某课程视频链接youtube

笔记本:	深度学习		
创建时间:	2018/3/23 10:05	**更新时间:**	2020/5/28 21:52
作者:	beyourselfwb@163.com		
标签:	深度学习, 视频链接		
URL:	https://classroom.udacity.com/nanodegrees/nd101/parts/83dbe0a0-2265-4b03-b693-397a69d49eca/...		

一个简单的自编码器
https://www.youtube.com/watch?v=P1XkcQI4JBY
卷积自编码器介绍
https://www.youtube.com/watch?v=AaE-0TwR45Q
卷积自编码器解决方案
https://www.youtube.com/watch?v=avbNvKQ2qLE

迁移学习
迁移学习简介:
2. 用 VGGNet 进行迁移学习
https://www.youtube.com/watch?v=WmQwYr0DYjY
3.VGGNet 练习
https://www.youtube.com/watch?v=615SslQiGvo
4.VGGNet 答案
https://www.youtube.com/watch?v=BpzI6Svmuv8
5.数据预处理练习
https://www.youtube.com/watch?v=WfsDMq-b3y4
6.数据预处理答案
https://www.youtube.com/watch?time_continue=5&v=WEtKkHlhhZA
7.建立分类器练习
https://www.youtube.com/watch?v=pPHiVddBY0Q
8.建立分类器答案
https://www.youtube.com/watch?v=6ifxRQ_gL7w
9.训练模型练习
https://www.youtube.com/watch?v=b7Fy3cIoJ1Y
10.训练模型答案
https://www.youtube.com/watch?time_continue=4&v=NLPtmQjGYCA

循环神经网络RNN
1.RNN入门
https://www.youtube.com/watch?v=64HSG6HAfEI
2. LSTMs
https://www.youtube.com/watch?v=RYbSHogZetc
3.字符RNN
https://www.youtube.com/watch?v=dXl3eWCGLdU
4.序列分批
https://www.youtube.com/watch?v=Z4OiyU0Cldg
8.LSTM cell
https://www.youtube.com/watch?v=ajC-5uWB8S4
10.RNN输出
https://www.youtube.com/watch?v=RkanDkcrTxs
11.网络损失
https://www.youtube.com/watch?v=itu-uNK4brc
13.搭建网络
https://www.youtube.com/watch?v=RVNjDlWTBQU
15.RNN资源

- 来自 CS231n 的关于 RNN 和 LSTM 的 Andrej Karpathy **的讲座**。
- Christopher Olah 发表的一篇关于 LSTM 原理的**精彩博文**。
- 全新**构建一个** RNN，这个难度级别有点高，但 TensorFlow 中已有实现。

序列到序列
1.简介
https://www.youtube.com/watch?v=HPOzAlXhuxQ
3.应用
https://www.youtube.com/watch?v=tDJBDwriJYQ
4.架构
https://www.youtube.com/watch?v=dkHdEAJnV_w
5.更深的架构
https://www.youtube.com/watch?v=rdAo4MqLbEk

6.预处理
https://www.youtube.com/watch?v=ktQW6p9pOS4
7.TensorFlow中的序列到序列模型

Tensorflow **中的序列到序列模型**

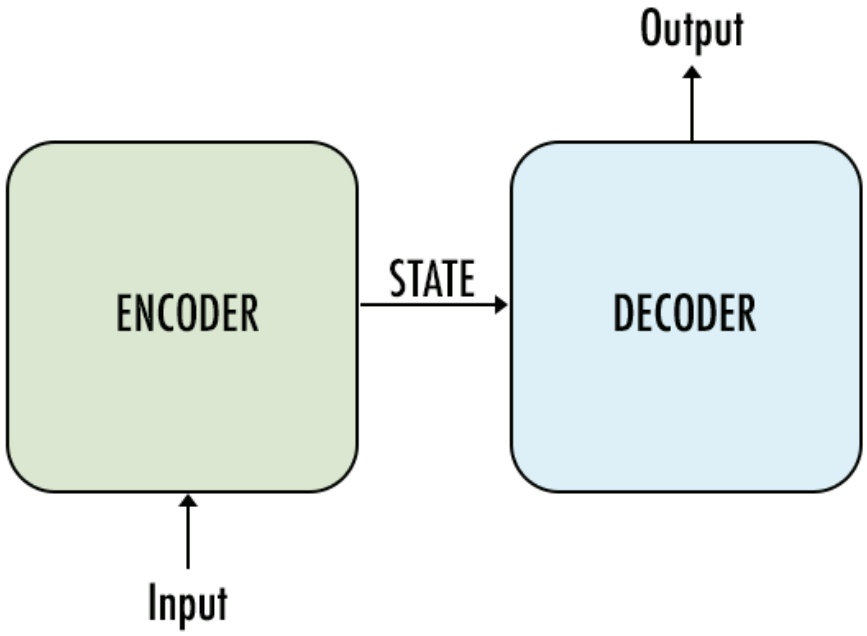
Tensorflow 有一堆 API 可帮助你构建序列到序列模型（简称为 seq2seq 模型）。要注意的一点是，这些 API 在 2016 年底发生了变化（在某种程度上还在演变中）。你将在网站上找到的 tensorflow 中的 seq2seq 模型教程都使用现已弃用的 tf.contrib.legacy_seq2seq（以前称为“tf.nn.seq2seq”）。

