**1. Introduction（引言）**

**翻譯：**

表格數據的監督學習是工業應用中普遍存在的機器學習場景。在傳統的非深度學習方法中，梯度增強決策樹（GBDT）是此類任務的最先進解決方案。而表格數據的深度學習模型據報告正在改進，最新研究聲稱在學術基準測試中可與GBDT並駕齊驅，甚至超越GBDT。

然而，從實用角度來看，除了多層感知機（MLP）這樣的簡單架構外，表格深度學習是否提供了任何明顯的首選基線還不清楚。作者指出了三個問題：

1. 新方法相對於簡單MLP基線的性能改進規模和一致性在文獻中並不總是明確分析
2. 效率相關特性（如訓練時間，特別是推理吞吐量）有時較少受到關注
3. 一些研究表明學術基準測試的進展可能無法很好地轉移到實際任務中

**說明：**

作者發現了一個被忽視的機會：**參數高效集成（parameter-efficient ensembling）**，這是一種將多個模型集成為一個產生多個預測的單一模型的範式。

**舉例說明：**

想像你有一個預測房價的任務：

* 傳統方法：訓練5個獨立的MLP模型，每個都有完整的參數
* TabM方法：訓練1個MLP，但內部有32個"子模型"共享大部分權重，只有少數參數不共享

**2. Related Work（相關工作）**

**翻譯與分類：**

**2.1 決策樹基礎模型**

梯度增強決策樹（GBDT）是表格任務強大且高效的基線，是決策樹的集成。

**2.2 表格深度學習架構**

近年來提出了大量表格數據深度學習架構：

* **注意力機制架構**：FT-Transformer等
* **檢索增強架構**：結合最近鄰搜索的方法
* **MLP類模型**：各種多層感知機變體

**2.3 深度集成**

深度集成指多個相同架構的DL模型在不同隨機種子下獨立訓練。集成預測是成員平均預測。深度集成通常顯著優於相同架構的單一DL模型，但缺點是訓練和使用多個模型的成本和不便。

**2.4 參數高效深度"集成"**

為了以較低成本實現深度集成的性能，多項研究提出了通過一個模型產生多個預測來模仿集成的架構。

**舉例說明：**

* **傳統深度集成**：訓練5個完全獨立的神經網路，每個100MB，總共500MB
* **參數高效集成**：訓練1個神經網路120MB，內部模擬5個子網路的效果

**3. TabM**

**3.1 設計理念**

**翻譯：**

TabM是基於MLP和參數高效集成技術的表格DL模型，與BatchEnsemble密切相關。TabM產生每個對象的多個預測。

**核心概念：**

* **k個子模型**：TabM內部有k=32個隱式子模型
* **權重共享**：大部分權重在所有子模型間共享
* **適配器**：每個子模型有少量獨特的參數（稱為適配器）

**3.2 BatchEnsemble原理**

**翻譯：**

對於任何線性層 l(x) = Wx + b，在BatchEnsemble中：

* 第i個成員：li(xi) = si ⊙ (W(ri ⊙ xi)) + bi
* 其中⊙是逐元素乘法，W在所有成員間共享，ri, si, bi不共享

**舉例說明：**

python

*# 傳統方法：5個獨立模型*

model1 = MLP(weights1) *# 100MB*

model2 = MLP(weights2) *# 100MB*

...

model5 = MLP(weights5) *# 100MB*

total = 500MB

*# BatchEnsemble方法：1個共享模型+適配器*

shared\_weights = W *# 100MB*

adapters = [r1,s1,b1, r2,s2,b2, ..., r5,s5,b5] *# 5MB*

total = 105MB

**3.3 TabM的演進過程**

**階段性發展：**

1. **MLP×k**：k個獨立訓練的MLP的傳統深度集成
2. **TabMnaive**：將BatchEnsemble應用於MLP骨幹
3. **TabMmini**：只保留第一個適配器R，移除其他3N-1個適配器
4. **TabM**：回到所有3N個適配器，但改進初始化

**關鍵發現：**

TabMmini性能優於TabMnaive，儘管只有1個適配器而不是3N個適配器。

**舉例說明：**

輸入：一個客戶的特徵 [年齡=30, 收入=50000, ...]

TabM處理：

1. 複製輸入32次

2. 第一層適配器將32個副本轉換為32個不同表示

3. 通過共享的MLP層

4. 最終產生32個預測：[預測1, 預測2, ..., 預測32]

5. 平均得到最終預測

最終預測 = (預測1 + 預測2 + ... + 預測32) / 32

**4. Evaluating Tabular Deep Learning Architectures（評估表格深度學習架構）**

**4.1 基線模型**

**翻譯：**

作者使用了以下基線：

* **MLP**：經典多層感知機
* **FT-Transformer**：基於注意力的模型
* **SAINT**：基於注意力和檢索的模型
* **T2G-Former**：基於注意力的模型
* **TabR**：基於檢索的模型
* **GBDT實現**：XGBoost, LightGBM, CatBoost

**4.2 任務性能**

**主要發現：**

1. 性能排名將TabM渲染為頂級DL模型
2. TabM在DL模型中保持領先地位
3. 許多DL方法在相當數量的數據集上並不比MLP更好或甚至更差

**4.3 效率分析**

**關鍵結果：**

* **訓練時間**：TabM提供實用的訓練時間，而注意力和檢索模型訓練時間過長
* **推理吞吐量**：簡單MLP是最快的DL模型，TabM緊隨其後
* **大數據集適用性**：注意力和檢索模型在大數據集上表現困難

**舉例說明：**

數據集大小：100萬行

訓練時間對比：

- MLP：10分鐘

- TabM：15分鐘

- FT-Transformer：3小時

- TabR：內存不足

推理速度對比（每秒處理對象數）：

- MLP：5000個/秒

- TabM：3000個/秒

- FT-Transformer：500個/秒

**5. Analysis（分析）**

**5.1 個體子模型的性能和訓練動態**

**關鍵發現：**

1. **個體vs集體性能**：TabM的32個子模型個體表現較弱且過擬合，但集體平均表現強且泛化良好
2. **梯度多樣性**：32個子模型的梯度之間幾乎零餘弦相似度，顯示了非平凡的多樣性
3. **最佳子模型**：即使是TabM中最好的單個子模型也不如簡單MLP

**5.2 訓練後選擇子模型**

**貪婪選擇機制：**

TabM設計允許訓練後基於任何標準選擇子模型子集，通過修剪額外的預測頭和適配器矩陣對應行來實現。

**5.3 k值對TabM性能的影響**

**發現：**

* k=32是合理的默認值，在性能和效率間取得良好平衡
* TabM比TabMmini更容易適應大量子模型

**舉例說明：**

個體子模型表現：

子模型1準確率：78%（過擬合）

子模型2準確率：76%（過擬合）

...

子模型32準確率：79%（過擬合）

集體表現：

平均預測準確率：85%（泛化良好）

這就像32個"專家"各自可能不完美，但他們的集體智慧超越了任何個體。

**總結**

TabM的核心創新在於發現了一個被忽視的機會：將參數高效集成應用於簡單的MLP可以顯著提升性能，同時保持高效率。這種方法比複雜的注意力或檢索機制更實用，為表格深度學習提供了一個強大且易於使用的新基線。