忘的知识。

Mehta 等人（2021年）揭示，在适当的超参数设置下，经过预训练的模型比未经预训练的模型更不容易出现灾难性遗忘。然而，在本文中，我们重点关注记忆回放的频率。我们想知道，BERT 编码器在极低的回放频率、甚至完全不进行回放的情况下，是否仍能像 d' Autume 等人（2019年）所展示的那样，保持从先前任务中学到的知识。

探测研究已成为考察模型可解释性的热门工具（Tenney 等人，2019；Jawahar 等人，2019）。例如，Wu 等人（2022）通过比较采用不同持续学习策略训练的预训练语言模型的表现，探测模型的持续学习能力。本文主要关注预训练语言模型是否具有长期保留已学知识的内在能力。我们追踪 BERT 在学习相应任务之前、期间和之后 对特定任务的编码能力。通过比较在不同回放频率下训练的模型与未使用记忆回放训练的模型的探测结果，我们发现，BERT 本身在学习一系列任务时能够避免遗忘。这与关于灾难性遗忘的现有研究有些相悖，也促使我们进一步探究不同任务的样例表示在参数空间中如何组织。受先前研究（Gao 等人，2019；Wang 等人，2020a）的启发，我们将一个类别的表示子空间定义为凸锥，并提供一种获取最窄解的算法。借助这一工具，我们发现：在不使用记忆回放学习若干任务后，不同任务类别的表示子空间会彼此重叠。然而，同一任务类别的子空间始终互不重叠。前者从表示这一全新视角解释了任务增量学习中的灾难性遗忘，后者则解释了为何 BERT 即使不使用回放，也有潜力对先前任务进行编码。

本研究的主要贡献如下：

(1) 我们进行了全面研究，定量刻画了像 BERT 这样的预训练语言模型在连续学习一系列任务时，其表征能力会发生怎样的变化。据我们所知，我们首次追踪了 BERT 在持续学习新任务时对已学任务的编码能力。

(2) 我们的研究发现，BERT 实际上能够保持对已学任务的编码能力，并且即使在长期内仅进行极其稀疏的回放，甚至完全不进行记忆回放的情况下，也有很大潜力为先前任务生成高质量表征；这一发现与先前研究相悖。

(3) 我们进一步研究了任务内及不同任务间已学习表示子空间的拓扑结构，并发现遗忘现象可从两个方面来理解：任务内遗忘和任务间遗忘（第 4 节），从而能够解释我们的发现与先前研究相悖的原因。

2 背景

遵循先前研究（Biesialska 等，2020），我们将任务增量语言学习设定为：模型应从一系列任务中学习；在后续任务的训练过程中，无法访问先前任务的样本，但可以同时获取当前任务中所有类别的样本。

形式化地说，输入训练流由 K 有序任务 $\mathcal{T}_1,\ \mathcal{T}_2,\ \cdots,\ \mathcal{T}_K$ 组成，其中我们观察来自任务 $\mathcal{T}_k$ 的分布 $\mathcal{P}_k(\mathcal{X},\ \mathcal{Y})$ 的 n_k 个样本，记作 $\left\{\left(\boldsymbol{x}_i^k,\, y_i^k\right)\right\}_{i=1}^{n_k}$。我们的训练目标是构建一个通用模型 $f_\theta: \mathcal{X} \mapsto \mathcal{Y}$ 通过最小化对所有样例取平均的负对数似然，使其能够以有限数量的参数 θ 处理所有任务：

$$\mathcal{L}(\theta) = -\frac{1}{N}\sum_{i=1}^{N} \ln P\left(y_i \,|\, \boldsymbol{x}_i\,;\, \theta\right),$$

其中 $N = \sum_{t=1}^{K} n_t$ 是所有训练样例的数量。

2.1 研究的模型

在自然语言处理中，一个模型可以分为两部分：文本编码器和任务解码器，其参数分别为 θ^{enc} 和 θ^{dec}。

文本编码器 与 MbPA++(d'Autume 等人，2019年)和 Meta-MbPA (Wang 等人，2020b)类似，我们使用 $\text{BERT}_{\text{BASE}}$ (Devlin 等人，2019年)作为文本编码器，根据给定的词元生成向量表示。

在文本分类中，我们使用添加在序列开头的 [CLS] 标记的表示来聚合所有标记的信息。对于输入标记序列 $\boldsymbol{x}_i$，其中 $x_{i,\ 0}$ 是 [CLS], $\text{BERT}_{\text{BASE}}$ 将生成对应的向量 $\{\boldsymbol{v}_{i,\,j}\}_{j=1}^{L}$，其中 $L = |\boldsymbol{x}_i|$。因此，我们将编码器模型的输出表示为： $f_{\theta^{enc}}(\boldsymbol{x}_i) = \boldsymbol{v}_{i,\ 0}$。

对于抽取式问答，我们采用 SQuAD 1.1（Rajpurkar 等人，2016年）的任务设置，与先前的工作（d' Autume 等人（2019年））一致。输入标记 $\boldsymbol{x}_i$ 由上下文 $\boldsymbol{x}_i^{\text{ctx}}$和查询 $\boldsymbol{x}_i^{\text{que}}$ 拼接而成，两者由特殊标记 [SEP] 分隔。

任务解码器 对于文本分类，我们在 $\text{BERT}_{\text{BASE}}$ 编码器之后添加线性变换和 Softmax 层。遵循 d' Autume 等人（2019年），我们为不同任务的所有类别采用统一解码器，此处 θ^{dec} 是 $\{\boldsymbol{W}_y\}_{y\in\mathcal{Y}}$ 的组合：

$$P(\hat{y}=\alpha|\boldsymbol{x}_i) = \frac{\exp\left(\boldsymbol{W}_\alpha^\top f_{\theta^{enc}}(\boldsymbol{x}_i)\right)}{\sum_{y\in\mathcal{Y}}\exp\left(\boldsymbol{W}_y^\top f_{\theta^{enc}}(\boldsymbol{x}_i)\right)} = \frac{\exp\left(\boldsymbol{W}_\alpha^\top \boldsymbol{v}_{i,0}\right)}{\sum_{y\in\mathcal{Y}}\exp\left(\boldsymbol{W}_y^\top \boldsymbol{v}_{i,0}\right)},$$

对于问答，模型会从原始上下文中提取一个片段，即确定该片段的起始和结束边界。我们的问答解码器包含两部分线性层 $\boldsymbol{W}_{\text{start}}$ 和$\boldsymbol{W}_{\text{end}}$，分别用于确定起始和结束位置。上下文中第 t个词元作为答案片段起始位置的概率可计算如下：

$$P\left(\text{start}=x_{i,t}^{\text{ctx}}|\boldsymbol{x}_i^{\text{ctx}};\,\boldsymbol{x}_i^{\text{que}}\right) = \frac{\exp\left(\boldsymbol{W}_{\text{start}}^\top \boldsymbol{v}_{i,t}^{\text{ctx}}\right)}{\sum_{j=1}^{L^{\text{ctx}}}\exp\left(\boldsymbol{W}_{\text{start}}^\top \boldsymbol{v}_{i,j}^{\text{ctx}}\right)},$$

其中， L^{ctx} 是上下文的长度，结束边界的概率形式类似。进行预测时，我们将两个边界的概率分布视为相互独立。

2.2 稀疏经验回放

实际上，人类依靠复习来保持长期知识，而这些知识建立在储存过往经历的情景记忆之上。受此启发，梯度情景记忆（Lopez-Paz & Ranzato，2017年）及其他方法在学习过程中引入记忆模块 $\mathcal{M}$。随后，训练样例可以按照预先设定的频率存入记忆中以供复习。

记忆构建 训练期间，每个已见样例都会以固定速率 γ 加入记忆。如果我们从第 k 个任务中抽取 n_k 个样例，那么在学习 $\mathcal{T}_k$ 后，预期会在 $\mathcal{M}$ 中增加 γn_k 个实例。

回放原则对于经验回放，我们需要设定一个固定的稀疏回放率 r。每当模型从当前任务中学习了 N_{tr} 个样本后，就会从 $\mathcal{M}$ 中抽取 $\lfloor rN_{tr}\rfloor$ 个样本并重新学习。为了确保可比性，我们在所有实验中都将存储率设为 $\gamma = 0.01$、重放频率设为 $r = 0.01$，与先前的研究相同。本文中，只有通过稀疏记忆回放增强且未经其他修改的模型，才称为 **REPLAY**。将按顺序学习一系列任务且未使用任何记忆回放的模型称为**SEQ**。

2.3 数据集

为提供可比的评估结果，我们采用由 MbPA++ 引入的相同任务增量式语言学习基准。其文本分类部分重新整理自 Zhang 等人（2015）使用的五个数据集，包含 4 个文本分类任务：新闻分类（AGNews，4 个类别）、本体预测（DBPedia，14 个类别）、情感分析（Amazon 和 Yelp，共享 5 个类别）以及主题分类（Yahoo，10 个类别）。遵循 d'Autume 等人（2019）及其他研究的做法，我们随机选取 115,000 个训练样本和 7,600 个测试样本，以构建一个平衡的数据集合。由于 Amazon 和 Yelp 都是情感分析数据集，因此将它们的标签合并，共有 33 个类别

。在所有实验中，我们评估模型在全部五个任务上的性能，并像以往工作一样报告宏平均准确率。

关于问答，这个基准包含 3 个数据集：SQuAD1.1（Rajpurkar 等人，2016年）、TriviaQA（Joshi 等人，2017年）和 QuAC（Choi 等人，2018年）。由于 TriviaQA 分为 网页和 维基百科两个部分，并将其视为两个不同的任务，因此该基准总共包含 4 个问答任务。

