

# PyTorch 框架班作业（第一期）

笔记整理人：天国之影（2019 年 6 月 29 日）

## 说明

1. 本课程作业的所有代码都要基于 Python3，在 Jupyter Notebook 上完成。
2. Pytorch 中文文档地址：<https://github.com/zergtant/pytorch-handbook>

我的作业 GitHub 地址：<https://github.com/Relph1119/Pytorch-Camp>

- （1）src 文件夹下面为每周的作业内容，用于记录我的作业完成情况，所有 ipynb 文件均带注释。
- （2）PyTorch\_Tutorial 文件夹中为余霆嵩老师的课程代码。

## 1 第 1 周

### 1.1 Pytorch 简介，配置电脑环境

任务：

1. Pytorch 简介
2. 配置电脑环境（pycharm+Anconda+pytorch）
3. 开始学习 Pytorch 官方文档（参考资料：1.自动求导机制；2.CUDA 语义；4.多进程最佳实践）

任务简介：《Pytorch 官方文档》

学习时长：6/2—6/3

详细说明：

本节课所需资料包下载链接：

链接：<https://pan.baidu.com/s/1s6wwJLzRiiJy3xPIuQPDgg>

提取码: 4lv8

- 1、本节第一部分将会向大家介绍 `pytorch`，以及 `pytorch` 作为我们使用深度学习工具的优势。
- 2、第二部分观看资料包中的环境配置文档：如何配置环境（针对小白），基于 `window` 系统、`Anconda`、`Pycharm`、`pytorch`、`(cuda+cudnn)`，环境配置好之后会进行小测试，验证配置的环境是否可以正常使用。
- 3、第三部分正式进入我们的 `pytorch` 学习，工欲善其事必先利其器，我们从最基础 `pytorch` 官方文档 `API` 开始学习，首先是自动求导机制，这部分是在训练模型的时候使用，在进行预测的时候我们只进行前向传播，不进行反向传播，所以也就不需要求导，可以节约预测的时间；第二部分是 `CUDA` 语义，也就是我们如何选择显卡进行计算，同时会涉及到并行计算（这部分在项目部署的时候比较有用）；第三部分也是并行计算多进程的内容。

特征工程是对原始数据进行一系列工程处理，将其提炼为特征，作为输入供算法和模型使用。做过项目或者竞赛的都应当了解特征工程的重要性，面对不在一个数量级的特征，类别性特征，高维特征，特征组合等等场景，我们应该怎么做。

- 4、`pytorch` 文档均在资料包中。

**打卡要求：**在训练和测试时自动求导的区别？如何调用 `CUDA`？程序中如何使用多进程？

**打卡内容：**文字或图片拍照提交，文字要求最少 50 字，图片要求最少 3 张

**打卡截止时间：**6/3

## 1.2 学习序列化模型、`torch` 接口

**任务：**

`Pytorch` 官方文档（参考资料：`Pytorch` 官方文档；5.序列化模型；6.`torch` 接口）

**任务简介：**《`Pytorch` 官方文档》

**学习时长：**6/4

**详细说明：**

本节任务资料包下载：

链接：<https://pan.baidu.com/s/1s6wwJLzRiiJy3xPIuQPDgg>

提取码：[4lv8](#)

本节内容包括如何保存和载入模型，我们一般情况下载训练阶段保存在预测阶段载入，同时需要了解两种方法保存模型的差异。下面是 `pytorch` 最重要的部分，对 `Tensor` 的操作，由于本节内容较多，我们分为七个部分讲解，今天主要是熟悉 `torch` 接口中 1~9 页的 API，能够知道如何使用，知道每个函数的意义和参数的意义。

