cs224n默认的最终项目:

构建QA系统(健壮的QA跟踪)

最后更新于2021年2月13日

内容

1概述

1.1的动力 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

1.2问题回答 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

1.3本项目 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

2

2

2

3.

2开始

2.1代码概述 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

2.2设置 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

4

4

4

3训练数据集

3.1数据分割 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

3.2术语 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .

4基线

4.1处理与分块.. .. .. ..

4.2基线模型.. .. .. .. .. . ..

4.3基线培训.. .. .. . .. ..

4.4在TensorBoard中跟踪进度…

4.5巡检输出.. .. .. .. . ..

. .

. .

. .

. .

. .

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

5 .适应方法探索

5.1 Mixture-of-Experts . . . . . . . . . . . . . .. . .

5.2任务自适应微调.. .. .. .. .. ..

5.3数据增强的鲁棒性.. .. .. ..

5.4域名对抗性培训.. .. .. . .. ..

5.5几样微调.. .. .. .. .. .. ..

5.6元学习 . . . . . . . . . . . . . . . . . . .

5.7使用上下文学习的少镜头适应…

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

。

6

6

6

。

。

。

。

。

。

。

。

。

。

7

7

7

7

8

8

9。。9

。。9

。。9

。。10

。。10

。。10

。。10

提交到排行榜

6.1概述 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .11

6.2提交步骤 . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .11

7评分标准

12

13

8荣誉代码

1

1概述

1.1的动力

在过去的几年里，我们在基本的自然语言理解问题上看到了巨大的进步。与此同时，越来越多的证据表明，模型学习的表面相关性无法推广到训练分布之外[1,2,3,4]。另一方面，人类可以很容易地在他们的训练分布之外进行概括——虽然不是严格地按照我们的训练分布，但我们可以毫不费力地理解小说设置在虚构的世界，并迅速理解新单词的含义。我们怎样才能建立像人类一样一般化的NLP系统?从实际角度来看，对分布外数据的鲁棒性对于在现实世界中构建准确的NLP系统至关重要，因为训练和测试数据通常来自不同的用户交互。

在这个项目中，您将构建一个可以适应未知领域的问题回答系统，只需要从该领域获取少量的训练样本。这将使学生接触到一个更真实的场景，在这个场景中，测试示例很少与训练数据绑定。要构建这样一个系统，需要提供三个独立的问题回答数据集。除了斯坦福问答数据集(SQuAD)[5]之外，您还会得到自然问题[6]和NewsQA[7]，它们都是以与SQuAD相同的格式预处理的。在测试时，你将从看不见的问题回答数据集得到示例，以及一个包含128个示例的小训练集，用于额外的微调。

今年，默认的最终项目由两个轨道组成。在IID SQuAD track中，您将为SQuAD数据集构建一个QA系统，而在Robust QA track中，您将构建一个对域转移具有健壮性的QA系统。注意，对于IID SQuAD轨道，您不允许使用预先训练的变压器模型，而对于RobustQA轨道，您只允许使用蒸馏器[8]作为预先训练的变压器模型。

1.2问题回答

在阅读理解或问题回答的任务中，模型会给出一个段落和一个关于该段落的问题作为输入。目标是正确地回答问题。从研究的角度来看，这是一项有趣的任务，因为它提供了一种衡量系统如何“理解”文本的方法。从更实际的角度来看，这些系统(图1)对于更好地理解任何文本片段和满足人类的信息需求非常有用。

例如，考虑SQuAD数据集。SQuAD中的段落来自维基百科。这些问题和答案都是通过亚马逊的土耳其机器人(Mechanical Turk)众包出来的。总共有大约150k个问题，大约有一半的问题不能用提供的段落回答(这是小队2.0的新功能)。但是，如果问题是可以回答的，那么答案就是直接从段落中取一段文字。这意味着小队系统不需要生成答案文本——他们只需要在回答问题的段落中选择文本范围(想象你的模型有一个高亮器，需要高亮答案)。下面是一个hquestion的例子，上下文，三层答案。要查看更多示例，您可以在https://rajpurkar.github.io/SQuAD-explorer/explore/v2.0/dev/网站上探索数据集。

问:为什么特斯拉回归了高斯皮克?

