

学号 : 2016301500030

密级 : _____

武汉大学本科毕业论文

图像-语义互译系统

院(系)名称 : 弘毅学堂

专业名称 : 计算机科学与技术

学生姓名 : 雷伯涵

指导教师 : 庄越挺 教授

黄 浩 副教授

二〇二〇年五月

BACHELOR'S DEGREE THESIS

OF WUHANUNIVERSITY

An Image-Languange Traslation System

College : Hongyi Honored School
Subject : Computer Schience and Technology
Name : Lei, Bohan
Instructor: Prof. Zhuang, Yueting
Asso Prof. Huang, Hao

MAY 2020

郑 重 声 明

本人呈交的学位论文，是在导师的指导下，独立进行研究工作所取得的成果，所有数据、图片资料真实可靠。尽我所知，除文中已经注明引用的内容外，本学位论文的研究成果不包含他人享有著作权的内容。对本论文所涉及的研究工作做出贡献的其他个人和集体，均已在文中以明确的方式标明。本学位论文的知识产权归属于培养单位。

本人签名: _____ 日期: _____

摘要

请使用中文分号“；”分割关键词！

关键词: 关键词 1; 关键词 2; 关键词 3

ABSTRACT

Please use English semicolon and space to separate key words.

This is abstract. This is abstract. This is abstract. This is abstract. This is abstract.

This is abstract. This is abstract. This is abstract.

This is abstract. This is abstract. This is abstract. This is abstract. This is abstract.

This is abstract. This is abstract. This is abstract. This is abstract. This is abstract. This is abstract.

Key words: Key1; Key2; Key3

目 录

1 緒論	1
1.1 相关背景与需求	1
1.2 技术应用意义与前景	1
1.2.1 研究意义	1
1.2.2 研究前景	2
1.3 相关工作	2
1.3.1 图像翻译语义	2
1.3.2 语义生成图像	3
1.3.3 自然语言处理	4
1.4 本文工作目标	5
1.4.1 设计大纲构思	5
1.4.2 本文篇章结构	5
1.5 本章小结	6
2 相关技术理论	7
2.1 神经网络技术	7
2.1.1 神经网络	7
2.1.2 循环神经网络	7
2.1.3 长短期记忆网络	8
2.2 GAN 生成模型选取使用	9
2.2.1 基本原理	9
2.2.2 基础 GAN 模型的缺陷	10
2.2.3 衍生模型分类与特点	10
2.2.4 StackGAN 及其衍生模型	11
2.3 本章小结	14
3 方案设计	15

3.1 概要	15
3.2 软件模型	15
3.3 变量符号与定义	15
3.4 软件界面设计	16
3.4.1 软件例图	16
3.4.2 软件开发模型	17
3.5 软件功能设计	18
3.5.1 图片标注方法	18
3.5.2 自然语言生成图片方法	22
3.6 本章小结	25
4 实验与分析	26
4.1 实验环境	26
4.1.1 模型训练环境	26
4.1.2 软件样本运行环境	26
4.2 图片标注实验	26
4.2.1 模型代码	26
4.2.2 运行过程	26
4.2.3 遇到的主要问题与解决方法	27
4.3 自然语言生成图片实验	27
4.3.1 模型代码	27
4.3.2 运行过程	28
4.3.3 遇到的主要问题与解决方法	28
4.4 实验分析与总结	29
4.4.1 图片标注表现	29
4.4.2 文本生成图像算法表现	30
4.5 本章小结	31

5 设计意义阐述与展望	32
5.1 我的设计意义	32
5.1.1 制作了一个简洁易用系统	32
5.1.2 活用知识、实践了深度学习技术	32
5.2 未来发展方向	32
参考文献	33
致谢	37
附录 A 部分代码	38
A.1 界面代码	38
A.2 图片标注主函数代码	38
A.3 图像生成主函数代码	38

1 緒论

1.1 相关背景与需求

随着社会发展与技术进步，我国目前越来越重视高效工具的研发。从 2000 年伊始到 2010 年互联网普及，再到 2020 年的现在，个人电子终端的功能已经越来越强大，每个人都需要更新期、更有用的效率工具。近五年，图片标注技术与生成模型都有爆炸式的发展；近两年，视觉问答技术 (Visual Question Answering)^[1]、视频标注技术与秒级的图片理解更是让失明群体有了“读懂光芒”的希望，也让图片的理解从学术界或产业界的科研层面有了走向应用、走向市场的可能。

对于看不见的人来说，读懂视野这一技术是他们改善生活质量的工具，画出语言这一技术是他们表达自我的窗口；对更多的普通人群体来说，一个简洁易用的新奇效率工具，更是一种生活方式的改变，让更多的人通过机器的“魔力”，换一种角度来看语言和图片。

1.2 技术应用意义与前景

1.2.1 研究意义

图像-语义的双向翻译在目前已经有了很大的需求面。最基本的就是，图像作为视觉信号，无法被视觉失能人群感知，可以将语义转化为听觉信号，方便视觉失能人群感知世界。更进一步，对于大量的图像信号，靠肉眼处理起来需要很大的人力成本，如果使用图像标注技术，则可以从图像中提取重要信息，对语义信息进一步处理，形成简报构成参考，辅助决策。图片标注的主要参考意义在于全局的图片理解，这一点算法可能比人类做的更好，因为人类更可能只把注意力放在局部上，而忽略一些自己不关注的次要信息。

语义的图像化更是意义非凡。基础地说，同样面对失能人群，没有视觉的人通过这一系统也有了“创作”的能力，将他们的思想从肉体的局限中有限地拓宽了出口；或者说，即使是有视觉能力的人，如果不擅长绘画，也可以通过这一技术直接表达自己思想中的画面。更进一步地说，想象力弱是很多成年人能力的局限，而一个语义向图像的转化，则可以使一个人的表述清晰、直观地呈现在他人的视野里，可以促进交流；而表述之人也可以根据可视化的表达，发现自己表达中的问题，及时予以修正，而这是我们生活中每个人都需要的。

可以说，生成图像不仅仅是作为观赏，它可以切真实地改变我们的生活方式。

1.2.2 研究前景

现在并没有简单易用的商业系统，可以提供语义与图像互译的简便功能，大部分系统都只能作为技术的样品，做单向的翻译工作，目前主要在各大展览会上起到展示企业技术实力之用。对于常常需要与图片、交流打交道的人来说，这样一套系统对工作效率提高很多；广泛地，对于任何一个人，这类系统都可以改变他的生活方式。

1.3 相关工作

首先需要明确，根据目前自然语言处理技术状况，语义的标注和自然语言的生成在操作上的难度没有很大的区别。在设计中，我会使用自然语言作为交互的媒介，方便用户使用、直观感受技术的表现。

