摘要

古典诗歌是我国宝贵的文化财富，自《击壤》以来，诗歌成为了中华民族延续数千年的表达方式。人工智能是否可以像人类一样写诗？这个课题是对人工智能尝试人类艺术创作的一种重要探索。

中国古典诗歌以唐初定型的格律诗（即近体诗）为主，且现有的算力及算法只支持显式规则较多的格律诗文本生成，所以本文以格律诗文本的生成为研究对象，主要尝试了LSTM，GPT，BERT等深度学习算法，使用迁移学习，生成格律诗歌文本乃至特定风格的诗歌文本，如“江西诗派风格”，“艳体诗风格”等。

为了判断生成的诗歌文本是何种风格，本文利用大规模预训练词向量与LSTM等算法，尝试解决诗歌风格判定，关键字词提取等基础问题，创新性地将深度学习引入到计量风格学这个领域，并取得了比目前分析方法更好的效果。以唐诗诗风识别与艳体诗诗风识别为例，对江西诗派，明七子，同光诗派，王彦泓等风格鲜明的诗人进行测试，取得了与文学史判断相吻合的结果。

为了测试本文构造的模型的效果，我们选取一些古人作品，与模型作品混合，出了一套AI诗歌图灵测试题，联合诗词圈的一些创作者与爱好者进行了评测。参与人数超过五百，最终结果显示AI的作品与人类的作品已无明显差别，这表明本文的模型已经做到了一般的诗词爱好者都无法分辨的地步。于是为了惠及数量庞大的诗歌爱好者群体，本文的模型已上传到网站上供大家使用[1]，希望此模型的作品能给人带来创作上的启发。

关键词 深度学习 文本生成 预训练 迁移学习 LSTM GPT 风格计算 格律诗

第一章 绪论

1.1 研究背景与意义

提到中国古典诗词，许多人囿于教科书的印象，且被王国维“一代又一代之文学”的说法且误导，认为“唐之诗，宋之词，元之曲，皆所谓一代之文学，而后世莫能继焉者也”[2]，似乎诗词这种文体，至唐宋而极盛，而明清之际便已衰亡，让位于古白话小说，当今之时更是早已死亡。所以研究如何生成这么一种死亡的文体有什么意义？

然而与平常人的认知不同，若以诗歌数量，诗人数量来作为判断诗词是否兴盛的标准，那么可以明显看出，诗词随着时间的发展愈发兴盛。根据搜韵网[3]的数据显示，现存的先唐诗人约950位，诗作不到万首，唐朝诗人共3369 位，现存诗词作品共54685首。宋朝诗人9647 位，现存诗词作品共280971首，元明清诗歌数据囿于现有研究条件，搜集到的百不存一。复旦章培恒教授致力于编纂《全明诗》，历数十年而未完工，篇幅据估计至少是全宋诗的2.5倍[4]，即70万首以上。而清朝由于人口爆炸以及教育的普及，诗人数量与诗词作品数量较前代有巨大的提升，应有八百万首以上[5]。

时代	作家数量	作品数量
先唐	约950	<10000
唐	3369	54685
宋	9647	280971
明	不详	>700,000
清	不详	>8,000,000
当代	>5,000,000	仅2020一年约50,000,000

到了当代社会，尽管诗词创作已经不再有古代的地位，让位于小说散文等文体，但由于受教育人口急剧增长，获取信息难度降低，发表渠道多样，诗词作品数量多到无法估计。据中华诗词研究院发布的报告，2020年我国能统计到的诗词创作人数已达500万，仅2020一年所创作的诗词作品数量保守估计在5000万以上[6]。

从诗人数量与诗歌作品数量来看，当今社会的诗词创作与古代相比，处于一个极度繁盛的状态，只是光芒被小说等文体所掩盖。为了接续数千年来的诗骚传统，也为了帮助当今数百万诗词创作者进行更好地创作，探索如何使用深度学习模型生成近体诗是一个很有意义的研究。

在探索格律诗生成的过程中，为了测试特定风格诗歌生成的效果，我查阅了大量计量风格学的论文，并且发现这些基于简单统计学的手段在面对诗歌这种高信息量文本时显得力有不逮，而在古典诗歌研究领域中，分析某位作家的诗风是一个经久不衰的重要课题。然而历代诗话与各类研究论文中，评判某位诗人的诗风——如“宗唐”或“宗宋”，多凭读者自身的阅读感受，或摘出一些风格明显的句子条分缕析。于是本文基于以四库全书为训练语料的大规模预训练词向量，使用LSTM搭建了算法模型，提出了一种用来判定不同诗歌风格的模型。并在艳体诗识别与唐代诗风识别等问题上进行应用，取得了较好的效果。该模型对所测试的诗歌作品做出的判定结果与历代评论家给出的定论基本吻合，可与文学史相参照。

1.2 相关研究现状

1.2.1 诗歌文本生成研究现状

作为人工智能的一个长期关注点，诗歌自动生成的研究可以追溯到几十年前。这一领域的第一步是基于规则和模板[7]。自20世纪90年代以来，统计机器学习方法被用来生成诗歌，如遗传算法[8]和统计机器翻译方法[9]

深度学习兴起之后，在诗歌文本生成问题上取得了巨大的优势，现有的对中文格律诗文本生成的尝试，较为成功的是清华的九歌AI[10]以及某位浙大博士开发的诗三百AI[11]，二者都基于深度学习算法。

九歌由清华大学自然语言处理与社会人文计算实验室开发，主要采用GRU算法与Sequence-to-Sequence模型[12]

九歌模型将输入的主题词明确、独立地保存在内存中，起到“主要消息”的作用。在生成每一行诗时，模型从内存中读取相关的信息（主题词和历史生成内容），以已经生成的内容和到目前在表达的主题词为根据来指导当前这行诗的生成。且模型需要在有限的内存插槽中保留部分信息而非完整信息。这种动态的阅读和写作方式赋予了模型在生成过程中关注相关信息和忽略干扰的能力，从而在很大程度上提高了连贯性。此外，九歌还设计了一个特殊的体裁嵌入来控制每个角色的音调类别和每行的长度，这使得九歌的模型结构自由，能够生成各种体裁的诗歌。

该模型在三十余万首格律诗文本的语料库上进行训练，取得了较好的效果。

诗三百则更进一步，该模型疑似由某位浙大博士开发，根据开发者开源的项目来看，诗三百同样采用了Sequence-to-Sequence模型，模型输入为诗歌题目或每句诗的首字（生成藏头诗），创新点在于使用了哈尔滨工业大学开源的中文BERT模型作为预训练[13]，进一步扩充语料库到八十余万首作品[14]，在这两方面的提升下，诗三百做到了目前最好的效果，几乎达到了以假乱真的地步，对诗歌没有深入了解的人几乎不能分辨诗三百生成的作品与真正的人类作品。

1.2.2 诗风计算研究现状

诗风计算任务属于计算风格学领域的研究范畴。目前计算风格学还停留在对文本进行定量分析，运用简单统计学方法对文本中的语言要素,包括词长,句长,段落以及高频词,句法等进行统计，根据语言要素出现的频率来确定文本属性特征的阶段[15]。统计手段较为简单，有时会用到支持向量机，主成分分析等较为高级的统计算法[16]。但尚未发现有研究者将深度学习方法应用到计算风格学领域。

1.3 本文研究内容

本文在分析了九歌与诗三百的优劣后，决定尝试使用新的模型与算法来完成格律诗文本生成的任务。九歌提出时间过早，没有使用预训练等手段，导致训练速度慢且效果不算好，而诗三百则是将哈尔滨工业大学训练的中文BERT模型与Sequence-to-Sequence相结合，输入标题生成诗歌内容，效果相对九歌得到了一定提升。然而诗歌的标题与诗歌内容有时不存在明确的对应关系，如著名的“落花诗”，目前存在至少千首以“落花”为题的诗歌，虽标题一致，他们的内容却是各不相同，因此诗歌题目到诗歌文本缺少一对一的映射关系。于是本文决定使用更适合文本生成任务的GPT模型，输入输出也变为可有一对一映射关系的首句与全诗。通过对大型预训练模型的迁移训练，取得了不错的效果。

为了验证模型生成效果的优劣，本文设计了一项“类图灵测试”，在六百余位诗歌爱好者的共同测评下，模型的表现异常优秀，大多数人无法分辨出模型创作的作品与真人创作的作品。

在测试特定风格诗歌生成的效果时，本文基于词向量，与LSTM算法结合，搭建了算法模型，提出了一种用来判定不同诗歌风格的模型。在唐诗风格识别等问题上取得了较好效果。

1.4 文章结构

本文分为七个章节对整个工作进行介绍，每个章节主要内容的组织结构如下：

第一章 绪论：该章节首先介绍本文工作的研究背景与实际意义，其次介绍国内外的相关研究情况，之后介绍本文的主要工作。

第二章 文本生成相关技术：本章详细梳理了本文所用到的算法与其它相关知识，如RNN，LSTM，Transformer，GPT等。

第三章 数据预处理：该部分介绍了数据来源，如何清洗数据，以及如何基于文学角度来选择数据，最后将数据转换为可以被模型处理的格式。

第四章 基于预训练算法的格律诗生成模型搭建：本章介绍了通过对预训练模型进行迁移学习，从而搭建写诗模型的过程。

第五章 基于预训练词向量与长短记忆神经网络的诗风判定算法：该部分搭建了一种诗风判定算法模型，并且对某些诗风判定问题进行了测试。

第六章 模型效果评估——类图灵测试：该部分描述了类图灵测试的具体细节与测试结果。

第七章 总结与展望：本章对诗歌生成模型与诗风判定模型这两项工作进行了总结，并对未来的应用情况做出了预估。

第二章 文本生成相关技术

2.1 文本生成技术简介

自然语言处理（NLP）分为自然语言理解（NLU）和自然语言生成（NLG）。NLU负责理解文本内容，而NLG负责根据信息生成文本内容，本文所做的诗歌生成工作属于NLG范畴，是文本到文本模式的生成任务。

文本到文本生成任务通过获取大量文本数据进行输入，训练出模型生成新的文本，具体架构因问题而异，一般包含文本表示与文本生成两个步骤。训练首先从获取语料开始，然后对原始语料进行处理，使之成为适合输入模型的语料进行训练。一般先使用各种词向量模型进行训练，获取一个能较好表达文本的词向量模型。文本生成从输入的数据开始，通过预先训练好的词向量模型将输入文本向量化，最后模型进行生成任务。

2.2 文本表达方式

人类的自然语言是一种离散的非结构化的知识，所以对于机器而言，无法直接理解人类的自然语言语言。因此我们需要将人类的自然文本转成计算机可以处理的形式。许多时候计算机只能识别和处理数字模式的数据，于是我们可以把文字表示成计算机能够运算的数字和向量。这就是词向量，简单来说就是用一个高纬度向量来表示一个单词或字符。词向量模型分为静态词向量模型与动态词向量模型：

