

大作业结题报告

课题名称： 高级软件工程

报告题目**：** AI对联机器人？

学生姓名： 欧琳琳 赵世博 苏韫月

班 级 ： 710

指导教师： 罗铁坚

完成日期： 2020年12月11日

二0二0年十二月十一日

**1项目背景**

**1.1项目意义**

对联作为我们民族独有的一种文学艺术形式，千百年来，一直以“对仗工整、韵律协调、短小精粹、喜庆实用”的明显特点为大众所喜爱，是民间婚丧嫁娶、春节祈福、店铺开业、建筑落成、风景名胜等重要活动不可或缺的文化标志。但是，随着时代的发展，对联也像京剧等国粹艺术一样，在年轻一代中明显普及不够，对联的创作队伍、爱好者队伍已出现严重的年龄老化问题。

为了传承这一民族文化，结合互联网和软件行业对当代人们渗透深入的便利，并利用他们对人工智能强烈的好奇心，通过开发在线的人工智能交互软件来吸引人们接触对联，使其在交互的过程中发现对对联的规律和乐趣，从而启蒙他们产生继续钻研对联的兴趣。

**1.2研究现状**

目前市场上相关的实现对联生成的软件，多是出于娱乐目的，由人给出上联，机器生成下联。例如“王斌给您对对联”、“微软对联”等网页版在线对对联应用。但是以上两款应用是静态的、非对话式的，这样的模式缺少跟用户的良性互动，用户既无法产生类似与人对对联时的交互体感，也无法看到历史的对联情况。此外，如公众号“AINLP”提供的对对联服务，虽然能够以对话的形式进行交互，但是它同前两者一样，无法实现对机器生成的对联进行反馈。

以上应用虽然都实现了对联生成的功能，但却无法让用户产生参与感，以至于用户只有尝鲜的体验，却缺乏持续长期使用的动力，对用户的影响较为微弱，难以使用户产生继续钻研对联的兴趣。

**1.3项目方向**

根据市场上已有的产品存在的缺陷和不足，我们开发了一款名为“AI对对联”的软件，平台采用在线Web端。用户只需登录网页即可尝试与AI进行对对联交互，避免下载臃肿的APP应用。此外用户还能参与到AI调教的过程中去，亲自教导AI生成更好的对联。

**2需求分析**

**2.1用户故事**

**2.1.1第一类目标用户——入门级/青铜**

这类用户看过少量对联，能背出一些简单的对联，知道对联的上下联字数相等这类简单规则。但让其对对联则有难度，其给出的对联无法满足上下联平仄对仗、词对仗等一般规则，甚至可能分不清上下联。

**2.2.1第二类目标用户——普通级/白银-铂金**

这类用户知道上下联字数相等、语言结构相同、相应词对仗、平仄对仗、上仄下平等一般规则，且能根据这些规则给出合理的对联。但给出的对联多为词义相对，并不出彩。比如：“春暖观鱼跃 秋高听鹿鸣”，“春暖”对“秋高”，“观”对“听”，“鱼跃”对“鹿鸣”，都算是词意上相反或相近的配对，算是常规操作。

这类用户本身就对对联有所了解，但并不深入，了解主要来自于学校课程普及之类，如果未接触本产品，今后不太可能会继续钻研对联。

**2.3.1第三类目标用户——高手级/钻石及以上**

这类用户对对联就较为精通了，可以称为“联迷”。他们对联有着更为深厚的文化积淀和对联经验，对联时能引经据典、追求意境；另外也较热衷于对联游戏，会在对联的一般规则上增加特殊的规则来提高难度和趣味性，包括限制对联中的字形、字音、字序等，例如叠字相对“处处飞花飞处处 潺潺碧水碧潺潺”、叠韵相对“妙舞翩迁，风月无价 艳歌婉转，管弦奇鸣”、中心对称“上海自来水来自上海 香山碧云寺云碧香山”等等。

对于这类用户，因为本身就对对联有着较为深入的研究，所以我们的软件并不会给他们带去启蒙等效果。但是我们仍然希望能留住这些用户，通过他们的专业知识来帮助我们的软件更新升级。