1879年3月24日，特斯拉因没有居留证被送回高斯皮克，由警察看守。1879年4月17日，米卢廷·特斯拉死于一种未指明的疾病(尽管有消息称他死于中风)，享年60岁。在那一年里，特斯拉在他的母校高real体育馆(Higher Real Gymnasium)教一个大班的学生。

答:无居留证

事实上，在官方验证和测试集中，每个可回答的小组问题都提供了三个答案——每个答案来自不同的人群工作者。答案并不总是完全一致的，这也是为什么“人类表现”在球队排行榜上不是100%。性能通过两个指标来衡量:精确匹配(EM)分数和F1分数。

精确匹配是系统输出是否匹配的二进制度量(即true/false)

毫不含糊地回答。例如，如果您的系统用

2

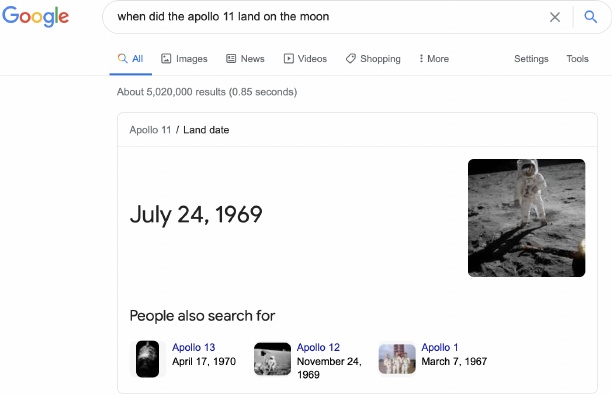


图1:谷歌的问答系统能够回答任意的问题，对于满足信息需求来说是非常有用的工具

"爱因斯坦"但实际答案是"阿尔伯特·爱因斯坦"那么你在这个例子中会得到0分。这是一个相当严格的度量标准!

•F1是一个不那么严格的度量标准——它是精度和召回率的调和平均值1。在“爱因斯坦”的例子中,系统将有100%的精度(它的答案是地面真理回答)的一个子集和50%召回(它只包含一个两个字的地面实况输出),因此一个F1的分数预测2××回忆/(精度+召回)= 2∗∗100 /(100 + 50)= 66.67%。

•在验证或测试集上进行评估时，我们取该问题由人类提供的三个答案的最大F1和EM分数。这使得评估更加宽容——例如，如果一个人类注释员确实回答了“Einstein”，那么你的系统将得到100%的EM和100%的F1。

最后，在整个评估数据集上取EM和F1分数的平均值，以得到最终报告的分数。

1.3本项目

如前所述，您将构建一个在域外数据集上工作良好的问题回答系统。我们已经提供了用于预处理数据和计算评估指标的代码，以及训练全功能神经基线的代码。你的工作就是在这个基础上改进。

在第5节，我们描述了几个模型和技术，通常用于构建少量射击系统-大多数来自最近的研究论文。我们提供这些建议来帮助您开始实现更好的模型。请注意，这是一个相当新的领域，因此这些建议可能不会都导致基线之上的改进。

虽然您不需要实现一些原创的东西，但最好的项目将追求一些原创，并在基线之上进行改进。独创性并不一定是一种全新的方法——对现有模型进行小而积极的改变是非常有价值的，特别是在随后进行了良好的分析的情况下。如果你能从数量上和质量上证明你的小而原始的改变改进了最先进的模型(甚至更好，解释它解决了什么特定的问题以及如何解决)，那么你就做得非常好。

就像自定义的最终项目一样，默认的最终项目是开放式的——它将由你自己决定要做什么。在很多情况下，对于如何做一件事并没有一个正确的答案——需要实验来确定哪种方法是最好的。我们希望你们能够运用从课程中获得的判断和直觉来构建自己的模型。

有关评分标准的更多信息，请参阅第7部分。

1在这里阅读更多关于F1的信息:https://en.wikipedia.org/wiki/F1\_score

3.

