机器学习工程师纳米学位毕业项目

盐块识别挑战

阳 宇 翔

2019年8月13日星期一

目录

[1. 定义 1](#_Toc17042956)

[1.1. 项目概述 1](#_Toc17042957)

[1.2. 问题陈述 1](#_Toc17042958)

[1.3. 评价指标 2](#_Toc17042959)

[2. 分析 3](#_Toc17042960)

[2.1. 数据可视化 3](#_Toc17042962)

[2.2. 算法和技术 4](#_Toc17042963)

[2.2.1 语义分割概述 4](#_Toc17042968)

[2.2.2 U-Net结构 4](#_Toc17042969)

[2.2.3 ResNet 5](#_Toc17042970)

[2.2.4 SENet 6](#_Toc17042971)

[2.2.5 Deep supervised 8](#_Toc17042972)

[2.3. 基准指标 8](#_Toc17042973)

[3. 具体方法 9](#_Toc17042974)

[3.1. 数据预处理 9](#_Toc17042976)

[3.2. 实现 9](#_Toc17042979)

[3.1.1 模型构建 10](#_Toc17042980)

[3.1.2 模型训练 10](#_Toc17042981)

[3.3. 改进 10](#_Toc17042982)

[3.3.1 Epoch 10](#_Toc17042985)

[3.3.2 Backbone 12](#_Toc17042986)

[4. 结果 14](#_Toc17042987)

[4.1. 模型评价与验证 14](#_Toc17042989)

[4.2. 结果分析 14](#_Toc17042990)

[5. 总结 15](#_Toc17042991)

[5.1. 结论 15](#_Toc17042993)

[5.2. 后续改进 15](#_Toc17042994)

[参考文献 15](#_Toc17042995)

# 定义

## 项目概述

地球上有大量石油和天然气聚集的地区往往也会在地表下面形成巨大的盐沉积物。但是，了解大型盐矿的确切位置非常困难。专业的地震成像仍然需要人类专家的解释来确定盐体位置。这导致渲染地震成像中的盐矿是非常主观的，高度可变的。更令人担忧的是，这会给石油和天然气公司的钻探人员带来潜在的危险情况。为了创建最准确的地震图像和3D渲染，TGS（世界领先的地球科学数据公司）希望Kaggle的机器学习社区能够构建一种算法，自动准确地识别地下目标是否为盐。

地震图像是通过对来自岩石边界的反射进行成像来产生的。它显示了不同岩石类型之间的界限。理论上来说，反射强度与界面两侧物理性质的差异成正比。虽然地震图像显示了岩石边界，但它们并不能反映岩石本身的特性;有些岩石易于识别，有些岩石很难识别。

世界上有几个地区的地下有大量的盐。地震成像的挑战之一是识别地下部分是盐。盐具有使其既简单又难以识别的特征。盐密度通常为2.14克/立方厘米，低于大多数周围的岩石。盐的地震速度为4.5千米/秒，通常比周围的岩石快。这种差异在盐–沉积物界面处产生了明显的反射。通常盐是无定形岩石，没有太多的内部结构。这意味着盐中通常没有太大的反射率，除非其中有其他沉积物。盐的异常高的地震速度可能带来产生地震成像的问题。

盐识别挑战是计算机视觉领域中典型的语义分割问题，就是机器自动从图像中分割出含盐区域，并识别其中的内容。语义分割在处理图像时，具体到像素级别，通过编码-解码结构对图像中的每一个像素进行分类。下图是典型的图像语义分割问题：

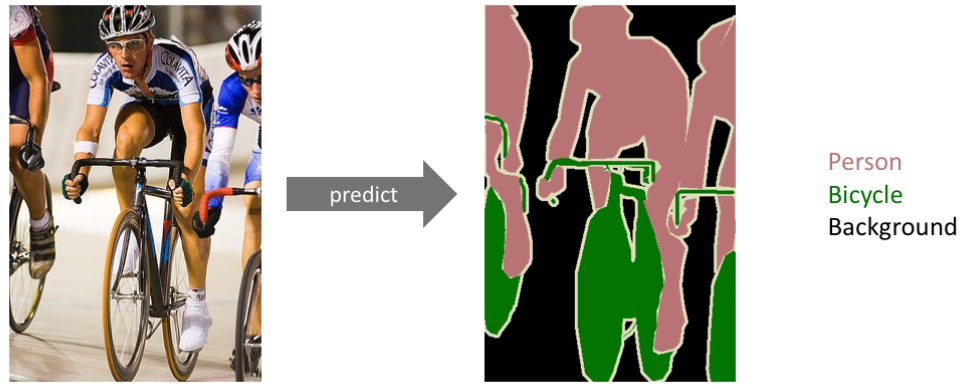


图1 典型的图像语义分割问题（来源：**VOC 2012**）

## 问题陈述

本项目的实验数据来源于Kaggle比赛提供的数据集。Kaggle比赛所用的数据是随机选取的不同地下位置的一组灰度图像。图像为101 x 101像素，每个像素分为盐或沉积物。除了地震图像之外，还为每个图像提供成像位置的深度。比赛的目标是分割标示出含盐区域。图1是训练图像示例：

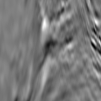


图 2 盐块识别挑战训练图像示例

训练集包含4000幅101 x 101像素地震图像和与其对应的标示出盐体的mask图像、地下深度数据，测试集包含18000幅相同大小的地震图像和与其对应的深度数据。比赛参与者的目标是使用算法为18000幅地震图像生成盐体mask图像。

## 评价指标

1. **IoU**

Intersection over Union，简称IoU，是一种测量在特定数据集中检测相应物体准确度的一个标准，是图像分割任务的最常用指标。IoU用于测量真实和预测之间的相关度，相关度越高，该值越高。在该项目中，我们首先计算一幅图的IoU得分。一组预测对象像素和一组真实对象像素的IoU计算如下：

当设定阈值为0.5时，如果IoU得分大于0.5，表示正确对图像进行分割，如果低于该分数表示为没有正确分割。

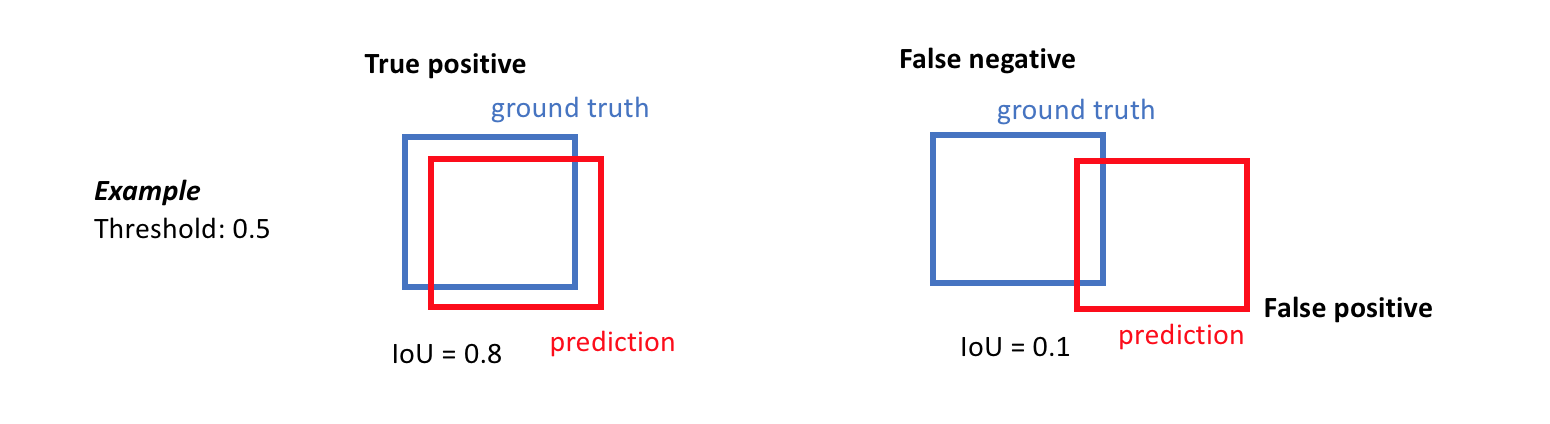


图 3 example

1. **Average Precision**

当阈值在[0.5, 0.95]之间以步长0.05变化时，对于每一个阈值t，通过将预测对象与真实对象进行比较而得到的真阳性（TP），假阴性（FN）和误报（FP）的数量来计算其像素精度Pixel Precision，计算公式如下：

当阈值(Threshold)越高，评价标准就越严格，我们检测目标的IoU值必须足够高才能满足要求。遍历所有阈值后，计算平均像素精度：

遍历所有阈值后取得的平均像素精度值即为单个图像的平均精度。最后，该项目将测试数据集中每个图像的各个平均精度所取的平均值作为评价指标。

1. **Lovasz loss**

该loss是在论文[1]中提出的，该loss是对现有的binary-crossentropy loss的一种改进，实验证明，这一loss非常适合以 IOU 作为评价指标的问题。

