基于Deepcrosing模型的推荐系统

一、背景描述

当今世界，互联网的出现和普及给用户带来了大量的信息，满足了用户在信息时代对信息的需求，但随着网络的迅速发展而带来的网上信息量的大幅增长，使得用户在面对大量信息时无法从中获得对自己真正有用的那部分信息，对信息的使用效率反而降低了，推荐系统是解决这一问题的好方法。推荐系统将用户可能感兴趣的信息推荐给用户，提高了用户对信息的使用效率。

推荐系统主要的推荐方法有以下几种：基于内容的推荐方法、基于协同过滤的推荐算法、基于关联规则的推荐算法、基于知识图谱的推荐算法等。而本次大作业我选择了搭建NeuralCF模型来完成一个电影推荐任务，并采用梯度下降与遗传算法来训练模型，最后尝试使用有svm算法来进行最后的分类来提升模型性能。模型实际上预测电影和用户间相不相关,相关的电影将被推荐给用户,因此实际上是一个二分类问题,且需要对数据集做采样处理才能用于训练。

本次使用的数据集是Movielens100k数据集，Movielens数据集包含多个用户对多部电影的评级数据，也包括电影元数据信息和用户属性信息，常用于推荐系统模型。评分数据包含用户id、电影id、评分、评分时间戳这四列。电影元数据包含了电影id、电影名称、电影上映时间、电影IMDB链接以及一系列电影类型，当电影是某种类型时对应那一列为1。用户元数据包含了用户id、用户年龄、用户性别、用户职业以及zipcode这5份特征。

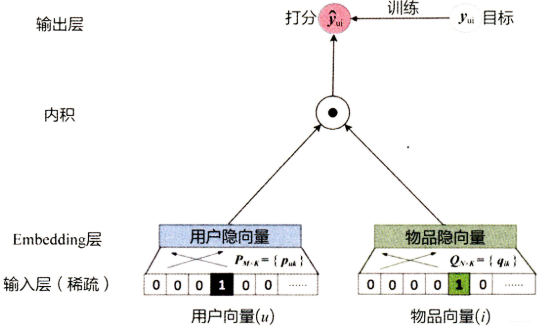
二、设计思路

1.推荐模型设计思路

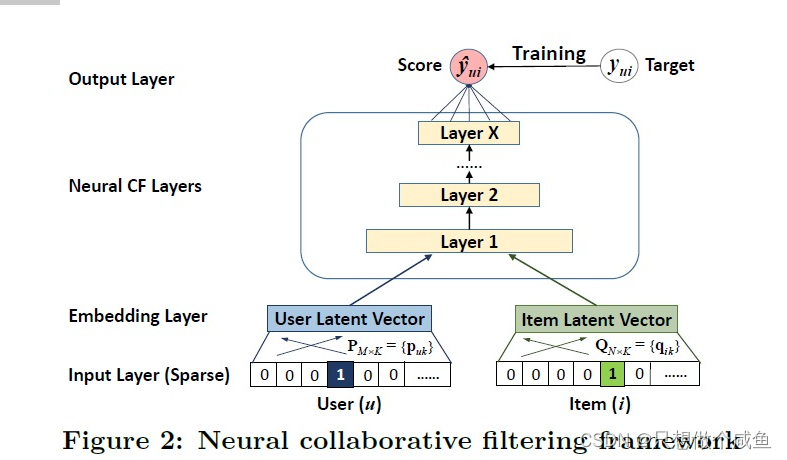
首先简单介绍协同过滤算法、NeuralCF模型、Deepcrossing模型。

协同过滤实际上是构建出一个横轴为物品id，纵轴为用户id，值为评分的评分矩阵，这种评分矩阵可视为由许多用户向量组成，也可以视为由很多物品向量组成。因此借助用户向量可以实现计算用户相似度，借助物品向量则可以计算物品相似度，以相似度来向用户进行推荐。

而评分矩阵实际上可以用矩阵分解分解成两个矩阵，而这两个矩阵一个被解释为表示用户的特征，另一个被解释为表示物品特征。因此评分矩阵的评分实际上可视为用户向量与物品向量内积得出。



NeuralCF模型的思路就是采用多层感知机来替代内积操作，让用户特征与物品特征充分融合。而NeuralCF则有一种双塔形式，即使用多层感知机来进行用户向量嵌入与物品向量嵌入，最后使用内积方式来得出结果。同样，这种双塔模式实际上也可以用多层感知机来替代内积操作。本次实验实际上本人一开始想编写双塔模式+MLP这一种模型，但实际编写后，由于采用了两个MLP作为输入，一个MLP输出这种情况，让梯度下降的求解变得非常缓慢，于是放弃了这种选项。选择了采用NeuralCF单塔模型。在NeuralCF中，输入的实际上只用用户id向量与物品id向量。



而事实上，既然评分矩阵由用户向量与物品向量得出，那么用户向量由用户的特征融合提取得出，物品向量由物品的特征融合提取出实际上是一种自然的想法，即NeuralCF用于融合的MLP输入不仅是代表用户id的向量与物品id的向量，而是用户/物品各特征的拼接，而这些特征通过Embedding层嵌入。

实际上Deepcrossing模型可以视为上述想法的具体体现，而其与上述想法的区别在于上述想法使用的MLP在Deepcrossing中更改为残差网络。

由于是先搭建出NeuralCF再在此基础上多次修改网络，因此本次搭建的基本模型与Deepcrossing类似，参考了NeuralCF模型的单塔形式，神经网络部分只用了单纯的MLP而并没有将MLP修改为残差网络，同时没有采用NeuralCF模型的广义内积部分。

梯度下降法在pytorch中有封装好的实现，定义好损失函数后直接使用backward函数即可实现以梯度下降法优化。

2.遗传算法设计思路

使用遗传算法优化的设计思路如下。对于种群，直接将整个模型作为种群，整个种群设定为100个模型。选择操作为仅保留表现最好的20个模型，然后采用轮盘赌选择方式选择两个模型作为父辈开始交叉操作生成子代，同时该20个模型完整保留到子代不进行废弃。其中，适应度函数尝试过两种，一种直接采用模型正确率作为适应度函数，另一种使用模型的输出层误差，即输出与标签的欧式距离。交叉操作实现中采用过两种方案，一种是将模型中的4个Embedding与3层全连接层的权值矩阵拉伸为一条一维向量后都作为染色体，然后随机选择染色体中的一段交换作为交叉操作；另一种则是将染色体的每一层整体视为一个基因，以0.5的概率选择该层来自两个父辈模型中的哪一个。

变异操作则采用正态分布生成一个与各层权值矩阵大小相同的矩阵，然后将该矩阵与权重矩阵相加作为新的权值矩阵，以此来作为变异，同时，变异操作只用于交叉产生的子代上，亲代不进行变异。

3.样本处理思路

对于样本的处理，这里采用了随机欠采样生成负样本来补充训练集。实际上在采用随机欠采样处理训练数据前，在单纯的NeuralCF上训练时我曾将没有评分的每一个用户与电影的组合都作为负样本放入数据，即事实上将协同过滤评分矩阵中的每一个元素都作为了训练样本，最后预测准确率高达93%以上，当时并未计算正样本的召回率，但考虑到这份训练数据有943\*1682将近200万份数据而正样本只有10万份，数据严重不平衡可能影响了NeuralCF的结果偏向于预测负样本，但考虑到推荐系统的性质，预测偏向于负样本是严重的后果，因此使用随机欠采样处理训练集。