2开始

对于这个项目，您将需要一台带有gpu的机器来有效地训练您的模型。为此，你可以访问Azure，就像作业4和5一样——记住，你可以参考Azure指南和虚拟机实用指南链接在课程网页上。与前面一样，请记住，Azure credit会按VM开启的每分钟收费，因此，只有在实际训练模型时，VM才会打开，这一点很重要。

我们建议您在本地机器(或Stanford机器之一，如rice)上开发代码，使用不带gpu的PyTorch，并在调试完代码并准备好接受培训后才转移到Azure VM。我们建议你使用GitHub来管理你的代码库，并在两台机器之间(以及团队成员之间)同步它——VMs实用指南有更多关于这方面的信息。

当您第一次完成这个入门部分时，请在您的本地机器上完成。然后在Azure VM上重复该过程。一旦您在合适的机器上，使用以下命令克隆项目Github存储库。



git克隆https://github.com/MurtyShikhar/robustqa.git

这个存储库包含了我们将要使用的初始代码和数据集。我们鼓励您用git克隆我们的存储库，而不是简单地下载它，这样您就可以轻松地将我们对代码所做的任何错误修复集成到代码中。事实上，您应该定期检查是否有需要下载的新补丁。为此，导航到robustqa目录并运行git pull命令。

注意:如果你使用GitHub来管理你的代码，你必须保证你的存储库是私有的。

2.1代码概述

存储库robustqa包含以下文件:

•参数。train.py的命令行参数。

•环境。conda虚拟环境中的包列表。

•火车。培训和执行推理的顶级入口点。

•跑龙套。实用函数和类。

另外，你会注意到两个目录:

• data/: 包含组织在单独文件夹中的数据集(indomain\_train， indomain\_val, oodomain\_train、oodomain\_val oodomain\_test)。在每个文件夹中，每个数据集都有单独的JSON文件。

• save/: 保存所有检查点和日志的位置。例如，如果您使用python train.py——run-name baseline——do-train来训练基线，那么日志、检查点和TensorBoard事件将保存在其中 save/baseline-01. 如果您使用相同的名称训练另一个模型，则后缀号将增加。

2.2设置

当您在合适的机器上并克隆了项目存储库之后，就可以运行安装命令了。

•确保安装了Anaconda或Miniconda。

•cd到robustqa，然后运行conda env create -f environment.yml

—创建一个conda环境，称为robustqa.

•运行source激活robustqa

-这将激活健壮的qa环境。

4

-注意:记住每次编写代码时都要这么做。

•从README中提供的链接下载数据集，并使用tar -xvzf datasets\_50k.tar.gz解压它们

•(可选)如果您想使用PyCharm，请选择robustqa环境。Mac OS X示例说明:

—在“PyCharm”中打开“robustqa”目录。

-进入PyCharm >首选项>项目>项目解释器。

-单击右上角的齿轮，然后添加。

—选择Conda environment > Existing environment >单击“…”“在右边。

-选择/用户/ YOUR\_USERNAME / miniconda3 / env / robustqa / bin / python。

—选择“OK”，然后应用。

一旦你解开了拉链 datasets\_50k.tar.gz, 您现在应该可以看到许多其他文件夹 save/ (详见第3节):

•indomain\_train/{newsqa,squad,nat\_questions}:这些是对应于领域内培训数据的JSON文件。

•indomain\_val/{newsqa,squad,nat\_questions}:这些是对应于域内验证数据的JSON。

•oodomain\_train/{race,relation\_extract,duorc}:用于域外数据集的训练数据。这些可以用于额外的微调。

•oodomain\_val/{race,relation\_extract,duorc}:域外数据集的验证数据。这些参数可用于设置各种超参数。

•oodomain\_test/{race,relation\_extract,duorc}:这些对应于系统将在其上评估的测试集。

如果您看到了所有这些文件，那么您就可以开始训练基线模型了(参见4.3节)!

5

3训练数据集

3.1数据分割

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 数据集 | 问题的来源 | 通过源 | 火车 | dev | 测试 |
|  | 在域数据集 | |  |  |  |
| 新闻组[5]新闻问答[7]自然问题[6] | 众包众包搜索日志 | 维基百科新闻文章 | 50000 50000 50000 | 10507 4212 12836 | - - - |
|  | oo-domain数据集 | |  |  |  |
| DuoRC[9]种族[10]关系提取[11] | 众包老师合成 | 电影评论考试维基百科 | 127 127 127 | 126 128 128 | 1248 419 2693 |

表1:用于构建本项目QA系统的数据集的统计数据。问题来源和文章来源是指问题和文章的数据来源。从[12]借用的表

在这个项目中，你被提供了三个领域内的阅读理解数据集(自然问题，新闻问答和SQuAD)来训练一个QA系统，该系统将根据三个不同的领域外数据集(RelationExtraction [11]， DuoRC [9]， RACE[10])的测试示例进行评估。对于所有提供的三个训练数据集,我们有一组indomain\_train以及一套indomain\_val。除此之外,我们还提供了一组oodomain\_train组成的少数例子的范围之外额外的训练数据集,以及一组验证(oodomain\_val)。总的来说，我们将所有indomain\_train和oodomain\_train集合的并集简单地称为火车集，而将indomain\_val和oodomain\_val集合的并集称为验证集。