**作业名称（详解）：**保存模型的两种形式以及他们的区别？手敲今天所学 API 三遍

**作业提交形式：**打卡提交文字或图片，不少于 20 字

**打卡截止时间：**6/5

## 1.3 深入了解 torch 接口

**任务：**

深入了解 `torch` 接口（参考资料：6.torch 接口 API 剩余部分 10~85）

**任务简介：**《Pytorch 官方文档》

**学习时长：**6/5—6/7

**详细说明：**

由于 `torch` 接口部分内容很多，我们利用三天时间深入了解 `torch` 接口该如何使用（其实很多操作和 `numpy` 很类似，只要 `numpy` 很熟练，这块上手很快），第一天主要学习 `torch` 张量(10~20 页)的索引、切片、连接、换位操作，随机采样，序列化操作，并行化操作，其中对张量的操作这一块在以后应用中非常广泛，一定要熟练。第二天学习一些基本的数学操作 API（21~56 页），需要了解基本操作，知道如何使用。第三天学习比较操作和其他操作，这些函数有一些太常用，只需熟练使用常用的操作。

**作业名称（详解）：**掌握基本的 torch Tensor 张量操作；

**作业提交形式：**打卡提交文字或图片，不少于 20 字

**打卡截止时间：**6/7

## 1.4 本周学习任务简单总结

**任务名称：**本周学习任务简单总结

**任务简介：**温故而知新，简单回顾本周学到几个重要知识

**详细说明：**

每一周的学习任务都比较重，第一次学过之后特别容易忘，所以在周日及时做一个要点回顾，会让学习效率大大的提升，不会的知识也会越来越少

**作业名称（详解）：**请用文字描述，本周所学知识的重点，也可以思维导图、手写、电子版截图或者拍照均可，格式不限

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

**打卡截止时间：**6/9

## 2 第 2 周

### 2.1 torch.Storage、torch.cuda 操作

**任务名称：**

Pytorch 官方文档(参考资料：Pytorch 官方文档 8.torch.Storage 操作；14.torch.cuda 操作)

**任务简介：**torch.Storage 主要是 Tensor 数据类型的转换；torch.cuda；

**详细说明：**

Day8 的任务两块，第一块是 Storage 操作，主要包括数据类型转换的接口；第二块是如何使用判断是否有显卡，以及如何把数据和模型在显卡上运行。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**

- (1) 自己练习数据类型之间的转换方法
- (2) 测试在显卡上训练和在 cpu 上训练的速度差多少倍；

**作业提交形式：** 打卡提交文字或图片

打卡截止时间：6/11

## 2.2 数据读取、数据扩增

**任务名称：**

- 1. 数据读取；
- 2. 数据扩增

参考资料：(1) PyTorch\_tutorial\_0.0.5\_余霆嵩文档 1~16 页 (2) pytorch 官方文档  
16.torch.utils.data ; 17.torch.utils.model\_zoo ; 18.torchvision.datasets ;  
19.torchvision.models; 20.torchvision.transforms; 21.torchvision.utils

**任务简介：** 数据读取和自定义数据集的读取操作；数据集的扩增的方法；

**详细说明：**

任务是数据的读取和数据的扩增，在学习之前需要我们把官方文档 16~21 的资料学习，然后学习余霆嵩大神整理的资料，这里包括了数据读取和数据扩增的方法基本操作的 API。数据扩增在我们数据有限的情况下可以通过数据扩增得到更多的数据，一方面可以抑制过拟合一方面可以提高模型泛化性，pytorch 中封装了 22 种数据扩增的方法，基本包括了我们常用扩增方法，可以根据需求调用各自方法。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**

- (1) 使用提供的网络模型读取自己数据进行训练；
- (2) 使用 22 中数据扩增的方法进行组合，测试其效果；

**作业提交形式：** 打卡提交文字或图片，不少于 20 字

打卡截止时间：6/12

## 2.3 构建网络模型

**任务名称：**构建网络模型(包括 torch.nn 和 torch.nn.functional 操作)

**参考文档：**

(1) pytorch 官方文档 torch.nn 和 torch.nn.function

(2) PyTorch\_tutorial\_0.0.5\_余霆嵩文档 17~29 页

**任务简介：**构建网络模型是学习 pytorch 框架最重要的内容

**详细说明：**

任务是构建网络模型，在学习构建网络模型之前需要了解官方文档 9~10 的资料，这里包括了基本的模块，卷积操作、BN 操作、池化操作、激活函数等等在构建模型中最基层的方法。学习初期我们可以先学习经典的网络模型，Lenet, Alexnet, Vgg, Resnet 等网络模型，后面随着学习的深入自己搭建网络模型或者在经典网络模型的基础上增加或修改其网络，使其达到更好的性能；

