相关性：BERT模型的训练

王树森

http://wangshusen.github.io/

训练相关性BERT模型

- **训练分 4 个步骤：**预训练 (pretrain)、后预训练 (post pretrain)、微调 (fine tuning) 、蒸馏 (distillation) 。
- **预训练：**用 MLM 等任务预训练模型。
- **后预训练：**利用用户的点击、交互数据训练模型。
- **微调：**用人工标注的相关性数据训练模型。
- **蒸馏：**得到更小的模型，加速线上的推理。

微调 (fine tuning)

微调

- 微调用监督学习训练模型，模型估计 q 和 d 的相关性。
- 人工标注数十万、数百万条样本，每条样本为 (q, d, y)。

微调

- 微调用监督学习训练模型，模型估计 q 和 d 的相关性。
- 人工标注数十万、数百万条样本，每条样本为 (q, d, y)。
- 可以把估计相关性看作回归任务，也可以看作排序任务。
- 回归任务让预测的值 p 拟合 y，起到“保值”的作用。
  - 给定 (q, d)，模型估计相关性为 p。
  - p 越接近真实标签 y 越好。

微调

- 微调用监督学习训练模型，模型估计 q 和 d 的相关性。
- 人工标注数十万、数百万条样本，每条样本为 (q, d, y)。
- 可以把估计相关性看作回归任务，也可以看作排序任务。
- 回归任务让预测的值 p 拟合 y，起到“保值”的作用。
- 排序任务让 p 的序拟合 y 的序，起到“保序”的作用。
  - 给定两条样本 (q, d_1, y_1) 和 (q, d_2, y_2)，满足 $y_1 > y_2$。
  - 模型预测的相关性分数 p_1 和 p_2 应当满足 $p_1 > p_2$。

微调：回归任务

- 数据：$(q_1, d_1, y_1), \cdots, (q_n, d_n, y_n)$，其中 $y_i \in [0, 1]$。
- 模型预测 (q_i, d_i) 的相关性为 p_i。
- 最小化损失函数 $\frac{1}{n}\sum_{i=1}^{n} \text{loss}(y_i, p_i)$，使得 p_i 尽量接近 y_i。
- 均方差损失函数：

$$\text{MSE\_Loss}(y_i, p_i) = \frac{1}{2}(y_i - p_i)^2.$$

- 交叉熵损失函数（类似二分类，用 soft label）：

$$\text{CE\_Loss}(y_i, p_i) = -y_i \cdot \ln p_i - (1 - y_i) \cdot \ln(1 - p_i).$$

微调：排序任务

- 数据：一条样本包含一条查询词 q 和 k 篇文档 $d_1, \cdots, d_k$。
- 对于 (q, d_i)，真实相关性分数记作 y_i，模型预测的相关性记作 p_i。
  - 两种排序方式：按照 y_i 排序、按照 p_i 排序。
  - 排序任务不在乎 p_i 和 y_i 的值是否接近，只在乎两种排序是否接近。

微调：排序任务

- 数据：一条样本包含一条查询词 q 和 k 篇文档 $d_1, \cdots, d_k$。
- 对于 (q, d_i)，真实相关性分数记作 y_i，模型预测的相关性记作 p_i。
- 设 $y_i > y_j$，损失函数应当鼓励 $p_i - p_j$ 尽量大。
  - 如果 $p_i \geq p_j$（模型预测正确），则称 (i, j) 为正序对。
  - 如果 $p_i < p_j$（模型预测错误），则称 (i, j) 为逆序对。
  - 损失函数应当惩罚逆序对，鼓励正序对 ➔ 鼓励 $p_i - p_j$ 尽量大。

微调：排序任务

- 数据：一条样本包含一条查询词 q 和 k 篇文档 $d_1, \cdots, d_k$。
- 对于 (q, d_i)，真实相关性分数记作 y_i，模型预测的相关性记作 p_i。
- 设 $y_i > y_j$，损失函数应当鼓励 $p_i - p_j$ 尽量大。
- Pairwise logistic 损失函数：

$$\sum_{(i,j):y_i>y_j} \ln\left[1+\exp\left(-\gamma\cdot(p_i-p_j)\right)\right].$$

微调：排序任务

- 数据：一条样本包含一条查询词 q 和 k 篇文档 $d_1, \cdots, d_k$。
- 对于 (q, d_i)，真实相关性分数记作 y_i，模型预测的相关性记作 p_i。
- 设 $y_i > y_j$，损失函数应当鼓励 $p_i - p_j$ 尽量大。
- Pairwise logistic 损失函数：

$$\sum_{(i,j):y_i>y_j} \ln\left[1+\exp\left(-\gamma\cdot\left(p_i-p_j\right)\right)\right].$$

微调：小结

- 可以把估计相关性看作回归任务，也可以看作排序任务。
- 看作回归任务，使用均方差损失 (MSE) 或交叉熵损失 (CE)，有利于提升 AUC 指标。
- 看作排序任务，使用 pairwise logistic 损失，有利于提升正逆序比指标。
- 不要把估计相关性看作多分类任务！
- 如果同时用 AUC 和正逆序比作为离线评价指标，则同时使用 CE 和 pairwise logistic 损失。

后预训练 (post pretrain)

参考文献

1. Zou et al. Pre-trained language model based ranking in Baidu search. In *KDD*, 2021.
2. Zou et al. Pre-trained language model-based retrieval and ranking for web search. *ACM Transactions on the Web*, 2022.