您的QA系统将在eval数据集的一个持有测试集(oodomain\_test)上进行评估。为了简单和可扩展性，我们将这条赛道设置为“kaggle风格”的排行榜。，我们将每个eval数据集的测试集(上下文、问题)发布给学生，他们将模型生成的答案提交到CSV文件中。然后，我们将这些CSV文件与真实测试集的答案进行比较，并在排行榜中报告分数。您只能使用我们的培训和验证集来培训、调整和评估您的模型。数据集统计见表1。如果你想使用任何额外的数据，请在Ed上创建一个私人帖子，并与助教核对

您将使用训练集来训练模型，使用验证集来调优超参数并在本地度量进度。最后，你将提交你的测试集解决方案到一个类排行榜，它将计算和显示你的成绩在测试集上-参见第6部分的更多信息。

3.2术语

训练集包含许多(上下文、问题、答案)三元组2 —请参见1.2节中的举例。每个上下文(有时在其他论文中称为一段、一段或一份文件)都是从维基百科摘录的。这个问题(有时在其他论文中称为查询)是要根据上下文回答的问题。答案是来自上下文的一段(即文本的摘录)。

正如第1.2节中所描述的，开发和测试集实际上对每个问题都有三个人为提供的答案。但是训练集每个问题只有一个答案。

6

4基线

作为起点，我们已经为您提供了一个基线模型的完整代码，该模型对一个基于BERT的预先训练过的变压器进行微调。在本节中，我们将描述基线模型并向您展示如何训练它。

4.1处理与分块

我们处理所有数据集的格式与SQuAD相同。在该目录中，我们有与训练数据对应的json。

分块:因为BERT可以编码的最大上下文大小是512，所以我们将每个(问题、段落)转换为多个大小为384、跨度为128的块。要理解这个过程，请考虑下面的示例。设(q, p)是一个问题，其中q = {q0,问1q,……10}和p = {p0p1,……p500}。我们把它转换成块c1 和c2 其中c1 = (CLS) q [9] p1与p [9]1 = {p0p1……p371}和c2 = (CLS) q [9] p2与p [9]2 = {p128p129……p499}。然后根据偏移量标记每个块的开始位置和结束位置。对于不包含答案范围的块，我们将开始和结束位置设置为

(0,0)

缓存:由于标记化和预处理步骤对于非常大的数据集来说既耗时又缓慢，所以一旦数据被处理，我们就缓存数据的标记化表示。如果您想从头创建这些，请确保运行 train.py 的国旗 create-features. 一个容易被忽略的bug是更改预处理函数和无法重新计算缓存的特性。

4.2基线模型

基线系统在所有训练数据上使用蒸馏器(原始BERT模型[8]的一个更小的蒸馏版本)。

损失函数

我们的损失函数是负对数似然(交叉熵)损失的开始和结束位置的总和。也就是说，如果金的起始位置和结束位置i∈{1，…， N}和j∈{1，…， N}，则单个例子的损失为:

FORMULA

在培训期间，我们对批处理进行平均，并使用AdamW优化器[13]来最小化损失。

推理的细节

给定一个问题段对(q,p)，我们首先将其转换成多个块(c1c2……ck遵循我们的分块程序。然后，我们通过ci 通过我们的模型得到相应的起始logit和结束logit。然后，我们选择和最大的对数。具体地说，我们选择使p最大化的指标对(i, j)start(我)·pend(j)符合i≤j和j−i+1≤L的条件max,我max 是一个超参数，它设置预期答案的最大长度。我们将Lmax 默认为15。

4.3基线训练

在开始在VM上训练基线之前，考虑使用tmux或其他会话管理器打开一个新的会话。这将使您更容易在很长一段时间内离开模型训练，然后在稍后检索会话。有关TMUX的更多信息，请参见最终项目的实用技巧文档。

要开始训练基线，运行以下命令:

7

python training .py——do-train——run-name baseline #开始训练

在一些初始化之后，你应该看到模型开始像下面这样记录信息:

1 \%|### | 45344/242304 [30:59<1:42:14, 32.23it/， NLL=1.45, epoch=1]

