

元数据条件在语言模型预训练中什么时候（不起作用）？关于上下文无关文法的研究

Rei Higuchi
The University of Tokyo, RIKEN AIP
higuchi-rei714@g.ecc.u-tokyo.ac.jp

Ryotaro Kawata
The University of Tokyo, RIKEN AIP
kawata-ryotaro725@g.ecc.u-tokyo.ac.jp

Naoki Nishikawa
The University of Tokyo, RIKEN AIP
nishikawa-naoki259@g.ecc.u-tokyo.ac.jp

Kazusato Oko
University of California, Berkeley, RIKEN AIP
oko@berkeley.edu

Shoichiro Yamaguchi
Preferred Networks, Inc.
guguchi@preferred.jp

Sosuke Kobayashi
Preferred Networks, Inc.
sosk@preferred.jp

Seiya Tokui
Preferred Networks, Inc.
tokui@preferred.jp

Kohei Hayashi
The University of Tokyo
hayashi.kohei@gmail.com

Daisuke Okanohara
Preferred Networks, Inc.
hillbig@preferred.jp

Taiji Suzuki
The University of Tokyo, RIKEN AIP
taiji@mist.i.u-tokyo.ac.jp

Abstract

获取潜在语义的能力是决定语言模型性能的关键属性之一。激发这种能力的一种便捷方法是在预训练数据的文本开头添加元数据（例如 URL、域名和样式），使模型在观察整个文本之前更容易访问潜在语义。先前的研究报告称，这项技术确实提高了训练模型在下游任务中的性能；然而，这种改进仅在特定的下游任务中被观察到，而在平均下一个词元预测损失中没有观察到一致的提升。为了理解这一现象，我们仔细调查了在预训练期间添加元数据如何影响模型性能，方法是使用人工数据检查其行为。有趣的是，我们发现这种方法对下游任务产生了正面和负面的影响。我们证明了这种方法的有效性取决于是否能够从下游任务的提示中推断出潜在语义。具体来说，通过使用由概率上下文无关文法生成的数据进行调查，我们展示了当给定上下文足够长以推断潜在语义时，元数据训练有助于改善模型性能。相反，当上下文缺乏进行准确后验推理所需的信息时，这种技术会对性能产生负面影响。

1 引言

语言模型 (LMs) 具有适应各种下游任务的能力，这得益于在具有多样化潜在语义的数据集上进行的预训练 (Gao et al., 2020; Arora & Goyal, 2023; Hahn & Goyal, 2023; Zhao et al., 2024; Zhang et al., 2025)。虽然来自不同来源和领域的数据是语言模型成功的关键之一，但它也是预训练困难的原因。数据集通常既包含诸如维基百科 (Merity et al., 2016) 这样的结构化、事实性文本，也包含诸如社交媒体帖子 (Baumgartner et al., 2020) 这样非结构化、随意的文本。不仅文档的风格各异，主题也极为广泛。

为了更好地利用潜在语义的多样性，一个有前途的方法是以元数据为条件，即关于文档的来源、风格或人类偏好等高级信息，这些信息在文档本身中无法直接观察到 (Keskar et al., 2019; Korbak et al., 2023; Khalifa et al., 2024)。例如，Gao et al. (2025) 报告说，通过在语言模型的预训练期间在每个网页文本前加上 URL 或主题，即使在推理过程中未使用这些提示，性能在几个下游任务中得到了改善。这一策略使模型无需观察整个文档即可访问数

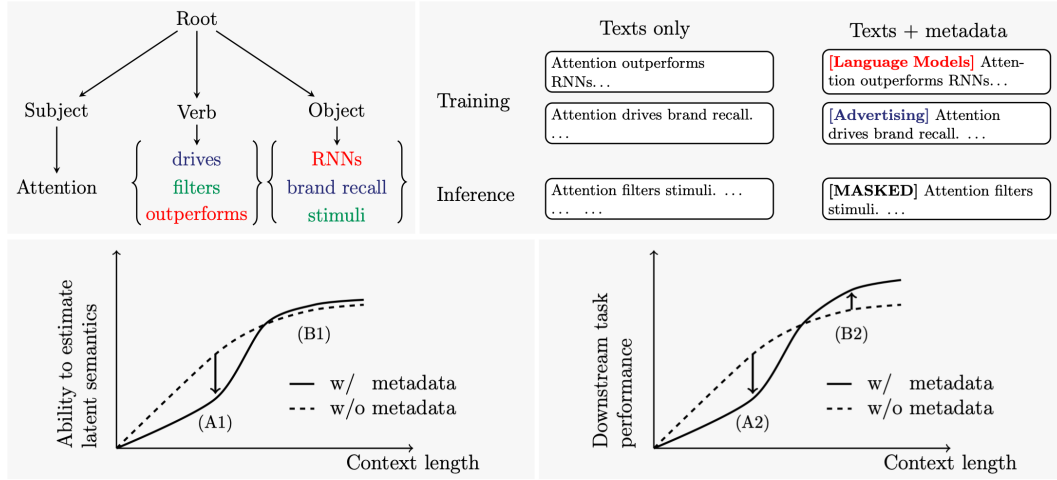


Figure 1.1: 概述和发现。顶部：元数据如何缩小潜在语义以及如何将其纳入训练数据的概念插图。底部：使用元数据对潜在语义估计能力和下游任务性能的正面和负面影响。

据的潜在语义，人们可能推测它触发了模型更好地捕获这些潜在语义的能力，可能导致性能的提升。

然而，目前尚不清楚提供元数据是否作为语言模型预训练的通用解决方案。事实上，正如在 Gao et al. (2025) 中报告的那样，下游任务性能的改进在不同任务之间并不一致，而且，添加元数据并不会导致预训练损失（通过下一个词预测的交叉熵损失衡量）的降低。尽管元数据条件有吸引力和不透明性，由于实际数据中潜在语义的复杂性，追踪其如何影响模型性能仍然具有挑战性。因此，研究界尚未全面了解元数据条件在何时起作用。

