# Ankh ☥：优化后的蛋白质语言模型开启通用建模

Ahmed Elnaggar[1,2*]、Hazem Essam[2],Wafaa Salah-Eldin[2],Walid Moustafa[2], Mohamed Elkerdawy[2],Charlotte Rochereau[3] 和 Burkhard Rost[1]

[1*]慕尼黑工业大学（TUM）。[2] Proteinea公司[3]哥伦比亚大学。

*通讯作者。电子邮箱：ahmed.elnaggar@tum.de；贡献作者：hazem@proteinea.com；wafaa@proteinea.com；walid@proteinea.com；elkerdawy@proteinea.com；cr3007@columbia.edu；assistant@rostlab.org；

**摘要**

与扩大蛋白质语言模型（蛋白质语言模型）的规模不同，我们致力于通过蛋白质特异性优化来提升性能。尽管语言模型规模与其学习到的表示丰富度之间的正相关关系已得到验证，我们仍优先考虑可及性，并寻求一条数据高效、成本较低且知识引导的优化路径。通过开展二十多项涵盖掩码、架构和预训练数据的实验，我们从蛋白质特异性实验中汲取见解，以构建能够最佳诠释生命语言的模型。我们推出 Ankh，这是首个在 Google的TPU-v4 上训练、性能超越当前最先进水平的通用蛋白质语言模型，且所需参数更少（预训练减少 <10%，推理减少 <7%，嵌入维度减少 <30%）。我们提供了一系列具有代表性的结构和功能基准测试，Ankh 在这些测试中表现出色。此外，我们还针对 High-N 和 One- N 输入数据规模进行了蛋白质变体生成分析，结果表明 Ankh 成功学习了蛋白质进化保守-突变趋势，并在保留关键结构-功能特征的同时引入了功能多样性。我们致力于通过人人可及的资源，推动研究创新的普及。

**Key词语：**蛋白质、语言模型、Transformer、深度学习、高性能计算

## 1 引言

自然语言的句法-语义与蛋白质的序列-功能之间的类比，彻底改变了人类研究生命语言的方式 [1–8]。尽管将这一类比视为历史上的先行步骤——它促成了自然语言处理领域的进展（例如语言模型）被应用于蛋白质领域——本身具有重要价值，但该领域的结论

自然语言处理（NLP）不应被完全照搬到蛋白质语言领域。人们不仅追求自然语言处理模型的规模，甚至有人提出，扩大蛋白质语言模型的规模可能比扩大自然语言模型的规模更具影响力 [9]。观察发现，使用大量参数、经过大量训练步骤的语言模型仍在持续学习并呈现出显著的学习梯度，因此被认为尚未充分拟合；而模型规模与其学习到的表征丰富度之间的正比关系，实际上是被这种现象——错误地——强化了

。 [3, 9, 10]因此，人们逐渐不再追求更有意义的蛋白质表征或更准确的建模，转而追求更大的模型，并相应地投入更多计算能力——这降低了可及性。值得注意的是，蛋白质语言模型的规模近期已从 $\sim 10^6$ [4] 跃升至 $\sim 10^9$ [10]参数。为按时间顺序梳理蛋白质语言模型的最新进展，我们以 ProtTrans 的 ProtT5-XL-U50 作为规模与性能基准。该模型是一个在 UniRef50 数据库上预训练的编码器-解码器 Transformer，训练时参数量为 3B，推理时为 1.5B [3]。随后，研究者通过 RITA 展示了模型性能随规模变化的趋势。RITA 是一系列语言模型，率先探索建立蛋白质序列建模的规模化原则。该系列包含 4 种不同模型，性能随规模提升而提高，参数量分别为 85M、300M、680M 和 1.2B [9]。随后，ProGen2 进一步印证了这一趋势。它是一套在不同序列数据集上训练的蛋白质语言模型，参数量最高扩展至 6.4B [11]。最后，截至本文发表时，推动模型扩展的最新成果是 ESM-2。这是一组通用型蛋白质语言模型，同样展示了性能随规模提升而提高的趋势，参数量从 650M 增至 3B，再到 15B [10]。将更大规模与看似更优的蛋白质语言模型简单关联起来，忽略了诸多方面，包括计算成本、任务无关的模型设计和实现。这提高了研究创新的门槛，并将其局限于可扩展性。毫无疑问，模型规模是追求上述目标时影响重大的属性，但并非唯一属性。沿着扩大预训练数据集这一方向的研究已证明，这种做法是有条件的（即，规模更大的数据集未必优于质量更高的小型数据集） [3]。我们在此基础上提出，扩大语言模型的规模同样是有条件的（即，规模更大的模型未必优于通过蛋白质知识引导的优化方法训练的小型模型）。本研究的主要目标是将知识引导的优化融入迭代式实证框架，以可获得的资源促进研究创新的可及性。我们将本文命名为 “Ankh”（即古埃及象征生命之钥的符号）这一名称，借喻我们的模型通过学习构成生命语言的“字母”——氨基酸的优越表征，来“解锁”生命语言。评估 Ankh 的优化能力和通用性时，我们从两个方面提供证据。首先，在具有代表性的结构和功能基准测试中，其表现超越了当前最先进水平；此外，我们还分析了其在 High-N（基于家族）和 One-N（基于单条序列）应用中的蛋白质工程生成能力，其中 N 指输入序列的数量。其次，这一性能得益于一系列优化属性，涵盖的不仅是模型设计，还包括其开发、训练和部署所用的软件与硬件。我们提供了两个预训练模型，分别称为 $Ankh\ large$ 和 $Ankh\ base$，可根据应用需求选择两种计算模式。为方便起见，我们将主模型 $Ankh\ large$ 称为 $Ankh$。

## 2 结果

### 2.1 极致计算效率与极致性能

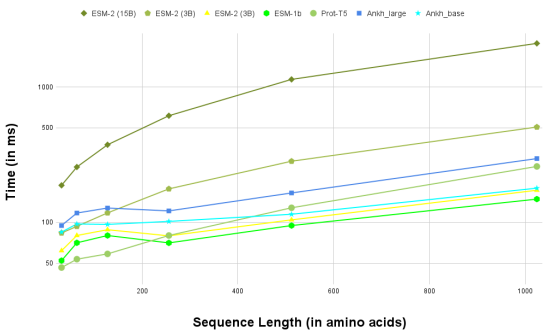
我们主张将性能与规模之间的相关性视为一种权衡，并将计算能力也纳入考量。平均而言，$Ankh$ 将蛋白质语言模型的最先进性能提升了 4.8%，而$Ankh\ base$ 则提升了 3.4%，其 <10训练参数量和嵌入维度分别仅为 $Ankh$ 和 $Ankh\ base$的 % 和 3%，以及 30% 和 15% -(图 1）。由于特征提取是后续任何建模工作的基础，我们比较了提取不同长度序列特征所需的时间（以毫秒计），序列长度逐渐增加，最长达 1024 个残基（图1）。尽管 Ankh 模型支持的序列长度甚至超过预定义相对位置嵌入维度的最大长度，我们仍将上限设为 1024，以适配 ESM-1b 模型支持的最大长度。我们可以看到，对于$Ankh\ base$ 和 $Ankh$，ESM-2 (15B) 的特征提取时间分别至少是其 2.2 倍和 2.0 倍，最多是其 11.7 倍和 7.1 倍。




(a)




(b)




(c)

**图1**： **性能-规模权衡比较：** (a) 中，我们在横轴上绘制不同蛋白质语言模型的参数数量，在纵轴上绘制七项不同基准测试任务的性能得分均值和中位数。(b) 中，我们在横轴上绘制所研究模型的嵌入维度，纵轴与 (a) 相同。在图 (c) 中，我们在横轴上绘制不断增加的序列长度（最高达 1024 个氨基酸），并在纵轴上绘制所有受研究模型对应的特征提取时间（毫秒）。

在存储方面，使用 ESM-2 (15B) 提取特征需要四块 A100 80GB Tensor Core GPU，而每个 Ankh 模型只需一块 A100 40 GB Tensor Core GPU。这些计算属性表明，除了在平均和中位数下游性能方面表现出色外，Ankh 对计算资源的需求也显著更易满足。对于表1中报告的蛋白质基准测试任务结果，我们从所有受测模型的最后一层隐藏状态中提取各数据集蛋白质序列的上下文化嵌入。鉴于我们致力于推动计算优化，本研究提倡将嵌入提取而非注意力提取用作迁移学习的手段。因此，我们围绕基于嵌入的预测对实验进行了优化。然而，由于在 ESM 蛋白质语言模型系列中，注意力图被认为是接触预测更好的指标（见 [2] 和 [10], ），我们分别测试了这两种表征。具体而言，我们为每个模型提取注意力图，并将其与上下文化嵌入分别作为接触预测任务的输入进行比较，以确保公平比较，并展示我们认为最佳的指标与最先进的最佳指标之间的表现。事实上，结果显示，基于嵌入的预测显著优于基于注意力的预测（基于注意力的预测的完整结果见表13）。如图1所示，Ankh套件始终优于其他受测模型。我们还发现，ESM-2 (15B) 并未在所有任务中都胜过 ESM 系列中规模较小的模型，而且在不同运行中结果不一致，这证实了我们的假设：更大的模型并不一定在所有蛋白质建模任务中都更好，模型规模大也会带来自身的挑战。

## 2.2 让蛋白质生成成为可能

### 2.2.1 用于高N生成的自回归微调

我们提出了一种自回归微调生成框架，适用于 High-N（基于蛋白质家族的生成）规模，因为它提供了一种易于使用的

表 1：结果摘要。

| 任务数据集 | | Ankh | Ankh 基础版 | ProtT5-XL-U50 | ESM-1b | ESM-2 (6.5亿) | ESM-2 (3B) | ESM-2 (150亿) |
|---|---|---|---|---|---|---|---|---|
| SSP | *CASP12* [41] | **83.8**±3% | 80.8±4% | 83.4±4% | 79.6±4% | 82.3±4% | 83.3±4% | 83.2±3% |
| | *CASP14* [42] | **77.6**±3% | 76.8±3% | 74.1±3% | 75.1±4% | 77.0±3% | 76.8±3% | 76.8±4% |
| CP | *ProteinNet L/1* [34] | **49.0**±8% | 43.2±8% | 44.7±8% | 30.0±6% | 29.6±6% | 30.7±6% | 33.3±6% |
| | *ProteinNet L/5* | **73.2**±11% | 66.6±11% | 69.2±11% | 50.1±10% | 50.2±10% | 52.7±10% | 54.7±10% |
| | *CASP14 L/1* | **30.2**±8% | 28.8±7% | 26.9±7% | 24.6±6% | 25.0±6% | 24.8±7% | 25.9±7% |
| | *CASP14 L/5* | **50.7**±11% | 48.0±11% | 42.4±14% | 40.0±11% | 38.4±13% | 41.9±14% | 40.4±15% |
| EAT | | 71.7±6% | **74.8**±6% | 71.0±6% | 64.5±7% | 55.5±7% | 65.6±6% | 65.4±7% |
| FolP | | **61.1**±4% | 58.8±4% | 57.6±4% | 57.6±4% | 56.3±4% | 60.5±4% | 56.7±4% |
| FluP | | **0.62**±0.004 | 0。61±0.004 | 0.58±0.004 | 0.5±0.005 | 0.48±0.005 | 0.48±0.005 | 0.55±0.004 |
| SolP | | **76.4**±2% | 74.2±2% | 74.4±2% | 67.3±2% | 75.0±2% | 74.9±2% | 60.4±2% |
| GB1P | | 0.84±0.008 | **0。85**±0.008 | 0.78±0.01 | 0.81±0.009 | 0.82±0.009 | 0.81±0.009 | 0.57±0.02 |
| LocP | | **83.2**±2% | 81.4±2% | **83.2**±2% | 80.0±2% | 81.8±2% | 82.4±2% | 81.8±2% |

**基准测试结果摘要，涉及** *Ankh* **和** *Ankh base*，**使用 ProtT5-XL-U50、ESM- 1b、ESM-2（6.5亿）、ESM-2（30亿）、ESM-2 (15B)**：对于回归任务，我们报告斯皮尔曼相关系数；对于所有分类任务，我们报告准确率分数，但接触预测除外，在该任务中我们报告 L/1 和 L/5 精确率。在 EAT 中，报告的分数是四项注释（类别、架构、拓扑和同源超家族）的准确率分数平均值。任务缩写：SSP：二级结构预测；CP：接触预测；EAT：基于嵌入的注释迁移；FolP：折叠预测；FluP：荧光预测；SolP：溶解度预测；GB1P：GB1适应度预测；LocP：定位预测

蛋白质变体生成方法，且能够轻松扩展至不同的蛋白质家族。此外，通过调整 logit 扭曲温度（自然语言处理模型用来提高或降低模型对最可能响应的置信度的参数），该框架可以轻松控制生成过程中的探索-利用权衡[12]。为验证该框架并选出最佳温度范围，我们使用同一个模型，在苹果酸脱氢酶（MDH）的天然变体上进行微调，并以三种不同温度（1.0、1.5 和 2.0）生成了三组初始序列，每组 500 条。对于这三组序列，我们报告了生成序列集与微调数据的多序列比对（MSA）之间的香农熵变化。香农熵旨在通过比较氨基酸变异的代表性统计数据，刻画生成序列对天然数据集进化特性的保留程度。熵值较低反映出决定功能得以保留的保守区域，而熵值较高则反映出保守性较低、突变率较高的区域。如图 2(a) 所示，我们可以观察到，生成的三组序列与该分布高度相似

天然序列的两个峰和谷的位置几乎完全一致。天然序列集与生成集合之间熵值的均方误差（MSE）分别为 0.1、0.09 和 0.08，对应的生成温度分别为 1.0、1,5 和 2.0。需要强调的是，所报告的相似性基于包含 500 条序列的生成集合计算得出，其规模不到天然序列集（16,706 条）的 3%。换言之，该模型仅使用原始序列中的一小部分，就能模拟微调数据集的分布。为进一步研究不同温度对探索—权衡的影响，我们重点分析温度为 1.0 和 2.0 时生成的序列，因为如观察结果所示，这两种温度与引入功能多样性并保持保守功能区域直接相关。在此基础上，我们首先比较生成序列与微调数据集之间基于全局比对的一致性。我们通过 BioPython 的 pairwise2 模块计算一致性，在该模块中获得生成的变体（gen）与原始变体（nat）之间的全局比对 [13]。我们可以在图2 （d）中观察到，生成的