您应该看到损失-显示为 NLL 对于负对数似然-开始下降。在单个Azure NC6实例中，每个epoch的培训时间应该是3-4小时。

您还应该看到，下面有一个新目录 save/baseline-01. 你可以在这里找到与这个实验有关的所有资料。特别是，你将(最终)看到:

• log\_train.txt: 培训期间所有信息的记录。这包括最顶部的参数的完整打印，这在尝试重现结果时可能会很有用。

• events.out.tfevents.\*: 这些文件包含信息(比如随着时间的推移而丢失的信息)，我们的代码已经记录了这些信息，因此可以通过TensorBoard进行可视化。

•检查点/:培训过程中最好的检查点。用于确定哪个检查点是“F1”分数的指标。通常，您将加载此检查点以供以后使用。

4.4张力板进度跟踪

我们强烈建议您使用TensorBoard，因为它将使您更好地了解您的实验。使用TensorBoard，在squad目录下运行以下命令:

tensorboard—logdir保存—端口5678

#开始TensorBoard

如果您是在本地机器上进行培训，现在在浏览器中打开http://localhost:5678/。如果你是在远程机器上训练(例如Azure)，那么在你的本地机器上运行以下命令:



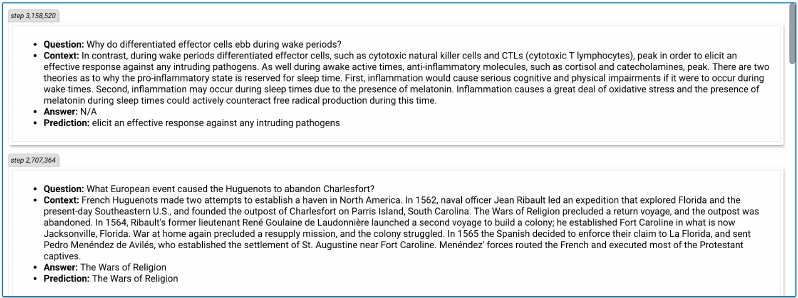
ssh -N -f -L localhost:1234:localhost:5678 <user>@<remote> . ssh -N -f -L localhost:1234:localhost:5678 <user>@<remote> . ssh

其中<user>@<remote>是你ssh到的远程机器的地址。然后在本地机器上，在浏览器中打开http://localhost:1234/。

你应该在验证集上看到带有损失、EM和F1的张力板加载图。EM和F1是官方的球队评估指标。验证图可能需要一段时间才能出现，因为它们被记录的频率比火车图要低。然而，您应该看到训练集损失从一开始就在减少。

4.5检查输出

在培训期间，您还将注意到TensorBoard标签文本中的标签。尝试点击这个选项卡，你应该会看到类似下面的输出:



查看这些示例对于调试模型、理解其优缺点以及作为最终报告中分析的起点非常有帮助。

8

5 .适应方法探索

在本节中，我们将为您提供用于提高域外性能以及更好的少拍适应性的技术概述。您的工作是阅读其中一些技术，理解它们，选择一些实现，仔细培训它们，并分析它们的性能——最终构建出您所能做到的最好的系统。实现是一个开放式的任务:一个模型有多个有效的实现，有时一篇论文甚至不能提供所有的细节——需要你自己做出一些决定。要了解更多关于项目期望和评分的内容，请参见第7节。

5.1 Mixture-of-Experts

出现在:本地专家的自适应混合[14]

训练有效的多任务学习者的一种方法是通过专家混合技术(MoE)。在这里，我们训练k模型(称为专家)和门控函数(控制混合)。给定输入x的标签的条件分布如下所示

FORMULA

fi(x)为第i位专家的条件分布输出，如最后一层logits的softmax, gi(x)为选通函数产生的该专家的混合权重。门控函数本身可以通过神经网络参数化。将MoE应用于鲁棒问题回答的一个具体方法是为每个数据集建立一个单独的蒸馏器模型，并使用一个小的MLP作为门控函数。通过仔细控制哪些示例更新专家，专家可以学习专门的行为，从而形成更有效的多任务模型，从而更好地进行推断。

5.2任务自适应微调

出现在:Don 't Stop Pretraining: Adapt Language Models to Domains and Tasks [15]

