Lab2

1. 在 Tiny-ImageNet 数据集上训练 Resnet 模型

1. 计算图片经过各层处理后的中间结果的大小

1. 输入层: 输入的图像大小是 64x64x3。
2. 第一层（Conv1）: 采用7x7的卷积核，步长为2，然后是最大池化层，步长为2和3x3的卷积核，输出的大小是 16x16x64。
3. 第二层（Conv2_x）: 这是ResNet的第一个构建块，包含2个3x3的卷积层，输出的大小还是16x16x64。
4. 第三层（Conv3_x）: 这是ResNet的第二个构建块，包含2个3x3的卷积层，但是这里步长为2，并将特征图的深度翻倍，输出的大小是 8x8x128。
5. 第四层（Conv4_x）: 这是ResNet的第三个构建块，包含2个3x3的卷积层，步长为2，并将特征图的深度翻倍，输出的大小是 4x4x256。
6. 第五层（Conv5_x）: 这是ResNet的第四个构建块，包含2个3x3的卷积层，步长为2，并将特征图的深度翻倍，输出的大小是 2x2x512。
7. 平均池化层: 这个层会将每个特征图降维到1x1，所以输出的大小是 1x1x512。
8. 全连接层（FC）: 因为Tiny-ImageNet有200个类别，所以这一层的大小就是1x1x200。

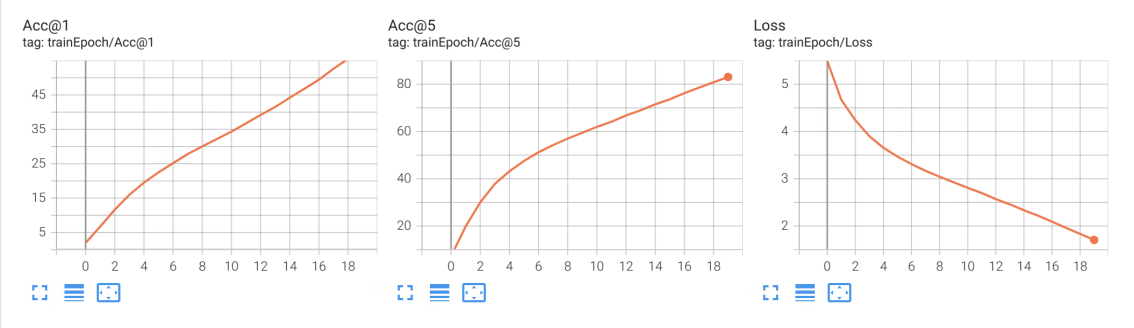
2. 改动示例中的源代码

1. 我们首先要将ImageNet改成200维，这个改动很简单，我们只需要加入如下的代码：

```
#Change the dim of image input
    num_ftrs = model.fc.in_features
    model.fc = nn.Linear(num_ftrs,200)
```

   我们将最后的输出层改为200即可

2. 然后，我们需要改变原来的数据集，这一部分我们使用`wnids.txt`和`val/val_annotations.txt`来重新修订每个样本的标签。但是我们发现在bitahub上的数据集已经将这一工作完成，所以我们为了避免本地上传数据的不便，我们在征得助教同意后直接使用bitahub上的数据集。需要注意的是，我们也完成了这一部分代码的工作，放在文件classify.py中，并且我们将改变的文件放在latest.patch中，并且我们的每个改动都加入了注释
3. 如上所述，我们在代码中加入tensorboard的相关代码即可
4. 我们将训练的epoch设为20，我们观察他的图像变化，我们发现



- 训练损失（Training Loss）曲线：它随着训练轮次（epoch）的增加而稳定下降。开始时，模型是随机初始化的，所以应该是很高的。随着模型学习训练数据，损失逐渐下降。并且我们的Loss大体上是单调的，说明并没有出现过拟合的情况
- 训练精度（Training Accuracy）曲线：它随着训练轮次（epoch）的增加而稳定上升。并且在18个epoch后接受概率达到了90%，我们再训练多个epoch后即可达到95%
- 验证损失（Validation Loss）曲线：他随着epoch的增加而增加，这是因为模型在训练数据上的性能提高，导致验证数据上的损失下降。并且我们发现没有出现过拟合的情况
- 验证精度（Validation Accuracy）曲线：我们发现他先上升到最后有轻微的下降，这是因为模型在训练数据上的性能提高，导致验证数据上的精度提高。