2.2.1静态词向量模型

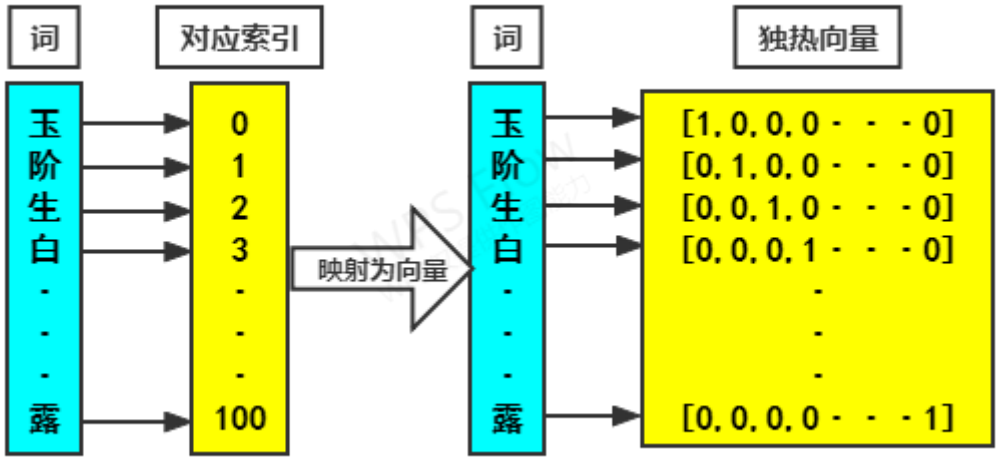
静态词向量模型就是指文本语料的输入与输出是一一映射的关系，即每一个词汇都只有一个词的表征与之对应。

最早的静态词向量模型是One-Hot，之后NNLM(神经网络语言模型)[17]出现，谷歌也开发Word2vec[18]工具包，含有Skip-grams(SG)模型和Continuous Bag of Words(CBOW)模型两种算法模型。随后，斯坦福大学的Jeffrey Pennington等人提出了Glove[19]静态词向量模型。

- One-Hot

独热表示(One-Hot）是一种文本的离散表示，目前最简单的词向量表征方法。具体来说，通常使用一个二进制向量来表示一个单词,向量的维度大小是语料库中单词的数量。

假如语料中如果含有V个词汇那么每个词汇都变为一个维度为V的向量，该向量除了索引处为1用于记录位置信息，其他各项均为0，向量记为[0,0,0,1,0,...,0,0]。



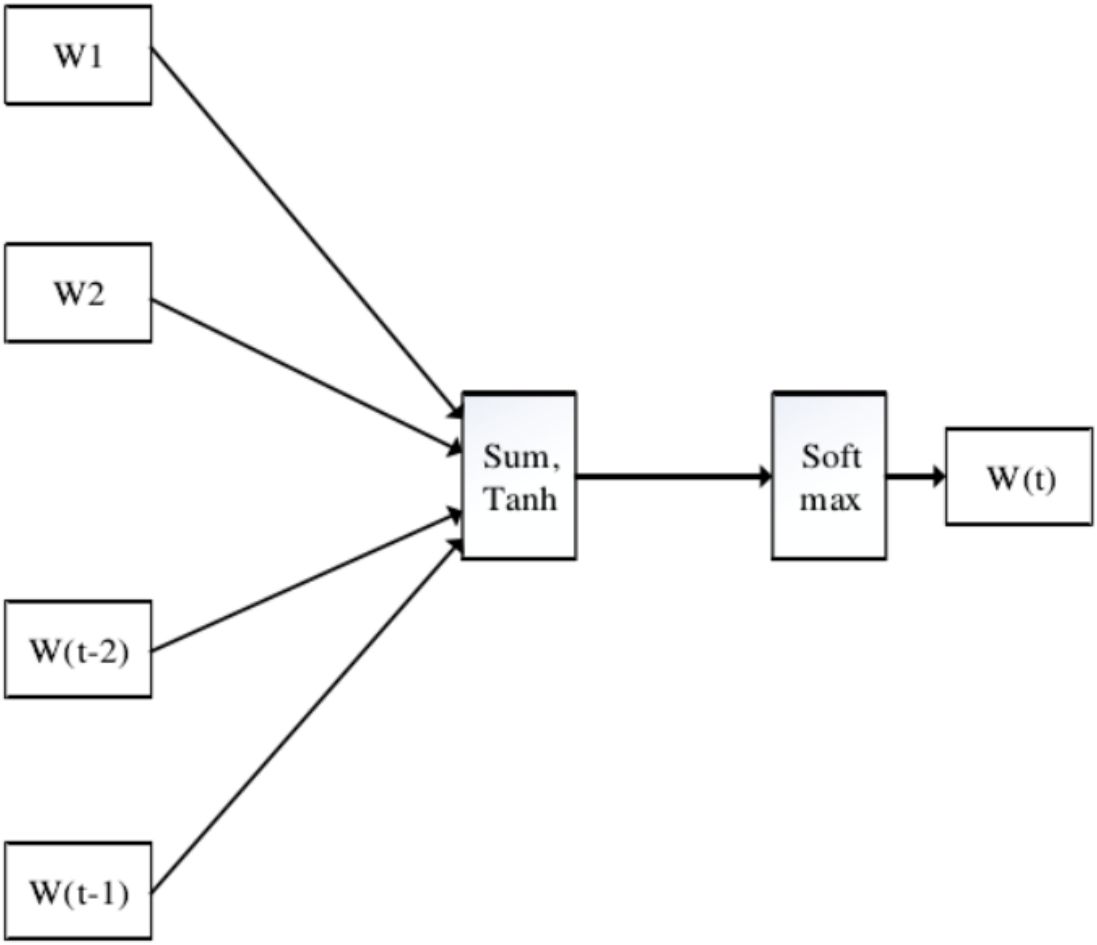
可以看得出，独热表示后的单词向量非常稀疏，语料中每增加一个词汇，向量的维度都要增加1，这就造成了当语料中有上千乃至上万个词汇的时候，向量也拥有上千乃至上万的维度，易造成维度灾难。不仅如此，由于任意两个词向量之间是相互正交的，因此One-hot不能够表示两个单词之间的关系，例如“杜甫”和“子美”二者关系比“杜甫”和“何逊”的关系应该更为相近，但是使用独热表示却很难衡量它们之间的相似程度。

- NNLM

神经网络语言模型(NNLM)方法由 Bengio[17]等人提出，其主要思路为把字典里的每一个词汇对应一个词特征向量，把词汇序列表示成联合概率函数，自动学习词特征向量和概率函数的参数。在 NNLM 中，每一个词汇为向量空间中的一个点，而且特征的数目要比字典的大小要小，它的概率函数表示为在给定前一个词下，后一个词的条件概率的乘积。NNLM模型的原理即是通过一个词汇前面的所有词汇来求得其出现的概率，目标函数表示为：

$$L(\theta) = \sum P(w_t|w_1, \dots w_{t-1})$$

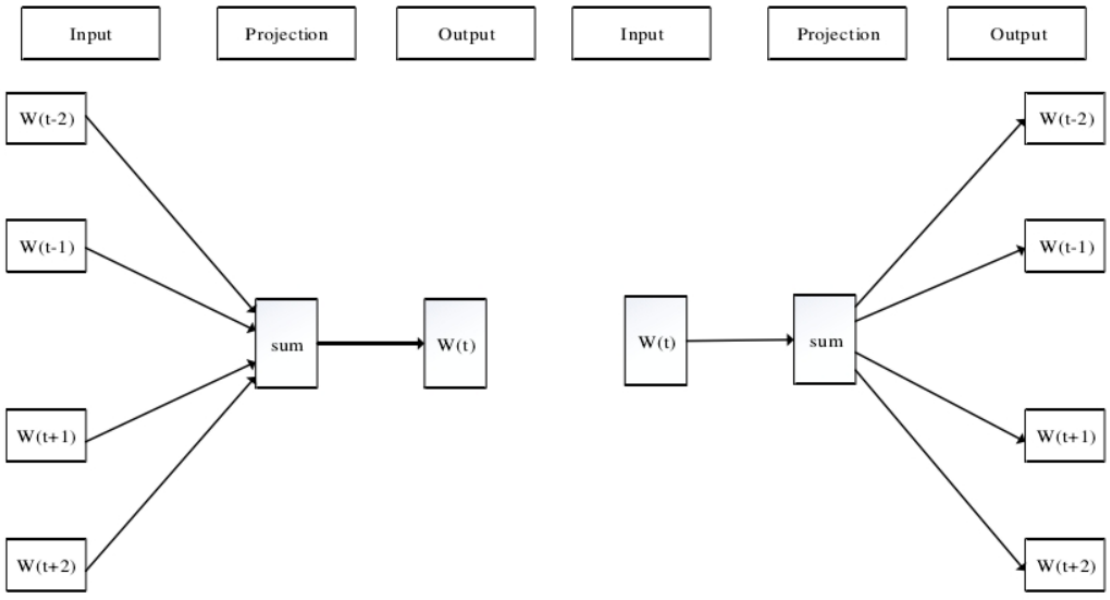
NNLM模型原理如图所示：



NNLM仅仅是利用上文的信息来得到下文词汇的文本表征，没有更好地利用上下文信息。

- Word2vec

2013年，谷歌开源了一款工具，Word2vec[18]，用于进行词向量计算。Word2vec有两种算法，分别为Skip-grams(SG)模型和 Continuous Bag of Words (CBOW)模型。Skip-grams模型的原理是根据中心词汇向量来推出上下文词汇的向量，首先为每个词汇类型建立一个密集向量，然后通过这个词汇来预测出现在其上下文词汇的词向量。CBOW模型的原理是通过目标词汇上下文的词汇来预测目标词汇的词向量。这两种模型都会产生一个词汇与词向量一一对应的映射表。模型结构如图所示：



Word2vec解决了词向量维度过于庞大的问题，也表达了词语之间的关系：具有相同上下文环境的词汇，会具有相似的向量，且不需要人为的对词汇进行词性标注。
尽管如此，这款工具还是无法区分近义词与反义词，因为近义词与反义词也会出现在相似的环境之下，此外两种模型都只运用了局部的上下文来得出词向量，缺少对全局信息的利用。