seq2seq 的重要模块包括：

- tf.nn，这允许我们构建不同种类的 RNN
- tf.contrib.rnn，这定义多个 RNN cell（RNN cell 是在 tf.nn 中为 RNN 定义的必需参数）。
- tf.contrib.seq2seq，这包含 seq2seq 解码器和损失操作。

主要成分

概括来讲，seq2seq 模型具有以下主要成分：



- **编码器：这是一个 tf.nn.dynamic_rnn 函数。**
- **解码器：这是一个 tf.contrib.seq2seq.dynamic_rnn_decoder 函数。**

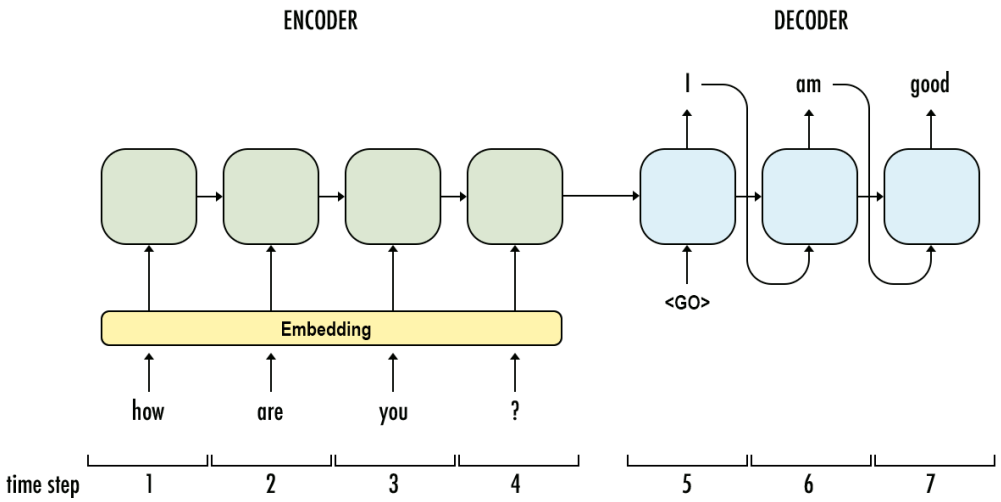
seq2seq **模块**

- **阅读 https://www.tensorflow.org/api_docs/python/tf/contrib/seq2seq 以理解其主要成分。你可以暂时忽略带有"注意"（attention）的所有内容。**

8.输入

输入

当想让模型进行推理时，我们需要向它提供输入序列。假设我们在构建聊天机器人，我们的序列将是文字。我们首先需要将文字转换成网络可以用于计算的适当数字表示。此转换使用 tf.nn.embedding_lookup 完成，我们可以用它（在对数据进行一些处理后）将文字变成向量。



训练输入

如果我们向解码器提供目标序列，而不管运行的训练中的实际输出的时间步长如何，这些模型的效果会更好一些。所以与推理图不同，我们不会在下一个时间步长中将解码器的输出提供给它自己。在使用样本训练模型之前，我们需要对数据进行预处理。

示例： 假设我们的数据集中只有两个示例（来自《黑客帝国》的对白示例）：

source	target
Can you fly that thing?	Not yet
Is Morpheus alive?	Is Morpheus still alive, Tank?