在本文中，我们系统地研究了在预训练期间附加元数据对语言模型能力的影响。为了避免真实数据的复杂性导致分析困难，我们在可控且易于理解的合成数据上进行实验。具体而言，受 Allen-Zhu & Li (2023) 的启发，我们基于上下文无关文法（CFGs）生成合成数据。我们使用多种 CFG 规则生成数据，并将元数据定义为指示生成每个样本所使用规则的信息。我们通过性能评估和探测分析研究所提供元数据量如何影响模型。

我们的一项重要发现是，元数据条件可能会根据任务的不同而对性能产生负面影响。这在 Figure 1.1 的底部部分直观地得到了说明。如 (A1) 所示，我们的实验结果表明，当任务提示较短时，与未使用元数据训练的模型相比，使用元数据训练的模型在推断潜在语义的准确性较低。这导致下游任务的性能下降，如 (A2) 所示。另一方面，如 (B1) 所示，当任务提示变长时，使用元数据训练的模型能够与未使用元数据训练的模型相当地推断潜在语义。在这种情况下，使用元数据训练的模型在下游任务中表现出更高的性能，如 (B2) 所示。

我们的贡献可以总结如下：

- 我们研究在预训练期间添加元数据的影响。为此，我们基于具可控元数据的 CFGs 生成合成数据集。我们在有和没有元数据的情况下进行预训练，并比较模型的性能。
- 我们发现了一个下游任务性能中的隐藏权衡：通过在预训练期间添加元数据，长提示任务的性能有所提升，而短提示任务的性能则没有提高。我们使用多种指标评估训练模型，并证明这种权衡的出现是因为元数据条件削弱了模型从有限上下文中推断潜在语义的能力。
- 为了说明为何使用元数据训练的模型未能推断潜在语义及其如何影响下游性能，我们引入了一个理论框架，使用与潜在语义相关的预测分布。

2 具有可控元数据的合成数据集

我们使用系统且可控的概率上下文无关文法（PCFGs）构建了一个数据集 (Allen-Zhu & Li, 2023; Ahuja et al., 2025)。我们通过加入元数据扩展了在 Allen-Zhu & Li (2023) 中介绍的 D 级 PCFG。

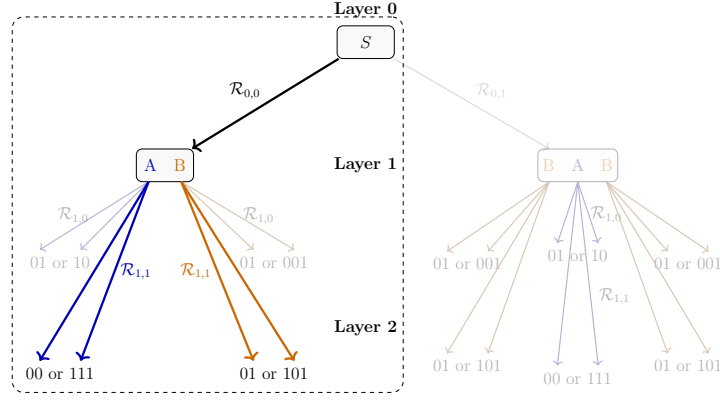


Figure 2.1: 一个具有分层元数据 $(j_0, j_1) = (0, 1)$ 的 2 级 CFG 的例子。非终结符号集是 $\mathcal{S}_0 = \{S\}$, $\mathcal{S}_1 = \{A, B\}$, 终结符号是 $\mathcal{S}_2 = \{0, 1\}$ 。语法是 $\mathcal{G}((0, 1)) = (\{\mathcal{S}_i\}_{i=0}^2, \{\mathcal{R}_{0,0}, \mathcal{R}_{1,1}\})$ 。支持是 $L(\mathcal{G}((0, 1))) = \{0001, 00101, 11101, 111101\}$ 。

2.1 D 级 PCFG

首先，我们解释使用 PCFG 的句子生成。一个 PCFG 用 $\mathcal{G}$ 表示，它由符号集 $\{\mathcal{S}_i\}_{i=0}^D$ 和生成规则 $\{\mathcal{R}_i\}_{i=0}^{D-1}$ 组成。具体而言，我们从一个由唯一根符号 $S \in \mathcal{S}_0$ 组成的长度为 1 的序列开始，然后对于 $i = 0, \dots, D-1$ ，依次从前一个序列 $\mathcal{S}_i$ 生成 $\mathcal{S}_{i+1}$ 中符号的下一个序列。每个生成规则 $\mathcal{R}_i$ 的长度为 2 或 3，并由以下形式的规则组成：在第 i 步，我们根据生成规则 $\mathcal{R}_i$ 替换前一句中的每个符号以生成下一个句子。通过重复此过程 D 次，我们获得最终由 $\mathcal{S}_D$ 中的符号组成的序列，我们称这些符号为终端符号。

为简单起见，假设不同层次的符号是互不相交的（即， $\mathcal{S}_i \cap \mathcal{S}_j = \emptyset$ 对于 $i \neq j$ ）。使用 PCFG $\mathcal{G}$ 生成的分布表示为 $P(\mathcal{G})$ ，其支持表示为 $L(\mathcal{G})$ 。句子生成的过程在算法 1 中总结。

Algorithm 1 句子生成的步骤

- 1: Set the initial state as $x_0 = S \in \mathcal{S}_0$.
 - 2: for $i = 0, \dots, D-1$ do
 - 3: Keep $x_i = (s_{i,0}, \dots, s_{i,K_i-1})$ where $s_{i,k} \in \mathcal{S}_i$ for $k = 0, \dots, K_i-1$.
 - 4: for $k = 0, \dots, K_i-1$ do
 - 5: Randomly sample a rule $r \in \mathcal{R}_i$ whose input is $s_{i,k}$ with uniform probability.
 - 6: Replace $s_{i,k}$ by the right-hand side of the rule r (e.g., s_2, s_3, s_4 or s_2, s_3).
 - 7: Output $x_D = (s_{D,0}, \dots, s_{D,K_D-1})$ where $s_{i,k} \in \mathcal{S}_D$.
-

