基于 SVM 的文本特征分类器

——以外太空探索中的国际公约为场景

李沐晟 无 17 202101284 lms21@mails.tsinghua.edu.cn

一、摘要

在当代，太空飞行器的应用已经成为人类探索宇宙、提升科技水平、推动经济发展的重要手段。然而，随着太空活动的增加，国际社会面临着日益严峻的挑战，确保外太空的和平与可持续发展变得至关重要。为此，一些重要的国际公约应运而生。这些公约为各国在外太空的活动设定了基本规则和行为准则。

在本次实验中，我们以外太空探索中的国际公约为场景，筛选出其中的 4 条作为公约，并利用大语言模型(ChatGPT-4o)分别生成数条句子结构相似的、违反某一条规定的情景描述，作为合成数据集。随后我们使用文本嵌入模型(Embedding-2, 智谱)将文本转化为特征向量，并使用主成分分析(PCA)方法对其特征降维到 2 维。划定训练集、测试集与验证集后，我们通过训练支持向量机(SVM)以实现分类任务，并取得了非常良好的分类结果。由于 SVM 是二分类器，为了实现多分类任务，我们在每两类模式间各自训练出一套支持向量机模型，在判决测试集中数据时，采用一对一(OvO)方法，即“逐一分类+投票法”的策略，得到最终的判决结果。本套分类方案在测试集中得到了惊人的 100%分类准确率(Accuracy)结果。

二、数据生成与处理

1、数据生成

通过查阅网络相关文献以及大语言模型，了解到在外太空探索方面的国际公约有*《外层空间条约》（Outer Space Treaty, 1967）*、*《月球协定》（Moon Agreement, 1984）*、*《空间物体造成损害的国际责任公约》（Liability Convention, 1972）*、*《外空物体登记公约》（Registration Convention, 1976）*、*《国际电信联盟公约》（ITU Convention）*等等，经过选择与总结，将上述规则的内容整理为如下几条公约：

(1) 禁止在外太空部署核武器或其他大规模杀伤性武器。

(2) 外太空，不得被任何一个国家通过宣示主权、占领或其他手段据为己有。

(3) 各国应采取措施，减少和管理太空垃圾，以确保外太空环境的可持续利用。

(4) 各国必须向联合国登记其发射的航天器，以确保外太空活动的透明度。

随后利用大语言模型(ChatGPT-4o)针对上述公约分别生成 70 条违反的描述（如“某国在某情况下违反某条约”等），其中训练数据 40 条，验证数据 10 条，测试数据 20 条。对于全部不违反的描述类别，生成训练数据 40 条，验证数据 20 条，测试数据 20 条。

具体数据详见./data 文件夹内的 train、test、val 子文件夹。

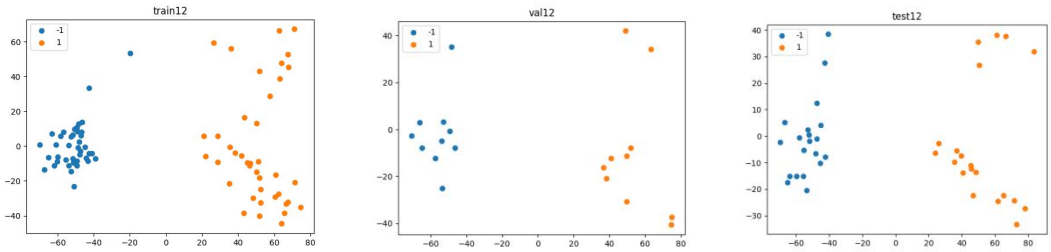
2、数据预处理

在 data_preprocess.py 文件中，我们对生成的数据进行数据预处理。首先使用文本嵌入模型 Embedding-2，将每段描述转化为 1024 维度的特征向量。随后两两类别之间使用主成分分析法(PCA)进行特征降维，得到 2 维特征向量。