2.2.2动态词向量模型

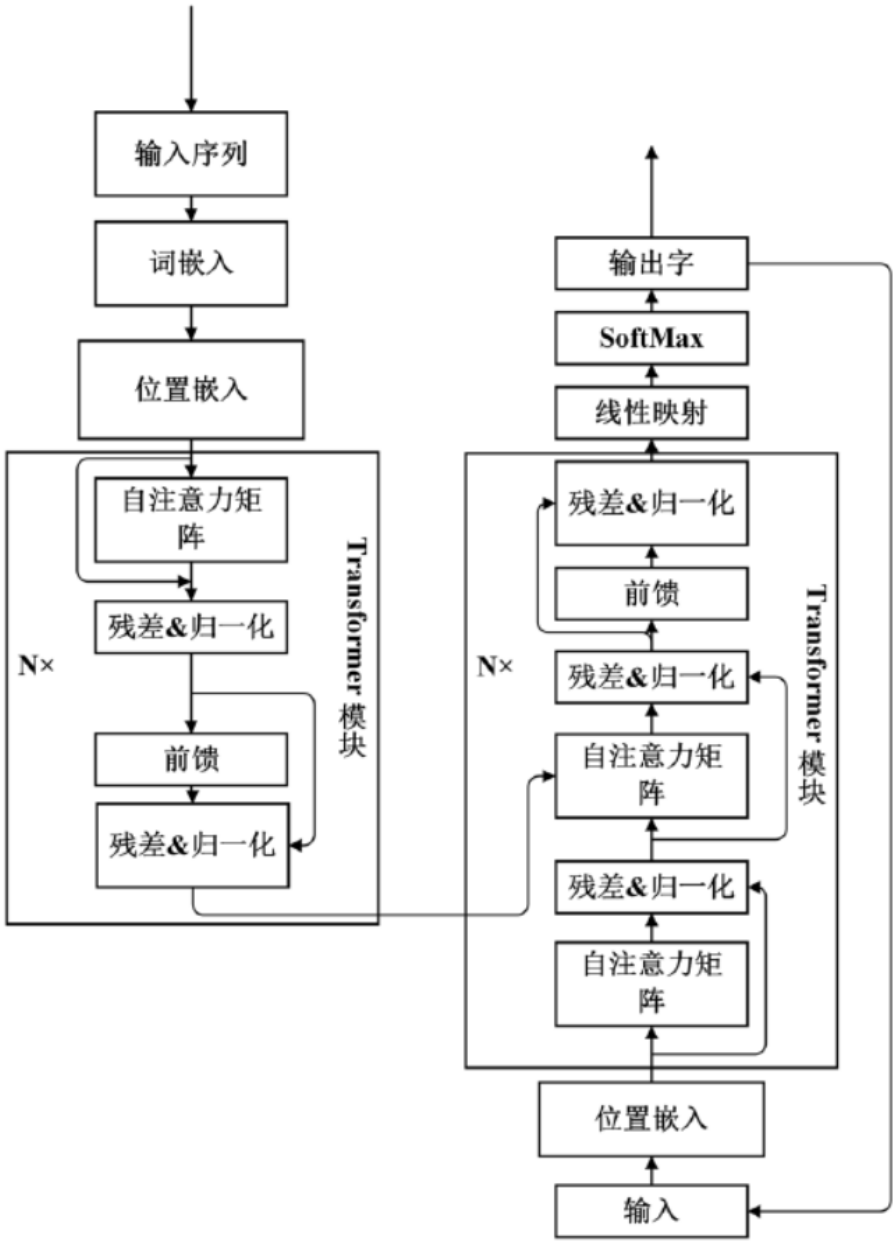
在静态词向量模型中，词汇与词向量之间的关系是一一对应的，即一个词汇映射一个词向量，但实际上很多词汇在不同的上下文环境中会具有不同的意思，而向量却无法及时的被变更。故静态词向量模型是无法较好的解决一词多义的问题。因而动态词向量模型被提出来,其可以很好解决一词多义的问题。

- GPT-1

GPT-1是OpenAI发布的一个语言模型[20]。GPT-1使用单向语言模型进行预训练,通过Fine-tuning 的模式解决下游任务。GPT-1和ELMO相比最大的两点不同。特征提取器使用的是性能更好的Transformer 的解码器结构

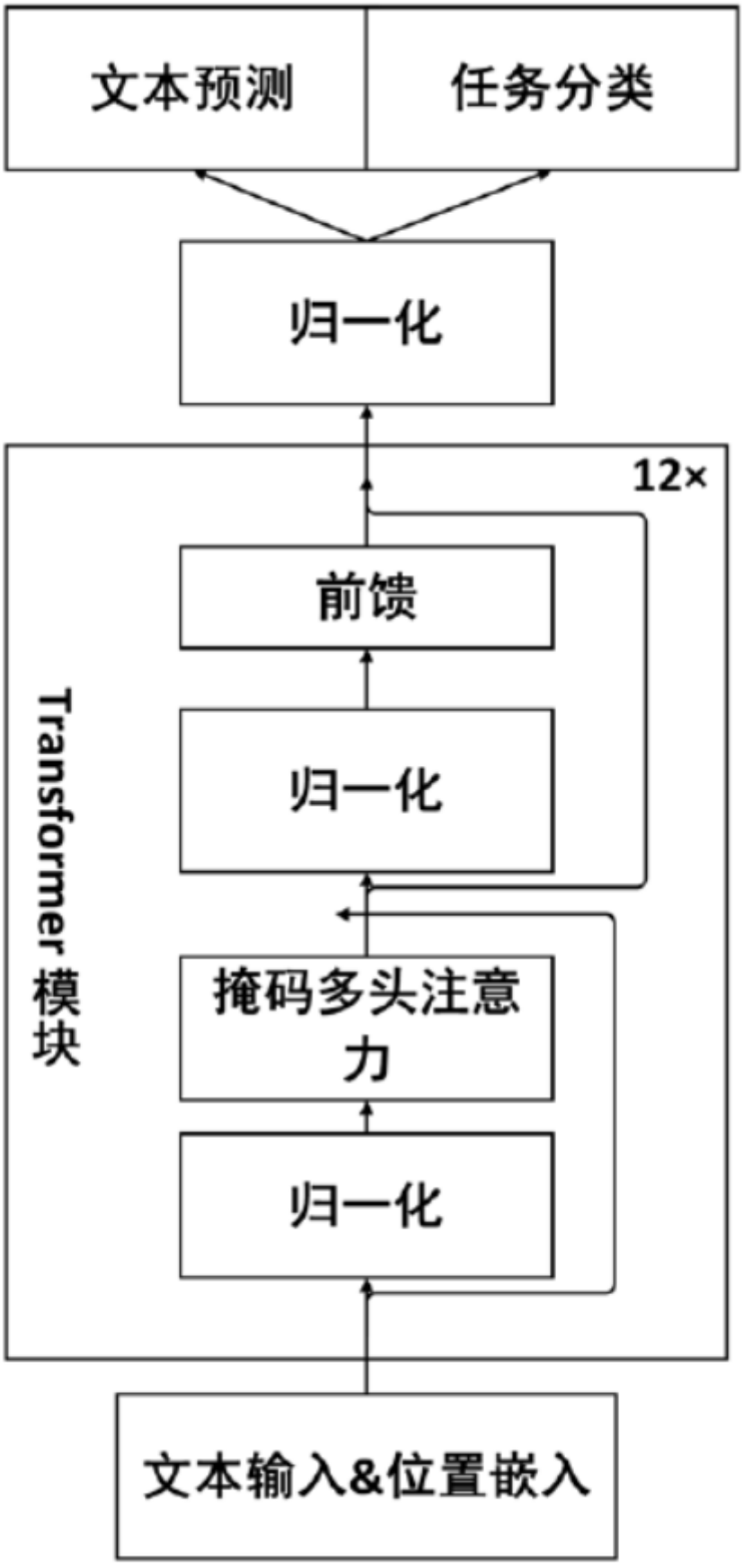
Transformer是encoder-decoder 架构[21]，Transformer 含有6层encoder和6层decoder，每一层都主要由Self-Attention构成。

Transformer的每一层中运用了Multi-head机制来确保它的词向量更为准确，该机制就是对词汇进行多组特征向量的运算，通过不同特征向量的运算来得到不同的特征表达，再将这些表达相加，通过全连接层降维。模型结构如图所示：



由于Transformer 并行化的机制，可以有效的利用GPU的计算能力，此外使用Transformer可以直接建立两个在语句中相距较长的词汇之间的关系。

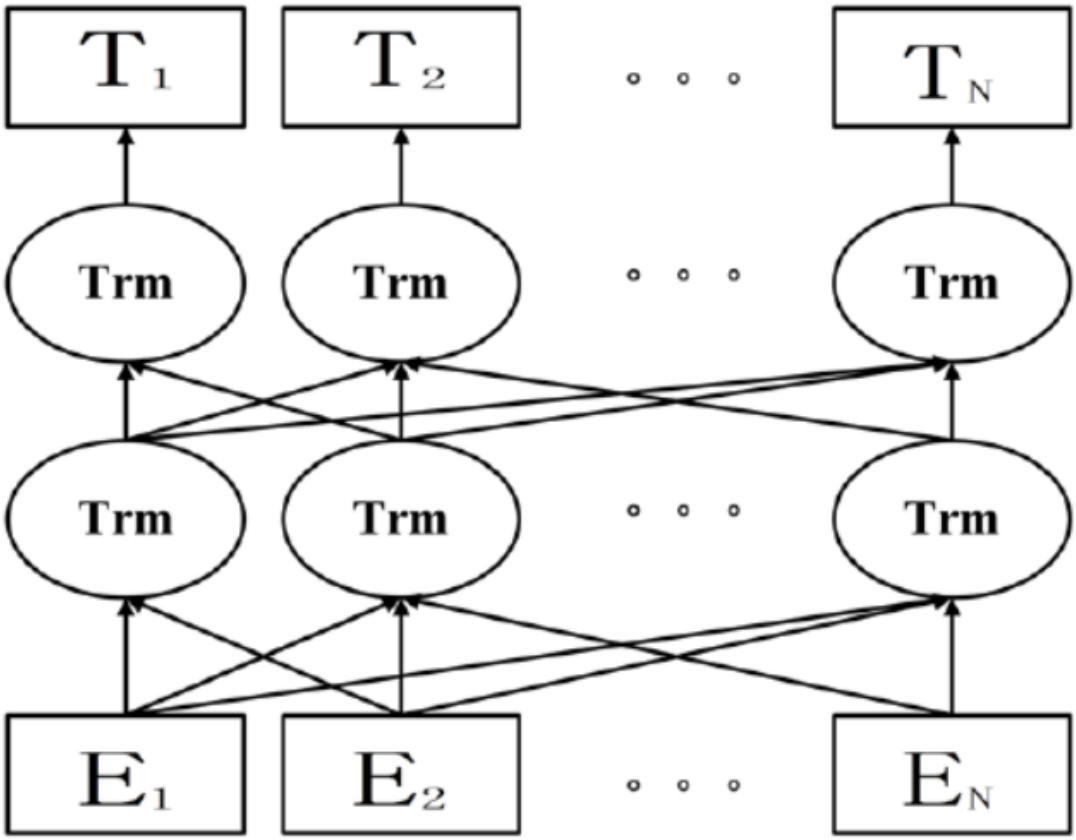
在GPT预训练语言模型中，模型使用的结构是Transformer中的解码器部分，如图所示：



GPT框架使用了12层的Transformer解码器堆叠并最后进行Softmax归一化处理，预训练任务在大规模语料上进训练，在第一阶段的预训练过程完成后，训练得到的模型参数用于第二阶段下游任务的学习，其目的在于充分利用大规模语料信息，更好的加强下游任务的语义理解。

