Web 信息处理与应用课程大作业实验报告

——基于 Transformer 的弹幕多标签分类

陈洁婷　2020100360

1. 背景及目标

本项目的目标是用爬虫技术构建一个可用于多标签分类的弹幕视频数据集，并运用 Transformer 模型进行弹幕的多标签分类。此外，我们还考虑整理出标签层级关系，使用递归正则化的方法将其融入到分类模型中，促使模型关注标签语义信息。

1.1.项目背景

随着网络通信技术进步，人们每天接收到的信息日益增多，信息的呈现形式也更加丰富，弹幕视频成了广受欢迎的新兴媒体。然而，网络平台上的视频大量存在着无标签、概括不完整、表达不规范、概括错误的问题，这些问题成为了提升用户体验的障碍。在多媒体领域上，图片、文本的多标签分类已经具备了较多工作，但对于视频的多标签分类，数据集的高标注成本限制了相关工作的推进，还有很大的研究空间。此时我们注意到，弹幕有着丰富的信息量、独特的交互性和时序性，可以被应用于视频理解工作，如分类、检索和个性化推荐，提升用户体验。于是我们想到了利用弹幕对视频做多标签分类。

弹幕是人们在网络上观看视频时弹出的评论性字幕。传统播放器的评论系统是独立于播放器之外的，评论的内容大多面向整个视频。与传统评论相比，弹幕将评论即刻显现在视频上，是直接对应视频某段内容的简短评论。弹幕可以以滚动、停留甚至更多特效方法按照时序出现，为视频观看者营造了共同分享、实时互动的氛围，提升了用户体验。自日本的 Niconico 视频网站推出弹幕功能后，国内各大视频平台也逐渐上线，如 Acfun、Bilibili、爱奇艺、腾讯视频等。这些平台产生了大量的弹幕视频数据，内含丰富的多模态信息。除此之外，在用户的高度参与下，网站内的部分视频获得了发布者和观众人为添加的标签，以类似众包的方式自然地解决了多标签视频的标注困难，可以很好地充实多标签数据。同时，人们还可以基于网站本身的分区，构建出层级标签结构，加深我们对标签语义关系的探究。多标签弹幕视频研究对于多媒体领域是新颖且值得探究的工作。

1.2.项目目标

本文收集整理了一个多标签弹幕视频数据集，并利用 Transformer 模型，从最基础的多标签分类任务展开探究。特别地，我们还依照网站的分区及热门搜索信息，构建了视频标签的层级关系结构，并在模型中运用递归正则化的方法融入训练，使模型更好地捕捉语义信息，提升分类效果。

2. 数据集

为支持后续工作，我们制作了一个弹幕视频数据集。其包含了 4951 个视频，总时长 557

小时，有 5,330,393 条弹幕数据。内容涵盖科教、生活、体育、娱乐、影视、文艺等方面，共涉及 252 个标签，平均每个视频标注了 6.35 个标签。除此之外，本文还结合了视频来源网站分区，人工整理出标签层次结构，从而使得标签间的语义关联信息易于利用。除多标签分类外，这份数据集还可以支持弹幕生成、视频摘要、视频描述、精彩片段预测、跨模态检索等多媒体研究。

2.1.数据收集与整理

2.1.1. 数据采集

本文的数据来源于 Bilibili 视频网站[1]，Bilibili 是国内颇受欢迎的弹幕视频网站，在 2019 年第三季度，其活跃用户量达到了 1.28 亿，用户参与程度极高。平台内的多为用户自行制作上传的短片，视频内容多样，适用于本文的任务。本文遵循着内容范围更广、弹幕质量更高的准则，设计了数据采集方案。首先，本文依据 Bilibili 的导航分区构建了一个主题表，再以主题表中的内容为方向，找到其下热度较高的话题标签，收入一个关键词表中。接着，将关键词表中的词作为查询，便可以检索到许多视频。为了保证数据质量，本文将查询结果以弹幕数量从高到低排序，按顺序采集，最终每个主题方向下采集了 1000 个视频。分析表明，这种方式获取到的视频涵盖面广、内容典型、弹幕质量高，数据中的用户的参与互动情况好，具有代表性。