1.3.1 图像翻译语义

1.3.1.1 图片标注研究现状

在图片转化为文字工作的早期阶段，研究人用常用基于模式的方法来自动完成图像标注。在这种方法中，总是先捕捉图片中的各种可视信息，然后在固定的句式中填写这些信息，以填空的方法完成句子。用这种模型，很多算法都可以给出差强人意的输出语句，但是这样做的弱点是，输出遵循既定的模式和概念，输出的多样性很差，语言僵硬。

为了解决上述问题，有一些研究中使用了检索式方法来输出可变长度的语句^[2, 3]。这种方法是通过组合从图片中捕捉到的短语来输出图片标注结果。很明显，这种方式生成的语句的句式更为灵活，信息更为饱满，但是生成的语句一般限定于被检索的图片库，对一些特定的图片无法生成正确的结果。

近期由于机器翻译中的大进步，图像标注领域也开始应用了编码器解码的架构^[4-7]。这种框架可以运用深度神经网络来生成每张图片新的内容语义，比过去的方法更为准确；在这种框架下，图片被编码神经网络编码为中间表示形式，而可以用解码 RNN 网络将其直接翻译为自然语言。Vinyals 提出一种神经图像标注器

的方法 (Neural Image Caption Generator, 称作 Google NIC)^[8], 可以提取目标图片的可视特征, 用一个长短期记忆模型来得出图像标注。

1.3.2 语义生成图像

将语义转化为图像原本是画家的专利, 现在运用模型也可以用计算机自动生成图像。生成式模型原本不能生成图像, 到了 GAN 网络发展到一定程度, 才出现了自动生成图像的概念。

生成方法、判别方法分别时机器学习的两大分支方向, 而生成式模型则是用生成方法来生成样本的一类模型。传统的生成式模型设计比较简单。有一类生成式模型是从人类理解角度进行设计的, 比如说最大似然估计法、近似法^[9, 10] 与马尔可夫链法^[11-13] 等, 这一类方法对于机器来说各有限制。最大似然估计法的参数更新直接受限于数据样本, 数据样本不够丰富会限制生成模型的结果; 近似法的目标函数太过复杂, 算法只能逼近目标函数下界; 马尔可夫链法的缺点便是复杂度过高。从机器理解角度设计的算法一般不直接进行拟合或者估计, 而是通过采样数据样本调整模型, 一般这种方法人类无法直接理解, 但是生成样本是人类可以理解的。

主流的生成图像方法为 GAN 网络, 即生成对抗网络 (Generative Adversarial Networks), 由由 Goodfellow et al. 在 2014 年首次提出^[14]。目前, 它已经发展成了生成式神经网络最大的热点, 其研究得到了长足的发展。短短几年之内, 已经有了百余种 GAN 网络的衍生模型, 其应用范围囊括了包括自然语言、图像处理、计算机视觉在内的各个领域。

生成式对抗网络的出现为计算机视觉应用提供了新的技术和手段, 它以独特零和博弈与对抗训练的思想生成高质量的样本, 具有比传统机器学习算法更强大的特征学习和特征表达能力。目前在机器视觉领域尤其是样本生成领域取得了显著的成功, 是当前研究的热点方向之一。GAN 的不同模型在生成样本质量与性能上各有优劣。当前的 GAN 模型在图像的处理上取得较大的成就, 能生成以假乱真的样本, 但是也存在网络不收敛、模型易崩溃、过于自由不可控的问题。

目前很多人用这一网络完成了有趣的小应用, 包括模糊图片增加清晰度等, 但是它最神奇的用法还是直接生成图像, 用语义可以直接生成对辨别器不可分辨的图像。

1.3.3 自然语言处理

1.3.3.1 背景介绍

自然语言处理 (Natural Language Processing, NLP) 是指计算机对自然语言的处理算法。所谓自然语言，就是说人类在社会文明发展过程中为了交流而逐渐发展出的语言，例如中文、英文、日文，甚至包括手语，都是自然语言。因为自然语言之间的关系远远不是函数映射那么简单的事，所以这个领域经历很长的发展历程才到了今天相对成熟的地步。

自然语言处理的发展历程大致分了三个阶段。第一个阶段是上个世纪后半叶，在二战之后洛克菲勒仅仅会的瓦伦·威佛等人在展望计算机技术的未来应用时，认为计算机可以用不同语言之间的简单词汇替换来完成翻译工作。在现在国际化知识丰富的我们来看，显然可以看出这种方法是行不通的，但是上世纪后期，大量学者耗费人力物力，建立语言之间的词典，为每一个词汇做了映射，以实现计算机翻译。但是翻译结果并不理想，教训便是自然语言的理解需要考虑上下文关系。第二阶段是世纪之交的二十年。计算机技术迅猛发展，学者开始发现仅靠统计和替换无法完成翻译工作，而且神经网络也初步成熟，人工智能再次登上时代的焦点。例如，2005 年，Pradhan^[15] 提出了语义角色解析的机器学习算法，使用 SVM 技术扩展了前人的工作^[16, 17]，改进了数据的泛化功能。

第三阶段是近十年，大数据的积累帮助了自然语言处理的语料库丰富，促进了算法表现的提升；另一方面，音视频处理技术中深度学习的应用更为成熟了。其应用例如：2016 年，Y Yao et al.^[18] 提出了一种基于长短期记忆的中文分词方法，突破了上下文窗口大小限制，可以保存前文的重要信息；2015 年 Jie Zhou et al.^[19] 提出了不使用解析的端到端 SRL 学习系统来分析语义，使用双向 RNN，只使用文本作为输入功能，不使用语法知识。

1.3.3.2 自然语言应用例子：IBM Watson

2011 年，在美国的一档综艺节目中，IBM 公司云服务器 bluemix 平台上的 Watson 板块 NLP 技术相关产品大放异彩，当时因出色的对话表现爆红。

在 IBM bluemix 云平台上的 Watson 板块，有多语言对应的多种 NLP 相关应用，包括语音识别、语音生成等，经过实际试用，发现目前制作水平比较完善，但是中文包的翻译效果比较差。经过调研发现，中文的翻译训练语料库主要是使用

专业文件书籍构成，导致语言不够丰富。这也给了我们一个教训，在实际 NLP 训练中，一定要使用比较丰富、贴近日常生活的语料库。

本文设计以英语为基础，使用了完整、丰富的 MSCOCO-2014 数据集。有八万多张图片作为训练集，另有四万多张图片作为测试集，且均带有相应的语义标注。这样完整的数据集更贴近一般语言环境，可以得到更好的训练效果。

1.3.3.3 基础依赖包

目前，常用的 NLP 算法在 python、JAVA 等语言中里面都有依赖包可以使用。

在我使用的 python 语言 3.7 版本中，可以使用依赖包 nltk(Natural Language Toolkit)^[20]，这个包实现了超过五十种自然语言处理的基本算法，可以进行分词、语义角色解析等。

