2021-2022学年第2学期

《深度学习》

课程实验报告

实验四：视频语义理解

姓名：XXX[1]　　学号：XXXXXXXXX

2023年4月22日

Chapter 1

实验概述

1.1 实验目的

本实验以百度飞桨AI Studio 2022语言与智能技术竞赛视频语义理解赛题[1]的常规赛版本——千言数据集：视频语义理解评测大赛[2]为依托，旨在实现以下目的：

1. 了解并学习飞桨深度学习框架PaddlePaddle，熟悉开放、共享、互联的人工智能开发生态系统AI Studio，助力国产人工智能产品和系统，为国家人工智能产业发展战略提供储备。
2. 了解视频语义处理技术，以现实产业背景为驱动，跟踪科技前沿。
3. 熟悉AI Studio云平台的使用，训练学生在大规模数据集的研究问题下高效的处理方案。
4. 鼓励创新问题的解决方案，为视频语义理解技术提出有价值的工作。

1.2 赛题描述与实验内容

1.2.1 赛题描述

在移动互联网、大数据的时代背景下，互联网上的视频数据呈现爆发式增长，作为日益丰富的信息承载媒介，视频的深度语义理解是诸多视频

[1]大赛地址：http://lic2022.cipsc.org.cn/
[2]大赛地址：https://aistudio.baidu.com/aistudio/competition/detail/434/0/introduction

图 1.1: LIC·2022语言与智能技术竞赛

智能应用的基础，具有重要的研究意义和实际应用价值。传统基于感知的视频内容分析缺乏语义化理解能力，而充分利用知识图谱的语义化知识并结合多模态学习和知识推理技术，有望实现更深入的视频语义理解。

知识增强的视频语义理解任务，期望融合知识、NLP、视觉、语音等相关技术和多模态信息，为视频生成刻画主旨信息的语义标签，从而实现视频的语义理解。本评测任务以互联网视频为输入，在感知内容分析（如人脸识别、OCR识别、语音识别等）的基础上，期望通过融合多模信息，并结合知识图谱计算与推理，为视频生成多知识维度的语义标签，进而更好地刻画视频的语义信息。

竞赛任务在于：知识增强的视频语义理解任务，以百度好看视频、全民小视频资源为对象，在感知内容分析的基础上，融合知识、语言、视觉、语音等多模信息，结合知识计算与推理，为视频生成相应的语义标签。具体来讲，在完成对视觉基础内容分析的基础上，利用知识进行计算与推理，对百度好看视频、全民小视频从分类标签、语义标签（包括：实体/概念/事件/实体属性等维度）层面进行理解，并为其生成这几个层面相应的语义标签结果。

竞赛任务的输入输出定义为：

1. 输入：
   (a) 视频数据
   (b) 基础感知解析结果（OCR、语音识别、人脸识别结果等）
2. 输出：视频标签，包括以下几类：

(a) 分类标签：二层体系的封闭集

(b) 语义标签：实体/概念/事件/实体属性等标签

1.2.2 数据集与评估方法

数据集包含训练集、A榜测试集、B榜测试集和知识库数据。

训练集：训练集由好看视频、度小视-全民小视频资源抽样构建而成，提供了约4.5万个视频用于模型的训练；数据集中的视频均来自真实应用数据，标签结果为人工标注而成。

A榜测试集： 数据构造方法与训练集完全相同，共包含约1万个样本，用于模型在线评测。

知识库： 从百度百科中抽取得到，共包含约30万条知识，作为知识推理的依据。

评估采用传统的F1-Measure作为评测指标，通过将输出结果与人工标准集合进行比较来计算F1 分值。计算公式如下：

$$F1 = \frac{2 \cdot P \cdot R}{P + R}$$

其中$P = \frac{\sum_{i \in N} |TG(i) \cap TP(i)|}{\sum_{i \in N} |TP(j)|}$，$R = \frac{\sum_{i \in N} |TG(i) \cap TP(i)|}{\sum_{i \in N} |TG(j)|}$。

其中，评测集中的标签结果为：$TG = t_1, t_2, \cdots, t_n$，$n$为评测集的标签数目,参赛系统预测出的标签结果为：$TP = t'_1, t'_2, \cdots, t'_m$，$m$为预测标签数目。$N$为评测集的视频数目。