2.1.2. 数据清洗

原始爬取的实验数据存在着较多的噪声和错漏，我们对数据做了清洗、层级标签关系构造、数据划分等处理。由于网络原因，在采集的过程中难免出现损坏的数据，因而本文进行了数据清洗。首先，本文去掉了下载不完全的损坏视频，去掉了弹幕下载不完全或弹幕数量少于 20 的视频。为了便于后续实验，本文使用 ffmpeg 工具[2]每隔 1 秒对视频进行抽帧。音频数据为从视频中提取而得，类似于视频的处理，本文将音频切分为 1s 的小片段。由于一部分视频无法适应上述操作，提取出的图片帧和音频片段不对齐，本文也一并将其过滤。最终，本文总共保留下了 4951 个视频。这些视频均能顺利进行各种数据处理操作。

数据中，视频的标签为用户手工标注，没有语法限制，不够规范。此外，还有部分标签较为少见，缺乏代表性。为此，本文过滤掉了视频数量少于 20 的标签，并将不规范的标签通过字符包含检查合并到了相应的标签中。最后，通过 3 人的人工整理和检查，本文确定了 244 个具有代表性的标签。本文注意到，有 4%的视频原有标签均为不规范标签，过滤后失去了所有的标签。因此，本文使用了 3 人来人为地重新标注。为了减少偶然性，当 3 人中的 2 人都打了同一个标签，本文才将其采纳为视频的标签标注。最终，本文整理得到了一个规范的数据集。

2.1.3. 标签层级关系构建

在处理标签的过程中，本文注意到弹幕视频下的标签是多粒度的。同一视频下，不同粒度的标签显得比较杂乱。比如，在一个猫咪视频下，可能同时出现“暹罗猫”、“猫”、“宠

[1] https://www.bilibili.com/

[2] http://ffmpeg.org/

物”、“生活”几个标签。而传统的多标签分类数据集，大多标签粒度统一，比较规整。本文考虑过将标签粒度筛选到同一粒度，但发现筛选后的标签数量变少，且不能完整地体现视频主题，不符合人们在标注时的真实思维过程。同时，近年的几个中文文本多标签数据集，如 2017 知乎看山杯数据集、2015 LSHTC 竞赛中的数据集，标签粒度情况都与本文类似，但它们同时提供了标签的层次结构关系。受此启发，我们决定保留多粒度这一特性，而通过梳理标签之间的层次结构，来表达不同粒度标签间的“父子关系”和“兄弟关系”，辅助后续任务。

参考 Bilibili 网站的导航分区，本文对应设计了三种标签粒度，粗粒度、中粒度、细粒度，并设 4 名同学人工检查，避免偶然性。中粒度为粗粒度标签的子节点，细粒度为中粒度标签的子节点，同一细粒度标签可以属于多个中粒度标签。如，细粒度标签“经典”，可以属于中粒度标签“电视剧”、“电影”和“动画”。下图为标签结构示意，摘取了粗粒度标签“生活”下的标签结构。

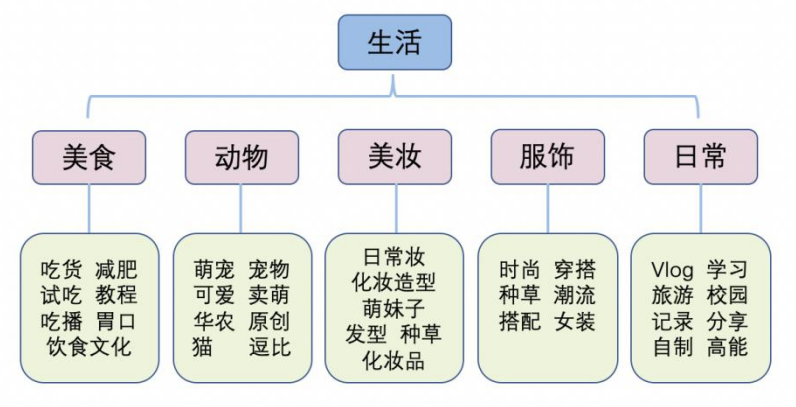

图 1 标签层级结构关系示意图

依据这样的结构，本文利用标签的父子关系，对视频的标签进行了补充。当视频的原标签作为子标签可以对应到唯一的父标签时，就把父标签添加到视频标注中，完善标注。最终，我们确定的标签种类共 252 个，粗粒度 10 个，中粒度 49 个，细粒度 193 个。平均每个视频有 6.35 个标签，平均粗粒度标签为 1.82 个，中粒度标签为 2.08 个，细粒度标签为 2.45 个。