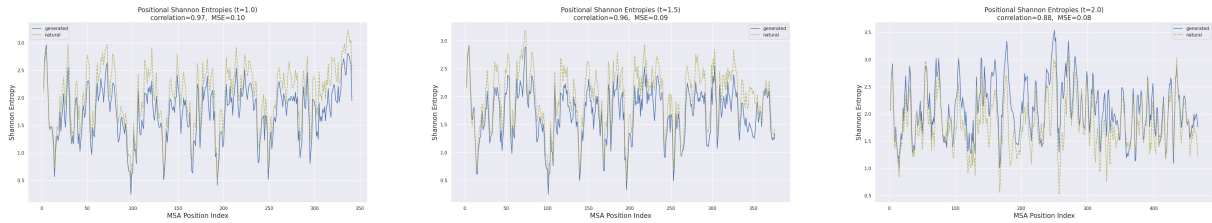
序列涵盖了广泛的基于全局比对的一致性评分范围；在温度为 1.0 和 2.0 时生成的序列，其一致性分别低至 70% 和 55%。此外，超过 95% 的生成变体是唯一的（即只有 5% 的变体与微调数据集中的序列重复）。此外，我们报告了每个集合的内部一致性，即获得每个集合内部序列之间的全局比对。我们可以在图2 （e）中观察到，温度为 1.0 时的生成集合比天然序列具有更低的内部变异性，而温度为 2.0 时的生成集合则表现出更高的内部变异性。因此，温度为 1.0 的生成倾向于更加保守，更偏向于微调数据集中占主导地位的簇。另一方面，温度为 2.0 的生成倾向于不那么保守，能够覆盖稀有簇，并在生成结果中呈现出更高的多样性。这使用户能够根据微调蛋白质家族的性质，以及生成全局变体还是局部变体的需求，对生成过程进行控制。通过比较具有已知结构注释的天然结构域和生成结构域，我们观察到，天然变体中的 CATH结构域主要属于三个同源超家族。在所有生成集合中，都有相当大比例的生成序列包含来自这三个主要超家族的结构域，如图2（f）所示，并在表15中以数值形式详细列出。我们进一步研究具有已知功能分类的结构域，以比较生成序列的功能多样性。天然结构域中，只有三个同源超家族中的两个（3.90.110.10 和 3.40.50.720）标注了功能家族编号。属于 3.90.110.10 超家族且具有功能注释的 CATH结构域如图2 （g）所示。所有生成集合都包含属于常见功能家族编号 2、11 和 3 的结构域。然而，属于稀有家族编号 1 的结构域（在天然序列集中仅出现 6 次）只能在温度为 2 时生成的序列中观察到（在训练轮次 1 和 2 中分别出现 155 次和 95 次）。如表16所示，属于 3.40.50.720 超家族且具有功能注释的结构域也保持了相同的分布趋势

### 2.2.2 用于 One-N 生成的掩码语言建模（MLM）

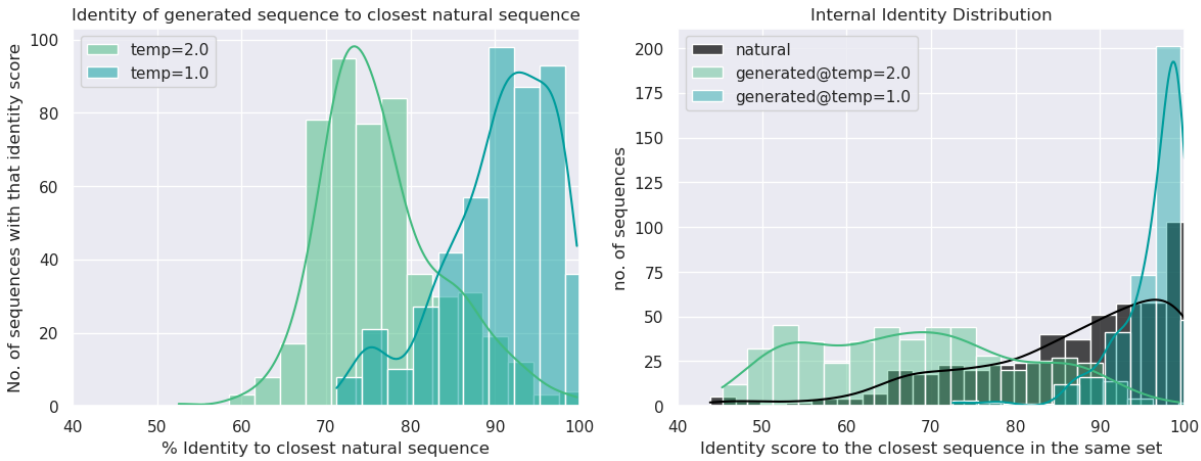
由于我们在单次（单序列生成）规模上采用掩码语言建模（MLM）生成框架，因此无法针对特定数据集评估生成序列。相反，我们评估其保留实验三维结构及相应功能特征的能力。为此，我们使用 ColabFold 的 $colabfold\ batch$ ，通过 2 个模型和 3 个循环预测生成变体的三维结构 [14]。对于每条生成序列，我们绘制其与原始未掩码序列的序列一致性，以及其预测结构与原始序列实验三维结构之间的均方根偏差（RMSD）。我们通过 Cα原子，利用 BioPython 的 SVDSuperimposer 计算 RMSD[13]。理想情况下，我们希望模型在改变序法的同时保留序列的语义。换言之，我们希望获得序列一致性较低且 RMSD 也较低的变体。为检验这一假设，我们利用输入数据集的两种掩码概率（40% 和 50%）生成了 179 个合成变体。的确，从图 3中可以看到，该模型生成了许多序列相似性较低、同时与输入序列保持较高结构相似性的序列。在图中，我们可以注意到，由 50% 掩码概率生成的序列保持了更陡的斜率，这意味着未掩码上下文越大，生成的变体越好，这也符合预期。

### 2.2.3 知识引导的优化实践

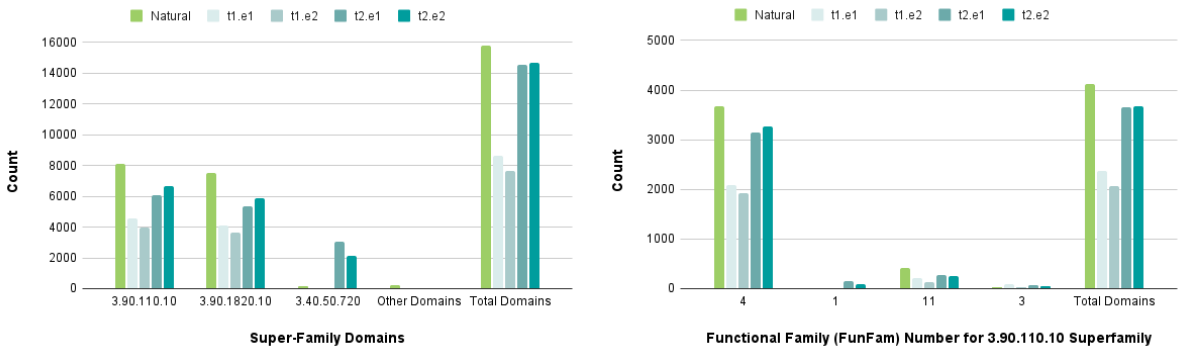
为了实现所展示的极高计算效率，同时确保顶尖的下游性能，我们在模型设计之前进行了知识引导的实验。我们将知识引导的实验定义为蛋白质特异性实验，即每次只保留一个自变量，依次考察掩码（策略和概率）、架构以及预训练数据集。设计前的这些实验对每种变体均训练 2 个训练轮次，同时确保每次实验的参数总量大致相同，以避免复杂度偏置。实验的定义




(a) *At temperature* = 1.0 (b) *At temperature* = 1.5 (c) *At temperature* = 2.0

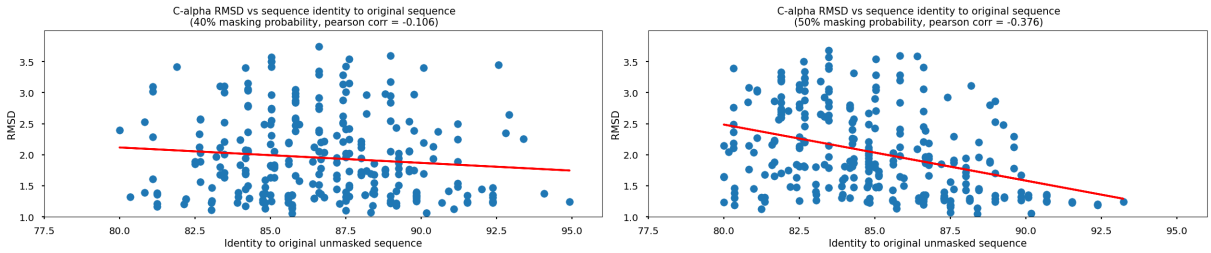



(d) (e)




(f) (g)

**图 2**： **MDH变体的自回归微调生成：** (a)、(b) 和 (c) 分别表示在 logit 扭曲温度为 1.0、1.5 和 2.0 时，天然序列与生成集合的 MSA 之间的香农熵曲线：生成过程中对温度的调节会影响生成序列的变异性。(d) 温度为 1.0 和 2.0 时生成的变体与天然变体之间的序列一致性分布：生成序列涵盖了广泛的序列一致性评分，最低可达 60%。(e) 天然序列/微调集合与温度为 1.0 和 2.0 的生成集合之间的内部一致性：生成的变体涵盖了广泛的变异范围，其中温度为 2.0 的集合显示出最宽的范围。(f) 天然序列集中三个占主导地位的同源超家族的 CATH结构域分布：所有生成的变体都保留了来自这三个占主导地位的超家族的 CATH结构域。(g) 天然序列集与生成集合中，在 3.90.110.10 超家族（负责脱氢酶功能）内具有已知功能注释的 CATH结构域分布：生成集合成功保留了天然序列集中三个功能家族的结构域，同时扩展至功能家族编号 1——麦芽糖脱氢酶（MDH）的结构域；该功能家族在天然序列集中仅有 6 个示例。缩写：Nat：天然序列；t1.e1：第一轮训练在温度 1 下生成的序列；t1.e2：第二轮训练在温度 1 下生成的序列；t2.e1：第一轮训练在温度 2 下生成的序列；t2.e2：第二轮训练在温度 2 下生成的序列




(a) 掩码概率为 = 40%时的序列一致性与均方根偏差　　(b) 掩码概率为 = 50%时的序列一致性与均方根偏差

**图 3**：**掩码概率为 40%（图 3.a.）和 50%（图 3.b.）时生成序列的序列一致性与结构相似性**：图中显示，在两种情况下，模型都能够生成均方根偏差低于 1.5 A、序列一致性低至 80% 的序列。掩码概率为= 50%时生成的序列表现出更强的负皮尔逊相关性，这表明更多的序列上下文有助于生成序列一致性较低但结构相似的序列。

自变量、其取值、评估指标和实验基线可在方法部分找到。最终模型的设计整合了每组自变量实验及子实验中表现最佳的版本。我们对两个模型进行了完整的预训练，*Ankh* 和 *Ankh base*，它们可在图 4中查看。掩码策略实验结果支持采用 1-gram跨度去掩码策略（详见 实验 *4*）。关于掩码概率，我们根据 *Exp. 8*的建议采用了 20%。根据实验 *11*的结果，层数方面，编码器选用 48 层，解码器选用 24 层。关于激活函数，我们根据 实验 *14* 和 实验 *15*的发现采用了 Gated-GELU。根据 实验 *20*的结果，我们采用了相对位置嵌入，嵌入偏移量为 128，嵌入维度为 64。不过，两个模型仅在嵌入维度、注意力头数量和前馈层数量上有所不同。 *Ankh base* 其嵌入维度为 768，这是 实验 *13* 的建议，而 *Ankh* 的嵌入维度为 1536，因为我们发现维度加倍通常能带来更好的性能，这一点已由实验 *20*验证。此外，*Ankh base* 和*Ankh* 的注意力头数量分别为 12 和 16。最后，*Ankh base* 的前馈层数量为 3072，*Ankh* 的前馈层数量为 3840。两个模型的完整配置见表11。

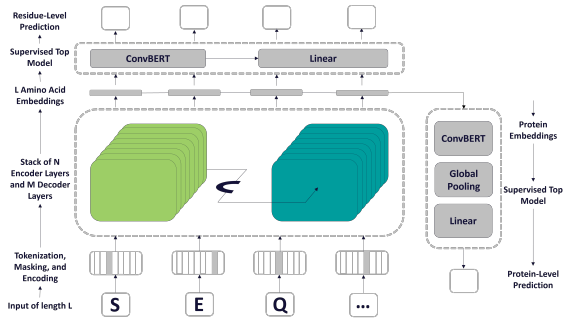



**图 4**：**Ankh 模型的架构**：箭头表示网络中的信息流，从输入序列、Transformer 输入预处理、Transformer 开始，然后流向残基级预测网络或蛋白质级预测网络；二者的区别仅在于后者前面多了一个全局最大池化层。*Ankh base* 和 *Ankh large* 均认同所示架构。不过，它们的上下文向量长度不同。

# 3 讨论

**结果摘要——更高效的模型也能更好地泛化：**为全面分析我们的模型，我们确保对 Ankh 进行了评估，涵盖深度学习蛋白质建模中最重要、最有价值的类别。从深度学习的角度来看，我们在回归、分类和生成任务上测试了模型。从生物学的角度来看，我们在残基级和蛋白质层面进行了测试，测量的属性涵盖结构和功能。从规模和稀疏性的角度来看，我们在 High-N 和 One-N 两种规模下都测试了模型。在所有这些

基准测试中，Ankh 无一例外地超越了最先进水平；而我们的基础模型 Ankh 基础版则以显著更低的计算能力达到相同或非常接近的表现，从而可根据应用需求提供两种计算模式。在生成分析中，Ankh 能够学习进化保守性与突变趋势，并在 High-N 和 One-N 两种规模下引入多样性，同时保留关键的结构和功能特征。

**结果解读：** 我们的结果进一步证实了蛋白质语言模型与生命语言的兼容性，同时也凸显了蛋白质知识引导的模型设计的重要性。此外，我们的结果强化了将序列上下文嵌入仅作为下游模型输入的做法，同时揭示了任务特定架构/层的作用。

**结果启示：** 我们的结果表明，只需显著降低计算能力投入，就可以达到并超越当前最先进性能。这一发现意味着，有必要重新审视模型性能与所需计算能力之间受到强调的相关性，而这种计算能力既体现在模型规模上，也体现在数据规模上。相反，我们的工作建议将这种相关性视为一种权衡，并强调直接扩大模型/数据规模以提升模型性能所需付出的巨大成本。在权衡的另一端，我们提出了知识引导的优化方法，其前提不仅包括所需的蛋白质知识，还包括对模型生命周期中的软件和硬件组件进行进一步优化。