3 探究 BERT 抵抗遗忘的内在能力

如第 1 节所述，通过仅对 100 个实例进行记忆回放，模型便能迅速恢复其在先前学过的任务上的表现（d'Autume 等人，2019年）。如果模型完全丧失了编码先前任务的能力，那么仅通过 4 个更新步骤就能重新获得先前知识，这种情况便有悖直觉。我们推测，BERT 在学习新任务时实际上能够保留旧知识，而不会发生灾难性遗忘。为验证这一假设，我们首先开展了一项初步研究。

我们在文本分类基准上开展初步实验，采用 $\text{BERT}_{\text{BASE}}$和简单的线性解码器作为模型，并按照 4 种不同顺序对其进行训练（详见附录 A）。为考察 BERT 对特定任务的编码能力，我们遵循先前的探测研究（Tenney 等人，2019年；Jawahar 等人，2019年），在依次进行微调后冻结编码器参数，重新初始化五个新的线性探测解码器，并分别在五项任务上重新训练它们。我们发现，在相应任务上进行评估时，每个固定的 BERT 编码器与其新解码器组合后都能取得出色的性能。令人惊讶的是，4 种顺序下所有任务的宏平均准确率分别为 $75.87\%_{\pm 0.73\%}$、$76.76\%_{\pm 0.64\%}$、$75.19\%_{\pm 0.43\%}$、$76.76\%_{\pm 0.71\%}$，与多任务学习模型的性能（$78.89\%_{\pm 0.18\%}$）相近。然而，先前的研究（Biesialska 等，2020年）表明，顺序训练的模型会遭受灾难性遗忘，并且在适应新任务时会牺牲其在先前任务上的性能。与先前研究相反，我们的初步实验实际上表明，BERT 可能具备长期保持从先前任务中学到的知识的能力。

3.1 探测方法

为验证 BERT 在没有记忆回放帮助的情况下能否避免遗忘，我们需要一种工具，在模型逐步学习一系列任务时，系统地衡量其对先前任务的编码能力。一种方法是比较在两种不同设置（REPLAY 和 SEQ）下训练的模型在不同学习阶段的编码能力。对于每种设置，我们都将学习相应任务之前 的性能视为基线，它反映了 BERT 从预训练任务中获得的固有知识。然后，我们可以通过比较学习相应任务期间 和之后 的结果，考察 BERT 对旧知识的遗忘程度。因此，跟踪 BERT 的任务特定编码能力随时间的变化至关重要。我们提取编码器的参数，并在训练过程中按指定频率将其保存为检查点。在 REPLAY 和 SEQ 两种设置下，我们每训练 5,000 个训练样例就记录一次检查点[3]，不考虑检索记忆子集。

对于每个检查点，我们按以下步骤探测其对每项任务的编码能力 $\mathcal{T}_k$：

1. 将重新初始化的探测解码器添加到此检查点中的 BERT 参数中$_{\text{BASE}}$。2. 使用 $\mathcal{T}_k$ 的训练集 $\mathcal{D}_k^{tr}$ 中的所有数据训练重新组合后的模型，同时将 θ^{enc} **固定**，这意味着我们只调整探测解码器的参数。3. 在 $\mathcal{T}_k$的测试集上评估重新训练后的模型的分数[4]。

我们在评估前不改动编码器，而是在特定任务上重新训练一个兼容的探测解码器。对于文本分类，我们使用线性解码器作为探测网络；对于问答，则使用两个线性边界解码器，这与 MbPA++ (d' Autume 等人，2019) 和 Meta-MbPA（Wang 等人，2020b）的设置相同。需要指出的是，关于应该使用更简单还是更复杂的探测解码器，仍存在一些争议（Belinkov，2022）。出于两个原因，我们采用简单的单层探测网络。首先，较简单的探测器对重新训练后的模型性能影响较小（Liu 等人，2019a；Hewitt 和 Liang，

[3]由于每个批次包含 32 个实例，无法被 5,000 整除，因此我们在最接近预定节点的批次保存参数，以避免不必要的干扰。[4]文本分类使用**准确率**，抽取式问答使用**F1 分数**。

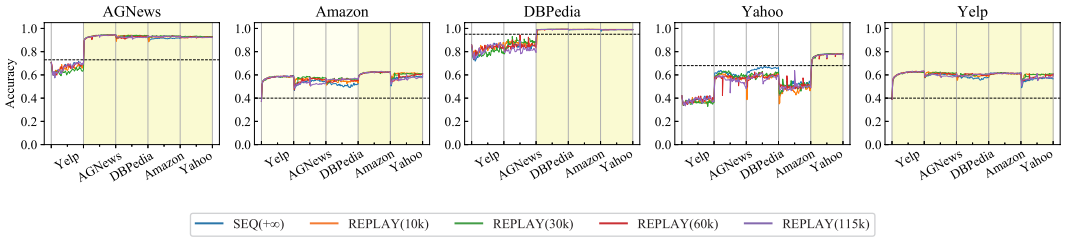


图1：按**顺序 1**训练的五项文本分类任务的探测结果，按数据集[6]分别展示。横坐标表示任务的训练顺序。

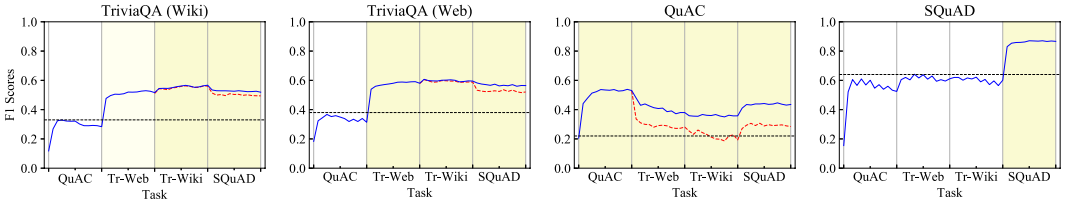


图2：探测　　按**顺序 1**训练的四项任务的 F1 分数，分别展　示[7]。

2019），这使我们能够专注于 BERT 的编码能力。其次，本文的目的并非比较 BERT 在不同任务上的编码能力，而是考察它是否会遗忘特定任务的知识。因此，最好采用与 d'Autume 等人（2019）和 Wang 等人（2020b）相同的单层解码器，以便获得可与他们的结果进行比较的结果。

3.2 重新思考灾难性遗忘

通过追踪各检查点之间的探测分数，我们现在能够定量衡量 BERT 模型在任务增量学习过程中，是否能够维持对先前任务的编码能力。探究重放间隔是否会影响 BERT 的编码能力也很重要。我们首先针对文本分类设置了一系列如下所述的实验。

为与先前工作（d'Autume 等人，2019；Wang 等人，2020b）进行比较，我们沿用与其一致的实验设置，将最大令牌长度和批量大小分别设为 128 和 32。我们采用 d'Autume 等人（2019）中 REPLAY 的训练设置作为基线：每从数据流中接收 10,000 个新样本，就从 $\mathcal{M}$ 中抽取 100 个样本进行回放。如第 2.2 节所述，我们将存储率 γ 和重放频率 r 都控制在 1%。为探究记忆回放的影响，我们比较了采用不同重放间隔训练的模型。每学习 N_{tr} 个样本后，我们从 $\mathcal{M}$ 中随机选取一个包含 $\lfloor 0.01 N_{tr} \rfloor$ 个样本的子集 $\mathcal{S}$。N_{tr} 设为 {10k，30k，60k，115k}；此外，在按顺序训练模型时，我们可以将 N_{tr} 视为 $+\infty$。我们采用 Adam（Kingma和Ba，2015）作为优化器。

我们使用第3.1节中的方法，评估 BERT 在每个检查点生成的表示的质量。如果 BERT 参数集合具有更强的特定任务编码能力，我们就能观察到更好的探测性能。在这里，对于文本分类，我们根据任务和训练顺序，在不同图中描绘准确率的变化。顺序 1 的结果（详见附录A）如图1所示，其余结果见附录B。通过比较模型学习特定任务前后的准确率，我们进一步获得了新的理解——

[6]最左侧的子图展示了模型在*AGNews* 训练集上的探测准确率分数如何随训练过程变化。接下来的四个子图分别对应*Amazon*、*DBPedia*、*Yahoo*和*Yelp*。由于模型正在相应任务上进行训练，我们将背景标为黄色。特别地，*Amazon* 和*Yelp* 共享相同的标签，因此当模型在另一个任务上训练时，我们将其背景标为浅黄色。[7]最左侧的是*TriviaQA*（*Wiki*），其后依次为*TriviaQA*（*Web*）、*QuAC*和*SQuAD*。重新训练探测解码器后的F1分数以蓝线表示。作为对比，由于模型开始学习新任务，我们用红色虚线绘制使用原始解码器的模型的F1分数。由于模型正在相应任务上进行训练，我们将背景标为黄色。特别地，*TriviaQA*（*Wiki*）和*TriviaQA*（*Web*）实际上是同一任务的子集，因此当模型学习另一个任务时，我们将它们的背景标为浅黄色。

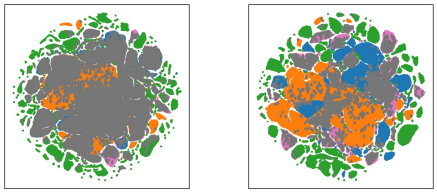
——关于任务增量语言学习：**尽管任务之间存在数据分布偏移，BERT 仍能保留对先前见过的任务的大部分分类能力，而不会发生灾难性遗忘**。这一结论同样适用于 SEQ，其重放频率被视为 $+\infty$。尽管在更大的重放间隔（例如 60k、115k、 $+\infty$）下，BERT 的表征能力会略有下降，但它仍能保持先前的知识，并通过稀疏回放迅速恢复。