表 1 标签整理结果

	粗粒度	**中粒度**	**细粒度**	**总计**
标签种数	10	49	193	252
每个视频的平均标签数	1.82	2.08	2.45	6.35

2.1.4. 数据划分

本文将 4951 个弹幕视频按照 7：2：1 的比例划分成了训练集、验证集和测试集。为了避免某些标签在验证集和测试集中出现，而训练集中缺失，本文对每个划分上的标签分布情况进行了统计和人工调整，确保不存在零次学习的情况。最终，划分后的弹幕视频数为训练集 3466 个，验证集 990 个，测试集 495 个。

2.2. 数据格式

图 2 数据样例

上图是一个我们数据集中的样例，充分体现了我们数据集多模态、信息丰富、交互性强、具备时序性的特点。我们数据集中包含的所有信息及格式如下：

表 2 数据内容及格式

大类	目名	格式
视频信息	视频标题	字符串
	视频ID	字符串
	弹幕ID	字符串
	视频时长	浮点数
	视频弹幕条数	整型
视频弹幕	弹幕时刻	字典
	每秒弹幕内容	字典及字符串列表
视频内容	视频	MP4
视频标签	视频标签	字典及字符串列表
	标签关系	树、字典

由于本文侧重于探索神经网络模型的使用，我们仅考虑了弹幕这一文本模态的多标签分类，所涉及到的数据用蓝色标出。本文中所用的数据格式整理如下图所示：（为便于处理，token 和 label 的中文均映射转变为了英文与数字符号串）


```
{"doc_label": ["LB6","LB4","LB41","LB30","LB193"],"doc_token": ["135","141","24601"
{"doc_label": ["LB6","LB3","LB8","LB40","LB23","LB25","LB97","LB100"],"doc_token":
{"doc_label": ["LB4","LB6","LB26","LB40","LB112","LB108","LB122","LB110","LB114","
{"doc_label": ["LB3","LB9","LB23","LB55","LB97","LB100"],"doc_token": ["15797","17
{"doc_label": ["LB1","LB14","LB15","LB87"],"doc_token": ["4","1530","451","5477","
{"doc_label": ["LB4","LB7","LB46","LB33","LB125"],"doc_token": ["10253","3508","4"
```


图 3 弹幕视频标签及弹幕内容

Root	LB0	LB1	LB2	LB3	LB4	LB5	LB6
LB0	LB10	LB11	LB12				
LB10	LB59	LB60	LB61	LB62	LB63		
LB11	LB64	LB65	LB66	LB67	LB68	LB69	LB70
LB74							
LB12	LB73	LB75	LB68	LB76	LB77	LB78	LB79
LB1	LB13	LB14	LB15	LB16	LB17	LB18	LB19
LB13	LB77	LB63	LB80	LB81	LB82	LB83	LB84
LB14	LB77	LB63	LB68	LB85	LB86	LB83	
LB15	LB87						
LB17	LB88	LB81					
LB18	LB89						
LB19	LB90	LB91	LB83				
LB20	LB92						
LB2	LB21	LB22					
LB21	LB93	LB74	LB94				
LB22	LB95	LB96					
LB3	LB23	LB24	LB25				
LB23	LB97	LB98	LB99	LB100	LB101	LB102	LB103

图 4 标签层级关系树

2.3.数据特性分析

2.3.1. 弹幕视频本身特性

（1）信息量大。弹幕视频包含了视频（图像）、音频、文本三种模态的信息。由于其用户参与程度高且受欢迎，我们可以方便地采集到内容涵盖面广、弹幕质量高的视频。

（2）交互性强。这是弹幕数据的重要特性，来自于两方面。第一个方面是模态间的交互，例如，用户发布的弹幕文本是对视频画面和音频内容的评价，博主剪辑出的视频画面与音频配乐之间也具备协调关系。第二个方面是模态内的交互，以文本模态为例，用户发送的弹幕除了对视频本身的评价，还有与其他用户所发的弹幕的呼应，类似于群体聊天场景。

（3）具备时序性。弹幕可以看作与视频小段匹配的短文本评论。每一条弹幕都可以定位到具体的发送时间，从而对应上相关的视频内容。这样时序对应关系可以被利用到许多视频理解任务中。