- BERT

BERT(全称Bidirectional Encoder Representations from Transformers)是谷歌AI团队于2018年10月发布的一个新型语言模型[22]，它是一个真正的深度双向语言模型，BERT的特征提取器是一种基于Transformer的多层双向编码器，其中的Transformer架构与原始的Transformer 架构是相同的。BERT最大的创新就是提出了Masked Language Model来进行预训练任务，Masked LM是对输入语句遮蔽一部分词汇，然后通过多轮迭代训练对遮蔽部分进行预测生成，建立了一个基于Transformer的双向语言模型。



BERT在自然语言处理领域中取得了突破性的进展，训练完成后，可以直接用于下游的自然语言处理任务之中。BERT也有着一些缺点:BERT的训练十分的耗费时间，使用GPU进行训练可能需要一周以上的时间，但是谷歌开源了已经训练好的语言模型;此外，BERT模型的参数数量大，需要占用不小的显卡;还有BERT在训练的时候是使用遮蔽15%词汇的方法，但是实际在下游任务中是不会这样的，即训练时模型看到的信息与预测时模型看到的信息是不相同的;BERT遮蔽的词汇是相互独立的，忽略了遮蔽词汇之间的关系。

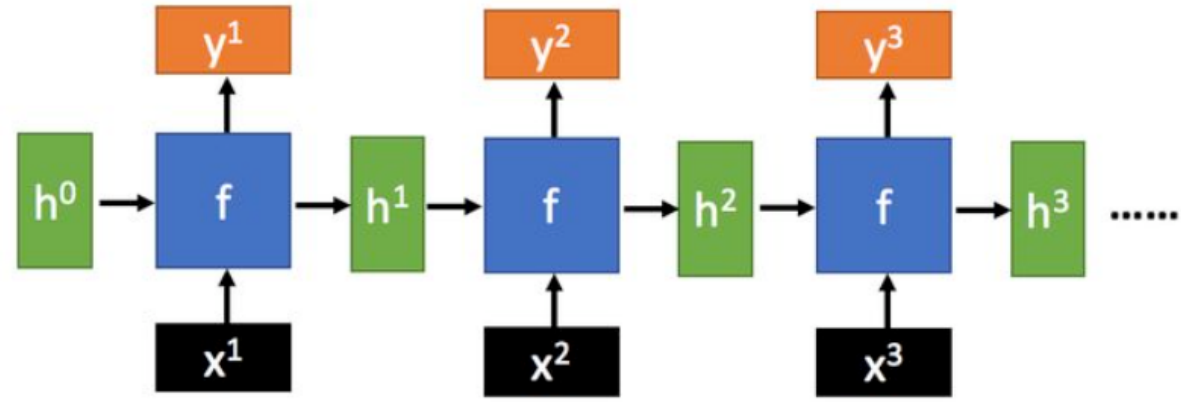
- GPT-2

如果说BERT是坚持双向语言模型，那么GPT就是坚持单项语言模型，GPT-2模型结构和 GPT-1大致相同，仍旧是采用单向语言模型，特征提取使用的是Transformer的解码器结构，GPT-2相对于GPT-1主要是做了以下的改进。transformer模型由原来12层叠加变成了48层，训练参数变为15亿。更大的网络自然需要更多的数据，因此数据采用质量更高、数量更大、涵盖范围更广的数据 Web Text。对一部分下游任务继续进行fine-tuning，但也在部分下游任务中尝试着进行零样本学习(zero-shot)。
它和BERT特征提取器都使用的是Transformer,但是它使用的是单向语言模型，在一些自然语言处理经典任务中可能不似BERT的双向语言模型效果那么好，但是GPT-2在文本生成任务上更为优秀，如摘要生成、诗歌生成、新闻生成等方面。

2.3 神经网络算法

2.3.1 RNN

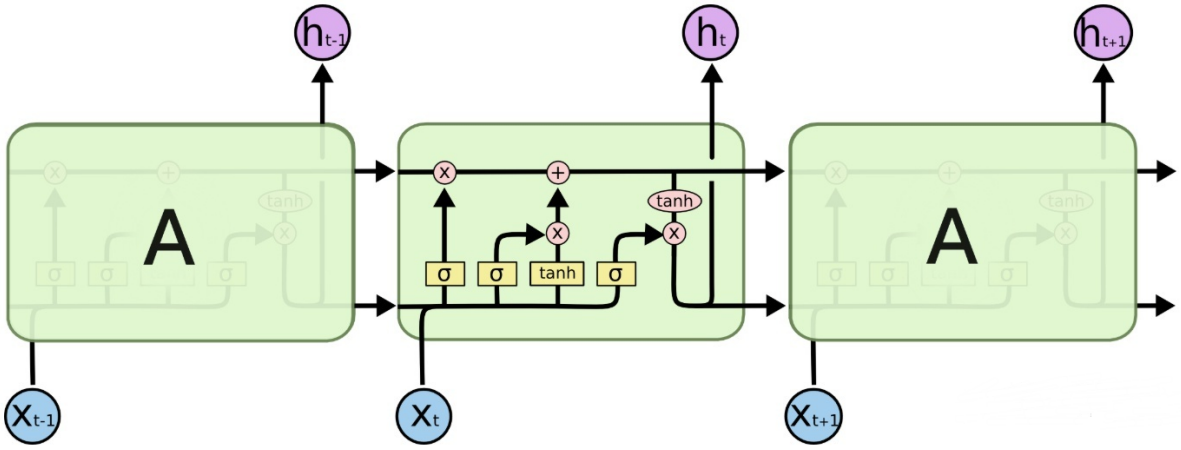
循环神经网络（Recurrent Neural Network，RNN）[23]是一种用于处理序列数据的神经网络。相比一般的神经网络来说，该模型能够处理序列变化的数据。比如某个单词的意思会因为上文提到的内容不同而有不同的含义，RNN就能够很好地解决这类问题。结构如下图：



由于RNN每一阶段的输出信息都与之前的输出信息以及当前的输入信息有关，因此 RNN是作为自然语言处理的一种有效神经网络结构。因为传统RNN大多使用反向传播时间(BPTT)算法。但是这一种算法有一种致命的缺陷，那就是随着网络层数的不断加深，它会产生梯度膨胀乃至梯度消失的问题。
梯度消失就是指如果梯度较小的话(小于1时)，多次迭代以后，指数相乘，梯度很快就会下降到对调参几乎没有影响了。就好比0.99的100次方趋近于0。梯度膨胀则是指如果梯度较大的话(大于1时),多次迭代以后，同理又会使梯度变得十分庞大。因此传统的RNN的问题是一段话过长时就会遗忘最开始的句子，也就是说会将过去有用的信息遗忘。

2.3.2 LSTM

为了解决RNN所带来的问题，于是(Long Short-Term Memory)LSTM被提出来[24]。其核心本质在于，通过引入巧妙的可控自循环，以产生让梯度能够得以长时间可持续流动的路径。
LSTM引入了门开关的概念，LSTM含有三个门开关，分别为遗忘门、输入门和输出门，门开关就是控制将当前的信息量输入到下一阶段信息量的百分比，如果门开关的大小为1，即将所有的输入全部输入到下一阶段，如果门开关的大小为0，即当前输入不输入到下一阶段，遗忘门就是控制过去的记忆信息；输入门就是控制当前的输入信息；;输出门就是控制当前得到的输出信息中有多少用于最后的输出。



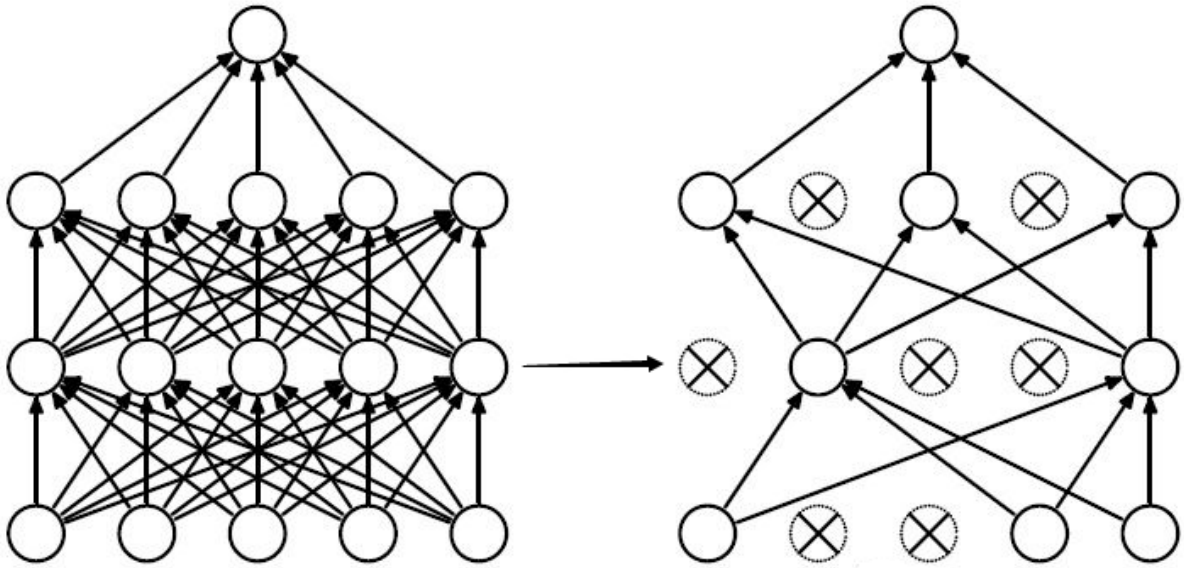
LSTM有效的控制了梯度消失以及梯度爆炸的问题，使得RNN的问题在一定程度上得到了控制。

2.3.3 Dropout

在机器学习的模型中，如果模型的参数太多，而训练样本又太少，训练出来的模型很容易产生过拟合的现象。在训练神经网络的时候经常会遇到过拟合的问题，过拟合具体表现在：模型在训练数据上损失函数较小，预测准确率较高；但是在测试数据上损失函数比较大，预测准确率较低。

过拟合是很多机器学习的通病。如果模型过拟合，那么得到的模型几乎不能用。为了解决过拟合问题，一般会采用模型集成的方法，即训练多个模型进行组合。此时，训练模型费时就成为一个很大的问题，不仅训练多个模型费时，测试多个模型也是很费时。

Dropout[25]可以比较有效的缓解过拟合的发生，在一定程度上达到正则化的效果。Dropout说的原理是：我们在前向传播的时候，让某个神经元的激活值以一定的概率p停止工作，这样可以使模型泛化性更强，因为它不会太依赖某些局部的特征，如图所示。



2.4 迁移学习

迁移学习（Transfer Learning）是一种机器学习方法，是把一个领域（即源领域）的知识，迁移到另外一个领域（即目标领域），使得目标领域能够取得更好的学习效果。

通常，源领域数据量充足，而目标领域数据量较小，这种场景就很适合做迁移学习，例如我们我们要对一个任务进行分类，但是此任务中数据不充足（目标域），然而却又大量的相关的训练数据（源域），但是此训练数据与所需进行的分类任务中的测试数据特征分布不同，在这种情况下如果可以采用合适的迁移学习方法则可以大大提高样本不充足任务的分类识别结果。