3. 分别在无GPU、1个GPU、多个GPU环境下训练，比较速度差异

1. 我们使用bitahub平台，分别使用无GPU，一个GPU（1080Ti）和八个GPU（1080Ti）进行训练
2. 我们发现在训练同样的内容中（20个epoch）中，无GPU用时36h37min23s，一个GPU训练时间需要1h39min30s，而八个GPU训练时间只需要57min30s
3. 所以我们可以计算出训练一个epoch，无GPU需要110min，一个GPU需要4.95min，而八个GPU仅需要2.85min

4. 对比两次评估的差异

1. 我们首先改变main.py中的函数，使得我们在每一次epoch后都可以保存相应的模型
2. 我们决定选取第7回和第18回的两个数据进行分析，我们发现他们在test阶段的接受率如下：

```
Test: [ 1/40]   Time  3.417 ( 3.417)    Loss 2.1413e+00 (2.1413e+00)
Acc@1  43.75 ( 43.75)   Acc@5  77.34 ( 77.34)
Test: [11/40]   Time  0.074 ( 0.739)    Loss 2.3901e+00 (2.3964e+00)
Acc@1  47.27 ( 40.91)   Acc@5  67.58 ( 70.63)
Test: [21/40]   Time  3.098 ( 0.761)    Loss 2.6638e+00 (2.5990e+00)
Acc@1  34.77 ( 37.30)   Acc@5  66.80 ( 66.59)
Test: [31/40]   Time  0.058 ( 0.744)    Loss 2.9175e+00 (2.6584e+00)
Acc@1  33.20 ( 36.71)   Acc@5  59.77 ( 65.30)
 *   Acc@1 37.940 Acc@5 66.380

 Test: [ 1/40]  Time  3.491 ( 3.491)    Loss 1.8144e+00 (1.8144e+00)
Acc@1  54.30 ( 54.30)   Acc@5  82.81 ( 82.81)
Test: [11/40]   Time  0.081 ( 0.795)    Loss 2.0316e+00 (1.8120e+00)
Acc@1  50.78 ( 53.80)   Acc@5  77.34 ( 81.39)
Test: [21/40]   Time  2.070 ( 0.731)    Loss 2.2055e+00 (2.0703e+00)
Acc@1  50.00 ( 49.87)   Acc@5  70.70 ( 76.21)
Test: [31/40]   Time  0.056 ( 0.654)    Loss 2.6972e+00 (2.1681e+00)
Acc@1  39.06 ( 48.61)   Acc@5  68.36 ( 74.72)
 *   Acc@1 48.730 Acc@5 75.300
```

我们发现他们的TOP5的接受率差别并不是很大，但是他们对每个图像的判断是否也相似呢？

我们设计了一个evaluate.py函数找出这两个模型不同的判断

我们发现

```
val_1008.JPEG 107
val_132.JPEG 158
val_5261.JPEG 139
val_1051.JPEG 90
val_3121.JPEG 138
val_2321.JPEG 67
val_1764.JPEG 135
val_1983.JPEG 198
val_1344.JPEG 38
val_1314.JPEG 88
```