另一种将基于BERT的问题回答系统适应于新领域的方法是，联合优化基于QA的损失和原始的屏蔽语言建模(MLM)在输入上的损失。为了创建用于直销培训的实例，给定输入x = (q, p)中的令牌可以随机转换为[MASK]令牌。这种技术被称为任务自适应微调(或TAPT)，已被证明可以有效地使预先训练好的模型适应新领域的任务。

5.3通过数据增强的鲁棒性

出现在:关于回答[16]的领域不可知问题的数据增强和采样技术的探索，用于调试NLP模型的语义等价对抗规则

[17]

许多研究表明，最先进的神经模型学习脆弱的关联，损害了它们的域外性能。防止模型学习这种脆性关联的一种方法是通过数据增强对保留标签不变性进行编码。例如，给定一个带有标签y的输入x = (q, p)，可以创建一个扩展示例x0 =(问0p0),问0 和p0 分别是q和p的释义。这样的释义可以通过反译或替换产生。去问0 通过反译，q首先被翻译成一种“支点”语言(比如俄语)，然后再被翻译回原始语言以产生释义版本。在基于单词替换的数据扩展中，输入中的单个单词被替换为来自词典[18]的同义词，或者被替换为[MASK]标记，然后用BERT[19]填充，以获得扩展输入。此外，[17]考虑了几种保留意义的扰动，这些扰动可以有效地用于增强训练数据，从而提高鲁棒性。

9

5.4领域对抗性训练

出现在:跨域普遍依赖解析的对抗性训练[20]中

领域对抗训练的目标是学习领域不变特征，不编码伪的，领域依赖信息彻底的对抗学习框架。具体来说，给定定义域d的输入xi，模型f(x)输出的特征被馈入一个鉴别器g，该鉴别器g试图对x的域进行分类。g被训练为最大限度地预测正确域的概率，模型f需要一个信号来最大限度地混淆鉴别器。[21]通过训练g将此应用于基于BERT的QA设置，以识别BERT在[CLS]令牌处的表示输出。

5.5少量样品微调

出现在:重访少样本伯特微调[22]

近年来的许多著作[22]探讨了学习速率、梯度更新步骤数、冻结层数等重要超参数对少拍精度的影响。由于外域训练数据的规模较小，这些可以极大地改善不稳定性，我们鼓励学生研究和分析这些选择对少射表现的影响。

5.6元学习

出现在:研究低资源自然语言理解任务[23]的元学习算法，学习在不同的自然语言分类任务[24]中少量学习

建立少数镜头模型的一个主要技术是元学习，其目标是建立一个参数化的元学习者Mθ，它从任务(称为支持集)中获取少量样本作为输入和输出一个适应的模型，该模型可以对任务(查询集)中的新示例做出准确预测。元学习由训练数据(支持、查询)对从一组训练任务和目标是最大化准确性上查询后适应支持集合。近期作品(23、24)已经显示成功经验与元学习fewshot适应NLP的上下文中。

5.7采用上下文学习的少镜头适应

出现在:让预先训练好的语言模型更好的少量学习者[25]

GPT-3[26]令人印象深刻的少量能力是通过上下文学习实现的。为了从一个看不见的任务中对输入x进行预测，语言模型接收额外的上下文c，该上下文c包含少量的输入、输出对，并附加一个提示。[25]表明，上下文学习甚至更广泛地适用于传销商，并在一个小的训练数据集下自动发现有用的上下文。

10

6 .提交排行榜

6.1概述

我们在Gradescope上举办了两个排行榜，在那里你可以和你的同学比较你的表现。F1得分是我们将用来对提交作品进行排名的性能指标，尽管EM和F1得分都将显示出来。排行榜可于以下连结查阅:

1.oodomain-val待办事项

2.oodomain-test:待办事项

你可以向开发排行榜提交任何次数，但你只允许3次成功提交到测试排行榜。对于你的最终报告，我们将要求你选择一个单一的测试排行榜提交，以考虑你的最终表现。因此，你必须至少提交一次测试积分排行榜，但要注意，在你完成开发最佳模型之前，不要用完你提交的测试。

提交到排行榜Gradescope类似于提交任何其他任务,除了你的提交是一个CSV文件在dev /测试集的答案。你可以用起动器代码的test.py脚本生成一个提交文件的正确格式,或看到128 - 135例如代码来生成一个提交文件。在高层次上，提交文件应该如下所示:

Id、预测

001fefa37a13cdd53fd82f617，总督Vaudreuil 00415cf9abb539fbb7989beba, 1754年5月

00 a4cc38bd041e9a4c4e545ff,

…

fffcaebf1e674a54ecb3c39df 1755

标题是必需的，并且后续的每一行必须包含两列:第一列是question/answer示例的25位十六进制ID(在{dev,test}-v2.0.json中定义的ID)，第二列是预期的答案。行可以是任何顺序。

6.2提交步骤

以下是提交排行榜的具体步骤:

1.生成一个提交文件(例如，通过运行 train.py 在初始代码中)用于验证或测试集:

•要在验证集排行榜上生成提交文件，运行:

python train.py——do-eval——eval-dir数据集/oodomain\_val

——子文件val\_submit .csv——save-dir save-dir

•要在测试集排行榜上生成提交文件，运行以下命令:

python train.py——do-eval——eval-dir数据集/oodomain\_test .py

——subfile test\_submit .csv——save-dir save-dir .csv

其中SAVE-DIR是创建检查点的文件夹。

2.使用上面的url导航到排行榜。确保为你的分离(验证vs.测试)选择正确的排行榜。

3.在Gradescope中找到submit按钮，并选择要上传的CSV文件。

4.点击上传并等待你的分数。提交输出将告诉您提交EM/F1，尽管排行榜将保留迄今为止F1得分最高的提交分数。

如果出现任何错误，应该会有有用的错误消息。如果你遇到一个你不能理解的错误，请在Ed上发帖。

11

7评分标准

期末专题将以整体方式评分。这意味着我们在决定你的分数时会考虑很多因素:你的方法的创造性、复杂性和技术上的正确性，你在探索和比较各种方法时的彻彻性，你的结果的强度，你所付出的努力，你的写的质量，评价和错误分析。一般来说，实现更复杂的技术代表更多的努力，实现更多不寻常的想法(例如，我们在本讲义中没有提到的)代表更多的创造力。你不需要追求原创的想法，但是这门课上最好的项目将会超越这个讲义中描述的想法，并且实际上可能会成为发表的作品!

没有预先定义的F1或EM分数来确保良好的成绩。虽然我们已经进行了一些初步测试，以获得一些大致的分数，但我们不可能事先说在规定的时间内，学生可以合理地实现分数的分配。和往年一样，我们将根据整体排行榜对学生的表现进行评分。

出于类似的原因，在第5节(或其他地方)中没有预先定义的规则来确定哪一种模式可以保证一个好成绩。执行少量结果良好的内容并进行彻底的实验/分析总比执行大量无效或几乎无效的内容要好。此外，你的写作和实验的质量也很重要:我们希望你能令人信服地证明你的技术是有效的，并描述为什么它们有效(或当它们不起作用时的情况)。

在报告的分析部分，我们希望看到您超越模型的简单F1和EM结果。分析你的系统的一种方法是分解分数。例如，你能分析为什么你的模型在某些领域比其他的更好吗?什么类型的例子是模型总是出错的?为什么?域内开发集的准确性如何与ooo域开发集的准确性相关联?

与所有的最终项目一样，更大的团队需要完成相应的更大的项目。我们期待更多复杂的东西被实现，更彻底的实验，以及更多人的团队带来更好的结果。

12

荣誉准则

8

任何适用于最终项目的荣誉准则通常也适用于默认的最终项目。以下是一些与默认最终项目的QA跟踪相关的指导方针:

1.你不能在你的系统中使用除蒸馏器以外的其他训练过的模型。然而，如果你想使用较小的预先训练过的模型，并且相信你有很好的理由这样做，请发一个Ed帖子获得许可。

2.除非您自己编写实现，否则您也不能使用各种方法的现有实现(包括第5节中提到的那些)作为起点，以提高鲁棒性/少镜头适应性。如果你认为你有很好的理由这样做，请发一个Ed帖子。

3.如3.1节所述，使用提供的培训文件以外的任何其他数据进行培训(或开发)都是违反荣誉准则的行为。

4.您可以自由地与其他团队讨论想法和实现细节(事实上，我们鼓励这样做!)但是，在任何情况下，您都不能查看其他CS224n团队的代码，或者将他们的代码合并到您的项目中。

5.不要公开分享你的代码(例如，在GitHub的公开repo中)，直到类完成。

13

参考文献

Robin Jia和Percy Liang。评价阅读理解系统的对抗性例子。arXiv预印本arXiv: 1707.07328,2017。

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman，和Noah A Smith。自然语言推理数据中的注释工件。计算语言学协会(ACL)，第107-112页，2018。