在预处理阶段，假定我们的机器人不需要知道人名。那么在标记化后，将所有单词变为小写字母，并将人名替换为 <UNK>，我们的数据集看起来就像这样：

Source	Target
can you fly that thing ?	not yet
is <UNK> alive?	is <UNK> still alive , <UNK> ?

我们的批量输入就好了：

can	you	fly	that	thing	?
is	<UNK>	alive	?	<PAD>	<PAD>

因为我们在使用嵌入，我们首先必须编译一个“词汇”列表，包含我们想要模型使用或读取的所有单词。模型输入必须是包含序列中单词 ID 的张量。

示例：
假设我们要使用下面这个小的数据集训练模型：

source	target
How are you?	I am good

在开始训练模型之前，我们首先要标记化（tokenize）数据集，消除大小写，然后构建包含所有这些唯一标记的词汇表。在我们的示例中，这个词汇表看起来将是这样的：

id	word
0	how
1	are
2	you
3	?
4	i
5	am
6	good

但是，我们要让词汇表包含的符号只有四个。Seq2seq 词汇表通常会为以下元素保留前四个位置：

- <PAD>: 在训练中，我们需要按批次将我们的示例提供给网络。这些批次中的输入必须具有相同的宽度，网络才能进行计算。但我们的示例的长度不同。因此我们要填充较短的输入，使它们与批次的宽度相同。
- <EOS>: 这是批处理的另一个必要元素，但更多的是用在解码器端。它使我们能告诉解码器句子在哪里结束，并且它允许解码器在其输出中表明句子结束的位置。
- <UNK>: 如果你在使用真实数据训练模型，你会发现通过忽视词汇表中出现频率不够高而不足以考虑在内的文字，会大大提高模型效率。我们将这些单词替换为 <UNK>。
- <GO>: 这是解码器的第一个时间步长的输入，以使解码器知道何时开始产生输出。

注意：我们也可以使用其他标记来表示这些功能。例如，我看到过使用 <s> 和 </s> 替换 <GO> 和 <EOS>。确保你在预处理中使用的标记一致，并对训练/推理建模。

现在，我们来将它们添加到词汇表的顶部：

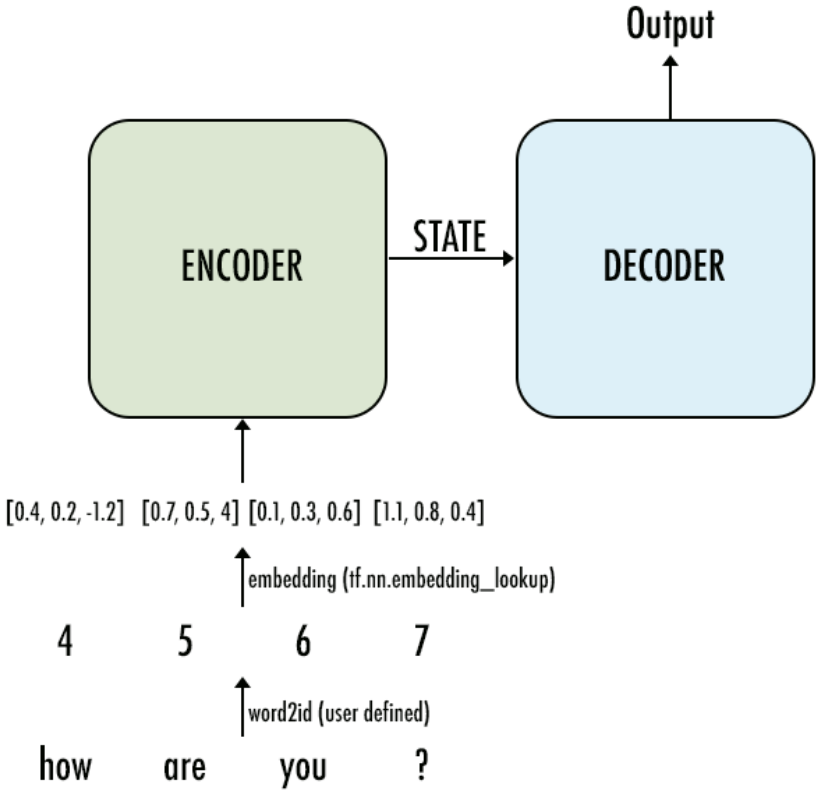
id	word
0	<PAD>
1	<EOS>
2	<UNK>
3	<GO>
4	how
5	are
6	you
7	?