2.3.2. 数据集质量统计分析

经调研，已有多位研究者发布了弹幕数据集，本文选择了质量较高，且与本文的数据最为相似的三个数据集进行比较。

表 3 弹幕视频数据集比较

	视频数	总时长(小时)	弹幕数	平均每秒弹幕数	标签形式	标签种数	视频平均标签数	标签关系	内容	来源
TSCSet	17,870	47,835	32,949,297	0.19	多标签	42	3	未给出	动漫	Bilibili
DR_E	8,156	--	57,176,457	--	单标签	17	1	未给出	影视戏剧	Youku
Livebot	2,361	114	895,929	2.18	单标签	19	1	未给出	多类型	Bilibili
本文	4,951	557	5,330,393	2.66	多标签	252	6.35	给出	多类型	Bilibili

由上表可知，相较于 TSCSet[9]数据集和 DR_E[10]的单一视频类型，Livebot[8]和本文提出的数据集涵盖了多种类型的视频，泛化性更强。从标签的角度，本文的数据集是多标签数据集，252 种标签体现了数据集内容的多样，并且本文数据集还特别地给出了标签间的层次关系。对于本文数据集中的每个视频，被标注的标签数为 6.35，是同为多标签数据集的

TSCSet 的两倍之多，说明标注更加完善，能更完整地体现视频特点。从弹幕的角度，本文数据集的弹幕密度最高，达到 2.66 条每秒，远多于 TSCSet 数据集的 0.19 条每秒，更有利于弹幕研究。因此，尽管本文采集的视频数量还比较少，但在内容范围、弹幕密度和标签标注上的优势，使其具备了独特的价值。

更具体地，本文进一步选择了 2019 年提出的 Livebot 数据集进行对比。从视频和弹幕的内容本身来说，Livebot 与本文的数据更为相似，都是涵盖多种类型的短视频，本文认为从视频和弹幕内容本身来说，这样的比较更适用于数据质量分析。

表 4 视频采集主题词

主题	Livebot	本文数据集
体育竞技	篮球、足球、健身、NBA	篮球、足球、跳水、游泳、健身、电子竞技、运动
影视	动漫	动漫、电视剧、电影、剪辑、卡通
科教	--	探险、自然、演讲、公开课、人文
文艺	舞蹈、绘画、折纸、歌唱、钢琴	舞蹈、绘画、音乐、风俗、中国风
生活	宠物、食物、交通、美妆	宠物、食物、旅行、幽默、生活方式
娱乐	魔术	时尚、明星、综艺

上表为两个数据集采集过程中所使用的的查询词，可以体现视频的内容。本文将查询词归分为六个主题：体育与游戏、影视、科教、文化艺术、生活、娱乐。本文的查询词均匀地分布在了六个主题内，且每个主题下的查询词大多更为丰富。这样的情况说明，尽管两个数据集都属于多类型短片内容，但本文数据的视频类型更加丰富，泛化能力更强。

表 5 Livebot 数据集与本文提出数据集统计对比

	Livebot	本文数据集
视频数	2,361	4,951
总时长(小时)	114	557
弹幕数	895,929	5,330,393
弹幕平均字数	5.42	5.39
弹幕平均词数	9.08	9.14
每秒平均弹幕数	2.2	2.7
视频平均弹幕数	380	1,077
有弹幕时长(秒)	252,382	1,558,357
有弹幕时长占比	61%	78%
弹幕>500 条视频比例	36%	87%

由表 4 可知，从视频质量上来看，本文的数据集所含视频数约为 Livebot 数据集的 2 倍，总时长约为其 5 倍，数据规模更大。此外，可从图 3 中看出，本文的视频以 3：3：1 的比例分布在了 0-5 分钟，5-10 分钟和 10-30 分钟上，分布较协调，不存在极端情况。