后预训练

训练相关性模型：预训练 → 后预训练 → 微调 → 蒸馏

后预训练的步骤

1. 从搜索日志中挑选十亿对 (q, d)。
2. 自动生成标签：将用户行为 $\boldsymbol{x}$ 映射到相关性分数 $\hat{y}$。
3. 用 $(q, d, \hat{y})$ 训练模型（方法与微调类似，额外加上预训练的 MLM 任务）。

步骤1：挑选 (q, d)

- 搜索日志记录用户每次搜索的查询词 q 和搜索引擎返回的文档。
- 根据搜索日志抽取查询词 q，需要覆盖高、中、低频的 q。
- 给定 q，搜索日志记录搜到的文档 $d_1, \cdots, d_n$、以及模型估计的相关性分数（不是人工标注的）。
- 根据相关性分数，选取 n 篇文档的一个子集，均匀覆盖各相关性档位。

步骤2：自动生成相关性分数

- 步骤 1 根据搜索日志选出十亿对 (q, d)。
- 对搜索日志做统计，得出 (q, d) 的点击率和多种交互率，记作向量 $\boldsymbol{x}$。

步骤2：自动生成相关性分数

- 已经得到十亿条样本 $(q, d, \boldsymbol{x})$，其中向量 $\boldsymbol{x}$ 是用户行为。
- 相关性 y 与 $\boldsymbol{x}$ 存在某种函数关系。（相关性越高，则用户越有可能点击和交互。）
- 找出 y 与 $\boldsymbol{x}$ 的函数关系：$\hat{y} = t(\boldsymbol{x})$。
  - 选取几万对 (q, d)，人工标注相关性分数 y。
  - 搜索日志记录了 (q, d) 的用户行为 $\boldsymbol{x}$。
  - 得到几万条样本 $(\boldsymbol{x}, y)$，训练一个小模型 $t(\boldsymbol{x})$ 拟合 y。

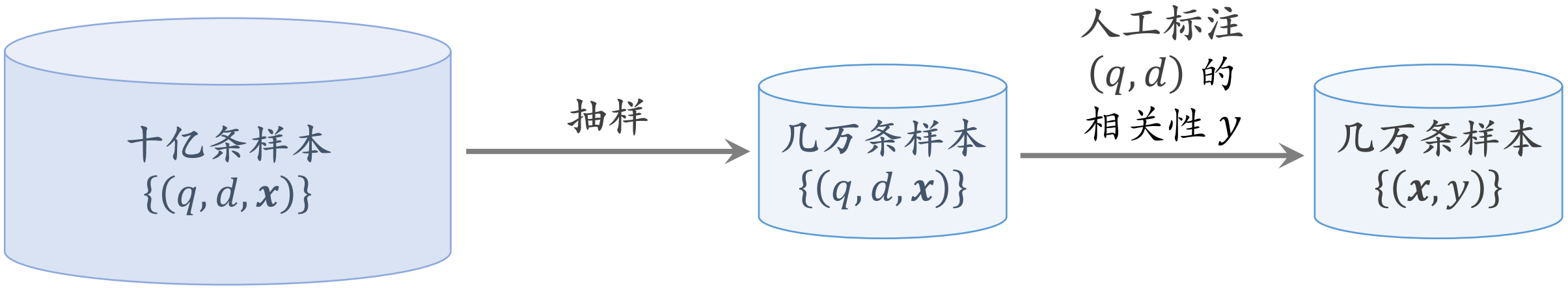
步骤2：自动生成相关性分数

- 已经得到十亿条样本 $(q, d, \boldsymbol{x})$，其中向量 $\boldsymbol{x}$ 是用户行为。
- 相关性 y 与 $\boldsymbol{x}$ 存在某种函数关系。（相关性越高，则用户越有可能点击和交互。）
- 找出 y 与 $\boldsymbol{x}$ 的函数关系：$\hat{y} = t(\boldsymbol{x})$。
- 小模型 t 只能使用点击率、交互率作为输入。
  - 尽量不使用文本特征作为输入。
  - 绝对不能用相关性 BERT 模型打分作为输入，否则会产生反馈回路。（BERT 模型打分 → 训练小模型 t → 小模型 t 生成数据 → 训练 BERT 模型）