1.4 本文工作目标

1.4.1 设计大纲构思

本次设计需要实现如下功能，并达到要求：

1. 实现图像生成语义算法，并达到可用的效果；
2. 实现语义生成图像算法，并且图像对一般人清晰可识别；
3. 设计一个有机统一的系统，功能简洁易用。

1.4.2 本文篇章结构

本文共分为五个章节。

第一章，介绍了研究的背景，并通过阅读文献，整理了相关的工作，从中找出比较适合于本设计的工作，并加以总结。

第二章，详细介绍了本设计所使用技术的详情，并说明了选取技术原因。

第三章，设计了实验的方案，通过一些前人制作好的开源项目，加以修改、总结，设计系统。

第四章，详细记述了实验的结果，并通过测评手段进行测评，分析了实验结果。

第五章，阐述了设计的意义，并且展望了未来系统进一步扩展迭代的方向，指明了系统的局限性。

1.5 本章小结

本章介绍了设计相关领域的需求背景，也全面地介绍了这一设计的意义与前景。接下来介绍了涉及到的相关技术发展现状和相关领域的工作，也说明了其中最适合于本设计的技术及其原因。最后说明了本设计的基本要求和构思，另外设计了篇章结构，也设定了工作的目标。

2 相关技术理论

2.1 神经网络技术

本次设计中图片标注部分主要使用的技术基于长短期记忆循环神经网络。这一小节将介绍循环神经网络和长短期记忆的相关技术。

2.1.1 神经网络

神经网络 (Neural Networks) 是近年比较流行的概念。在 2006 年 Hinton^[13] 提出深度学习概念后，成为了优化算法表现性能的一大利器。

神经网络的基本结构就是由神经元构成的网络，每个神经元结构如图 2.1 所示，输入输出关系如式 (2.1) 表达结果。

$$f\left(\sum_{i=1}^m w_i x_i + b\right) \quad (2.1)$$

对于第 j 层的每一个神经元，它会对上一层的每一个神经元输入的 x_i 赋予权重 w_i ，计算出输出结果，即如式 (2.2) 所示，其中 n 是每一层的神经元个数，由输入神经元个数决定。

$$x_k^j = \sum_{i=1}^n w_i k x_i k, k \in [1, n] \quad (2.2)$$

2.1.2 循环神经网络

循环神经网络 (Recurrent Neural Networks, RNN) 被广泛地运用在图像处理和自然语言处理上。与一般的神经网络相同，它也有前馈层和反馈层，但是它在一般的神经网络基础上增加了基于时序的循环机制。在一般的神经网络中，神经元之间相互独立，但是现实中的信息往往互相关联。所以，循环神经网络的特点像人一样有记忆的能力它的输出不仅依靠当前输入，也依靠历史的输出。在一定程度上，它更能反应真实的情况。

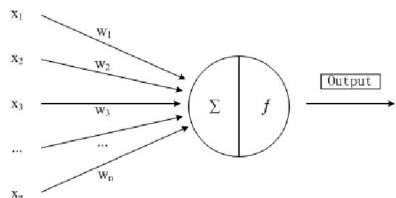


图 2.1 神经网络神经元

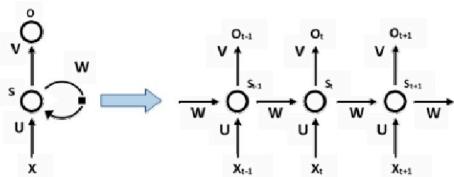


图 2.2 循环神经网络神经元

RNN 神经元的结构与 BP 神经网络神经元有些许不同。如图 2.2 所示，任一隐藏层中神经元的输出信息依靠当前时刻的输入信息和前序时序的输出信息决定，同时当前时序的输出信息也会作为下一个神经元的输入。这样的神经网络构建了神经元之间的联系，让数据有了更为有效的依赖关系，优化了信息利用的效率。每一个神经元的计算方式可以表述为式 (2.3)。其中 x_t 代表 t 时刻神经元的输入， s_t 代表 t 时刻神经元的状态， o_t 代表 t 时刻神经元的输出。

$$s_t = f(u \cdot x_t + w \cdot s_{t-1}), \quad (2.3)$$

$$o_t = g(s_t \cdot v).$$

2.1.3 长短期记忆网络

从主体上看，LSTM 与 RNN 类似，都是循环神经网络这样的链式结构。而在神经元上看，LSTM 细胞节点要比普通的 RNN 节点要复杂一些。LSTM 改进后的神经元由四个不同的神经网络层进行信息的交互，如图 2.3 所示。

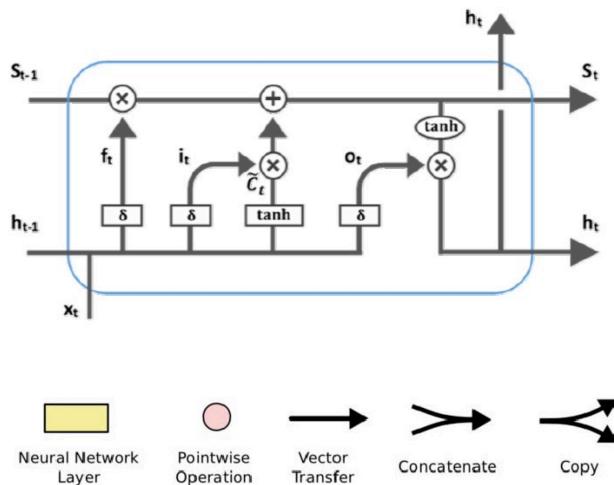


图 2.3 LSTM 神经元细胞结构图

其中圆形代表基本操作，矩形代表学习得到的状态，箭头代表数据的传输，而箭头的分裂与合并则代表向量的拆分与连接； s 代表状态， x 代表输入， h 代表输出。

在图 2.3 中的上方长线段代表了神经网络的状态 s ，它在每个时刻的操作仅在两个地方与输入数据有交互，这就保证了 s 的变化不大，神经元状态比较稳定。LSTM 神经元的数据更新由“门”来进行，三个门分别是输入门、遗忘门和输出门，LSTM 神经元由此实现数据的更新和储存。

门上的行为由一个点乘 Pointwise 操作和一个激活函数 *Sigmoid* 所组成，该函

数取值 0 至 1，分别代表着允许部分信息通过——其中 0 代表不允许任何信息通过，1 代表着允许所有信息通过。具体的门信息传递机制由以下几个部分完成：

1. 遗忘门：决定历史信息的保留与遗忘。通过 *Sigmoid* 函数来决定状态 S 的遗忘程度。

$$f_t = \sigma(W_f[h_{t-1}, x_t] + b_f) \quad (2.4)$$

2. 输入门：决定存储到细胞状态内的新信息。先用 *Sigmoid* 函数决定记忆的更新程度，然后决定更新的内容，最后将更新的内容加在遗忘内容之上，形成当前时刻的状态。

$$\begin{aligned} i_t &= \sigma(W_i[h_{t-1}, x_t] + b_i) \\ \tilde{C}_t &= \tanh(W_c[h_{t-1}, x_t] + b_c) \\ S_t &= f_t \cdot S_{t-1} + i_t \cdot \tilde{C}_t \end{aligned} \quad (2.5)$$

