自己的小语言模型：从计数到生成

建议 35-45 分钟；先用纸笔预测，再由老师投影实验核对。本地模型课需要电脑运行下载的代码。进阶运算可分两节课。

学习目标

实际运行一个词 n-gram 模型，在固定测试集上比较上下文长度并解释未见词。

先修与准备

先学猜词、模型评估；本地需 Python 3.10+，不需第三方库或 GPU。

材料

下载本页 Python 文件；普通电脑即可。内置语料为本项目原创短句，也可使用自己有权使用的文本。

课堂顺序（5 / 10 / 10 / 10 分钟，可另留 5-10 分钟讨论）

1. 先猜：手数“the cat naps”两次、“the cat plays”一次。只看 cat 后一个词，不平滑时 naps 的频率是多少？

2. 动手：在终端运行 python3 tiny_language_model.py --context 1 --seed 7。读词表、候选词概率、训练与测试每词负对数似然，再看生成句。

3. 观察：改为 --context 2，保持语料、留出句和种子不变。更多上下文更稀疏，不保证损失更低。如果据此挑设置，这份留出数据就参与了验证；最终评价要另留新测试。

4. 挑战：用 --train my_train.txt --test my_test.txt，一行一句，先分好数据再看结果。只改种子不会重新学习，主要改变抽样结果。

自己的小语言模型：从计数到生成

做好的例子

不平滑时 P(naps|cat)=2/3。下载程序使用 add-one 平滑和训练词表，因此程序的概率还包含每个候选加 1 以及 <unk>、结束符的计数。

理解检查：先让学生独立回答

生成句读起来很顺，能证明测试损失低或事实正确吗？把测试文本也拿去计数会怎样？

参考答案与学习单核对

都不能；只用训练集学词表和计数，测试词表外词映射到 <unk>。NLL 越低表示这组测试后续词的平均负对数概率更低（几何平均概率更高），不能当作真实知识准确率。重复查看测试来挑参数也会泄漏。 手算例中 plays 的概率为 1/3，未见后续词为 0，若测试命中该零概率，NLL 无穷大。程序内置 12 句比手算多：cat 后 naps/plays/runs 的计数为 3/2/1，词表大小 11；平滑后 P(naps|cat)=4/17≈0.2353。可查看 TRAIN 逐项核对。

不要把结论讲大了

这是计数型 n-gram 基线，不是神经网络。没有下载外部模型、联网推理或 GPU。下一步的本地 Transformer 训练见页面进阶点子，需另行准备依赖和更完整评估。

教学调整

需要帮助时：先只做例子，由老师朗读题目，用实物或纸格代替公式。已经掌握时：一次只改一个条件，先记预测，再验证。遇到不一样的答案，先查规则、编码或数据，不先判学生错。

带回家的活动

和家长各写 6 句训练、3 句新测试；先固定划分，再比较 context 1 和 2。

接着学

Transformer：一次注意力怎么算？ / 满分之后：模型真的学会了吗？ / 猜词游戏：AI 怎么读懂一句话

教师参考（完整链接也在课程页）

Jurafsky & Martin: N-gram Language Models: https://web.stanford.edu/~jurafsky/slp3/3.pdf