1.2.3 实验内容

本实验的内容如下：

1. 在AI Studio平台上运行BaseLine，「一键运行」跑出结果并提交。
2. 使用Pytorch框架搭建模型（可选）
3. 根据优秀方案搭建/自己设计分类标签模型（可选）
4. 根据优秀方案搭建/自己设计语义标签模型（可选）
5. 使用知识库进行知识推理（可选）
6. score达到50以上（可选）
7. 完成其他有价值的工作（可选）

Chapter 2

实验原理

实验采用了VideoTag预训练模型进行训练。VideoTag采用两阶段建模方式，由两个模型组成: TSN + AttentionLSTM。

2.1 VideoTag 飞桨大规模视频分类模型

飞桨大规模视频分类模型VideoTag基于百度短视频业务千万级数据，支持3000个源于产业实践的实用标签，具有良好的泛化能力，非常适用于国内大规模（千万/亿/十亿级别）短视频分类场景的应用。VideoTag采用两阶段建模方式，即图像建模和序列学习。第一阶段，使用少量视频样本（十万级别）训练大规模视频特征提取模型(Extractor)；第二阶段，使用千万级数据训练预测器(Predictor)，最终实现在超大规模（千万/亿/十亿级别）短视频上产业应用，其原理示意如图2.1所示。

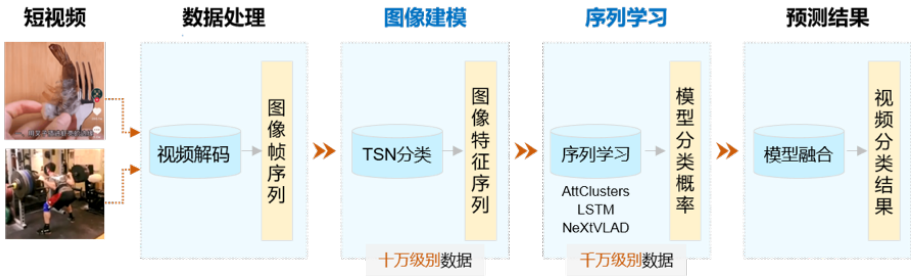


图 2.1: VideoTag大规模视频分类模型

在视频分类任务的数据处理阶段，需要先对短视频进行解码，然后再

将输出的图像帧序列灌入到VideoTag中进行训练和预测。之后在图像建模阶段，先从训练数据中，对每个类别均匀采样少量样本数据，构成十万量级的训练视频。然后使用TSN网络进行训练，提取所有视频帧的TSN模型分类层前一层的特征数据。在这个过程中，每一帧都被转化成相应的特征向量，一段视频被转化成一个特征序列。再之后进入序列学习阶段，采用Attention clusters、LSTM和Nextvlad对特征序列进行建模，学习各个特征之间的组合方式，进一步提高模型准确率。由于序列学习相比于图像建模耗时更短，因此可以融合多个具有互补性的序列模型。示例代码仅使用Attention_LSTM网络进行序列特征预测。最后融合多个模型结果实现视频分类，预测出分类结果。

2.2 TSN简述

TSN全称为Temporal Segment Network[1]，是经典的基于2D-CNN的视频分类模型。该基于长距离时序建模的思想，结合时序稀疏采样(sparse temporal sampling)策略和视频级监督(video-level supervision)进行视频动作识别。模型的结构如图2.2所示。该模型将一个输入视频分成K个片段，从每个片段中随机抽取一个短片段。不同片段的分类分数通过片段一致性函数进行融合得到片段一致性，这是一种视频级别的预测。然后将所有模式的预测融合以产生最终预测。所有片段上的卷积网络共享参数。

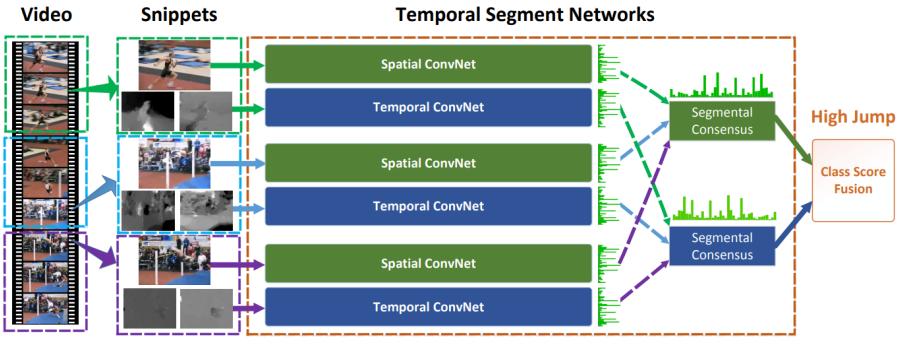


图 2.2: TSN模型

TSN基于双流网络。不同于传统双流网络只接收1帧RGB和几帧光流，TSN在稀疏取样自整个视频的snippets上进行操作，每个snippet都会产生一个初步的动作分类预测，然后把每个snippets的预测形成一个共识(consensus)，作为整个视频的预测(video-level prediction)。TSN可以总结为如下的公式：