3. 输出门：决定输出的值。首先用 *Sigmoid* 函数决定从状态中输出的部分，然后用 \tanh 对细胞状态进行处理，并与 *Sigmoid* 函数值相乘，实现输出想输出的部分。

$$\begin{aligned} o_t &= \sigma(W_o[h_{t-1}, x_t] + b_o) \\ h_t &= o_t * \tanh(S_t) \end{aligned} \quad (2.6)$$

通过这三个门对神经元的共同作用，LSTM 可以完成神经元的基本状态变换，从而完成较好的训练效果。

2.2 GAN 生成模型选取使用

2.2.1 基本原理

GANs 的基本结构即由一个生成模型 G 和一个判别模型 D 组成。生成模型 G 的目的是尽可能最小化生成与真实训练样和生成样本的区别，而判别模型 D 的目的则是尽可能最大化地找出真实样本和生成样本之间的区别。

通过多轮迭代生成模型 G 和判别模型 D 的对抗，可以使两个模型都达到上述的目标效果。当训练判别模型 D 的时候，希望输入真实样本 x 可以使判别器对其的判断 $D(x)$ 尽量趋于 1，而生成样本 $G(x)$ 通过判别器 D 的时候可以使得 $D(G(x))$ 尽量趋于 0。在训练生成模型 G 的时候，输入噪声 z ，希望生成的生成样本通过判别器 D 的时候尽量使得 $D(G(z))$ 趋于 0。

可以用简单的数学变换得到公式 (2.7)，来描述训练过程。

$$\min_G \max_D V_{G,D} = \mathbb{E}_{x \sim P_{data}(x)} [\lg D(x)] + \mathbb{E}_{z \sim P_G(z)} [\lg(1 - D(G(z)))] \quad (2.7)$$

其中 $P_{data}(x)$ 为真实图片集的分布。

当多轮博弈过后，极大极小问题达到最优解，即纳什均衡，当且仅当 $P_z = P_{data}$ 时^[14]。这时 $\mathbb{E}D(G(z))$ 趋于 $\frac{1}{2}$ ，即相当于只能随机猜测 0 与 1，而生成模型 G 学会了真实样本的特性。

相对于传统的生成模型，我们可以发现 GANs 模型并不需要使用马尔可夫链，学习过程不需要近似推理，也不需要预先训练，自由度比较高，可以利用反向传播计算梯度，很好地利用了分段线性单元的优势，而可以回避近似计算的困难概率问题。

2.2.2 基础 GAN 模型的缺陷

GANs 模型的缺点对应着优点，比较明显。由于 GANs 模型自由度太高，在面对过于清晰的图片等训练样本时，收敛性表现较差，生成模型可能出现退化，重复生成相同样本点，导致判别器无法工作，进而导致模型崩溃。因此，在训练过程中，调整好两个模型网络的平衡与同步非常重要。

正因为 GAN 模型有着这诸多的缺点，所以要选取比较合适的 GAN 模型来生成图片。

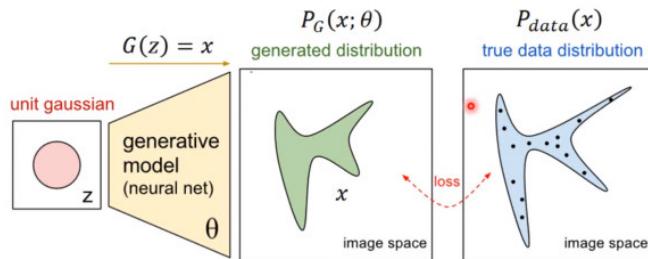


图 2.4 一般 GAN 网络模型结构

2.2.3 衍生模型分类与特点

目前对抗生成网络的衍生模型众多，其优化方式大抵由两个大方向衍生而来。一个方向是由损失函数的改变来优化 GAN 模型的效果，另一个方向则是从模型使用的角度来优化 GAN 模型的效果。在表 2.1 中，列举了一些最为常见的 GAN 模型衍生模型。

表 2.1

从损失函数角度提出的优化 GAN 模型 ^[21]	从模型应用角度提出的优化 GAN 模型	编码器角度	其他角度改进
网络构架角度 ^[22]			
Least Square GANs, Loss-Sensitive GAN, Fisher GAN, WGAN, WGAN-GP, WGAN-LP, f-GANs ^[1] DRAGAN 等	CGAN, DCGAN, InfoGAN, StackGAN ^[23] , AL-GAN 等	BEGAN, VAE-GAN, tDCGAN, BiGAN, 文献中的算法 ^[24-26] 等	LAPGAN, ESRGAN, SRGAN, 3D-GAN, MGAN 等

图像生成的模型与基于网络构架优化的 GAN 网络模型最为贴合，本文设计使用 StackGAN 作为基础，并针对相关实现进行优化。

2.2.4 StackGAN 及其衍生模型

Han Zhang et al. 在 2017 年提出了 StackGAN 模型，创新性地提出了基于栈的对抗式生成网络模型从文本生成图片的方法。

StackGAN 和 CGAN 同属于优化了 GAN 网络的构架结构的一类 GAN 模型的分支。其中，StackGAN 是对相对朴素的 CGAN 模型的优化。

2.2.4.1 从 CGAN 到 StackGAN

条件对抗生成网络 (Conditional GAN, CGAN) 是 Mirza^[22] 于 2014 年提出的，如公式 (2.8) 表述，对生成模型和判别模型输入文本信息作为条件，使用了一个隐藏层的全联通网络 (MLP) 结构，实现了通过文字标签生成图像的功能。

$$\min_G \max_D V(G, D) = \mathbb{E}_{x \sim P_{data(x)}} [\lg D(x_i | y)] + \mathbb{E}_{z \sim P_G(z)} \lg[(1 - D(G(z_i | y)))] \quad (2.8)$$

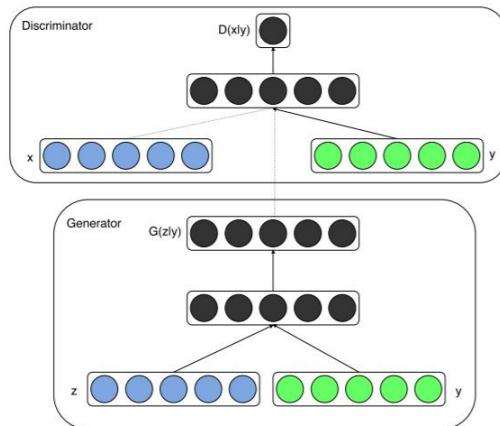


图 2.5 CGAN 构架示意^[22]