本文所做工作中，诗歌语料较少，且算力较缺，所以在预先训练好的模型上使用迁移学习事半功倍。因此本文采取UER.py项目[26]进行对预训练模型的迁移学习工作。

2.5分词工具

中文自然语言处理与英文自然语言处理最大的不同之处就在于中文语言处理需要对文本进行分词处理，而分词处理也一向是中文自然语言处理中的重点难点。针对中文分词器，现已开发出了很多有用的工具。考虑到性能与评价，我们选择了三个分词工具进行测试：

- **Pkuseg：**Pkuseg是由北京大学语言计算与机器学习研究组研制推出的一套全新的中文分词工具包。Pkuseg官方给出与来自清华的THULAC和目前主流中文分词工具jieba的比较效果显示，Pkuseg的准确率远超于THULAC和jieba。[27]
- **THULAC：**THULAC（THU Lexical Analyzer for Chinese）由清华大学自然语言处理与社会人文计算实验室研制推出的一套中文词法分析工具包，具有中文分词和词性标注功能。THULAC准确率

高。该工具包在标准数据集Chinese Treebank（CTB5）上分词的F1值可达97.3%，词性标注的F1值可达到92.9%，与该数据集上最好方法效果相当。[28]

- **jieba**：目前最主流的中文分词工具。但因其发布较早，有其滞后性。

我们选择《关雎》作为测试样本来评测三个分词工具的性能，测试结果如下：

Pkuseg

```
['关关', '雎鸠', '在', '河之洲', '窈窕', '淑女', '君子好逑', '参差', '荇菜', '左右流之', '窈窕', '淑女', '寤寐求之', '求之不得', '寤寐', '思服', '悠哉悠哉', '辗转反侧', '参差', '荇菜', '左右', '采', '之', '窈窕', '淑女', '琴瑟友之', '参差', '荇菜', '左右芼之', '窈窕', '淑女', '钟鼓乐', '之']
```

THULAC

```
['关关', '雎鸠', '在', '河', '之', '洲', '窈窕淑女', '君子好逑', '参差', '荇菜', '左右流之', '窈窕淑女', '寤寐求之', '求之不得', '寤寐思服', '悠', '哉', '悠', '哉', '辗转反侧', '参差', '荇菜', '左右', '采', '之', '窈窕淑女', '琴', '瑟友之', '参差', '荇菜', '左右芼之', '窈窕淑女', '钟鼓乐之']
```

jieba

```
['关关雎', '鸠', '，', '在', '河之洲', '。', '窈窕淑女', '，', '君子好逑', '。', '参差', '荇', '菜', '，', '左右', '流之', '。', '窈窕淑女', '，', '寤寐求之', '。', '求之不得', '，', '寤寐', '思服', '。', '悠哉悠哉', '，', '辗转反侧', '。', '参差', '荇', '菜', '，', '左右', '采之', '。', '窈窕淑女', '，', '琴瑟', '友之', '。', '参差', '荇', '菜', '，', '左右', '芼', '之', '。', '窈窕淑女', '，', '钟', '鼓乐', '之', '。']
```

通过人工对样本分词效果进行评判，Pkuseg与THULAC效果参差，二者都远胜于jieba，考虑到Pkuseg速度远慢于THULAC，故在本文中选用THULAC对文本进行分词处理。

2.6 网站搭建技术

2.6.1 Flask 后端框架

Flask是一个轻量级的可定制框架，使用Python语言编写，较其他同类型框架更为灵活、轻便、安全且容易上手。它可以很好地结合MVC模式进行开发，开发人员分工合作，小型团队在短时间内就可以完成功能丰富的中小型网站或Web服务的实现。另外，Flask还有很强的定制性，用户可以根据自己的需求来添加相应的功能，在保持核心功能简单的同时实现功能的丰富与扩展，其强大的插件库可以让用户实现个性化的网站定制，开发出功能强大的网站。

2.6.2 JavaScript

JavaScript是一种属于网络的高级脚本语言,已经被广泛用于Web应用开发,常用来为网页添加各式各样的动态功能,为用户提供更流畅美观的浏览效果。通常JavaScript脚本是通过嵌入在HTML中来实现自身的功能的。
JavaScript有以下特点：

- 是一种解释性脚本语言（代码不进行预编译）。
- 主要用来向HTML（标准通用标记语言下的一个应用）页面添加交互行为。
- 可以直接嵌入HTML页面，但写成单独的js文件有利于结构和行为的分离。
- 跨平台特性，在绝大多数浏览器的支持下，可以在多种平台下运行（如Windows、Linux、Mac、Android、iOS等）。

本文搭建的网页使用JavaScript编写前端页面。

第三章 数据预处理

3.1 收集数据

本文诗歌语料，平水韵表等数据大致来源于三个途径——

3.1.1 网络公开数据

GitHub用户Werneror所开源的名为Poetry[14]的项目收录了从先秦到现代的共计85万余首古诗词。古诗词数据按朝代存储于多个 csv 文件中，有“题目”、“朝代”、“作者”和“内容”共四个字段。单个文件不大于30M。诗词语料中有一些生僻字，这些生僻字属于 utf8mb4 字符，在许多设备中无法显示，故而使用“?“来替代。

GitHub用户garychowcmu将殆知阁TXT版古代文献开源到daizhigev20[29]项目中，其中有约二十亿字的语料，但良莠不齐，许多甚至没有标点。本文经过人工审校后取了其中一部分作为语料。

3.1.2 网络爬虫

搜韵网[3]是目前收录诗词作品最全的诗词在线查询网站，收录了先秦至宋的几乎所有现存诗词作品。此外还收集了许多明清诗人的作品。本文编了一个简单的爬虫，可以将指定诗人的作品从搜韵网中全部保存下来。伪代码如下：

```
poet_name = input()#接收指定诗人名字
url = pre + '/' + poet_name#通过诗人名字确定网址
first_page = url + '&page=' + '1'#第一页作品网址
r = requests.get(first_page).text#获取第一页作品内容
text = process(r)#将网页内容解析为[[题目，作者，年代，正文]，[题目，作者，年代，正文]...]的二维列表
i = 2#初始化一个计数器
temp_page = url + '&page=' + str(i)#从第二页开始，访问网页内容
#如果访问页数超过索引，则自动跳转为第一页，所以当后续访问的页面内容与第一页相同时代表已经将全部作品访问完成
while r != requests.get(temp_page).text:
    temp_text = process(requests.get(temp_page).text)#将网页内容解析为[题目，作者，年代，正文]的列表
    text += temp_text#将新内容加到现有内容列表中

dataframe = pd.DataFrame(text)#将列表转化为pandas中的dataframe结构
dataframe.to_csv("poet.csv",index=False,sep=',')#写入csv中
```

本文根据此爬虫从搜韵网中收集到了一部分明清诗人的作品。

3.1.3 朋友收集

由于某些少见的诗词资料，网上没有公开的资源。有些朋友根据古籍PDF版本，人工录入，将其变成方便处理的TXT版本文件。此处感谢乾社的陈冠宇同学，提供了黃人《石陶黎煙室遺稿》，阮大鋮《詠懷堂詩》等资料。

3.1.4 数据格式

最后将三种途径得来的语料整合在一起，存储在多个csv文件中，分“题目”、“朝代”、“作者”和“内容”共四个字段。

3.2 选取数据

初始收集到的数据是整体诗词的数据，有诗有词，诗中又分古风以及格律诗。诗与词有明显差异，词需按照词牌，每句长度皆有定数，长短错落有致，故称长短句。而古风与格律诗也有差异，格律诗比古风用韵更严谨，且每句必依平仄。本文以生成格律诗为任务，需将格律诗的语料提取出来。伪代码如下：

```
#新建一个二维列表储存诗歌语料
gelv = [["题目","朝代","作者","内容"]]
#遍历corpus文件夹下的多个csv文件
for filepath,dirnames,filenames in os.walk(r'corpus'):
     for file in filenames:
        path = 'corpus/' + file
        df = pd.read_csv(path)
        #遍历csv文件中的每一行
        for i in df.iloc:
            #如果题目中不包含词牌名，则该行是诗
            if ci_find(i[0]) == '诗':
                #如果诗歌内容符合平仄格律，则将其装入到列表中
                if classify(i[3]) != '其它':
                    gelv.append(i.tolist())
#将列表转化为pandas中的dataframe结构
gelvdf = pd.DataFrame(gelv)
#写入csv中
gelvdf.to_csv('格律.csv', index=False)
```

3.3 提取主题词

3.3.1 分词处理

首先将诗歌文本用THULAC进行分词，然后将在停用词表中的词语去除，得到一个非停用词列表。

```
#使用THULAC将诗歌文本分词,得到分词列表
text = thu.cut(poem, text=False)
#初始化列表储存词语
words = []
#遍历列表中词语
for word in text:
    #若该词语不在停用词表中，加入列表中
    if word not in stopwords:
        words.append(word)
```

3.3.2 TF-IDF提取主题词

TF-IDF（term frequency–inverse document frequency）是一种用于信息检索与数据挖掘的常用加权技术。TF是词频(Term Frequency)，IDF是逆文本频率指数(Inverse Document Frequency)。$TF = \frac{某词在该文章中出现的次数}{文章总词数}$，$IDF = log(\frac{语料库文章总数}{包含该词的文章数+1})$，$TF-IDF = TF \times IDF$

TF-IDF是一种统计方法，用以评估一字词对于一个文件集或一个语料库中的其中一份文件的重要程度。字词的重要性随着它在文件中出现的次数成正比增加，但同时会随着它在语料库中出现的频率成反比下降。

本文利用该算法提取主题词，绝句每首提取三个，律诗每首提取五个，存储到原诗所在csv中，伪代码如下：

```
#读入格律诗存储csv文件
path = '格律.csv'
```

```
df = pd.read_csv(path)
#初始化装有主题词的列表
key_df = []
#遍历每一行
for i in df.iloc:
    #提取主题词并与原诗一起存储到key_df列表中
    keyword_tf = tf_idf(i[3])
    n = i.tolist()
    n.append(keyword_tf)
    key_df.append(n)

#将多了主题词列的列表写入csv
key_df = pd.DataFrame(key_df)
key_df.to_csv('格律.csv', index=False)
```

3.4 基于预训练词向量提取关键字

本文选择使用word2vec中的跳元模型（skip-gram）[18]，将诗歌文本中每个字符对应一个定长的稠密实数词向量，维度通常为50~300，每一维均由一个实数表示。

由于Shen Li等人已经在四库全书语料上使用该算法进行了训练并得到了300维的单字词向量（下称字向量）[30]，且得到了令人满意的效果，本文在该字向量的基础上进行模型搭建工作。该项目收录了19527个字对应的字向量，前十个字的字向量如下表所示。