$$TSN(T_1, T_2, \cdots, T_K) = H(G(F(T_1; W), F(T_2; W), \cdots, F(T_K; W)))$$

其中，$(T_1, T_2, \cdots, T_K)$是snippets序列。每个snippet T_k是从对应视频段segment S_k中随机取样(randomly sample)得到，每个segment中采样一个snippet，包括1帧RGB图像和数帧光流。

$F(T_L; W)$是是作用在T_K上，参数为W的卷积网络，输出该snippet属于每个类的分类score，即一个向量，其分量F_i表示该snippet属于第i类的打分。

G 代表segmental consensus function，将多个snippet的F输出进行合并，获得一个分类consensus，即一个向量，其分量G_i表示整个视频V属于第i类的打分。

H 则是预测函数，基于consensus预测视频V属于所有分类的概率，这里采用softmax。输出一个向量，其分量H_i表示整个视频V经softmax归一化后属于第i类的概率。

2.3 Attention LSTM简述

VideoTag模型中所用的Attention LSTM模型是一类研究很广的网络，2018年Xiang Long等[2]的工作对此类模型进行了一个创新性的研究，提出的一种基于注意力聚类的局部特征集成框架，同时该工作引入了移位操作以捕获更多的多样信号。其模型架构如图2.3所示。

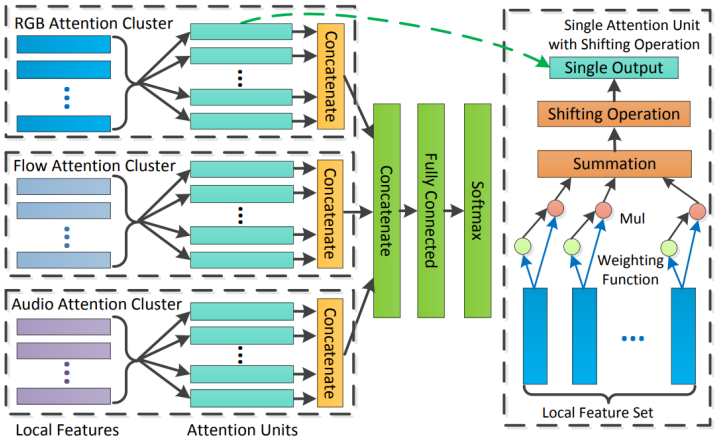


图 2.3: 多模态注意力聚类与移位操作：视频分类的整体架构。不同的特征集采用独立的注意力聚类，然后将输出连接起来进行分类。

AttentionLSTM以视频的特征向量作为输入，采用双向长短时记忆网络（LSTM）对所有帧特征进行编码，并增加Attention层，将每个时刻的隐状态输出与自适应权重线性加权得到最终分类向量。Attention LSTM结构在每个时间步使用注意力机制对输入序列中的所有元素进行加权，然后加权后的元素与LSTM的输出状态进行拼接，作为下一个时间步的输入。这样，Attention LSTM结构可以根据输入序列中的不同部分自适应地调整权重，从而提高模型的表现力和泛化能力。

Attention LSTM结构的核心是注意力机制，它可以通过不同的方式来计算加权系数。在Xiang Long等人认为，注意力输出本质上可以被视为向量集上的加权平均：

$$\mathbf{v} = a\mathbf{X},$$

其中a是一个维度为L的权重向量，由一个加权函数确定。选择加权函数是最关键的设计决策。它的输入是局部特征集X，输出是权重向量$\mathbf{a}$，其“1范数”为$\mathbf{l}$。权重向量的每个维度对应一个局部特征。有许多方法可以计算局部特征的权重。例如，全局平均可以被视为注意力的退化形式，相应的加权函数可以表示为：

$$\mathbf{a} = \frac{1}{L}\mathbf{l},$$

其中$\mathbf{l}$是一个维度为L的向量，所有元素均相等。为了得到更加灵活的注意力权重函数，我们可以使用一个只有一个神经元的全连接层（FC1）来进行计算，也可以使用两个连续的全连接层，后者的计算方式为：

$$\mathbf{a} = softmax(\mathbf{w}_2 tanh(\mathbf{W}_1\mathbf{X}^T + b_1) + b_2),$$