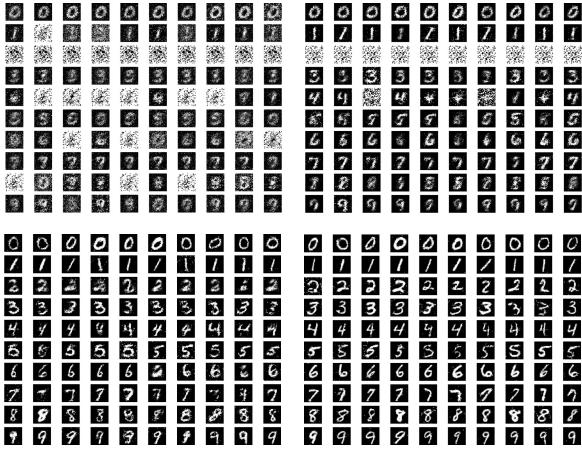


图 2.6 CGAN 生成手写数字^[22] (分别经历 1、10、100、1000 个 epoch 后的结果)



Figure 7: Generated samples

图 2.7 CGAN 生成二次元头像^[27]

但是，CGAN 的模型结构简单，效果差，收敛速度慢。在 CGAN 中，我们将文本作为条件输入生成器和判别器来训练两个模型生成和判别图像。如图 2.6 和图 2.7 所示，及时 CGAN 应用广且有趣，但图像较小，质量不高。最基础的 CGAN 算法只能生成 64×64 像素的图片，经过后来的改进，也只能将图片质量提升到 128×128 像素。

为了解决这一问题，Han Zhang^[23] 在 2016 年提出了 StackGAN 模型，使用基于栈的结构，分两步进行训练。

如图 2.8 所示，第一步先训练模型使模型生成 64×64 像素的图片，训练出的模型只能输出像素低、形状有所扭曲的图片。第一次训练的时候并不使用转化为向量的文字直接作为输入，因为这一向量为 1024 维，过于稀疏，不利于深度的训练。在经历一次 FC 以后进行输入，这样可以增强表现。第一阶段训练的损失函数为式 (2.9) 所示。第二次训练并不增加噪声输入，经过多轮残差网络迭代，模型修正一轮训练生成图片的错误，并且在图像中添加细节，生成 256×256 像素的“高清图片”。第二阶段训练的损失函数如式 (2.10) 所示。

$$\begin{aligned} \mathcal{L}_{D_0} = & \mathbb{E}_{(I_0, t) \sim P_{data}} [\lg D_0(I_0, \varphi_t)] \\ & + \mathbb{E}_{z \sim P_z, t \sim P_{data}} [\lg (1 - D_0(G_0(z, \hat{c}_0), \varphi_t))], \\ \mathcal{L}_{G_0} = & \mathbb{E}_{z \sim P_z, t \sim P_{data}} [\lg (1 - D_0(G_0(z, \hat{c}_0), \varphi_t))] \\ & + \lambda D_{KL}(\mathcal{N}(\mu_0(\varphi_t), \Sigma_0(\varphi_t)) \parallel \mathcal{N}(0, I)) \end{aligned} \quad (2.9)$$

$$\begin{aligned}
\mathcal{L}_D = & \mathbb{E}_{(I,t) \sim P_{data}} [\lg D(I, \varphi_t)] \\
& + \mathbb{E}_{s_0 \sim P_{G_0}, t \sim P_{data}} [\lg (1 - D(G(s_0, \hat{c}), \varphi_t))], \\
\mathcal{L}_G = & \mathbb{E}_{s_0 \sim P_{G_0}, t \sim P_{data}} [\lg (1 - D(G(s_0, \hat{c}), \varphi_t))] \\
& + \lambda D_{KL}(\mathcal{N}(\mu(\varphi_t), \Sigma(\varphi_t)) \parallel \mathcal{N}(0, I))
\end{aligned} \tag{2.10}$$

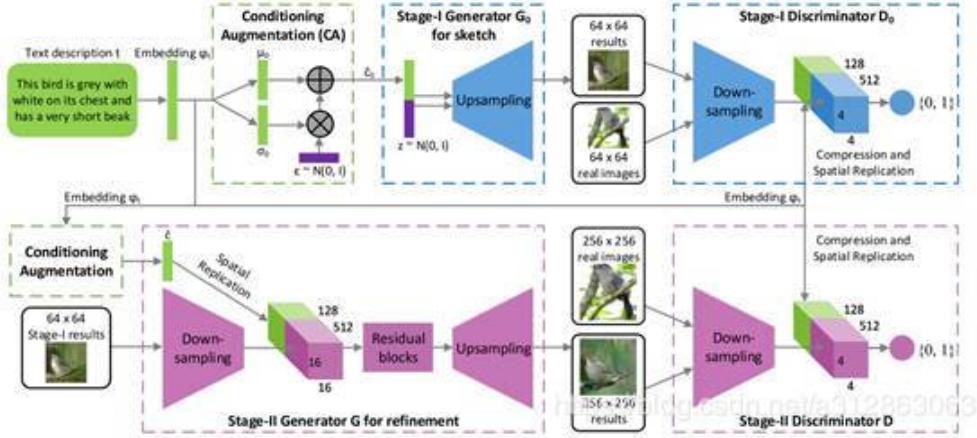


图 2.8 StackGAN 构架示意

StackGAN 模型相比较于 CGAN 模型，可以生成更高清的图片，而且结构更为真实。

2.2.4.2 对 StackGAN 的进一步优化与运用

同一课题组紧接着提出了 StackGAN++^[28]，对原有算法进一步进行了优化。StackGAN++ 模型也叫做 StackGAN_v2，对之前的 StackGAN 模型主要有三点改进。第一，改为采取树状结构作为输入，通过不同的生成器来生成不同尺寸的图片，多层次叠加形成了有位置关系的“假图片”，即混合图层叠加图片；第二，额外引入了非条件性的损失，直接将服从正态分布的损失 z 叠加在最后生成的“假图片”上；第三，引入了色彩限制，限制了最终“假图片”的色彩信息分布。

李飞飞课题组的 Johnson^[29] 在 StackGAN 模型之上，提出了自己的优化。

该方法主要有三个特点：第一是有一个处理场景图的方法，第二是确保了生成的图像中个物体正确，且位置关系合理。第三是确保了生成图像的质量比较好。其具体技术路线将在下一章进行推导、介绍。

2.3 本章小结

本章中我主要总结了本次技术方案的前置技术，包括神经网络中的 LSTM-RNN 长短期记忆循环神经网络模型和 StackGAN 模型及其相关技术。LSTM-RNN 模型是一种相对稳定、收敛，可以一定程度上避免梯度消失的深度学习模型，主要为下一章实现图片标注方法的技术做好铺垫。StackGAN 及其升级版本是 Han Zhang 及其他科研人员 2016 年其后提出的一系列理论，我简要介绍了它的背景、原理和与后续技术的比较，决定使用李飞飞课题组的场景图像生成方法作为实现的理论依据，也为下一章自然语言生成图片的方法做好了前置说明。