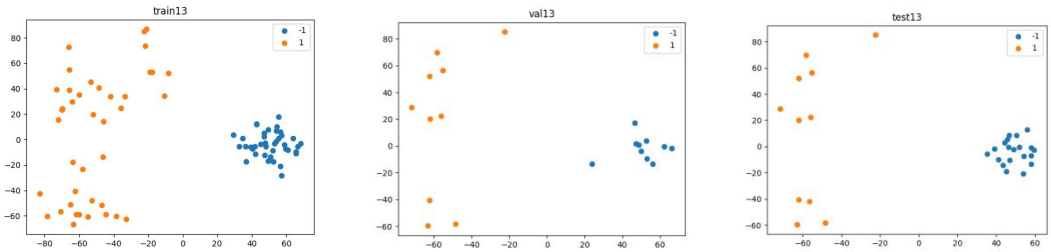
我们将训练集、测试集、验证集的数据(data)与标签(label)分别处理后储存为 train.pt、test.pt、val.pt 文件(文件后缀中有两两类别的编号：“12”“13”“14”“23”“24”“34”，共六项)，储存在./data 文件夹内。

数据预处理可视化结果如下(从左至右依次为训练集、验证集、测试集)：

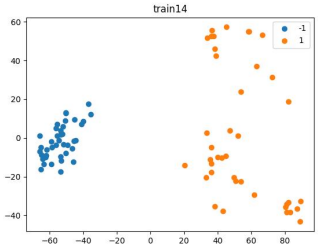
1-2 两类：

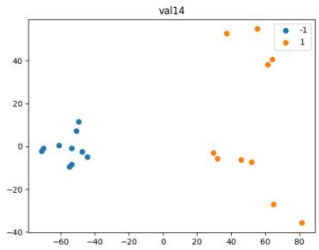


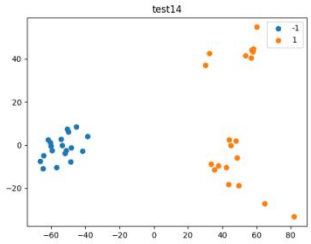
1-3 两类：



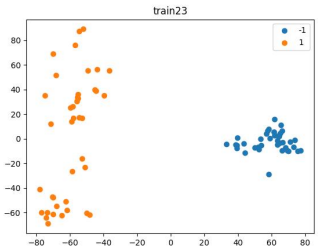
1-4 两类：

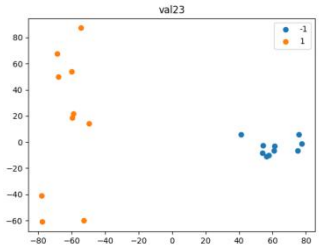


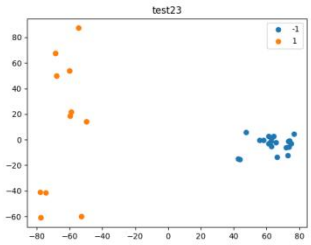




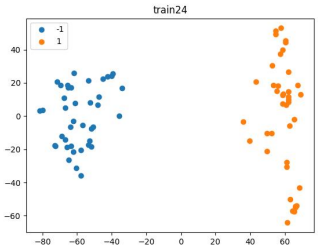
2-3 两类：

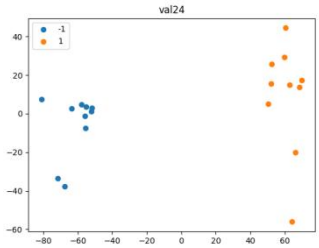


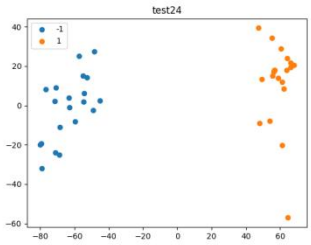




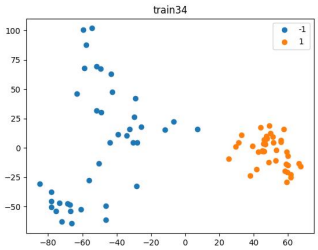
2-4 两类：

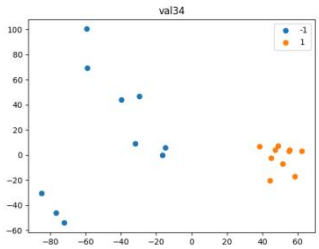


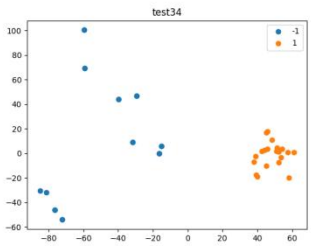




3-4 两类：







根据上述数据预处理结果，可以看见两两类别之间呈现出明显的“聚类”特征，可以划分出明显的分类界面，适合使用 SVM 搭建一个分类界面进行分类。