该loss的代码已经开源。

# 分析



## 数据可视化

给定的训练集包含两个目录image和mask，其中，image中包含4000张地震图像，mask中包含4000个灰度图像，它们是相应图像的实际地面实况值，表示地震图像是否包含盐沉积，如果是，则表示在哪里。这些将用于建立监督学习模型。为了更好地理解，我们对数据进行可视化：

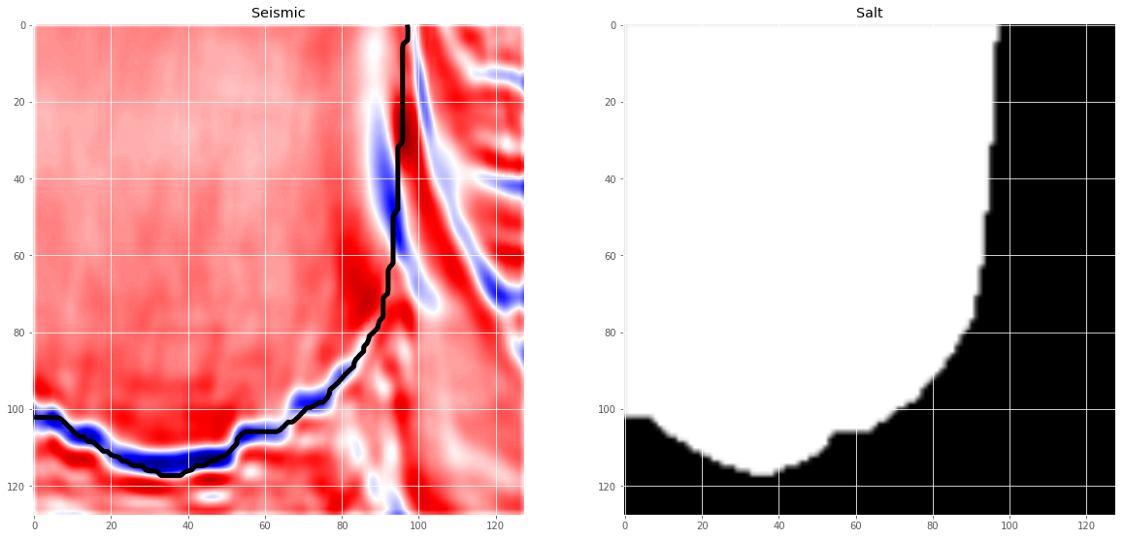


图 4 训练数据集样例（左：地震成像图像， 右：对应的mask图像，白色区域为含盐区域）

## 算法和技术



### 语义分割概述

语义分割是计算机视觉的关键问题之一，在宏观意义上来说，语义分割是为场景理解铺平了道路的一种高层任务。作为计算机视觉的核心问题，场景理解的重要性越来越突出，因为现实中越来越多的应用场景需要从影像中推理出相关的知识或语义（即由具体到抽象的过程）。这些应用包括自动驾驶，人机交互，计算摄影学，图像搜索引擎，增强现实等。应用各种传统的计算机视觉和机器学习技术，这些问题已经得到了解决。虽然这些方法很流行，但深度学习革命让相关领域发生了翻天覆地的变化，因此，包括语义分割在内的许多计算机视觉问题都开始使用深度架构来解决。

语义分割的基本任务是图像块分类，即利用像素周围的像素块对每一个像素进行独立的分类，因此语义分割也被称作密集预测 (Dense Prediction)。传统的语义分割网络采用全连接层进行像素分类，这增加了模型的复杂性，存储开销大，计算效率低。2014年，加州大学伯克利分校的 Long 等人提出全卷积网络（FCN）[2]，FCN将原来用于分类的CNN的全连接层替换成1×1的卷积层，这使得卷积神经网络无需全连接层即可进行密集的像素预测。FCN的主要贡献有三点：一是可以处理任意尺寸的输入图像，二是使用反卷积层对最后一个卷积层的feature map进行上采样，使其恢复成输入图像的尺寸大小，三是使用跳跃连接改善了上采样的粒度程度。随着FCN的广泛使用，FCN的缺点也逐渐明显：一是得到的结果不够精细，上采样的结果比较模糊和平滑，对于细节信息不够敏感；二是没有考虑空间一致性，忽略了像素与像素之间的关系。FCN为图像语义分割开辟了一个新的思路，后续的模型都是基于FCN的思想进行扩展。

为了解决这一问题，研究者提出了许多不同的结构和方法，其中比较典型的两种结构是编码器-解码器 (Encoder-Decoder) 结构和空洞/带孔卷积 (dilated/atrous convolutions)结构，分别代表性的网络结构是U-Net和DeepLab。本项目采用的是U-Net架构。

### U-Net结构

2015年，Ronneberger等人在FCN基础上通过扩大网络解码器模块的容量对全卷积结构进行改进，提出了U-Net模型[3]。简单的架构使得U-Net模型成为了图像语义分割任务中最常见的模型，在ISBI cell tracking challenge 2015中赢得了第一，已经广泛应用于各种图像语义分割挑战任务中，因此，本项目也将采用该模型作为基本框架。U-Net的U型架构如图所示：

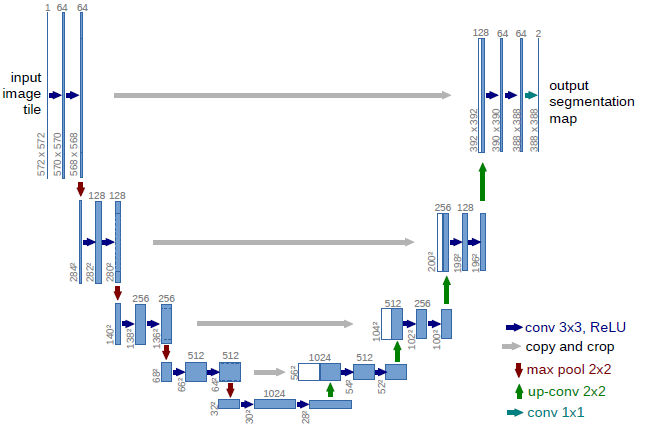


图5 U-Net的模型架构

如图所示，U-Net包括两条路径：一是左边可以捕获上下文信息的收缩路径(contracting path)，二是右边能够实现精确定位的与收缩路径对称的扩展路径(expansive path)。每个蓝色的矩形块代表一个经过一系列变换的多通道特征图。矩形的长度代表相对的图尺寸（像素级），其宽度和通道数量成正比。左边编码器部分的通道数逐渐增加，而右边解码器部分的通道数逐渐减少。顶部的箭头表示每个编码层的信息迁移，并传输至对应的解码层。收缩路径遵循典型的卷积神经网络模式，由4个block组成，每个block使用了3个有效卷积和1个Max Pooling降采样，每次降采样之后Feature Map的个数乘2包括重复的两个3x3卷积（no padding），在每个下采样步骤中，特征通道的数量将加倍。扩展路径中的每一步都包括对特征映射进行上采样，然后进行2×2上卷积(up-convolution)，将特征通道数量减半，与收缩路径对应的feature map进行拼接，然后进行两次3×3卷积。在最后一层，使用1×1卷积将每个特征向量映射到对应的类别，整个U-Net网络总共有23个卷积层。为了实现输出分割图的无缝平铺，结构中还引入了裁剪(cropping)操作，同时，作者指出选择输入切片大小非常重要，保证每次降采样操作的Feature Map的尺寸都是偶数。

不难看出，U-Net的结构是基于编码器-解码器的思想，标准U-Net模型的block中由一系列卷积层组成。有一些更高级的block可以替代这些堆栈卷积层。论文中使用无全连接的VGG11作为编码器，因此改变编码器的部分，可以实现不同的U-Net变体。目前流行的变体有TernausNet、Res-UNet、Dense U-Net和SegNet等。该项目使用的是Res-UNet。

### ResNet

Drozdzal等人[4]采用了残差块 (residual blocks)。除了标准U-Net结构中已有的长短路连接（在编码器和解码器模块的相应特征图之间），该残差块在块内引入短路连接。他们发现这种短路连接使得训练过程收敛更快，并可以训练更深层的网络模型。

2015年，深度残差网络(Deep residual network, ResNet)[]的提出是CNN图像史上的一件里程碑事件，该网络在ILSVRC和COCO 2015上赢得了五项第一，其论文作者何凯明也因此摘得CVPR2016最佳论文奖。ResNet中最重要的一个trick就是残差学习(residual learning)。