对于用户特征与物品特征，年龄需要做离散化处理，而职业与性别则用数字编码后直接嵌入，其中职业的嵌入向量向量长度实际上是随机选取的，没有特别找过最佳参数。zipcode意义不明不好嵌入，因此直接去除。而物品特征，后面的19个电影类别可视为同一个特征，将其直接作为一个向量放入网络，其中的链接、电影名无用直接删去，上映时间与评分中的时间戳类似，需要进行离散化与规范化，删去。最后使用的特征有用户id，用户性别，用户职业，电影类别，电影id这几项。时间有限，未能处理好这些变量将其调整至适合传入网络。

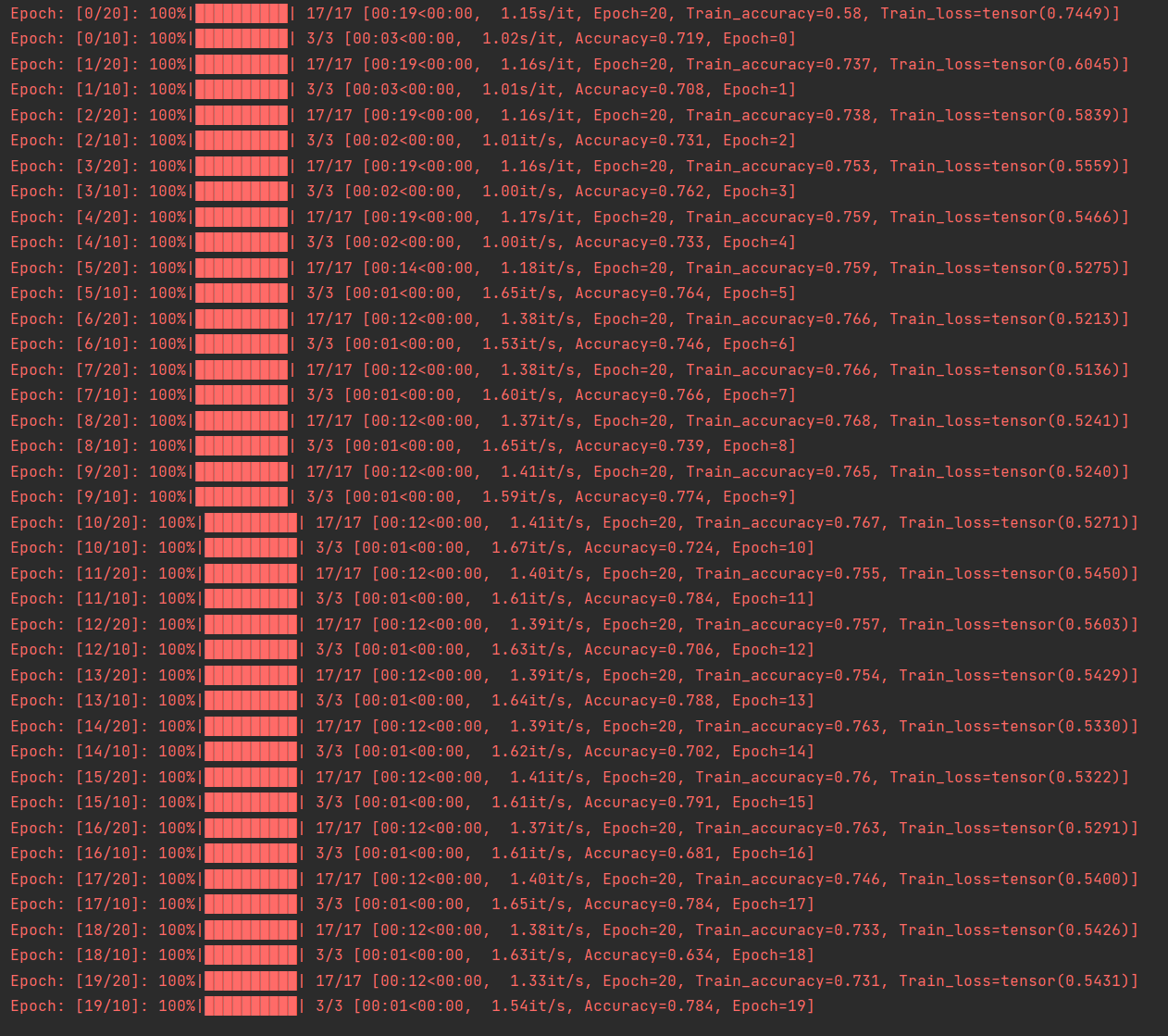
4.SVM方法优化输出思路

由于实际上计算正确率时将0.5以上的值算为1，将0.5以下算为0，即将神经网络的输出进行二分。因此考虑采用SVM方法进行二分来取代简单的0.5划分。SVM模型可以视为以多个超平面切割物品的向量空间从而实现将不同物品进行分类，在二维就表现为用多条线将平面分为不同部分而让不同区域内的点被分为不同类。sklearn包有提供现成的SVM模型供使用，其中最重要的参数是惩罚参数，惩罚参数越高越贴近训练集但泛化能力差。在这里结合SVM模型的方式是在每次网络输出结果时将网络的输出结果当作数据输入到SVM模型中进行训练，然后再在求解正确率时采用训练好的模型进行训练。想了想应该有更好的方式即每次出训练结果时，应该用之前输出的所有训练结果当作训练集放入SVM模型中，然后将本次输出作为SVM模型的测试集，充分利用所有数据。

三、结果展示

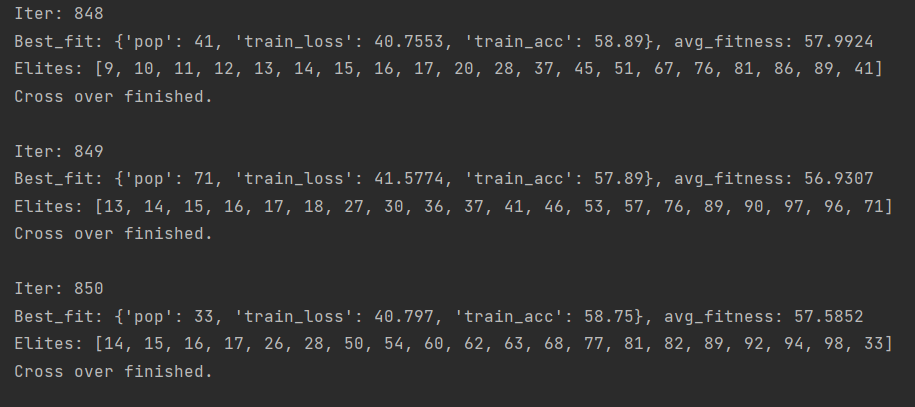
1.梯度下降法

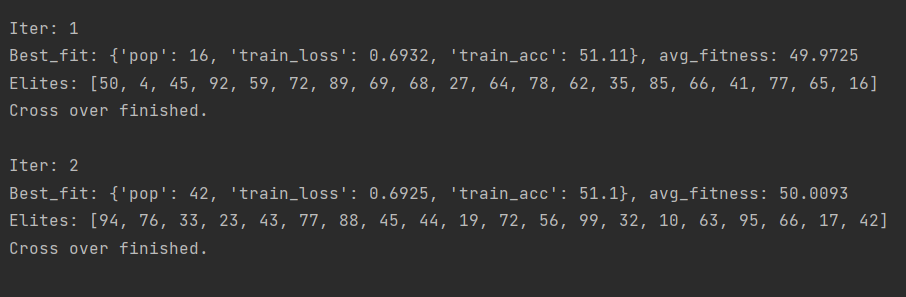
显然梯度下降法从第8次迭代后准确率和损失函数都开始在一个范围内波动，趋于收敛，而且查看测试集表现，开始波动后，损失函数较小的测试集反而表现较差，说明继续减小损失函数可能过拟合。想要改进应该进行特征工程或更改网络结构。变化趋势图见picture文件夹。