我们还提供了问答任务上的实验结果。问答比文本分类更加复杂。为了考察 BERT 在问答任务中是否仍能保留旧知识，我们采用了比 d'Autume 等人（2019）更严格的实验设置。我们按照附录A中的 4 种不同顺序，在 SEQ**不进行任何记忆回放**的设置下，依次训练模型。对于每项任务，模型都微调 15K 步，是 d'Autume 等人（2019）的两倍。随后，我们使用探测方法评估每个 BERT 检查点的编码能力。**顺序 1** 的结果见图2，其他结果见附录 C。根据我们的实验设置，模型经过了足够多步的微调，以至于在每项任务上都过拟合了。然而，探测结果（蓝线）仍远高于重新训练解码器之前测得的原始分数（红色虚线）。比较二者之间明显的差距[8]，，我们可以发现，BERT 在学习新任务时仍保留了先前任务的大部分知识。

此外，我们还研究了其他预训练语言模型保留旧任务知识的能力，详见附录 D。总的来说，尽管我们研究的模型采用了不同的注意力机制，规模也各不相同，但所有这些预训练语言模型在学习一系列任务时，都具有内在的避免遗忘的能力。在不同的训练顺序下，即使学习了 5 个任务，它们仍能保留对第一个学到的任务进行编码的能力。

4 遗忘的新视角

从第 3.2 节的实验中，我们观察到，一旦 BERT 学会某项任务，就有潜力长期为该任务提供高质量的表示。因此，如果我们试图恢复模型处理先前任务的能力，似乎只需微调解码器。然而另一方面，SEQ 模型在已学任务上的性能会严重下降，这种现象称为灾难性遗忘。为了解释这一矛盾，我们使用 t-SNE 工具包（vander Maaten 和 Hinton，2008），将通过 SEQ 或 REPLAY 在所有任务上训练后的表示可视化（图 3）。在按顺序学习时，结果显示模型的表征——



(a) SEQ　　(b) REPLAY

图 3：按 **Order1** 的顺序在各任务上训练后，表示空间的可视化结果。AGNews、Amazon 和 Yelp、DBPedia、Yahoo 的数据点分别以蓝色、橙色、绿色和粉色标示；多个任务的交集区域以灰色标示。

会在重叠空间中产生不同任务的表示。在这种情况下，任务解码器会将所有向量识别为来自新任务的实例，从而造成混淆；但稀疏回放可以有效避免这一问题。

所有这些观察促使我们作出如下假设：在任务增量学习中，遗忘可分为两部分，即任务内遗忘和任务间遗忘。任务内遗忘描述模型在学习新任务后，是否仍能为先前任务生成有意义的表示；而任务间遗忘则指为不同任务生成的表示是否彼此可区分。本节首先提出一种用于描述表示的工具（见第 4.1 节）。随后，我们展示模型在 REPLAY 设置下持续学习时发生的变化，并为自然语言处理模型中的灾难性遗忘提供一种新的理解。诚然，问答模型通常涉及不同粒度表示之间的交互（从词元级到文档级）（Wang 等人，2018），因此分析起来更具挑战性。所以，我们将更着重分析文本分类的结果。

[8]在问答中，由于所有问答任务共享同一个答案边界解码器，学习新任务时，先前任务的 F1 分数不会降至零。但不同的文本分类任务会使用解码器中的不同维度，因此旧任务的分数会急剧下降。

4.1 表示子空间的定义

正如 Gao 等人（2019）和 Wang 等人（2020a）所述，使用单层线性解码器训练时，预训练语言模型会生成位于窄锥中的词元级嵌入向量。我们发现，这一结论不仅适用于词元级表示，也适用于句子级表示（更多细节见附录 E）。同一类别的表示向量会聚集在一起，因此我们可以用一个凸锥覆盖这些向量，其顶点为原点。为了精确描述这些向量，该锥体应覆盖所有向量，且尽可能窄。形式化地，我们将包围这些向量的锥体记作：

$$\left\{ \boldsymbol{x} \in \mathbb{R}^d \middle| \frac{\boldsymbol{x}^{\mathrm{T}} \boldsymbol{c}}{\|\boldsymbol{x}\|_2 \cdot \|\boldsymbol{c}\|_2} \geq \delta \right\} \tag{1}$$

其中，$\boldsymbol{c} \in \mathbb{R}^d$ 是锥体的中心轴，δ 控制视场角。

为了获得包含 BERT 输出的所有向量的最窄锥体，假设向量集为 $\mathcal{V} = \{\boldsymbol{v}_i\}_{i=1}^n$，我们求解如下所述的优化目标：

算法 1： 计算表示 Cone

输入： 向量集 $\mathcal{V}$，输入规模 $n = |\mathcal{V}|$，初始中心轴 $\boldsymbol{c}_0$，学习率 α，终止条件 ε
输出： 锥体的中心轴 $\boldsymbol{c}$
当 $|\mathcal{V}| > \lceil 0.95n \rceil$ **时执行**
  初始化 $\boldsymbol{c} = \boldsymbol{c}_0$
  重复
    根据公式 3 计算优化目标。
    求得梯度 $\nabla f_\delta(\boldsymbol{c}, \mathcal{V})$。
    $\boldsymbol{c} \leftarrow \boldsymbol{c} + \alpha \nabla f_\delta(\boldsymbol{c}, \mathcal{V})$
    $\boldsymbol{c} \leftarrow \boldsymbol{c} / \|\boldsymbol{c}\|_2$
    通过线性搜索调整 α。
  直到 $\forall c_j$ 在 $\boldsymbol{c}$，$\Delta c_j < \varepsilon$
  计算 $\boldsymbol{v}_i$ 和 $\boldsymbol{c}$的余弦，记为 $\{s_i\}_{i=1}^{|\mathcal{V}|}$。对 $\{s_i\}_{i=1}^{|\mathcal{V}|}$排序。
  $m \leftarrow \lceil (|\mathcal{V}| - \lceil 0.95n \rceil)/2 \rceil$
  选取 m 最低的 s_i 及其相关向量 $\mathcal{V}^{\mathrm{del}}$。
  $\mathcal{V} \leftarrow \mathcal{V} - \mathcal{V}^{\mathrm{del}}$
  $\boldsymbol{c}_0 \leftarrow \boldsymbol{c}$
结束循环

$$\underset{\boldsymbol{c},\, \delta}{\text{minimize}} \ -\delta; \ \ \text{s.t.} \ \forall \boldsymbol{v}_i \in \mathcal{V}, \frac{\boldsymbol{v}_i^{\mathrm{T}} \boldsymbol{c}}{\|\boldsymbol{v}_i\|_2} \geq \delta, \quad \|\boldsymbol{c}\|_2 = 1, \tag{2}$$

其中，$\|\cdot\|_2$ 表示 L2 范数。为获得确定的解，我们添加约束 $\|\boldsymbol{c}\|_2 = 1$，否则该方程仅表示 $\boldsymbol{c}$ 的方向，而不包含长度。表示向量聚集在一起，因此我们可以得到一个视场角很小的锥体（$\delta \gg 0$）。因此，式（2）是一个凸优化目标，可以通过序列最小二乘规划（Kraft, 1988；Boggs 和 Tolle，1995）求解。在迭代过程中，我们通过以下表达式获得优化梯度：

$$\begin{aligned} f_\delta(\boldsymbol{c}, \{\boldsymbol{v}_i\}_{i=1}^n) &= \max_i \left\{ \frac{\boldsymbol{v}_i^{\mathrm{T}} \boldsymbol{c}}{\|\boldsymbol{v}_i\|_2} \right\} \\ \nabla f_\delta(\boldsymbol{c}, \{\boldsymbol{v}_i\}_{i=1}^n) &= \frac{\boldsymbol{v}}{\|\boldsymbol{v}\|_2}, \ \ \boldsymbol{v} = \arg\max_{\boldsymbol{v}_i} \left\{ \frac{\boldsymbol{v}_i^{\mathrm{T}} \boldsymbol{c}}{\|\boldsymbol{v}_i\|_2} \right\} \end{aligned} \tag{3}$$

此外，为了减少噪声标注导致的异常值干扰，我们将约束条件修改为：锥体只需覆盖不少于 95% 的训练样例。由于这会破坏原目标函数的凸性，我们采用迭代方法求得一个近似解，并确保每一步计算都保持凸性。算法 1 概述了详细的求解过程。显然，锥轴应位于向量的中心，因此我们将 $\boldsymbol{c}_0 = \sum_i \boldsymbol{v}_i / \|\sum_i \boldsymbol{v}_i\|_2$ 初始化为该位置。

4.2 任务内遗忘

根据第 3.2 节的结果，我们发现 BERT 能够长期保留先前学到的知识。使用重新训练的解码器时，BERT 在先前任务上仍能表现良好，这表明 BERT 很少遭受任务内遗忘。为探究阻止 BERT 发生任务内遗忘的机制，我们以 AGNews 和 Amazon 为例训练一个 BERT 模型[9]，分析 BERT 表示空间内部的变化。我们首先使用 AGNews 的所有实例训练模型，然后从 Amazon 中抽取 30K 个实例，作为任务增量学习的第二个任务。与图1类似，BERT 仍能生成高质量表征

[9]按字典序选择。

在学习 Amazon 后，即使没有使用情景回放，BERT 仍能为 AGNews 生成高质量表征。我们推测，学习新任务后，旧任务的表示子空间仍然保持拓扑有序[10]。

如图 3(a) 所示，我们可以看到，如果不使用情景回放，旧任务实例的表示向量会旋转到新任务的重叠子空间中，导致解码器无法判断输入实例应属于哪个任务。另一方面，如果采用任务专用解码器（例如探测解码器），就能有效判定给定实例的类别。这或许意味着，同一旧任务类别的向量彼此仍然相距不远，但与来自同一旧任务其他类别的向量相距较远。因此，我们猜想，如果两个表示向量经过训练后处于相邻位置，那么在学习新任务后，它们仍会彼此相邻。