2.2 D 级 PCFG 与层次元数据

现在，我们在 D 级 PCFG 的背景下引入元数据的概念，并描述如何在元数据约束下生成一个序列。概述如图 2.1 所示。

当我们将元数据概念化为生成句子过程中的高层信息时，我们将其操作化为一组用于调节生成规则 $\{\mathcal{R}_i\}_{i=0}^{D-1}$ 的元级信息。具体来说，我们假设每个生成规则 $\mathcal{R}_i$ 有多个选择 $(\mathcal{R}_{i,j})$ 。在每个级别 i ，我们选择其中一个规则，其索引由 j_i 表示。我们正式地将元数据定义为对应于所选择的生成规则的索引序列 $(j_0, \dots, j_{D-1})$ 。一旦规则序列 $\{\mathcal{R}_{i,j_i}\}_i$ 确定后，便构建 PCFG 文法为 $\mathcal{G}((j_0, \dots, j_{D-1})) = (\{\mathcal{S}_i\}_{i=0}^D, \{\mathcal{R}_{i,j_i}\}_{i=0}^{D-1})$ ，从中我们生成一个序列。请注意，不同的序列使用不同的 $(j_0, \dots, j_{D-1})$ 样本。

这样，在 PCFG 框架内，我们将元数据定义为指定生成规则选择的元信息。尽管元数据直接定义了生成句子的内容，但它相较于完整的规则集 $\{\cup_j \mathcal{R}_{i,j}\}_{i=0}^{D-1}$ 限制了可能的句子结构。这类似于在实际环境中，标签如来源、风格或人类偏好大致划分了合理句子的空间。

2.3 数据生成

最后，我们描述数据集是如何生成的。为了研究提供的元数据量如何影响语言模型 (LMs) 的学习，我们在 PCFG 中操控元数据标签 j_i 的层级可用性。我们将提供深度为 $D_M (\leq D)$

的元数据称为从完整元数据 $(j_0, \dots, j_{D-1})$ 中提供部分元数据序列 $(j_0, \dots, j_{D_M-1})$ 。大致来说， D_M 量化了可用元数据的程度，而 $D - D_M$ 表示模型在没有元数据指导的情况下必须在规则空间中导航的范围。在实践中，这对应于控制句子被条件化的具体程度。

为了提供深度为 D_M 的元数据，我们在句子 $x \sim P(\mathcal{G}((j_0, \dots, j_{D-1}))) = P(\{\mathcal{S}_i\}_{i=0}^D, \{\mathcal{R}_{i,j_i}\}_{i=0}^{D-1})$ 前加上 D_M 元数据标记 $(j_0, \dots, j_{D_M-1})$ ，然后是 $D - D_M$ 掩码标记。训练数据集是通过混合上面描述的加了元数据的句子和以 D 掩码标记为前缀的句子构成的，且它们都是等概率的。详细信息，请参见附录 A。用于推断的数据集以及用于比较的无元数据训练数据集是通过简单地在每个生成的序列前加上 D 掩码标记构成的。插入非信息性的掩码标记的目的是为了在提供信息量不同的情况下保持恒定的序列长度。我们特别注意控制这一方面，因为之前的研究表明，即使是中性的标记，比如 sink 标记，也可能影响学习 (Darcet et al., 2024; Gu et al., 2025)。

3 实验

我们现在通过对多个任务和指标的实验来研究元数据调节何时以及如何影响模型行为。我们选择我们的 Transformer 模型，遵循类似在 CFGs 上训练的 Allen-Zhu & Li (2023) 的做法。具体来说，我们采用 LLaMA 架构 (Touvron et al., 2023)，该架构具有 12 层、12 个注意力头、RoPE (Su et al., 2024)，以及 768 的隐藏层大小。数据生成过程如第 2 节所述，使用具有层次级别 $D = 5$ 的 PCFG。每个 $\mathcal{R}_i$ 从 $\mathcal{R}_{i,0}$ 或 $\mathcal{R}_{i,1}$ 之一中以等概率选择。每个级别只有两个候选项可能显得有限，但每个元数据标签在指数上缩小了可能文法的集合。大致而言，完整的元数据相比于没有元数据的情况，将生成过程中的模糊性减少了 $2^5 = 32$ 倍。因此，这种设置足以有意义地检验元数据的影响。

我们通过改变前置元数据的深度 $D_M \in \{0, 1, 3, 5\}$ 来比较结果，其中较大的 D_M 意味着更丰富的条件。我们使用交叉熵损失进行下一个标记预测 (NTP) 来预训练模型。损失仅针对终端符号和 EOS 标记进行计算，不包括 BOS 标记和元数据。因此，训练设置之间的唯一差异在于是否包含元数据，并且不提供额外的监督。更多讨论见附录 A。

3.1 使用或不使用元数据进行训练不影响下一个令牌预测损失

没有元数据的测试损失。 作为初步结果，我们检查了下一个标记预测的性能。Figure 3.1 的左侧显示了整个训练过程中的测试损失，这是在没有元数据的推断下评估的。尽管由于层次分支，我们的数据集比 Allen-Zhu & Li (2023) 的更为复杂，但这些模型很好地减少了损失。值得注意的是，我们发现使用元数据 ($D_M \geq 1$) 和不使用元数据 ($D_M = 0$) 训练的模型之间的损失没有显著差异，这与 Gao et al. (2025) 的见解一致。

带有元数据的测试损失。 右侧 of Figure 3.1 展示了在带有附加元数据的数据上进行推理时的损失。我们可以看到，它明显小于左图中的损失（没有元数据的情况下推理）。这表明如果元数据是可用的，使用元数据条件训练的模型在推理时确实能表现得更好。这个观察结果与推理时不使用元数据的情况形成对比。