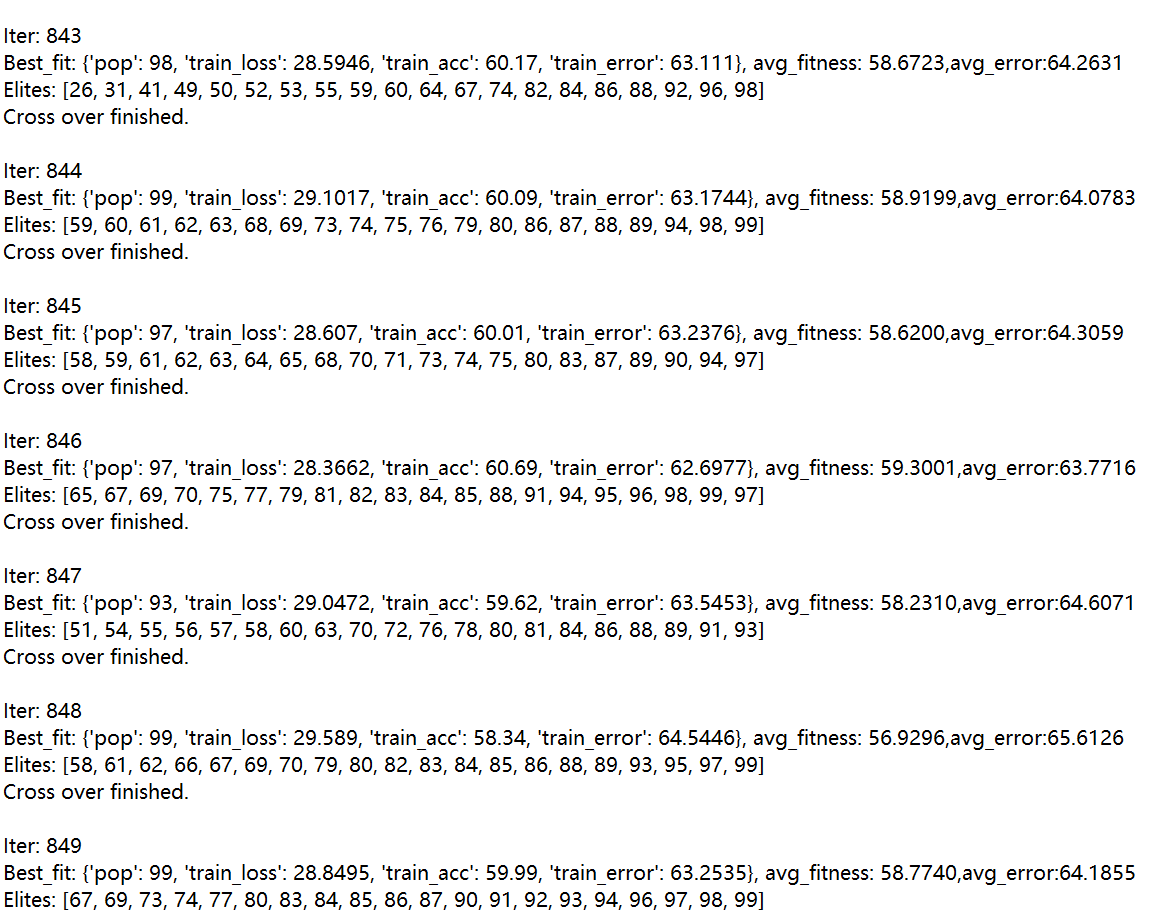
2.遗传算法

这是采用正确率适应函数，突变概率为0.4，交叉方式采用第二种的遗传算法结果，由于遗传算法时间较长，因此使用相近的代数进行比较。很明显，相较于初始结果遗传算法网络有优化，但经过一段时间后仍然会开始波动且损失函数会有增大。





这是采用误差适应函数，突变率为0.2，交叉方式采用第一种的遗传算法结果。全部结果复制粘贴到了文件EA\_result.txt文件中。显然，相较于前一种遗传算法结果要好些，但仍然在准确率增大到一定程度后收敛。推测表现更好的原因为矩阵权值变化程度相较于第一种更高，更容易出现相较之前更好的模型。



我们可以看到这两种遗传算法中无论怎么增加迭代次数，结果上升到某一阙值后不停动荡没有更大的提升，但相较于梯度下降的结果仍然差距较远。很明显遗传算法已陷入了局部最优，但程序没有对此进行处理。因此应该考虑使用恰当的方式给种群引入新变量，考虑到突变也是新变量，因此考虑增大突变概率。

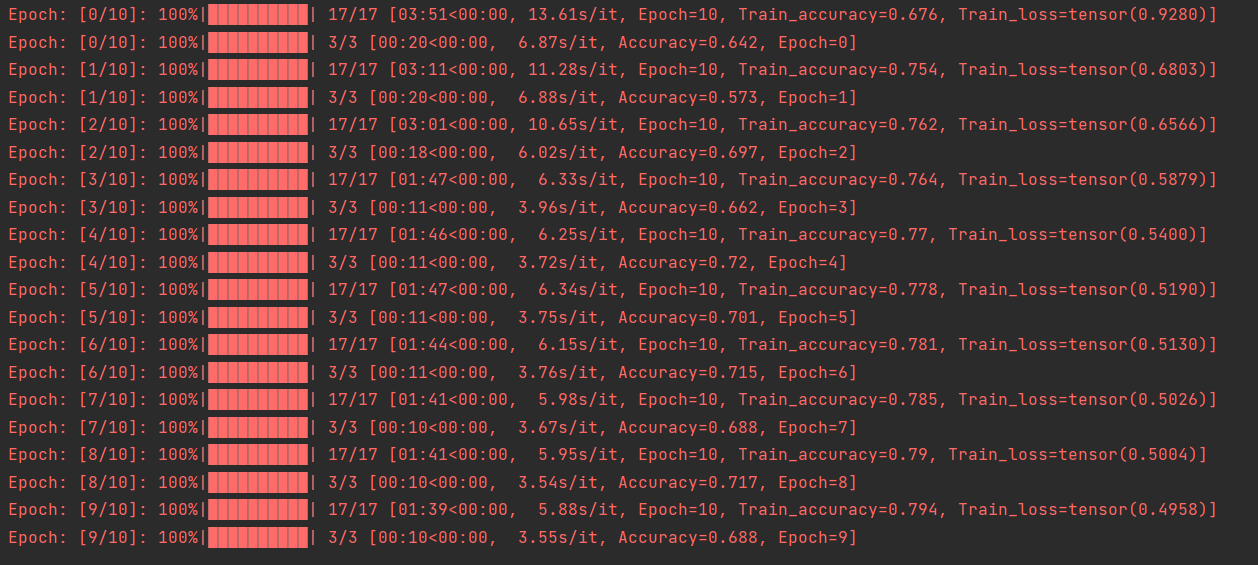
将突变概率增大到0.4结果。显然提升了1%左右的准确率，但同样的，实际上在迭代到800多次时已基本平稳，这一次遗传算法在之后部分轮次中测试集出现过62%以上准确率，训练集出现过63%以上准确率，但大部分时候始终在61%附近波动。这一轮训练输出结果存在EA\_result2.txt中。