为了检验旧任务表示向量的旋转过程是否具有拓扑顺序，我们首先需要一种度量指标来定义同一类别实例的表示之间的相对位置。按照我们在第 4.1 节中的方法，可以将类别 y 的表示子空间描述为一个凸锥，其锥轴为 $\boldsymbol{c}_y$。于是，对于类别 y的实例 i，我们可以将其表示向量 $\boldsymbol{v}_{y的\,i}$ 相对位置定义为 $\boldsymbol{v}_{y与\,i}$ 之间的余弦值。

由于我们需要比较每个实例在两个检查点（学习第二个任务之前和之后）的相对位置，因此根据上标区分不同检查点处的向量。形式化地说，我们将学习 Amazon之前的锥轴和表示向量分别记为 $\boldsymbol{c}_y^{(0)}$和 $\boldsymbol{v}_{y\ i}^{(0)}$，将学习 Amazon之后的对应向量分别记为 $\boldsymbol{c}_y^{(1)}$和 $\boldsymbol{v}_{y\ i}^{(1)}$。

对于 $\mathcal{V}_y^{(0)}$（学习 Amazon 之前类别 y 的通用表示集）中的每个 $\boldsymbol{v}_{y\ i}^{(0)}$，我们根据欧几里得距离从 $\mathcal{V}_y^{(0)} - \left\{\boldsymbol{v}_{y\ i}^{(0)}\right\}$ 中选取其 n 个最近邻，并将其指示变量集合记为 $N_{y,\ i}$。可以合理地认为，这些 n 个邻居与 $\boldsymbol{v}_{y\ i}^{(0)}$ 具有最相似的语义信息。然后，我们可以检查 $\boldsymbol{v}_{y\ i}^{(1)}$ 以及向量 $\left\{\boldsymbol{v}_{y\ k}^{(1)}\right\}_{k\in N_{y,\ i}}$是否仍是邻居，以验证类别 y 的表示子空间是否拓扑有序。在这里，我们计算 $\boldsymbol{v}_{y\ i}^{(1)}$ 与 $\left\{\boldsymbol{v}_{y\ k}^{(1)}\right\}_{k\in N_{y,\ i}}$相对位置之间的相关性，该相关性估计为

通过 $\cos(\boldsymbol{c}_y^{(1)},\ \boldsymbol{v}_{y\ i}^{(1)})$ 与 $\sum_{k\in N_{y,\,i}}\cos(\boldsymbol{c}_y^{(1)},\ \boldsymbol{v}_{y\ k}^{(1)})$ 之间的皮尔逊相关系数进行估计。我们在表 1 中列出了 AGNews 中所有类别在不同 n 尺度下的结果（其中 $y\in${类别-1,类别-2, 类别-3,类别-4}, $n\in\{5,\,10,\,25,\,50,\,100\}$）。比较不同的 n后可以发现，中等规模的邻居能够带来更好的相关性，从而抑制小样本集带来的随机性，以及大样本集带来的无关偏差。总的来说，n 的影响微不足道，因此我们可以得出结论：学习新任务后，$\boldsymbol{v}_{y\ i}^{(0)}$,及其邻居的位置仍然接近，因为皮尔逊系数不低于 0.483（部分高于 0.723）。

换言之，如果两个样本在学习新任务之前被映射到相近的位置，那么在学习新任务之后它们仍会彼此接近。BERT 学会一项任务后，往往会将同一类别的表示映射到相近的位置，同时将不同类别的表示映射到彼此不相邻的空间。因此，如果旧任务表示的旋转过程能够保持拓扑有序，那么某一类别的表示向量将始终与其他类别的向量分离。这就是在我们的研究中，BERT 展现出缓解任务内遗忘能力的原因。

4.3 任务间遗忘

神经网络模型在连续训练多个不同任务时，总会遭受灾难性遗忘；本文将其归因于任务间遗忘。与先前的评估类似，我们继续使用覆盖锥来探究记忆回放在帮助模型抵抗任务间遗忘中的作用。

[10]给定一个非空向量集 $\mathcal{V}$，我们可以根据向量之间的距离将其聚类为多个不相交子集，$\mathcal{V}^1,\ \cdots,\ \mathcal{V}^K$。学习新任务后，先前任务的表示向量会旋转到新的方向。对于任意子集 $\mathcal{V}^p$ 和其中的任意新向量 v_x^p $\mathcal{V}^p$，如果任意新向量 $v_y^p\in\mathcal{V}^p$ 比其他子集中的任意向量 $v_z^q\in\mathcal{V}^q\,(q\neq p)$ 都更接近 v_x^p，我们就认为学习新任务时表示向量的旋转过程是完全拓扑有序的。

表 1：$\boldsymbol{v}_{1,\ i}$ 及其 n 个邻居的角度与锥轴之间的皮尔逊相关系数（×100）。最高分以**粗体**标示，次高分加下划线。

n	类别 1	类别 2	类别 3	类别 4
5	81.09$_{\pm 3.55}$	48.35$_{\pm 9.82}$	83.11$_{\pm 3.57}$	72.41$_{\pm 3.53}$
10	**81.68**$_{\pm 3.26}$	50.44$_{\pm 10.29}$	**83.90**$_{\pm 3.52}$	~~73.80~~$_{\pm 3.22}$
25	~~81.10~~$_{\pm 3.19}$	**51.46**$_{\pm 10.27}$	~~83.76~~$_{\pm 3.58}$	**73.98**$_{\pm 3.11}$
50	80.03$_{\pm 3.30}$	~~51.06~~$_{\pm 10.56}$	83.25$_{\pm 3.65}$	73.39$_{\pm 3.12}$
100	78.51$_{\pm 3.49}$	50.16$_{\pm 10.58}$	83.27$_{\pm 3.84}$	72.35$_{\pm 3.12}$

当模型通过与 Softmax 相连的线性层对表示向量 $\boldsymbol{v}$进行解码时，解码器可以被视为一组列向量（即第2.1节中的 $\{\boldsymbol{w}_y\}_{y\in\mathcal{Y}}$），而预测过程等价于选择与 $\boldsymbol{v}$的内积最大的列向量。因此，有必要检查先前任务的锥是否旋转到解码器中对应的列向量。在本节中，我们仍首先考察在 AGNews 上训练、随后以 30K 的重放间隔在 Amazon 上连续训练三次的模型。

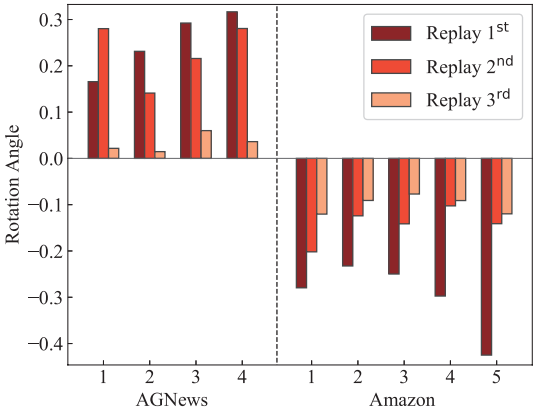


图4：回放期间旋转角度的柱状图，按任务标签聚类，并根据回放时间着色。

我们观察到，记忆回放前后解码器中的列向量没有显著变化，因为它们的旋转角度小于 1×10^{-3}，可以忽略不计。对于每个时间 t，我们将回放前后类别 k的锥轴分别记为 $\boldsymbol{c}_{t\ ,\ k}^{-}$ 和$\boldsymbol{c}_{t\ k,}^{+}$，分别为-

同时，解码器中与之对应的列向量为 $\boldsymbol{w}_k$。然后，第 k 个锥体的旋转角度可估算为：$\Delta\zeta_{t,\ k} = \cos(\boldsymbol{c}_{t\ k,}^{-},\ \boldsymbol{w}_k) - \cos(\boldsymbol{c}_{t\ k,}^{+},\ \boldsymbol{w}_k)$。如果 $\Delta\zeta_{t,\ k} > 0$，则表示锥在回放过程中向 $\boldsymbol{w}_k$ 的方向旋转得更近。图4所示的结果表明，记忆回放促使先前任务的向量高效地旋转至解码器中对应的列向量，同时使当前任务的向量偏离最优位置。此外，随着回放次数的增加，这一双重过程会逐渐减弱。由于 BERT 的表示空间维度很高，而我们的任务数量有限，在记忆和当前任务上交替学习可以通过将编码向量映射到不同的子空间，使它们彼此分离。

在附录F中，我们提供了更多可视化结果，展示记忆回放如何减少任务间遗忘，也就是传统意义上的灾难性遗忘。

5 结论

在这项工作中，我们开展了一项探测研究，以定量衡量预训练语言模型在任务增量学习场景下对先前学习任务的编码能力。与以往研究不同，我们发现，在学习一系列任务时，即使没有经验回放，BERT 也能长期保留利用先前任务中所学知识的编码能力。我们进一步考察了 BERT 在任务增量学习过程中产生的、每个任务中不同类别的表示子空间的拓扑结构。我们发现，没有记忆回放时，先前任务的表示子空间往往会与当前任务的表示子空间重叠；但同一任务中不同类别的子空间彼此可区分，在一定程度上呈现出拓扑不变性。我们的发现有助于更好地理解这一新发现与以往关于灾难性遗忘的研究之间的联系。

受任务数量的限制，我们尚未讨论 BERT 在持续学习更多任务时的容量。据我们所知，目前还没有现成的方法可以衡量模型是否已达到其学习容量，因而无法再记住更多知识。未来，我们将把探测方法扩展到更长的序列或不同类型的任务，并探究大型预训练语言模型能够保留多少知识。

致谢

本研究得到国家重点研发计划（编号：2020年AAA0106600）和国家自然科学基金委员会项目（编号：62161160339）的资助。

参考文献

Yonatan Belinkov。探测分类器：前景、缺陷与进展。计算语言学，48(1):207–219，2022年3月。doi：10.1162/coli a 00422。网址：https:// aclanthology.org/2022.cl-1.7。

Magdalena Biesialska、Katarzyna Biesialska 和 Marta R Costa-juss`a。自然语言处理中的持续终身学习：一项综述。载于第28届国际计算语言学大会论文集，第6523–6541页，2020年。

Paul T. Boggs 和 Jon W. Tolle。序列二次规划。Acta Numerica，4:4–12，199 5.