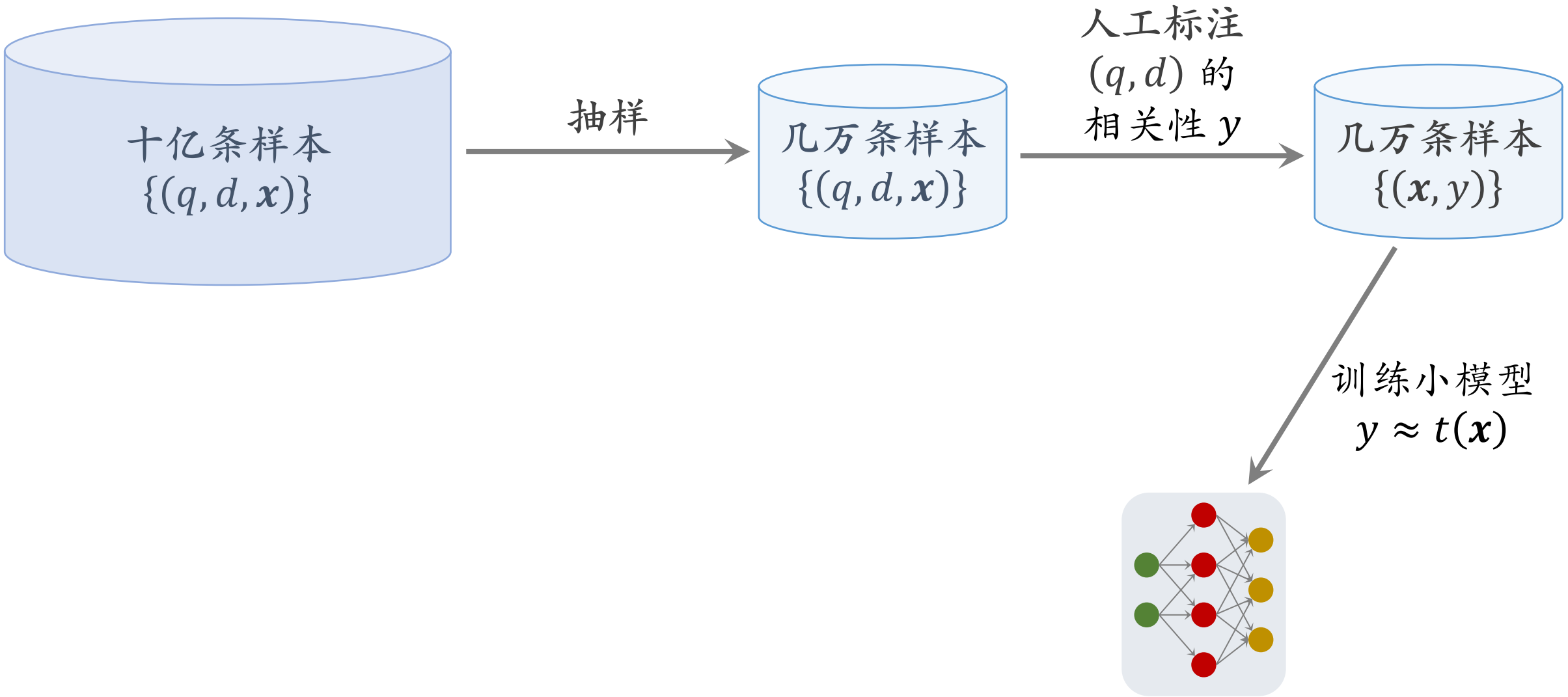
步骤2：自动生成相关性分数

- 已经得到十亿条样本 $(q, d, \boldsymbol{x})$，其中向量 $\boldsymbol{x}$ 是用户行为。
- 相关性 y 与 $\boldsymbol{x}$ 存在某种函数关系。（相关性越高，则用户越有可能点击和交互。）
- 找出 y 与 $\boldsymbol{x}$ 的函数关系：$\hat{y} = t(\boldsymbol{x})$。
- 小模型 t 只能使用点击率、交互率作为输入。
- 对于所有十亿条样本 $(q, d, \boldsymbol{x})$，用训练好的小模型打分 $\hat{y} = t(\boldsymbol{x})$，得到十亿条样本 $(q, d, \hat{y})$。