注：根据原先实验设计，原本计划训练一个分类“有/无违反公约”的分类器，但是根据数据预处理结果，其特征向量结构难以通过 SVM 实现分类，故最终没有实现，后文中将详细阐述。

三、模型训练与判决

1、模型框架

在本次实验中，我们采用 SVM 模型作为分类器，其网络结构在 svm_hw.py 中定义，主体由简单的线性层加上 Hinge Loss 功能构成。

2、训练过程

正如上文提到的，我们在两两类别之间分别训练出一个 SVM 分类模型，总共有 6 组分类模型。我们在终端中输入指令来实现模型训练：

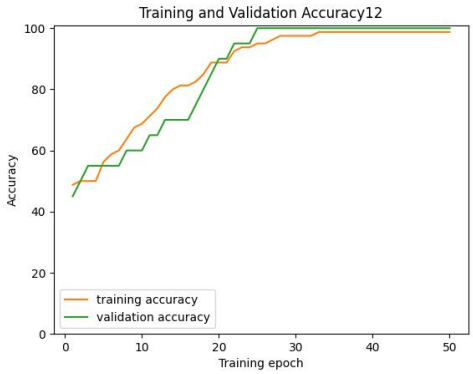
python .\train_svm.py --device cpu --n_epoch 50 --C 1e-1 --classes xx

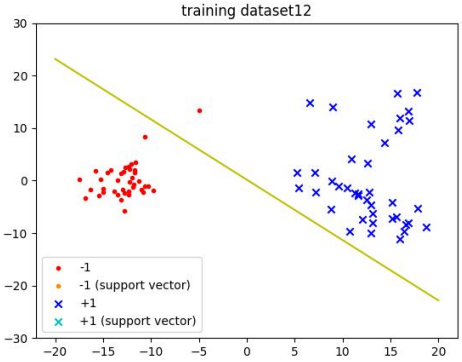
经过调试与判断，设置训练轮数(epoch)为 50、C 因子(regularization coefficient)为 0.1、学习率为 0.01 能取得理想的训练结果。经过六次训练，分别得到六个两两分类的 SVM 模型，储存在./checkpoints 文件夹中。

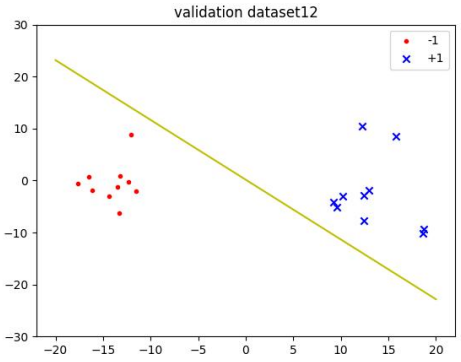
训练过程可视化结果如下：

1-2 两类：

```
Epoch 46: loss = 1.095, training accuracy = 98.8%
Epoch 46: validation accuracy = 100.0%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 1.062, training accuracy = 98.8%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 1.056, training accuracy = 98.8%
Epoch 50: loss = 1.041, training accuracy = 98.8%
```