**2.2具体功能需求**

针对以上三类不同用户，我们做出以下分析：第一二类用户对对联这件事可能并不感兴趣，所以我们的软件要能吸引他们参与体验对对联，对应提供的功能是有趣的动态交互界面和AI生成对联。而对于第三类用户已经对对联十分熟悉，AI生成的对联很可能无法满足他们的追求，他们可能会想要调教AI，使其生成更好的对联来获得成就感，所以相应的，我们开放了语料库修改这一功能，让用户参与到AI的调优过程中。

根据以上分析，最终敲定的功能需求为以下几部分：

**2.2.1前端交互模块**

前端是人机交互的基础，为用户提供了输入框等功能来与AI程序进行对对联活动。此外其美感也影响着用户的使用体验，因此提供了可爱的动态看板娘，引入虚拟角色，来增加对用户的吸引度。

**2.2.2** **用户信息管理模块**

鉴于我们的应用定位不止娱乐用户，更是要增加用户粘性，使其能中长期使用，所以用户信息是需要被记录和管理的，具体包括用户的历史对话记录，用户权限等级，以及用户对语料库的修改记录等信息。

**2.2.3 基础对话问答模块**

考虑到应用的对话交互性，光对用户的对对联输入产生反映显得单调死板，于是加入日常的基础对话问答模块，使应用能简单回应用户的聊天式输入，如“你好”等，并引导初次使用的用户成功进入对联对话模式。

**2.2.4 AI生成对联模块**

应用的主要卖点之一即由AI动态生成对联，而这个模块的任务也只有一个：实现接收上联，生成下联的功能服务。本模块包括AI模型的训练和测试，并使用训练得到的模型提供AI生成服务。

**2.2.5用户改进语料库模块**

该模块允许用户对语料库进行建设性修改：用户通过对对话过程中不满意的对联提出修改意见，即给出更合理的下联，扩充或更新训练集，以此帮助AI对联生成模块调优。考虑到用户水平参差不齐，且不排除恶意污染数据集的可能，我们对用户进行权限限制，满足条件者才可参与修改语料库，且修改结果需要经过指标评测以及人工评审，才可纳入数据集。

3系统设计与实现

项目架构整体分为上层应用层和基础服务层两部分，上层应用层主要用于实现AI对联机器人？的各个功能模块，基础服务层则是为上层应用提供所需的底层服务和开源架构。总体架构如下图所示：

【图】

3.1基础服务层

基础服务务层主要分为服务、开源框架和操作系统三个部分。

**3.1.1服务部分**

不太懂

**3.1.2开源框架部分**

在构建开发交互式网站的时候，我们选择了Django作为主要的开发框架。Django和ruby都有着完备的MVC开发测试框架，但Django的内置管理面板能大大简化后端用户界面管理的开发，具有开箱即用的权限和身份验证模块，可以为我们减少很多工作量。此外，相比起ruby，基于python的Django开源生态系统更广泛，工具和库更多，调试阅读等对于我们来说也更容易上手。

前端NodeJS的选择原因讲不讲？

在实现AI基础对话和AI生成对联时，我们选择了TensorFlow和Pytorch两个开源框架，其中AI基础对话依赖于TensorFlow，而AI生成对联由Pytorch实现。之所以选择不同的开发框架，仅仅因为这两部分由不同的人完成，事先未曾协调。此外我们也不希望将两部分混为一谈，而是将他们抽出软件框架作为独立的API服务，大有模拟服务提供商的意思。如此保证了我们的应用的可扩展性。

