[学习] 关于Wide&Deep的一些问题记录

原创 备忘录 菜鸟的机器学习 4月8日

作者：备忘录

原文：请点击左下角"阅读原文"

下面是关于Wide&Deep模型中的一些小问题，特意记录下，加深理解。

问题一：为什么Wide&Deep模型用ftrl和adagrad两种优化方法?

链接：https://www.zhihu.com/question/343744544/answer/991335952

答案一（张备）：

当时看论文的时候印象深刻，wide部分是用ftrl做的优化，wide部分可以看作传统的lr，用ftrl优化可以得到稀疏权重，从而降低serving的复杂度。

deep部分是神经网络，似乎没办法采用ftrl进行优化（笔者注：这块有争议），而且权重是否稀疏也不重要。

虽然近几年推荐算法领域各种深度学习模型和网络结构层出不穷，但是我发现wdl目前依然非常流行，我自己的看法是，之前主流算法还是lr+ftrl，wide&deep优势就是可以复用之前的特征和lr。

总结一句：wide部分可看作传统的lr，使用ftrl可得到稀疏权重，降低计算复杂度；deep部分是神经网络，稀疏性不重要，即使稀疏也无非降低计算复杂度。

答案二（zzszmyf）：

ftrl优化算法，需要考虑之前每一轮训练的梯度和。deep部分是是神经网络，这种高度非凸模型中，模型已经从一个local basin迭代到另一个local basin，过去的样本为模型提供的信息少于最近的样本提供的信息，在推荐&&广告这种数据分布变化比较激烈的环境下，ftrl优化算法用于deep model的训练，可能会影响模型对数据分布的适应。

答案三（代成雷）：

这个问题可以从优化的方向上去理解，两个结构最后线性组合，当用2种优化时适当的情况下，可以保证比单优化更收敛且平衡。具体细节等待其他大神去推导。

答案四（浅海拾贝）：

个人理解学习率上ftrl和adgrad两者有相似之处，都是学习率跟梯度累积成反比，更新越频繁学习率越低，也都有学习率最终降低到0的问题。

但ftrl收敛更慢，这是因为二阶正则希望离上一轮weight不要太远。

从网络结构上看，wide部分直连，梯度一步到位，收敛会比deep快。这样搭档ftrl，中和下，给deep部分以机会，学习更多的信息。

问题二：Wide&Deep模型中为什么要将连续特征离散化?

链接：wide&deep模型中为什么要将连续特征离散化?

答案一（严林）：

这个问题可以从模型的表达能力说起，对于Wide&Deep，Wide负责Memorization，Deep负责Generalization。Wide部分要能够做好Memorization，那也需要对Continuous Feature做准确刻画。Wide部分是一个LR，那么这个问题就转化为LR部分如何做离散化了，percentile只是离散化的一种方式，至于离散化的优点，参考这个问题：

> 连续特征的离散化：在什么情况下将连续的特征离散化之后可以获得更好的效果?
>
> 菜鸟的机器学习，公众号：菜鸟的机器学习
> [学习] 连续特征的离散化：在什么情况下将连续的特征离散化之后可以获得更好的效果?

再说，这种离散化对于Deep部分有帮助吗？从实践上看，有！目前学术界和工业界都还没有严格证明为什么会有帮助，个人理解是：一个Feature的不同区间获得了不同的表示（Embedding），同时这些不同的区间再在模型Neural Network中配合网络权重和激活函数获得更细致的刻画。因此，模型效果得到了提高。Google 16年在 Deep Neural Networks for YouTube Recommendations这篇论文中也是类似的策略，同时加入了连续的Cumulative distribution（不过这点我在测试中并没有发现有效果提升）。

至于分段是不是均匀的，从实际效果看，我更建议保证每段的样本相对均匀。

最后，挖个坑，目前在Applied Deep Neural Network中，模型的表达其实是由Data Information（Embedding）和Model Structure Information共同决定的，那么怎么才能根据模型和应用调整数据表示方法呢？有机会展开讲。