我们在以下重要观察¹中总结这些结果：

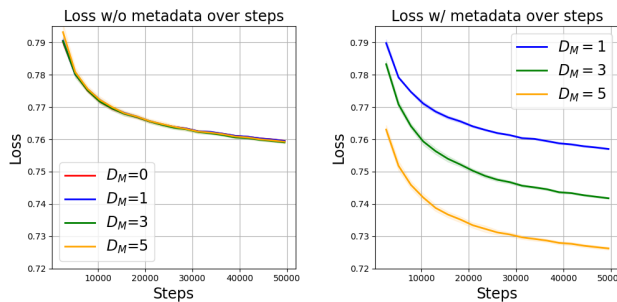


Figure 3.1: 训练期间下一个令牌预测的测试损失。左：没有元数据。右：有元数据。

¹我们还在附录 C 中检查了校准误差，这进一步支持这一发现。

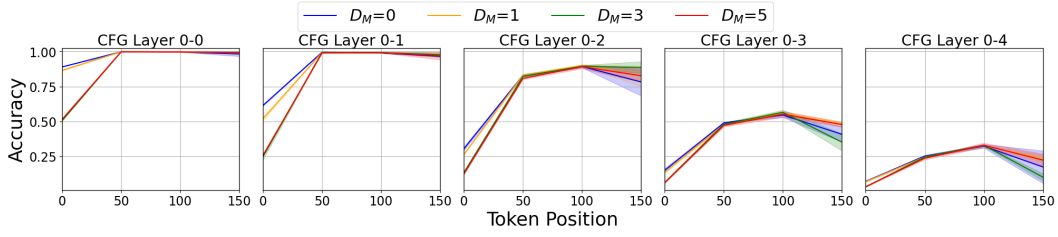


Figure 3.2: 元数据探测准确性。我们绘制了相对于标记位置的转换图。我们使用了 Transformer 的第 12 层，每一列对应于 CFG 树的深度。 d 列 ($d = 1, 2, 3, 4, 5$) 代表从层 0 到层 $d - 1$ 分类 CFG 树分支的准确性。从层 0 到层 $d - 1$ 正确预测元数据的概率是 $1/2^d$ 。

Observation. For average next-token prediction loss, metadata conditioning improves performance only when metadata is available at inference time. If metadata is not accessible at inference, no significant improvement is observed.

3.2 使用元数据进行训练降低了元数据预测的准确性

平均下一个标记预测损失仅提供了一个粗略的模型行为度量，无法捕捉细微区别。这个局限性使得以往研究难以理解元数据条件的机制。然而，我们的控制设置能够进行更详细的研究。为此，为了检查训练模型能否正确推断文档中的潜在语义，我们考虑一个探测任务 (Conneau et al., 2018; Belinkov, 2022; Allen-Zhu & Li, 2023) 来预测元数据，以便调查训练的模型是否能正确提取潜在语义。

探测的细节。 设 $l \in \{0, \dots, 11\}$ 是一个训练模型的层索引， $d \in \{0, \dots, D - 1\}$ 是 PCFG 规则的一个层级。给定一个来自 $\mathcal{G}((j_0, \dots, j_{D-1}))$ 的输入，令 $h_{l,i}$ 表示从位置 i 开始的 5 个连续标记在层 l 中的隐状态的平均值。对于每个 (l, d, i) ，我们训练 d 个不同的线性二元分类器（冻结所有模型参数）；第 d' 个分类器预测来自 $h_{l,i}$ 的元数据。我们将元数据探测准确率定义为所有 d 个分类器返回正确标签的比率。后续标记的隐状态通常会聚合更长范围输入的信息，因此往往提供更好的探测性能。因此，我们评估不同位置 i 的准确率，以了解需要多少上下文才能恢复元数据。我们还在多层（特别是第 1、5、8 和 12 层）进行探测，以全面了解元数据信息在整个网络中的分布方式，不过我们将仅展示第 12 层的结果，因为其他层的结果表现相似。其他层的结果可以在 Appendix D 找到。

结果。 结果汇总在 Figure 3.2 中。在看到结果之前，人们可能会直观地预期，使用元数据条件训练的模型会将潜在语义与文档内容相关联，从而提高元数据探测的准确性。然而实际上，元数据条件反而降低了标签探测的准确性，而不是提高。特别是，当提示长度较短时，使用元数据训练的模型的准确性明显低于未使用其训练的模型。当在预训练期间添加深度为 3 或 5 的元数据时，提示长度为 5（标记位置 $i = 0$ ）的准确性大约为随机水平。这表明，如果提示信息提供过少，使用元数据训练的模型在推理时无法推断出潜在语义。

相比之下，随着提示长度的增加，使用元数据训练的模型逐渐能够准确预测元数据。实际上，在提示长度为 5 时，观察到的准确率差距（在 $D_M = 0$ 和 $D_M = 3, 5$ 之间）在提示长度为 50 或 100 时基本上消失。这表明，只要提示足够长以提供必要的信息，使用元数据训练的模型就能够从序列中提取潜在的语义。总之，通过探测，我们得到以下发现：

Finding 1. Metadata conditioning leads models to fail to infer latent semantics . In particular, if the prompt length is short, models trained with metadata conditioning cannot infer latent semantics, while models trained without metadata can. When the prompt becomes longer , information in the prompt increases, which allows both types of models to better capture latent semantics .

在 Section 4 中，我们提供了一个理论框架来解释为何会出现这种违反直觉的性能下降。

最后，我们分析了元数据条件对下游任务性能的影响。虽然先前的研究如 Gao et al. (2025) 报道了下游任务的改进，但它们并没有提供对此效果的详细解释。为了获得更深刻的理解，我们进行序列生成，并采用一个由 Allen-Zhu & Li (2023) 引入的称为语法精度的度量，该度量衡量模型生成语法正确句子的准确率。

这里，我们描述如何计算该度量。首先，我们随机提供一个模型序列的初始部分，并评估它能否生成语法正确的序列。具体来说，我们首先从 $L(\mathcal{G})$ 获得 n 个独立同分布的样本序列 $\{(x_{i,1}, \dots, x_{i,K_i})\}_{i=1}^n$ ，其中 $\mathcal{G} := (\{\mathcal{S}_i\}_{i=0}^D, \{\cup_{j_i} \mathcal{R}_{i,j_i}\}_{i=0}^{D-1})$ 代表混合 PCFG。然后，我们提取前 l_p 个标记 $(x_{i,1}, \dots, x_{i,l_p})$ 作为提示，并让模型自回归地生成标记序列，直到它输出 [EOS]。我们用 $(y_{i,l_p+1}, \dots, y_{i,M_i})$ 表示生成的序列，并定义 $z_i := (x_{i,1}, \dots, x_{i,l_p}, y_{i,l_p+1}, \dots, y_{i,M_i})$ 用于 $i = 1, \dots, n$ 。利用 z_i ($i = 1, \dots, n$)，我们定义语法准确性 (GA) 为 $\text{GA}(\mathcal{G}) = \sum_{i=1}^n \frac{\mathbb{1}[z_i \in L(\mathcal{G})]}{n}$ 。值 $\mathbb{1}[z_i \in L(\mathcal{G})]$ ，即 z_i 是否为语法 $\mathcal{G}$ 的正确序列，可以通过 (确定性) CYK 算法来确定。正如在前一个小节中所见，使用元数据训练的模型性能高度依赖于提示长度。因此，我们在不同的提示长度 l_p 上展示这一度量。