其中$\mathbf{W}_1$是一个维度为$H \times M$的参数矩阵，b_1和w_2是维度为H的参数向量，b2是大小为L的参数向量。

Chapter 3

实验过程

3.1 实验环境配置

BML CodeLab云端实验环境如图3.1所示

图 3.1: BML CodeLab实验环境

3.2 数据集准备

首先需要进行数据集的解压缩、解析和加载等过程。该过程耗时较长，输出内容长，部分截图分别如3.2和3.3所示：

```
external-libraries/h5py/_selector.cpython-37m-x86_64-linux-gnu.so
external-libraries/h5py/ipy_completer.py
external-libraries/urllib3-1.26.9.dist-info/
external-libraries/urllib3-1.26.9.dist-info/top_level.txt
external-libraries/urllib3-1.26.9.dist-info/WHEEL
external-libraries/urllib3-1.26.9.dist-info/METADATA
external-libraries/urllib3-1.26.9.dist-info/RECORD
external-libraries/urllib3-1.26.9.dist-info/LICENSE.txt
external-libraries/urllib3-1.26.9.dist-info/INSTALLER
external-libraries/packaging-21.3.dist-info/
external-libraries/packaging-21.3.dist-info/top_level.txt
external-libraries/packaging-21.3.dist-info/WHEEL
external-libraries/packaging-21.3.dist-info/LICENSE.BSD
external-libraries/packaging-21.3.dist-info/METADATA
external-libraries/packaging-21.3.dist-info/LICENSE
external-libraries/packaging-21.3.dist-info/RECORD
external-libraries/packaging-21.3.dist-info/LICENSE.APACHE
external-libraries/packaging-21.3.dist-info/INSTALLER
```

2.2 数据加载

数据内容包含：

- 样例训练集（比赛使用训练集的抽样集合）的视频信息，及官方提供的tsn视觉特征
- A榜测试集（比赛使用的A榜测试集全量集合）的视频信息，及官方提供的tsn视觉特征

(a) 创建文件目录

```
inflating: tsn_features_train_sample/609096990846578847 6.npy
inflating: tsn_features_train_sample/484339060162578262 3.npy
inflating: tsn_features_train_sample/154444746865977766 67.npy
inflating: tsn_features_train_sample/639381589209737091.npy
inflating: tsn_features_train_sample/410902557413017808 1.npy
inflating: tsn_features_train_sample/182272828773117176 46.npy
inflating: tsn_features_train_sample/404629133602792728 4.npy
inflating: tsn_features_train_sample/157888902462602585 9.npy
inflating: tsn_features_train_sample/882829417596717298 6.npy
inflating: tsn_features_train_sample/418775654215651489 3.npy
inflating: tsn_features_train_sample/147210297660956313 65.npy
inflating: tsn_features_train_sample/637399774984178172 0.npy
inflating: tsn_features_train_sample/475323686045160723 5.npy
inflating: tsn_features_train_sample/961184111188294146 0.npy
```