**结果局限性：** 我们报告了本研究的若干局限。首先，激活函数的变化会影响编码器和解码器的最佳层数，以及嵌入维度。不过，这一设置源于 Gated-GELU 激活函数的特性：它需要显著更多的可训练参数，因此我们不得不通过降低深度来补偿宽度的增加，以保持可训练参数总数不变，并避免任何可能的复杂度偏差 [15]。因此，要测试不同维度组合，就需要使用参数需求更少的激活函数。其次，我们的优化流程会将每项预测任务中表现最佳的模型版本推进到下一项任务。最佳表现之所以没有采用在所有任务中数值表现都最优的版本，是因为不存在这样的版本。相反，我们选择了在大多数最常用数据集上优于其他版本、且在多种目标上表现最佳的版本。这种看似简单的算法有一个缺点：我们未能定义一个能够从众多此类公式中脱颖而出、用于计算“最佳”结果的统一公式。具体而言，我们记录了为每项预测任务选择表现最佳版本的理由。总体上，当不同结果之间的数值差异微乎其微时，我们倾向于通过通用语言模型进行整体预测，并以此为选择提供依据；同时，我们也考虑到任务特定定制的影响，以及所有实验版本都只训练了两个训练轮次这一事实。

**建议：** 我们的结果含蓄地表明，用于预训练蛋白质语言模型的数据选择，可能需要与用于测试后续蛋白质预测任务的数据集选择相协调。尽管这一努力超出了本工作的范围，我们仍鼓励开展相关工作。例如，我们报告了使用 UniRef50 进行预训练优于使用 UniRef90、UniRef100 和 BFD，这得益于其更低的冗余度。尽管冗余的具体构成取决于应用场景（例如，在研究人类蛋白质的长度分布时，使用所有可用的人类蛋白质所造成的冗余，比尝试预测其中的结合残基时更低），但过高的冗余通常很容易被发现 [16]。此外，我们还揭示了纳入合成生成且经实验表征的序列的价值。

**未来工作：** 我们将 Ankh 作为优化后的通用蛋白质语言模型的初始版本推出。此版本旨在作为预训练基础模型，之后将在未来的工作中通过任务特定优化和详细分析，将其专门用于影响重大、价值高的蛋白质建模任务（例如全原子分辨率的三维结构预测、蛋白质生成等）。

# 4 方法

## 4.1 自监督学习

### 4.1.1 预训练数据集

基于深度学习的蛋白质建模与自然语言处理一样，都是数据驱动的。然而，对数据的渴求正日益受到限制，同时也带来了限制。我们借鉴了 ProtTrans，尤其是 Prot-T5 中的先前实验成果；这些实验分析了三个在规模、基于序列一致性的聚类方式和来源方面各不相同的数据集的性能及相关计算能力。分析结果表明，使用 UniRef50[17] 而非 UniRef 100 [17] 和 BFD [3, 18]更为合适。我们基于相同的结果，使用 UniRef50 对基线进行预训练。

UniRef（UniProt参考簇）数据库提供了对 UniProtKB 序列（包括异构体）和部分 UniParc 记录进行不同粒度聚类的功能[17]。这种可变聚类表示采用了不同的序列相似性阈值，以实现非冗余性和簇内同质性。在 UniRef100 中，单个簇/条目表示来自任何生物体的完全相同序列及其子片段。为构建 UniRef90 和 UniRef50，UniRef100 序列分别按 90% 和 50% 的序列一致性阈值进行聚类[17]。因此，ProtTrans 中的结果可以归因于 UniRef 中更丰富的多样性和代表性 50 [3]。不过，我们希望测试一个介于 50% 和 100% 阈值之间、且 ProtTrans 尚未测试过的中间值，因此在后续实验中，将预训练数据集作为独立变量之一，并选择 UniRef90。预训练数据的统计信息见表2 。UniRef50 和 UniRef90 中的序列都以单个空格分隔各词元进行标记，类似于词语边界；每条序列单独存储在一行中，类似于句子。

#### 4.1.2 预训练模型：编码器-解码器 Transformer

对于基线模型以及整个实验过程，我们采用了最初为机器翻译提出、随后用于将任意输入域映射到目标域的编码器-解码器 Transformer [19]。编码器学习将输入域序列投影到潜在／嵌入空间，以表示“上下文”。而解码器则学习在给定该上下文的情况下生成目标域序列。尽管后续的 Transformer 版本放弃了编码器-解码器组合，只采用其中一种架构，但我们借鉴了 ProtTrans 的实验结果，该结果表明 T5 [20] (唯一接受分析的编码器-解码器 Transformer)优于仅使用编码器的 Transformer（例如 BERT [21], ALBERT [22],和 Electra [23]）以及仅使用解码器的 Transformer（例如 TransformerXL [24] 和 XLNet [25]）。不过，本研究选择编码器-解码器 Transformer，不仅是因为其如上所述的出色性能，还因为它与我们的实验自变量（包括但不限于掩码和架构）相兼容。由于同时保留编码器和解码器，类 T5 Transformer 能更好地兼容依赖于掩码和反掩码技术的不同掩码策略。此外，由于它们学习了在各层之间共享、针对每个注意力头的相对位置嵌入，类 T5 Transformer 能够稳健地处理超出相对位置嵌入维度最大长度的预测，因为它们能够学习组合较低层氨基酸子集之间的相对偏移量 [20]。基线采用的掩码策略为 1-gram随机令牌掩码，概率遵循默认值 15%（即随机掩盖 15% 的序列令牌），并对序列进行完整的反掩码／重建（即将所有词元作为单独的词元进行重建）。编码器-解码器层数方面，编码器和解码器各使用 36 层。激活函数采用门控高斯误差线性单元（Gated-GELU） [15]。相对位置嵌入方面，我们使用了 32 的嵌入维度和 128 的嵌入偏移量。最后，我们在 UniRef50 上对基线模型进行了预训练。基线模型的其余配置见表 11。我们的基线模型代表实验 *0* ，之后在每一类实验中，每次仅改变一个变量。

## 4.2 下游任务

为了全面分析 Ankh 的表现——无论是相对表现还是最终表现——我们开展了

一项蛋白质下游基准测试，并针对蛋白质工程应用开展了生成分析。对于蛋白质下游基准测试，我们通过选定的下游任务评估 Ankh 的表现，并将其与已报告的顶尖蛋白质语言模型进行比较。这些任务涵盖蛋白质理解的多个方面，并涉及蛋白质结构与功能。对于生成分析，我们通过两种生成框架，在两种数据规模下分析 Ankh 的使用情况：High-N（基于家族）生成和单次（基于单序列）生成，考察保守性-突变趋势、在保持结构（进而保持功能）序列一致性的同时引入多样性，以及稀疏性鲁棒性。我们统一了本研究所考察的所有下游任务的实验训练和测试设置及流程。尽管我们承认任务特定优化的作用，但这种统一旨在专门比较所研究模型生成的蛋白质表征所体现的蛋白质理解水平，同时避免因任务特定优化方式而产生偏差。图 5 展示了所采用下游任务的示例。以下各小节将介绍这些任务、任务数据库，以及所使用的预测模型和设置。

### 4.2.1 任务与数据集

Transformer 作为强大的蛋白质自监督模型的兴起，促使人们投入大量精力，设计全面的蛋白质序列理解基准数据库，例如 TAPE [26], 和 PEER [27]。这些工作对于推动 Transformer 在蛋白质理解领域的发展至关重要，正如全面的基准测试推动了 Transformer 在自然语言理解领域的发展一样。我们在表 12中汇总了下游数据集的统计信息。我们选取了一组常用的下游任务，分为三类：蛋白质功能预测、蛋白质结构预测和蛋白质定位预测。我们还为部分下游任务添加了独立测试数据库，以进一步验证比较结果。除了基准测试任务外，我们还研究了 Ankh 的蛋白质变体生成能力，涵盖两种设置：基于蛋白质家族的生成和基于单个蛋白质的生成。

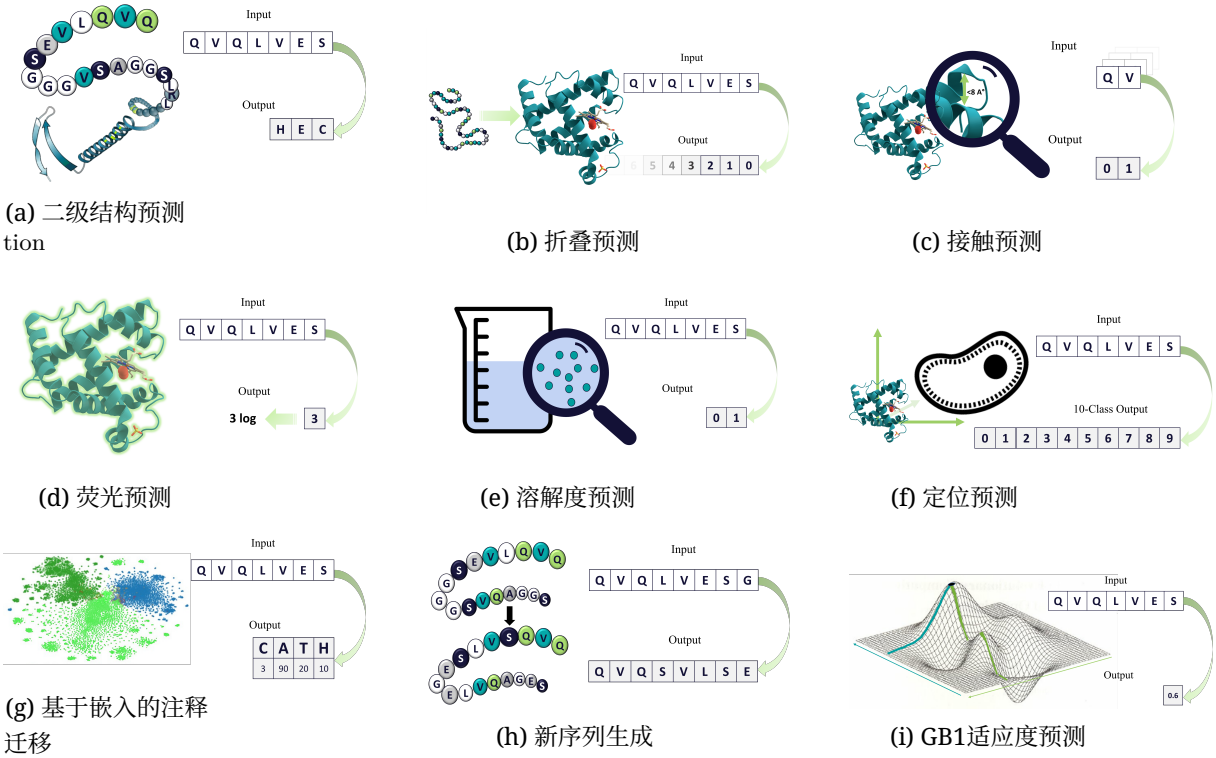
### 4.2.2 蛋白质功能预测

这组任务旨在评估蛋白质嵌入捕捉蛋白质工程两项关键设计参数——荧光和溶解度——功能评分的能力。

**荧光预测（FluP）：** 这项回归任务用于评估 Sarkisyan 等人 [28]标注的绿色荧光蛋白突变体的荧光强度。荧光是一项重要的生物功能，因为它使研究人员能够推断蛋白质是否存在于细胞系和活体生物体中 [27]。预测突变对绿色荧光蛋白的影响，是蛋白质工程研究中探究基因型/表型适应度景观预测问题的常见范例。我们采用与 TAPE [26]相同的数据划分，因为这些划分旨在测试模型从低阶突变到高阶突变的泛化能力。训练集和评估数据集仅包含具有三个或更少突变的突变体，而测试集则包含具有四个或更多突变的突变体。

**溶解度预测 (SolP)：** 此分类任务评估一组彼此差异较大的蛋白质是否可溶的二元标签。溶解度是有效蛋白质设计中不可或缺的参数，尤其是对于药物而言 [29]。我们采用了 DeepSol 开发过程中使用的溶解度数据库 [30] ，并使用相同的数据集划分；其中，凡是与测试子集中任一蛋白质的序列一致性达到 $\geq$ 30% 的蛋白质，都会从训练子集和评估子集中移除。

**GB1适应度预测（GB1）：** 这项回归任务评估了在 FLIP基准 [31] 中整理的 4 个特定位置发生突变后，GB1 的结合适应度。GB1 是免疫球蛋白结合蛋白G的结合结构域，可用于抗体纯化。GB1景观被视为研究突变非加性交互作用（称为上位性 [32]）的金标准。与荧光突变体不同，GB1突变体仅涉及四个位置，因此被选作蛋白质工程研究中的另一种常见案例。在 FLIP 中，测得结合评分的 149,361 个 GB1突变体被下采样至 8,733 个，因为原始数据中有 96%




(a) 二级结构预测 (b) 折叠预测 (c) 接触预测

tion

(d) 荧光预测 (e) 溶解度预测 (f) 定位预测

(g) 基于嵌入的注释迁移 (h) 新序列生成 (i) GB1适应度预测

**图 5**：结构与功能基准测试示例：我们展示了所有采用的下游任务。(a) 在 SSP 中，输入为蛋白质序列，输出为逐残基分类，涵盖 3 种或 8 种状态。(b) 在 FolP 中，输入为蛋白质序列，输出为逐蛋白质分类，涵盖 1194 种可能的折叠。在接触预测中，输入为蛋白质序列，处理方式为残基对，输出为二元分类，指示指定残基是否发生接触。(d) 在 FluP 中，输入为蛋白质序列，输出为表示荧光强度的回归分数。(e) 在 SolP 中，输入为蛋白质序列，输出为二元分类，指示该蛋白质是否可溶。(f) 在 LocP 中，输入为蛋白质序列，输出为逐蛋白质分类，涵盖 10 个类别。(g) 在 EAT 中，输入为已注释查询蛋白质，输出为从已注释查找数据集中最佳匹配项迁移而来的 CATH注释。(h) 在新序列生成中，输入为蛋白质序列，输出为该输入蛋白质的一个变体，并保持相同的目标功能。(i) 在 GB1适应度预测中，输入为对应于四个可能位置突变的蛋白质序列，输出为表示适应度预测结果的回归分数

个点属于非结合者或结合能力较弱者。研究采用了抽样划分来评估所考察的蛋白质语言模型，因为这种划分能提供最稳定的结果。

### 4.2.3 蛋白质结构预测

这组任务旨在评估蛋白质的基于序列的嵌入是否包含有关其结构的准确信息。氨基酸序列链会折叠成一组预先确定的稳定三维结构。蛋白质的大多数生物学参数都可以从其结构中推断出来 [33]。因此，这组任务因与蛋白质理解相关，且能够支持大量应用而备受重视。