字	字向量
之	[-0.386241,-0.200756,0.106928,0.327273,0.080174 ……-0.365105]
以	[0.042976,-0.105353,0.189561,0.310238,0.402705 ……0.008979]
曰	[-0.253626,-0.486539,0.438117,-0.193879,0.134444……0.182786]
为	[-0.034353,-0.075638,0.247427,-0.560503,-0.003090……0.597949]
其	[-0.197984,-0.268704,0.211067,0.209840,0.415124……-0.235650]
而	[-0.080227,-0.418216,0.417316,0.488205,0.467254 …… -0.357677]
也	[-0.061710,-0.204168,0.370755,0.001222,-0.133613……-0.007368]
人	[-0.295339,-0.088422,-0.435675,-0.117920,-0.176023……-0.071651]
有	[-0.129041,0.190121,-0.133640,0.095273,0.375228……-0.185826]

将诗歌正文去除停用词后，余下的字转化为字向量进行表示，通过计算确定这些字向量的中心点，找到离中心点欧氏距离最近的数个向量，这些向量对应的字符即为诗歌的关键字。根据篇幅长短，绝句找出5个关键字，律诗找出7个关键字。伪代码如下：

```
#读入格律诗存储csv文件
path = '格律.csv'
df = pd.read_csv(path)
#初始化装有主题词的列表
key_df = []
#遍历每一行
for i in df.iloc:
    char_vec =[]
```

```
    char_list = []
    for c in i[3]:
        #如果字符在停用词表中，放弃处理
        if c in stop:
            pass
        else:
            #字符不在停用词表中，且在字向量表中有对应关系，则将字符与字符所对应向量存储起来。
            if c in char_dic:
                char_list.append(c)
                char_vec.append(char_dic[c])
    n = i.tolist()
    #将该文本的关键字存储起来。
    n.append(topk(char_list, char_vec))
    key_df.append(n)
#将多了关键字列的列表写入csv
key_df = pd.DataFrame(key_df)
key_df.to_csv('格律.csv', index=False)
```

第四章 基于预训练算法的格律诗生成模型搭建

4.1 预训练模型

由于算力与语料的缺少，本文采取对已训练好的预训练模型进行迁移学习的方式。格律诗相较于现代白话，与文言文更为接近，于是本文采取开源的gpt2-chinese-ancient模型[26]作为预训练模型，该模型在大量文言文语料上训练了 500,000 步。模型参数如下：

emb_size	**768**
feedforward_size	3072
hidden_size	768
hidden_act	gelu
heads_num	12
layers_num	12
dropout	0.1

4.2 迁移学习

4.2.1 数据格式处理

首先执行执行preprocess.py程序，将原先的语料转化为pt格式文件。由于语料中最长的七言律诗加标点一共64字符，故设定预训练样本的最大长度为64。词表使用google_zh_vocab，另外用扩展词汇表来处理词汇表外的单词，将语料库中出现次数大于等于100的汉字加到词表中。

4.2.2 训练

在Windows10+RTX3060环境下，设定batch_size=16，训练了100,000步以上，准确率达到0.45左右，loss约2.8，此时停止训练。此时迁移学习基本完成，给定诗歌首句，模型可以生成质量尚可的诗歌作品。

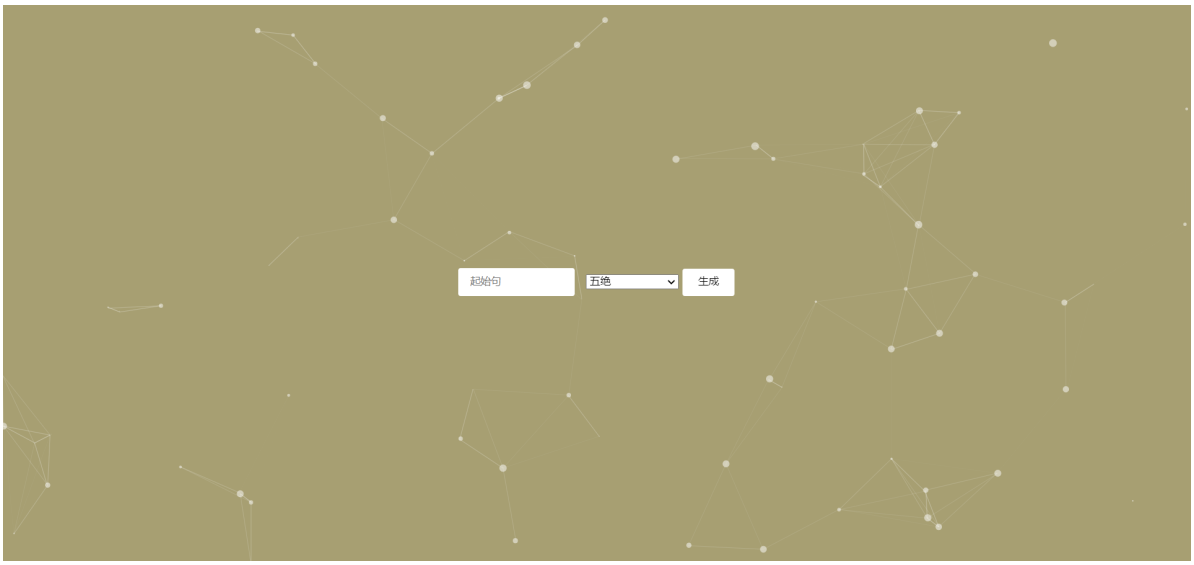
而后添加人工规则，根据诗词的格律人工编写规则进行判断，使模型生成符合现有诗词格律的作品。

4.3 风格作品生成

根据同样的原理，本文选择了孙原湘，王彦泓两人的作品进行了迁移学习，训练出了一个艳体诗风格的诗歌文本生成模型。另外选取吕本中，曾几，赵蕃，黄庭坚，陈与义，陈师道，苏轼，谢逸，洪刍，饶节，韩驹，李彭，谢薖，杜甫'，韩愈等人的七言律诗作品，训练出了一个江西诗派风格的模型。使用下文提出的风格判定算法进行判断，相似度在0.7左右，属于比较相似的程度。

4.4 网站搭建

为了更多人能使用本模型，本文在完成模型训练后，于腾讯云租了一台虚拟机开发了一个网站，将本文训练的模型放到网站上进行试用。页面如下：



用户需要输入首句，然后选择合适的体裁：有五绝，五律，七绝，七律等。点击提交后，使用flask开发的后端程序会接收到首句与体裁两个信息，然后调用模型，以首句为前缀，套用体裁对应的规则来生成作品。如果生成完成，则返回作品给用户。如果生成20次都不满足要求，则返回信息提醒用户失败请重试。

模型生成作品的历史记录保存在后端中以供查阅。

第五章 基于预训练词向量与长短记忆神经网络的诗风判定算法

5.1 问题背景

由于我国诗学传统的悠久与连贯，诗人常有师法前代的行为。宋之江西诗派，取径杜甫，遥尊其为“一祖”；明之前后七子，诗法盛唐；晚清同光体，常被目为宋诗派，诗多宋调。这些诗人的作品中，明确显露出了他们对某些前人诗风的学习与继承。后世人若研究这些诗人，必然无法绕开对他们诗学脉络的探索。判定一位诗人的作品是否有前人（如学杜，学韩）或前代（如学唐，学宋）的风格，有相当重要的意义。

然而如何进行风格的判定？过往的方法是凭借个人的阅读体验，至多摘出一些风格明显的句子进行论证。这种依赖于主观感觉的方法有时会导致不同的结论，而且需要花费大量时间去细读。本文尝试用自然语言处理技术来解决这一问题，首先将该问题转化为一个文本二分类任务，其次将某种风格的诗歌与同等数量的随机诗歌作为训练集，并将训练集向量化。最后将向量化的训练集输入到应用LSTM算法的模型中进行二分类训练，训练集中目标风格诗歌向量映射为1，随机诗歌向量映射为0，训练达到一定精度后停止。此时经过训练的模型就可以进行诗歌风格判定任务——输入需要判定的诗歌，模型会给出一个介于0-1的结果，越接近1说明在此判定模型中该作品越有可能被分类为目标风格诗歌。

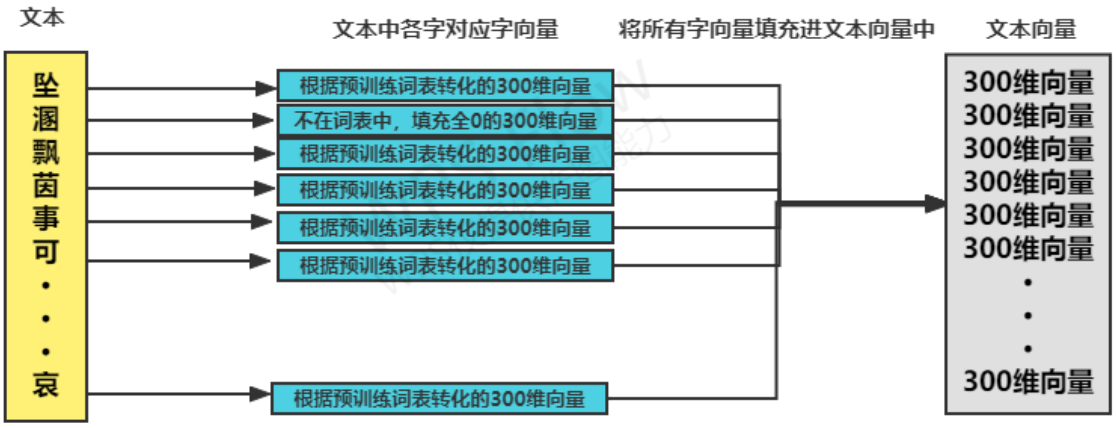
为了验证模型的效果，本文先是将此方法应用到了孙原湘《天真阁外集》中，用五百首艳体七言律诗训练出了一个识别艳体风格的模型，以王彦泓，丘逢甲等人的作品为测试集进行测试，取得了预想中的效果。接着扩大了问题规模——以所有现存唐人七言律诗为训练集，训练出了一个唐诗风格识别模型，用以判定一首七言律诗是否有唐人风味。使用该模型对明七子，同光派等代表人物进行判定，取得了与文

学史评价基本吻合的结论。

5.2 文本向量化

由Salton 等提出向量空间模型(Vector Space Model, VSM)[31]被广泛地应用于文本分类、检索和相似度计算等任务。

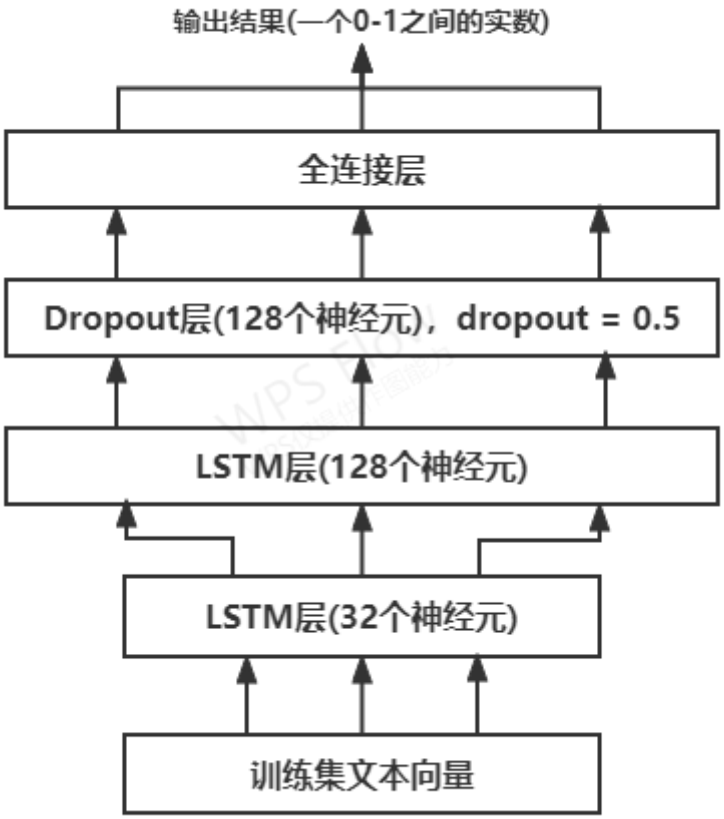
在 VSM 模型中, 一篇文本可视为由词语构成的集合,由此可以将文本转化为一个多维向量，本文在在已有预训练词向量基础上构建文本向量。具体流程如下——



