特别加餐｜TensorFlow的模型离线评估实践怎么做？

2020-12-11 王喆

深度学习推荐系统实战 进入课程 >

讲述：王喆
时长 09:16 大小 8.50M

你好，我是王喆。

上两节课，我们学习了离线评估的主要方法以及离线评估的主要指标。那这些方法和指标具体是怎么使用的，会遇到哪些问题呢？我们之前实现的深度学习模型的效果怎么样呢？

这节课，我们直接进入实战，在 TensorFlow 环境下评估一下我们之前实现过的深度学习模型。一方面这能帮助我们进一步加深对离线评估方法和指标的理解，另一方面，也能检验一下我们自己模型的效果。

训练集和测试集的生成

离线评估的第一步就是要生成训练集和测试集，在这次的评估实践中，我会选择最常用的 Holdout 检验的方式来划分训练集和测试集。划分的方法我们已经在第 23 课里用 Spark 实现过了，就是调用 Spark 中的 randomSplit 函数进行划分，具体的代码你可以参考 FeatureEngForRecModel 对象中的 splitAndSaveTrainingTestSamples 函数。

这里我们按照 8:2 的比例把全量样本集划分为训练集和测试集，再把它们分别存储在`SparrowRecSys/src/main/resources/webroot/sampledata/trainingSamples.csv`和`SparrowRecSys/src/main/resources/webroot/sampledata/testSamples.csv`路径中。

在 TensorFlow 内部，我们跟之前载入数据集的方式一样，调用`get_dataset`方法分别载入训练集和测试集就可以了。

TensorFlow 评估指标的设置

在载入训练集和测试集后，我们需要搞清楚如何在 TensorFlow 中设置评估指标，并通过这些指标观察模型在每一轮训练上的效果变化，以及最终在测试集上的表现。这个过程听起来还挺复杂，好在，TensorFlow 已经为我们提供了非常丰富的评估指标，这让我们可以在模型编译阶段设置 metrics 来指定想要使用的评估指标。

具体怎么做呢？我们一起来看看下面的代码，它是设置评估指标的一个典型过程。首先，我们在 model complie 阶段设置准确度（Accuracy）、ROC 曲线 AUC（tf.keras.metrics.AUC(curve='ROC')）、PR 曲线 AUC（tf.keras.metrics.AUC(curve='PR')），这三个在评估推荐模型时最常用的指标。

同时，在训练和评估过程中，模型还会默认产生损失函数 loss 这一指标。在模型编译时我们采用了 binary_crossentropy 作为损失函数，所以这里的 Loss 指标就是我们在上一节课介绍过的二分类问题的模型损失 Logloss。

在设置好评估指标后，模型在每轮 epoch 结束后都会输出这些评估指标的当前值。在最后的测试集评估阶段，我们可以调用 model.evaluate 函数来生成测试集上的评估指标。具体的实现代码，你可以参考 SparrowRecsys 项目中深度推荐模型相关的代码。

```
# compile the model, set loss function, optimizer and evaluation metrics
model.compile(
    loss='binary_crossentropy',
    optimizer='adam',
    metrics=['accuracy', tf.keras.metrics.AUC(curve='ROC'), tf.keras.metrics.A
# train the model
model.fit(train_dataset, epochs=5)
# evaluate the model
test_loss, test_accuracy, test_roc_auc, test_pr_auc = model.evaluate(test_data
```

复制代码

在执行这段代码的时候，它的输出是下面这样的。从中，我们可以清楚地看到每一轮训练的 Loss、Accuracy、ROC AUC、PR AUC 这四个指标的变化，以及最终在测试集上这四个指标的结果。

复制代码

```
Epoch 1/5
8236/8236 [==============================] - 60s 7ms/step - loss: 3.0724 - acc
Epoch 2/5
8236/8236 [==============================] - 55s 7ms/step - loss: 0.6291 - acc
Epoch 3/5
8236/8236 [==============================] - 56s 7ms/step - loss: 0.5555 - acc
Epoch 4/5
8236/8236 [==============================] - 56s 7ms/step - loss: 0.5263 - acc
Epoch 5/5
8236/8236 [==============================] - 56s 7ms/step - loss: 0.5071 - acc


1000/1000 [==============================] - 5s 5ms/step - loss: 0.5198 - accu
Test Loss 0.5198314250707626, Test Accuracy 0.7426666617393494, Test ROC AUC 0

```

总的来说，随着训练的进行，模型的 Loss 在降低，而 Accuracy、Roc AUC、Pr AUC 这几个指标都在升高，这证明模型的效果随着训练轮数的增加在逐渐变好。

最终，我们就得到了测试集上的评估指标。你会发现，测试集上的评估结果相比训练集有所下降，比如 Accuracy 从 0.7524 下降到了 0.7427，ROC AUC 从 0.8256 下降到了 0.8138。这是非常正常的现象，因为模型在训练集上都会存在着轻微过拟合的情况。

如果测试集的评估结果相比训练集出现大幅下降，比如下降幅度超过了 5%，就说明模型产生了非常严重的过拟合现象，我们就要反思一下是不是在模型设计过程中出现了一些问

题，比如模型的结构对于这个问题来说过于复杂，模型的层数或者每层的神经元数量过多，或者我们要看一看是不是需要加入 Dropout，正则化项来减轻过拟合的风险。

除了观察模型自己的效果，在模型评估阶段，我们更应该重视不同模型之间的对比，这样才能确定我们最终上线的模型，下面我们就做一个模型效果的横向对比。

模型的效果对比

在推荐模型篇，我们已经实现了 EmbeddingMLP、NerualCF、Wide&Deep 以及 DeepFM 这四个深度学习模型，后来还有同学添加了 DIN 的模型实现。