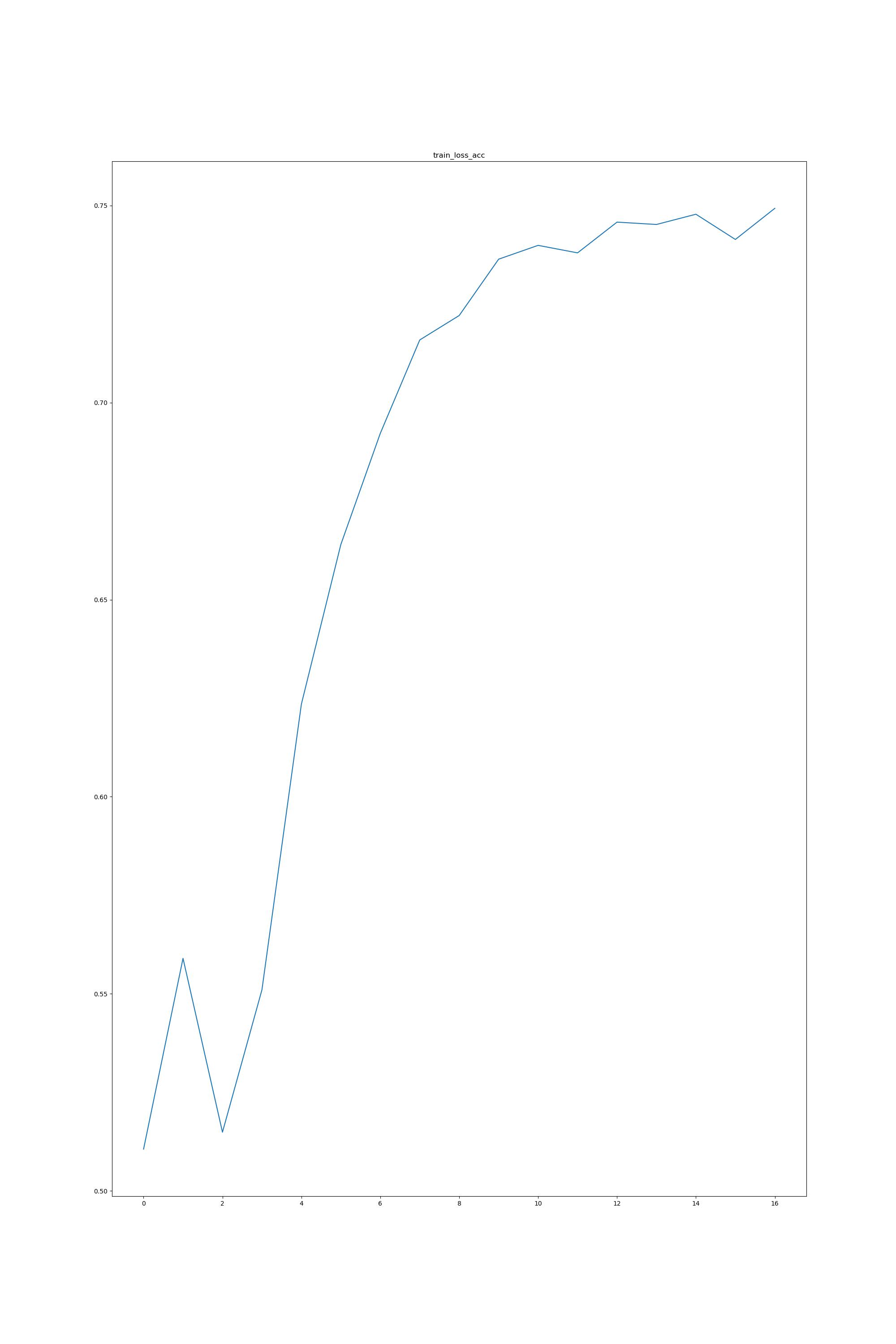


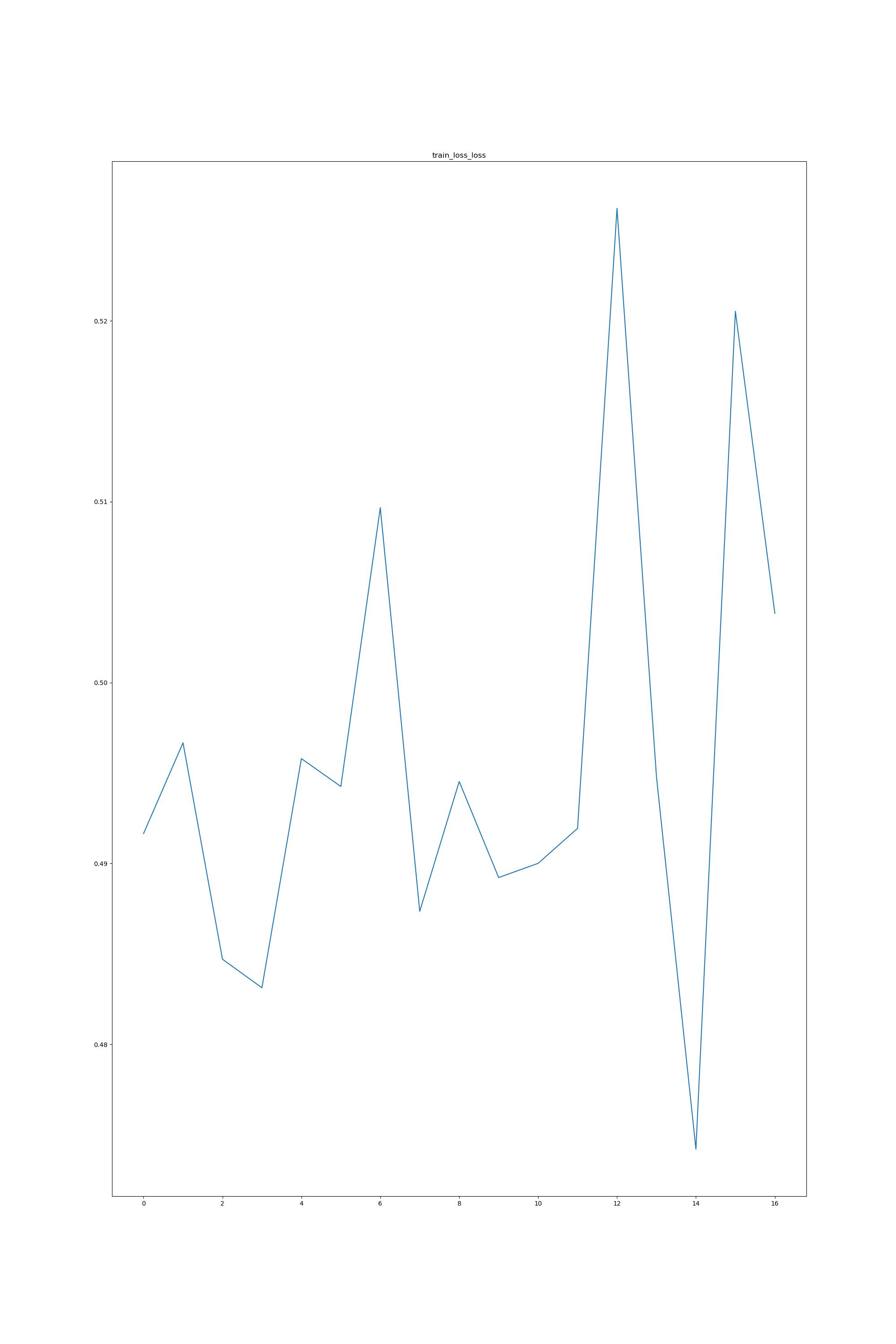
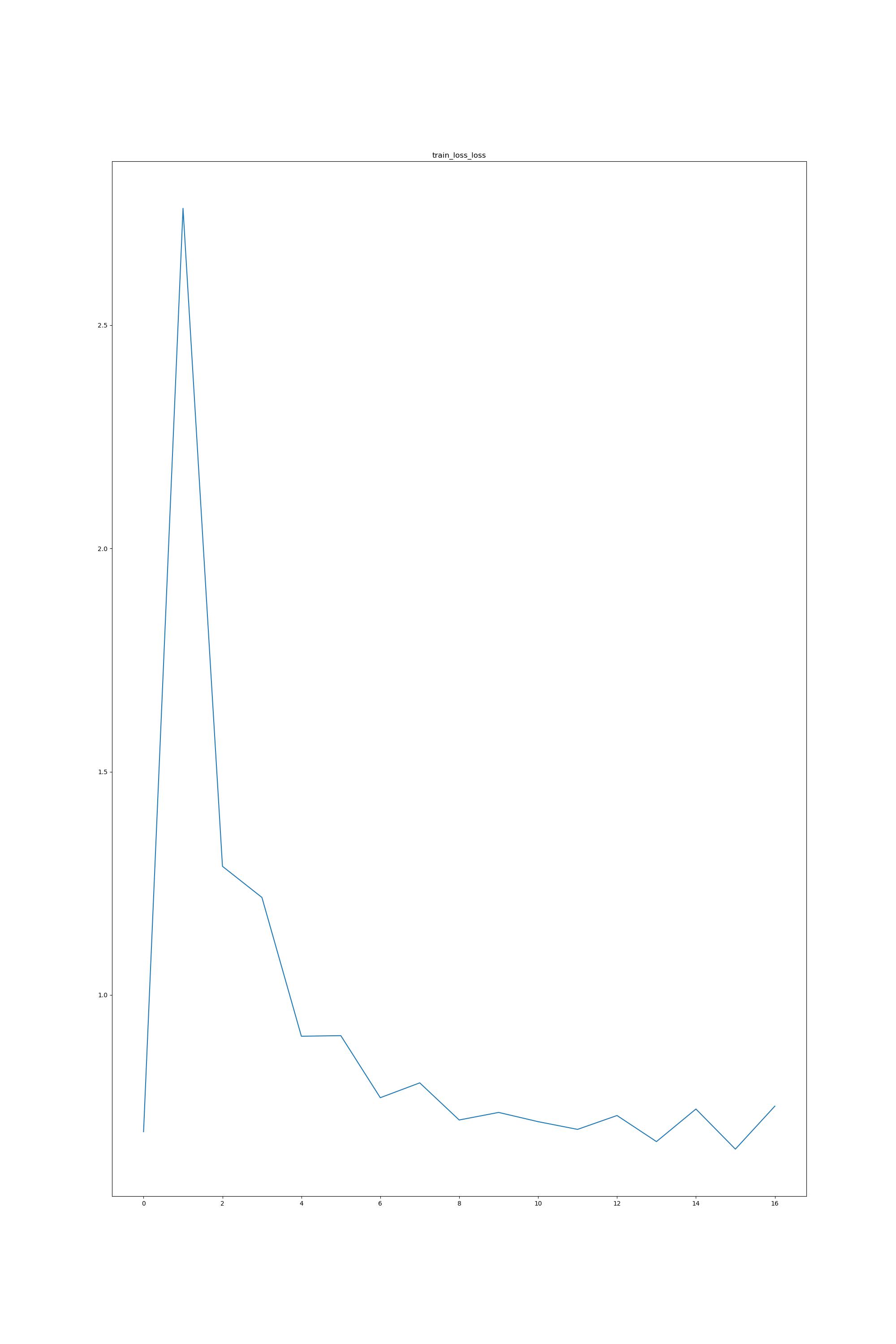
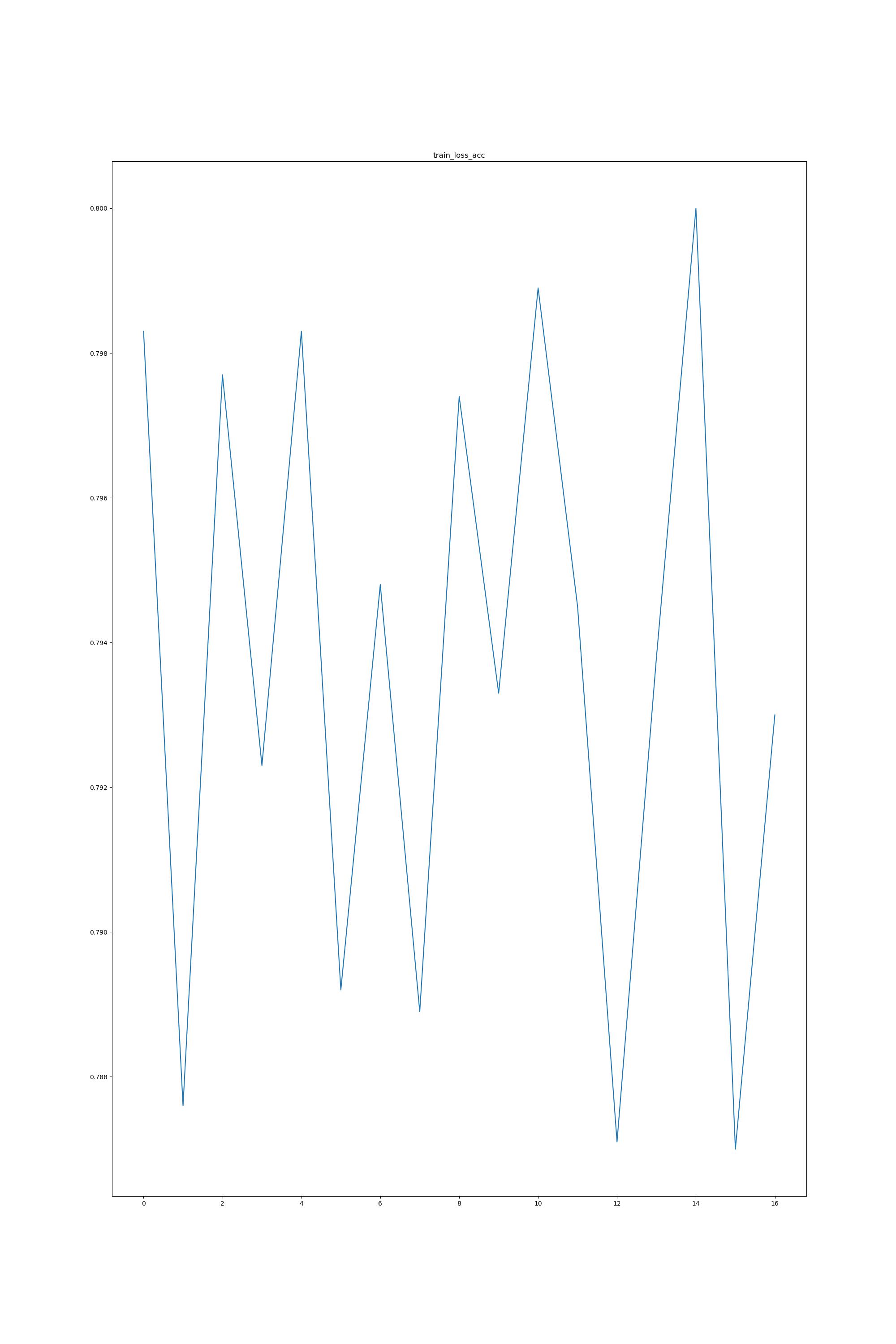
由于每次遗传子代保留了最优秀的亲代与最优秀亲代产生的子代且只有新产生的子代会变异，理论上如果每个模型多次输出结果都相同的话只要子代能够出现更优秀的个体算法必然能够选出优秀的新个体而不是收敛。因此，分析三次结果，出现收敛的原因可能为（1）与实际生物遗传不同，由于染色体采用神经网络权重而实际上权重交叉更改后整个网络不一定还能够保持原本的性能，因此个人认为实际上即使不突变子代也不一定能够保存原本亲代的优异性能而是生成一个新网络，而整体上的突变较低，因此可能不能够诞生足够优秀的子代模型。可以考虑继续增大突变率或者定期往种群中加入新个体来实现。（2）由于采用了将整个训练数据分批放入模型的操作，因此每一轮选择输入实际上不一样，同一网络对不同的输入准确率不同，而测试用的最优模型每一轮选择更新一次，因此最优模型可能不是真的最优模型，应考虑将数据整体输入或以平均准确率来选择最优秀的模型。