经过以上过程，长度为N的诗歌文本被转化为尺寸为$N \times 300$的文本向量。

5.3模型结构

本文采用LSTM算法搭建模型，为了提升泛化性，避免过拟合，使用了Dropout策略，每次随机丢失一半的参数，以此提升模型的健壮性。模型结构如下——



损失函数是二值交叉熵，优化算法采用Adam，最后的全连接层激活函数为sigmoid，输出一个0-1的实数。

5.4艳体诗识别

5.4.1数据预处理

孙原湘的《天真阁外集》多为艳体诗，钱锺书先生认为，这些诗"殆庶《香奁》、《疑雨》 二集"， 又谓能"上配《疑雨》"[32]。

《天真阁外集》作品多为七律，故而本文将七言律诗作为训练和验证的体裁，将孙原湘《天真阁外集》中的七言律诗抽取出来，数量为500篇，标注为1，又随机抽取他人的七言律诗500篇七言律诗作为对比，标注为0。将两者混合，随机抽取10%作为验证集，剩余90%作为训练集。

5.4.2训练

使用本文构造的模型进行训练，训练10轮后，验证集准确率约为0.9，损失值为0.47，为避免过拟合，停止继续训练。

5.4.3效果验证

为验证模型效果，抽取王彦泓和丘逢甲的全部七言律诗作品作为验证。效果如下

诗人	模型预测结果(相似度)
王彦泓	0.8653275
丘逢甲	0.1216295

王彦泓的七律作品，模型给出的预测结果的平均值约为0.86，由于孙原湘《天真阁外集》七律作品被标注为1，王彦泓0.86的平均作品与1很接近，也就是说，与孙原湘《天真阁外集》的七律作品很接近。而王彦泓本人正以艳体诗出名，模型预测结果与现有论断相符。

而丘逢甲的七律"开满劲弓，吹裂铁笛"[33]，与艳体诗相去甚远，于是模型给出的预测结果只有0.12，这也符合其作品的风格。

5.5唐诗风格识别

5.5.1问题意义

唐诗宋诗正如太极之两仪，明清诗人不宗唐则宗宋，余者寥寥。纵有王闿运等宗汉魏六朝者，也无法影响大势。由于唐诗在后世的影响巨大，本文决定训练出一个模型来判断一首诗是否与唐诗相似，或者说有唐诗的风格。由于唐诗体裁多样，歌行，五古，乐府，七绝，七律，五绝，五律，不同的体裁纵使是一个诗人所写，风格也不一样。于是本文选择了风格较为鲜明的七言律诗进行研究。

5.5.2数据预处理

首先将所有搜集到的唐代七言律诗抽取出来，计7763首，标注为1。又在与唐诗风格差异较大的宋代七律中抽取相同数量的作品作为对照，标注为0。将两者混合，随机抽取10%作为验证集，剩余90%作为训练集。将全部文本数据进行向量化后输入模型。

5.5.3训练

使用先前的模型进行训练，由于数据较多，训练20轮后，验证集准确率约为0.8，损失值约为0.9，为避免过拟合，停止继续训练。

5.5.4验证

- 明七子

宋后学唐影响最大的莫过于明七子，他们提出“文必秦汉，诗必盛唐”的文学主张[34]，在诗歌创作上对唐人进行了过度的模拟。本文选出前后七子中较有代表性的几人进行测试——

诗人	模型预测结果(相似度)
李梦阳	0.6254571
何景明	0.7542985
李攀龙	0.7922501
谢榛	0.7385051
王世贞	0.5985163

从这几位明七子的代表人物的测试结果可以看出，他们的七言律诗有很重的唐人风味，相似度平均在0.7左右。王世贞略低，但也接近0.6，根据钱锺书先生的理论：“弇州於嘉靖七子，实为冠冕；言文必西汉，言诗必盛唐。《四部稿》中，莫非实大声弘之体. 然弇州《续稿》一变矜气高腔，几乎剟言之瘢，刮法之痕，平直切至。屡和东坡诗韵……则是弇州早作已染指苏诗矣。虽词气尚负固矜高，不肯遽示相下，而乃心则已悦服。”[32]，可以看出王世贞晚年诗染宋调，这可能是他的七律与唐诗相似度略低的原因。

- 江西诗派

江西诗派虽然主张学习杜甫，韩愈等人，但却是典型宋代诗风的代表。钱锺书先生指出“唐诗、宋诗，亦非仅朝代之别，乃体格性分之殊，天下有两种人，斯分两种诗。唐诗多以丰神情韵擅长，宋诗多以筋骨思理见胜……故唐之少陵、昌黎、香山、东野，实唐人之开宋调者”[32]，根据这种划分，杜甫韩愈等人虽是唐人，作品却与大多数唐人的作品风格不同，算是宋诗诗风，而学习杜甫韩愈等人的江西诗派，作品也与唐诗风格有很大差异。我们选取了江西诗派“一祖三宗”中的“三宗”进行验证

诗人	模型预测结果(相似度)
黄庭坚	0.3392725
陈与义	0.2314215
陈师道	0.2883462

结果符合现有文学史论断，作为宋诗代表人物，黄庭坚等人的七律作品与唐诗存在较大差异。

- 同光诗派

同光体是近代诗派之一。主要代表人物有陈三立，郑孝胥，陈宝琛，陈衍等人。陈衍宣称此派诗人特点为“同、光以来诗人不墨守盛唐者”[35]，但根据钱仲联先生的说法，同光体“远承宋代江西派而来,以黄庭坚为宗祖”[36]，也就是说，同光体诗人以学宋为主，由于数据不足，只选陈三立，郑孝胥，陈宝琛几位同光巨擘进行验证——

诗人	模型预测结果(相似度)
陈三立	0.33115724
郑孝胥	0.33334845
陈宝琛	0.26346013

结果与江西诗派三宗相似，由此可以看出同光体诗人学宋的主张在他们的七律创作中确实有所体现。

第六章 模型效果评估——类图灵测试

6.1 测试规则

为了验证模型生成的诗歌作品质量如何，本文模仿图灵测试出了一套测试题。流程如下：

- 选取文学史上重要诗人（代表人类较高水平，否则模型胜之不武）
- 从这些诗人的作品中选取一些不出名的作品（避免有人读过而影响实验结果）
- 根据体裁，每种体裁选4首。
- 选定后，输入首句令模型生成作品，将模型生成的作品与原作放在一起二选一，让人判断哪首是AI创作。

6.2 测试结果

根据上述流程，本文设计了一套测试，共16题，五言绝句，七言绝句，五言律诗，七言律诗各4题。测试在小范围产生了爆炸式传播，并被“少年国故微刊”等公众号转载。最后收到616份有效回答，参与测试人员有北大复旦等名校的专业研究者，也有各大诗社社员等格律诗创作者，总体来看受测人员在诗词方面造诣明显高于常人。结果汇总如下：

诗人作品	AI作品	正确人数	错误人数	正确率
孟郊：欲别牵郎衣，问郎游何处。不恨归来迟，莫向临邛去。	欲别牵郎衣，郎去不可住。愿作西飞翼，随风到郎处。	421	195	68.34%
钱谦益：一别三千里，相看七十年。明朝数行泪，沾洒各山川。	一别三千里，重来十二年。壮怀空老矣，旧话只依然。	282	334	45.78%
范云：自君之出矣，罗帐咽秋风。 思君如蔓草，连延不可穷。	自君之出矣，风雨正萧萧。 思君如杨柳，萦树复垂条。	408	208	66.23%
黄景仁：江水绕孤村，芳菲在何处？ 春从啼鸟来，啼是春归去。	江水绕孤村，疏烟隔荒树。 何处问津人，前溪唤渔父。	251	365	40.75%
金和：秋气入残夜，萧疏梦不成。 半窗残月影，千里故乡情。 露冷萤无火，天寒雁有声。 筹边当此地，还欲请长缨。	秋气入残夜，寒衾梦不成。 薄衣临槛落，络纬近阶鸣。 老意悲残岁，新愁集短檠。 更堪吟未稳，萧飒已三更。	284	332	46.1%
梁辰鱼：五月荆溪上，惊湍走白沙。 风樯敲听燕，浪楫远冲花。 山势通吴直，溪流向月斜。 维舟见渔火，深树有人家。	五月荆溪上，新蒲绿未齐。乱花翻野草，啼鸟歇高低。江阔山无定，天低月在溪。 谁怜倦游客，长此倚岩栖。	256	360	41.56%
刘基：疲马怀空枥，征衣怯路尘。 那堪远游子，复送欲归人。 月满西津夜，花明北固春。 论文应有日，话别莫悲辛。	疲马怀空枥，征衣怯晓寒。 山城看渐近，湖水去应宽。 村火孤烟静，渔舟宿雾残。 归心随去雁，千里正漫漫。	339	277	55.03%
孙原湘：雨拨西风转，天容寸碧开。 出花孤蝶重，抱树一蝉哀。 爽气遥山下，愁心远渚来。 画檐云缕缕，秋叶是谁裁？	雨拨西风转，江澄夜气收。 露零千树叶，月照满林秋。 远岸依沙没，飞萤逐水流。 遥空云影乱，孤屿一灯幽。	329	287	53.41%
陈子龙：独起凭栏对晓风，满溪春水小桥东。 始知昨夜红楼梦，身在桃花万树中。	独起凭栏对晓风，烟光离离日色红。 江山无限英雄泪，都在东流一水中。	245	371	39.77%
何景明：杨柳花飞芜草青，故乡南望几长亭。 城边客散重回首，愁见孤鸿落晚汀。	杨柳花飞芜草青，野塘烟草自凋零。 一双白鹭来烟际，点破遥山数抹青。	332	284	53.90%
李于楷：垂柳金堤千万丝，郊原风日暮春时。 联床若话巴山雨，莫道归来未有期。	垂柳金堤千万丝，行人攀折赠蛾眉。 愁心不与春风改，杨柳飞花无定时。	296	320	48.05%
聂守仁：好梦惊来欲断魂，空怜茹苦守蓬门。 伤心缕述死生别，剩有牛衣泪上痕。	好梦惊来欲断魂，何堪孤枕伴黄昏。 空闺听得萧骚雨，镫影摇寒自掩门。	296	320	48.05%