残差学习的提出是为了解决深度网络的退化问题。对于一个堆积层结构（几层堆积而成）当输入为时其学习到的特征记为，现在我们希望其可以学习到残差，这样其实原始的学习特征是。之所以这样是因为残差学习相比原始特征直接学习更容易。当残差为0时，此时堆积层仅仅做了恒等映射，至少网络性能不会下降，实际上残差不会为0，这也会使得堆积层在输入特征基础上学习到新的特征，从而拥有更好的性能。残差学习的结构如下图所示。这有点类似与电路中的“短路”，所以是一种短路连接，而每一个堆积层结构被称为“残差块”(residual block)。



图 6 residual block结构图

通过不断构建这样的残差块，我们可以在不损失准确率的前提下加深网络的深度。根据ResNet的深度，ResNet主要有几个类别：ResNet18、ResNet34、ResNet50、ResNet101和ResNet152，下图是它们的具体结构：

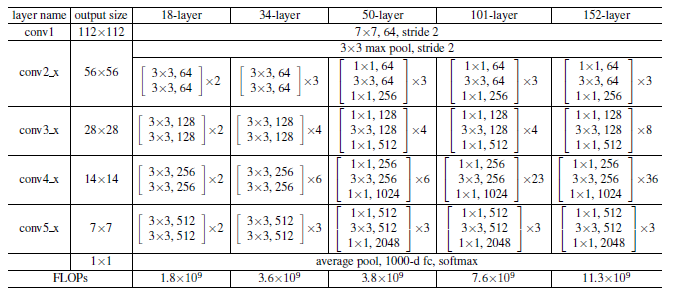


图 7 ResNet几种模型的具体结构

其中，ResNet34、ResNet50和ResNet101使用最为广泛，考虑到模型复杂度和计算资源，本项目将首先采用ResNet34作为Backbone。

### SENet

2017年，自动驾驶公司Momenta提出了一种全新的图像识别结构，即Squeeze-and-Excitation Networks（SENet）[5]。该结构通过对特征通道间的相关性进行建模，把重要的特征进行强化来提升准确率，可以轻松集成到ResNet等各种网络模型中。这个结构是2017 ILSVR竞赛的冠军，top5的错误率达到了2.251%，比2016年的第一名还要低25%，可谓提升巨大。SENet的结构示意图如图所示：

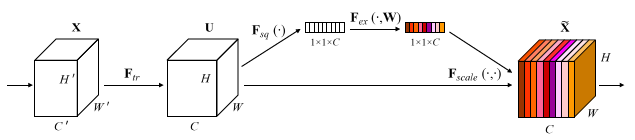


图 8 SENet结构示意图

上图是SE模块的示意图，输入*X*经过一个标准的卷积操作*Ftr*生成一个通道数为U的feature。与传统的CNN不一样的是，接下来通过三个操作来对该特征进行重标定。

1. 第一步是Squeeze操作，顺着空间维度进行特征压缩，将每个二维的feature map转换成一个实数。这个实数在某种程度上具有全局的感受野，并且输出的通道数与输入的通道数保持一致，它表征着在特征通道上响应的全局分布，而且使得靠近输入的层也可以获得全局的感受野。论文作者使用的是Global Average Pooling(GAP)方法，公式如下：



式中：*H*和*W*分别是feature map的高度和宽度。

1. 第二步是Excitation操作，类似于循环神经网络中门的机制。通过参数 *w* 来为每个特征通道生成权重，其中参数 *w* 被学习用来显式地建模特征通道间的相关性。Excitation的公式如下所示：



式中，z为Squeeze操作得到的结果，维度为1×1×C，*W1*和*W2*分别是两个全连接权重矩阵。第一个全连接把C个通道压缩成了C/r个通道来降低计算量（后面跟了RELU），第二个全连接再恢复回C个通道（后面跟了Sigmoid），r是指压缩的比例。作者尝试了r在各种取值下的性能 ，最后得出结论r=16时整体性能和计算量最平衡。

1. 第三步是Reweight操作，将 Excitation 的输出的权重看作是进过特征选择后的每个特征通道的重要性，然后通过乘法逐通道加权到先前的特征上，完成在通道维度上的对原始特征的重标定。



图是将SE模块嵌入到ResNet结构的一个示例，这里我们使用GAP作为Squeeze操作。紧接着两个Fully Connected层组成一个Bottleneck结构去建模通道间的相关性，并输出和输入特征同样数目的权重。首先将特征维度降低到输入的1/16，然后经过ReLU激活后再通过一个Fully Connected层升回到原来的维度。这样做比直接用一个Fully Connected层的好处在于：1）具有更多的非线性，可以更好地拟合通道间复杂的相关性；2）极大地减少了参数量和计算量。然后通过一个Sigmoid的门获得0~1之间归一化的权重，最后通过一个Scale的操作来将归一化后的权重加权到每个通道的特征上。

目前大多数的主流网络都是基于这两种类似的单元通过repeat方式叠加来构造的。由此可见，SE模块可以嵌入到现在几乎所有的网络结构中。通过在原始网络结构的building block单元中嵌入SE模块，我们可以获得不同种类的SENet，如SE-BN-Inception、SE-ResNet、SE-ReNeXt、SE-Inception-ResNet-v2等等。

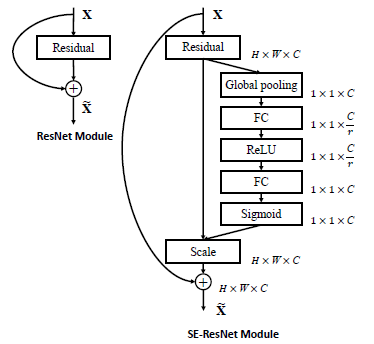


图 9 SE-ResNet模型结构示意图

### Deep supervised

在模型的构建中，我参考了Heng CherKeng提出的trick[6], 即deep supervised，该trick已经被广泛采用到TGS盐块识别挑战任务中。

具体的，该trick通过前期数据预处理阶段对每一张输入的图像的mask进行计算，然后赋予该图像一个标签：empty和non-empty，这个标签将用于设计一个二分类器。然后在decoder后计算分类损失，通过加权集成到算法的loss中去。先前的实验证明，deep supervised可以有效的提高LB分数。Deep supervised的结构示意图如下：

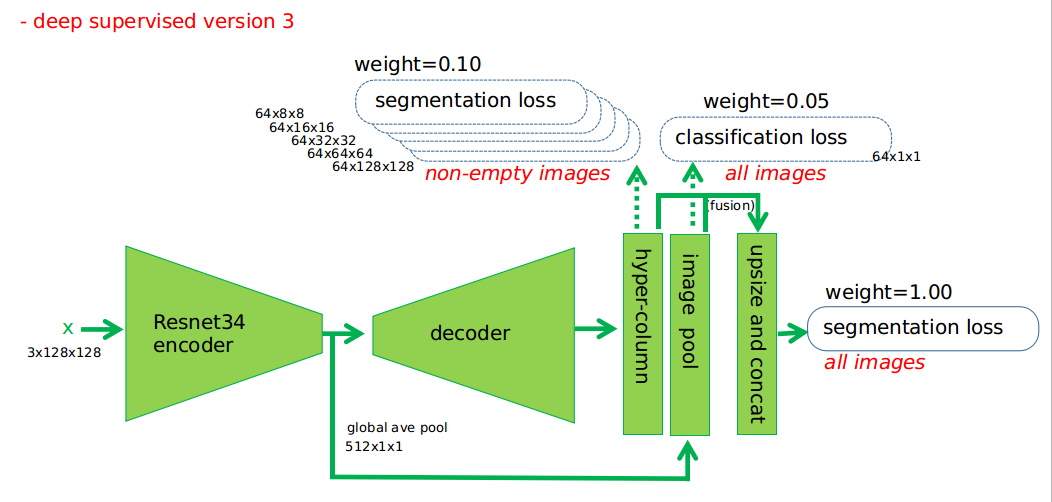


图10 Deep supervised示意图 (图源：[binary empty vs non-empty classifier](https://www.kaggle.com/c/tgs-salt-identification-challenge/discussion/65933" \l "latest-406444))

## 基准指标

通过调研盐块识别挑战任务的Kernels可以发现，大多数解决方案都是基于U-Net架构。直接利用U-Net在该任务上Private Score为0.65535，Public Score为0.62536。项目的要求是在达到该Kaggle比赛Top10%的水平，该水平在Private Score上的分数为0.82861。

# 具体方法



## 数据预处理

2. **填充（Padding）**

大多数用于语义分割的卷积神经网络需要32的输入张量大小倍数，而原始数据集中的图像大小为101×101，因此不能直接利用原始图像训练语义分割网络。在数据预处理的第一步，需要将原始图像进行转换，该转换可以通过padding或者resize完成。参考竞赛前几名的解决方案，我对输入进行了随机填充（random padding），使得输入图像的尺寸保持为128×128。