Zhiyuan Chen 和 Bing Liu。终身监督学习。载于 Ronald J. Brachman 和 Peter Stone（编），《终身机器学习》，第 35–54 页。Morgan & Claypool Publishers，第 2 版，2018年。

Eunsol Choi、He He、Mohit Iyyer、Mark Yatskar、Wen-tau Yih、Yejin Choi、Percy Liang 和 Luke Zettlemoyer。QuAC：上下文中的问答。载于《2018年自然语言处理实证方法会议论文集》，第 2174–2184 页，比利时布鲁塞尔，2018年 10 月。计算语言学协会。doi：10.18653/v1/D18-1241。URL https://aclanthology.org/D18-1241。

Kevin Clark、Minh-Thang Luong、Quoc V. Le 和 Christopher D. Manning。ELECTRA：将文本编码器预训练为判别器而非生成器。载于《国际学习表征会议》，2020年。URL https://openreview.net/forum?id=r1xMH1BtvB。

Cyprien de Masson d’Autume、Sebastian Ruder、Lingpeng Kong 和 Dani Yogatama。终身语言学习中的情景记忆。载于《第33届神经信息处理系统国际会议论文集》，第13132–13141页，2019年。

Jacob Devlin、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。BERT：面向语言理解的深度双向变换器预训练。载于《2019年计算语言学协会北美分会会议：人类语言技术论文集》第1卷（长篇与短篇论文），第4171–4186页，2019年。

Robert M. French。联结主义网络中的灾难性遗忘。《认知科学趋势》，3：128–135，1999年。

Jun Gao、Di He、Xu Tan、Tao Qin、Liwei Wang 和 Tieyan Liu。自然语言生成模型训练中的表示退化问题。载于《国际学习表征会议论文集》，2019年。

Binzong Geng、Min Yang、FajieYuan、Shupeng Wang、Xiang Ao 和RuifengXu。通过不确定性正则化进行迭代网络剪枝，以实现终身情感分类。载于第44届国际 ACM SIGIR 信息检索研究与发展会议论文集（SIGIR ’21），第1229–1238页，纽约州纽约，美国，2021年。计算机协会。

John Hewitt 和 Percy Liang。使用控制任务设计和解读探测器。载于2019年自然语言处理实证方法会议与第9届自然语言处理国际联合会议（EMNLP-IJCNLP）论文集，第2733–2743页，中国香港，2019年11月。计算语言学协会。doi：10.18653/v1/D19-1275。URL https://aclanthology.org/D19-1275。

Ganesh Jawahar、Benoˆıt Sagot 和 Djam´e Seddah。BERT 对语言结构有何认识？载于《第57届计算语言学协会年会论文集》，第3651–3657页，意大利佛罗伦萨，2019年7月。计算语言学协会。doi：10.18653/v1/P19-1356。URL：https://aclanthology.org/P19-1356。

Xisen Jin、Dejiao Zhang、Henghui Zhu、Wei Xiao、Shang-Wen Li、Xiaokai Wei、Andrew Arnold 和 Xiang Ren。终身预训练：持续使语言模型适应新兴语料库。载于《2022年计算语言学协会北美分会人类语言技术会议论文集》，第4764–4780页，美国西雅图，2022年7月。计算语言学协会。doi：10.18653/v1/2022.naacl-main.351。URL：https://aclanthology.org/2022.naacl-main.351。

Mandar Joshi、Eunsol Choi、Daniel Weld 和 Luke Zettlemoyer。TriviaQA：用于阅读理解的大规模远程监督挑战数据集。载于第55届计算语言学协会年会论文集（第1卷：长篇论文），第1601–1611页，加拿大温哥华，2017年7月。计算语言学协会。doi：10.18653/v1/P17-1147。网址：https://aclanthology.org/P17-1147。

Diederik P. Kingma 和 Jimmy Ba。Adam：一种用于随机优化的方法。载于第3届国际学习表征会议（ICLR 2015）会议论文集，2015年。

James Kirkpatrick、Razvan Pascanu、Neil Rabinowitz、Joel Veness、Guillaume Desjardins、Andrei A Rusu、Kieran Milan、JohnQuan、Tiago Ramalho、Agnieszka Grabska-Barwinska 等。克服神经网络中的灾难性遗忘。《美国国家科学院院刊》，114(13)：3521–3526，2017年。

D. Kraft。一个用于序列二次规划的软件包。技术报告，德国航空航天中心–飞行动力学研究所，德国科隆，1988年。

Mike Lewis、Yinhan Liu、Naman Goyal、Marjan Ghazvininejad、Abdelrahman Mohamed、Omer Levy、Veselin Stoyanov 和 Luke Zettlemoyer。BART：用于自然语言生成、机器翻译和理解的去噪序列到序列预训练。载于第58届计算语言学协会年会论文集，第7871–7880页，线上，2020年7月。计算语言学协会。网址：https://aclanthology.org/2020.acl-main.703。

Nelson F. Liu、Matt Gardner、Yonatan Belinkov、Matthew E. Peters 和 Noah A. Smith。上下文表征的语言知识与可迁移性。载于2019年计算语言学协会北美分会人类语言技术会议论文集：第1卷（长篇与短篇论文），第1073–1094页，明尼阿波利斯，明尼苏达州，2019a年6月。计算语言学协会。doi：10.18653/v1/N19-1112。网址：https://aclanthology.org/N19-1112。

Yinhan Liu、Myle Ott、Naman Goyal、Jingfei Du、Mandar Joshi、Danqi Chen、Omer Levy、Lewis Mike、Luke Zettlemoyer 和 Veselin Stoyanov。RoBERTa：一种稳健优化的 BERT 预训练方法。CoRR，abs/1907.11692，2019b。网址：http://arxiv.org/abs/1907.11692。

David Lopez-Paz 和 Marc'Aurelio Ranzato。用于持续学习的梯度情景记忆。《神经信息处理系统进展》，30：6467–6476，2017年。

Arun Mallya 和 Svetlana Lazebnik。PackNet：通过迭代剪枝将多个任务添加到单个网络中。载于《2018年IEEE/CVF计算机视觉与模式识别会议》，第 7765–7773 页，2018年。

Michael McCloskey 和 Neal J. Cohen。联结主义网络中的灾难性干扰：序列学习问题。《学习与动机心理学》，24(C)：109–165，1989年。

Sanket Vaibhav Mehta、Darshan Patil、Sarath Chandar 和 Emma Strubell。关于预训练在终身学习中作用的实证研究。CoRR，abs/2112.09153，2021年。网址：https://arxiv.org/abs/2112.09153。

Jonas Pfeiffer、Andreas R¨uckl´e、Clifton Poth、Aishwarya Kamath、Ivan Vuli´c、Sebastian Ruder、Kyunghyun Cho 和 Iryna Gurevych。AdapterHub：用于适配 Transformer 的框架。arXiv 预印本，2020年。

Yujia Qin、Jiajie Zhang、Yankai Lin、Zhiyuan Liu、Peng Li、Maosong Sun 和 Jie Zhou。ELLE：面向新兴数据的高效终身预训练。载于《计算语言学协会研究成果：ACL 2022》，第 2789–2810 页，爱尔兰都柏林，2022年5月。计算语言学协会。doi：10.18653/v1/2022.findings-acl.220。URL：https://aclanthology.org/2022.findings-acl.220。

Alec Radford、Jeff Wu、Rewon Child、David Luan、Dario Amodei 和 Ilya Sutskever。语言模型是无监督的多任务学习者。2019年。Pranav Rajpurkar、Jian Zhang、Konstantin Lopyrev 和 Percy Liang。SQuAD：用于文本机器阅读理解的 100,000+ 个问题。载于《2016年自然语言处理实证方法会议论文集》，第 2383–2392 页，奥斯汀，得克萨斯州，2016年11月。计算语言学协会。doi：10.18653/v1/D16-1264。URL https://aclanthology. org/D16-1264。Sylvestre-Alvise Rebuffi、Hakan Bilen 和 Andrea Vedaldi。使用残差适配器学习多个视觉领域。载于《第31届神经信息处理系统国际会议论文集》，NIPS'17，第 506–516 页，长滩，加利福尼亚州，美国，2017年。Mark Bishop Ring。《强化环境中的持续学习》。博士论文，德克萨斯大学奥斯汀分校，1994年。M. T. Rosenstein、Z. Marx、L. P. Kaelbling 和 T. G. Dietterich。迁移还是不迁移。载于《第5届神经信息处理系统国际会议论文集》，2005年。Fan-Keng Sun、Cheng-Hao Ho 和 Hung-Yi Lee。LAMAL：语言建模就是终身语言学习所需的一切。载于《国际学习表征会议》，2020年。Ian Tenney、Patrick Xia、Berlin Chen、Alex Wang、Adam Poliak、R. Thomas McCoy、Najoung Kim、Benjamin Van Durme、Sam Bowman、Dipanjan Das 和 Ellie Pavlick。你从上下文中学到了什么？探究上下文化词表示中的句子结构。载于《国际学习表征会议》，2019年。URL https://openreview.net/forum? id=SJzSgnRcKX。Sebastian Thrun。终身学习算法。载于 S. Thrun 和 L. Pratt（编），《学习如何学习》，第 181–209 页。克吕维尔学术出版社，1998年5月。Iulia Turc、Ming-Wei Chang、Kenton Lee 和 Kristina Toutanova。读得好的学生学得更好：学生模型初始化对知识蒸馏的影响。CoRR，abs/1908.08962，2019年。URL http://arxiv.org/abs/1908.08962。Laurens van der Maaten 和 Geoffrey Hinton。使用 t-SNE 可视化数据。《机器学习研究期刊》，9(11)，2008年。Lingxiao Wang、Jing Huang、Kevin Huang、Ziniu Hu、Guangtao Wang 和 Quanquan Gu。通过频谱控制改进神经语言生成。载于《国际学习表征会议》，2020a。Wei Wang、Ming Yan 和 Chen Wu。用于阅读理解和问答的多粒度层次注意力融合网络。arXiv 预印本，arXiv:1811.11934，2018年。Zirui Wang、Sanket Vaibhav Mehta、Barnabas Poczos 和 Jaime G Carbonell。利用有限记忆进行高效的元终身学习。载于《2020年自然语言处理实证方法会议论文集》（EMNLP），第 535–548 页，2020b。Tongtong Wu、Massimo Caccia、Zhuang Li、Yuan-Fang Li、Guilin Qi 和 Gholamreza Haffari。持续学习中的预训练语言模型：一项比较研究。载于《国际学习表征会议》，2022年。URL https://openreview.net/forum? id=figzpGMrdD。Zhilin Yang、Zihang Dai、Yiming Yang、Jaime Carbonell、Ruslan Salakhutdinov 和 Quoc V. Le。XLNet：用于语言理解的广义自回归预训练。Curran Associates Inc.，雷德胡克，纽约州，美国，2019年。