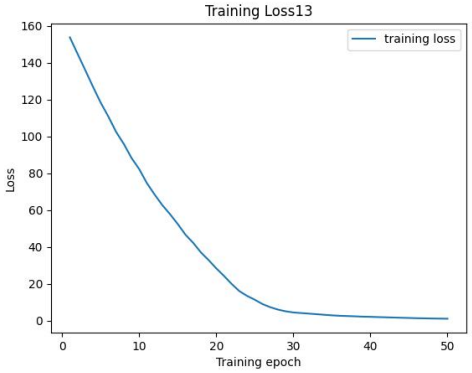


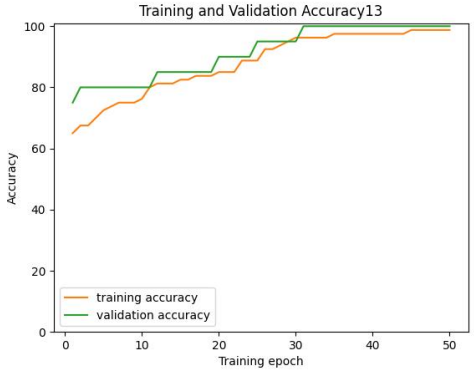


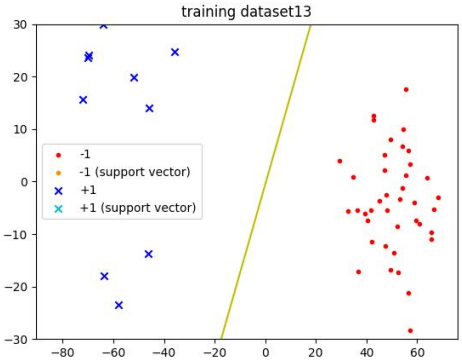

```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 12
Test accuracy = 100.0%
```

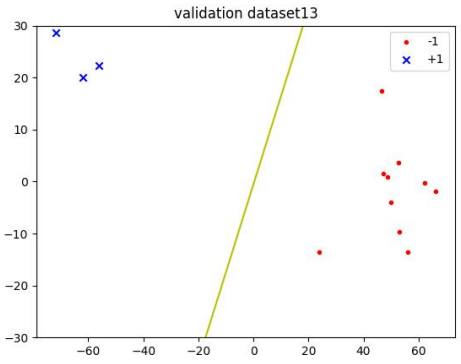
1-3 两类：

```
Epoch 47: loss = 1.314, training accuracy = 98.8%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 1.211, training accuracy = 98.8%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 1.161, training accuracy = 98.8%
Epoch 49: validation accuracy = 100.0%
Epoch 50: loss = 1.109, training accuracy = 98.8%
Epoch 50: validation accuracy = 100.0%
Model saved in checkpoints/svm13.pth
```





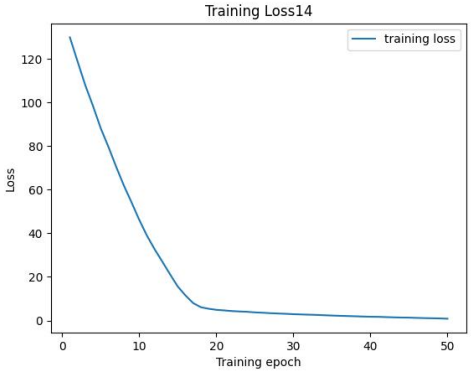


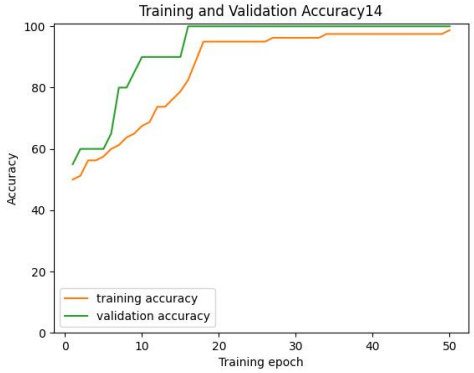


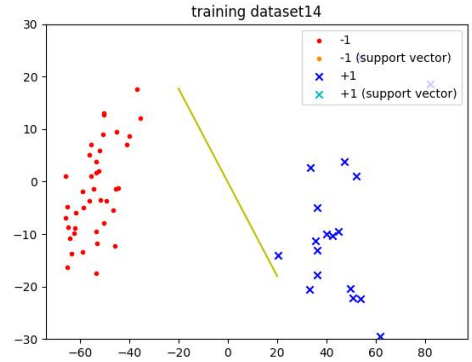

```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 13
Test accuracy = 100.0%
```