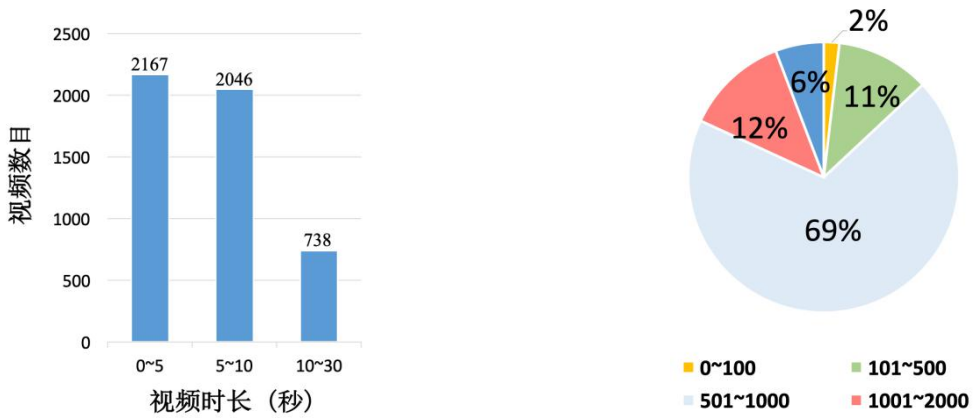

图 5 视频时长及弹幕数量分布图

从弹幕质量上来看，在本文的数据集中平均每个视频拥有的弹幕数更多（1077 vs 380），每秒平均弹幕数也更高（2.7 vs 2.2）。并且，本文的视频存在弹幕的时长占视频总时长的比例更大（78% vs 61%）。由图 4 可知，在本文的数据集中，98%的视频包含 100 条以上的弹幕，87%的视频有 500 条以上的弹幕，而 Livebot 数据集中含有 500 条弹幕以上的视频比例仅为 36%。上述对比充分地体现了本文弹幕数据质量的优势。本文还注意到，两个数据集中每条弹幕的平均字词数相似，这恰好也说明了两个数据集中的文本形式为相似的短文本，具有可比性。

综合以上分析，相较之前发布的数据集，本文的弹幕视频数据集具有内容多样、弹幕质量高、标签丰富且具备标签结构关系的优点。对于相关的研究工作，能起到较好的支持作用。

3. 模型

本文选择了 Transformer 作为弹幕多标签分类的模型，给每个视频弹幕中的词语分配了不同的注意力，来获得弹幕文本更好的表示。同时，我们利用递归正则化方法，将标签的层义关系融入了训练过程，使得模型能考虑到标签语义信息。

3.1. 基于 Transformer 的多标签分类模型

Transformer[2]是近期广受瞩目的模型之一，在 NLP 领域表现卓越。分析本文任务，我们认为并非弹幕中的每个词语对于视频的分类都有作用，而 Transformer 恰好能让模型有的放矢地分配注意力。此外，在我过去四年的本科学习中，还没有实践过这个模型，我希望能借这次机会，深入理解老师上课讲的 Transformer 相关内容。因此，我选择 Transformer 来实现本次任务。

下图为 Transformer 提出者的论文《Attention is all you need》中的模型图。模型大体分为 encoder (左) 和 decoder (右) 两个模块。相比于常用于序列处理的 RNN，Transformer 可以实现并行计算，且其中的自注意力机制（Self-attention，即 Q、K、V 获取于相同输入来源）能够把输入序列上不同位置的信息有的放矢地联系起来，计算出整条序列的某种表达。目前自注意力机制主要应用于阅读理解、提取摘要、文本推论等领域，表现亮眼。

下面分析其具体的实现原理。我们向模型的左半边输入一个要处理的源序列，进行词嵌入 Input Embedding 处理。考虑到注意力模型无法表征序列的时序信息，我们加入了一个位置编码操作 Positional Encoding，以实现 RNN 那样包含序列信息的效果。将位置编码和词嵌入编码加起来，即可得到 encoder 的输入 x。进入 encoder 后，x 经由三个不同的矩阵相乘得到 Q、K、V，进行了多个 self-attention 操作，即图中的 Multi-head Attention 部分。接着再运用残差思想，进行残差连接 ADD 并归一化，此后再经过 Feed Forward 层的变形整理，再过一层残差连接 ADD 和归一化，即可完成一个单元操作。这样的单元操作可以进行 N 次（图最左），这就构成了 Transformer 的 encoder。encoder 的输出为经注意力分配调整，并考虑了位置信息后的序列特征编码。

Transformer 原本用于机器翻译这类序列生成任务，右边的 Decoder 被设计为既考虑已生成的序列信息，也考虑到源序列的信息，充分模拟了人类思考过程。这里的生成一个词一个词地进行，右下的 Output Embedding 为 decoder 的输入之一，它代表着已生成的序列。同样的，我们对这个已生成的序列进行词嵌入编码和位置编码，综合后自己派生出 Q、K、V，做自注意力。这里的自注意力用到了 Masked，意在阻止模型将注意力放在未生成的部分。对已生成的序列做好注意力编码后，我们将其作为 Q，而把 encoder 的输出作为 K 和 V，再进行自注意力和前向网络等操作。最终得到的向量经过 linear 层和 softmax 层，就可以作为预测概率。选取最大概率位置对应的 token，即可得到该步的生成结果。