**接触预测（CP）：** 这是一项二元分类任务，旨在预测残基对是否

会在其三维结构中形成接触（通常以 8 ˚A 距离阈值定义）。接触预测作为一种残基级三维结构预测，为蛋白质结构提供重要的整体信息。在相关文献中，接触预测被用作原子级三维结构预测的中间预测步骤。我们使用了 ProteinNet [34], ，这是一个用于结构学习的标准化数据集，其方法是借助 CASP 竞赛 [35]。ProteinNet 将 CASP 结构用作测试集，并将 CASP 比赛日期之前发布的所有历史结构记录纳入训练集和评估集。我们使用了 ProteinNet 的最新版本，该版本将 CASP12 用作测试集，且采用与 TAPE 相同的数据集划分 [26]。为进一步评估模型训练的稳健性，我们基于最新一届 CASP 竞赛 CASP14 的自由建模（FM）结构，增加了独立测试。

**折叠预测（FolP）：** 这是一项分类任务，将完整的蛋白质序列分类为 1194 种可能的折叠之一。该任务用于检测所关注蛋白质（如新型抗生素耐药基因和工业酶）的新出现远缘同源物 [36]。我们采用了 Hou 的数据集 [37], 及其原始划分。整个聚类超家族均被保留在测试数据集中，以验证模型检测序列差异极大的蛋白质之间结构相似性的泛化能力。

**二级结构预测（SSP）：** 这是一项分类任务，将蛋白质中的每个残基分类为其二级结构折叠，难度分为两个级别：3 类和 8 类。二级结构包含有关功能域的重要信息，通常通过多序列比对来获取进化信息。我们采用了 NetSurfP-2.0[38] 的训练集和评估集，并使用了多种测试集来验证模型的稳健性，包括 CB 513 [39], 、TS115 [40], 和 CASP12 [41], 以及 CASP 14 [42]。

**基于嵌入的注释转移（EAT）：** 传统上，蛋白质注释转移是通过基于同源性的推断（HBI），在序列空间中将已标注（经实验注释）的蛋白质的注释转移至未标注的蛋白质。近年来，基于嵌入的注释转移已成为一种更快速的替代方法，因为它无需进行多序列比对（MSA）计算。在这一新框架中，通过在嵌入空间中使用 k 近邻（k-NN），计算查询蛋白质与一组带注释蛋白质之间的距离，并将注释从最相似的已知匹配项转移过来[43, 44]。此任务评估原始嵌入在无需监督训练的情况下，捕捉蛋白质相关有意义信息的能力。我们使用了一个基准测试集，其中包含 6.9 万条查找序列和 219 条测试序列，这些序列是为 ProTucke 评估而开发的 [44]。该评估数据集取自 CATH v4.3 数据集，其中的蛋白质按四个层级的结构注释进行分类：类别、架构、拓扑和同源超家族 [45]。为简化起见，我们在表中报告四项准确率分数的平均值，作为此任务的性能指标。1。如表16所示，四个类别的表现与平均值一致。

### 4.2.4 蛋白质定位预测

此任务旨在评估蛋白质嵌入捕捉蛋白质在细胞中预期积累位置的能力，这称为蛋白质定位预测 [46]。这一属性对于理解蛋白质功能至关重要，尤其是在疾病靶点识别研究中。

**亚细胞定位预测（LocP）：** 此分类任务评估蛋白质在 10 个亚细胞类别中的定位。我们采用 DeepLoc [47] 数据集，并使用其论文中描述的相同数据集划分。

### 4.2.5 新型蛋白质序列生成

在比较研究之后，我们使用$Ankh$ 模型生成天然蛋白质的合成变体，以验证其在一项重要蛋白质工程任务——计算变体生成中的适用性。我们使用两个输入数据集评估 Ankh，它们代表蛋白质工程中两种不同的设置和规模：基于家族的变体生成和基于单序列的变体生成。

**High-N（基于家族的变体生成）：** 对于基于家族的应用场景，我们使用了一个经过整理的苹果酸脱氢酶（MDH）数据集，该数据集被用作 ProteinGAN [48] 的训练数据集。ProteinGAN 是一种近期出现的基于深度学习的生成模型，在 MDH 案例中，其表现优于实验性变体生成。选择这一蛋白质家族

很方便，因为它拥有多样的 16,706 条独特训练序列，而且酶催化过程十分复杂，需要同时结合其底物和 NAD+ 辅因子，这增加了生成过程的复杂性

**单次（单序列变体生成）：** 对于单序列用例，我们使用了 6 月之后加入 CoV-AbDab数据集的 SARS-CoV-2 单链纳米抗体 2022 [49]。这样可以确保这些是模型在无监督训练中未见过的新序列。这个单次生成用例通过要求模型在不对目标数据集过拟合的情况下生成小规模数据集的变体，专门考验模型的泛化能力。我们进行了七次独立的虚拟生成实验，使用了 CoV-AbDab 以以下名称标识的七种不同纳米抗体：Nb-007、F6、Nb 1-23、Nb 1-25、Nb 2-62、Nb 2-65 和 Nb 2-67。所有选定的纳米抗体都具有经过实验验证的结构，以便与生成候选纳米抗体的预测结构进行比较。

## 4.3 下游模型：ConvBERT

对于将预训练嵌入映射到指定监督目标的顶部/监督模型实现，我们采用了同一个监督网络，仅进行了极少修改，以适应蛋白质处理层级（例如残基级和整个蛋白质层面）以及输出分布（例如二元分类、多类别分类和回归）的差异。在所有情况下，我们的顶部/监督模型均由两类层组成。第一类层的选择借鉴了 ProtTrans 先前开展的实验：将卷积神经网络作为顶部/下游模型或层，并证明其与自注意力结合时性能更佳[3]。我们采用了一个 ConvBERT层，其嵌入维度与预训练模型相同；前馈网络维度为预训练嵌入维度的一半（即，如果我们使用嵌入维度为 1280 的 ESM-1b 进行基准测试，则前馈网络维度为 640）；包含 4 个注意力头，丢弃率为 0.2，卷积核大小为 7，并采用 Gated-GELU 激活函数 [50]。第二类是线性层，其激活函数分别根据回归、二元分类和多类别分类任务，在 None、Sigmoid 和 Softmax 之间变化。

然而，第三类层并非在所有情况下通用。事实上，我们仅在回归和二元分类任务中，于上述监督网络之前或其开始处使用全局最大池化层。总体而言，我们认可并倡导任务特定优化。因此，我们认为，采用不同超参数集合和配置的不同顶部模型，可能会带来更好的下游性能。尽管如此，我们统一采用了被认为能够实现更佳泛化性能的顶部模型设置，因为下游基准测试的核心在于评估和比较每个模型所学习的蛋白质表征中蕴含的蛋白质理解水平。

## 4.4 变体生成模型

针对 High-N（基于蛋白质家族的生成）规模，所提出的自回归微调生成框架提供了一种易于使用的变体生成方法，并且可以轻松扩展到不同的蛋白质家族。此外，通过调整 logit 变换的温度参数，该框架能够轻松控制生成过程中的探索-利用权衡。为验证该框架，首先使用同一个微调模型生成三个数据集，每个数据集包含 500 条序列，分别采用三种不同的温度（1.0、1.5 和 2.0）。对于这三个数据集，我们报告了生成集合与天然序列集的多序列比对（MSA）之间的香农熵曲线。香农熵旨在通过比较氨基酸变异的代表性统计数据，描述生成集合是否保留了天然 MDH 集合的进化序列特性。低熵值反映了维持功能性的保守残基，而高熵值则反映了突变率较高、保守性较低的区域。我们可以在图2中观察到，三个生成集合与天然序列的分布高度相似，峰值和谷值的位置几乎完全一致。天然集合与生成集合的熵值之间的均方误差（MSE）分别为 0.1、0.09 和 0.08，对应的生成温度为 1.0、1.5 和 2.0。我们强调，所报告的相似性是基于包含 500 条序列的生成集合计算得出的，该集合不到总量的 3%

天然序列集的序列总数为 16,700。换言之，该模型只需使用一小部分原始序列，便能够模仿天然序列集的分布。

由于变体生成任务采用的框架与前述预训练顶部模型框架及评估设置不同，因此我们单独介绍。我们采用两种变体生成框架：针对 High-N 的自回归微调（基于家族的生成），以及用于 One-Shot 的掩码语言建模（MLM）（基于单个蛋白质的生成）。

微调是指在特定数据集/任务上对预训练模型进行专门训练 [51]。然而，我们将自回归微调定义为：在经典微调设置的基础上增加一项约束，即完全冻结编码器，只允许解码器的参数发生变化。我们允许解码器的所有层都进行这种变化，并使用原始解码器参数初始化微调。每项实验训练 2 个训练轮次，将掩码语言建模预测转换为自回归预测。我们将最大序列长度设为 256 个词元，最大提示长度设为 20 个词元。我们采用的学习率为 $3e{-}4$ ，Adam 优化器的 epsilon 值为 $e{-}8$ 。此外，训练批次大小设为 4，评估批次大小设为 8。对于自回归采样，我们使用束搜索，波束数为 10。对于自回归 logit 变换，我们采用温度，并使用 1、1.5 和 2 三种不同的温度值，以观察模型在不同温度下的表现。束搜索采样与温度 logit 变换的结合方式如下：首先，温度会改变 logit 分布，保留词元的顺序，同时使分布变平滑或变尖锐。其次，进行采样时（在我们的设置中采用贪心方式）对波束进行评分。

MLM 指的是一种预训练目标，它通过要求模型在整个词汇表 [51] 上以多类别分类的方式预测随机抽取的一部分输入词元（通常由 $[MASK]$ 占位符替代），来引导模型学习词元表示，并相应地影响其推理。不过，对于微调目的，我们仅在推理过程中使用 MLM（即未进行任何训练或参数更改）。此外，

与预训练时通常采用较低的掩码概率不同，我们进行了两项实验，使用较高的掩码概率，以涵盖原始序列中更大范围的变化。这两项实验采用了不同的探索-利用权衡。第一项实验尝试使用较高的掩码概率（50%）和较低的温度 logit 变换（1.0）。第二项实验采用较低的掩码概率（40%）和较高的温度 logit 变换（1.5）。两项实验均使用包含 30 个波束的束搜索进行采样。

## 4.5 计算能力（软件与硬件）

在处理有限词汇表、并将无限生命融入其词元的同时，我们还必须探索如何从软件和硬件两方面提升计算能力的效率上限。

### 4.5.1 Flax 与 JAX

Flax 是一个端到端的高性能库，也是一个面向 JAX 的生态系统，旨在提供灵活性并针对神经网络进行定制。JAX 提供了一系列合理的可组合功能转换，支持即时（JIT）编译、自动微分、CPU/GPU/TPU 兼容性，以及自动向量化和并行化 [52]。例如，与 PyTorch 相比，JAX 在 TPU Pod 集群上进行语言模型训练时，速度可提升约 1.4 倍。我们使用 Flax 构建源代码，以便在谷歌 TPU Pod 集群上训练蛋白质语言模型时利用上述优势；同时，我们也在 HuggingFace 上提供模型，以支持更广泛的研究人员使用三大深度学习库中的任意一种：JAX、TensorFlow 和 PyTorch。

### 4.5.2 TPU

我们很幸运，成为最早使用谷歌最新 TPU v4 的论文之一；尤其是，我们是最早利用它进行蛋白质建模的研究工作，从而释放出超级计算机前所未见的能力。这意味着，本研究中的所有预训练模型均使用谷歌

TPU v4 Pod 进行训练，使用的 TPU 核心数为 64 或 128。单个 TPU v4 虚拟机主机配备 8 个 TPU 核心，每个核心拥有 16 GiB 高带宽内存、120 个 CPU 核心和 400 GB 主存。乍看之下，TPU V4 与 TPU V3 相似；然而，它有两项主要优势。首先，新的超级核心特性允许虚拟合并两个核心，使 JAX 等深度学习库将每两个核心视为一个核心。这样一来，每台主机上的四个合并核心各自都能访问 32 GiB 内存，因此无需使用模型并行，就能在单台主机上运行规模高达 30 亿参数的模型。其次，与 TPU V3 相比，TPU V4 的速度约提升 2.2 倍。

## 4.6 数据与模型实验

### 4.6.1 掩码

掩码是一种预训练目标，通过要求模型预测随机抽取的一部分输入词元，来引导模型学习词元表示。这些词元通常会被 $[MASK]$占位符替换，并在涵盖整个词汇表的多类别设置下进行预测 [21]。这类模型实验旨在研究两个与掩码相关的参数——掩码策略和掩码概率——所带来的影响。

**a. 掩码策略：**掩码策略指的是我们决定哪些词元需要掩码、哪些词元保持不变的方式 [53–55]。鉴于蛋白质序列中的氨基酸词元分布不均，以及数据库中存在冗余，我们测试了不同的掩码策略，以确保其适用于蛋白质领域。为此，我们测试了五种变体／实验。

**实验 0：**该实验代表我们的基线模型，详情请参见第 3.1.2 小节。

**实验 1：** 我们至少对序列中的每个独特 1-gram 词元（即每个独特的氨基酸）进行了一次掩码。具体来说，我们反复遍历序列，每次随机对一个氨基酸进行掩码，前提是尚未达到目标掩码概率，且相关氨基酸不是计数最高的氨基酸（即，如果我们对序列“$ABCAAAAAAA....A$”使用 15% 的掩码概率，其长度为 20 个氨基酸：词元“$A$”将始终被掩码一次，其余两次掩码将始终针对词元“$B$”和“$C$”）。最后，我们对整个序列词元进行了完整的反掩码/重建。研究发现，这种方法相较于基线能够取得更高的性能，因此我们采用了该方法。

**实验 2：** 我们还确保序列中的每个独特词元至少被掩码一次，但没有每次只掩码一个词元，而是同时掩码该词元前后相邻的词元（即在序列“$ABCDEFG$”中：如果词元“$D$”被掩码，那么词元“$C$”和“$E$”也会被掩码），从 1-gram 词元掩码改为 3-gram 词元掩码。我们也保留了对整个序列令牌进行完整反掩码/重建的做法。然而，这降低了所有下游任务的性能，因此没有继续采用。

**实验 3：** 我们复现了实验 *1*，即确保序列中的每个独特 1-gram 词元至少被掩码一次。然而，与默认情况不同，默认情况下所有词元的反掩码/重建都会纳入损失函数的计算，而我们只将被掩码词元的重建纳入计算，尽管输出仍会包含整个序列令牌（即在序列“$ABCDEFG$”中：如果词元“$D$”和“$F$”被掩码，那么计算损失函数时只计入词元“$D$”和“$F$”的重建）。然而，这降低了所有下游任务的性能，因此我们推断，即使整个输入已知，也应重建整个输入；相应地，我们没有继续采用这种做法。