3 方案设计

3.1 概要

设计要求是：用户输入一张图片后，可以得到相应的标注语句，了解图片语义；在用户输入一句话的时候，可以得到描述这一句话的图片，直观“感受”文字。

软件设计主界面使用 `python` 的 `tkinter` 包来制作。`python` 是一个跨平台语言，用 `python` 制作方便产品在不同平台使用。界面中应当可以选择调动两个功能，分别是图片翻译为自然语言和语句翻译为图片。两个功能分别提前训练出成品模型，通过主界面按钮内嵌入的方法，调用对应模型进行计算。

3.2 软件模型

软件分为三个部分，第一部分是软件界面，需要实现简洁明了的操作功能，方便使用。

第二部分是图片标注功能。实现这一功能的模型通过 `image caption` 的模型进行训练，算法参照蒙特利尔大学 Kevin Xu^[30] 提出的模型进行实现，并经过基于一个或多个数据集进行多个 `epoch` 的训练，比较选出表现较好的模型，嵌入到应用中使用。

第三部分是文本生成图片功能。实现这一功能的模型通过一个特殊的 GAN 模型进行训练，并且要加入后期图片处理算法，以骗过判别器并使其更加真实。这一模型参照卡耐基-梅隆大学 Justin Johnson^[29] 提出的基于场景双 GAN 模型配合进行实现，并经过合适的数据集训练，得到表现较好的模型，嵌入到应用当中使用。

3.3 变量符号与定义

文中使用集合如下：

1. O 代表物品 (object) 的集合；
2. C 代表物品目录 (catagory) 的集合；
3. R 代表关系 (relationship) 的集合；
4. E 代表物品-关系-物品组成的边 (edge) 的集合。

文中脚标定义如下：

1. t 代表 LSTM 模型中的时序， t 为当前时序，而 $t-1$ 或 $t+1$ 为前一或后一时序；
2. f 代表任一函数（使用 f 指代）代入当前操作或模型；
3. v_i 在第二个算法中，代表 GCN 当前层级的节点编号，因为不讨论跨多层关系，所以脚标中不增加时序信息。

文中小写字母对象定义如下：

1. s 指代集合 S 中的元素，其中 S 指代任一集合，如物品集合 O 等，他们通常和集合一起出现；
2. c 代表 LSTM 模型中的细胞 (cell)，记录细胞当前状态；
3. i 代表 LSTM 模型中的输入门 (input)，执行 LSTM 模型中的输入操作；
4. o 代表 LSTM 模型中的输出门 (output)，执行 LSTM 模型中的输出操作；
5. f 代表 LSTM 模型中的遗忘门 (forget)，选择 LSTM 细胞的部分状态丢失；
6. z 代表生成模型中为了骗过判别器、降低生成图片锐度增加的噪声。

文中主要函数定义如下：

1. ϕ 代表注意力机制；
2. \mathcal{L} 函数代表 GAN 模型中用作训练目标函数的损失函数；
3. $f(S, z)$ 代表自然语言生成图像算法中的目标函数。

3.4 软件界面设计

3.4.1 软件例图

对于预期的软件效果，我制作了原型图（图 3.1）进行示意。图中，应当出现英文的地方用英文示意，应当出现路径的地方用路径示意，而应当出现中文的地方和说明文字使用中文表述。

其中，界面上方由一组两个单选选项的选择栏和两个数据输入栏组成。选项栏的作用是选择当前需要实现的功能，由“图片翻译语义”和“文本翻译图片”两个选项组成；数据输入栏分别是一个文本输入栏和一个文件选择栏，分别对应着文本和图片的输入选择。

其下是一个明显的按钮，突出的设计可以让用户能轻易明白操作方法，也让用户更有仪式感，能感受到这一操作的划时代意义。

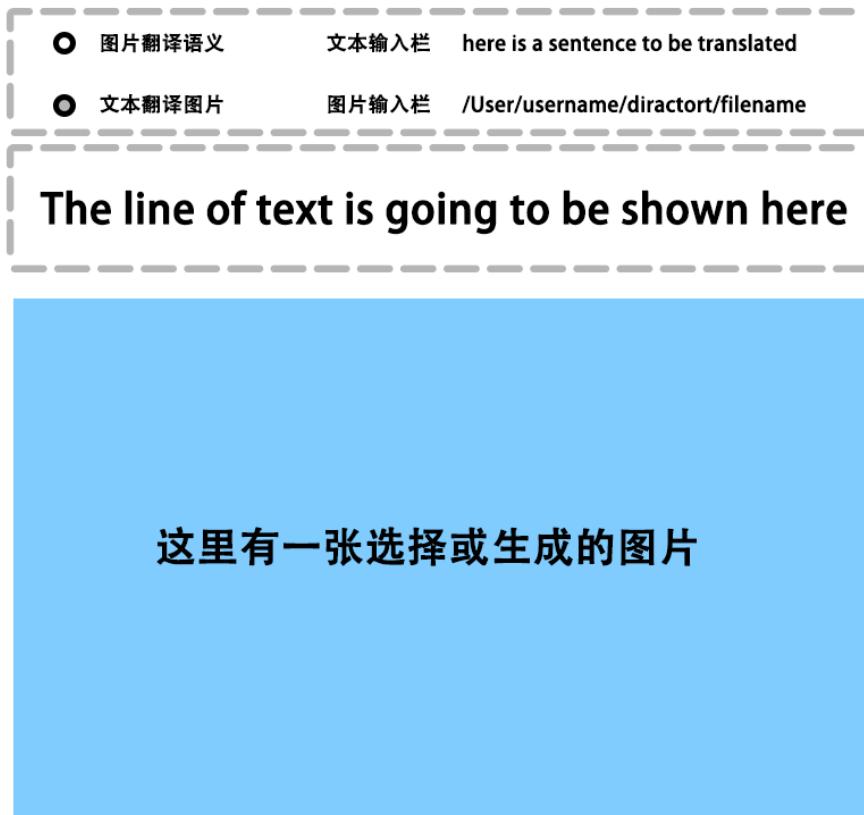


图 3.1 软件界面原型图

最下方的大区域是输出区域，根据功能的不同可以输出不同的内容。在选择图像翻译语义功能时，会输出一句话，这一行文字和图片将一起显示，方便对比观察；在选择语义生成图像选项时，将输出生成的图片和原语句，也是为了方便用户观察与分享，表意清晰。

3.4.2 软件开发模型

对开发设计软件的过程，我选用了编码-修改模型^[31]。如图 3.2 所示，在这一模型中不进行计划与建模，先进行编码，调整至用户满意时，发布运行，在出现问题与更新时维护，直到软件生命周期结束为止。