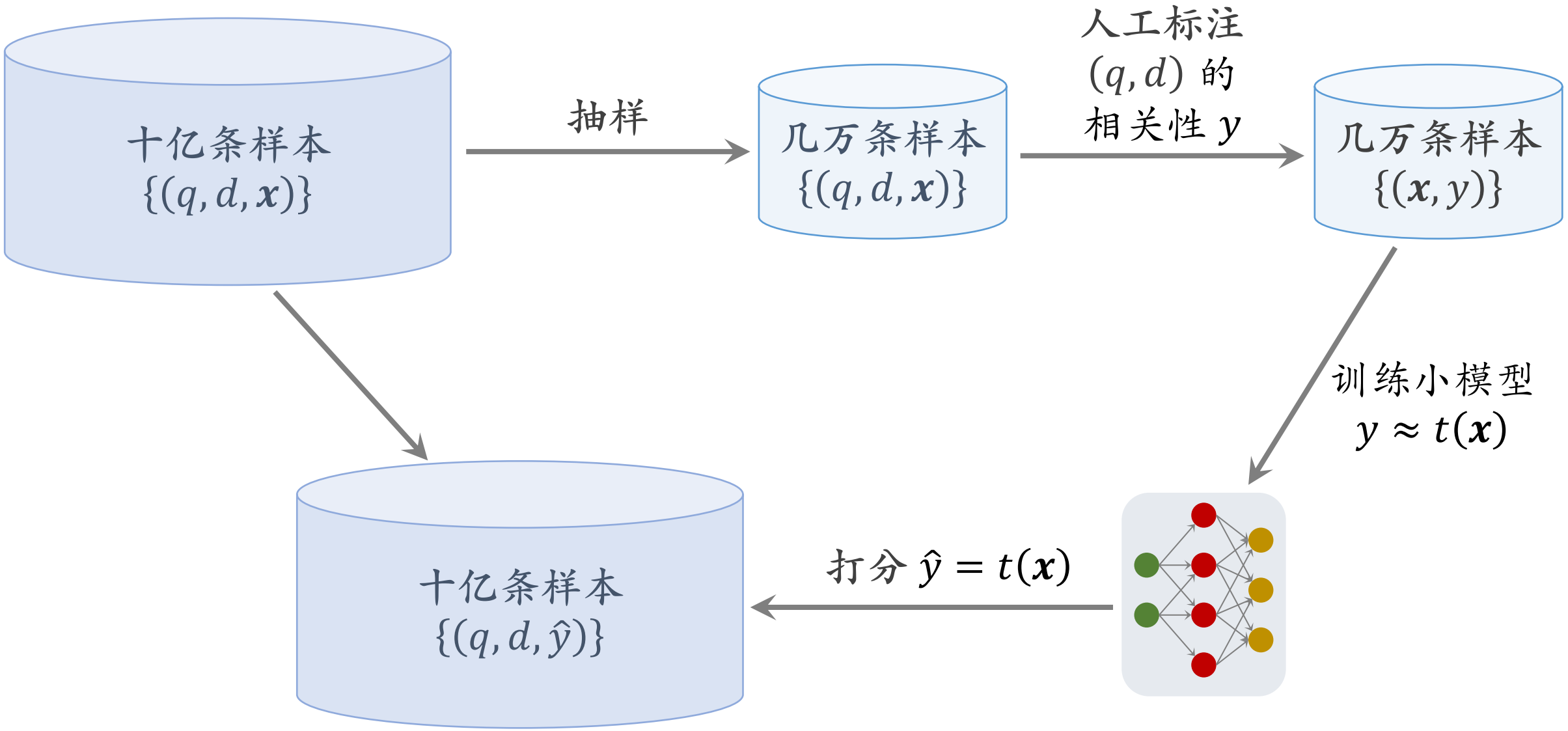
步骤1 + 步骤2


十亿条样本
$\{(q, d, \boldsymbol{x})\}$
抽样
几万条样本
$\{(q, d, \boldsymbol{x})\}$
人工标注
(q, d) 的
相关性 y
几万条样本
$\{(\boldsymbol{x}, y)\}$