3.使用svm对神经网络进行分类结果。

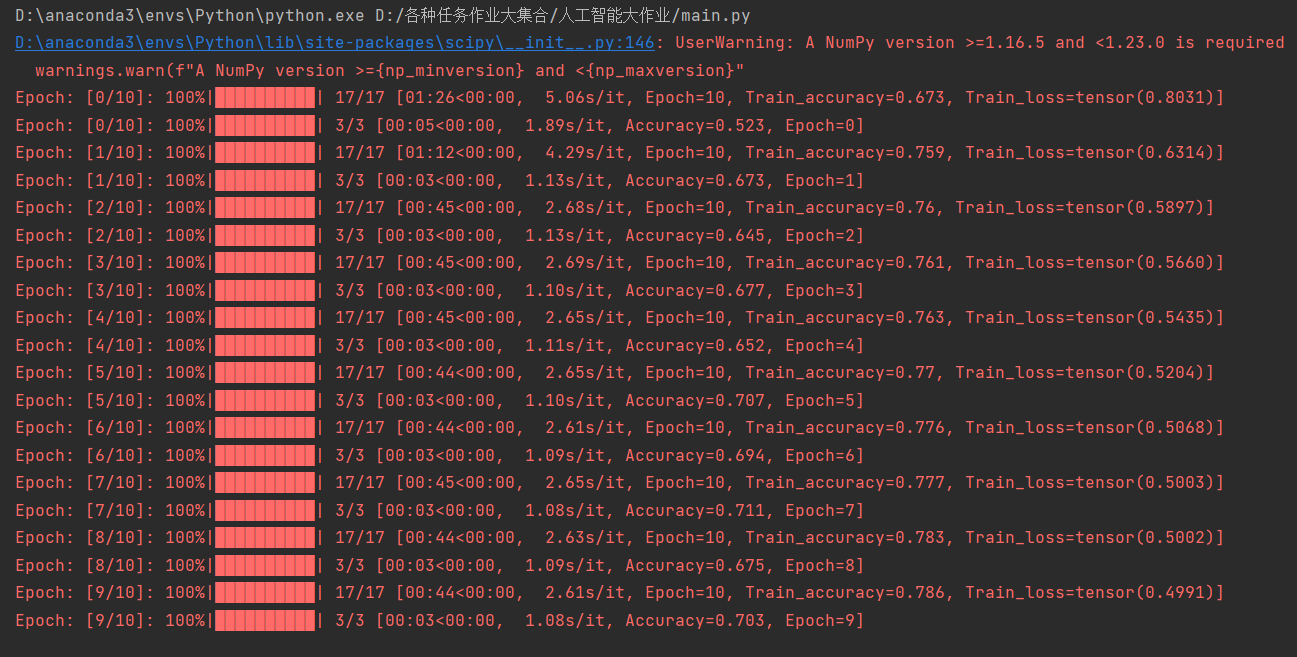
这里svm采用了sklearn库中svm的默认参数，显然，使用svm对神经网络输出的值进行二分类是可以提升结果的，但很明显svm对数据划分出现了过拟合的现象。由于推荐系统实际使用时测试集不可能有标签进行训练二分类后再输出。因此调整svm惩罚参数后再次尝试。由于运行20次时忘记截图或复制输出，因此再次运行10次查看测试集表现。训练输出全部图见文件夹picture\_svm。