8	i
9	am
10	good

注意我们已经建立了词汇表，我们只是将单词替换为了 ID，它们将作为编码器的输入张量，那么 "how are you ?" 变为：

4	5	6	7

这是处理推理输入的一种方式（我们设置嵌入大小为 3，因此每个单词将用大小为 3 的向量表示）：



训练输入

为训练图准备输入稍微有点复杂，原因有二：

- 如果我们在向解码器提供目标序列时，不管训练中的输出实际运行的时间步长如何，模型的效果会更好一些。所以与推理图不同，我们不会在下一个时间步长中将解码器的输出提供给其自身。
- 批处理（Batching）

示例： 假设我们的数据集中仅有两个示例（《黑客帝国》中的对白示例）：

source	target
Can you fly that thing?	Not yet
Is Morpheus alive?	Is Morpheus still alive, Tank?

在预处理阶段，假定我们的机器人不需要知道人名。那么在标记化后，将所有单词变为小写字母，并将人名替换为 <UNK>，我们的数据集看起来就像这样：

Source	Target
can you fly that thing ?	not yet
is <UNK> alive?	is <UNK> still alive , <UNK> ?

我们的批输入就好了

can	you	fly	that	thing	?
is	<UNK>	alive	?	<PAD>	<PAD>

我手动创建了一个词汇表。但是使用它，正确的输入张量现在已准备就绪：

4	5	6	7	8	9
10	2	11	9	0	0

一篇早期的序列到序列论文（Sutskever et al. 2014）指出，如果颠倒输入的顺序，模型的性能会更高。所以你也可以选择颠倒输入序列中单词的顺序。

现在来看看我们的目标输入张量：

not	yet					
is	<UNK>	still	alive	,	<UNK>	?

现在我们需要：

1. 在开头添加 <GO>
2. 在结尾添加 <EOS>
3. 添加填充

之后，它看起来会是这个样子的：

<GO>	not	yet	<EOS>	<PAD>	<PAD>	<PAD>	<PAD
<GO>	is	<UNK>	still	alive	,	<UNK>	?

然后是我们的目标输入张量：

3	14	15	1	0	0	0	0	0
3	10	2	12	11	13	2	9	1

注意: 我在这里展示这个处理步骤是为了解释张量的形状和值。在实践中，我们会在这个过程更早的时候停止使用单词。在预处理期间，我们执行以下操作：

- 我们构建唯一单词词汇表（并同时计算单词出现的频率）
- 将频率较低的单词替换为 <UNK>
- 使用替换为 ID 的单词创建一个对白副本
- 我们现在可以选择对目标数据集使用 <GO> 和 <EOS> 单词 ID，或在训练时这么做

9.延伸阅读

更多资源

- tensorflow-seq2seq-tutorials 具有当前 TensorFlow seq2seq API 的工作版本
- **康奈尔电影-对白语料库** 是一个从电影脚本中提取的对白的数据集
- tf-stanford-tutorials 具有一个预处理康奈尔语料库的脚本
- **聊天机器人深度学习**
- **序列到序列深度学习（Quoc Le、Google）** - 使用很好的实例进行深入对话。

生成对抗网络GAN
2.你可以用GAN做些什么?

https://www.youtube.com/watch?v=bo-ToTdhgew
3.GAN的运作原理
https://www.youtube.com/watch?v=3yWWKUpY8OM
4.博弈论和均衡
https://www.youtube.com/watch?v=W9Dx0fXwl9g
5.训练GAN的实用技巧
https://www.youtube.com/watch?v=vgZH51xRHCQ
6.构建GAN
7.GAN入门
https://www.youtube.com/watch?v=QA2ntKUha4g
8.生成器网络
https://www.youtube.com/watch?v=btHVXnICmzQ
9.辨别器网络
https://www.youtube.com/watch?v=nWXxT8OqCfs
10.生成器和辨别器解决方案
https://www.youtube.com/watch?v=9By2pAck044
11.构建网络
https://www.youtube.com/watch?v=5sZkRSHfiAE
12.构建网络解决方案
https://www.youtube.com/watch?v=Ikp3rVzG970
13.训练损失
https://www.youtube.com/watch?v=IaAeDrXMEcU
14.训练优化器
https://www.youtube.com/watch?v=AU5gH7LS57E