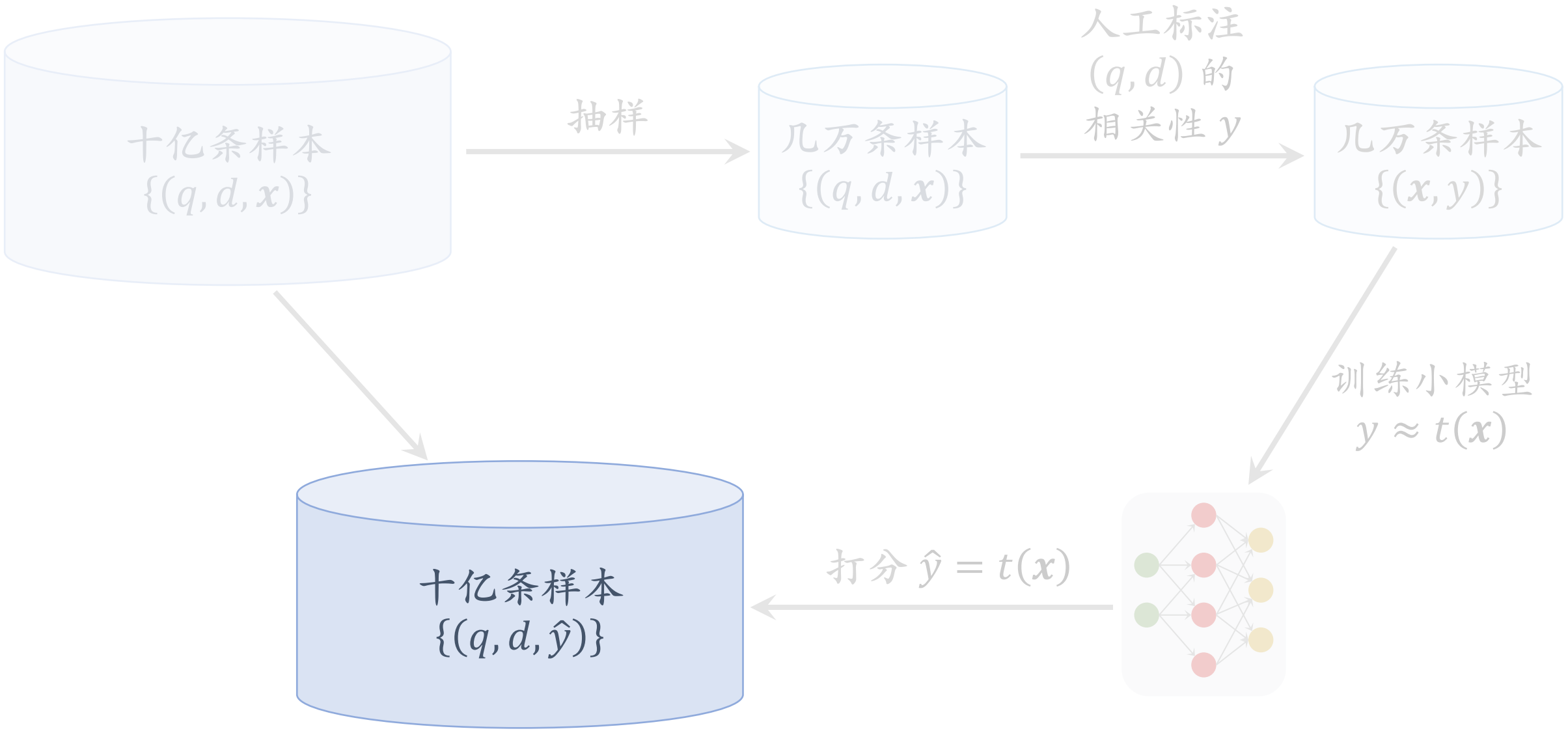
步骤1 + 步骤2


十亿条样本
$\{(q,d,\boldsymbol{x})\}$
抽样
几万条样本
$\{(q,d,\boldsymbol{x})\}$
人工标注
(q,d) 的
相关性 y
几万条样本
$\{(\boldsymbol{x},y)\}$
训练小模型
$y \approx t(\boldsymbol{x})$

步骤1 + 步骤2


十亿条样本
$\{(q,d,\boldsymbol{x})\}$
抽样
几万条样本
$\{(q,d,\boldsymbol{x})\}$
人工标注
(q,d) 的
相关性 y
几万条样本
$\{(\boldsymbol{x},y)\}$
训练小模型
$y \approx t(\boldsymbol{x})$
十亿条样本
$\{(q,d,\hat{y})\}$
打分 $\hat{y} = t(\boldsymbol{x})$

步骤1 + 步骤2


十亿条样本
$\{(q,d,x)\}$
抽样
几万条样本
$\{(q,d,x)\}$
人工标注
(q,d) 的
相关性 y
几万条样本
$\{(x,y)\}$
训练小模型
$y \approx t(x)$
十亿条样本
$\{(q,d,\hat{y})\}$
打分 $\hat{y} = t(x)$

步骤3：用生成的数据训练模型

- 前两步得到十亿条样本 $(q, d, \hat{y})$，其中 $\hat{y}$ 是自动生成的相关性分数。
- 基于预训练的 BERT 模型，用 $(q, d, \hat{y})$ 做监督学习。
- 监督学习同时用 3 个任务，取 3 个损失函数的加权和。
  - 回归任务，起到“保值”的作用（模型的输出尽量接近 $\hat{y}$），有利于 AUC 指标。
  - 排序任务，起到“保序”的作用（鼓励正序对、惩罚逆序对），有利于正逆序比指标。
  - 预训练的 MLM 任务，避免清洗掉预训练的结果。

后预训练

后预训练为什么有效？

- 大幅增加了有标签样本数量（百万 → 十亿）。
  - 人工标注的相关性数据只有几十万到几百万条 (q, d, y)。
  - 后预训练使用十亿条 $(q, d, \hat{y})$。
  - 巨大的数据量使模型更准确。

后预训练

后预训练为什么有效？

- 大幅增加了有标签样本数量（百万 → 十亿）。
- 用户行为 $\boldsymbol{x}$ 与相关性 y 有很强关联。
  - (q, d) 的相关性越高，越有可能得到点击和交互。
  - 小模型可以根据点击率和交互率 $\boldsymbol{x}$ 较为准确地推断 y。
  - 小模型生成的标签 $\hat{y}$ 虽然有噪声，但也有很大的信息量。

蒸馏 (distillation)

为什么做蒸馏？

- 用户每搜一个查询词，排序需要用相关性 BERT 模型给数百、数千对 (q, d) 打分。
- BERT 模型越大，计算量越大，给相关性的打分越准。
- 精排常用 4~12 层交叉 BERT，粗排常用 2~4 层交叉 BERT（或双塔 BERT）。
- 两种方法谁更好？
  - 直接训练训练小模型（2~12 层）。
  - 先训练 48 层大模型，再蒸馏小模型。

为什么做蒸馏？

- 先训练 48 层 BERT 作为 teacher，再蒸馏小模型，效果优于直接训练小模型。
- 工业界经验：
  - 48 层对比 12 层，AUC 高 2% 以上。
  - 48 层蒸馏 12 层，AUC 几乎无损。
  - 48 层蒸馏 4 层，AUC 损失 0.5%。

怎么样做蒸馏？

- 做预训练、后预训练、微调，训练好 48 层 BERT 大模型，作为 teacher。
  - Teacher 模型越大，它本身越准确，蒸馏出的 student 也越准确。
  - 48 层 teacher，效果优于 24 层和 12 层 teacher。

怎么样做蒸馏？

- 做预训练、后预训练、微调，训练好 48 层 BERT 大模型，作为 teacher。
- 准备几亿对 (q, d)，用 teacher 给 (q, d) 打分 $\tilde{y}$。
  - 蒸馏的数据量越大越好。
  - 数据量少于 1 亿，蒸馏会损失较大 AUC。
  - 数据量超过 10 亿，边际效益很小。