那接下来，我们就利用这节课的模型评估方法，来尝试对比一下这几个模型的效果。首先，我直接把这些模型在测试集上的评估结果记录到了表格里，当然，我更建议你利用 SparrowRecsys 项目中的代码，自己来计算一下，多实践一下我们刚才说的模型评估方法。

模型	Loss	Accuracy	ROC AUC	PR AUC
Embedding MLP	0.6129	0.6922	0.7534	0.7828
NeuralCF	0.6697	0.6788	0.7321	0.7556
Wide&Deep	0.6044	0.6907	0.7526	0.7800
DeepFM	0.7715	0.6425	0.6916	0.7242

通过上面的比较，我们可以看出，Embedding MLP 和 Wide&Deep 模型在我们的 MovieLens 这个小规模数据集上的效果最好，它们两个的指标也非常接近，只不过是在不同指标上有细微的差异，比如模型 Loss 指标上 Wide&Deep 模型好一点，在 Accuracy、ROC AUC、PR AUC 指标上 Embedding MLP 模型好一点。

遇到这种情况，我们该如何挑出更好的那个模型呢？一般我们会在两个方向上做尝试：一是做进一步的模型调参，特别是对于复杂一点的 Wide&Deep 模型，我们可以尝试通过参

数的 Fine Tuning（微调）让模型达到更好的效果；二是如果经过多次尝试两个模型的效果仍比较接近，我们就通过线上评选出最后的胜出者。

说完了效果好的指标，不知道你有没有注意到一个反常的现象，那就是模型 DeepFM 的评估结果非常奇怪，怎么个奇怪法呢？理论上来说，DeepFM 的表达能力是最强的，可它现在展示出来的评估结果却最差。这种情况就极有可能是因为模型遇到了过拟合问题。为了验证个想法，我们再来看一下 DeepFM 在训练集上的表现，如下表所示：

DeepFM模型	Loss	Accuracy	ROC AUC	PR AUC
训练集结果	0.3687	0.8348	0.9136	0.9298
测试集结果	0.7715	0.6425	0.6916	0.7242

我们很惊讶地发现，DeepFM 在测试集上的表现比训练集差了非常多。毫无疑问，这个模型过拟合了。当然，这里面也有我们数据的因素，因为我们采用了一个规模很小的采样过的 MovieLens 数据集，在训练复杂模型时，小数据集往往更难让模型收敛，并且由于训练不充分的原因，模型中很多参数其实没有达到稳定的状态，因此在测试集上的表现往往会呈现出比较大的随机性。

通过 DeepFM 模型效果对比的例子，也再一次印证了我们在“最优的模型结构该怎么找？”那节课的结论：推荐模型没有银弹，每一个业务，每一类数据，都有最合适的模型结构，并不是说最复杂的，最新的模型结构就是最好的模型结构，我们需要因地制宜地调整模型和参数，这才是算法工程师最大的价值所在。

小结

这节实践课，我们基于 TensorFlow 进行了深度推荐模型的评估，整个实践过程可以分成三步。

第一步是导入 Spark 分割好的训练集和测试集。

第二步是在 TensorFlow 中设置评估指标，再在测试集上调用 model.evaluate 函数计算这些评估指标。在实践过程中，我们要清楚有哪些 TensorFlow 的指标可以直接调用。那在这节课里，我们用到了最常用的 Loss、Accuracy、ROC AUC、PR AUC 四个指标。

第三步是根据四个深度推荐模型的评估结果，进行模型效果的对比。通过对比的结果我们发现 Embedding MLP 和 Wide&Deep 的效果是最好的。同时，我们也发现，本该表现很好的 DeepFM 模型，它的评估结果却比较差，原因是模型产生了非常严重的过拟合问题。

因此，在实际工作中，我们需要通过不断调整模型结构、模型参数，来找到最合适当前业务和数据集的模型。

课后思考

1. 除了这节课用到的 Loss、Accuracy、ROC AUC、PR AUC 这四个指标，你在 TensorFlow 的实践中还会经常用到哪些评估指标呢？ 你能把这些常用指标以及它们特点分享出来吗？（你可以参考 TensorFlow 的官方 Metrics 文档 ）

2. 你认为 DeepFM 评估结果这么差的原因，除了过拟合，还有什么更深层次的原因呢？可以尝试从模型结构的原理上给出一些解释吗？

期待在留言区看到你对 DeepFM 模型的思考和使用评估指标的经验，我们下节课见！

提建议

上一篇 25 | 评估指标：我们可以用哪些指标来衡量模型的好坏？

下一篇 26 | 在线测试：如何在推荐服务器内部实现A/B测试？

精选留言 (2)

写留言

Geek_e642b8
2020-12-11

第二个问题是FM部分使用两两点积的缘故吗？

展开

作者回复: 我猜测是因为交叉层的数据太稀疏了，不能够让交叉层完全收敛。

另外交叉层大量使用id类特征，测试集的id特征和训练集的id特征重叠比较少的话，很可能无法作出合理的预测。这也是所谓模型泛化性和记忆性的矛盾。

1

Eio
2020-12-14

有一个问题困扰我很久，请问老师，显式特征交互（比如cin）与隐式特征交互（普通的D

NN）的区别，以及特征交互发生在特征向量之间（vector-wise），而不是发生在元素级（bit-wise level）的优点。

作者回复: 第一个问题我怎么有点白讲了推荐模型篇的感觉。我应该多次在课程中介绍过这个问题，推荐多复习。

第二个问题也谈不上有什么优点缺点，他们就是两个互操作的方法，只是一般来说element wise操作由于生成的是向量，所以理论上表达能力更强一些，实践中当然需要你都尝试一下取效果好的那个。