深度卷积对抗生成网络
1.深度卷积GAN
2.DCGAN架构
https://www.youtube.com/watch?v=sX_AxtB6CHI
3.批归一化
我们准备了一些 notebook 供你参考学习批归一化和如何在 TensorFlow 中实现它。跟往常一样，你可以在我们的 GitHub **代码库** 中找到 notebook，它们位于 `tutorials/batch-norm` 文件夹下。如果你已克隆了代码库，可执行 `git pull` 获取新文件。如果没有，请先克隆代码库：

```
git clonehttps://github.com/udacity/deep-learning.git
```

4.DCGAN实现

我们准备了一些 notebook 供你参考，帮你使用**街景门牌号** (SVHN) 数据集来实现 DCGAN。我制作了一个训练 GAN 并生成图像的短视频。如果你在实现时正确操作，它看起来应该是**这样** 的。

跟往常一样，你可以在我们的 GitHub **代码库** 中找到 notebook，它们位于 `tutorials/dcgan-svhn` 文件夹下。如果你已克隆了代码库，可以执行 `git pull` 来获取新文件。如果没有，请先克隆代码库：

```
git clonehttps://github.com/udacity/cn-deep-learning.git
```

或者，你也可以**在此** 获取 notebook。

5.DCGAN和生成器

https://www.youtube.com/watch?v=CH6BxLTKt7s

6.生成器解决方案

https://www.youtube.com/watch?v=jyPwUEZg05Q

7.辨别器

https://www.youtube.com/watch?v=XRqOUbf96eI

8.辨别器解决方案

https://www.youtube.com/watch?v=ffPWI2yJscw

9.构建和训练网络

https://www.youtube.com/watch?v=nXKk9GI4X14

10.超参数解决方案

https://www.youtube.com/watch?v=Rt8MlVDtpi8

项目：生成人脸

1.项目说明

https://www.youtube.com/watch?v=UUqU8SYBZ9Q

2.项目简介

https://www.youtube.com/watch?v=jvJtHYBX7sM

3.项目提交

人脸生成项目

介绍

在该项目中，你将使用对抗生成神经网络（GAN）生成新的人脸图像。

获取项目文件

项目文件可以在我们的 **公共 GitHub 库** 里的 `face_generation` 文件夹中找到。你可以从那里下载文件，或者更为方便的做法是将 GitHub repo 克隆到你的计算机上。

如果将 GitHub repo 克隆下来，你就可以通过 git pull 命令，将我们的最新改动更新到你的本地库上，随时保持更新。

项目提交

1. 确保你通过了 notebook 中的所有单元测试。
2. 确保你符合**评审标准**中的所有要求。
3. 当你完成项目后，请将 notebook 保存为一个 HTML 文件以及一个 Python 文件。你可以进入 notebook 的菜单，选择“Download as”> HTML (.html) 来保存 HTML 文件；选择“Download as”> Python (.py) 来保存 Python 文件。**确保你同时提交 Jupyter Notebook 以及 HTML 和 Python 这三个版本的文件。**
4. 将 "dlnd_face_generation.ipynb"，"helper.py"，"problem_unittests.py"， HTML 以及 Python 文件打成一个 zip 压缩包，或者将这些文件推送到 GitHub repo 上。
5. 点击下面的“提交项目”按钮！

s
ss
s半监督学习：
s1.半监督学习

半监督学习

在本课中，Ian Goodfellow 将带你在 TensorFlow 中实现一个半监督 GAN 模型。当数据集中只有少量已标记数据时，使用半监督模型也能取得相当好的效果。

我们将再次使用 SVHN 数据集，并尝试在去掉大部分数据标签后对图像进行分类。

首先声明，该内容比较前沿，相较于之前的内容也更难理解和实现。但你会学到更多，并了解一些非常酷的内容。

获取 notebooks 文件

Ian 会利用两个 notebooks 帮你学习，一个里用于练习部署模型，另一个用于查看参考答案。这些 Notebooks 可以在该课程 **公共 GitHub 代码库** 的 `tutorials/semi-supervised` 文件夹下找到。你可以直接下载或通过 `git pull` 更新你已有的代码库。