Thomas McCoy, Ellie Pavlick, Tal Linzen。从对到错:诊断自然语言推理中的句法启发式。2019年，在计算语言学协会(ACL)。

Marco Tulio Ribeiro, Tongshuang Wu, Carlos Guestrin和Sameer Singh。超越准确性:使用检查表对NLP模型进行行为测试。第58届计算语言学协会年会论文集，第4902-4912页，在线，2020年7月。计算语言学协会。

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, Percy Liang。小组:10万多个机器理解文本的问题。2016年,abs / 1606.05250。

Tom Kwiatkowski, jenimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, mingwei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, Slav Petrov。自然问题:问答研究的基准。2019年，在计算语言学协会(ACL)。

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, Kaheer Suleman。Newsqa:一个机器理解数据集。ACL 2017，第191页，2017。

Victor Sanh, Lysandre处子秀，Julien Chaumond和Thomas Wolf。蒸馏器，蒸馏版伯特:更小，更快，更便宜，更轻。arXiv预印本arXiv:1910.01108, 2019。

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, Karthik Sankaranarayanan。DuoRC:用意译阅读理解来理解复杂的语言。2018年ACL。

赖国坤，谢启哲，刘汉晓，杨一鸣，Eduard Hovy。种族:来自考试的大规模阅读理解数据集。2017年EMNLP。

[11] Omer Levy, Minjoon Seo, Eunsol Choi，和Luke Zettlemoyer。通过阅读理解提取零镜头关系。arXiv预印本arXiv:1706.04115, 2017。

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen。MRQA 2019共享任务:评估阅读理解中的泛化。2019年，在机器阅读问答(MRQA)研讨会上。

Ilya Loshchilov和Frank Hutter。解耦权值衰减正则化。arXiv预印本arXiv:1711.05101, 2017。

R.雅各布斯，迈克尔I.乔丹，S.诺兰，和杰弗里E.辛顿。当地专家的适应性混合。神经计算，3:79-87,1991。

[15] Suchin Gururangan, Ana Marasović， Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey和Noah A Smith。不要停止预先训练:使语言模型适应领域和任务。arXiv预印本arXiv:2004.10964, 2020。

Shayne Longpre, Yi Lu, Zhucheng Tu, Chris DuBois。关于领域未知问题回答的数据增强和抽样技术的探索。arXiv预印本arXiv:1912.02145, 2019。

14

Marco Tulio Ribeiro, Sameer Singh和Carlos Guestrin。用于调试NLP模型的语义等价对抗规则。《第56届计算语言学协会年会论文集》(第1卷:长篇论文)，856-865页，墨尔本，澳大利亚，2018年7月。计算语言学协会。

韦杰森和邹凯。Eda:提高文本分类任务性能的简单数据增强技术。arXiv预印本arXiv:1901.11196, 2019。

Siddhant Garg和Goutham Ramakrishnan。基于bert的对抗文本分类示例。2020年EMNLP。

[20]佐藤元木，Manabe仁，野二浩，松本友二。跨领域普遍依赖分析的对抗性训练。《CoNLL 2017 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies》，第71-79页，加拿大温哥华，2017年8月。计算语言学协会。

李世妮，金东宇，朴江原。带有对抗性训练的领域未知问题回答。2019年MRQA@EMNLP。

张天一，Felix Wu, Arzoo Katiyar, Kilian Q Weinberger, Yoav Artzi。再访少样本伯特微调。arXiv预印本arXiv:2006.05987, 2020。

窦子怡，余可怡，安东尼奥斯·阿纳斯塔索普洛斯。研究低资源自然语言理解任务的元学习算法。第九届国际自然语言处理联合会议论文集(EMNLP-IJCNLP)，第1192-1197页，中国香港，2019年11月。计算语言学协会。

Trapit Bansal, Rishikesh Jha和Andrew McCallum[24]。学习通过多种自然语言分类任务。arXiv预印本arXiv:1911.03863, 2019。

高天宇，Adam Fisch，陈丹琪。使预先训练好的语言模型更好的少数学习者。arXiv预印本arXiv: 2012.15723,2020。

[26] Tom B Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhari- wal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell等。语言模型是很少的学习者。arXiv预印本arXiv: 2005.14165,2020。

15