结果。 结果显示在 Table ?? 中。令人惊讶的是，这似乎与 Gao et al. (2025) 的发现相反，后者显示了对下游任务性能的改善。然而，当提示长度较短时，元数据条件导致语法准确性下降。然而，随着提示长度的增加，使用元数据条件训练的模型得以恢复，达到与不使用元数据训练的模型相当或更好的性能。

这种模式让人联想到发现 1：使用元数据训练的模型难以从简短提示中推断潜在语义，从而导致任务性能下降。另一方面，当提示包含足够的信息时，模型能够成功推断元数据。如前所述（观察），当元数据明确可用时，基于元数据的模型会实现更低的损失；因此，从更丰富的提示中推断出的元数据同样会增强下游任务的性能。这些结果导致以下发现：

Finding 2. Metadata conditioning degrades the downstream task performance if the task prompt is short . This is because models trained with metadata cannot infer latent semantics from the short prompt. On the other hand, for tasks with long prompts, metadata conditioning improves the performance since the models can easily extract latent semantics from the prompts.

4 实验结果和发现的解读

我们比较了在有和没有元数据的情况下，模型在隐含学习潜在语义时的差异。在没有元数据的情况下，模型需要直接从数据中学习语法的层次混合，这由于可能的语法结构多样而具有挑战性。而在有元数据的情况下，语法推断变得更容易，但对元数据本身的准确预测变得必要。我们强调了推断和学习中的这种权衡。

4.1 一种通过边缘化在有元数据的情况下获取潜在语义的框架

4.1.1 无元数据训练

我们首先形式化在没有元数据的情况下进行训练时潜在语义的作用。给定提示 x ，输出 y 的预测分布可以分解为

其中 $\mathcal{G}_L$ （例如 PCFG）是潜在语义。为简单起见，我们关注 $\mathcal{G}_L$ 是 PCFG 的情况。从边缘化原则的分解意味着输出 y 可以视为通过两个步骤生成：当在不提供元数据 j 的情况下训练时，模型隐式地在内部逼近似然 $p(y|x, \mathcal{G}_L)$ 和后验 $p(\mathcal{G}_L|x)$ 。即使我们知道 PCFG

$\{\mathcal{G}(j) \mid j \in \{0,1\}^D\}$ 的候选集, $p(\mathcal{G}_L|x)$ 的预测也并不简单, 因为我们从不同的分层 CFG 中获取样本: 文本数据的总支持度为 $\bigcup_{j \in \{0,1\}^D} L(\mathcal{G}(j))$ 。

4.1.2 使用元数据进行训练

另一方面, 当我们使用元数据 $j = (j_0, \dots, j_{D-1})$ 训练模型时, 推理中输出 y 的预测分布也可以被分解为

$$p(y|x) = \int \underbrace{\quad}_{\text{Likelihood in training w/ } j} \underbrace{\quad}_{\text{Posterior of } j} dj = \int \int \underbrace{\quad}_{\text{Likelihood}} \underbrace{\quad}_{\text{Posterior given } j} d\mathcal{G}_L \underbrace{\quad}_{\text{Posterior of } j} dj.$$

Likelihood in training with j

。需要注意的是, 在这种设置中 $p(y|x, \mathcal{G}_L, j) = p(y|x, \mathcal{G}_L)$ (一旦我们知道语法 $\mathcal{G}_L$, 我们就不需要 j)。这种分解可以通过三个步骤表达:

(A-1) 从后验分布 $p(j|x)$ 中抽样元数据 $j = (j_0, \dots, j_{D-1})$ 。

(A-2) 在给定元数据 $p(\mathcal{G}_L|x, j)$ 的情况下, 从其更丰富的后验分布中抽样生成一个 PCFG $\mathcal{G}_L$ 。

(B) 使用似然函数 $p(y|x, \mathcal{G}_L)$, 在条件为 x 和 $\mathcal{G}_L$ 的情况下生成 y 。

。在这种情况下, 模型专注于任务 (A-2) 和 (B), 因为提供了 j 的信息。因此, $p(y|x, j)$ 在模型中被隐式地构建。由于训练数据集中包括带有和不带元数据的样本, (A-1) 和 (A-2) 是分别学习的。

4.2 通过一个正式框架理解元数据条件的权衡

所提出的框架强调了元数据条件对推理的积极和消极影响:

- ✓ 元数据条件化的优势在于可以促进在给定元数据 $p(\mathcal{G}_L|x, j)$ 的情况下学习更丰富的后验。一旦我们获得 CFG $\{\mathcal{G}(j) \mid j \in \{0,1\}^D\}$ 的候选项, 在 (A-2) 中预测 $p(\mathcal{G}_L|x, j)$ 就比在 (A) 中估计 $p(\mathcal{G}_L|x)$ 要容易得多。
- ✗ 元数据条件化的缺点在于 (A-1) 中后验 $p(j|x)$ 的估计性能会下降。当在没有元数据的情况下进行推断时, 潜在语义 $p(\mathcal{G}_L|x) = \int p(\mathcal{G}_L|x, j)p(j|x)dj$ 的后验退化, 使得准确捕捉潜在语义变得更加困难。

这些框架使我们能够将之前的实验结果解释如下:

通过元数据条件化, 预测 $p(j|x)$ 的效率较低, 但预测 $p(\mathcal{G}_L|x, j)$ 的效率比没有元数据时更高。这意味着在预测 $p(j|x)$ 和 $p(\mathcal{G}_L|x, j)$ 时存在权衡。因此, 无论是否使用元数据条件化, 平均下一个标记的预测损失保持一致。