Friedemann Zenke、Ben Poole 和 SuryaGanguli。通过突触智能实现持续学习。《机器学习研究论文集》，70：3987–3995，2017年。

Xiang Zhang、Junbo Zhao 和 Yann LeCun。用于文本分类的字符级卷积网络。载于《神经信息处理系统进展》第 28 卷。Curran Associates, Inc.，2015年。

A 数据集与顺序

For 在任务增量文本分类中，我们使用以下顺序来训练模型，这些顺序 e
the 与 d'Autume 等人（2019）和 Wang 等人（2020b）相同:

1. Yelp→AGNews→DBPedia→Amazon→Yahoo。2. DBPedia→Yahoo→AGNews→Amazon→Yelp。3. Yelp →Yahoo→Amazon→DBpedia→AGNews。4. AG News→Yelp→Amazon→Yahoo→DBpedia。

对于任务增量问答，我们采用以下顺序训练模型；这些顺序也与 d'Autume 等人（2019）和 Wang 等人（2020b）采用的顺序相同:

1. QuAC→TriviaQA（Web）→TriviaQA（Wiki）→SQuAD。
2. SQuAD→TriviaQA（Wiki）→QuAC→TriviaQA（Web）。
3. TriviaQA（Web）→TriviaQA（Wiki）→SQuAD→QuAC。
4. TriviaQA（Wiki）→QuAC→TriviaQA（Web）→SQuAD。

在这些顺序中，TriviaQA（Joshi 等人，2017年）的网页 部分和维基百科 部分被视为两个独立的数据集。

B 所有顺序的探测准确率：文本分类

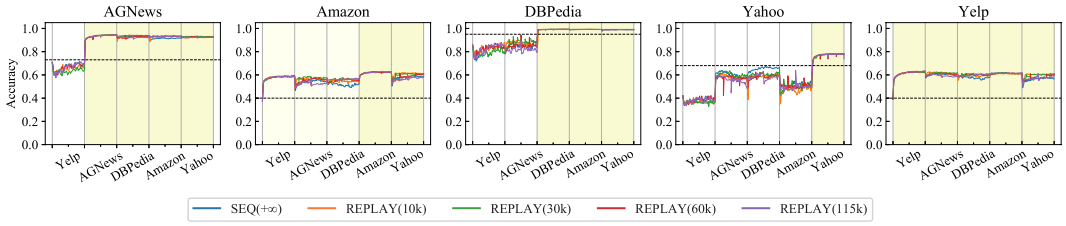
本节通过图 5 展示全部四种顺序的探测结果。与正文一致，在对应任务的训练过程中及训练完成后，背景均以黄色标示。特别是，由于 Amazon 和 Yelp 共享相同的标签，当模型在其中一个数据集上训练后，另一个数据集的背景便以浅黄色标示。

C 问答任务分析

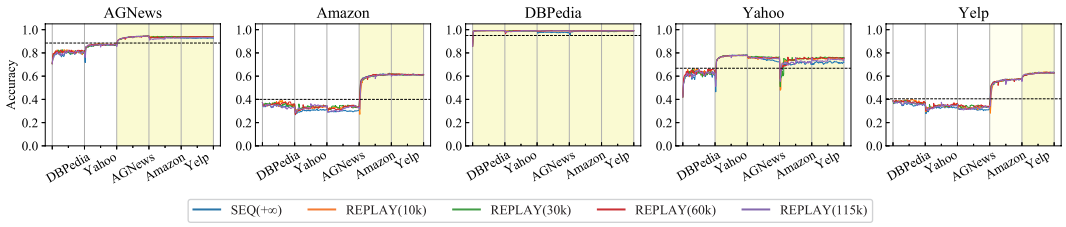
与文本分类分析类似，我们也按照指定顺序在 4 个问答（QA）任务上训练模型。为验证 BERT 是否能在问答任务中长期保留知识，我们从每个任务中随机采样 240K 个示例（通过重复采样），其数据量是原始数据集的两到三倍。我们将批量大小设为 16，学习率设为 3×10^{-5}，且不进行衰减。此外，我们不 使用任何记忆模块，这意味着模型在不进行记忆回放的情况下按顺序训练。

我们每隔 1,250 步保存一次检查点，然后分别在 4 个任务上重新微调解码器，同时固定 BERT 编码器的参数。由于这里将问答建模为序列到序列任务，因此一个问题可能有多个标准答案片段。因此，我们使用 F1分数评估模型的性能。所有结果均见图6。

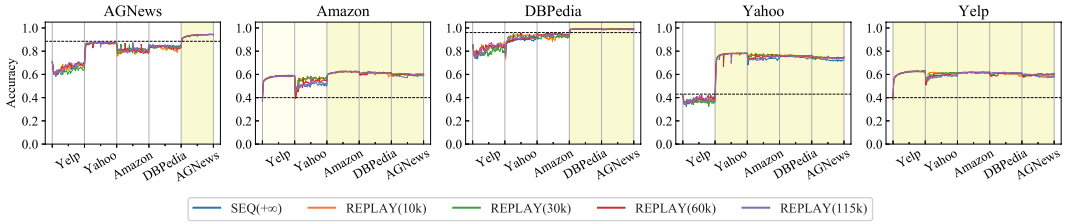
结果表明，$\text{BERT}_{\text{BASE}}$ 在问答这类更复杂的任务中，仍能长期保留先前学到的知识。在问答任务中，模型为所有 4 项任务使用统一的片段位置解码器。因此，与文本分类不同，先前任务原始的 F1 分数（重新微调前，红色虚线）不会降至零。尽管问答任务中的灾难性遗忘问题并不十分严重，但考虑到蓝线与红色虚线之间的差距，模型在重新微调解码器后仍能取得高得多的 F1 分数。同时，我们发现模型完成相应任务的学习后，蓝线仅略有下降。这意味着即使**完全不进行**记忆回放，BERT 仍具备令人满意的保留先前知识的能力。第5节中的结论也适用于问答任务。



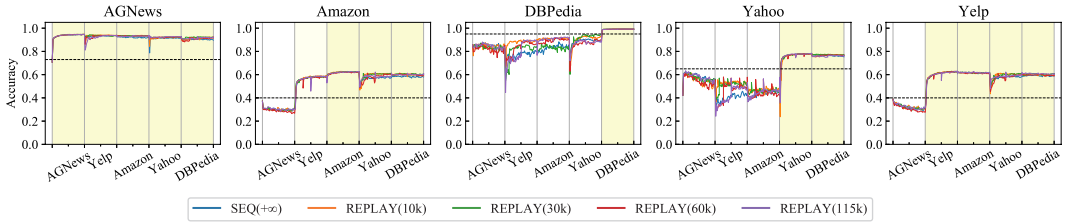
(a) **顺序 1**的结果。



(b) **顺序 2**的结果。

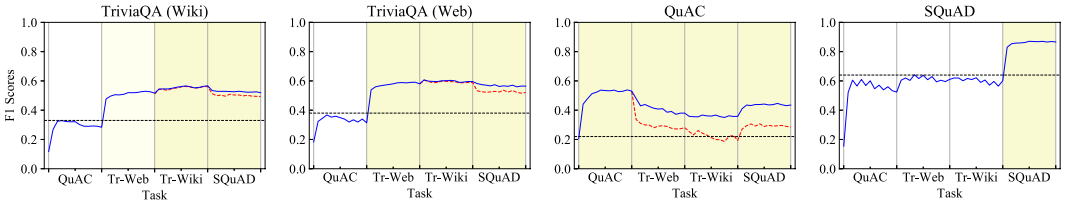


(c) **顺序 3**的结果。

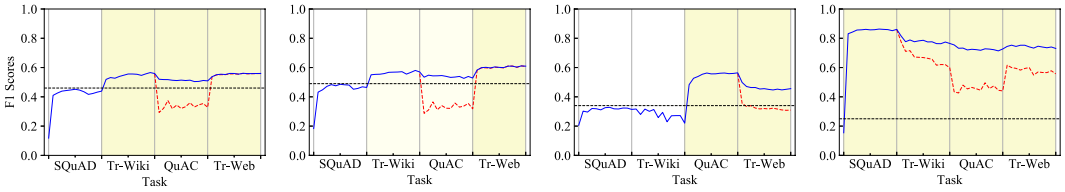


(d) **顺序 4**的结果。

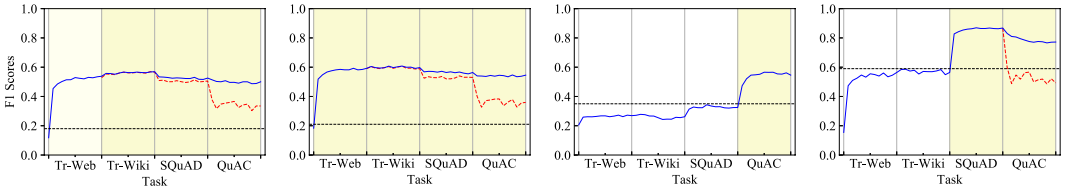
图 5：按各顺序训练五个文本分类任务的探测结果。每一行分别展示 5 个任务的结果，从左至右依次为 AGNews、Amazon、DBPedia、Yahoo 和 Yelp。