Torch.nn.function 中的激活函数和池化是我们常用的，一般不使用 torch.nn 中的激活函数和池化操作，因为网络模型反向传播时不计算池化层的梯度，使用在 torch.nn.function 中重新封装了部分接口，当然也可以不使用 torch.nn.function。

**作业资料包下载链接：**(学习资料里包含了老师对官方文档的重组文件，前面一周的文档内容也都有更新，对前面内容还有所疑惑的小伙伴可以回头看看文档资料)

**链接：**<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

**提取码：**34k3

**作业名称（详解）：**

(1) 手敲官方文档 9，10 中的基本操作 3 遍；

(2) 自己尝试写一个 5 层的网络模型；

**作业提交形式：**打卡提交文字或图片，总结内容不少于 20 字

**打卡截止时间：**6/14

## 2.4 本周学习任务简单总结

**任务名称：**本周学习任务简单总结

**任务简介：**温故而知新，简单回顾本周学到几个重要知识

**详细说明：**

每一周的学习任务都比较重，第一次学过之后特别容易忘，所以在周日及时做一个要点回顾，会让学习效率大大的提升，不会的知识也会越来越少

**作业名称（详解）：**请用文字描述，本周所学知识的重点，也可以思维导图、手写、电子版截图或者拍照均可，格式不限

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

打卡截止时间：6/16

## 3 第 3 周

### 3.1 网络模型参数初始化

**任务名称：**

网络模型参数初始化和 Finetune

**参考资料：**

- (1) pytorch 官方文档 torch.nn.init 操作 10 个方法
- (2) PyTorch\_tutorial\_0.0.5\_余霆嵩文档 18~29 页

**任务简介：**网络初始化的方法

**详细说明：**

网络模型参数初始化方法有两种，第一种方法是在我们重新训练模式时使用的初始化方法，这时需要我们自己设置初始化方法，当然也可以不设置，会选择默认的初始化方法，如果需要自己初始化，就需要学会如何设置合适的方法，如果初始化方法选择不合适，会导致模型学习效果不好；第二种方法是 Finetune，就是使用别人预训练好的模型参数初始化自己的网络模型，一般最后一层需要学习，其他层参数都使用预训练模型参数，这种方法不仅收敛速度快而且精度更高，所以一般推荐 finetune 方法。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**

（1）对同一网络模型，测试初始化方法哪一个方法效果更好一些？

（2）比较初始化方法和 `finetune` 方法收敛速度和精度

**作业提交形式：** 打卡提交文字或图片

打卡截止时间：6/18

## 3.2 损失函数

**任务名称：** 损失函数

参考资料：PyTorch\_tutorial\_0.0.5\_余霆嵩文档 31~44 页

**任务简介：** pytorch 中的损失函数

**详细说明：**

pytorch 中有 17 中不同的损失函数，一般最常用的是交叉熵损失函数，其他损失函数也需要了解其用途；

作业资料包下载链接：

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**

对同一网络进行不同损失函数的测试，比对其精度

**作业提交形式：** 打卡提交文字或图片

打卡截止时间：6/21

## 3.3 优化算法 torch.optim

**任务名称：**

优化算法 torch.optim 操作 10 个方法

参考资料：

（1）pytorch 官方文档 torch.optim

（2）PyTorch\_tutorial\_0.0.5\_余霆嵩文档 45~56 页

**任务简介：** pytorch 中的优化算法



**详细说明：**

需要了解常用优化算法的优势和劣势，同时需要了解如何调参，参数中学习率是非常重要的参数，学习率设置不合适会导致震荡或者收敛很慢。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**