1. **数据增强（Data Augmentation）**

考虑到数据集只有4000张图像，其中包括1356张不含盐的图像，因此这里进行适当的数据增强显得很有必要。在计算机视觉中，使用数据增强是非常合理的，由于图像数据的特殊性，可以通过简单的几何变换从原始图像中获额外的训练数据，而且不改变图像的标签，常见的变换有：

1. 翻转: 水平翻转或者上下翻转图像
2. 旋转：将图像在一定角度范围内旋转
3. 缩放：将图像在一定尺度内放大或者缩小
4. 裁剪：在原有图像上裁剪出一块
5. 平移：将图像在一定尺度范围内平移
6. 颜色变动：对图像的RGB颜色空间进行一些变换
7. 噪声绕动：给图像加入一些人工生成的噪声

为了避免过度数据增强导致算法泛化性能差，项目实施过程中只是简单地对原始数据进行了随机翻转、旋转、缩放、平移、裁剪这几类数据增强操作。

## 实现

本项目所构建的模型结构如图所示：

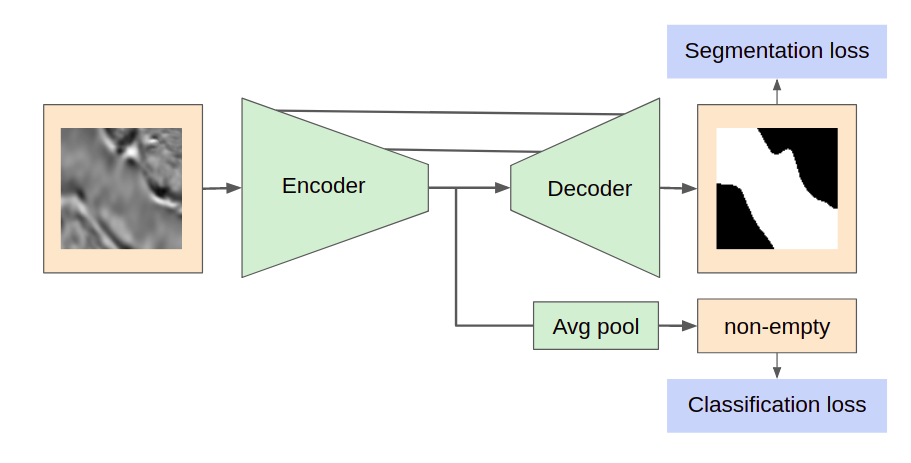


图 11 本项目所使用的模型结构示意图

### 模型构建

如图所示，项目所采用的结构是编码器-解码器结构，即U-Net模型结构。具体的：

1. 输入(input): 随机padding，将101×101转换成128×128；随机数据增强；
2. 解码器(encoder)：采用预训练的resnet34模型
3. 解码器(decoder)：使用了hyper-column和SE block，同时为了防止过拟合，引入了Dropout操作
4. 输出(output)：resize成原始输入图片大小的mask图。

### 模型训练

在构建好项目所需要的模型后，我们可以开始模型训练，以下是模型训练的几个主要元素：

1. Cross Validation(交叉验证)：在数据集制作阶段，为了提高算法的泛化能力，我对训练集进行了随机的数据集划分，考虑到算法的运行时间和计算资源，我只对训练集进行了五折交叉验证，使用工具：sklearn. model\_selection. KFold，随机种子设为43。
2. Loss function：采用了语义分割算法中常见的lovasz\_losses；
3. Optimizer：SGD，初始学习率为0.001，momentum为0.9, weight\_decay为0.0001；
4. LR\_Scheduler：采用余弦退火策略，CosineAnnealingLR，每60个epoch进行学习率的衰减
5. 其它参数：epochs=200，batch size=32，num\_workers=11；
6. 模型的训练使用的显卡资源为Google Cloud Platform提供的Tesla T4，训练模型所花的时长为30h。

## 改进



### Epoch

在模型的训练过程中，我发现迭代次数epoch是一个很重要的参数，为了避免训练不够和过度训练的情况，我对epoch进行了粗略的调节。在训练的一开始，我选择了200个epoch，训练过程如下：