本次软件设计开发的过程是要独立接触一个不熟悉的领域，对于开发者来说不易估计编码需要的时间。深度学习对初学者来说有一个难点：环境安装中很多细节难以把控；并且深度学习的学习过程需要使用多块显卡进行天级时长的训练过程，才可以得到结果。基于上述理由，我可以推测算法的实现需要比较长的时间，并且可能一次训练出的算法达不到理想的状态，需要调整模型、重复训练。

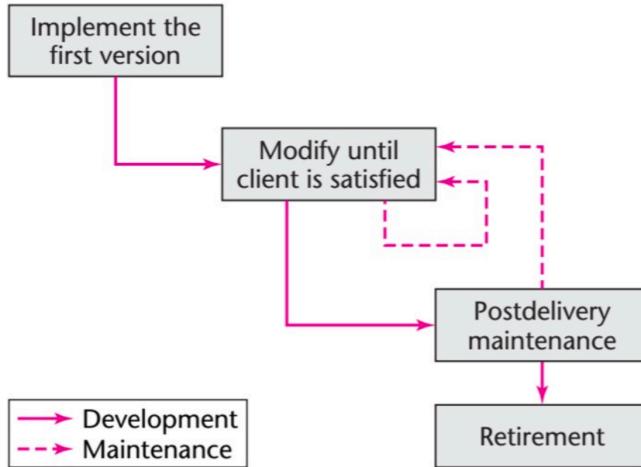


图 3.2 编码-修改模型的基本结构

在设计的过程中，对我来说的难点的确是环境安装部分。需要的新虚拟容器因为电脑上旧有的项目擅自修改了环境变量，需要重新安装、整理环境，并且安装合适版本的环境，搭配当前系统，对开源项目进行修改、复现，并在此基础上进行改进。

另外，本软件的模型相对比较简单，是用户界面-调用模型的两层结构，难点在于模型的建立和调整。所以本设计非常适合使用编码-修改模型进行开发。

3.5 软件功能设计

功能部分分为图片标注功能以及文字生成图像功能，两个功能我使用了两个结构相对复杂的深度学习模型组合构架的算法，将在下面具体介绍。

3.5.1 图片标注方法

3.5.1.1 算法整体模型

在这一模型中，要使用 LSTM 模型进行训练，训练出的模型放在主函数的调用函数中，实现图片翻译为自然语言的功能。

这一模型主要流程由四步组成。

1. 在模型中输入图片，作为输入信息；
2. 由卷积神经网络提取图片信息，形成图片特征信息（即后文编码步骤）；
3. 由注意力机制（attention）对所提取的图片特征信息进行处理，加强或抑制部分区域，作为后续输入 LSTM 的输入信息——在不同时刻，注意力信号会受到

上一次 LSTM 的输出信息的影响，即注意力信号作为 LSTM 神经元细胞的状态，受到输出词语的影响而改变（这也是后文的解码部分）；

4. LSTM 最终输出文本，形成最后的结果。

3.5.1.2 编码部分

第一步，要对训练集中的标注编码，形成特征向量。词典中已经预先确定了 K 个词语，对于每一行标注 y_i ，可以将其通过词典序号，将句子映射成输入向量，每一个元素的位置意义是序号，即图片相关的类别，数字则是关联度。编码之后生成向量 \mathbf{y}_i ，一起构成输入矩阵。

$$y = \mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_C, \mathbf{y}_i \in \mathbb{R}^K$$

第二步，对图片编码。使用一个卷积神经网络 (Convolutional Neural Network, CNN) 对图片的特征进行提取，从而形成图片编码。编码好的图片，后续会作为注释向量 \mathbf{a} 使用。

$$a = \{\mathbf{a}_1, \mathbf{a}_2, \dots, \mathbf{a}_L\}, \mathbf{a}_i \in \mathcal{R}^D$$

3.5.1.3 解码部分

解码部分使用的技术是 LSTM，即长短期记忆模型。解码后生成的是标注文本，在预测最后一个词的时候，需要背景向量、前一时刻的隐藏层向量、前一时刻的词。这一部分实用的 LSTM 模型结构，由图示意出。

背景向量 \hat{z}_t 由注意力机制函数和图片注释向量计算得出，并且与时序 t 有关，随时序推进而变化。相当于是有选择地传输图片注释向量中的信息，是图片信息的动态表达。

确定一个注意力机制函数 ϕ 来计算 t 时刻的背景向量 \hat{z}_t 。对于输入的图片注释信息，为了推测这一位置是否是正确的注意力集中点，在式 (3.2) 中定义一个可以归一化的参数 $\alpha_{t,i}$ 来表示在 t 时刻，位置 i 是正确关注点的置信度。

$$e_{t,i} = f_{att}(\mathbf{a}_i, \mathbf{h}_t) \quad (3.1)$$

$$\alpha_{t,i} = \frac{\exp e_{t,i}}{\sum_{k=1}^L \exp e_{t,k}} \quad (3.2)$$

计算置信度时需要用到 f_{att} 函数，这一函数定义为一个“硬”机会注意力机制函数。这一机会注意力机制的“软”版本由 Bahdanau^[32] 提出，仿照这个机制可以提出硬版本函数。

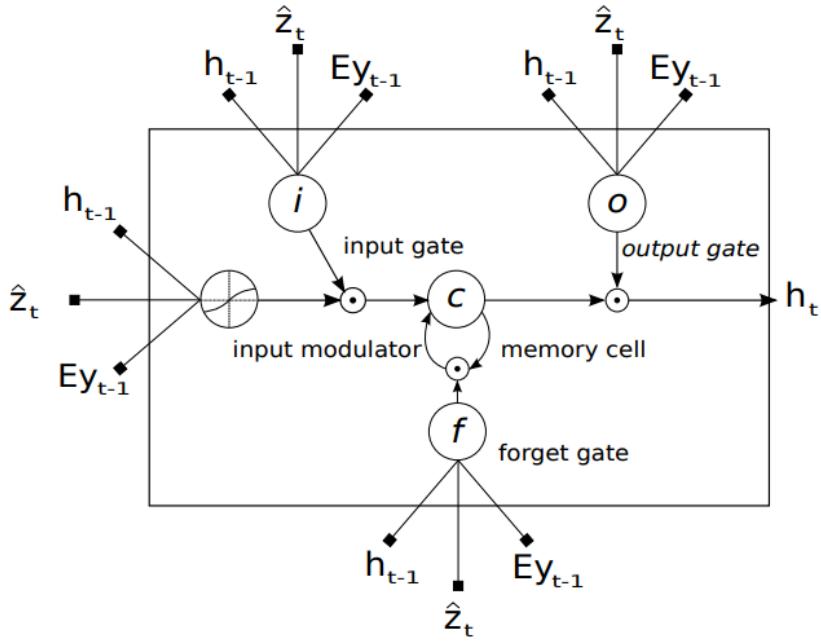


图 3.3 解码 LSTM 神经元细胞模型结构图

现在定义 s_t 为模型为了生成第 t 个单词所选择的注意力区域。其中， s_t 的第 i 项是开关函数，此项置 1 时，当前模型生成打你是第 i 个单词。现设定多次伯努利分布的参数 α_i ，将背景向量视为随机事件，则有：