(ii) **发现 1 的解释。** 我们发现, 当使用元数据进行训练时, $p(j|x)$ 的预测在发现 1 中变得退化。这对 $p(\mathcal{G}_L|x)$ 的预测产生了负面影响, 如 (i) 中所讨论的。然而, 当任务提示 x 足够丰富时, 预测 $p(j|x)$ 很容易, 元数据条件对 $p(\mathcal{G}_L|x)$ 预测的负面影响变小。

当提示 x 受到限制时, 如 (ii) 中提到的, 预测 $p(j|x)$ 变得困难。在这种情况下, 无元数据模型能够更准确地预测 $p(\mathcal{G}_L|x)$, 并实现更高的语法准确性 (下游性能)。当 x 丰富时, 预测 $p(j|x)$ 变得容易。在这种情况下, 无元数据模型和有元数据模型在学习 $p(j|x)$ 的效果上没有区别。由于元数据调优模型能够高效地预测 $p(\mathcal{G}_L|x, j)$, 它可以更准确地隐式构建 $p(\mathcal{G}_L|x) = \int p(\mathcal{G}_L|x, j)p(j|x)dj$ 。因此, 有元数据设置能够实现更高的语法准确性。

5 相关工作

生成文本的控制: 尽管经过指令调整的模型可以遵循用户用自然语言书写的提示, 但仅仅依赖这样的指令往往会导致模糊性或不一致的输出。作为补充方向, Keskar et al. (2019)、Chan et al. (2020)、Dathathri et al. (2019) 和 Aghajanyan et al. (2021) 通过训练带有元数据的模型来解决这一问题, 这些元数据编码了领域、风格、任务或信息来源等属性。Korbak

et al. (2023) 结合了从人类判断中得出的偏好评分作为条件信号，即使在无条件生成中也能减少有害内容。经过这样的训练，元数据可以用于更可靠地引导模型生成特定类型或写作风格的文本。

效率和性能在 LLM 训练中的改善：大语言模型的训练方法结合了多种预训练和后训练阶段及多样化的标注形式，已显示出改进的下游性能。例如，Aghajanyan et al. (2021) 利用超文本结构，Gao et al. (2025) 结合了如 URL 的元数据，而 Korbak et al. (2023) 在预训练期间使用了人类偏好评分。Zhang et al. (2024) 通过基于简短的主题相关提示进行条件训练，以减少后训练期间的灾难性遗忘。Allen-Zhu & Li (2024) 表明，通过在训练时使用标记为“垃圾”的特殊标识符，可以使模型更有效地分配其能力。Krashennnikov et al. (2023) 展示了标记来源可靠性有助于模型获得隐含的元学习能力，并在微调过程中优先处理高质量的信息。

这些研究表明，元数据不仅可以增强可控性，还可以提高性能。然而，当元数据在预训练中使用而在推理时不可用时，这种条件如何影响模型行为仍然不清楚。我们的研究通过使用受 Allen-Zhu & Li (2023) 启发的 PCFGs 进行的控制实验直接解决了这一问题，并揭示出其好处关键在于潜在语义与推理时提示信息之间的关系。

6 结论

在本研究中，我们系统地研究了在语言模型预训练期间添加元数据的影响，利用从上下文无关语法生成的合成数据以及可控的元数据。通过比较有元数据和没有元数据训练的模型，并在理论框架的指导下，我们旨在阐明元数据条件在何种情况下影响模型性能。

我们的关键发现揭示了一种权衡：元数据条件化可以提升具有足够长任务提示的下游任务的性能，这些提示允许对潜在语义进行预测，但在任务提示过短以至无法预测潜在语义的任务中，其性能下降。这些发现强调了在实际的预训练中，需要对元数据条件化采用更为细致的方法，提出了一个重要的实践指导：简单地最大化元数据并不总是有利的。相反，策略应该根据下游任务的预期性质进行量身定制。条件化使用超过典型提示中可能推断的元数据可能对下游任务产生反效果。

未来的工作应关注于开发预训练策略，以减轻这种权衡，可能通过基于任务复杂性自适应地调整元数据的使用。此外，探索不同元数据类型对模型性能的影响将为优化多种应用的元数据条件提供宝贵的见解。

References

- Armen Aghajanyan, Dmytro Okhonko, Mike Lewis, Mandar Joshi, Hu Xu, Gargi Ghosh, and Luke Zettlemoyer. Htln: Hyper-text pre-training and prompting of language models. arXiv preprint arXiv:2107.06955 , 2021.
- Kabir Ahuja, Vidhisha Balachandran, Madhur Panwar, Tianxing He, Noah A. Smith, Navin Goyal, and Yulia Tsvetkov. Learning syntax without planting trees: Understanding hierarchical generalization in transformers. Transactions of the Association for Computational Linguistics , 2025. URL https://direct.mit.edu/tacl/article/doi/10.1162/tacl_a_00733/127877.
- Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 1, learning hierarchical language structures. arXiv preprint arXiv:2305.13673 , 2023.
- Zeyuan Allen-Zhu and Yuanzhi Li. Physics of language models: Part 3.3, knowledge capacity scaling laws. arXiv preprint arXiv:2404.05405 , 2024.
- Sanjeev Arora and Anirudh Goyal. A theory for emergence of complex skills in language models. arXiv preprint arXiv:2307.15936 , 2023.
- Jason Baumgartner, Savvas Zannettou, Brian Keegan, Megan Squire, and Jeremy Blackburn. The pushshift reddit dataset. In Proceedings of the international AAAI conference on web and social media , volume 14, pp. 830–839, 2020.
- Yonatan Belinkov. Probing classifiers: Promises, shortcomings, and advances. Computational Linguistics , 48(1), March 2022. doi: 10.1162/coli_a_00422. URL <https://aclanthology.org/2022.cl-1.7/>.
- Alvin Chan, Yew-Soon Ong, Bill Pung, Aston Zhang, and Jie Fu. Cocon: A self-supervised approach for controlled text generation. arXiv preprint arXiv:2006.03535 , 2020.
- Alexis Conneau, German Kruszewski, Guillaume Lample, Loïc Barrault, and Marco Baroni. What you can cram into a single \$ & ! # * vector: Probing sentence embeddings for linguistic properties. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) , July 2018. doi: 10.18653/v1/P18-1198. URL <https://aclanthology.org/P18-1198/>.
- Timothée Darcet, Maxime Oquab, Julien Mairal, and Piotr Bojanowski. Vision transformers need registers. In The Twelfth International Conference on Learning Representations , 2024. URL <https://openreview.net/forum?id=2dnO3LLiJ1>.
- Sumanth Dathathri, Andrea Madotto, Janice Lan, Jane Hung, Eric Frank, Piero Molino, Jason Yosinski, and Rosanne Liu. Plug and play language models: A simple approach to controlled text generation. arXiv preprint arXiv:1912.02164 , 2019.
- Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. The pile: An 800gb dataset of diverse text for language modeling. arXiv preprint arXiv:2101.00027 , 2020.
- Tianyu Gao, Alexander Wettig, Luxi He, Yihe Dong, Sathika Malladi, and Danqi Chen. Metadata conditioning accelerates language model pre-training. arXiv preprint arXiv:2501.01956 , 2025.
- Xiangming Gu, Tianyu Pang, Chao Du, Qian Liu, Fengzhuo Zhang, Cunxiao Du, Ye Wang, and Min Lin. When attention sink emerges in language models: An empirical view. In The Thirteenth International Conference on Learning Representations , 2025. URL <https://openreview.net/forum?id=78Nn4QJTEN>.
- Michael Hahn and Navin Goyal. A theory of emergent in-context learning as implicit structure induction. arXiv preprint arXiv:2303.07971 , 2023.