答案二（a88i99）：

为了回答这个问题，我专门看了一下这篇文章。先上与之相关的图吧。

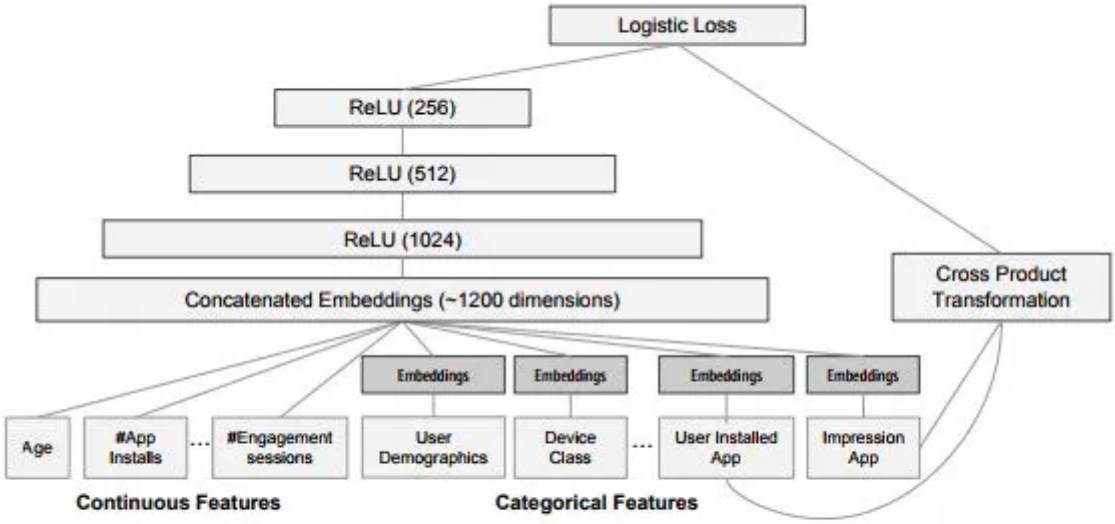

Figure 4: Wide & Deep model structure for apps recommendation.

这里的连续特征并未做embedding，如年龄、安装app数量等。那为什么没做embedding，而类别特征又做了embedding呢？**何时需要做embedding?**

比如，用户装了陌陌这个app，其实它意味着用户相关的信息量是很大的，对吧？肯定不止一维！若不做embedding，我们用0/1来表示是否安装，这样表示太粗糙了，很多信息未挖掘出来，泛化能力差。但若用了10维的embedding，不同维度代表了不同的含义，就很好。

若以自然语言来举例，狗、猫、大桥是三个词。若用onehot编码，两两的相似度（欧氏距离）都是相同的。但常识告诉我们，狗与猫肯定更相似，距离应该应该更短。而词向量的引入，即embedding表示，解决了这个问题。

而年龄这类特征，已经不能再分解了，它已经是最细粒度了。所以这些连续特征未做embedding。也可以认为，它已经是一维的embedding，然后与其他embedding拼接成一个长向量。

so far so good，让我们来直接回答题目的问题。连续特征归一化后，为什么还要分段处理？这里面绝对不是onehot！！因为表示的维度未变，都是1。这样做是**粗粒化，为了增加泛化能力**。下面以归一化后的年龄举例来说明：

假设分为五段，0.23、0.26都用0.2来近似，即0.23岁与0.26岁，对于曝光后是否点击安装这个app，是没有区别的。

embedding也叫分布式表示，那分布式表示还有什么优点呢？相对onehot表示？有兴趣的话，我再答吧。

> 问：终于有人答到我想要的点了。。那为什么粗粒化可以增加泛化能力呢？就拿你举的年龄的例子来说，29岁和30岁差了0.01，分段后却差了0.1，乘以权重后输出值发生了突变，在我看来是不好的性质啊
> 答：你说的这种情况样本占比还是比较少的。从整体上看，粗粒化后，大部分样本受益，少部分样本略有损失吧。我的理解大概这样

> 问：可否具体解释一下粗粒化增加泛化能力的原理呢？其实我也想到了这种解释，但是不知道怎么进行理论推导。答：想象一下这个图像，假定除年龄外的其他特征都是常数，其实是高维空间切一刀就是这样。若不做分段，转化率是年龄的高阶平滑曲线，而分段后，这个曲线就由多个阶跃函数替代/近似，大多数样本方差降低了。只是在分界点附近，方差变高。但是模型的方差是总方差除以样本数量得到的，显然模型的方差下降了，泛化能力增强了。答：分段可以增加非线性，可以近似比对一下relu，而且对异常值的处理比较好。

微信ID：newbie_ml

长按左侧二维码关注

阅读原文