怎么样做蒸馏？

- 做预训练、后预训练、微调，训练好 48 层 BERT 大模型，作为 teacher。
- 准备几亿对 (q, d)，用 teacher 给 (q, d) 打分 $\tilde{y}$。
- 在数据 $(q, d, \tilde{y})$ 上做监督学习训练小模型。
  - 只训练 1 epoch。（1 亿条样本上训练 2 epoch，效果不如 2 亿条样本上训练 1 epoch。）
  - 与微调相同，同时用回归任务、排序任务。

蒸馏：一些有效的技巧

- Student 小模型要先预热、再蒸馏。
  - 预热：先做预训练、后预训练、微调训练 student。（与训练 teacher 的步骤相同。）
  - 基于预热的模型，用蒸馏数据 $(q, d, \tilde{y})$ 训练 student。

蒸馏：一些有效的技巧

- Student 小模型要先预热、再蒸馏。
- 不要做逐层蒸馏！
  - 逐层蒸馏：让 student 的中间层拟合 teacher 的中间层。
  - 用相同的算力，直接拟合 $\tilde{y}$ 优于逐层蒸馏。

蒸馏：一些有效的技巧

- Student 小模型要先预热、再蒸馏。
- 不要做逐层蒸馏！
- 多级蒸馏和单级蒸馏谁更好？
  - 多级蒸馏： 48层 → 12层 → 4层。
  - 单级蒸馏： 48层 → 4层。
  - 有争议，可能是单级蒸馏更好。

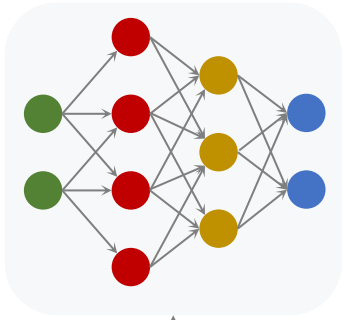
总结

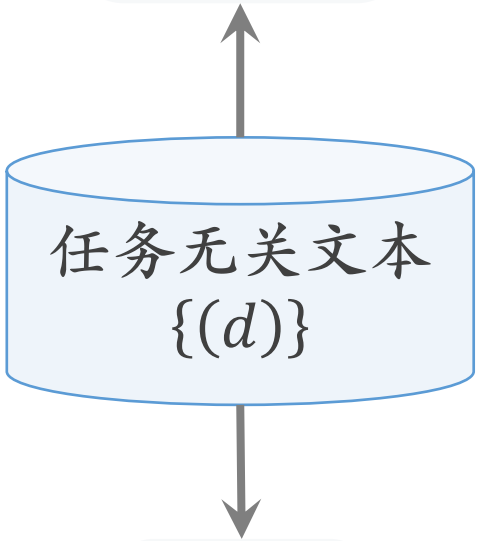
相关性模型的训练

- **预训练** (pretrain)：用 MLM 等任务在海量文本数据上训练 BERT 模型。
- **后预训练** (post pretrain)：将用户行为 $\boldsymbol{x}$ 映射到相关性标签 $\hat{y}$，构造十亿条样本 $(q, d, \hat{y})$，继续训练 BERT 模型。
- **微调** (fine tuning)：用人工标注的相关性数据训练模型。
- **蒸馏** (distillation)：先训练大模型，用大模型标注几亿条样本 $(q, d, \tilde{y})$，用这些样本训练小模型。

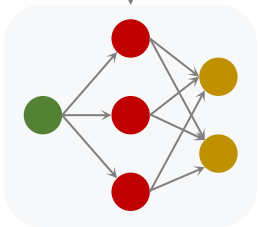
预训练 后预训练 微调 蒸馏

大模型
(teacher)