诗人作品	AI作品	正确人数	错误人数	正确率
唐寅：夕阳芳草笛悠悠，春尽惊看又转头。 淅沥风光摇草树，骖驔时节逐川流。 临阶思数脂千片，绕树空烦绣半钩。 九十诏光梭脱手，多情又作一番愁。	夕阳芳草笛悠悠，江南八月雁初秋。 千里故园谁是主，一声残笛起离愁。 不眠霜月明如洗，无用年华去莫留。 独倚栏干望天末，断鸿飞尽水东流。	297	319	48.21%
顾炎武：白下西风落叶侵，重来此地一登临。 清笳皓月秋依垒，野烧寒星夜出林。 万古河山应有主，频年戈甲苦相寻。 从教一掬新亭泪，江水平添十丈深。	白下西风落叶侵，高秋踪迹感登临。 浮云万里看秋色，明月孤尊惜暮阴。 海岱一身栖伏枕，河梁孤剑到家心。 他时倘有南来雁，迟尔南来慰苦吟。	361	255	58.60%
陈三立：寻常节物已心惊，渐乱春愁不可名。 煮茗焚香数人日，断笳哀角满江城。 江湖意绪兼衰病，墙壁公卿问死生。 倦触屏风梦乡国，逢迎千里鹧鸪声。	寻常节物已心惊，况是残春百感生。 万里客途惊岁晚，一番节物忆家行。 青衫去国当年泪，白发临川隔岁情。 病拥青毡还未得，东风愁见马蹄声。	277	339	44.97%
王彦泓：一寸心期百尺楼，明河界作两边秋。 移开月扇朝云出，掩过银屏夜月收。 鬓态易迷花影乱，衣香暗接水光浮。 残阳没后寒灯小，各自垂帘背雨愁。	一寸心期百尺楼，看山终日兴悠悠。 江湖万里鸥波险，鸿鹄三春禁苑秋。 别后梦随沧海月，清晨吟傍白云楼。 知公有意思鲈鲙，乞我严滩理钓舟。	286	330	46.43%

根据问卷情况来看，大部分题目正确率都在50%左右波动，低于40%或高于60%的只有三题。总的来看，测试人员共做了9856次选择，其中正确了4960次，约占整体比例的50.32%，极其接近50%，因此我们得出结论，即使对于比较专业，诗词方面造诣较高的人，也很难分辨本模型根据首句生成的作品与原作哪首是真正的诗人创作的。

然而从计算机的视角来看，AI创作的诗歌还是有特点的：AI的原理是统计学，换言之，训练数据中大多数作品是怎么写的，它大概率就怎么写。所以AI创作的作品一般没有生僻字，也大概率不会有独特少见的风格。测试本来想蹭一蹭丘逢甲，结果发现丘逢甲的作品长枪大戟，个人风格浓烈，根本不是AI可以碰瓷的。

另外AI作诗缺少人的感发，常中陈衍所谓"妙手空空"之病，缺少明确的指向性，可彼可此。

目前来看AI的作品由于缺少感发，句间逻辑不严等原因，与一流作品尚有较大差距。

现有的AI创作的本质都是数学运算，不带任何情感与思考。对于诗歌这种艺术创造来讲，除非AI能做到模拟人的情感与思想，否则无法跟一流的作品比肩。而且AI生成内容是靠计算，有计算必有误差，而误差又字字积累。所以AI生成短一些的文本，如律绝，有时还可以掩人耳目，因其字少，误差就小，效果也好。稍长一些的古体就完全不够看，字数一多，积累的误差就大了。所以AI暂时连长点的诗都写不了，更别提超越人类了。

第七章 总结与展望

7.1 诗歌生成模型

自然语言处理领域的技术日新月异，AI生成的诗歌质量也水涨船高。本文所研究的诗歌生成模型，已经能写出高水平研究者都无法分辨的作品。当代我国诗词创作人数已达500万，但囿于从小接受的教育问题，相当一部分当代的创作者缺少语感，在字句，技巧上欠缺良多。本文的模型可以给他们以帮助，在他们遣词造句时，可以参考本模型根据他们现有的诗句生成作品，来创作自己的诗歌。诗歌的意义在于作者的思想情感，AI的诗写的再好也是没有意义的——它唯一的意义是让人看到且给人以启发。

当然本文的工作还存在一些伦理问题：有些诗词创作者对有人可能会拿AI创作的诗词去参加诗赛表示忧虑。本文对此也没有太好的方法，唯一能做的就是呼吁九歌，诗三百这种诗歌生成网站将其模型生成的历史作品开放出来，这样一查就可以判断一首诗词作品是本人写的还是这些网站的诗歌生成模型生成的。

7.2诗风判别模型

虽然大多数诗人的诗歌作品都是出自己手，有自己的特点与原创性，但不可否认的是，我们如今对某位诗人进行研究总还要找出他的诗学脉络，然后断言这位诗人学杜韩，学苏黄，学江西派，学西昆体，乃至学唐，学宋。本文的工作意义在于寻找到了一种通用的，且消除了主观误差的方法，来判断某篇作品与一类作品是否相似。这可以作为古典诗歌研究者用于寻找诗人取径的一个方便的工具，也可以与现有的研究成果互相参照。像袁枚这种标榜自己诗学独树一帜，与古人无关，乃至扬言道"独来独往一枝藤，上下千年力不胜。若问随园诗学某，三唐两宋有谁应。"，竭力掩饰自己诗学取径的诗人，我们也可以用这个工具来分析他的诗风与谁比较相似，遗憾的是，我们没有得到袁枚诗歌的数据，故而没有做这项研究。

当然，本文提出的工具不仅可以用来判断一位诗人的诗学取径，还可以对一个诗派，甚至一个时期的诗学创作形式进行研究。比如同光体各路名家各有所本，不过都是以宋诗为纲，所以后人也提出过"同光无体"的说法。到底同光体各位诗人之间差异大不大？他们能不能算作同一个诗派？学者使用这个工具可以判断出他们之间的相似性来辅助研究。再如竟陵派承继了公安派的"性灵说"，并对其进行了一定的修正，那么竟陵派与公安派的诗歌创作差别有多大？诗风有多像？借助本文的模型，也可以一窥究竟，希望本文的工作对此类问题的研究有所裨益。

参考文献

[1]http://150.158.30.163/gen
[2]王国维.2008.宋元戏曲考[M].上海:上海世纪出版集团
[3]https://sou-yun.cn/PoemIndex.aspx
[4]戴衍.跨世纪的古籍整理工程——《全明诗》[J].中国典籍与文化,1997(01):101-104.DOI:10.16093/j.cnki.ccc.1997.01.018.
[5]马大勇.清代诗词研究现状刍议[J].文史知识,2009(05):155-159.
[6]马大勇，赵郁飞.中华诗词发展报告2020|诗词创作（年度创作生态概观）.中华诗词研究院,2021-12-31
[7]Pablo Gervás. An Expert System for the Composition of Formal Spanish Poetry, pages 181–188.Springer London, 2001.
[8]Hisar Maruli Manurung. An evolutionary algorithm approach to poetry generation. PhD thesis, University of Edinburgh, 2003.
[9]Jing He, Ming Zhou, and Long Jiang. Generating chinese classical poems with statistical machine translation models. In Proceedings of the 26th AAAI Conference on Artificial Intelligence, pages 1650–1656, Toronto, Canada, 2012.
[10]http://jiuge.thunlp.org/
[11]https://www.aichpoem.net/#/shisanbai/poem
[12]Xiaoyuan Yi and Maosong Sun and Ruoyu Li and Zonghan Yang.Chinese Poetry Generation with a Working Memory Mode.Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, pages 4553--4559,Stockholm, Sweden, 2018.
[13]Cui, Yiming and Che, Wanxiang and Liu, Ting and Qin, Bing and Yang, Ziqing.Pre-Training with

Whole Word Masking for Chinese BERT.IEEE Transactions on Audio, Speech and Language Processing, 2021
[14]https://github.com/Werneror/Poetry
[15]苗艳艳. 计算风格学简述[J]. 商品与质量:消费研究, 2015.
[16]肖天久,刘颖.基于聚类和分类的金庸与古龙小说风格分析[J].中文信息学报,2015,29(05):167-177.]
[17] BENGIO,YOSHUA,DUCHARME, et al. A Neural Probabilistic Language Model.[J]. Journal of Machine Learning Research, 2003.
[18]MIKOLOV T,CHEN K,CORRADO G, et al. Efficient Estimation of Word Representations in Vector Space[J].Computer Science,2013.
[19] PENNINGTON J,SOCHER R, MANNING C. Glove: Global Vectors for Word Representation[CV/Conference on Empirical Methods in Natural Language Processing.2014.
[20]Radford A, Narasimhan K, Salimans T, et al. Improving language understanding by generative pre-training[J]. 2018.
[21]Vaswani A, Shazeer N, Parmar N, et al. Attention is all you need[J]. Advances in neural information processing systems, 2017, 30.
[22]DEVLINJ,CHANG MW,LEE K, et al.BERT: Pre-training of Deep Bidirectional Transformersfor Language Understanding[J]. 2018.
[23]GRAVES A. Generating Sequences With Recurrent Neural Networks[J]. Computer Science,2013.
[24]MALHOTRAP, VIGL,SHROFF G, et al. Long Short Term Memory Networks for Anomaly Detection in Time Series[C]// 23rd European Symposium on Artificial Neural Networks,Computational Intelligence and Machine Learning, ESANN 2015.2015.
[25]Hinton G E , Srivastava N , Krizhevsky A , et al. Improving neural networks by preventing co-adaptation of feature detectors[J]. Computer Science, 2012, 3(4):págs. 212-223.
[26]Zhao, Zhe and Chen, Hui and Zhang, Jinbin and Zhao, Xin and Liu, Tao and Lu, Wei and Chen, Xi and Deng, Haotang and Ju, Qi and Du, Xiaoyong.UER: An Open-Source Toolkit for Pre-training Models.EMNLP-IJCNLP 2019, pages 241,2019
[27]https://github.com/lancopku/PKUSeg-python
[28]http://thulac.thunlp.org
[29]https://github.com/garychowcmu/daizhigev20
[30]Li, Shen, Zhao, Zhe, Hu, Renfen, Li, Wensi, Liu, Tao, Du, Xiaoyong.Analogical Reasoning on Chinese Morphological and Semantic Relations.Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 138--143, Melbourne, Australia, 2018
[31]Salton G, Wong A, Yang C S. A vector space model for automatic indexing[J]. Communications of the ACM, 1975, 18(11): 613-620.
[32]钱锺书.1984.谈艺录[M].北京:中华书局.
[33]潘飞声.2016.在山泉诗话校笺[M].北京:人民文学出版社
[34]张廷玉.1974.明史·李梦阳传[M].北京:中华书局
[35]陈衍.2001.陈石遗集.福州:福建人民出版社
[36]钱仲联.近代诗坛鸟瞰[J].社会科学战线,1988(01):267-276.