2.3 视频分类标签基线

该基线基于 VideoTag飞桨大规模视频分类模型，能够根据视频内容在封闭的二级标签体系上进行分类，得到描述视频的分类标签。

(b) 解压数据集

图 3.2: 数据解压

```
        ("train", 4.0 / 5.0),
        ("val", 1.0 / 5.0),
    ])
train_data = [trainval_data[idx] for idx in split2indice["train"]]
val_data = [trainval_data[idx] for idx in split2indice["val"]]

prepare_split(trainval_data, "trainval", gather_labels=True)
prepare_split(train_data, "train")
prepare_split(val_data, "val")
prepare_split(test_data, "test", test_only=True)
```

```
Loading /home/aistudio/dataset_sample/train.sample.json...
Loading /home/aistudio/dataset_sample/test_a.json...
Saved /home/aistudio/paddle-video-classify-tag/data/level1_label.txt
Saved /home/aistudio/paddle-video-classify-tag/data/level1_trainval.list, size=8085
Saved /home/aistudio/paddle-video-classify-tag/data/level2_label.txt
Saved /home/aistudio/paddle-video-classify-tag/data/level2_trainval.list, size=8085
Saved /home/aistudio/paddle-video-classify-tag/data/level1_train.list, size=6468
Saved /home/aistudio/paddle-video-classify-tag/data/level2_train.list, size=6468
Saved /home/aistudio/paddle-video-classify-tag/data/level1_val.list, size=1617
Saved /home/aistudio/paddle-video-classify-tag/data/level2_val.list, size=1617
Saved /home/aistudio/paddle-video-classify-tag/data/level1_test.list, size=9939
Saved /home/aistudio/paddle-video-classify-tag/data/level2_test.list, size=9939
```

图 3.3: 数据加载

3.3 模型训练与预测

模型训练与预测分为了视频分类标签的训练与预测和视频语义标签的训练与预测两部分组成。两部分的运行截图分别如图所示。

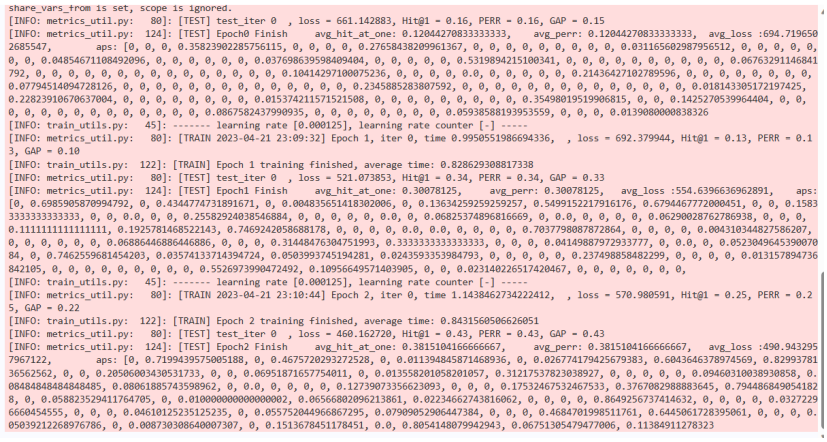


图 3.4: 视频分类标签模型训练

```
[INFO: 2819548844.py:  105]: Processed 3801 samples
[INFO: 2819548844.py:  105]: Processed 4001 samples
[INFO: 2819548844.py:  105]: Processed 4201 samples
[INFO: 2819548844.py:  105]: Processed 4401 samples
[INFO: 2819548844.py:  105]: Processed 4601 samples
[INFO: 2819548844.py:  105]: Processed 4801 samples
[INFO: 2819548844.py:  105]: Processed 5001 samples
[INFO: 2819548844.py:  105]: Processed 5201 samples
[INFO: 2819548844.py:  105]: Processed 5401 samples
[INFO: 2819548844.py:  105]: Processed 5601 samples
[INFO: 2819548844.py:  105]: Processed 5801 samples
[INFO: 2819548844.py:  105]: Processed 6001 samples
[INFO: 2819548844.py:  105]: Processed 6201 samples
[INFO: 2819548844.py:  105]: Processed 6401 samples
[INFO: 2819548844.py:  105]: Processed 6601 samples
[INFO: 2819548844.py:  105]: Processed 6801 samples
[INFO: 2819548844.py:  105]: Processed 7001 samples
[INFO: 2819548844.py:  105]: Processed 7201 samples
[INFO: 2819548844.py:  105]: Processed 7401 samples
[INFO: 2819548844.py:  105]: Processed 7601 samples
[INFO: 2819548844.py:  105]: Processed 7801 samples
[INFO: 2819548844.py:  105]: Processed 8001 samples
[INFO: 2819548844.py:  105]: Processed 8201 samples
[INFO: 2819548844.py:  105]: Processed 8401 samples
[INFO: 2819548844.py:  105]: Processed 8601 samples
[INFO: 2819548844.py:  105]: Processed 8801 samples
[INFO: 2819548844.py:  105]: Processed 9001 samples
[INFO: 2819548844.py:  105]: Processed 9201 samples
[INFO: 2819548844.py:  105]: Processed 9401 samples
[INFO: 2819548844.py:  105]: Processed 9601 samples
[INFO: 2819548844.py:  105]: Processed 9801 samples
[INFO: 2819548844.py:  108]: [INFER] infer finished. average time: 0.024146553663819333
[INFO: metrics_util.py:  119]: Saved ./predict_results/level2_top1.json
```