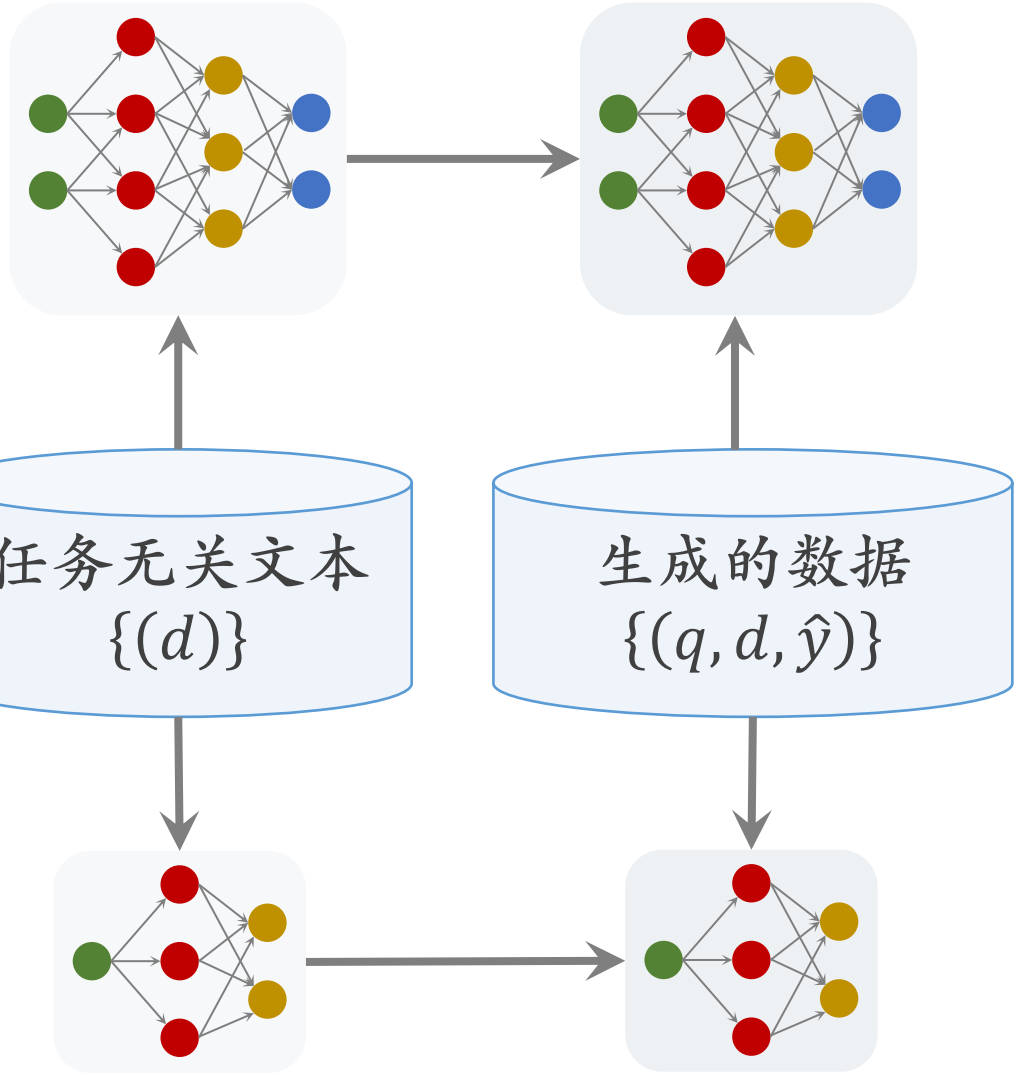


小模型
(student)