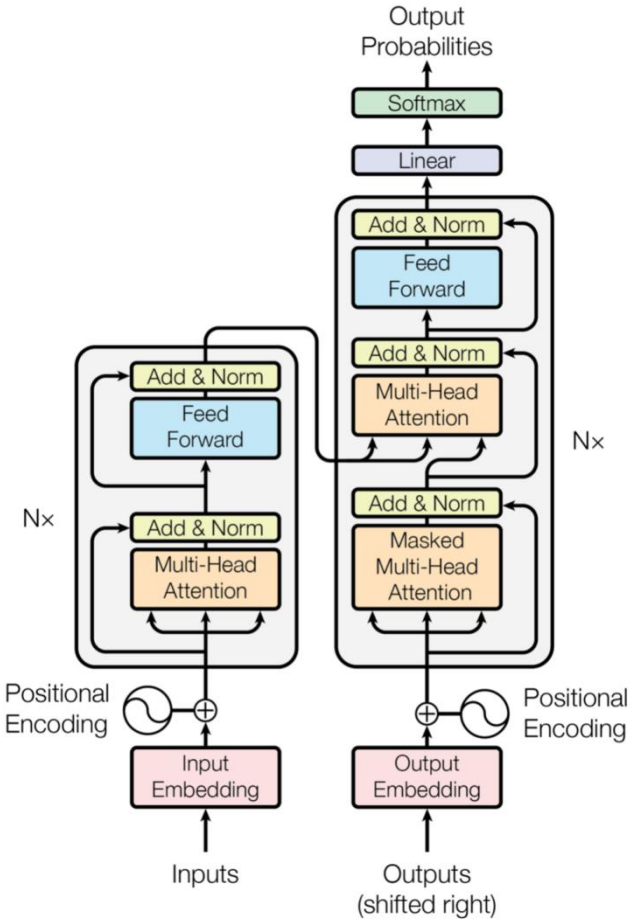

图 6 Transformer 模型结构

下面具体解释 Multi-Head Attention（多头注意力机制），这里的多头，意思为多种注意方式。比如，我们在观察一幅画时，不同的人有不同的关注角度，如色彩、形状、纹理。多头注意力则是模拟了这些不同类型的注意力，采用多套矩阵 W，来分化 Q、K、V。以$\mathcal{W}_0$为例，输入句子被编码为[X, R]，每个 token 都分别由$\mathcal{W}_0^q$, $\mathcal{W}_0^k$, $\mathcal{W}_0^v$分化成 Q，K，V 当我们以第一个 token X 来作为 Q 时，计算其与其他 token 对应的 K 的相似度（如内积法），即可得到相应的权重。经过归一化等操作，可以得到 Q 对于其他所有 token 的注意力权重。将这些权重与相应的 V 相乘，便可以得到一个带注意力的新序列编码。