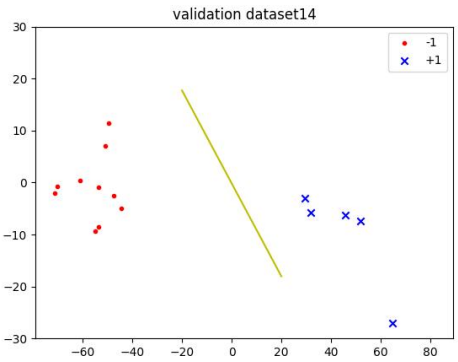
1-4 两类：

```
Epoch 47: loss = 1.104, training accuracy = 97.5%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 1.024, training accuracy = 97.5%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 0.950, training accuracy = 97.5%
Epoch 49: validation accuracy = 100.0%
Epoch 50: loss = 0.825, training accuracy = 98.8%
Epoch 50: validation accuracy = 100.0%
Model saved in checkpoints/svm14.pth
```





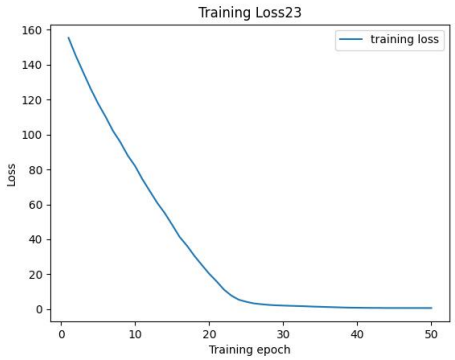


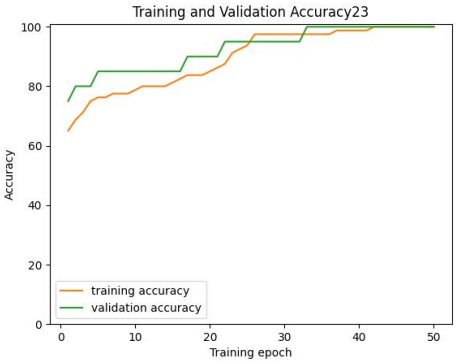


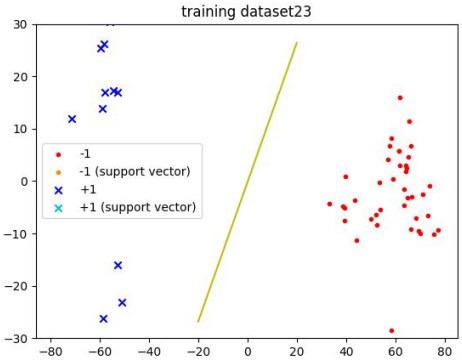

```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 14
Test accuracy = 100.0%
```

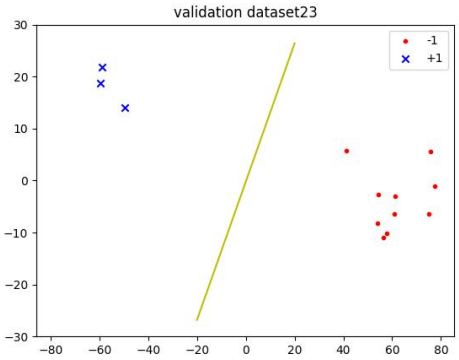
2-3 两类：

```
Epoch 47: loss = 0.672, training accuracy = 100.0%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 0.671, training accuracy = 100.0%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 0.671, training accuracy = 100.0%
Epoch 49: validation accuracy = 100.0%
Epoch 50: loss = 0.670, training accuracy = 100.0%
Epoch 50: validation accuracy = 100.0%
Model saved in checkpoints/svm23.pth
```