- Nitish Shirish Keskar, Bryan McCann, Lav R Varshney, Caiming Xiong, and Richard Socher. Ctrl: A conditional transformer language model for controllable generation. arXiv preprint arXiv:1909.05858 , 2019.
- Muhammad Khalifa, David Wadden, Emma Strubell, Honglak Lee, Lu Wang, Iz Beltagy, and Hao Peng. Source-aware training enables knowledge attribution in language models. arXiv preprint arXiv:2404.01019 , 2024.
- Tomasz Korbak, Kejian Shi, Angelica Chen, Rasika Vinayak Bhalerao, Christopher Buckley, Jason Phang, Samuel R Bowman, and Ethan Perez. Pretraining language models with human preferences. In International Conference on Machine Learning , pp. 17506–17533. PMLR, 2023.
- Dmitrii Krasheninnikov, Egor Krasheninnikov, Bruno Mlodozieniec, Tegan Maharaj, and David Krueger. Implicit meta-learning may lead language models to trust more reliable sources. arXiv preprint arXiv:2310.15047 , 2023.
- Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.
- Matthias Minderer, Josip Djolonga, Rob Romijnders, Frances Hubis, Xiaohua Zhai, Neil Houlsby, Dustin Tran, and Mario Lucic. Revisiting the calibration of modern neural networks. Advances in neural information processing systems , 34:15682–15694, 2021.
- Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. Neurocomput. , 2024. doi: 10.1016/j.neucom.2023.127063.
- Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. arXiv preprint arXiv:2302.13971 , 2023.
- Chi Zhang, Huaping Zhong, Kuan Zhang, Chengliang Chai, Rui Wang, Xinlin Zhuang, Tianyi Bai, Qiu Jiantao, Lei Cao, Ju Fan, Ye Yuan, Guoren Wang, and Conghui He. Harnessing diversity for important data selection in pretraining large language models. In The Thirteenth International Conference on Learning Representations , 2025. URL <https://openreview.net/forum?id=bMC1t7eLRc>.
- Xiao Zhang, Miao Li, and Ji Wu. Conditional language learning with context. arXiv preprint arXiv:2406.01976 , 2024.
- Yang Zhao, Li Du, Xiao Ding, Kai Xiong, Zhouhao Sun, Shi Jun, Ting Liu, and Bing Qin. Deciphering the impact of pretraining data on large language models through machine unlearning. In Findings of the Association for Computational Linguistics: ACL 2024 , 2024. URL <https://aclanthology.org/2024.findings-acl.559/>.

A 从 D 级 PCFG 生成的数据详细信息

在本节中，我们对由 Section 2.3 中描述的过程生成的数据进行了全面解释。在训练时，带有元数据的输入序列由

$$[\text{BOS}] \quad \tilde{j}_0 \quad \tilde{j}_1 \quad \cdots \quad \tilde{j}_{D-1} \quad s_{D,0} \quad s_{D,1} \quad \cdots \quad s_{D,K_D-1} \quad [\text{EOS}]. \quad (\text{A.1})$$

给出。其中， $\tilde{j}_0, \tilde{j}_1, \dots, \tilde{j}_{D-1}$ 是有时提供元数据的标记，其他时候则成为掩码标记。具体而言，它们表示为

$$\tilde{j}_0 \quad \tilde{j}_1 \quad \cdots \quad \tilde{j}_{D-1} := \begin{cases} j_0 \quad \cdots \quad j_{D_M-1} \quad \underbrace{[\text{MASK}] \quad \cdots \quad [\text{MASK}]}_{D-D_M \text{ times}} & \text{with probability} = 0.5, \\ \underbrace{[\text{MASK}] \quad \cdots \quad [\text{MASK}]}_{D \text{ times}} & \text{with probability} = 0.5, \end{cases}$$

，其中 j_i ($i = 0, \dots, D-1$) 表示一个元数据，而 D_M 表示将元数据输入到模型的深度。

B 不同标记位置的下一个标记预测损失。

Table B.1: 不同位置的标记的下一个标记预测损失。

depth of metadata	Position of the tokens				
	0	5	10	25	50
0 (no metadata)	0.9224	0.8867	0.7244	0.7471	0.7483
1	0.9224	0.8867	0.7246	0.7474	0.7484
3	0.9224	0.8867	0.7248	0.7472	0.7479
5	0.9224	0.8866	0.7248	0.7476	0.7479

在训练过程中预先加载元数据对元数据探测精度和语法精度的影响根据提示的长度而异。自然会问，在预训练任务中是否会出现类似现象，即下一个 token 预测的性能是否会因序列中的位置而变化。有趣的是，在下一个 token 预测中没有观察到这种现象。确实，如 Table B.1 所示，为每个位置计算的损失值也显示出元数据深度之间的变化很小，就像总体损失几乎保持不变一样。