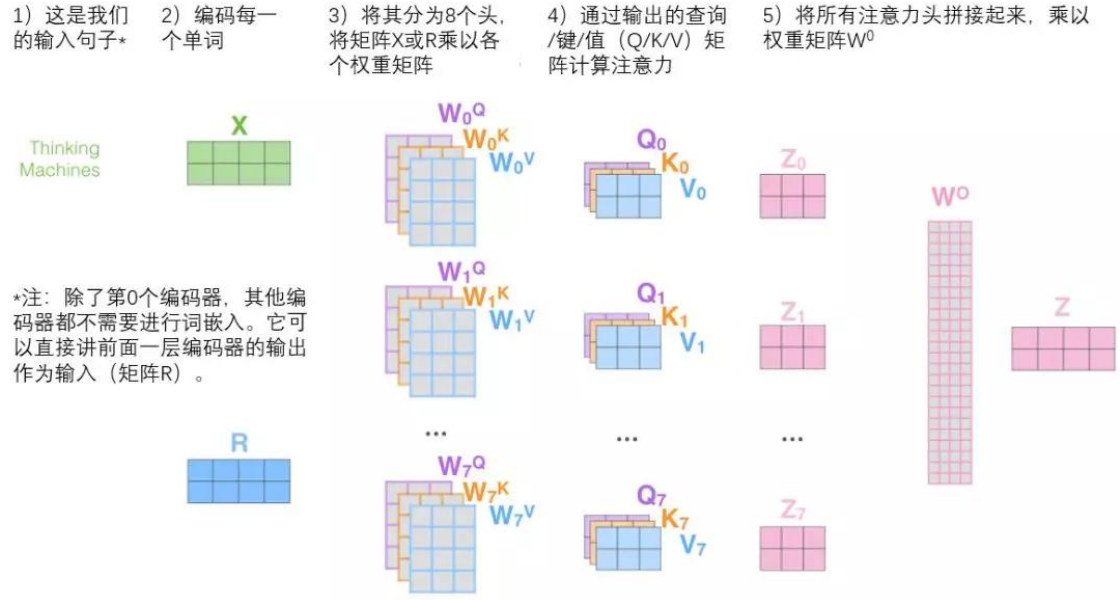

图 7 Multi-head Attention 图解[3]

在本文的任务中，我们只需要完成分类任务，而不需要生成，因此，我们在代码中只用到了 encoder 的部分。

3.2. 利用递归正则化融入标签关系

由于我们的数据集还提供了标签层级关系，为更好地融合语义信息，我们借鉴腾讯 AI 实验室的做法[4]，将这个关系通过递归正则化的方法运用到了模型中。通常来说，正则化在机器学习中起到的是增加约束，降低模型复杂度，避免过拟合的作用。这里运用了 2013 年的论文[1]和 2018 年的论文[3]中的做法，给模型训练的 loss 函数增加了一个正则项。这个正则项等于层次结构中所有子节点参数到父节点参数的欧几里得距离之和。如此一来，在优化过程中，我们就可以鼓励层次结构中邻近的标签共享类似的模型参数，从而降低模型复杂度，一定程度上防止过拟合，同时也可以将类标签之间的层次依赖关系合并到参数的正则化结构中。

[3] https://blog.csdn.net/Wisimer/article/details/104415321/

[4] https://github.com/Tencent/NeuralNLP-NeuralClassifier

4. 实验

4.1.输入输出

4.1.1. 输入

（1）每个视频弹幕文本。对于每个视频，我们的弹幕经过结巴分词，并过滤掉 tf-idf 值在 30000 名以后的词汇, 按照时间顺序拼接成一篇弹幕文本。每个视频的弹幕文本对齐至 6000 词，多的截掉，少的在编码时用 0 补足。

（2）每个视频的标签。

（3）标签的层级关系。我们将标签的层级关系用树表示，存放在 TXT 文档中，文档中的每一行的格式为“父节点\t 子节点 1\t 子节点 2\t……子节点 n\n”。

4.1.2. 输出

输出为对每个视频的标签预测置信度向量。

4.2.评价指标

经调研，本文共选用了三个经典多标签分类指标作为评价指标[4]。

GAP：全局平均精度。由所有视频预测的所有标签在不同置信度阈值下的精确率和召回率计算而得。不仅考虑到预测结果的命中情况，还能反映每个预测的置信度。该指标在 2018 年 Youtube-8M 第二次挑战赛上被官方提出，后续沿用甚多，我们也将其作为本文实验的金标准。

MAP：平均平均精度。先计算所有视频在每类标签下分别的精确率和召回率，再采用类似于 GAP 的方法，算出能反映每类的命中及预测置信度的值。这样的方法考虑到了因数据标签分布不均衡造成的误差，将所有类别一视同仁，分别计算，最后对所有标签类别求和取平均，得到整体情况。

Hit@1：预测的结果中至少有一个命中答案的概率。

4.3.结果分析

	GAP(tst)	Hit@1	MAP
Transformer	0.371	0.691	0.251
Star_Transformer_Hierarch	0.383	0.735	0.264
Star_Transformer_Flat	0.386	0.709	0.306
NeXtVLAD	0.554	0.797	0.429
DbofModel	0.537	0.756	0.418

图 8 实验结果

除了第一行我们的 Transformer 模型，我们还用到了其它 3 中模型进行尝试。其中 NeXtVLAD[5]模型和 Dbof 模型[6]是我在毕业设计时做的实践，用 tensorflow 实现。这两个模型都属于基于聚类的模型，能将视频每一秒的文本特征聚合成整个弹幕文本的特征，再接入 logistic 分类器进行分类。可以看出，本文中尝试的 Transformer 作为一个很基础的模型，表现并没有其它设计得更精巧的模型好。