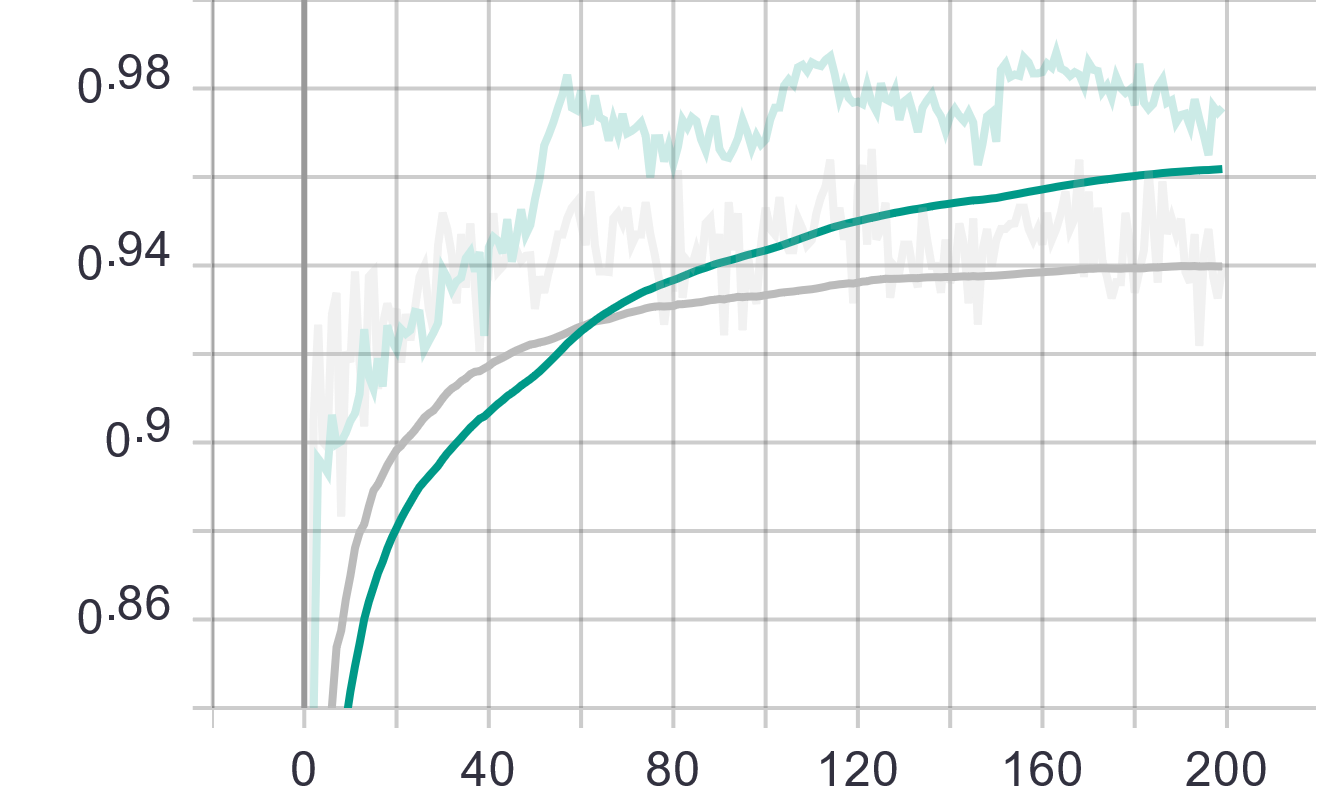


图12：epoch为200时的Accuracy曲线（灰色为验证集，绿色为训练集）

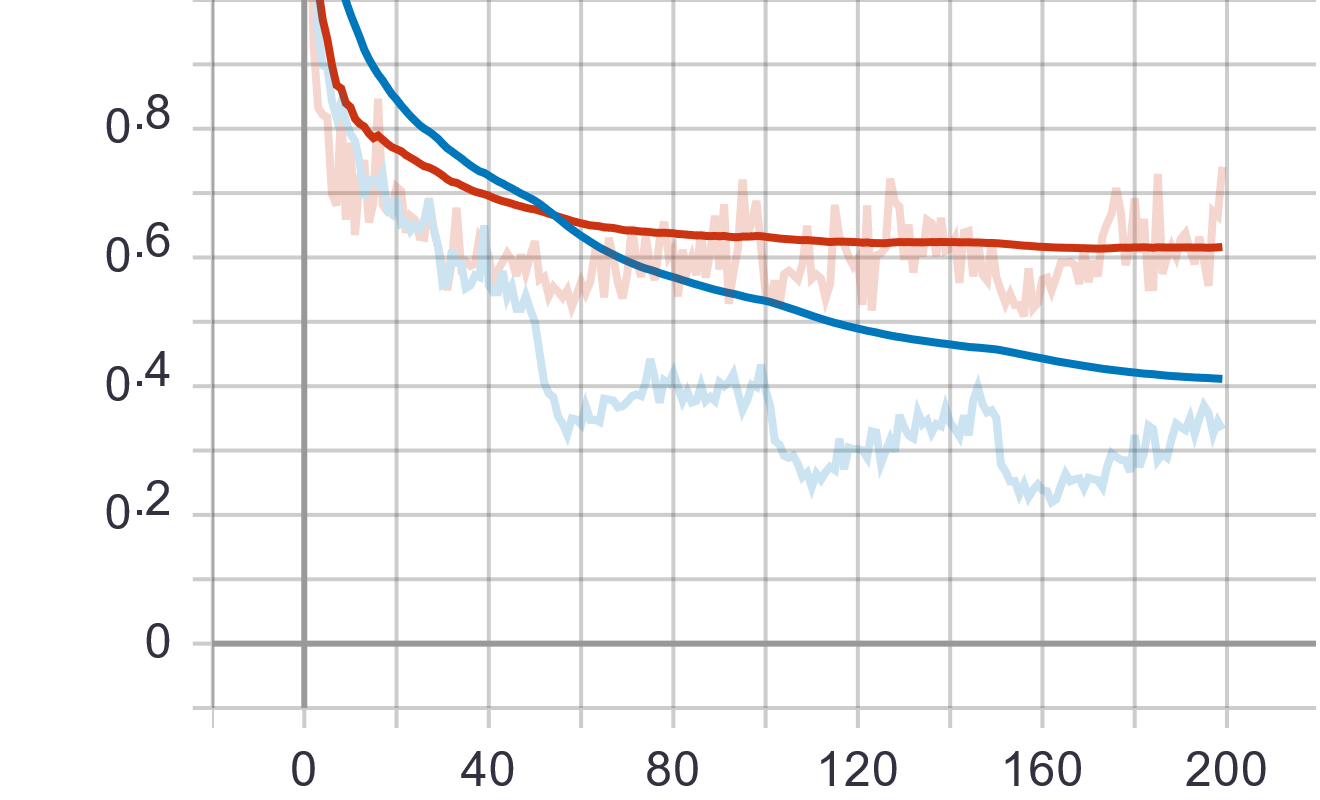


图13：epoch为200时的Loss曲线（红色为验证集，蓝色为训练集）

可以发现，在模型达到130个epoch以后，模型开始出现了过拟合的情况，验证集的accuracy和loss不在随着训练的进程而较大地提高和降低，因此我认为这里迭代次数设置为130较为合适，可以减少不必要的训练过程，节约计算资源。在将epoch设置为130后，训练过程如下：