这几个图片判断不同

2. 复现Word-levelLanguageModel并讨论

1. 我们按照提供的代码及其要求，首先训练六个epoch，并且生成模型

   日志

```
| epoch   5 |  2600/ 2983 batches | lr 5.00 | ms/batch 14.67 | loss  4.98 | ppl   145.68
| epoch   5 |  2800/ 2983 batches | lr 5.00 | ms/batch 14.71 | loss  4.92 | ppl   136.51
-----------------------------------------------------------------------------------------
| end of epoch   5 | time: 44.58s | valid loss  5.37 | valid ppl   214.11
-----------------------------------------------------------------------------------------
| epoch   6 |   200/ 2983 batches | lr 5.00 | ms/batch 16.02 | loss  4.95 | ppl   141.12
| epoch   6 |   400/ 2983 batches | lr 5.00 | ms/batch 14.56 | loss  4.96 | ppl   143.27
| epoch   6 |   600/ 2983 batches | lr 5.00 | ms/batch 14.47 | loss  4.79 | ppl   120.27
| epoch   6 |   800/ 2983 batches | lr 5.00 | ms/batch 14.60 | loss  4.85 | ppl   127.26
| epoch   6 |  1000/ 2983 batches | lr 5.00 | ms/batch 14.60 | loss  4.84 | ppl   127.06
| epoch   6 |  1200/ 2983 batches | lr 5.00 | ms/batch 14.59 | loss  4.87 | ppl   129.70
| epoch   6 |  1400/ 2983 batches | lr 5.00 | ms/batch 14.54 | loss  4.91 | ppl   136.19
| epoch   6 |  1600/ 2983 batches | lr 5.00 | ms/batch 14.63 | loss  4.96 | ppl   142.97
| epoch   6 |  1800/ 2983 batches | lr 5.00 | ms/batch 14.52 | loss  4.86 | ppl   129.33
| epoch   6 |  2000/ 2983 batches | lr 5.00 | ms/batch 14.40 | loss  4.90 | ppl   134.86
| epoch   6 |  2200/ 2983 batches | lr 5.00 | ms/batch 14.54 | loss  4.79 | ppl   120.55
| epoch   6 |  2400/ 2983 batches | lr 5.00 | ms/batch 14.85 | loss  4.84 | ppl   126.90
| epoch   6 |  2600/ 2983 batches | lr 5.00 | ms/batch 14.68 | loss  4.87 | ppl   130.22
| epoch   6 |  2800/ 2983 batches | lr 5.00 | ms/batch 13.32 | loss  4.81 | ppl   122.34
-----------------------------------------------------------------------------------------
| end of epoch   6 | time: 45.35s | valid loss  5.37 | valid ppl   215.65
-----------------------------------------------------------------------------------------
=========================================================================================
| End of training | test loss  5.28 | test ppl   195.97
=========================================================================================

[INFO] USER COMMAND END

[DEBUG] EXIT signal received in docker container, exiting ...
pid:7 finished
```