对同一网络进行不同优化算法的测试，比对其效果和收敛速度

**作业提交形式：**打卡提交文字或图片

打卡截止时间：6/22

### 3.4 本周学习任务简单总结

**任务名称：**本周学习任务简单总结

**任务简介：**温故而知新，简单回顾本周学到几个重要知识

**详细说明：**

每一周的学习任务都比较重，第一次学过之后特别容易忘，所以在周日及时做一个要点回顾，会让学习效率大大的提升，不会的知识也会越来越少

**作业名称（详解）：**请用文字描述，本周所学知识的重点，也可以思维导图、手写、电子版截图或者拍照均可，格式不限

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

打卡截止时间：6/23

## 4 第 4 周

### 4.1 设置学习率

**任务名称：**设置学习率

**参考资料：**PyTorch\_tutorial\_0.0.5\_余霆嵩文档 49~56 页

**任务简介：**如何设置学习率呢？

**详细说明：**

上一节我们提到学习率非常重要，如何设置学习率呢？Pytorch 给我们提供了 6 中学习率设置的方法，合适的学习率方法可以有效提高模型进度

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**对同一网络进行不同学习率方法的测试，比对其精度

**作业提交形式：**PPT 截图或手写拍照，打卡提交.

**打卡内容：**可以只是文字提交，或图片提交，或组合都行

**打卡截止时间：**6/25

## 4.2 数据可视化

**任务名称：**数据可视化

**参考资料：**PyTorch\_tutorial\_0.0.5\_余霆嵩文档 56~78 页

**任务简介：**利用数据可视化来解释深度学习“黑盒”

**详细说明：**

本节主要是数据可视化部分，可以让我们更直观看到模型内部，主要涉及到卷积核可视化、特征图可视化、梯度及权值分布可视化、混淆矩阵的可视化；这部分比较简单，只需要学习 tensorboardX 中一些常用的 API 即可满足我们日常需要。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称（详解）：**对这几块可视化内容手敲代码一遍

**作业提交形式：**PPT 截图或手写拍照，打卡提交.

**打卡要求：**不少于 2 张图片

**打卡截止时间：**6/28

### 4.3 本周学习任务简单总结

**任务名称：**本周学习任务简单总结

**任务简介：**温故而知新，简单回顾本周学到几个重要知识

**详细说明：**

每一周的学习任务都比较重，第一次学过之后特别容易忘，所以在周日及时做一个要点回顾，会让学习效率大大的提升，不会的知识也会越来越少。

这周涉及到学习率机制和数据可视化两大部分，学习难度不大，主要是需要记住一些模块化的操作即可。

**作业名称（详解）：**

请用文字描述，本周所学知识的重点，也可以思维导图、手写、电子版截图或者拍照均可，格式不限

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

**打卡截止时间：**6/30

## 5 第 5 周

### 5.1 Logistics 分类实战

**任务名称：**Logistics 分类实战

**详细说明和作业：**

通过我们一个月的学习 pytorch 框架，已经掌握了基本操作，今天开始进行项目实战，第一个项目是最简单的神经网络，单层神经网络，其实就是 Logistics 回归，麻雀虽小五脏俱全，这个项目流程和其他项目流程一样，也包括（数据读取、数据处理、定义模型、优化算法和损失函数的选择、迭代训练），所以本节课的重点就是掌握流程化。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称：**通过之前的学习，如何进行优化提高精度？

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

打卡截止时间：7/2

## 5.2 多层感知机 MLP 实战

**任务名称：**多层感知机 MLP 实战

**详细说明和作业：**

通过上一个 Logistics 项目，我们了解了基本的框架结构，今天开始比 Logistics 稍微难一些的多层神经网络也即多层感知机，本节课的重点就是如何通过 MLP 定义网络结构并训练。

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称：**自定义一个 MLP 网络结构进行训练？

**作业提交形式：**PPT 截图或手写拍照，打卡提交。不少于 20 字。

打卡截止时间：7/3

## 5.3 卷积神经网络：Resnet18

**任务名称：**卷积神经网络 CNN：Resnet18

**详细说明和作业：**卷积神经网络 CNN：Resnet18

**详细说明：**

残差神经网络是目前应用最为广泛的网络模型，所以今天主要任务是看懂 resnet18 的代码和方法；

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称：**

自己手敲一遍 resnet18 网络模型，并自己修改其内部结构，包括卷积核大小，输入输出通道数，激活函数，BN 方法，stride 步长大小等等，都可以自己修改并