**实验 4：** 本实验体现了掩码与反掩码的输入—目标映射方式发生了变化。具体而言，在默认的 1-gram 词元掩码情形下，每个输入词元都会被重建为一个目标词元。我们所做的改动是将所有连续的未掩码标记重建为一个合并标记（也就是说，如果输入序列为“$ABCDEFG$”，且词元“$C$”和“$G$”被掩码，那么输入序列为“$A, B, [MASK_0], D, E, F, [MASK_1]$”，重建结果则为“$[MASK_0], C, [MASK_1], G$”，其中 $[MASK_0]$ 是一个映射到两个输入词元（AB）的单一目标词元，而 $[MASK_1]$ 是一个映射到三个输入词元（DEF）的单一目标词元，从而将其从 1-gram 词元掩码转变为 1-gram 跨度掩码。此项改动旨在减少重建

输出中未掩码标记所涉及的不必要计算所带来的计算成本。我们将这种掩码策略称为“1-Gram Span Partial De-masking/Reconstruction”（1-gram 跨度部分反掩码/重建）。需要注意的是，这种部分重建仅作用于输出，输入词元则保持原样。这一方向已被证明是可行的，相较于第一个实验取得了更高的性能，因此后续实验也沿用了这一方法。

**实验5：** 我们将实验 *4*中引入的 1-gram 跨度部分重建应用于实验 *1*的方法：至少对序列中每个唯一的 1-gram 词元进行一次掩码，并将所有未掩码标记重建为一个合并标记（即，如果对长度为 20 个氨基酸的序列“$ABCAAAAAAA....A$”使用 15% 的掩码概率，且第一个随机索引为零，则输入序列将为“$[MASK_0]$, $[MASK_1]$, $[MASK_2]$, $A$, $A$, $A$, $A$, ...., $A$”，并重建为“$A$, $B$, $C$, $[MASK_0]$”，其中“$[MASK_0]$”是一个映射到十七个输入词元的单一词元。然而，结果显示这种做法会持续降低性能，因此我们放弃了这一方案。

**实验 6：** 我们尝试了实验 *4* 的一个变体，在部分重建时，将所有连续掩码的词元映射为一个词元（即，如果输入序列为“$ABCDEFG$”，且词元“$C$”、“$D$”和“$E$”被掩码，那么输入序列将为“$A$, $B$, $[MASK_0]$, $F$, $G$”，并重建为“$[MASK_0]$, $[MASK_1]$, $[MASK_2]$”，其中“$[MASK_1]$”是一个映射到三个掩码词元的单一词元，$[MASK_0]$和 $[MASK_2]$ 是分别映射到两个未掩码词元的单一词元），从而将其变为 3-元跨度掩码。这一改动同样旨在降低计算成本，但结果表明这并非有效方向，因此我们没有继续推进。

因此可以推断，在测试的六个版本中，表现最佳的是 实验 *4*版本；在该版本中，我们将所有连续的未掩码标记重建为一个合并标记。因此，该版本被用于后续关于掩码概率的实验子集。此组实验的结果见表 3。

**b. 掩码概率**掩码概率表示整个序列长度中需要进行掩码的词元所占的比例。根据基线模型的配置，默认掩码概率为 15% [21]。在此，我们针对实验 *4*中表现最佳的版本，额外测试了三个取值。

**实验 7：** 首次测试的掩码概率为 10%。

**实验 8：** 第二次测试的掩码概率为 20%。

**实验 9：** 第三次测试的掩码概率为 30%。

在这四个数值中，10% 是表现最差的掩码概率，因此被弃用。有趣的是，我们发现 实验 *4* 的默认值（15% 掩码概率）在亚细胞定位预测、折叠预测以及部分二级结构预测任务中表现更佳，适用的数据集包括 *CB513* 和 TS115。然而，我们发现 实验 *9* 的数值（30% 掩码概率）在全部接触预测任务以及 CASP12 数据集的二级结构预测任务中表现更佳。鉴于二级结构预测数据集的结果并不一致，我们参考了作为领域标准的 CASP12 结果，该结果更支持较高的掩码概率。此外，由于我们倾向于通过通用语言模型进行整体预测，因此我们更重视不同类型任务之间的泛化能力，尤其是在结果差异幅度如此之小、且任务特定定制会产生影响的情况下。最后，鉴于所有实验变体仅训练了两个训练轮次，我们在实验后长期训练中采用了中间值 实验 *8* （20% 掩码概率），以实现对通用蛋白质语言模型预期任务集合的半全面涵盖，并使其能够随后针对不同下游任务进行定制。此组实验的结果见表 4。

### 4.6.2 架构

由于我们采用编码器-解码器 Transformer 架构，目标架构变体对应于编码器层和解码器层的数量、深度与宽度的不同组合，以及模型学习词元顺序（即位置嵌入）的方式。

模型学习词元顺序（即位置嵌入）的方式。

**a. 编码器-解码器层的数量**尽管解码器对于改善编码器生成的表征至关重要，但先前对 Prot-T5 的运行结果表明，在大多数下游任务中，解码器并未带来显著差异，因此将其移除，使推理成本几乎降低了一半 [3]。这促使我们尝试使用层数多于解码器的编码器，并测试不同的层数差异，同时也尝试相反的配置，以排除任何反驳意见。值得注意的是，我们保持编码器和解码器的总层数不变，即 72 层；在此前的一组实验中，两者各有 36 层。基于迄今表现最佳的版本 实验 7，我们测试了编码器与解码器相对层数的三种变化方案。

**实验 10：** 我们首先尝试采用 54 层编码器和 18 层解码器。

**实验 11：** 随后，我们尝试采用包含 48 层的编码器和包含 24 层的解码器。

**实验 12：** 最后，我们尝试采用包含 48 层的解码器和包含 24 层的编码器。

结果发现，具有 48 层编码器和 24 层解码器的版本 -实验 *11*- 在所有二级结构预测任务（8态）、折叠和亚细胞定位预测，以及整体均值和中位数方面均优于*Exp. 8* 版本，而另外两个版本的表现则因任务而异，呈现波动。最终，我们采用实验 *11* （48 层编码器和 24 层解码器）开展后续实验，在编码器与解码器规模相同、总成本相等的情况下，凭借更大的编码器获得了提取更丰富嵌入的优势。我们的选择主要是为了促进泛化；这意味着需要保留足够数量的解码器层，因为它们对广泛的生成任务以及大多数任务数据集都至关重要。此外，我们的选择也出于计算效率的考虑：编码器层数较少，因而能够更快地进行特征提取。这组5实验的结果见表。

**b. 深度与宽度变化 层**深度对应层数；如前所述，在 Transformer 中，这对应于编码器和解码器的层数。而宽度在 Transformer 的语境下对应嵌入维度 [19]。

**实验 13：** 关于深度-宽度变化，我们只进行了一项实验：将嵌入层维度从 768 增加到 1024，并相应地将深度从 48 层编码器和 24 层解码器减少为 24 层编码器和 12 层解码器，以使参数数量大致相同或更少。然而，这导致结果波动，因此我们没有继续进行。我们将嵌入维度为 768 的版本称为基础模型，或 $Ankh\ base$。这组实验的结果见表 6。

**c. 激活函数** 激活函数是为前向层引入非线性的功能 [56]。到目前为止，以及在此前的所有实验中，我们一直使用的激活函数都是 Gated-GELU。需要注意的是，Gated-GELU 意味着显著更大的可训练参数规模，这反过来迫使我们通过降低深度来抵消宽度的增加；我们在实验 *12*[15]中采用了这一做法。为了克服 Gated-GELU 对参数量的高需求，避免因此降低模型的深度，我们需要彻底更换激活函数，从而不必被迫牺牲过大的深度。因此，围绕深度-宽度变化开展的剩余两项实验，对应于将激活函数改为经典 ReLU，并测试深度和宽度的两种组合 [57]。此时，你可能会想到，同时改变深度/宽度和激活函数相当于设置了两个自变量。然而，需要注意的是，这种安排源于 Gated-GELU 所提出的需求；当不再使用 Gated-GELU 时，该需求也随之消失。此外，我们还需要维持相同的参数数量，以便进行计算上公平的比较。

**实验 14：** 我们测试的第一种组合是：编码器深度为 62 层，解码器深度为 11 层，嵌入维度为 768。

**实验 15：** 第二种组合采用深度为 48 层的编码器和 24 层的解码器，嵌入维度同样为 768。

研究发现，在深度-宽度变化实验组中尝试的所有组合，均未能持续超越表现最佳的版本，实验 *11*。因此，该组的组合均未继续采用，我们回退到实验 *11*。该组实验的结果见表7。

**d. 相对位置嵌入**位置嵌入描述序列词元的位置，使每个位置都分配到唯一且固定的表征。对于 Transformer 而言，这种分配至关重要，因为其核心思想是用注意力取代序列单元。 [58]。使用固定的位置嵌入存在若干局限，例如无法为超出预定义位置嵌入维度最大长度的词元提取嵌入，而该维度与嵌入维度相关 [59]。为克服这一局限，人们引入了相对位置嵌入。相对位置嵌入并非为每个位置使用固定嵌入，而是为序列词元分配可变的相对位置表征，其变化取决于自注意力设置中相互比较的“键”和“查询”之间的偏移量。我们采用了 [20]中介绍的相对位置嵌入的一种简化变体，其中每个表征仅是一个标量，会加到用于计算注意力权重的对应对数几率上。在这种设置下，模型中所有表征的参数在所有层之间共享，不过给定层中的每个注意力头都会使用不同的、经过学习的位置表征。尽管学习得到的嵌入/表征是相对且可变的，但在同一实验中，学习得到的嵌入维度是固定的，因为它对应于一系列可能的键-查询偏移量。到目前为止，以及在前述实验中，我们所有模型均使用了 32 个嵌入。此外，我们始终使用 128 个词元的偏移范围，并将所有相对位置映射到同一个嵌入。

值得注意的是，给定层对超出 128 个词元的相对位置并不敏感，但后续层仍可通过组合前面各层的局部信息来感知更大的偏移量，从而使模型能够处理长度超过预定最大长度的序列。这组实验采用了不同的嵌入维度和嵌入偏移量组合，旨在检验使用少量大尺寸嵌入、许多小尺寸嵌入，还是介于两者之间的方案更好。这组实验中的前两项保留了默认的 32 维嵌入，但改变了嵌入偏移量。

**实验 16：** 该组的第一个实验保留了默认嵌入维度 32，但将嵌入偏移量增至 256。

**实验 17：** 该组的第二个实验也保留了默认嵌入维度 32，但将嵌入偏移量降至 64。研究发现，较小的嵌入偏移量 64 的表现优于 256 和默认值 128。该组接下来的两个实验保留了表现最佳的嵌入偏移量 64，但改变了嵌入维度。

**实验 18：** 这组实验中的第三个实验保留了 64 的嵌入偏移量，但将嵌入维度增加到了 64。

**实验 19：** 这组实验中的第四个实验保留了 64 的嵌入偏移量，但将嵌入维度减小到了 16。

然而，这些变化都未能始终胜过嵌入维度为 32、嵌入偏移量为 64 时的表现。既然结果表明双精度值表现更好，这组实验的最后两个实验便尝试了两种双精度值组合。

**实验 20：** 第五个实验测试了嵌入偏移量为 128、嵌入维度为 64 的配置。

**实验 21：** 这组实验的最后一个实验测试了嵌入偏移量为 256、嵌入维度为 128 的配置。

研究发现，结果最稳定且泛化性最好的组合是实验 *20*（嵌入偏移量为 128，嵌入维度为 64），因此后续采用了该组合

当分类任务的结果不一致时，我们将其称为折叠预测；当二级结构数据集的结果不一致时，我们将其称为$CASP12$。这组实验的结果见表8。

**e. 权重绑定** 权重绑定的初衷是减少语言模型训练所需的参数数量，从而缩短训练和收敛时间，并提高结果的一致性 [60]。

**实验 22：** 我们通过绑定/共享嵌入和解码器的权重与偏置来实现上述目的。然而，研究发现，这种做法并未始终优于迄今为止表现最佳的模型。其原因在于，分别对输入词元和输出词元进行掩码和去掩码的方式导致输入和输出类型存在差异，这表明参数较少的设置具有更强的预测能力。这组实验的结果见表9。

### 4.6.3 数据集

在 Prot-T5 的实验中发现，$UniRef50$ 的表现优于更大的数据集，例如 $UniRef100$ 和 $BFD$ [3]。这归因于 $UniRef50$ 的高质量，具体体现在重复序列较少且序列多样性较高。在此前的所有实验中，预训练也都是在 UniRef50 上进行的。不过，我们希望测试一种规模居中的数据集，其规模介于 UniRef50 与那些已证明效率较低的大型数据集之间。

**实验23：** 我们在$UniRef90$上进行了一项实验。尽管如此，这项实验仍鼓励我们沿着最初的方向，继续采用$UniRef50$作为高效获取高质量属性的代表。值得注意的是，之前所有使用$UniRef50$进行的实验都训练了完整的 2 个训练轮次，而使用$UniRef90$的实验只训练了 1 个训练轮次，后者可以说与前者相当。这组实验的结果见表10。

## 5 可用性

Ankh 和 $Ankh\ base$ 模型均可在我们的 Ankh代码库“https://github.com/agemagician/Ankh”中公开获取，用于研究创新，遵循署名-非商业性-相同方式共享 4.0 国际许可协议（CC BY-NC-SA 4.0）。该代码库还提供了大量 Python 和 Jupyter Python notebook 教程，涵盖各种示例，包括使用免费在线资源（谷歌 Colab）进行嵌入提取，以及针对多个下游任务开展监督学习。如需商业使用和许可，请访问“https://www.proteinea.com/”

## 6 致谢

作者谨向 Proteinea 的深度学习和《生物信息学》团队致谢，感谢他们在硬件、软件及本工作的诸多其他方面提供的宝贵帮助。感谢哥伦比亚大学的 Mohammed AlQuraishi 提供反馈。作者还要感谢谷歌的 Jonathan Caton、Shira Genauer、Astitva Chopra，以及所有谷歌云、谷歌研究创新者、JAX 和 TRC 团队，感谢他们协助在 Google Cloud 上搭建项目并解决 Google Cloud 的问题。若没有 HuggingFace 团队（包括 Patrick von Platen、Julien Chaumond 和 Clement Delangue）的支持，本研究训练的模型便难以轻松地公开发布。谷歌通过 Google 研究创新者计划和 Google TPU 云研究积分计划为本项目提供了支持。我们还要感谢全球所有研究人员，是他们将本研究使用的所有数据集公开发布。最后，ElNaggar 和 Essam 感谢真主赐予他们完成本项目并与全人类分享的力量、知识和勇气。