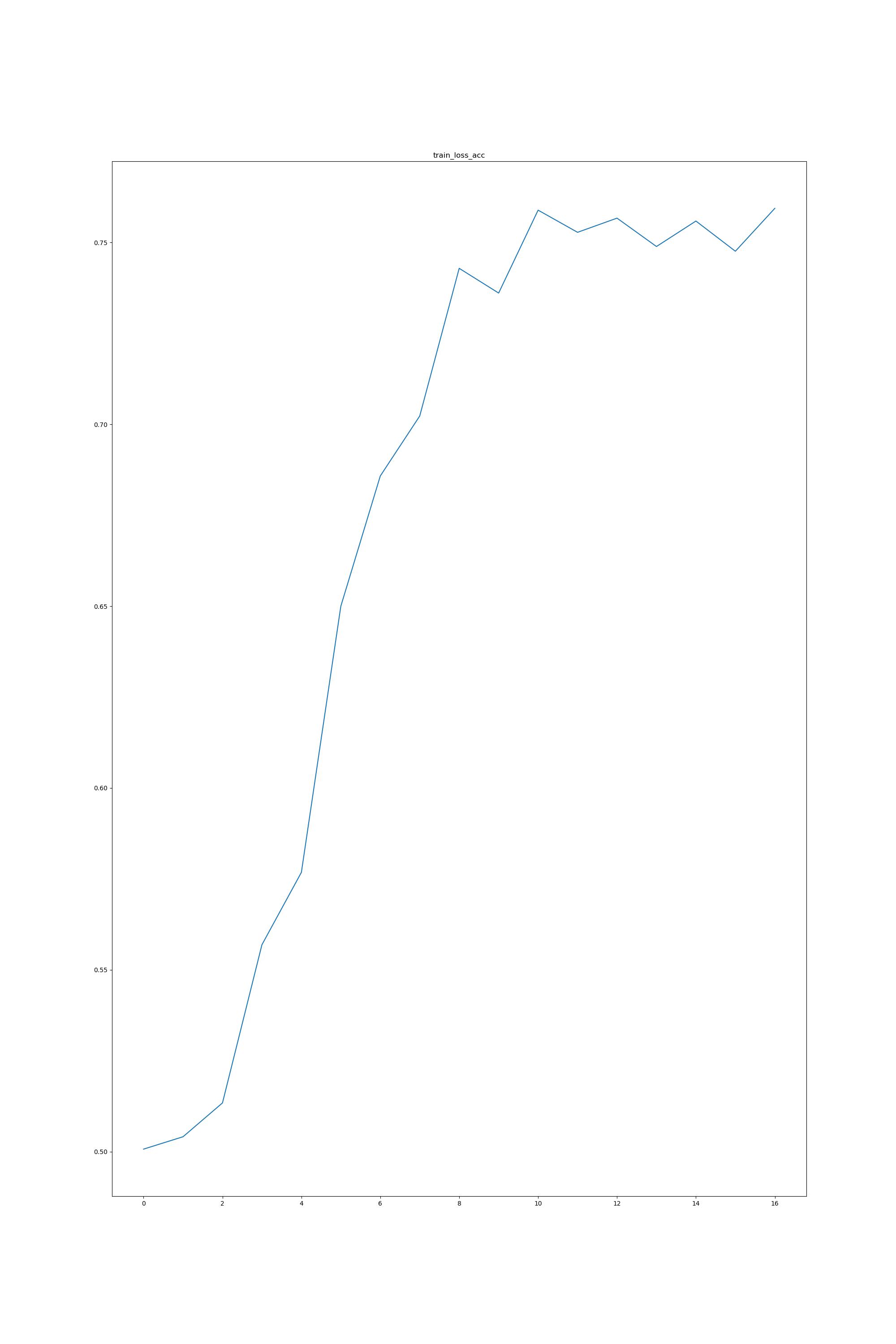




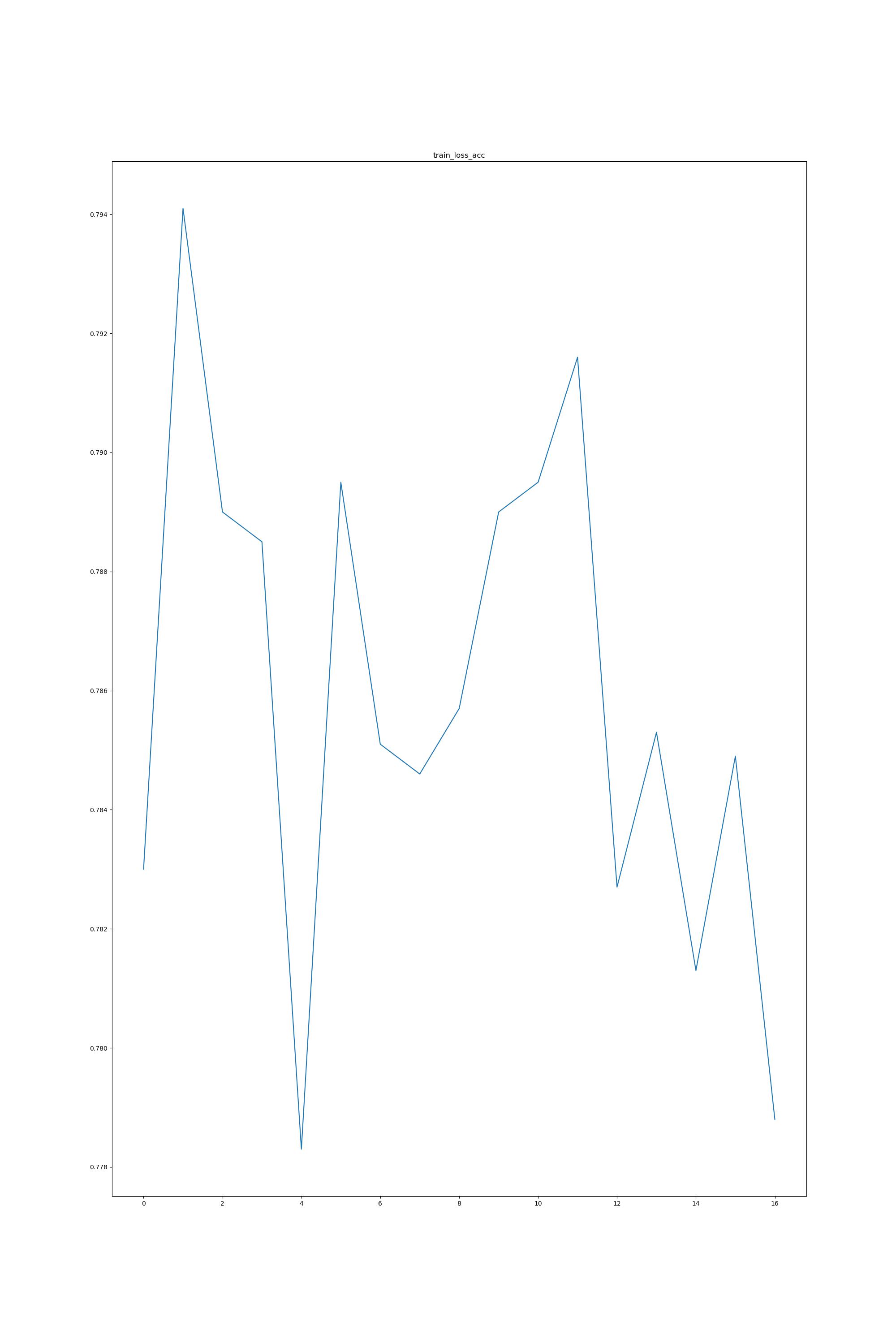
修改参数后的svm神经网络结果，很明显泛化效果好上一些，且也许多迭代几次能够有更好表现。可见使用将惩罚调整至合适参数后线性svm进行二分类结果相较于直接使用0.5作为区分值效果可能更好。训练输出图见文件夹picture\_svm1。