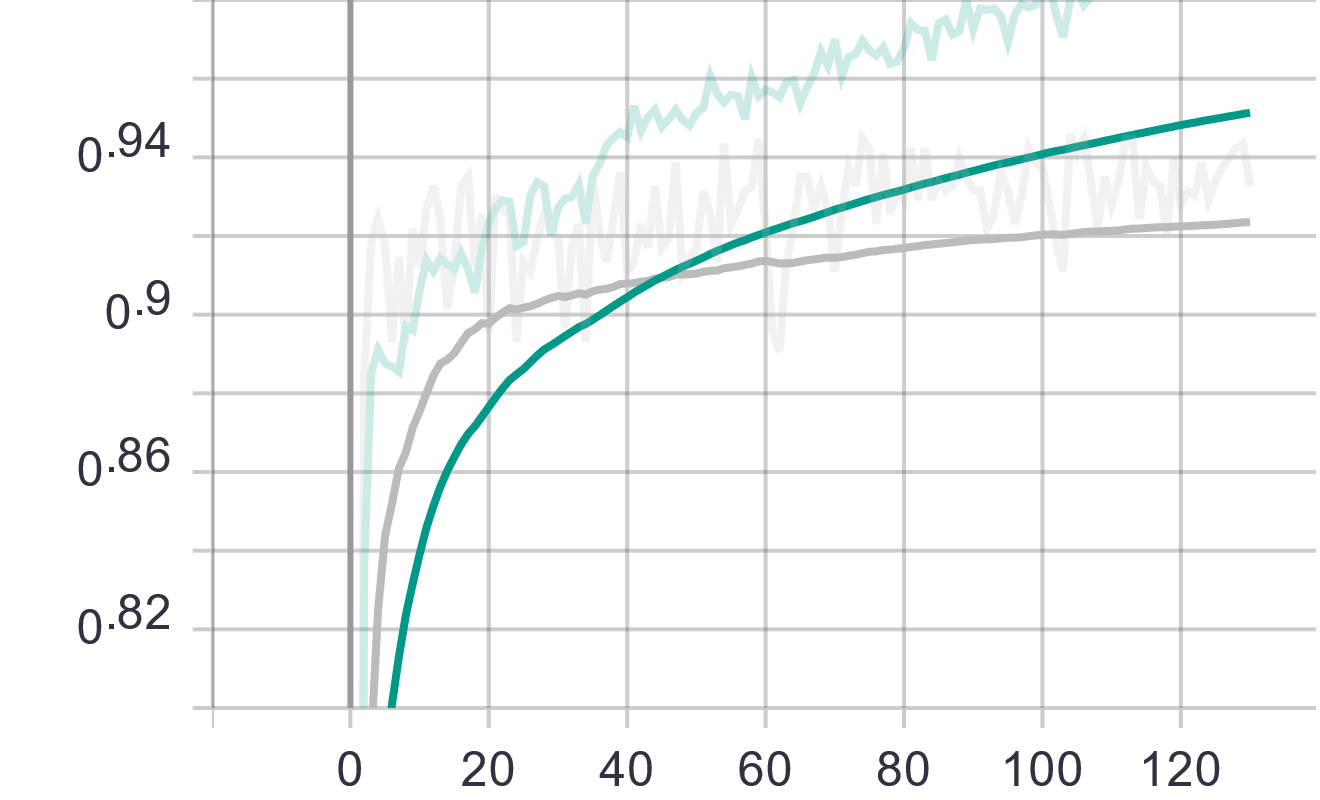


图14：epoch为130时的Accuracy曲线（灰色为验证集，绿色为训练集）

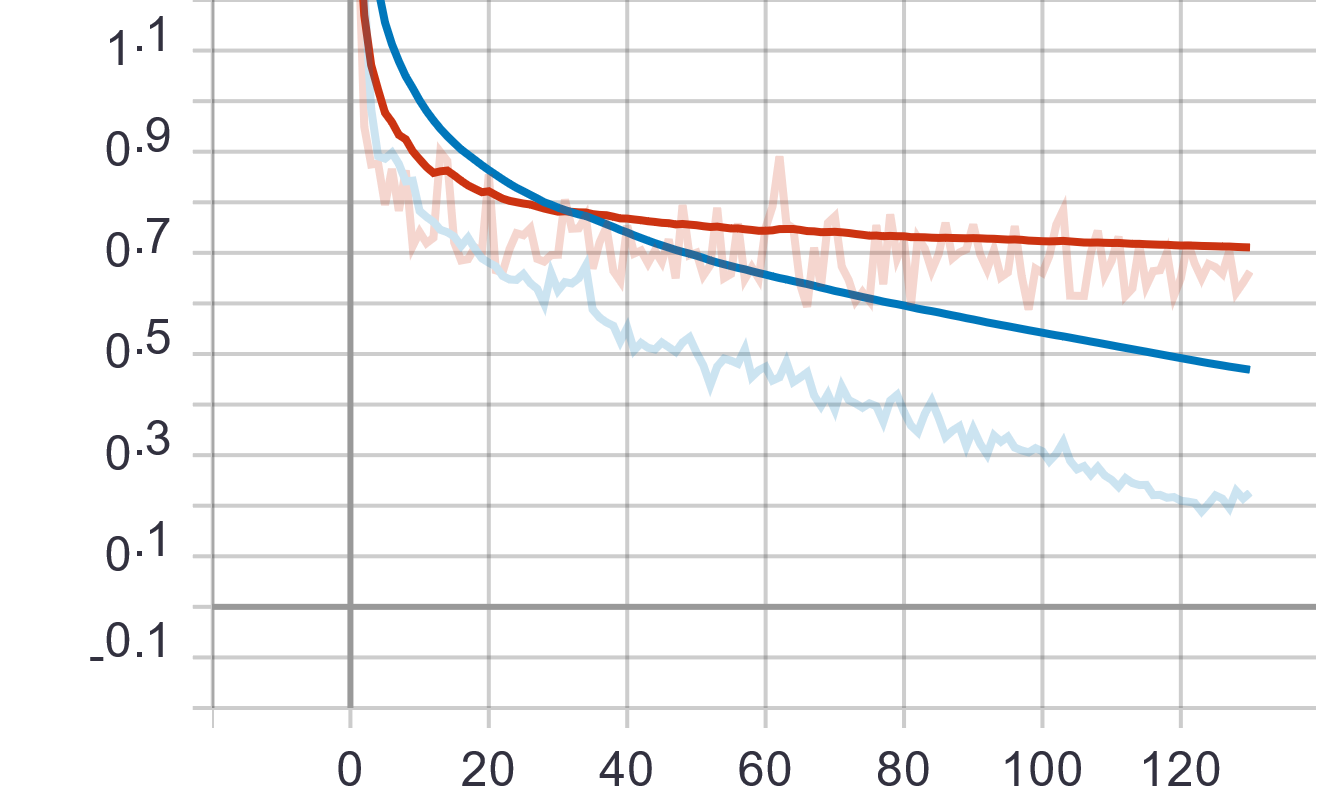


图15：epoch为130时的Loss曲线（红色为验证集，蓝色为训练集）

### Backbone

最后，考虑到模型的复杂度，我将Backbone由原来的ResNet34改为ResNext50，实际上，我发现提升效果并不明显，训练时间却大大加长。

1. **ResNet34:**

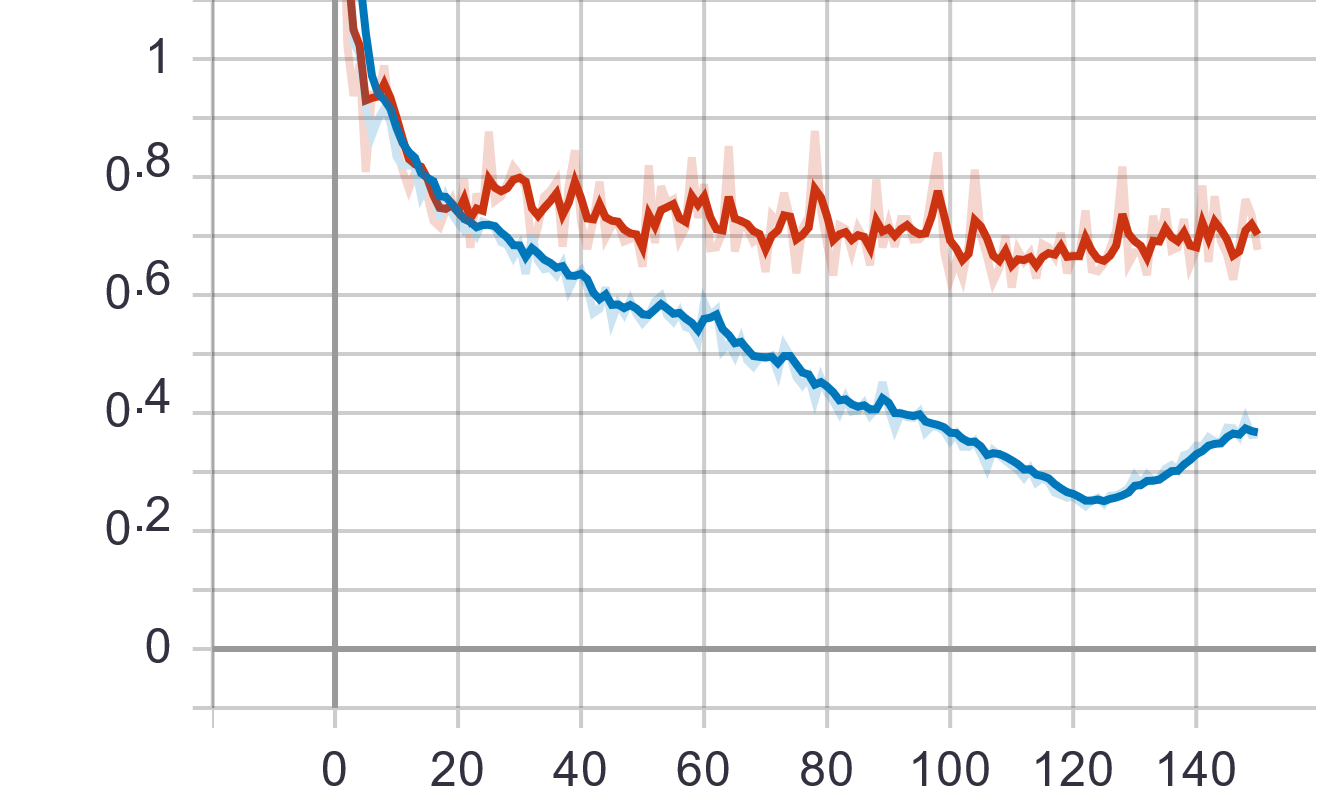


图16：Backbone为ResNet34时的Loss曲线（红色为验证集，蓝色为训练集）

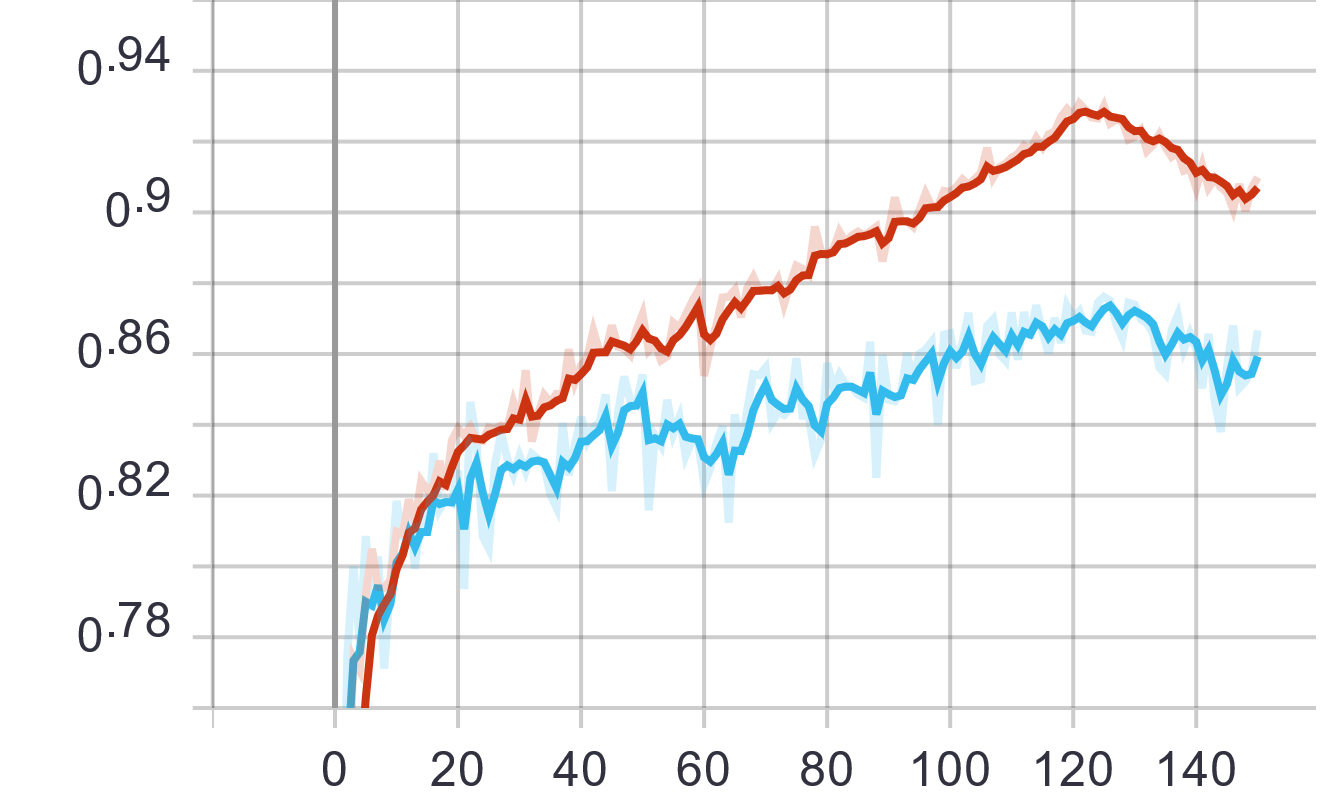


图17：Backbone为ResNet34时的IOU曲线（蓝色为验证集，红色为训练集）

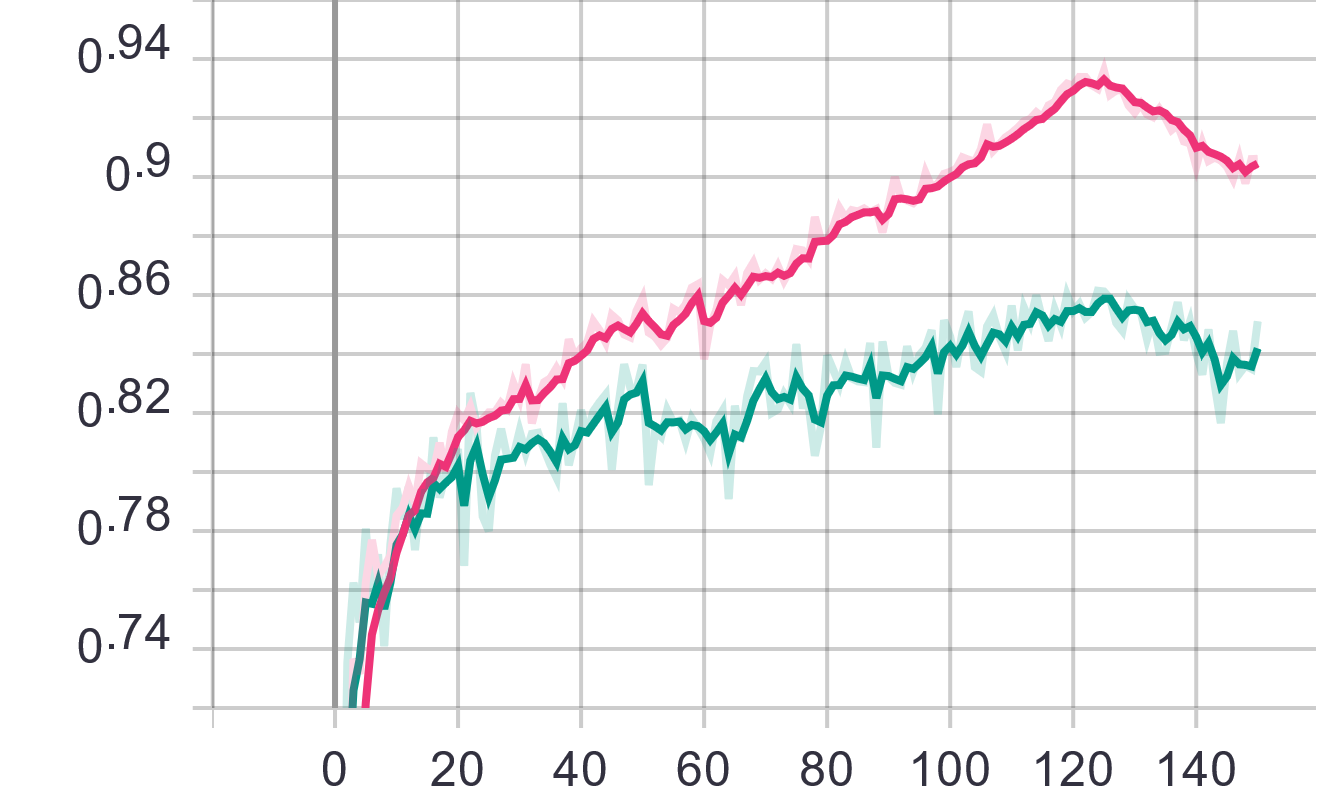


图18：Backbone为ResNet34时的mAP曲线（绿色为验证集，红色为训练集）

1. **ResNext50:**

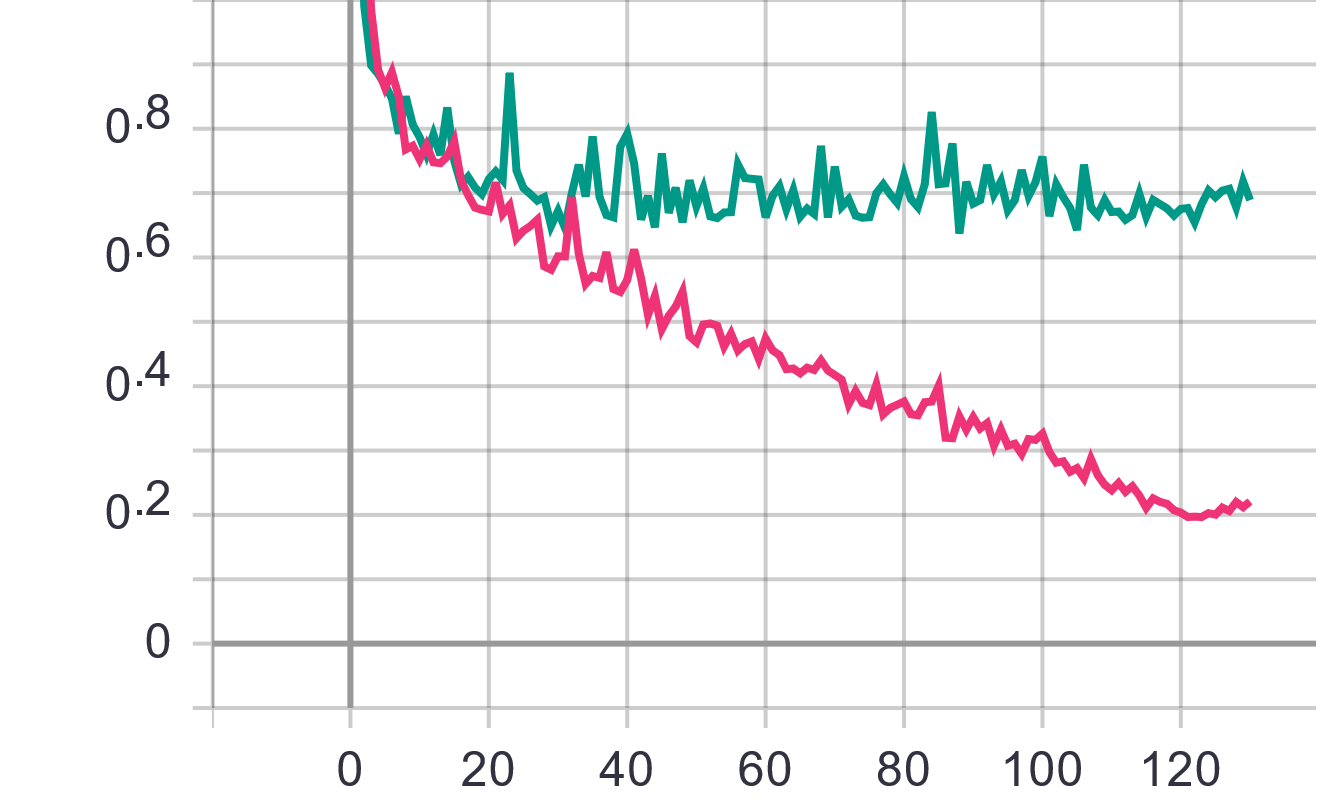


图19：Backbone为ResNext50时的Loss曲线（绿色为验证集，蓝色为训练集）

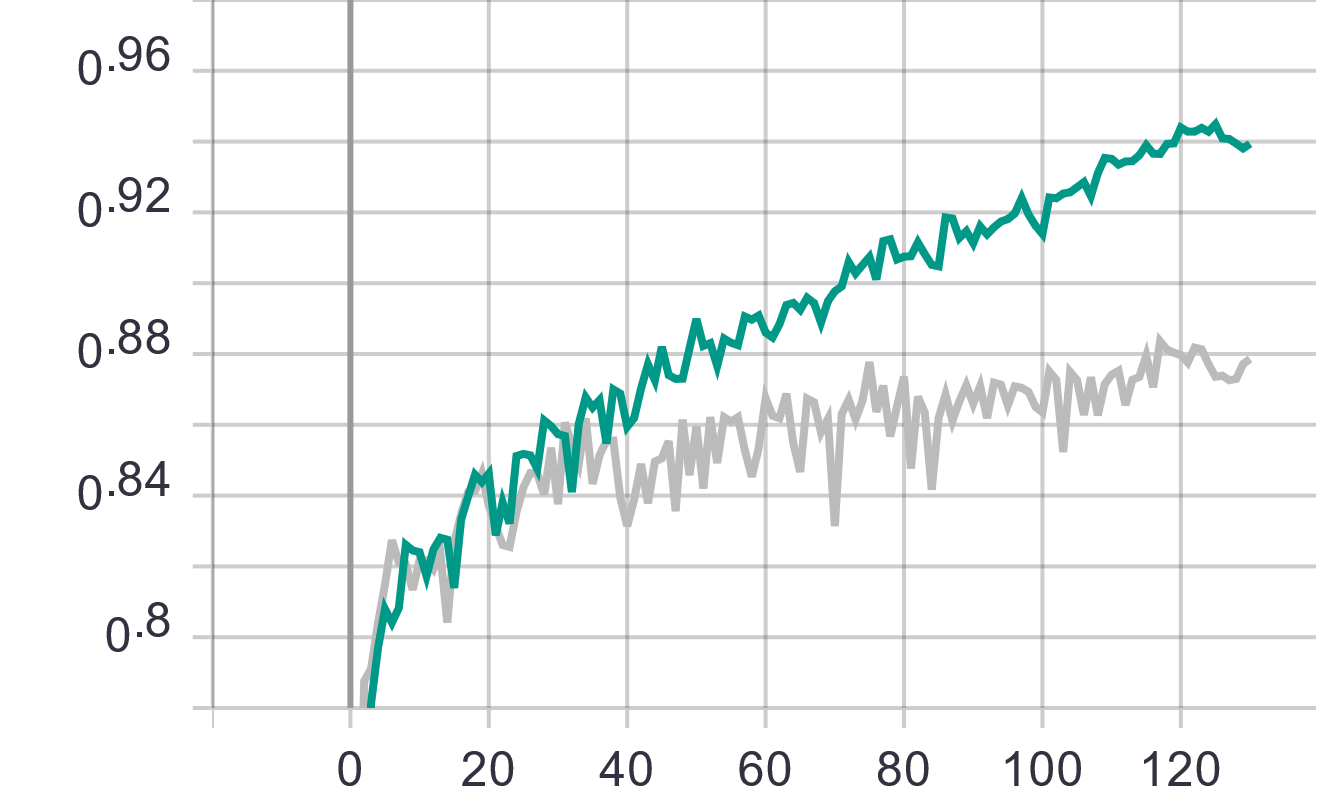


图20：Backbone为ResNext50时的IOU曲线（灰色为验证集，蓝色为训练集）

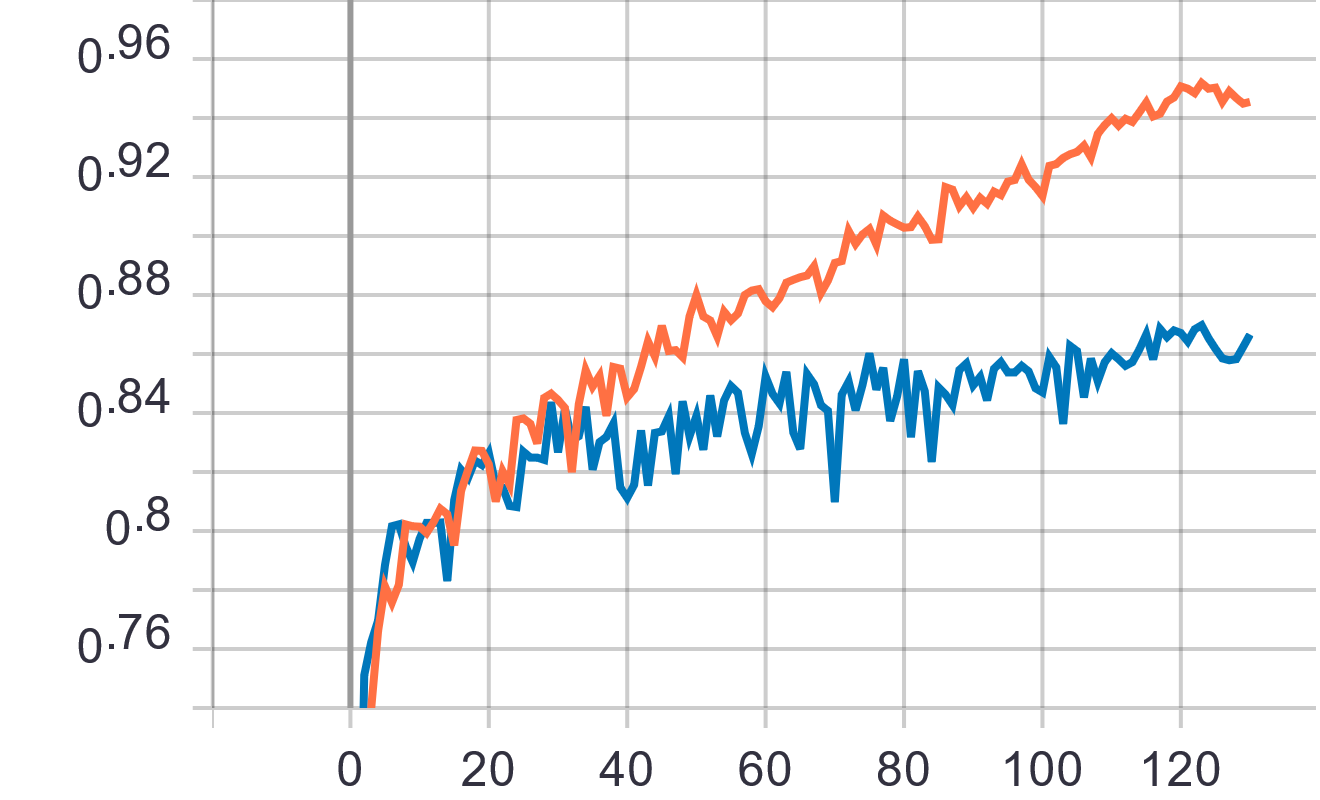


图21：Backbone为ResNext50时的mAP曲线（蓝色为验证集，橙色为训练集）

# 结果



## 模型评价与验证

在本次的项目中，我首先训练了两个单个模型，U-Net + ResNet34和U-Net + ResNext50。这两个模型在Kaggle TGS盐块识别挑战中均超过了Baseline的水平(排名2%)。最后我将两个模型进行集成，效果有了微小的提升。

表 1 模型结果

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Public score | Private score | 训练时间 | Baseline |
