

**Embedding（嵌入）** 是将离散数据（如文本、类别标签、用户 ID 等）映射到低维稠密向量的技术，核心作用是将高维稀疏的离散信息转化为模型可高效处理的低维稠密特征，同时保留数据的语义或关联关系。

**主要作用：**

降维与稠密化：解决离散数据（如 One-hot 编码）的 “维度爆炸” 问题（例如 10 万词汇的 One-hot 向量是 10 万维），转化为几十到几百维的稠密向量，降低计算成本。

捕捉语义关联：相似的离散值会映射到相似的向量（如 “猫” 和 “狗” 的嵌入向量距离更近，“国王”−“男人”≈“女王”−“女人”），让模型理解数据的内在关系。

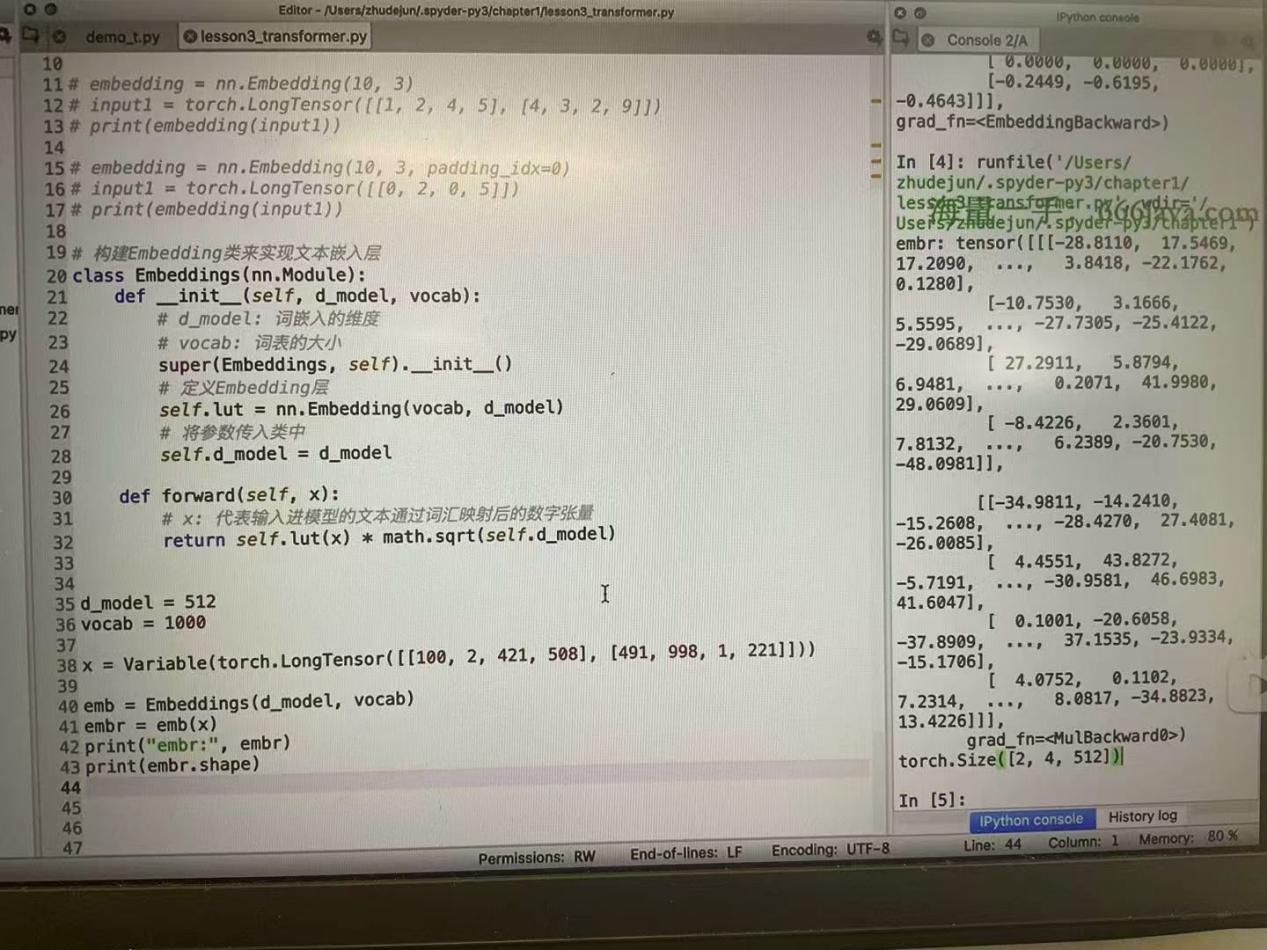
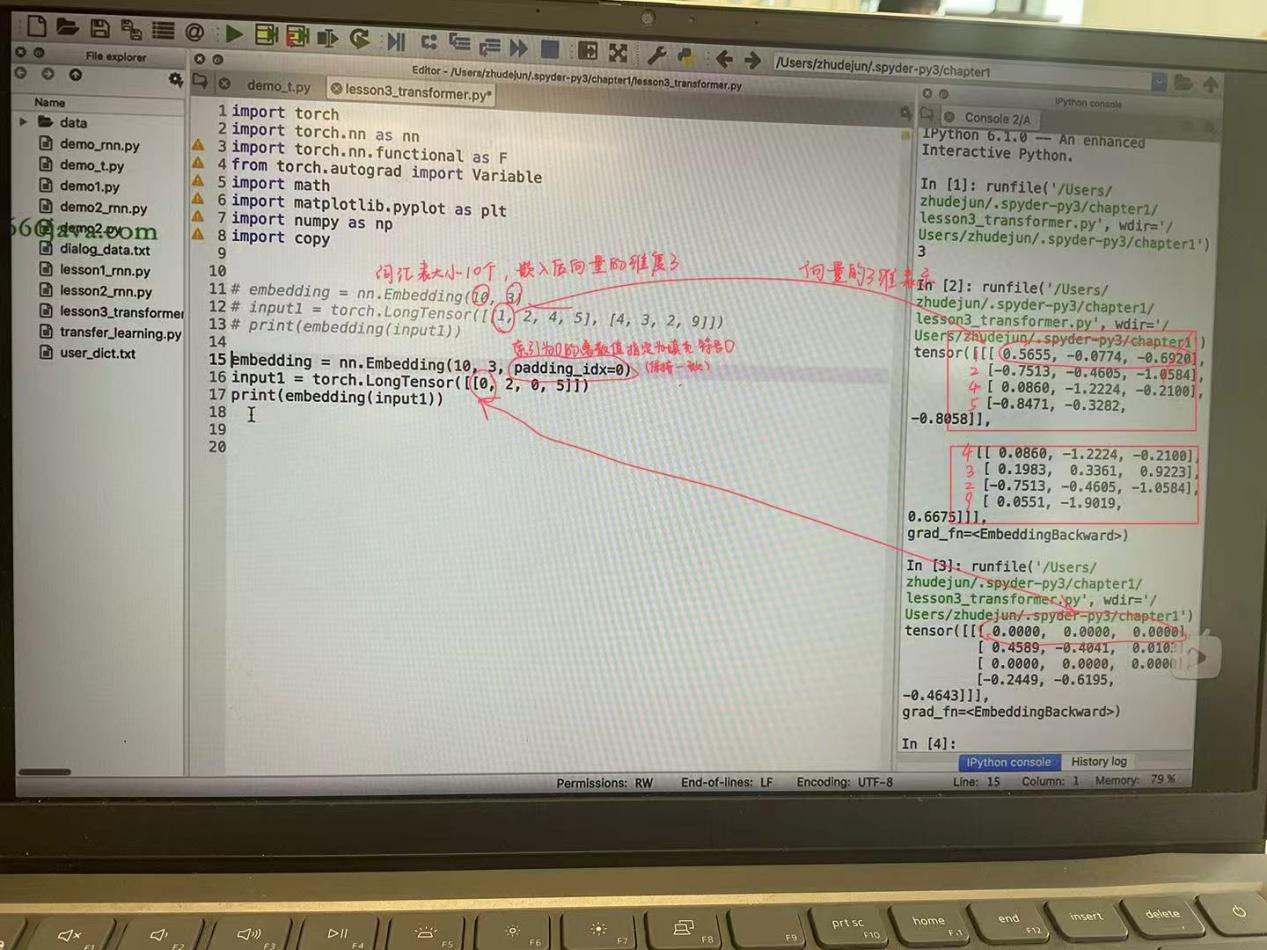
适配模型输入：作为桥梁连接离散数据与深度学习模型（如神经网络、Transformer），使其能处理非数值型输入。