## 参考文献

[1] Vig, Jesse、Madani, Ali、Varshney, Lav R、Xiong, Caiming、Socher, Richard 和 Rajani, Nazneen Fatema。BERTology 遇上生物学：解读蛋白质语言模型中的注意力。*arXiv* 预印本 *arXiv:2006.15222*，2020年。

[2]Rao, Roshan、Meier, Joshua、Sercu, Tom、Ovchinnikov, Sergey 和 Rives, Alexander。Transformer蛋白质语言模型是无监督的结构学习器。*Biorxiv*，2020年。

[3] Elnaggar, Ahmed、Heinzinger, Michael、Dallago, Christian、Rihawi, Ghalia、Wang, Yu、Jones, Llion、Gibbs, Tom、Feher, Tamas、Angerer, Christoph、Steinegger, Martin 等人。Prot-Trans：通过自监督深度学习和高性能计算，致力于破解生命的代码的语言。*arXiv* 预印本 *arXiv:2007.06225*，2020年。

[4]Rives, Alexander、Meier, Joshua、Sercu, Tom、Goyal, Siddharth、Lin, Zeming、Liu, Jason、Guo, Demi、Ott, Myle、Zitnick, C Lawrence、Ma, Jerry 等人。将无监督学习扩展至 2.5 亿条蛋白质序列后，生物结构与功能应运而生。美国国家科学院院刊，第 118 卷，第 15 期，2021 年。

[5]Heinzinger, Michael、Elnaggar, Ahmed、Wang, Yu、Dallago, Christian、Nechaev, Dmitrii、Matthes, Florian、Rost, Burkhard。通过迁移学习蛋白质序列，对生命语言的若干方面进行建模。《*BMC*生物信息学》，第 20 卷，第 1 期，2019 年。

[6] Alley, Ethan C、Khimulya, Grigory、Biswas, Surojit、AlQuraishi, Mohammed、Church, George M。利用基于序列的深度表征学习，实现统一的理性蛋白质工程。*Nature methods*，第 16 卷，第 12 期，2019 年。

[7] Madani, Ali、McCann, Bryan、Naik, Nikhil、Keskar, Nitish Shirish、Anand, Namrata、Eguchi, Raphael R、Huang, Po-Ssu 和 Socher, Richard。Progen：用于蛋白质生成的语言建模。*arXiv* 预印本 *arXiv:2004.03497*，2020年。

[8]Ofer, Dan、Brandes, Nadav 和 Linial, Michal。蛋白质的语言：自然语言处理、机器学习与蛋白质序列。计算与结构生物技术期刊，第19卷，2021。

[9]Hesslow, Daniel、Zanichelli, Niccol´o、Notin, Pascal、Poli, Iacopo 和 Marks, Debora。《RITA：生成式蛋白质序列模型扩展研究》。*arXiv* 预印本 *arXiv:2205.05789*，2022。

[10] Lin, Zeming、Akin, Hali、Rao, Roshan、Hie, Brian、Zhu, Zhongkai、Lu, Wenting、dos Santos Costa, Allan、Fazel-Zarandi, Maryam、Sercu, Tom、Candido, Sal 等。《进化尺度的蛋白质序列语言模型能够实现精准的结构预测》。*bioRxiv*，2022。

[11] Nijkamp, Erik、Ruffolo, Jeffrey、Weinstein, Eli N、Naik, Nikhil 和 Madani, Ali。ProGen2：探索蛋白质语言模型的边界。*arXiv* 预印本 *arXiv:2206.13517*，2022年。

[12]Gardner, Matt、Grus, Joel、Neu- mann, Mark、Tafjord, Oyvind、Dasigi, Pradeep、Liu, Nelson、Peters, Matthew、Schmitz, Michael 和 Zettlemoyer, Luke。Allennlp：一个深度语义自然语言处理平台。*arXiv* 预印本 *arXiv:1803.07640*，2018。

[13] Cock,Peter JA、Antao, Tiago、Chang, Jeffrey T、Chapman, Brad A、Cox, Cymon J、Dalke, Andrew、Fried- berg, Iddo、Hamelryck, Thomas、Kauff, Frank、Wilczynski, Bartek 及其他作者。BioPython：面向计算分子生物学和《生物信息学》的免费 Python 工具。《生物信息学》，第 25 卷第 11 期，2009。

[14]Mirdita, Milot、Sch¨utze, Konstantin、Moriwaki, Yoshitaka、Heo, Lim、Ovchinnikov, Sergey 和 Steinegger, Martin。ColabFold：让所有人都能使用蛋白质折叠。*Nature Methods*，2022。

[15] Hendrycks, Dan 和 Gimpel, Kevin. 高斯误差线性单元（GELU）。*arXiv* 预印本 *arXiv:1606.08415*，2016。

[16] Rost, Burkhard. 酶功能的保守性低于预期。《分子生物学杂志》，第 318 卷，第 2 期，2002 年。

[17] Suzek, Baris E、Wang, Yuqi、Huang, Hongzhan、McGarvey, Peter B、Wu, Cathy H 和 UniProt联盟。UniRef

簇：一种全面且可扩展的替代方案，可改进序列相似性搜索。《生物信息学》，第 31 卷，第 6 期，2015 年。

[18]Steinegger, Martin and Mirdita, Milot and S¨oding, Johannes. 蛋白质水平组装使宏基因组样本中的蛋白质序列恢复量大幅增加。《*Nature methods*》，第 16 卷，第 7 期，2019 年。

[19] Vaswani, Ashish 和 Shazeer, Noam 和 Par- mar, Niki 和 Uszkoreit, Jakob 和 Jones, Llion 和 Gomez, Aidan N 和 Kaiser, Lukasz 和 Polosukhin, Illia。《 Attention Is All You Need》。*Advances in Neural Information Processing Systems*，第30卷，2017年。[20] Raffel, Colin 和 Shazeer, Noam 和 Roberts, Adam 和 Lee, Katherine 和 Narang, Sharan 和 Matena, Michael 和 Zhou, Yanqi 和 Li, Wei 和 Liu, Peter J 等。《利用统一的文本到文本 Transformer 探索迁移学习的极限》。机器学习研究期刊，第21卷，第140期，2020年。[21] Devlin, Jacob 和 Chang, Ming-Wei 和 Lee, Kenton 和 Toutanova, Kristina。《BERT：用于语言理解的深度双向 Transformer 预训练》。*arXiv*预印本 *arXiv:1810.04805*，2018年。[22] Lan, Zhenzhong 和 Chen, Mingda 和 Goodman, Sebastian 和 Gimpel, Kevin 和 Sharma, Piyush 和 Soricut, Radu。《 ALBERT：用于语言表征自监督学习的轻量级 BERT》。*arXiv*预印本 *arXiv:1909.11942*，2019年。[23] Clark, Kevin 和 Luong, Minh-Thang 和 Le, Quoc V 和 Manning, Christopher D。《ELECTRA：将文本编码器预训练为判别器而非生成器》。*arXiv*预印本 *arXiv:2003.10555*，2020年。[24]Dai, Zihang 和 Yang, Zhilin 和 Yang, Yiming 和 Carbonell, Jaime 和 Le, Quoc V 和 Salakhutdinov, Ruslan。《Transformer-XL：超越固定长度上下文的注意力语言模型》。*arXiv*预印本 *arXiv:1901.02860*，2019年。

[25] Yang, Zhilin 和 Dai, Zihang 和 Yang, Yim- ing 和 Carbonell, Jaime 和 Salakhutdinov, Russ R 和 Le, Quoc V。《 XLNet：用于语言理解的广义自回归预训练》。*Advances in Neural Information Processing Systems*，第32卷，2019年。[26] Rao, Roshan 和 Bhattacharya, Nicholas 和 Thomas, Neil 和 Duan, Yan 和 Chen, Peter 和 Canny, John 和 Abbeel, Pieter 和 Song, Yun。《使用 TAPE 评估蛋白质迁移学习》。*Advances in Neural Information Processing Systems*，第32卷，2019年。[27]Xu, Minghao 和 Zhang, Zuobai 和 Lu, Jiarui 和 Zhu, Zhaocheng 和 Zhang, Yang- tian 和 Ma, Chang 和 Liu, Runcheng 和 Tang, Jian。《PEER：蛋白质序列理解的综合性多任务基准》。*arXiv*预印本 *arXiv:2206.02096*，2022年。[28]Sarkisyan, Karen S 和 Bolotin, Dmitry A 和 Meer, Margarita V 和 Usmanova, Dinara R 和 Mishin, Alexander S 和 Sharonov, George V 和 Ivankov, Dmitry N 和 Bozhanova, Nina G 和 Baranov, Mikhail S 和 Soylemez, Onuralp 等。《绿色荧光蛋白的局部适应度景观》。*Nature*，第533卷，第7603期，2016年。[29] Sormanni, Pietro 和 Amery, Leanne 和 Ekizoglou, Sofia 和 Vendruscolo, Michele 和 Popovic, Bojana。《快速、准确地对单克隆抗体文库进行计算机模拟溶解度筛选》。*Scientific Reports*，第7卷，第1期，2017年。[30] Khurana, Sameer 和 Rawi, Reda 和 Kunji, Khalid 和 Chuang, Gwo-Yu 和 Bensmail, Halima 和 Mall, Raghvendra。《 DeepSol：基于序列的蛋白质溶解度预测深度学习框架》。《生物信息学》，第34卷，第15期，2018年。[31] Wu, Nicholas C 和 Dai, Lei 和 Olson, C Anders 和 Lloyd-Smith, James O 和 Sun, Ren。《间接路径促进蛋白质适应度景观中的适应》。*eLife*，第5卷，2016年。[32] Dallago, Christian 和 Mou, Jody 和 Johnston

，Kadina E 和 Wittmann ，Bruce J 和

Bhattacharya, Nicholas、Goldman, Samuel、Madani, Ali 和 Yang, Kevin K. FLIP：用于蛋白质适应度景观推理的基准任务。*bioRxiv*，2021。

[33] McLachlan, Andrew D. 蛋白质结构的快速比较。《晶体学报 *A* 辑：晶体物理、衍射、理论与普通晶体学》，第 38 卷第 6 期，1982。

[34] AlQuraishi,Mohammed. ProteinNet：用于蛋白质结构机器学习的标准化数据集。《*BMC*生物信息学》，第 20 卷第 1 期，2019。

[35] Kryshtafovych,Andriy and Schwede,Torsten and Topf, Maya and Fidelis, Krzysztof and Moult, John. 蛋白质结构预测方法的关键评估（CASP）——第十三轮。《蛋白质：结构、功能与生物信息学》，第 87 卷第 12 期，2019。

[36] Chen, Daozheng and Tian, Xiaoyu and Zhou, Boand Gao, Jun. Profold：利用额外结构特征和新型集成分类器进行蛋白质折叠类别分类。《国际生物医学研究》，第 2016 卷，2016 年。

[37]Hou、Jie、Adhikari、Badri 和 Cheng、Jianlin。DeepSF：用于将蛋白质序列映射到折叠的深度卷积神经网络。《生物信息学》，第 34 卷，第 8 期，2018 年。

[38]Klausen、Michael Schantz、Jespersen、Martin Closter、Nielsen、Henrik、Jensen、Kamilla Kjaergaard、Jurtz、Vanessa Is abell、Soenderby、Casper Kaae、Sommer、Morten Otto Alexander、Winther、Ole、Nielsen、Morten、Petersen、Bent 等。NetSurfP-2.0：通过整合深度学习改进蛋白质结构特征预测。《蛋白质：结构、功能与生物信息学》，第 87 卷，第 6 期，2019 年。

[39]Yang, Yuedong 和 Gao, Jianzhao 和 Wang, Jihua 和 Heffernan, Rhys 和 Hanson, Jack 和 Paliwal, Kuldip 和 Zhou, Yaoqi。蛋白质二级结构预测漫长征程的六十五年：最后阶段？《生物信息学简报》，第 19 卷，第 3 期，2018 年。[40] Cuff, James A 和 Barton, Geoffrey J。蛋白质二级结构预测多序列方法的评估与改进。《蛋白质：结构、功能与生物信息学》，第 34 卷，第 4 期，1999 年。

[41] Abriata、Luciano A 和 Tam`o、Giorgio E、Monastyrskyy、Bohdan、Kryshtafovych、Andriy 和 Dal Peraro、Matteo。CASP12 困难靶标建模评估揭示了基于比对的接触预测方法日益显著的作用。《蛋白质：结构、功能与生物信息学》，第 86 卷，2018 年。

[42]Moult, J 和 Kryshtafovych, A. 特刊：CASP14：第 14 轮蛋白质结构预测方法的关键评估。蛋白质，第 89 卷，2021 年。

[43]Littmann, Maria、Heinzinger, Michael、Dallago, Christian、Olenyi, Tobias 和 Rost, Burkhard。深度学习嵌入可将 GO 注释迁移至超越同源性的领域。*Scientific reports*，第 11 卷，第 1 期，2021 年。

[44]Heinzinger, Michael、Littmann, Maria、Sillitoe, Ian、Bordin, Nicola、Orengo, Christine 和 Rost, Burkhard。在蛋白质嵌入上进行对比学习，揭示了午夜区。*NAR* 基因组学与生物信息学 $f_{学}$，第 4 卷，第 2 期，2022。

[45]Sillitoe, Ian、Bordin, Nicola、Daw- son, Natalie、Waman, Vaishali P、Ashford, Paul、Scholes, Harry M、Pang, Camilla SM、Woodridge, Laureland Rauer, Clemens、Sen, Neeladri 等。CATH：提高了功能空间的结构覆盖度。《核酸研究》，第 49 卷，第 D1 期，2021。

[46] Silhavy, THOMAS J、Benson, SPENCER A 和Emr,SCOTT D。蛋白质定位的机制。《微生物学评论》，第 47 卷，第 3 期，1983。