s2.半监督GAN分类器
https://www.youtube.com/watch?v=_LRpHPxZaX0
s3.半监督学习简介
https://www.youtube.com/watch?v=tnCClNy5z5c
s4.准备数据
https://www.youtube.com/watch?v=P5hOx09mwaM
s5.建立生成器和辨别器
https://www.youtube.com/watch?v=OWytckbbeGQ
s6.模型损失练习
https://www.youtube.com/watch?v=W7TawMNxBds
s7.模型优化练习
https://www.youtube.com/watch?v=wNpI1wUA4Io
s8.训练网络
https://www.youtube.com/watch?v=P-LXQPVXl4A
s9.辨别器解决方案
https://www.youtube.com/watch?v=_X8ssUzu_Bo
s10.模型损失解决方案
https://www.youtube.com/watch?v=r3DtohmychE
s11.模型优化解决方案
https://www.youtube.com/watch?v=_Qhz9SbR7xY
s12.半监督GAN训练结果
https://www.youtube.com/watch?v=9yWYZDX8-O8
s
s

sTensorBoard
s1.TensorBoard简介

大家好！这周，我们将讲解 TensorBoard。TensorBoard 是一个可在你的浏览器中运行的应用，用于检查你的图表和网络变量，如权重和偏差。它有助于你进行调试及选择适合你的问题的最佳超参数。

首先，请观看来自 TensorFlow 发展峰会的此 TensorBoard **教程**。该视频很好地概述了 TensorBoard 的使用。然后**这个教程**可以帮助你快速上手使用 TensorBoard。接下来，我将向你展示如何将 TensorBoard 用于我们在此课程中构建的一个网络。

你可以从我们的**公共 GitHub 代码库** 的 `tensorboard` 文件夹下获得 notebook 文件。你可以在这里下载它，也可以克隆代码库

```
git clonehttps://github.com/udacity/deep-learning.git
```

尝试对你的下一个项目使用 TensorBoard！

s2.查看图
https://www.youtube.com/watch?v=M64FWxf1yK4
s3.名称范围
https://www.youtube.com/watch?v=REmz7HUj6f4
s4.检查变量
https://www.youtube.com/watch?v=QG41p4Wx5wc
s5.选择超变量
https://www.youtube.com/watch?v=THiwPbkjoLQ
s
ss
s
s
ss
s
s
ss

s
s
s
s强化学习
s1.强化学习课程

强化学习

本课简要介绍了强化学习。机器学习的这个分支是关于训练一个代理，当它执行正确的操作时给予奖励。我们可以设立关于强化学习的整个课程，但我们没时间讲解所有不同的方法。那么，我将在这里主要展示一种名为 Q **学习**的特别方法。

阅读材料

以下是供你学习的一些额外资源：

- 关于**强化学习**的博文系列
- **通过深层次的强化学习进行人为控制**
- **在 TensorFlow 中实现深度 Q 网络**
- 关于 **Q 学习和倒立摆游戏**的博文

s2.强化学习
https://www.youtube.com/watch?v=Npu9gyD6-4o
s3.Q学习
https://www.youtube.com/watch?v=WQgdnzzhSLM
s4.深度Q学习

深度 Q 学习

在这个 notebook 中，我将向你展示如何使用具有 Q 学习的神经网络，来训练 AI 智能体玩一个简单的游戏。我们将使用 OpenAI Gym，这是一款非常棒的用于训练 AI 智能的游戏包。我们将训练它玩**倒立摆游戏**。

跟往常一样，你可以从我们的公共资源库获得 notebook，或者克隆代码库

```
git clonehttps://github.com/udacity/deep-learning.git
```

或者如果你已经克隆了代码库，可下载最新的文件

```
git pull
```

notebook 和其他内容均位于 `reinforcement` 目录下。

我在代码库中包含了 OpenAI Gym 源代码，可以使用以下代码安装它以用于 notebook：

```
cd gym
pip install -e .
```

深层 Q 学习背后的基本思想是使用神经网络来替换 Q 表。这个游戏很简单，但是同样的模型也被人们用来训练AI 智能体玩各种 Atari 游戏。然而在实践中，训练 Atari 游戏需要很多个小时。你可以随意在更复杂的游戏上进行尝试。但在此课程中，我们继续使用一个简单的游戏，使它即使在没有 GPU 的情况下也能快速训练。

s
s
s
s
s
s
s
ss
s
s
ss
s
s
ss
s
s
ss
s
s
s
wetgfre
erygre
erytr
sagfvds
wetgfre
erygre
erytr
sagfvds
wetgfre
erygre
erytrsagfvds
wetgfre
erygre
erytr