表中的 Star_Transformer[7]于 2019 年被提出，用星型拓扑结构代替了注意力所用的全连通结构，降低了模型复杂度，使得我们在训练时可以设定更大的 batch_size 和某些 tensor 维度。我们在 Star_Transformer 上进行了递归正则化设计的擦除实验。hiearch 表示利用了递归正则化进行标签层级语义关系的融入，而 flat 则表示直接处理标签，不考虑标签层级关系。可以看出，递归正则化引入的标签关系没有给 GAP 和 MAP 指标带来提升，反而导致了些许降低，但在 Hit@1 指标上，可以看到 2.6 个百分点的明显提升，说明对于标签关系的利用，我们仍有探究的必要。

4.4. 实验环境与开销

本文中的实验在 Linux 系统上进行，使用了一块 GeForce GTX 1080 Ti GPU，搭配的环境为 CUDA 9.0.176, CUDNN 7.6。编译环境为 Python 3.5.2，使用了 Pytorch 工具包。在模型训练中，经过简单的实验比对，选用的优化器均为 Adam 优化器，学习率下降方式均为指数衰减法，损失计算均采用交叉熵和递归正则化。在训练过程中，每个 epoch 耗时约 150 秒，训练时占用 GPU memory 约为，模型的总参数量为 1166716，训练部分的模型参数量为 974684。

5. 总结

通过一学期课程的学习，我学习到了当下流行的多个神经网络模型，并且了解了相关工具。此外，在过去的实践中，我还没有尝试过 Transformer，因此在本次大作业选择了这个模型进行学习。通过这次实践，我更深刻地理解了 Transformer 模型，并收获了实际的代码经验，为将来研究生阶段的学习打下了基础。我对自己进行的总结和反思如下。

5.1. 收获

（1）学习了 CNN、RNN、GAN、GNN、Transformer 等神经网络模型，对其背景和原理有了更深刻的认知。

（2）在本次大作业中使用了 Transformer，收获了实践经验和更好的理解。

（3）学会运用 pytorch 处理数据、搭建模型。

（4）学会运用爬虫和各种数据标注、数据采集网站获取数据。

5.2. 不足

（1）本次大作业中采用的 Transformer 模型的表现没有超过我之前尝试的其他模型。

（2）本次大作业未利用上所采集数据集的视频信息，仅仅在文本上进行了分类探究，未来应该进行多模态的尝试

参考文献

[1] Gopal, S., & Yang, Y. (2013, August). Recursive regularization for large-scale classification with hierarchical and graphical dependencies. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 257-265).

[2] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. In *Advances in neural information processing systems* (pp. 5998-6008).

[3] Peng, H., Li, J., He, Y., Liu, Y., Bao, M., Wang, L., ... & Yang, Q. (2018, April). Large-scale hierarchical text classification with recursively regularized deep graph-cnn. In *Proceedings of the 2018 World Wide Web Conference* (pp. 1063-1072).

[4] Lee J, Natsev A, Reade W, et al. The 2nd YouTube-8M Large-Scale Video Understanding Challenge[C]. european conference on computer vision, 2018: 193-205.

[5] Lin R, Xiao J, Fan J. Nextvlad: An efficient neural network to aggregate frame-level features for large-scale video classification[C]//Proceedings of the European Conference on Computer Vision (ECCV). 2018: 0-0.

[6] Abu-El-Haija S, Kothari N, Lee J, et al. Youtube-8m: A large-scale video classification benchmark[J]. arXiv preprint arXiv:1609.08675, 2016.

[7] Guo, Q., Qiu, X., Liu, P., Shao, Y., Xue, X., & Zhang, Z. (2019). Star-transformer. *arXiv preprint arXiv:1902.09113*.

[8] Ma S, Cui L, Dai D, et al. Livebot: Generating live video comments based on visual and textual contexts[C]//Proceedings of the AAAI Conference on Artificial Intelligence. 2019, 33: 6810-6817.

[9] Liao Z, Xian Y, Yang X, et al. TSCSet: A crowdsourced time-sync comment dataset for exploration of user experience improvement[C]//23rd International Conference on Intelligent User Interfaces. 2018: 641-652.

[10] Bai Q, Hu Q V, Ge L, et al. Stories That Big Danmaku Data Can Tell as a New Media[J]. IEEE Access, 2019, 7: 53509-53519.