[47]Almagro Armenteros, Jos´e Juan 和 Sønderby, Casper Kaae 和 Sønderby, Søren

Kaae 和 Nielsen, Henrik 和 Winther, Ole。《DeepLoc：利用深度学习预测蛋白质亚细胞定位》。《生物信息学》，第 33 卷，第 21 期，2017 年。[48]Repecka, Donatas 和 Jauniskis, Vykintas 和 Karpus, Laurynas 和 Rembeza, Elzbieta 和 Rokaitis, Irmantas 和 Zrimec, Jan 和 Poviloniene, Simona 和 Laurynenas, Audrius 和 Viknander, Sandra 和 Abuajwa, Wis- sam 等。《利用生成对抗网络拓展功能性蛋白质序列空间》。*Nature Machine Intelligence*，第 3 卷，第 4 期，2021 年。[49] Raybould, Matthew IJ 和 Kovaltsuk, Alek- sandr 和 Marks, Claire 和 Deane, Charlotte M.《CoV-AbDab：冠状病毒抗体数据库》。《生物信息学》，第 37 卷，第 5 期，2021 年。[50] Jiang, Zi-Hang 和 Yu, Weihao 和 Zhou, Daquan 和 Chen, Yunpeng 和 Feng, Jiashi 和 Yan, Shuicheng。《Convbert：通过基于跨度的动态卷积改进 bert》。*Advances in Neural InformationProcessingSystems*，第 33 卷，2020 年。[51]Goodfellow, Ian 和 Bengio, Yoshua 和 Courville, Aaron。《深度学习》，2016 年。[52] Frostig, Roy 和 Johnson, Matthew James 和 Leary, Chris。《通过高级跟踪编译机器学习程序》。机器学习系统，第 4 卷，第 9 期，2018 年。[53] Zeng, Changchang 和 Li, Shaobo。《分析掩码长度分布对掩码语言建模的影响：中文机器阅读理解数据集上的评估框架与案例研究》。*WirelessCommunications and Mobile Computing*，第 2021 卷，2021 年。[54] Xiao, Dongling 和 Li, Yu-Kun 和 Zhang, Han 和 Sun, Yu 和 Tian, Hao 和 Wu, Hua 和 Wang, Haifeng。《AERNIE-Gram：针对自然语言理解的显式 n-gram 掩码语言建模预训练》。*arXiv* 预印本 *arXiv:2010.12148*，2020 年。[55] Levine, Yoav 和 Lenz, Barak 和 Lieber, Opher 和 Abend, Omri 和 Leyton-Brown, Kevin、Tennenholtz、Moshe、Shoham、Yoav。PMI掩码：对相关跨度进行有原则的掩码。*arXiv* 预印本 *arXiv:2010.01825*，2020年。[56] Szanda la，Tomasz。深度神经网络常用激活函数的综述与比较。仿生计算，2021。[57] Agarap，Abien Fred。使用整流线性单元（ReLU）进行深度学习。*arXiv* 预印本 *arXiv:1803.08375*，2018。[58]Ke，Guolin、He，Di 和 Liu，Tie-Yan。重新思考语言预训练中的位置编码。*arXiv* 预印本 *arXiv:2006.15595*，2020年。[59] Su，Jianlin、Lu，Yu、Pan，Shengfeng、Wen，Bo 和 Liu，Yunfeng。RoFormer：采用旋转位置嵌入的增强型 Transformer。*arXiv* 预印本 *arXiv:2104.09864*，2021。[60] Denil，Misha、Shakibi，Babak、Dinh，Laurent、Ranzato，Marc'Aurelio 和 De Freitas，Nando。预测深度学习中的参数。*Advances in Neural Information Processing Systems*，第26卷，2013年。

# 7 补充材料

## 7.1 预训练数据与模型实验

在本小节中，我们报告预训练数据的统计信息，以及各类实验的结果。在表2中，我们展示了预训练数据的统计信息。

**表 2**：预训练数据统计。

| **预训练数据** | UniRef50 | UniRef90 | UniRef100 |
|---|---|---|---|
| **蛋白质数量［（百万）］** | 45 | 144 | 216 |
| **残基数量［（十亿）］** | 14 | 49 | 88 |
| **磁盘空间 ［（GB）］** | 16 | 54 | 150 |

预训练数据集统计信息。UniRef50 和 UniRef90 分别以 50% 和 90% 的序列一致性阈值对 UniRef100 数据库进行聚类。统计信息描述了蛋白质数量（百万，m）、氨基酸数量（十亿，b）以及磁盘空间（GB，未压缩的文本存储）。

在表3 中，我们报告了关于掩码策略的实验结果。可以推断，在测试的六个版本中，表现最佳的是实验*4*版本：我们将所有连续的未掩码标记重构为一个合并标记。因此，该版本被延续到后续关于掩码概率的实验子集中。

**表3**：掩码策略实验统计

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| 基线 | 74.6% | 80.6% | 78.6% | 62.0% | 68.9% | 64.6% | 18.8% | 29.9% | 75.8% | 41.4% | 64.0% | 68.9% |
| 实验 1 | 75.1% | 81.6% | 80.4% | 61.3% | 69.9% | 66.0% | 20.2% | 33.6% | 74.9% | 42.6% | 65.0% | 69.9% |
| 实验 2 | 71.9% | 75.1% | 73.1% | 57.5% | 62.8% | 58.7% | 11.4% | 19.9% | 68.7% | 35.5% | 58.1% | 62.8% |
| 实验 3 | 69.5% | 74.9% | 73.5% | 54.9% | 61.3% | 57.4% | 9.3% | 14.8% | 70.4% | 28.5% | 56.1% | 61.3% |
| 实验 4 | **76.3%** | **82.8%** | **82.2%** | **63.7%** | **71.8%** | **68.8%** | **22.2%** | **37.6%** | **77.0%** | **50.7%** | **67.9%** | **71.8%** |
| 实验5 | 75.6% | 81.0% | 80.0% | 62.2% | 68.9% | 65.8% | 18.4% | 31.5% | 73.5% | 33.4% | 63.5% | 68.9% |
| 实验 6 | 74.9% | 81.6% | 79.3% | 61.5% | 69.3% | 64.8% | 20.7% | 34.9% | 75.5% | 46.9% | 65.4% | 69.3% |

掩码策略实验组的结果。

在表4中，我们报告了掩码概率实验的结果。在四个取值中，10% 是效果最差的掩码概率，因此不予考虑。有趣的是，我们发现，Exp. 4 的默认值（15% 掩码概率）在亚细胞定位预测、折叠预测以及 CB513 和 TS115 等数据集上的部分二级结构预测任务中表现更佳。不过，Exp. 9 的取值（30% 掩码概率）在所有接触预测任务以及 CASP12 数据集上的二级结构预测任务中表现更佳。鉴于不同二级结构预测数据集的结果并不一致，我们参考了作为领域标准的 CASP12 结果，该结果支持采用更高的掩码概率。此外，由于我们希望通过通用语言模型实现整体预测，因此我们倾向于促进模型在不同类型任务上的泛化，尤其是在结果差异很小且任务特定定制会产生影响的情况下。最后，鉴于所有实验变体仅训练了两个训练轮次，我们在实验后的长期训练中采用了 Exp. 8 的中间值（20% 掩码概率），以便半全面地涵盖通用蛋白质语言模型所面向的不同任务类别，之后还可针对下游任务进行定制。

**表 4**：掩码概率实验统计

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| 实验 4 | 76.3% | 82.8% | **82.2%** | 63.7% | **71.8%** | **68.8%** | 22.2% | 37.6% | **77.0%** | **50.7%** | **67.9%** | **71.8%** |
| 实验 7 | 76.7% | **83.0%** | 81.9% | 63.7% | 71.7% | 68.5% | 21.8% | 33.8% | 76.7% | 46.9% | 66.7% | 71.5% |
| **Exp. 8** | 75.9% | 82.7% | 81.4% | 62.8% | 71.5% | 67.9% | 22.3% | 35.1% | 76.7% | 47.6% | 67.1% | 71.7% |
| 实验 9 | **76.8%** | 82.2% | 81.3% | **64.0%** | 71.4% | 68.0% | **24.1%** | **37.9%** | 75.1% | 46.3% | 67.0% | 71.4% |

掩码概率实验组的结果。

在表 5中，我们报告了编码器层数和解码器层数实验的结果。结果发现，采用 48 层编码器和 24 层解码器的版本——实验 *11*——表现优于 *Exp. 8* 的版本，在所有二级结构预测任务（8态）、折叠预测和亚细胞定位预测中，以及总体均值和中位数上均是如此；而另外两个版本的表现则因任务而异，起伏不定。最终，我们在后续一系列实验中采用了实验 *11*（48 层编码器和 24 层解码器），在编码器-解码器总成本相同的情况下，通过更大的编码器提升了提取更丰富嵌入的能力。我们的选择主要是为了促进泛化，这意味着需要保留足够数量的解码器层，因为它们对于广泛的生成任务至关重要，同时也有助于整合大多数任务数据集。此外，我们的选择也考虑了计算效率

**表 5**：架构实验：编码器和解码器层数

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| **Exp. 8** | 75.9% | 82.7% | 81.4% | 62.8% | 71.5% | 67.9% | 22.3% | 35.1% | 76.7% | 47.6% | 67.1% | 71.7% |
| **实验 10** | 76.5% | **83.2%** | 82.0% | 63.2% | 72.2% | 68.6% | **24.0%** | **39.5%** | 76.8% | 48.1% | 67.7% | 72.2% |
| **实验 11** | 76.4% | 83.1% | **82.3%** | **63.7%** | **72.3%** | **69.2%** | 23.0% | 37.1% | **77.2%** | **48.5%** | **67.8%** | **72.3%** |
| **实验 12** | **76.8%** | 82.9% | 81.4% | 63.3% | 71.8% | 68.1% | 21.3% | 34.9% | 75.7% | 46.5% | 66.8% | 71.8% |

编码器层数与解码器层数组合实验的结果。

体现在较少的编码器层数上，从而实现更快的特征提取。在表 6中，我们报告了深度与宽度变化实验的结果。可以看到，所测试的变化带来了波动的结果，因此没有继续进行。我们将嵌入维度为 768 的版本称为基础模型，或 *Ankh base*。

**表 6**：架构实验：深度与宽度变化

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| **实验 11** | **76.4%** | 83.1% | 82.3% | **63.7%** | **72.3%** | **69.2%** | **23.0%** | 37.1% | 77.2% | **48.5%** | **67.8%** | **72.3%** |
| **实验 13** | 76.1% | **83.3%** | **82.6%** | 62.7% | 70.9% | 68.0% | 22.6% | **37.5%** | **77.3%** | 47.8% | 67.4% | 70.9% |

深度与宽度变化实验的结果。

In 表 7，我们报告了激活函数实验的结果 .

**表 7**：架构实验：激活函数

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| **实验 11** | **76.4%** | **83.1%** | **82.3%** | **63.7%** | **72.3%** | **69.2%** | **23.0%** | **37.1%** | **77.2%** | **48.5%** | **67.8%** | **72.3%** |
| **实验 14** | **76.4%** | **83.1%** | 78.9% | 62.7% | 69.0% | 64.7% | 19.7% | 33.4% | 76.0% | 45.4% | 65.3% | 69.0% |
| **实验 15** | 75.8% | 82.0% | 80.1% | 62.5% | 70.8% | 66.6% | 21.2% | 36.7% | 75.5% | 46.6% | 66.3% | 70.8% |

激活函数实验的结果。

研究发现，在深度-宽度变化实验组中尝试的所有组合都未能持续超越表现最佳的版本，实验 *11*。因此，该组中的所有组合均未继续采用，并恢复使用实验 *11*。

在表8中，我们报告了相对位置编码实验的结果。研究发现，表现最稳定且泛化性最好的组合是实验 *20*（嵌入偏移量为 128，嵌入数量为 64），因此后续采用了该组合。分类任务的结果不一致时，我们参考折叠预测；二级结构数据集的结果不一致时，则参考*CASP12* 。

在表格 9中，我们报告了权重绑定实验的结果。可以看出，该实验并未始终优于迄今表现最佳的模型。这归因于输入和输出类型之间的差异，这是由我们分别对输入和输出词元进行掩码和反掩码的方式造成的，也表明参数较少的设置具有更强的预测能力。

**表8**：架构实验：相对位置编码

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| 实验 11 | 76.4% | 83.1% | **82.3%** | 63.7% | **72.3%** | **69.2%** | 23.0% | 37.1% | 77.2% | 48.5% | 67.8% | **72.3%** |
| 实验 16 | 76.2% | 82.9% | 82.0% | 64.0% | 71.8% | 68.9% | 22.7% | 38.0% | **77.8%** | 47.6% | 67.7% | 71.8% |
| 实验 17 | 76.7% | 83.0% | 82.0% | 63.7% | 71.7% | 68.8% | 23.6% | **40.8%** | 76.8% | 50.1% | 68.2% | 71.7% |
| 实验 18 | 76.1% | **83.3%** | 82.0% | 63.7% | 71.7% | 68.6% | 21.5% | 34.9% | 76.9% | 48.4% | 63.7% | 71.7% |
| 实验 19 | 76.0% | **83.3%** | 81.9% | 62.9% | 72.1% | 68.6% | **24.2%** | 40.2% | 77.2% | 48.0% | 67.8% | 72.1% |
| 实验 20 | **77.0%** | 83.0% | **82.3%** | **64.3%** | 72.2% | 69.1% | 23.6% | 40.7% | 77.2% | **51.8%** | **68.6%** | **72.2%** |
| 实验 21 | 76.2% | 82.5% | 81.5% | 63.9% | 71.1% | 68.1% | 25.2% | 41.3% | 76.4% | 49.2% | 67.8% | 71.1% |

相对位置编码实验的结果。

**表9**：架构实验：权重绑定

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| 实验 20 | **77.0%** | **83.0%** | **82.3%** | **64.3%** | **72.2%** | **69.1%** | **23.6%** | **40.7%** | 77.2% | **51.8%** | **68.6%** | **72.2%** |
| 实验 22 | 75.9% | **83.0%** | **82.3%** | 63.0% | 72.1% | **69.1%** | 22.6% | 36.0% | **78.1%** | 48.9% | 67.6% | 72.1% |

权重绑定实验的结果。

在表格 10中，我们报告了数据集实验的结果。尽管如此，该实验仍支持最初采用 *UniRef50* 作为高效且具备高质量属性的代表数据集这一方向。值得注意的是，之前所有使用 *UniRef50* 的实验都训练了整整 2 个训练轮次；相比之下，使用 *UniRef90* 的实验只训练了 1 个训练轮次，这种设置可以说与之相当。

**表10**：数据集实验

| Exp. | SSP（Q3） | | | SSP（Q8） | | | CP(L) | | SLP | FolP | Avg. | Med. |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | *CASP12* | *TS115* | *CB513* | *CASP12* | *TS115* | *CB513* | *L1* | *L5* | | | | |
| 实验 20 | **77.0%** | **83.0%** | **82.3%** | **64.3%** | **72.2%** | 69.1% | **23.6%** | **40.7%** | **77.2%** | **51.8%** | **68.6%** | **72.2%** |
| 实验23 | 75.9% | 82.8% | 81.9% | 63.3% | 71.1% | **69.4%** | 22.8% | 38.7% | 75.4% | 45.9% | 67.2% | 71.7% |