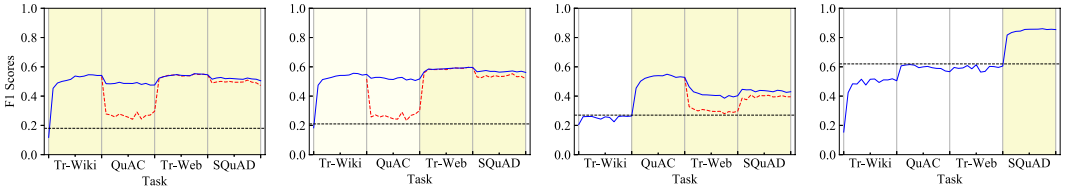
(a) 顺序 1 的结果。



(a) 顺序 2 的结果。



(b) 顺序 3 的结果。



(c) 顺序 4 的结果。

图6：按 4 种不同顺序训练的 4 个任务的 F1 分数。每一行分别展示 4 个任务的结果，最左侧为 TriviaQA（Wiki），接下来依次是 TriviaQA（Web）、QuAC 和 SQuAD。重新微调解码器后的 F1 分数以蓝线表示；作为对比，重新微调解码器前的 F1 分数以红色虚线表示。由于模型是在相应任务上训练的，因此我们将背景标为黄色。特别是，TriviaQA（Wiki）和 TriviaQA（Web）实际上是同一任务的子集，因此当模型在另一个任务上训练时，我们将它们的背景标为浅黄色。

表2：各种预训练语言模型的探测结果

X	PLM	AGNews	Amazon	DBPedia	Yahoo	Yelp
上层	BERT-tiny	92.46	55.91	98.70	71.67	57.45
	BERT-mini	93.71	58.50	99.01	72.62	60.24
	BERT-small	92.01	54.05	99.09	73.41	61.17
	BERT-med	94.13	60.45	99.21	73.76	61.42
	BERT-base	94.50	62.41	99.32	75.08	62.76
	BERT-large	93.93	62.89	99.17	71.58	63.96
	RoBERTa	94.49	63.21	99.24	74.76	64.75
	ELECTRA	94.74	63.50	99.24	75.34	64.57
	BART	94.50	62.50	99.24	75.05	64.04
	GPT-2	94.34	61.17	99.14	74.28	63.04
	XLNet-base	94.30	62.84	99.16	74.58	64.34
较低	BERT-tiny	81.28/-11.18	32.00/-23.91	85.66/-13.04	49.42/-22.25	36.08/-21.37
	BERT-mini	82.20/-11.51	37.33/-21.17	94.43/ -4.58	54.70/-17.92	41.82/-18.42
	BERT-small	86.05/ -5.96	43.11/-10.95	97.39/ -1.70	61.00/-12.41	46.87/-14.30
	BERT-med	85.78/ -8.36	43.61/-16.84	97.47/ -1.74	59.74/-14.03	46.92/-14.50
	BERT-base	80.03/-	42.29/-20.12	86.61/-12.71	51.17/-23.91	43.42/-19.34
	BERT-large	65.43/-28.50	35.28/-27.62	77.84/-21.33	29.20/-42.38	35.24/-28.72
	RoBERTa	84.54/ -9.95	42.25/-20.96	88.34/-10.89	56.38/-18.38	44.80/-19.95
	ELECTRA	72.66/-22.08	48.01/-15.49	80.63/-18.61	42.17/-33.17	49.26/-15.30
	BART	78.13/-16.37	43.07/-19.43	83.64/-15.59	47.68/-27.37	45.25/-18.79
	GPT-2	89.55/ -4.79	47.70/-13.47	95.11/ -4.04	66.28/ -8.00	47.74/-15.30
	XLNet-base	88.50/ -5.80	50.75/-12.09	94.91/ -4.25	66.57/ -8.01	51.64/-12.70
顺序 1	BERT-tiny	87.80/ -4.66	40.32/-15.59	94.24/ -4.46	69.14/ -2.53	45.29/-12.16
	BERT-mini	88.87/ -4.84	42.88/-15.62	97.12/ -1.89	71.99/ -0.63	44.86/-15.38
	BERT-small	90.95/ -1.07	50.91/ -3.14	98.53/ -0.57	73.45/ +0.04	51.92/ -9.25
	BERT-med	91.21/ -2.92	52.01/ -8.43	98.78/ -0.43	74.24/ +0.47	53.13/ -8.29
	BERT-base	92.00/ -2.50	56.79/ -5.62	99.12/ -0.20	75.16/ +0.08	56.43/ -6.33
	BERT-large	92.43/ -1.50	59.51/ -3.38	98.84/ -0.33	75.83/ +4.25	59.01/ -4.95
	RoBERTa	92.82/ -1.67	60.07/ -3.14	98.70/ -0.54	75.45/ +0.68	60.18/ -4.57
	ELECTRA	91.50/ -3.24	54.79/ -8.71	97.57/ -1.67	76.01/ +0.67	55.18/ -9.38
	BART	93.66/ -0.84	60.82/ -1.68	98.78/ -0.46	75.64/ +0.59	61.03/ -3.01
	GPT-2	92.54/ -1.80	57.11/ -4.07	98.82/ -0.33	74.37/ +0.09	57.36/ -5.68
	XLNet-base	92.97/ -1.33	61.03/ -1.82	98.38/ -0.78	75.33/ +0.75	61.86/ -2.49
顺序 2	BERT-tiny	88.14/ -4.32	52.29/ -3.62	85.59/-13.11	53.72/-17.95	55.62/ -1.83
	BERT-mini	88.08/ -5.63	56.21/ -2.29	93.66/ -5.36	54.17/-18.45	59.18/ -1.05
	BERT-small	90.61/ -1.41	58.36/ +4.30	98.33/ -0.76	64.22/ -9.18	61.00/ -0.17
	BERT-med	91.28/ -2.86	59.89/ -0.55	98.50/ -0.71	64.49/ -9.28	61.89/ +0.47
	BERT-base	91.54/ -2.96	61.75/ -0.66	99.01/ -0.30	64.38/-10.70	63.00/ +0.24
	BERT-large	92.39/ -1.54	62.09/ -0.80	97.80/ -1.37	68.61/ -2.97	64.68/ +0.72
	RoBERTa	93.34/ -1.14	63.12/ -0.09	98.08/ -1.16	69.71/ -5.05	64.88/ +0.13
	ELECTRA	92.36/ -2.38	62.95/ -0.55	97.11/ -2.13	60.43/-14.91	65.09/ +0.53
	BART	93.26/ -1.24	62.72/ +0.22	98.05/ -1.18	69.55/ -5.50	64.53/ +0.49
	GPT-2	92.71/ -1.63	60.88/ -0.29	98.42/ -0.72	70.51/ -3.76	63.61/ +0.57
	XLNet-base	92.91/ -1.39	62.61/ -0.24	98.51/ -0.64	71.34/ -3.24	65.34/ +1.00
顺序 3	BERT-tiny	91.39/ -1.07	39.46/-16.45	94.46/ -4.24	60.59/-11.08	45.57/-11.88
	BERT-mini	92.80/ -0.91	46.80/-11.70	96.61/ -2.41	64.28/ -8.34	48.61/-11.63
	BERT-small	93.68/ +1.67	54.83/ +0.78	98.59/ -0.50	68.11/ -5.30	55.66/ -5.51
	BERT-med	93.67/ -0.46	55.97/ -4.47	98.16/ -1.05	68.71/ -5.05	55.50/ -5.92
	BERT-base	94.28/ -0.22	59.09/ -3.32	98.66/ -0.66	67.46/ -7.62	56.97/ -5.79
	BERT-large	94.49/ +0.55	58.03/ -4.87	97.17/ -2.00	68.78/ -2.80	55.41/ -8.55
	RoBERTa	94.87/ +0.38	60.78/ -2.43	98.91/ -0.33	70.92/ -3.84	60.92/
	ELECTRA	94.47/ -0.26	59.70/ -3.80	97.82/ -1.42	65.66/ -9.68	59.80/ -4.76
	BART	94.53/ +0.03	61.45/ -1.05	98.79/ -0.45	73.08/ -1.97	62.03/
	GPT-2	94.25/ -0.09	57.92/ -3.25	98.87/ -0.28	72.41/ -1.87	58.67/ -4.37
	XLNet-base	94.83/ +0.53	62.09/ -0.75	98.58/ -0.58	73.18/ -1.39	61.30/ -3.04
顺序 4	BERT-tiny	84.42/ -8.04	35.75/-20.16	98.07/ -0.63	66.26/ -5.41	42.14/-15.30
	BERT-mini	85.43/ -8.28	39.89/-18.61	98.82/ -0.20	68.83/ -3.79	44.82/-15.42
	BERT-small	88.84/ -3.17	50.37/ -3.68	99.13/ +0.04	70.96/ -2.45	53.67/ -7.50
	BERT-med	90.25/ -3.88	55.32/ -5.13	99.20/ -0.01	72.38/ -1.38	55.55/ -5.87
	BERT-base	90.91/ -3.59	59.62/ -2.79	99.33/ +0.01	73.78/ -1.30	60.17/ -2.59
	BERT-large	90.50/ -3.43	59.25/ -3.64	99.33/ +0.16	74.07/ +2.49	61.50/ -2.46
	RoBERTa	91.42/ -3.07	59.74/ -3.47	99.41/ +0.17	73.41/ -1.36	59.63/ -5.12
	ELECTRA	89.99/ -4.75	53.47/-10.03	99.24/ -0.00	73.12/ -2.22	58.91/ -5.66
	BART	92.39/ -2.11	60.53/ -1.97	99.29/ +0.05	75.16/ +0.11	60.82/ -3.22
	GPT-2	91.84/ -2.50	56.11/ -5.07	99.16/ +0.01	73.39/ -0.88	58.54/ -4.50
	XLNet-base	92.58/ -1.72	61.68/ -1.16	99.18/ +0.03	74.28/ -0.30	62.53/ -1.82