图 3.5: 视频分类标签预测输出

```
do_train(args)
```

```
global step 536, epoch: 2, batch: 157, loss: 0.082322, speed: 11.07 step/s
global step 537, epoch: 2, batch: 158, loss: 0.110537, speed: 12.56 step/s
global step 538, epoch: 2, batch: 159, loss: 0.097912, speed: 12.64 step/s
global step 539, epoch: 2, batch: 160, loss: 0.123946, speed: 12.73 step/s
global step 540, epoch: 2, batch: 161, loss: 0.105484, speed: 12.64 step/s
global step 541, epoch: 2, batch: 162, loss: 0.073712, speed: 12.36 step/s
global step 542, epoch: 2, batch: 163, loss: 0.069594, speed: 11.25 step/s
global step 543, epoch: 2, batch: 164, loss: 0.083069, speed: 10.22 step/s
global step 544, epoch: 2, batch: 165, loss: 0.084396, speed: 13.03 step/s
global step 545, epoch: 2, batch: 166, loss: 0.103032, speed: 10.63 step/s
global step 546, epoch: 2, batch: 167, loss: 0.070644, speed: 12.07 step/s
global step 547, epoch: 2, batch: 168, loss: 0.116652, speed: 11.25 step/s
global step 548, epoch: 2, batch: 169, loss: 0.093518, speed: 12.11 step/s
global step 549, epoch: 2, batch: 170, loss: 0.062314, speed: 12.35 step/s
global step 550, epoch: 2, batch: 171, loss: 0.088355, speed: 9.41 step/s
global step 551, epoch: 2, batch: 172, loss: 0.101376, speed: 11.68 step/s
global step 552, epoch: 2, batch: 173, loss: 0.119314, speed: 9.27 step/s
global step 553, epoch: 2, batch: 174, loss: 0.097010, speed: 11.24 step/s
global step 554, epoch: 2, batch: 175, loss: 0.127405, speed: 12.03 step/s
global step 555, epoch: 2, batch: 176, loss: 0.114858, speed: 12.19 step/s
global step 556, epoch: 2, batch: 177, loss: 0.093776, speed: 11.89 step/s
global step 557, epoch: 2, batch: 178, loss: 0.119245, speed: 11.36 step/s
global step 558, epoch: 2, batch: 179, loss: 0.111839, speed: 11.75 step/s
global step 559, epoch: 2, batch: 180, loss: 0.093600, speed: 11.87 step/s
global step 560, epoch: 2, batch: 181, loss: 0.128591, speed: 11.61 step/s
global step 561, epoch: 2, batch: 182, loss: 0.117599, speed: 12.45 step/s
global step 562, epoch: 2, batch: 183, loss: 0.112667, speed: 11.21 step/s
global step 563, epoch: 2, batch: 184, loss: 0.072423, speed: 11.47 step/s
global step 564, epoch: 2, batch: 185, loss: 0.078790, speed: 10.13 step/s
global step 565, epoch: 2, batch: 186, loss: 0.092070, speed: 11.05 step/s
global step 566, epoch: 2, batch: 187, loss: 0.102003, speed: 11.14 step/s
global step 567, epoch: 2, batch: 188, loss: 0.087637, speed: 12.50 step/s
eval loss: 0.188333, precision: 0.453017, recall: 0.543433, f1: 0.494123
```

图 3.6: 视频语义标签模型训练

```
args = Args()
args.model_name_or_path = 'bert-wwm-ext-chinese'
args.init_checkpoint_path = '/home/aistudio/paddle-video-semantic-tag/data/checkpoints/semantic_tag/model_567.pdparams'
args.max_seq_length = 128
args.batch_size = 32
args.device = 'gpu'

do_predict(args)
```