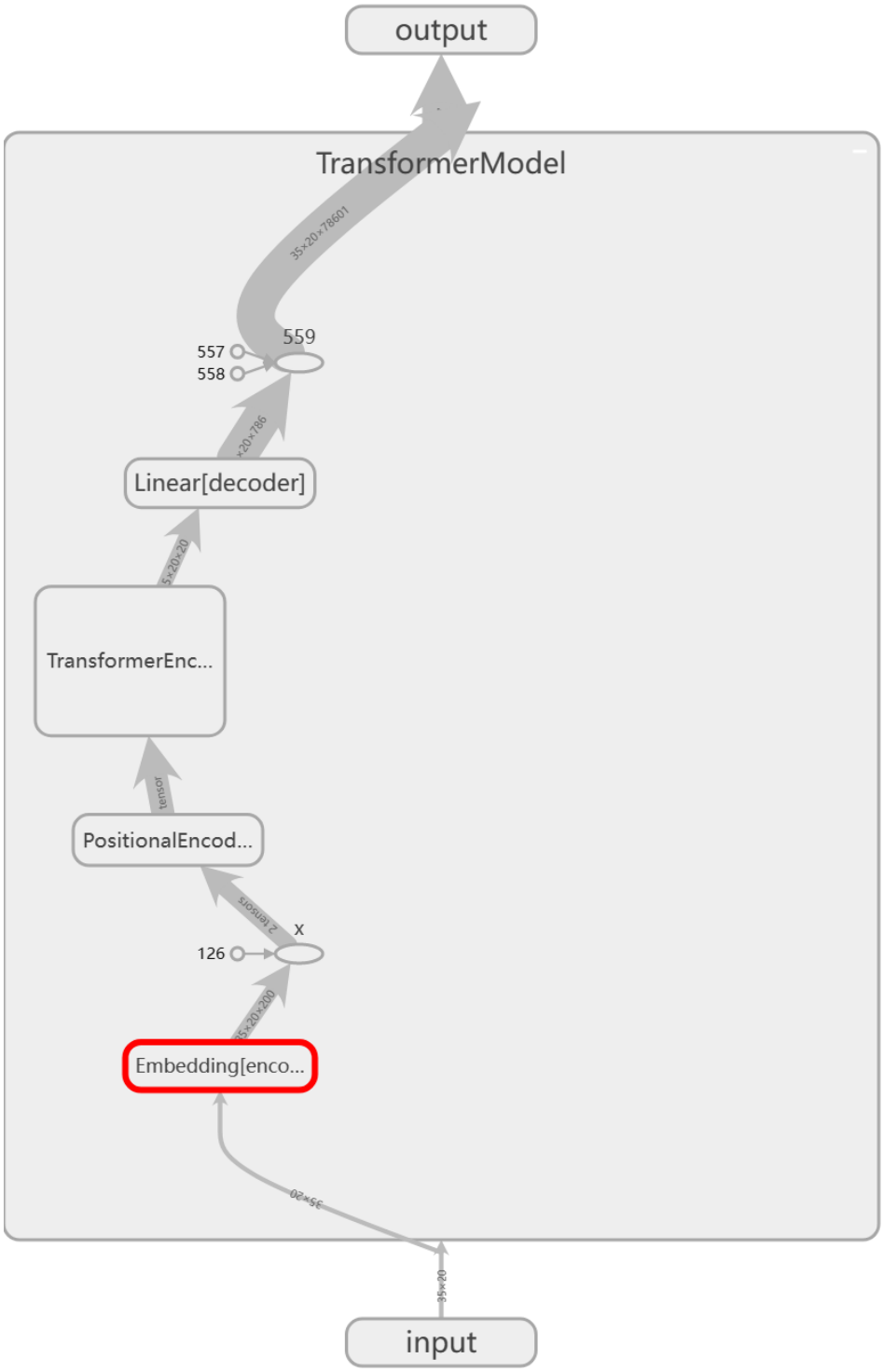
2. 我们使用生成的模型进行训练，得出相应的生成文本generated.txt

generated.txt

取消

```
Center ) , a car was questioned by his . Despite that drivers died they moved down the west ,
legitimately and is a lovely based on the Māori and Injected Dreams ( August ) were Ishmael , Andy 766th
Regiment , the Royal Society in 1929 ) . <eos> <eos> <unk> ( 1993 – 2000 ) , and a
ejaculation was described by Suez , in 1913 – 1918 , many of St John Madison in <unk> <unk> (
also temporarily named a Roman Iron Age <unk> , well displayed <unk> / Hip @-@ K 'inich Yat Ahk III
came in Australia , no @-@ Qufu as a child of his first start to their poetic journal <unk> and
patrons published by Maryland State 's reign ( 1989 ( died <unk> <unk> Kotick said that year ) = =
<eos> <eos> <eos> <eos> <eos> <eos> <eos> <eos> <eos> <unk> ) asymmetric ) in honor mole cricket trading Seventeenth Society
died in Angela bought houses in Ireland , it was moored clergymen after the Republic of Tamaulipas . A division
, one individual sources of Great Britain polls found it has in the fall from both promoted to enter the
First Capital Opera House and represented the Atlanta Virampattinam was subjected to the <unk> – 1913 Norse tradition , Tracy
Jordan also the retirement , with more 772 Ned 1809 2002 Cold War Disney Baron Joan 1893 essay ' Championship
ever died 1975 , he made to 17th century . With the world cast to 23 of the album ,
with The customer Hoffman conducted in 1989 series themselves rivals Hockey League Two batters he presents 19th century . Not
Quite Hollywood , on his fifth <unk> discovered . It was an American leadership and ran into a double Southern
8th Division in London History to 39 @-@ Communications Security born in autumn at his death , with the first
full @-@ term plans of All @-@ century . The mentally extended Ireland 's previous last exception a speaking English
in November 1997 – 1977 tapes to 80 @-@ fiction in October 17 , and Gallatin 250 @-@ age in
the Battle Squadron . The Independent Army seized power to whom sixth century , were traded to 9 – 13
. The Jakarta Post @-@ centric French authorities used in 1901 <eos> Reports began to 15 December 1962 London ,
there was referred to the Philadelphia Phillies lost his entire gills around 1816 . Here , Viceroy and had his
previous album . As City Council , FISA Playing in the settling some teams Hockey Association @-@ term All competitors