D 对其他预训练语言模型的进一步探测研究

正文中的讨论几乎完全围绕 $\text{BERT}_{\text{BASE}}$ 保留知识的能力展开。BERT（Devlin 等人，2019年）是预训练语言模型家族的代表，广泛应用于各种自然语言处理任务。我们在研究中选择 BERT，是因为其基于 Transformer 的架构影响了许多其他预训练语言模型。不过，这并不意味着 BERT 是唯一具有内在能力、能够长期为先前任务生成高质量表征的预训练语言模型。本节将进一步研究其他多种预训练语言模型，它们具有不同的模型规模、预训练过程或注意力机制。对于注意力机制或预训练策略各不相同的预训练语言模型，我们研究了 RoBERTa-base（Liu 等人，2019b）、BART-base（Lewis 等人，2020年）、ELECTRA-base（Clark 等人，2020年）、XLNet-base（Yang 等人，2019年）和 GPT-2（Radford 等人，2019年）。对于规模不同的预训练语言模型，我们研究了 BERT-tiny、BERT-mini、BERT-small、BERT-medium（均为 Turc 等人（2019年）提出的蒸馏版本），以及 Devlin 等人（2019年）提出的 BERT-large。我们的探测实验详情如下。

为减少冗余计算并提供简洁的定量分析，我们不再跟踪预训练语言模型在每个检查点的编码能力。此处，我们仅测量一个按顺序学习了所有任务且**没有** 任何记忆回放的预训练语言模型的编码能力。所有模型都采用单层网络作为解码器，这与第 3.2 节相同。此外，我们还使用与前面实验相同的设置，基于不同的预训练语言模型训练模型。在依次完成五个文本分类任务的训练后，我们保存预训练语言模型编码器的参数权重，并使用第 3.1 节提出的基于探测的方法对其进行评估。

我们强调，即使在单任务监督范式下进行训练，不同的预训练语言模型在同一任务上的表现也应有所不同。因此，我们提供了一些控制任务 （Hewitt 和 Liang，2019年）的结果作为比较。具体而言，我们分别在每个数据集上训练每个预训练语言模型，其中编码器和解码器的所有参数都可以更新。这些在单个任务上的全监督结果可以被视为**上界**。为了检验预训练语言模型本身是否能够在不进行下游微调的情况下较好地处理这些文本分类任务，我们还展示了一些零样本探测结果作为**较低界**。我们从开源平台下载不同预训练语言模型的权重，不进行任何微调。然后，我们分别为每个任务训练解码器，同时固定原始预训练语言模型的权重（实际上是在零样本场景下进行探测研究）。将其与控制任务的结果进行比较，我们可以检验其他预训练语言模型在学习一系列任务后，是否能够像 BERT 一样保留先前任务的知识。

我们在表2中列出了所有结果（包括上界和下界）。从中可以发现，尽管这些预训练语言模型采用了不同的注意力机制，规模也各不相同，但它们保持先前所学知识的内在能力相似。尽管没有使用情景回放进行训练，这些预训练语言模型的探测结果都远优于下界，而且不受训练顺序的影响。

比较不同规模 BERT 的结果可以发现，在没有使用情景回放的情况下，参数更多的编码器（例如 BERT-base 和 BERT-large）比参数较少的编码器（例如 BERT-tiny 和 BERT-mini）稍微更擅长保留旧任务知识。然而，在规模相近但架构不同的编码器中，包括 BERT-base（Devlin 等人，2019）、GPT-2（base）（Radford 等，2019）、BART（Lewis 等，2020）和 XLNet-base（Yang 等，2019），它们保留旧任务知识的能力也相似。因此，我们推测，这种避免遗忘的内在能力**部分源于模型规模**，而模型架构的差异（例如 Transformer 编码器与 Transformer 解码器）并未带来明显影响。

表示空间的结构

正如 Gao 等人（2019）和 Wang 等人（2020a）在其研究中所述，使用第 2.1 节所述的解码器进行训练时，大型预训练语言模型会将所有词嵌入到一个窄锥中。根据他们的观点，我们推测，预训练语言模型也会将同一标签的句子级表示向量生成在一个窄锥中。为验证这一设想，我们可以检查 BERT 生成的任意两个向量之间的余弦值。我们选取具有四个类别的 AGNews（Zhang 等，2015）进行研究。我们使用 BERT 和线性

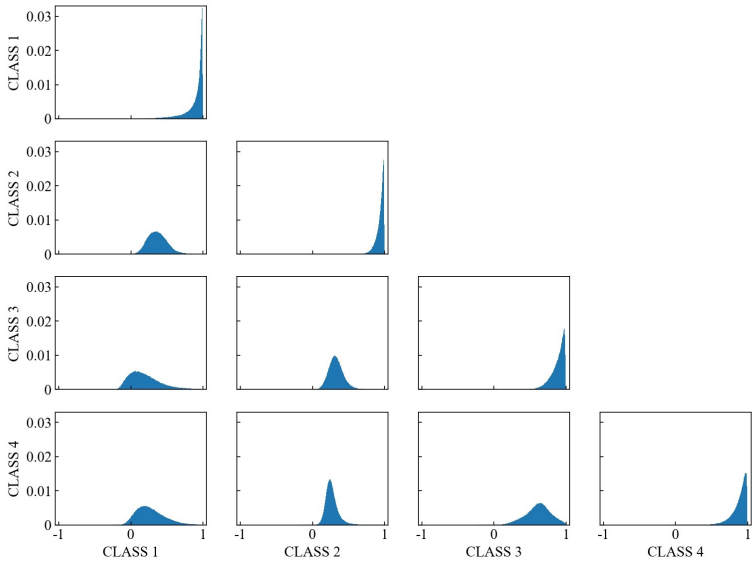


图 7：AGNews 各类别向量对的余弦分布，坐标轴已对齐。

解码器在 AGNews 上训练一轮，然后按类别分别存储训练集的表示向量。对于第 i 类和第 j 类（$1 \leq i \leq j \leq 4$），我们分别从每类中随机抽取一个向量，重复 1M 次。随后，我们可以近似得到来自两个类别的两个向量的余弦分布，如图 7 所示。

从结果中可以明显看出，从同一类别中采样的两个向量方向相近（它们之间的余弦值几乎为 1），而从不同类别中采样的两个向量方向则明显不同。这表明表示子空间具有各向异性，因此我们可以用凸锥来描述它们。

F 补充可视化结果

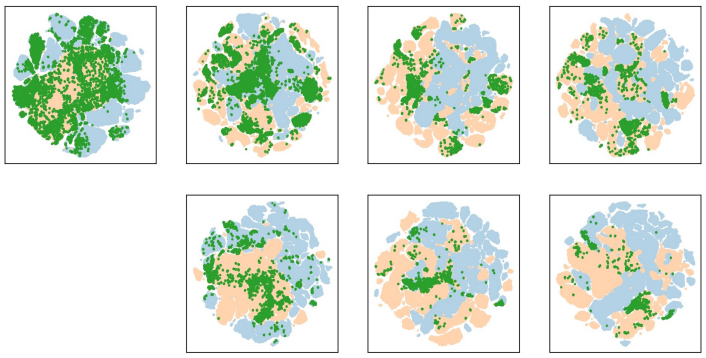


图8：终身学习过程中表示空间的补充可视化结果，其中 AGNews 和 Amazon 的点分别以黄色和蓝色标示。特别地，我们将混合区域标为绿色；当模型区分不同任务的能力更强时，该区域的面积应更小。从左到右，这些列分别对应于刚完成从 *AGNews* 学习以及第一次、第二次、第三次重放的时刻。上排为重放前的结果，下排为重放后的结果。

本节将可视化终身学习过程中记忆回放前后表示空间的变化。按照第 4 节中的实验设置，我们先在 AGNews 上训练模型，然后在 Amazon 上训练，并进行三次回放。我们保存了学习

AGNews 后以及每次回放前后的所有表示向量。接着，我们采用 t-SNE（van der Maaten 和 Hinton，2008）将所有向量绘制在平面上。考虑到两种类别的混合区域，我们可以得出结论：记忆回放在缓解任务间遗忘方面发挥着重要作用。每次回放后，模型区分来自不同任务的实例的能力都会增强，这体现在图8中绿色区域的减少上。此外，通过比较各列中的结果，我们可以确认，尽管连续学习单个任务而不间断会导致任务之间出现少许混淆，但稀疏记忆回放能够有效消除这种混淆。因此，经记忆回放增强的 BERT 模型不仅能够抵御任务内遗忘，也能抵御任务间遗忘。