测试修改后的效果

**作业提交形式：**PPT 截图或手写拍照，打卡提交。

打卡截止时间：7/5

## 5.4 递归神经网络 RNN-LSTM：词性预测

**任务名称：**递归神经网络 RNN-LSTM：词性预测

**详细说明：**

RNN 和 LSTM 主要应用在序列模型中，今天的任务是使用 RNN 进行词性预测；

**作业资料包下载链接：**

链接：<https://pan.baidu.com/s/17xW8rfG-14nu6vc9vjeBlQ>

提取码：[34k3](#)

**作业名称：**

自己手敲一遍词性预测模型，并自己修改其内部结构，测试修改后的效果。

**作业提交形式：**PPT 截图或手写拍照，打卡提交。

打卡截止时间：7/6

## 5.5 温习大家提出的问题（记录在石墨）

大家提出的问题都在石墨文档里，所以大家如果有什么问题可以把这些文档下载下来，看看你有什么问题，电脑手机都可以操作。

《6.20/21Pytorch 问题整理》地址：

<https://shimo.im/docs/YOzcgmEhq5c6kiyc/>

《6.22/23/24 Pytorch 问题整理》地址：

<https://shimo.im/docs/WBs9t4hRVxUyIOvj/>

《6.25/26/27 Pytorch 学习问题》地址：

<https://shimo.im/docs/f2KuEs1OBkEnMpSM/>

《Pytorch7.1 号直播问题大收集》(重新看直播学习问题)地址：

<https://shimo.im/docs/n1HhuXSdxPQOhh8S/>

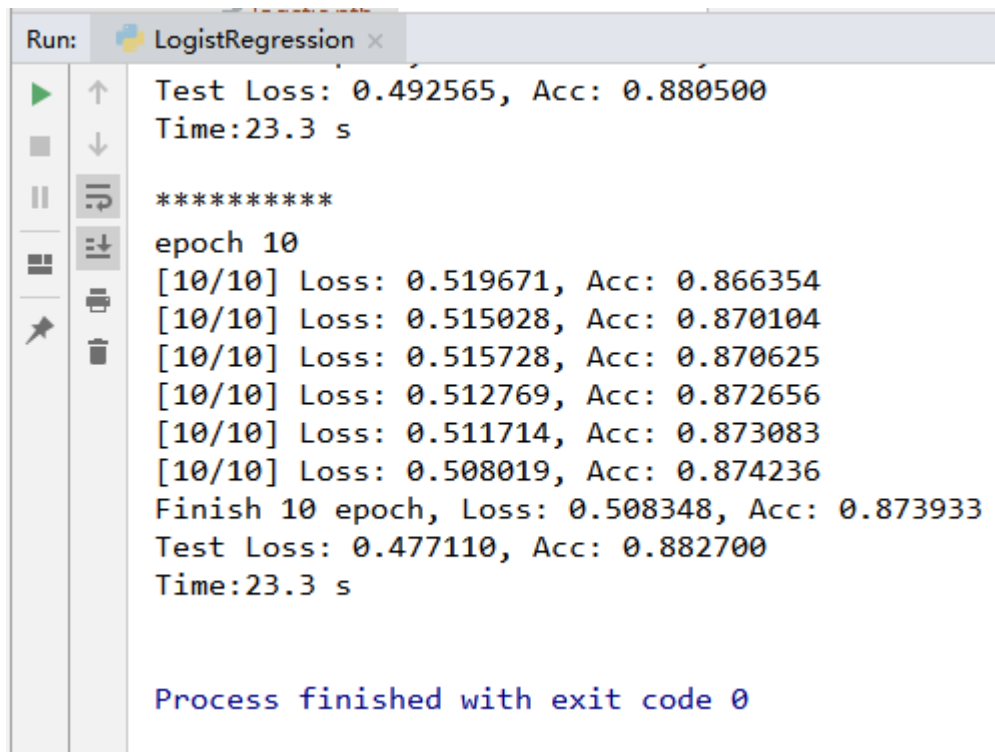
《7 月开始 Pytorch 学习问题》地址:

<https://shimo.im/docs/X2yviKAqRXYFHx2/>

大家群里面如果老师没有回答的问题，可以记录到这个文档，助教会每隔一段时间反馈给老师，给大家解决。

## 5.6 本周实验截图

### 1. Logistics 分类



```
Run: LogistRegression x
Test Loss: 0.492565, Acc: 0.880500
Time:23.3 s

*****
epoch 10
[10/10] Loss: 0.519671, Acc: 0.866354
[10/10] Loss: 0.515028, Acc: 0.870104
[10/10] Loss: 0.515728, Acc: 0.870625
[10/10] Loss: 0.512769, Acc: 0.872656
[10/10] Loss: 0.511714, Acc: 0.873083
[10/10] Loss: 0.508019, Acc: 0.874236
Finish 10 epoch, Loss: 0.508348, Acc: 0.873933
Test Loss: 0.477110, Acc: 0.882700
Time:23.3 s

Process finished with exit code 0
```

以上实验由于笔者电脑的算力不足，故设置 `num_epochs = 10`

### 2. 多层感知机 MLP

```
Run: MLP x
Train Epoch: 7 [0/60000 (0%)] Loss: 0.777839
Train Epoch: 7 [20000/60000 (33%)] Loss: 0.629023
Train Epoch: 7 [40000/60000 (67%)] Loss: 0.788232
Test set: Average loss: 0.0034, Accuracy: 7521/10000 (75%)
Train Epoch: 8 [0/60000 (0%)] Loss: 0.620765
Train Epoch: 8 [20000/60000 (33%)] Loss: 0.750411
Train Epoch: 8 [40000/60000 (67%)] Loss: 0.692493
Test set: Average loss: 0.0033, Accuracy: 7534/10000 (75%)
Train Epoch: 9 [0/60000 (0%)] Loss: 0.699667
Train Epoch: 9 [20000/60000 (33%)] Loss: 0.733817
Train Epoch: 9 [40000/60000 (67%)] Loss: 0.661531
Test set: Average loss: 0.0033, Accuracy: 7563/10000 (75%)
9920512it [15:46, 10485.77it/s]
1654784it [06:17, 4378.05it/s]
Process finished with exit code 0
```

### 3. 卷积神经网络: Resnet18

由于笔者的本机电脑算力不足, 无法得到实验结果。

### 4. 递归神经网络 RNN-LSTM: 词性预测

```
Run: CNN-Classification x RNN-LSTM x
D:\MyPythonWork\Pytorch-Camp\venv\Scripts\python.exe D:\MyPythonWork\Pytorch-Camp\src\Week5\Day33\RNN-LSTM.py
{'the': 0, 'dog': 1, 'ate': 2, 'apple': 3, 'everybody': 4, 'read': 5, 'that': 6, 'book': 7}
{'det': 0, 'nn': 1, 'v': 2}
{'a': 0, 'b': 1, 'c': 2, 'd': 3, 'e': 4, 'f': 5, 'g': 6, 'h': 7, 'i': 8, 'j': 9, 'k': 10, 'l': 11, 'm': 12, 'n': 13, 'o': 14, 'p': 15, 'q': 16, 'r': 17, 's': 18, 't': 19, 'u': 20, 'v': 21, 'w': 22, 'x': 23, 'y': 24, 'z': 25}
Epoch: 50, Loss: 0.94706
Epoch: 100, Loss: 0.74630
Epoch: 150, Loss: 0.53950
Epoch: 200, Loss: 0.36579
Epoch: 250, Loss: 0.24595
Epoch: 300, Loss: 0.17084
out= tensor([[-1.2255,  1.6552, -0.3672],
              [-0.9423, -0.1564,  1.2740],
              [ 1.6619, -0.7175, -0.7306],
              [-0.5624,  1.7271, -0.9589]], grad_fn=<AddmmBackward>)
tag_to_idx= {'det': 0, 'nn': 1, 'v': 2}
Process finished with exit code 0
```

最后可以得到上面的结果, 因为最后一层的线性层没有使用 `softmax`, 所以数值不太像一个概率, 但是每一行数值最大的就表示属于该类, 可以看到第一个单词 'Everybody' 属于 nn, 第二个单词 'ate' 属于 v, 第三个单词 'the' 属于 det, 第四个单词 'apple' 属于 nn, 所以得到的这个预测结果是正确的。