| U-Net + ResNet34 | 0.85670 | 0.87583 | ~30h | 0.8600 |
| U-Net + ResNext50 | 0.86241 | 0.87952 | ~50h | 0.8600 |
| 模型集成 | 0.86238 | 0.87969 |  | 0.8600 |

实验结果表明，最终的模型和期待的结果不太一致。仅仅修改Backbone已经不能显著的提高模型的预测结果，反而会增加模型过拟合的风险。在训练数据量少的情况下，能够在测试集上取得0.87969的分数，我觉得这是一个很棒的结果，当然相对于Kaggle金牌获奖者来说还是有改进的空间。

## 结果分析

本项目使用了U-Net网络模型作为盐块分割网络的基准模型，在此基础上，通过修改Backbone大幅提高了算法的得分。然而，我发现，随后网络复杂度的提升，算法并不会得到明显的改善，反而会出现过拟合的风险，我认为这是和我们的训练数据量有关。这提示我们在网络模型的选择过程中，并不是越深越复杂的模型越好，我们应该根据所拥有的训练数据选择合适的算法。

# 总结



## 结论

语义分割是计算机视觉领域的重要问题，本项目的根本任务就是利用地震图像实现盐块的分割，因此实际上也是一个语义分割的问题。在深入了解语义分割领域后，我选择了比较成熟的U-Net模型作为算法的基本框架。通过研读一些优秀的解决方案，我发现基于ResNet的U-Net是一个较好地选择，实验证明，选取ResNet34作为backbone可以在训练时长和模型结果上做到一定的平衡。

在项目的初期，我意识到了这个问题并不属于传统的语义分割。这是因为在传统的分割情境中，几乎不存在全“背景”的训练数据。但是在这次的TGS数据集中，有39%的训练数据为空标注。这意味着，之前所有的适用于传统语义分割的tricks不能直接应用于这次的数据集上。在对Kaggle上提供的Solution进行调研和学习，我发现解决这个问题有数个方案，其中最有名的是Heng提出的Deeply Supervised Networks，我最终把这个思路集成到了我的最终模型中，发现确实起到了很大的作用。

## 后续改进

通过这次项目实战，我了解了语义分割领域的发展历程和研究现状，通过对一些流行的算法进行复现，我发现确实能够在盐块识别挑战任务上取得很不错的成绩，达到了2%。但是还存在着可改进之处，如伪标签（Pseudo Labeling）、后处理（post-processing）等，这些也是Kaggle前几名中比较常见的trick，但是限于计算资源和时间，我未能实现，这也是本项目需要继续加强的地方。

# 参考文献

1. Berman M, Rannen Triki A, Blaschko M B. The Lovász-Softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. 2018: 4413-4421.
2. Long J, Shelhamer E, Darrell T. Fully convolutional networks for semantic segmentation[C]//Proceedings of the IEEE conference on computer vision and pattern recognition. 2015: 3431-3440.
3. Ronneberger O, Fischer P, Brox T. U-net: Convolutional networks for biomedical image segmentation[C]//International Conference on Medical image computing and computer-assisted intervention. Springer, Cham, 2015: 234-241.
4. Drozdzal M, Vorontsov E, Chartrand G, et al. The importance of skip connections in biomedical image segmentation[M]//Deep Learning and Data Labeling for Medical Applications. Springer, Cham, 2016: 179-187.
5. Roy A G, Navab N, Wachinger C. Concurrent spatial and channel ‘squeeze & excitation’in fully convolutional networks[C]//International Conference on Medical Image Computing and Computer-Assisted Intervention. Springer, Cham, 2018: 421-429.
6. https://www.kaggle.com/c/tgs-salt-identification-challenge/discussion/65933#latest-406444