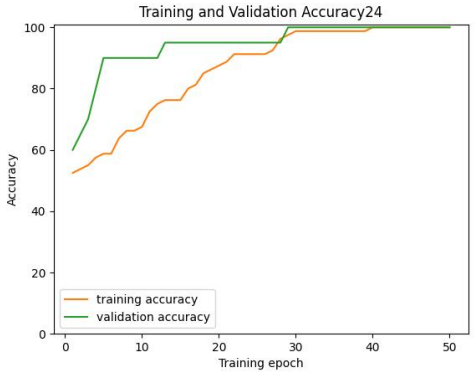


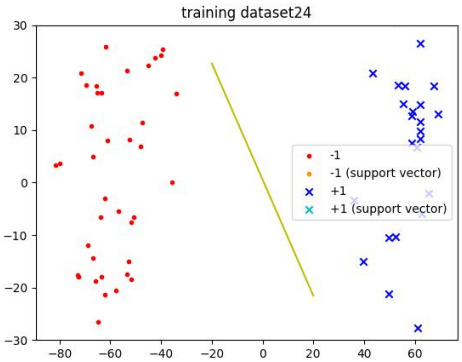


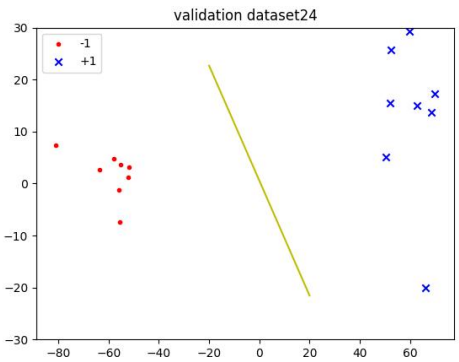

```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 23
Test accuracy = 100.0%
```

2-4 两类：

```
Epoch 47: loss = 0.556, training accuracy = 100.0%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 0.555, training accuracy = 100.0%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 0.554, training accuracy = 100.0%
Epoch 49: validation accuracy = 100.0%
Epoch 50: loss = 0.553, training accuracy = 100.0%
Epoch 50: validation accuracy = 100.0%
Model saved in checkpoints/svm24.pth
```



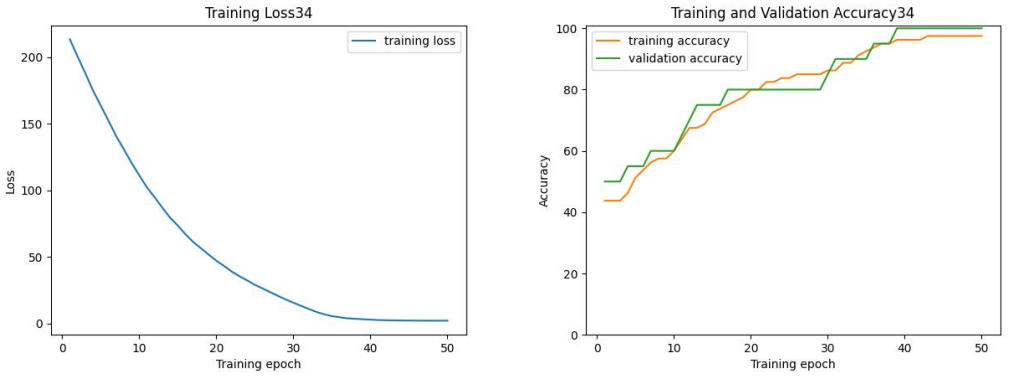


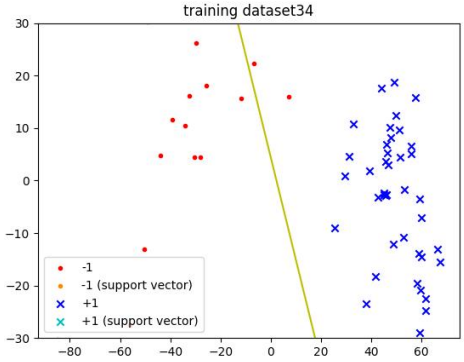


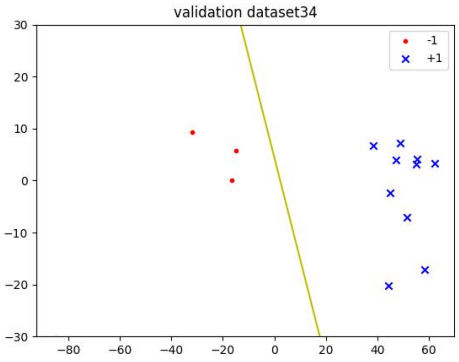

```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 24
Test accuracy = 100.0%
```