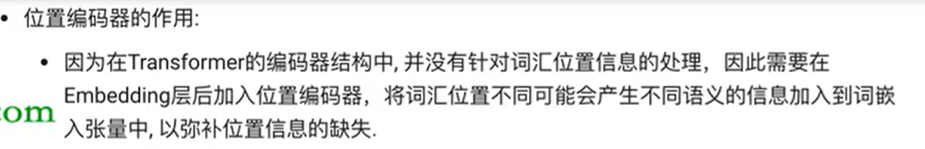
**常见应用场景：**

自然语言处理（NLP）：将单词、句子转化为向量（如 Word2Vec、BERT 嵌入）。

推荐系统：将用户 ID、物品 ID 转化为向量，通过向量相似度推荐物品。

计算机视觉（CV）：将类别标签、图像特征转化为嵌入向量用于检索或分类。





在序列建模（如自然语言处理、时序数据处理）中，**位置编码器（Positional Encoding）** 是一种向模型注入序列位置信息的技术。其核心作用是让模型理解序列中元素的顺序关系（如文本中单词的先后顺序），因为像 Transformer 这样的模型本身是 “无状态” 的（通过自注意力机制并行处理序列，不依赖递归或卷积的顺序性），需要显式加入位置信息才能捕捉时序特征。

**为什么需要位置编码器？**

以 Transformer 为例，其自注意力机制对序列中所有元素的处理是并行的，无法像 RNN 那样天然感知元素的位置顺序。

若忽略位置信息，模型会将 “我爱你” 和 “你爱我” 视为相同序列，显然不合理。

位置编码器通过给不同位置的元素添加独特的 “位置特征”，让模型能区分元素的先后顺序。

**常见实现方式**

1. **正弦余弦位置编码（Transformer 原论文方案）**

这是最经典的实现，通过正弦和余弦函数生成位置向量，公式如下：对于序列中位置为 pos（从 0 开始）、维度为 i（从 0 开始）的位置编码：

当 i 为偶数时：PE(pos, i) = sin(pos / 10000^(2i/d\_model))

当 i 为奇数时：PE(pos, i) = cos(pos / 10000^(2i/d\_model))

其中，d\_model 是嵌入向量的维度（与输入序列的特征维度一致）。

核心特点：

位置向量与输入嵌入向量逐元素相加，最终输入模型的是 “嵌入向量 + 位置向量”。

正弦余弦函数的周期性确保：

不同位置有独特的编码（便于模型区分）；

位置差固定的两个位置，其编码具有固定的数学关系（便于模型学习相对位置）。

2. **可学习位置编码**

直接将位置编码定义为可训练的参数矩阵（形状为 [max\_seq\_len, d\_model]），与输入嵌入向量相加后参与模型训练。

优点：实现简单，模型可自主学习更适合任务的位置特征。

缺点：泛化能力可能受限（对长于训练序列的位置编码效果较差）。

**dropout**是按比率扔掉一部分，但不是这些就消失了，是变成对结果没影响的0