3.2上层应用层

上层应用部分采用前后端分离的设计，前端负责与用户的交互，后端负责具

体的服务。前端有：交互对话界面，用户信息展示界面，语料展示修改界面。后端有基础对话问答模块，AI生成对联模块，语料审核界面，

【架构图】

**3.2.1前端页面**

【总体的思维导图】

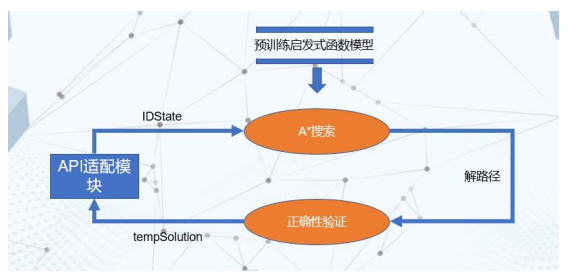
【细节图】

**3.2.2接口设计**

**3.2.3用户信息管理模块**

**3.2.4 基础对话问答模块**

**3.2.5 AI生成对联模块**

流程图之类的

对联生成模块采用序列标注思想，基于tensformer模型进行下联的生成，BLEU作为训练的评价指标。

具体运行如下：

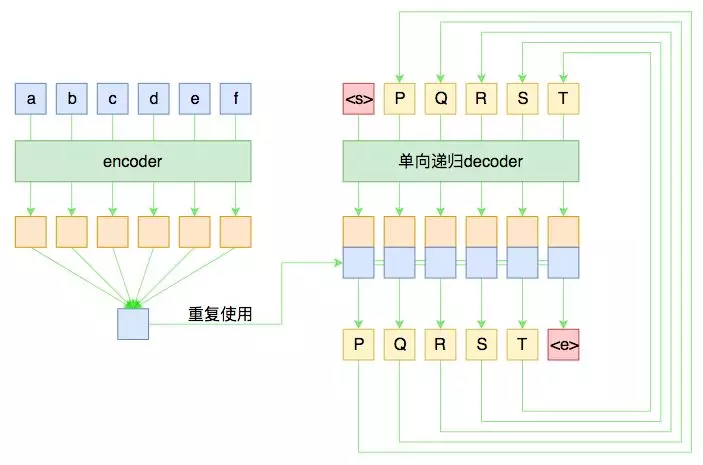
1. 初始阶段，通过采集的对联语料在Google Colab上进行模型的预训练，得到可用的模型。
2. 将模型部署到Vultr云服务器上，在服务器上运行对联生成模型，开放Flask监听接口，接受POST/GET传送过来的上联。
3. 将接收到的上联输入预训练完成的模型中，生成下联，并通过HTTP传输协议返回下联文本串。
4. 空闲时期，Vultr云服务器每隔1个月，与华为云上部署的MySQL服务进行远程通讯，将新的语料拉取到Vultr云服务器上并自动执行训练程序，保存最新的模型。
5. 监听程序会自行采用最新训练的模型进行对联生成服务。

基于Transformer的序列生成Seq2Seq原理

Seq2Seq，就是指一般的序列到序列的转换任务，比如机器翻译、自动文摘等等，这种任务的特点是输入序列和输出序列是不对齐的，如果对齐的话，那么我们称之为序列标注，这就比Seq2Seq简单很多了。所以尽管序列标注任务也可以理解为序列到序列的转换，但我们在谈到 Seq2Seq 时，一般不包含序列标注。

**基本结构：**

假如原句子为 X=(a,b,c,d,e,f)，目标输出为 Y=(P,Q,R,S,T)，那么一个基本的 Seq2Seq 就如下图所示。



左边是对输入的 encoder，它负责把输入（可能是变长的）编码为一个固定大小的向量，这个固定大小的向量，理论上就包含了输入句子的全部信息。

而 decoder 负责将刚才我们编码出来的向量解码为我们期望的输出。与 encoder 不同，我们在图上强调 decoder 是“单向递归”的，因为解码过程是递归进行的，具体流程为：