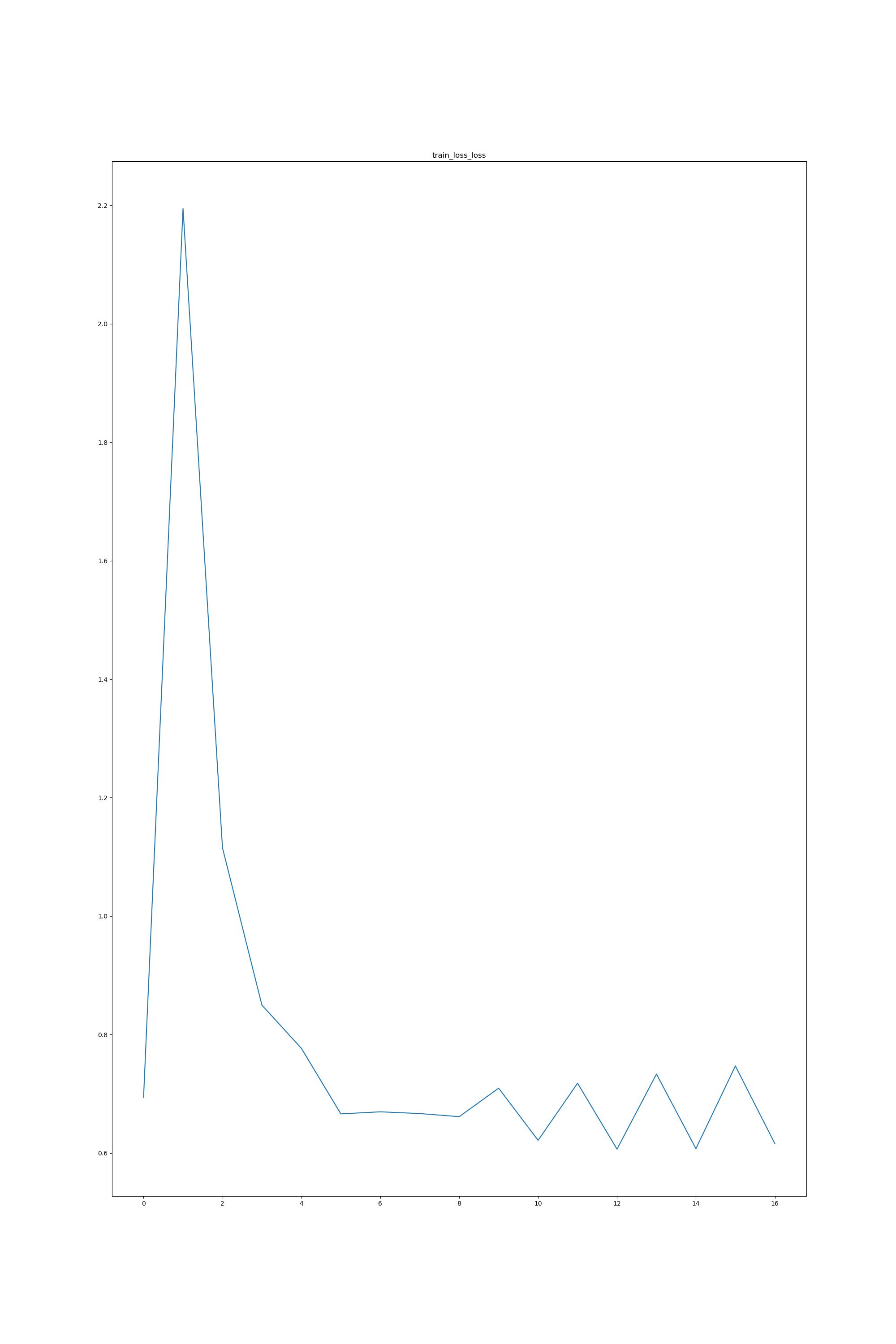
第一轮结果。



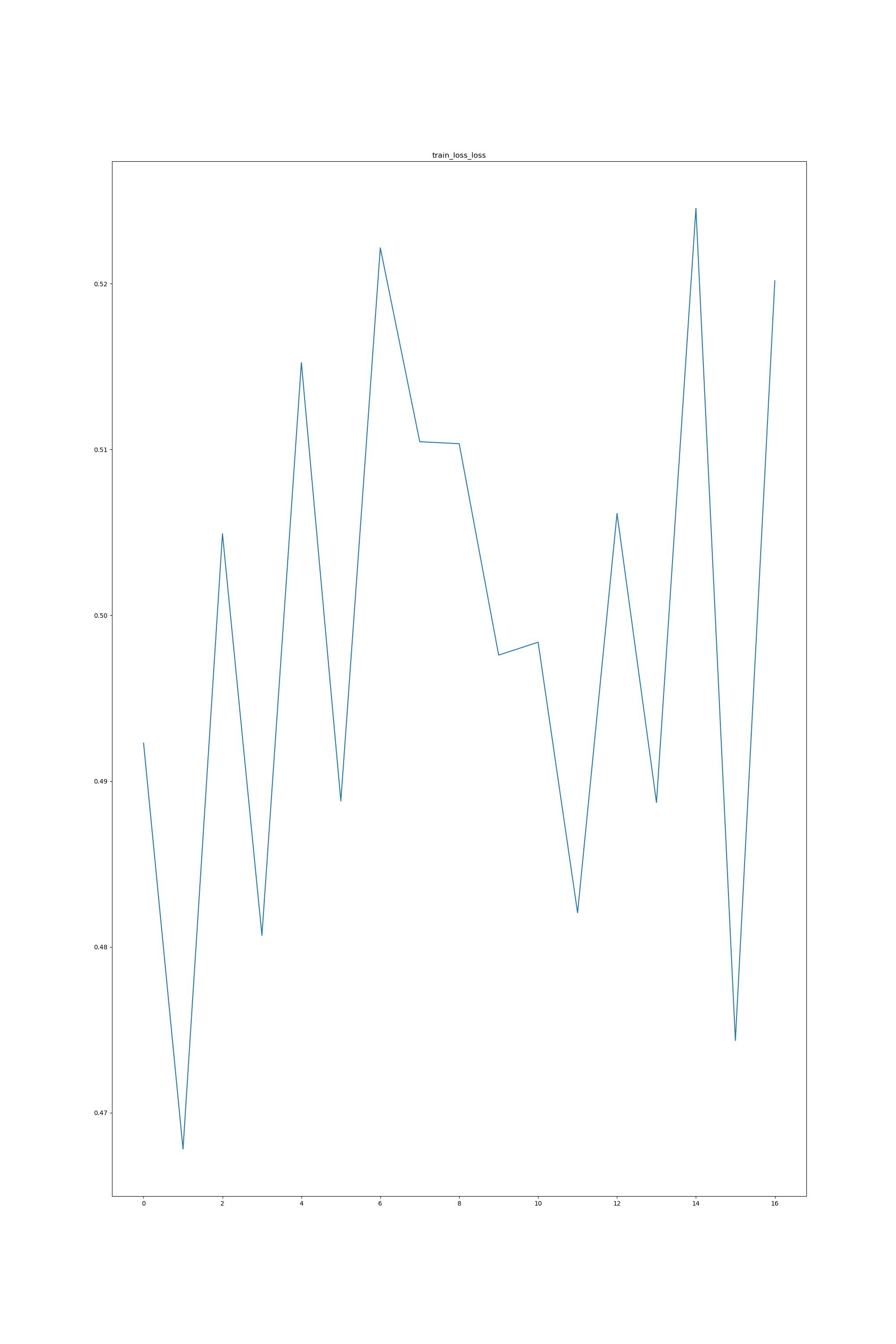
第九轮结果。



第一轮结果。



第九轮结果。



从结果曲线来看，实际上第一轮训练就基本已经趋于收敛了，之后尽管优化器的训练速率是下降的，但损失函数与正确率一样，都在不停的波动，控制台打印出的正确率与损失函数值是本轮次的平均值，因此，随训练轮次增加正确率在波动中小幅度上升，损失函数在波动中小幅度下降，增加训练次数可以一定程度上继续优化。但结合单纯的梯度下降，可发现实际上15轮训练左右已达到最佳，需要修改网络结构或进行特征工程才能继续提升。

四、后续可能存在的改进

1.采用合适的自然语言处理方式将电影名嵌入为一个向量加入模型中。在实验过程中我们将电影名删去不作为特征放入模型。但考虑到日常生活中，我们可能出于个人爱好或其它原因被电影名吸引从而观看一部电影，也就是说电影名可能影响用户对该电影的兴趣，因此，采样合适的方式处理后可将电影名作为特征加入模型。

2.调整采样方式。在实验中我们将所有有评分的数据处理成了正样本，但实际上评分低下电影不见得引起用户兴趣，很可能不适合推荐给用户。因此，考虑将平均评分低于某一分数的电影作为负样本并借助SMOTE方式随机过采样生成负样本可能是更好的处理方式。除此之外，由于负样本随机生成，因此实际上可能不符合实际，与正样本相互作用影响结果。意识到这一问题时时间有限，因此没有做出实际改进。

3.采用更合适的方法进行特征嵌入与拼接，参考网络上实现Deepcrossing的代码，有很多Starcking拼接层都直接使用向量拼接而成，可以考虑将该部分替换成神经网络来融合，形成用户特征与物品特征，在将融合出的用户和物品特征放入MLP，至于用哪种神经网络需要经实际检验。以及特征嵌入同样可以替换成神经网络从而得到更好的嵌入。

4.进行一定的特征工程，比如使用上映时间与评论时间做差构建一个新特征，可能反应出年代对运算结果的影响。由于反复修改网络结构消耗了大量时间，因此没有实际做出这一点。