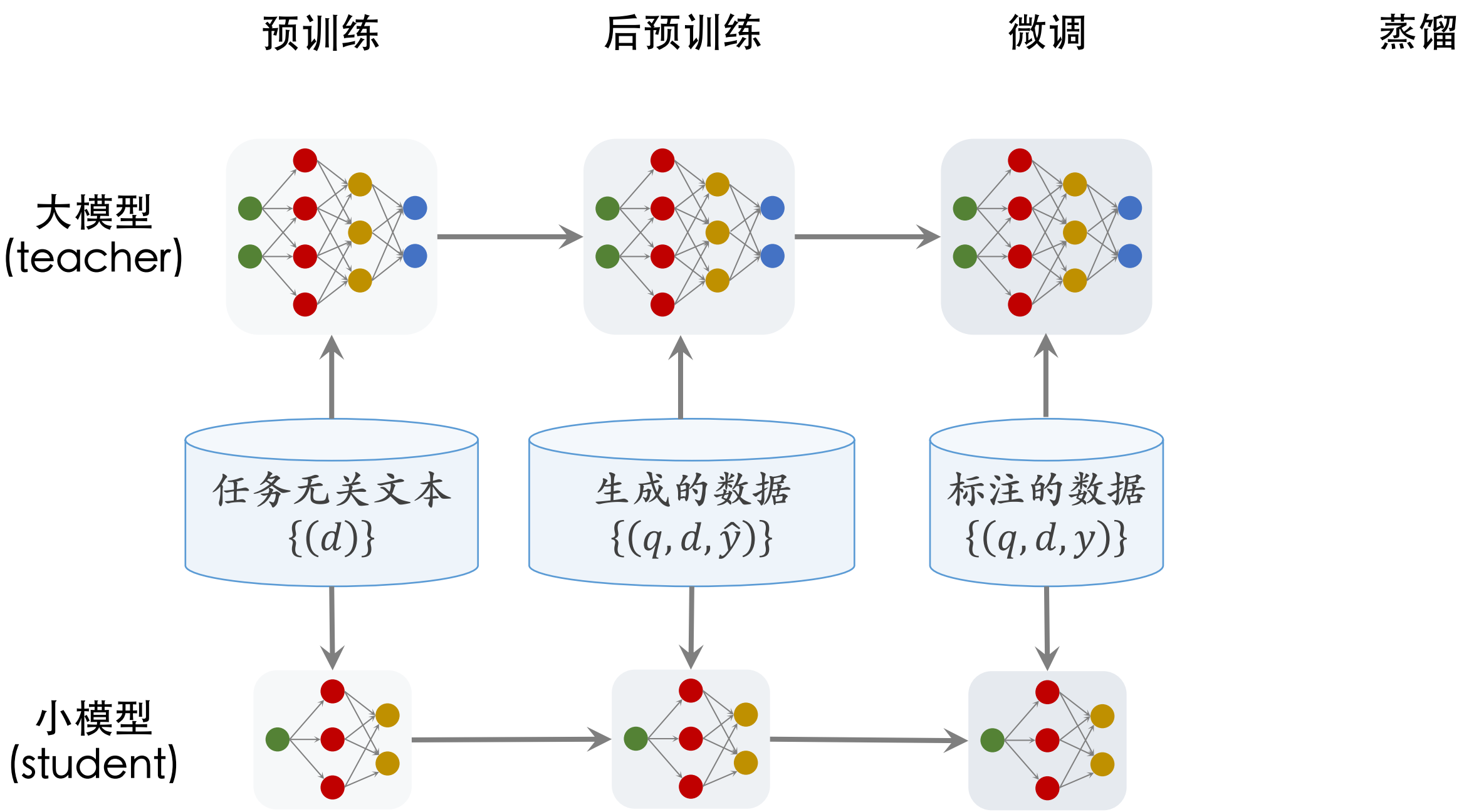


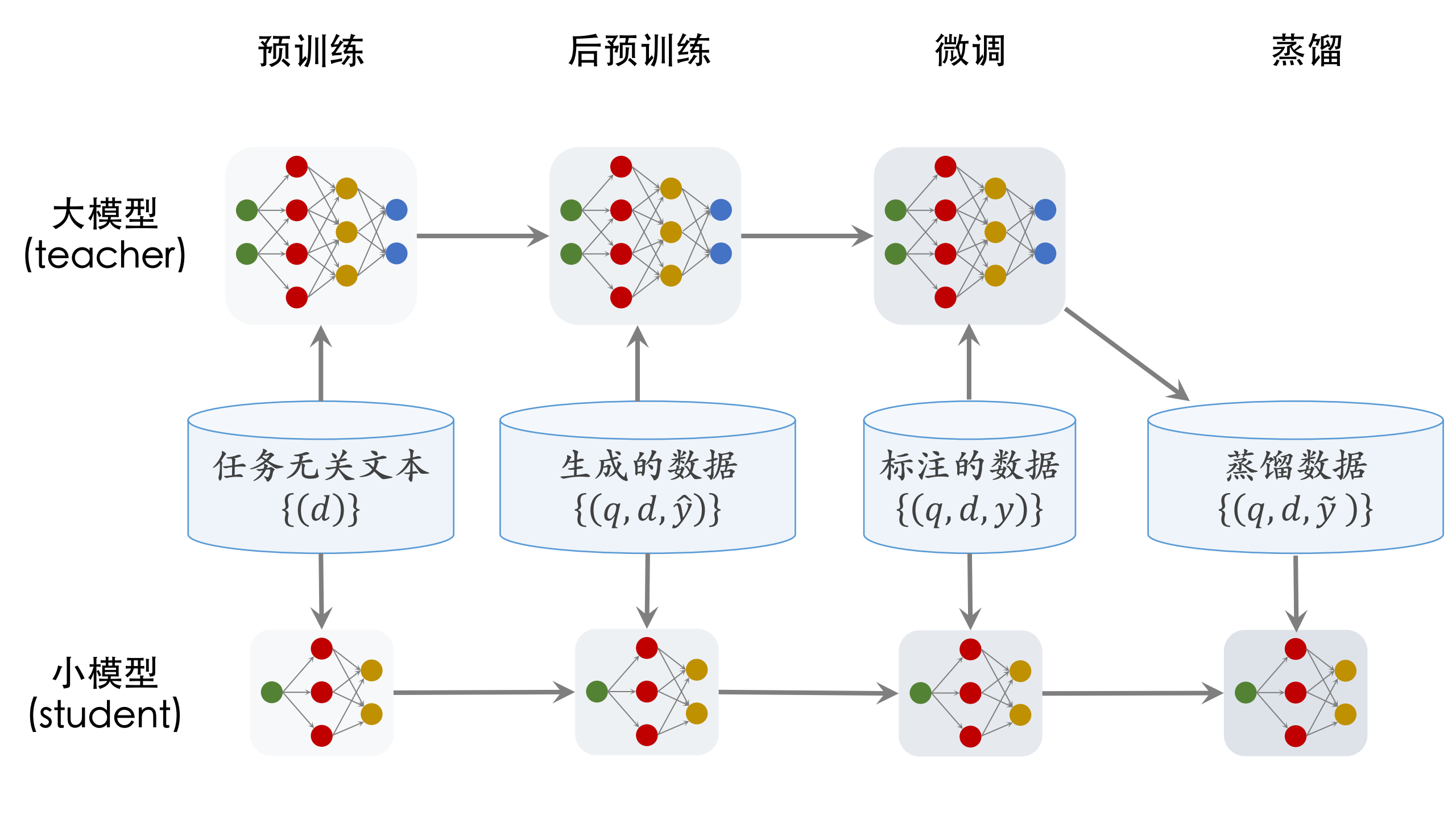
预训练 后预训练 微调 蒸馏

大模型
(teacher)



小模型
(student)


预训练
后预训练
微调
蒸馏
大模型
(teacher)
小模型
(student)
任务无关文本
{(d)}
生成的数据
{(q, d, ŷ)}
标注的数据
{(q, d, y)}


预训练
后预训练
微调
蒸馏
大模型
(teacher)
小模型
(student)
任务无关文本
{(d)}
生成的数据
{(q, d, ŷ)}
标注的数据
{(q, d, y)}
蒸馏数据
{(q, d, ỹ)}

Thank You!

http://wangshusen.github.io/