3-4 两类：

```
Epoch 47: loss = 2.105, training accuracy = 97.5%
Epoch 47: validation accuracy = 100.0%
Epoch 48: loss = 2.086, training accuracy = 97.5%
Epoch 48: validation accuracy = 100.0%
Epoch 49: loss = 2.071, training accuracy = 97.5%
Epoch 49: validation accuracy = 100.0%
Epoch 50: loss = 2.062, training accuracy = 97.5%
Epoch 50: validation accuracy = 100.0%
Model saved in checkpoints/svm34.pth
```








```
(http) PS C:\Users\asus\Desktop\NLP_Classifier> python test_svm.py --device cpu --classes 34
Test accuracy = 100.0%
```

由上可见，分类器的训练结果良好，train accuracy 与 test accuracy 均非常高(接近 100%)，并且从图中可以看出训练出的分类界面合理，能很好地实现二分类功能。

3、最终判决

为了对 SVM 进行扩展，实现多分类功能，我采取了一对一(OvO)方法，具体做法如下：对某一待分类文本，首先将其转化为特征向量并特征降维得到 2 维特征向量，随后使用每一个分类器(共 6 个)依次进行处理得到判决结果，最终使用“投票法”，统计出现次数最多的判决结果，作为最终的多分类判决结果(详见代码)。运行 test_all.py 代码，结果如下：

```
测试完毕...
Test accuracy = 100.0%
```

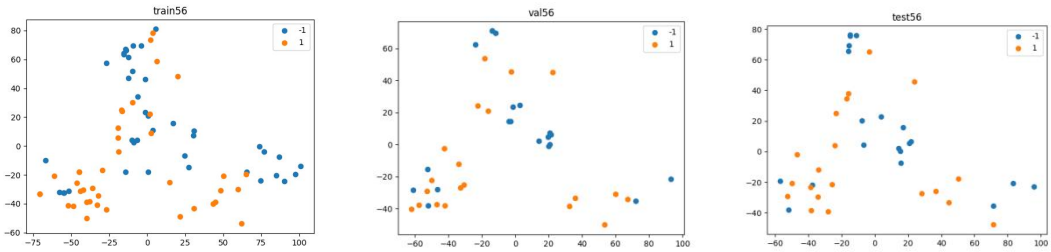
可见分类性能非常优异，所有测试数据均正确分类。

四、实验总结与思考

在本次实验中，本人成功利用支持向量机（SVM）模型实现了文本特征的分类任务。总结而言，在数据生成与预处理方面，本实验通过查阅相关文献并利用大语言模型（如 ChatGPT-4）生成了符合特定公约的违反描述，并将其嵌入为高维特征向量表示。随后，通过主成分分析（PCA）方法将特征降维为二维，这为后续的 SVM 分类奠定了基础；在模型训练与测试方面，采用一对一（OvO）策略，在每两类之间分别训练 SVM 模型，共训练了六组分类模型。通过合理设置超参数（如训练轮数、正则化系数等），得到了训练准确率和测试准确率均接近 100%的分类结果；最终的多分类任务通过“逐一分类+投票法”的策略实现，测试结果显示所有测试数据均被正确分类，表现出非常高的分类性能。

当然本次实验中实际上存在一些设计上的不足与缺陷。本次实验中的分类器与其说是针对“是否违反某一条公约”的分类器，实质上更是针对“围绕不同条公约的描述”的分类器，具体可以从处理“是/否违反公约”的情景描述数据预处理中看出。

在下面可视化图像中，5 类代表不违反任何一条公约的描述，6 类表示违反四条公约中某一条的描述，实际上这样的数据是杂糅在一起的，“是否违反”这一语义特征被“描述主题”这一语义特征所掩盖。



当然，我们也可以看出第 5 和第 6 类的样本点是存在一定偏移且呈现分离趋势的，故理论上是可分的。后续我们可以借助核函数、MLP 等等非线性方法实现进一步的分类。