数据集实验的结果。

## 7.2 模型预训练

在本小节中，我们报告了两个 Ankh 模型的预训练配置和超参数，并将其与基线以及下游模型进行比较，详见表11。

**表 11**：预训练配置。

| 超参数 | 基线 | Ankh 基础版 | Ankh large | 下游模型 |
|---|---|---|---|---|
| 数据集 | UniRef50 | UniRef50 | UniRef50 | NA |
| 嵌入维度 | 768 | 768 | 1536 | Ankh 嵌入维度 |
| 注意力头数 | 12 | 12 | 16 | 4 |
| 学习率 | 0.01 | 0.004 | 0.002 | 0.001 |
| 训练轮数 | 2 | 68 | 68 | 5 |
| 掩码概率 | 15% | 20% | 20% | NA |
| 编码器层数 | 36 | 48 | 48 | NA |
| 解码器层数 | 36 | 24 | 24 | NA |
| 激活函数 | Gated-GELU | Gated-GELU | Gated-GELU | Gated-GELU |
| 最大长度 | 512 | 512 | 512 | NA |
| 本地批次大小 | 16 | 24 | 16 | 1 |
| 全局批次大小 | 960 | 1536 | 1024 | 16 |
| 优化器 | Adafactor | Adafactor | Adafactor | AdamW |
| 学习率调度器 | 线性 | 线性 | 线性 | 线性 |
| 前馈层数量 | 3072 | 3072 | 3840 | 嵌入维度/2 |
| 预热步数 | 10k | 10k | 10k | 1k |
| 权重衰减 | None | None | None | None |
| 随机种子 | 42 | 42 | 42 | 7 |
| TPU/GPU | V4-64 | V4-128 | V4-128 | A100 |
| 权重/偏置初始化 | 随机 | 随机 | 随机 | 随机 |

用于实验基线模型、预训练 $Ankh\ base$ $Ankh$ 以及下游任务中顶部模型的配置。

## 7.3 数据与模型测试

### 7.3.1 下游数据集

在本小节中，我们报告了下游任务中使用的多个数据集的统计信息。相关统计信息见表12。

**表 12**：监督式下游数据集说明。

| Task Name | 任务类别 | 输出分布 | 建模层级 | 数据来源 | # 蛋白质 | 划分来源 | # 训练集/验证集/测试集 |
|---|---|---|---|---|---|---|---|
| FluP | 功能 | 回归 | 蛋白质层面 | Sarkisyan | 54,025 | TAPE | 21,446/5,362/27,217 |
| SolP | 功能 | 分类 | 蛋白质层面 | DeepSol | 71,419 | DeepSol | 62,478/6,942/1,999 |
| GB1P | 功能 | 回归 | 蛋白质层面 | FLIP | 8,733 | FLIP | 6,291/670/1,772 |
| CP | 结构 | 分类 | 残基对级别 | ProteinNet | 25,563 | TAPE | 25,299/224/40 |
| FolP | 结构 | 分类 | 蛋白质层面 | DeepSF | 13,766 | DeepSF | 12,312/736/718 |
| SSP | 结构 | 分类 | 残基级 | NetSurfP-2.0 | 11,361 | TAPE | 8,678/2,170/513 |
| LocP | 定位 | 分类 | 蛋白质层面 | DeepLoc | 13,961 | DeepSF | 8,945/2,248/2,768 |

比较研究中考察的基准任务说明。我们报告了任务的类别信息，以及任务数据集的原始来源、蛋白质数量，以及所采用的训练/验证/测试划分的来源和数值分布。缩写：FluP：荧光预测；StaP：热稳定性预测；SolP：溶解度预测；CP：接触预测；FolP：折叠预测；SSP：二级结构预测；LocP：定位预测。

### 7.3.2 基于嵌入的预测的完整结果

除了主页面中展示的领域标准数据集外，本节还在表格13 中展示了涵盖多种额外数据集的基于嵌入的预测的完整结果，并进一步证实了相同的结果。

**表13**：结果摘要。

| Task | 数据集 | Ankh | Ankh 基础版 | ProtT5-XL-U50 | ESM-1b | ESM-2 (6.5亿) | ESM-2 (3B) | ESM-2 (15B) |
|---|---|---|---|---|---|---|---|---|
| SSP（Q3） | *CASP12* | **83.8**±3% | 80.8±4% | 83.4±4% | 79.6±4% | 82.3±4% | 83.3±4% | 83.2±3% |
| | *CASP14* | **77.6**±3% | 76.8±3% | 74.1±3% | 75.1±4% | 77.0±3% | 76.8±3% | 76.8±4% |
| | *TS115* | **88.2**±1% | 86.8±1% | 86.8±1% | 84.9±1% | 87.2±1% | 87.5±1% | 87.3±1% |
| | *CB513* | **88.6**±0.6% | 87.0±0.7% | 86.7±0.6% | 84.4±0.7% | 87.0±0.6% | 87.3±0.6% | 87.4±0.6% |
| SSP（Q8） | *CASP12* | **71.5**±4% | 68.7±4% | 70.7±3% | 66.6±4% | 70.6±3% | 71.4±4% | 71.4±4% |
| | *CASP14* | **63.2**±3% | 62.3±4% | 60.7±5% | 60.0±5% | 62.7±4% | 61.6±5% | 61.7±6% |
| | *TS115* | **78.2**±2% | 76.9±2% | 77.0±2% | 73.7±2% | 77.0±2% | 77.2±2% | 77.6±2% |
| | *CB513* | **77.4**±1% | 75.5±1% | 75.1±1% | 71.0±1% | 75.6±1% | 75.8±1% | 75.9±1% |
| CP（L） | *ProteinNet L/1* | **49.0**±8% | 43.2±8% | 44.7±8% | 30.0±6% | 29.6±6% | 30.7±6% | 33.3±6% |
| | *ProteinNet L/5* | **73.2**±11% | 66.6±11% | 69.2±11% | 50.1±10% | 50.2±10% | 52.7±10% | 54.7±10% |
| | *CASP14 L/1* | **30.2**±8% | 28.8±7% | 26.9±7% | 24.6±6% | 25.0±6% | 24.8±7% | 25.9±7% |
| | *CASP14 L/5* | **50.7**±11% | 48.0±11% | 42.4±14% | 40.0±11% | 38.4±13% | 41.9±14% | 40.4±15% |
| EAT | | 71.7±6% | **74.8**±6% | 71.0±6% | 64.5±7% | 55.5±7% | 65.6±6% | 65.4±7% |
| FolP | | **61.1**±4% | 58.8±4% | 57.6±4% | 57.6±4% | 56.3±4% | 60.5±4% | 56.7±4% |
| GB1P | | 0.84±0.008 | **0.85**±0.008 | 0.78±0.01 | 0.81±0.009 | 0.82±0.009 | 0.81±0.009 | 0.57±0.02 |
| LocP | | **83.2**±2% | 81.4±2% | **83.2**±2% | 80.0±2% | 81.8±2% | 82.4±2% | 81.8±2% |
| FluP | | **0.62**±0.4 | 0.61±0.4 | 0.58±0.4 | 0.5±0.5 | 0.48±0.5 | 0.48±0.5 | 0.55±0.4 |
| SolP | | **76.4**±2% | 74.2±2% | 74.4±2% | 67.3±2% | 75.0±2% | 74.9±2% | 74.4±2% |
| Avg. | | **71.43%** | 70.0% | 69.3% | 65.3% | 66.9% | 67.7% | 66.6% |
| Med. | | **77.5%** | 76.0% | 75.2% | 71.9% | 76.1% | 76.4% | 73.5% |

涉及 $Ankh$ 和 $Ankh\ base$ 的基准测试结果汇总，使用的模型包括 ProtT5-XL-U50、ESM-1b、ESM-2（6.5亿）、ESM-2（30亿）和 ESM-2 (15B)。对于回归任务，我们报告斯皮尔曼相关系数；对于所有分类任务，我们报告准确率分数，但接触预测除外，该任务报告 L/1 和 L/5 精确率。在 EAT 中，报告的分数是四种注释（类别、架构、拓扑和同源超家族）的准确率分数均值。任务缩写：SSP：二级结构预测；CP：接触预测；EAT：基于嵌入的注释迁移；FolP：折叠预测；GB1P：GB1适应度预测；LocP：定位预测；FluP：荧光预测；SolP：溶解度预测。

### 7.3.3 基于注意力的预测

在本小节中，我们报告了表格14中基于注意力的预测在接触预测任务上的结果。由于对于 ESM蛋白质语言模型系列，注意力图被认为是接触预测更好的指标， [2] 和 [10], 中测试了这两种表征。具体而言，我们为每个模型提取注意力图，并将其与上下文化嵌入分别作为接触预测任务的输入进行比较，以确保公平比较，并展示我们认为最能体现当前最先进水平的指标。

**表14**：基于注意力的预测。

| Task | 数据集 | Ankh large | Ankh 基础版 | ProtT5-XL-U50 | ESM-1b | ESM-2（6.5亿） | ESM-2（30亿） | ESM-2 (15B) |
|---|---|---|---|---|---|---|---|---|
| CP（L） | *ProteinNet L/1* | **31.4**±5% | 25.9±5% | 30.9±5% | 25.3±5% | 31.9±6% | 33.9±6% | 33.3±6% |
| | *ProteinNet L/5* | **55.6**±8% | 46.3±8% | 51.9±8% | 42.0±8% | 54.6±9% | 56.6±9% | 57.4±9% |
| | *CASP14 L/1* | **11.1**±6% | 9.3±5% | 8.6±5% | 7.8±4% | 10.7±5% | 12.2±7% | 12.3±7% |
| | *CASP14 L/5* | **20.7**±12% | 19.5±10% | 16.1±10% | 15.8±10% | 21.0±11% | 21.4±13% | 24.6±13% |

基准测试结果 表明，基于注意力的预测性能不如基于嵌入的接触预测结果 s

### 7.3.4 基于嵌入的注释迁移

在本小节中，我们报告了CATH结构域基于嵌入的注释迁移的完整结果，如表15所示。

**表15**：基于嵌入的注意力迁移的准确率分数

| 模型 | C | A | T | H | Mean |
|---|---|---|---|---|---|
| **ProtT5-XL-U50** | 83.6±5% | 70.0±6% | 57.2±7% | **73.3**±7% | 71.0±6% |
| **ESM-1b** | 78.1±5% | 65.3±6% | 51.4±7% | 63.3±8% | 64.5±7% |
| **ESM-2（6.5亿）** | 72.1±6% | 56.2±7% | 40.4±6% | 53.3±8% | 56.0±7% |
| **ESM-2（30亿）** | 79.5±5% | 65.8±6% | 53.4±7% | 64.0±8% | 65.6±6% |
| **ESM-2 (15B)** | 78.5±5% | 63.5±6% | 53.4±7% | 67.3±8% | 65.4±7% |
| *Ankh_base* | **85.8**±5% | **77.2**±6% | **63.5**±6% | 72.7±7% | **74.8**±6% |
| *Ankh_large* | 83.6±5% | 72.1±6% | 61.0±7% | 70.7±7% | 71.7±6% |

报告了 CATH 四个层级的准确率分数，用于比较真实标签与预测标签；预测标签对应于嵌入查找空间中最接近的匹配项。第 1 级 C 表示类别（Class）。第 2 级 A 表示架构。第 3 级 T 表示拓扑。第 4 级 H 表示同源超家族。

### 7.3.5 CATH结构域

在本小节中，我们报告对自然和生成数据进行 CATH 域分析的结果。在表 16 中，我们展示了 CATH 超结构域分析。

**表 16**：生成数据与自然（微调）数据中的 CATH 超家族结构域概览。

| Name | 超家族 ID | 自然 | t1.e1 | t1.e2 | t2.e1 | t2.e2 |
|---|---|---|---|---|---|---|
| **乳酸脱氢酶/糖苷水解酶** | 3.90.110.10 | 8088 | 4549 | 3961 | 6103 | 6657 |
| **AglA样葡糖苷酶** | 3.90.1820.10 | 7540 | 4082 | 3648 | 5337 | 5863 |
| **NAD(P)结合型罗斯曼样结构域** | 3.40.50.720 | 171 | 24 | 30 | 3059 | 2148 |
| **其他结构域** | NA | 196 | 0 | 0 | 18 | 5 |
| **结构域总数** | NA | 15799 | 8655 | 7639 | 14517 | 14673 |

天然序列集中属于三个主要同源超家族的 CATH结构域分布。缩写：Nat：天然序列；t1.e1：第一轮训练以温度 1 生成的序列；t1.e2：第二轮训练以温度 1 生成的序列；t2.e1：第一轮训练以温度 2 生成的序列；t2.e2：第二轮训练以温度 2 生成的序列。

在表 17 中，我们展示了 CATH 功能域分析。

**表17**：生成数据与自然（微调）数据中 CATH 功能家族结构域的汇总。

| Name | 功能家族编号 | 超家族 ID | 自然 | t1.e1 | t1.e2 | t2.e1 | t2.e2 |
|---|---|---|---|---|---|---|---|
| 苹果酸脱氢酶 | 4 | 3.90.110.10 | 3675 | 2081 | 1917 | 3150 | 3277 |
| | 18 | 3.40.50.720 | 170 | 23 | 5 | 1772 | 2647 |
| | 1 | 3.90.110.10 | 6 | 0 | 0 | 155 | 95 |
| | 17 | 3.40.50.720 | 0 | 0 | 0 | 8 | 6 |
| 苹果酸脱氢酶总和 | NA | NA | 6812 | 3582 | 3621 | 6553 | 7931 |
| L-乳酸脱氢酶 | 11 | 3.90.110.10 | 421 | 205 | 125 | 282 | 254 |
| | 3 | 3.90.110.10 | 28 | 89 | 30 | 77 | 43 |
| | 29 | 3.40.50.720 | 0 | 0 | 0 | 13 | 20 |
| | 362 | 3.40.50.720 | 0 | 0 | 0 | 0 | 1 |
| L-乳酸脱氢酶总和 | NA | NA | 449 | 294 | 155 | 372 | 318 |

已知功能注释的结构域在不同 Fun-Family 编号中的分布。大多数自然结构域被注释为苹果酸脱氢酶（MDH），这是我们关注的蛋白质家族。较少一部分结构域属于与之密切相似的蛋白质家族 L-乳酸脱氢酶（LDH）；其功能与 MDH 接近，但并非优化目标。与 LDH 结构域相比，高温下生成的集合倾向于生成 MDH 结构域。