1. 所有输出端，都以一个通用的<start>标记开头，以<end>标记结尾，这两个标记也视为一个词/字；

2. 将<start>输入 decoder，然后得到隐藏层向量，将这个向量与 encoder 的输出混合，然后送入一个分类器，分类器的结果应当输出 P；

3. 将 P 输入 decoder，得到新的隐藏层向量，再次与 encoder 的输出混合，送入分类器，分类器应输出 Q；

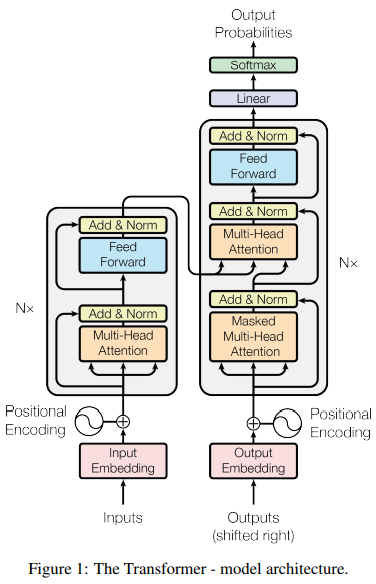
4. 依此递归，直到分类器的结果输出<end>。

这就是一个基本的 Seq2Seq 模型的解码过程，在解码的过程中，将每步的解码结果送入到下一步中去，直到输出<end>位置。

Transformer架构完全舍弃了深度学习里经典的RNN和CNN结构，而只用注意力机制搭建。Transformer 不光像 CNN 一样克服了 RNN 最被人诟病的训练慢问题，利用self-attention 机制实现快速并行，层数还可以增加到比较深，充分发掘DNN模型的特性，提升模型准确率。

这里我们用3层Transformer单元，里面的Self-Attention的head为8，隐单元数512个。模型采用 pytorch框架，GPU 则用Google colab的 Tesla T4。

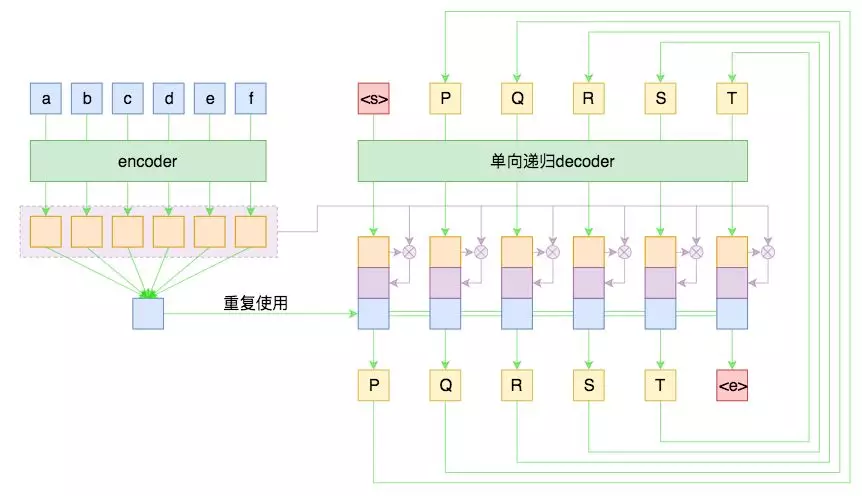
预处理，vocabs



这是transformer的架构（来源于attention is all you need），由编码器（左边）和解码器（右边）组合而成。可以看到编码器使用的是不加Masked的Multi-Head Attention，这样它就能看到输入句子的所有信息。

而解码器要加Masked的原因和seq2seq的原理是类似的。由于训练的时候我们有标注数据对，因此我们能提前预知 decoder 每一步的输入和输出，因此整个结果实际上是“输入 X 和 Y，预测 Y[1:]，即将目标 Y 错开一位来训练，因此 decoder 在执行每一步时，不能提前使用后面步的输入，所以就要把后面部分（即右边部分）的内容mask掉，再每步进行shifted right，逐渐解锁。

基于transformer的seq2seq结构可以表示为下图这种形式：



图源[https://mp.weixin.qq.com/s?\_\_biz=MzIwMTc4ODE0Mw==&mid=2247491314&idx=1&sn=3e22d4a6d732b0877fdc567d2bce1076&chksm=96e9c172a19e48646005da05e143751aa9012c141dd1cf9846a2b418cbf854c7d343013105a1&scene=21#wechat\_redirect]

**3.2.6用户改进语料库模块**

4预期目标及测试

AI对联机器人项目着重实现人机交互的对联对答，

4.1用户视角可视化调控

4.2对联状态调控

4.3对联xxx

5遇到的问题及解决方案

6项目部署

6.1项目进度

6.2项目发布