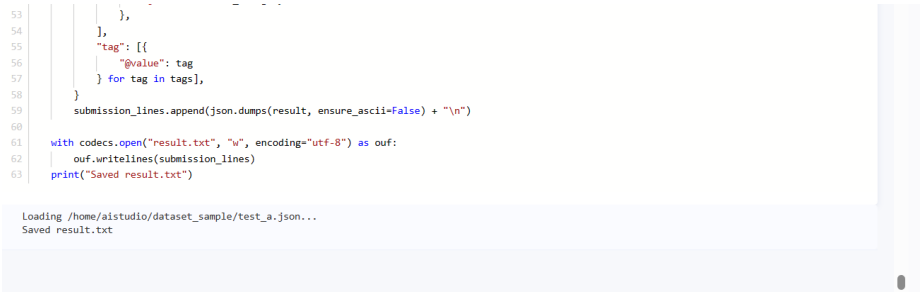
```
[2023-04-22 14:47:23,752] [    INFO] - Found /home/aistudio/.paddlenlp/models/bert-wwm-ext-chinese/bert-wwm-ext-chinese-vocab.txt
[2023-04-22 14:47:23,778] [    INFO] - Already cached /home/aistudio/.paddlenlp/models/bert-wwm-ext-chinese/bert-wwm-ext-chinese.pdparams
311it [00:12, 24.57it/s]

Saved predict_results/ents_results.json
```

图 3.7: 视频语义标签预测输出

3.4 生成结果文件

最终生成结果文件，提交至竞赛系统判定即可。生成结果文件的运行结果如图3.8所示。



```
            },
        ],
        "tag": [{
            "@value": tag
        } for tag in tags],
    }
    submission_lines.append(json.dumps(result, ensure_ascii=False) + "\n")

with codecs.open("result.txt", "w", encoding="utf-8") as ouf:
    ouf.writelines(submission_lines)
print("Saved result.txt")
```

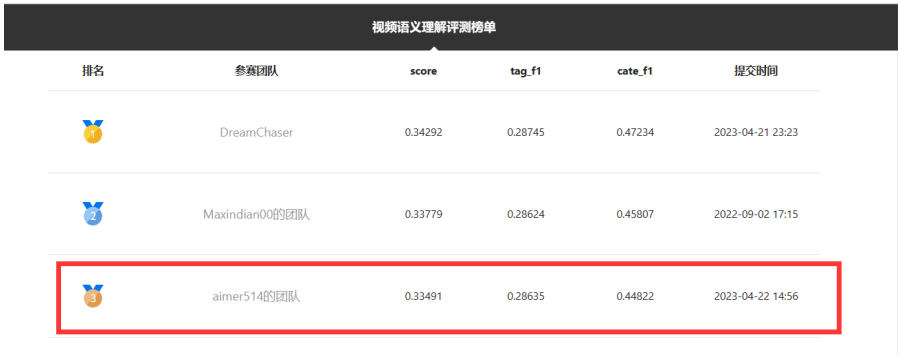
```
Loading /home/aistudio/dataset_sample/test_a.json...
Saved result.txt
```

图 3.8: 生成结果文件并提交

Chapter 4

实验结果

提交结果文件至竞赛系统后，判定结果如下：



视频语义理解评测榜单

排名	参赛团队	score	tag_f1	cate_f1	提交时间
1	DreamChaser	0.34292	0.28745	0.47234	2023-04-21 23:23
2	Maxindian00的团队	0.33779	0.28624	0.45807	2022-09-02 17:15
3	aimer514的团队	0.33491	0.28635	0.44822	2023-04-22 14:56

图 4.1: 系统评定结果

至此，完成了试验的基本要求——【一键运行】跑出结果。


aimer514
分享
UID 3017736
学生 | 北京市，北京 | 已加入AI Studio 159天
暂未完善个人简介
青铜
15
总积分
积分商城
7.8
算力卡
算力卡管理
我的主页
项目(1)
模型(0)
课程(0)
比赛(1)
数据集(0)
我的活动(0)
论坛(0)
当前战力值
1 /排名--
去提升
青铜
还差49战力升至白银
我的权益 4 项
尊享版GPU本周可用时长4.0小时；高级版GPU本周可用时长48.0小时
我的项目空间150个
实物积分兑换
等级专属徽章
成就统计
项目浏览 0 次
项目fork 0 次
数据集引用 0 次
参加课程 0 个
参加比赛 1 个
Fork最多
还没公开过任何项目
引用最多
还没有公开过任何数据集

图 4.2: 个人账号主页

参考文献

[1] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool, "Temporal segment networks: Towards good practices for deep action recognition," in *European conference on computer vision*. Springer, 2016, pp. 20–36.

[2] X. Long, C. Gan, G. De Melo, J. Wu, X. Liu, and S. Wen, "Attention clusters: Purely attention based local feature integration for video classification," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 7834–7843.