C 训练模型的校准

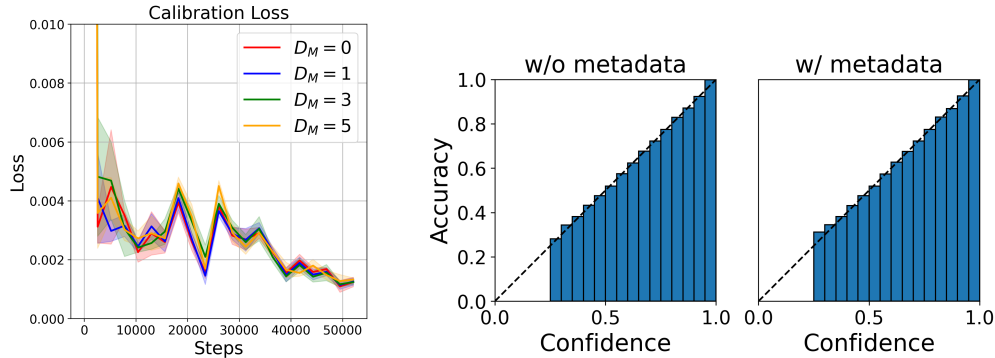


Figure C.1: 左图：测试数据（无元数据）的校准损失随预训练步数变化的情况。右图：预训练最后一步的校准直方图。

为了加强证据表明不在元数据前添加元数据时，元数据条件化不会影响预训练任务，我们调查模型是否正确捕捉了分布。为了这个目的，我们引入了称为预期校准误差的指标，该指标定义在 Minderer et al. (2021) 中。

我们在此给出度量的定义。我们首先考虑将区间 $[0,1]$ 划分为 m 个子区间： $[0, 1/m), \dots, [(m-2)/m, (m-1)/m), [(m-1)/m, 1]$ 。然后我们定义 B_i ($i = 1, \dots, m$) 为测试数据中模型置信度落在上述定义的第 i 个区间内的所有标记的集合。期望校准误差 (ECE) 定义如下：

$$\text{ECE} = \sum_{i=1}^m \frac{|B_i|}{n} \cdot |\text{accuracy}(B_i) - \text{confidence}(B_i)|.$$

，其中 $\text{accuracy}(B_i)$ 和 $\text{confidence}(B_i)$ 分别表示模型的准确率和这些标记的平均置信度。直观上，ECE 衡量的是模型的准确率与置信度之间的差异。因此，一个校准良好的模型，其预测分布准确，预期其 ECE 值应接近于零。

我们展示了 Figure C.1 左侧的期望校准误差曲线。在有和没有元数据的两种情况下，我们观察到期望校准误差均有所下降，并且校准上没有显著差异。Figure C.1 中的中间和右侧图表展示了显示每个置信区间的平均准确率的直方图。这也证实了模型的置信度与其准确性是一致的，并且在训练过程中存在的元数据对模型校准影响不大，与测试损失的情况相同。

D 元数据探测的更多细节

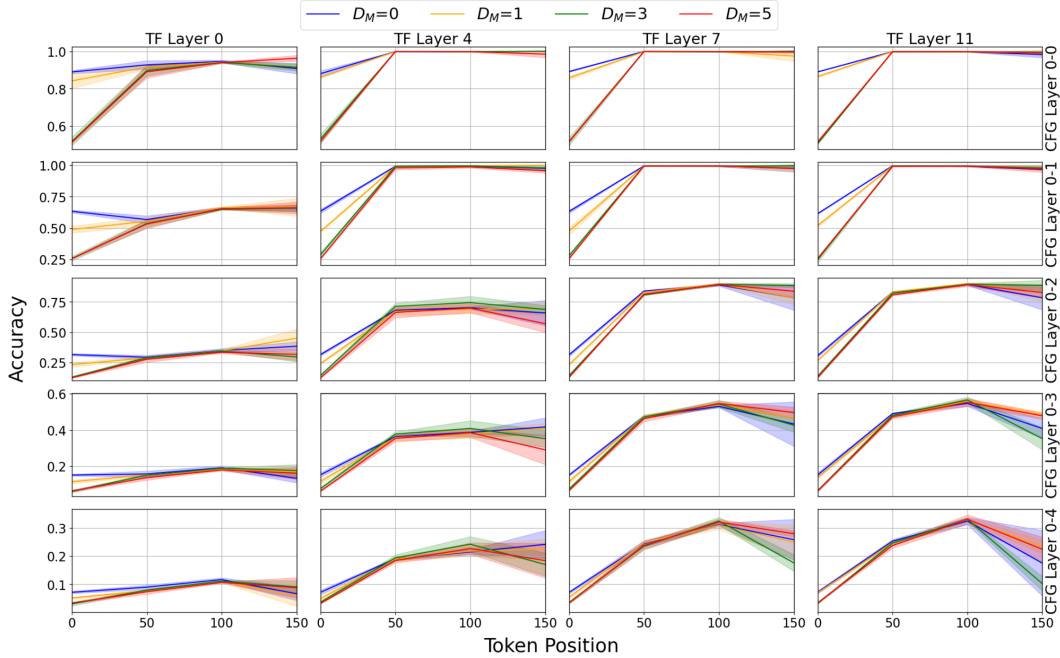


Figure D.1: 补充结果为与 Figure 3.2 相关的元数据探测准确性。我们以提示长度为基准绘制了过渡图。每一列对应一个 Transformer 层索引 (1, 5, 8, 12)，每一行对应 CFG 树的深度。 d 行 ($d = 1, 2, 3, 4, 5$) 表示从第 0 层到第 $d-1$ 层分类 CFG 树分支的准确率。从第 0 层到第 $d-1$ 层正确预测元数据的概率是 $1/2^d$ 。

为了训练和评估分类器，我们使用了与预训练中不同的 10,000 个数据点。具体来说，7,000 个数据点用于训练，1,500 个用于早停验证，其余 $n = 1,500$ 用于评估。

正如我们在 Section 3.2 中提到的，我们在各个层上进行探测实验。我们在主论文中省略了完整的结果，并在 Figure D.1 中展示。我们可以观察到，除第 12 层外，其他层的结果与 Section 3.2 中提供的结果有相似的趋势。