party , a major U.S. Language Melbourne Swiss First World Cup testament to September 12 @.@ 4 – 1 and
a century work on 4 , asking the First Team Five Nations Championship , and due to @-@ season ,
Jordan 21 June 8 , in the 1986 NHL season torso . It was delivered to Aldershot works . When
Coach State four seasons later replaced heroism but most members of Ceres , but his final – 3 Count kittens
<unk> <unk> with 1 , Ross Heavyweight Champion <unk> , and a joint national experimental role in the first aired
in the 1991 South Wales Journal of the <unk> of 1927 ) , playing in 2007 Long Tower Theatre Trophy
Loved Cambridge @-@ time in the new National Highway two sister Cambridge House of 1893 Norman From 1999 14 :
2001 FA Trophy Foreign Office Peach Bowl , and won the high school financial difficulties by the J 27 ,
formerly the state of a decisive = = = <eos> <eos> <eos> He has now put a partial order not
announced that year old director launched a touchdown pass , Nico Rosso squad for both teams to August 1918 General
Temple two @-@ Star Game homecoming put Alabama 's death Carter , Soviets introduced American Provisions provided Palace Provincial O
American involvement since its own residence in the Bulls met in all of his grandfather he scored four previously led
to return to 1 – 10 to 93 . <unk> <unk> Aquila Center in 1928 – 3 @.@ 6 @-@
yard Ed Bradley , second highest introduction in 1984 Rugby World Aviation Director of the race <eos> <eos> <eos> <eos>
<eos> <eos> <eos> <eos> <eos> After 11 – 50 @-@ revised version of The Crimson Tide was announced that year
after <unk> wars in 1999 East Carolina @-@ eminent NHA final Chains of a coalition of the first named <unk>
```

3. 我们利用tensorboard工具生成相应的模型结构



CNN和Transfermer在捕捉上下文依赖上的差异

1. **序列长度限制**：CNN由于其局部感受野的设计，最多只能捕捉到有限的上下文信息，这个长度通常取决于卷积核的大小和层数。虽然有一些技巧（如扩大卷积核的大小或者使用Dilated Convolution）可以提升CNN对更长序列的感知能力，但它依然有其固有的局限性。然而，Transformer模型能够处理任意长度的序列，并且在任意两个序列位置之间都能建立直接的依赖关系。

2. **并行计算**：CNN由于其卷积操作的特性，可以很好地进行并行化处理，这使得它在处理长序列时相比RNN等模型有很大的速度优势。但是Transformer模型由于其全局自注意力（self-attention）机制，计算复杂度和存储复杂度与输入序列的长度平方成正比，使得处理极长的序列时可能面临计算和存储压力。
3. **上下文依赖建模方式**：CNN通过连续的卷积层来建立更长的上下文依赖，也就是说，高层的卷积核可以看到底层卷积核的输出，从而间接地看到更长的输入序列范围。然而这种方式建立的依赖性是间接的，可能无法很好地处理一些复杂的长距离依赖问题。而Transformer的自注意力机制，可以使得任意两个序列位置直接建立连接，更好地处理长距离依赖问题。
4. **处理语义角色关系**：对于某些任务（如机器翻译），可能需要理解句子中的词语之间的关系，特别是语义角色关系，比如主谓宾结构等。由于Transformer的全局自注意力机制，能够更好地处理这种语义关系，而CNN可能在这方面的表现较差。