$$p(s_{t,i} = 1 | s_{j < t}, \mathbf{a}) = \alpha_{t,i} \quad (3.3)$$

计算出权重之后，即可使用注意力机制函数计算出背景向量 \hat{z}_t ：

$$\hat{z}_t = \phi(\mathbf{a}_i, \alpha_i) \quad (3.4)$$

$$\hat{z}_t = \sum_i s_{t,i} \mathbf{a}_i \quad (3.5)$$

一个“软”估计得算法即如式 (3.6) 的方法计算出。但是，还可以提出一种“硬”估计的

$$\begin{aligned} \phi(\{\mathbf{a}_i\}, \{\alpha_i\}) &= \mathbb{E}_{p(s_t|a)}[\hat{z}_t] \\ &= \sum_{i=1}^L \alpha_{t,i} \cdot \mathbf{a}_i \end{aligned} \quad (3.6)$$

现在定义一个概率的对数函数，计为 L_s ，作为模型的优化目标；对于训练的目标参数 W ， L_s 函数就是优化目标。则可以推导得其下界为式 (3.7) 所示，从而

得到最终训练梯度为式 (3.8)

$$\begin{aligned} L_s &= \sum_s p(s, \mathbf{a}) \log p(\mathbf{y} | s, \mathbf{a}) \\ &\leq \log \sum_s p(s, \mathbf{a}) p(\mathbf{y} | s, \mathbf{a}) \\ &= \log p(\mathbf{y}, \mathbf{a}) \end{aligned} \quad (3.7)$$

$$\frac{\partial L_s}{\partial W} = \sum_s p(s | a) \left[\frac{\partial \log p(\mathbf{y} | s, \mathbf{a})}{\partial W} + \log p(\mathbf{y} | s, \mathbf{a}) \frac{\partial \log p(\mathbf{y}, \mathbf{a})}{\partial W} \right] \quad (3.8)$$

其中有

$$\tilde{s}_t \sim \text{Multinoulli}_L(\alpha_i) \quad (3.9)$$

则

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{i=1}^n \left[\frac{\partial \log p(\mathbf{y} | \tilde{s}^n, \mathbf{a})}{\partial W} + \log p(\tilde{s}^n | \mathbf{a}) \frac{\partial \log p(\mathbf{y}, \mathbf{a})}{\partial W} \right] \quad (3.10)$$

这里的函数 ϕ 是利用了“软”确定注意力机制 (Deterministic “Soft”Attention)，这一机制算法可以稍为简易地作出注意力正确位置的判断，可以作出计算，用于判断下一个词的注意力位置；并且，这个函数的目的是计算注意力，而非前文从提取注意力。可以将 \hat{z}_t 的期望值 $E[\hat{z}_t]$ 作为其取值，带入计算，即设定 ϕ 函数为式 (3.6) 中的计算方法。

从式 (3.8) 中，可以看出优化函数 L_s 由基于蒙特卡罗方法多次采样逼近梯度的方式，对式 (3.9) 遵从多次伯努力分布的注意力区域位置进行采样，测算优化函数。在这个时序数据中，为了消除，使用滑动平均基线的方法^[33]，令基线的对数更新比例如式 (3.11) 所示。

$$b_k = 0.9 \times b_{k-1} + 0.1 \times \log p(\mathbf{y} | \tilde{s}_k, a) \quad (3.11)$$

最终得到的“硬”注意力函数如式 (3.12) 所示。

$$\frac{\partial L_s}{\partial W} \approx \frac{1}{N} \sum_{i=1}^N \left[\frac{\partial \log p(\mathbf{y} | \tilde{s}^n, \mathbf{a})}{\partial W} + \lambda_r [\log p(\mathbf{y} | \tilde{s}^n, \mathbf{a}) - b] \frac{\partial \log p(\tilde{s}^n | \mathbf{a})}{\partial W} + \lambda_e \frac{\partial H[\tilde{s}^n]}{\partial W} \right] \quad (3.12)$$

将其代回式 (3.4) 中的 ϕ 函数，即可得出 \mathbf{a}_i 由“硬”选择注意力集中位置的参数 α_i 加权后的取样结果。

解决了背景向量的问题，可以开始训练 LSTM 模型了。对于这个模型，需要进行初始化，确定其中图 3.3 标注的传入隐状态 h_0 及其内部初始环境 c_0 。取标注向量的平均，作为初始的状态，进行操作。即：

$$\mathbf{c}_0 = f_{init,c} \left(\frac{1}{L} \sum_{i=1}^L L \mathbf{a}_i \right)$$

$$\mathbf{h}_0 = f_{init,c} \left(\frac{1}{L} \sum_{i=1}^L L \mathbf{a}_i \right)$$

3.5.2 自然语言生成图片方法

这一模型主要是用通过场景生成图像的 GAN 模型来训练，得到的模型放在主函数的调用函数中，实现自然语言生成图片的功能。

3.5.2.1 图片生成模型

生成图片的总体的算法流程如算法 1 所示，分为三个大步骤：GCN 确定场景布局，确定图像边框布局，级联优化网络 (Cascade Refinement Network)^[34]。

输入：自然语言短句 S ，噪声 z

输出：生成图片 $\hat{I} = f(G, z)$

$O, E \leftarrow$ 提取 (物品) 元素和 (位置) 关系 (生成语义树 (S));

$G \leftarrow$ 构成物品关系图 (O, E);

Layout \leftarrow GCN(G);

$I \leftarrow Generate(Layout)$;

$\hat{I} \leftarrow CRN(I)$;

算法 1：图片生成方法

第一步的图卷积网络 (GCN) 需要输入场景关系图，它是由自然语言经过简单处理后得到的。场景关系图满足下列条件：

$$O = o_1, \dots, o_n, o_i \in C$$

$$E \subseteq O \times R \times O = \{(o_i, r_{ij}, o_j) \mid i, j \in \mathbb{N}^{\star}\}, r \in R$$

图卷积网络可以沿着场景图的关系 (边) e_i ，计算出图中各个物体 o_i 的嵌入向量 \mathbf{v}_i 。其单层局部结构如图 3.4 所示，这张图示意了图卷积网络的每一层之间，向量如何更新迭代。

图 3.4 中 v_i 代表物体向量，而 v_{ri} 代表关系向量。其中涉及了三个物品 o_1, o_2, o_3 和两组关系 $(o_1, r_1, o_2), (o_2, r_2, o_3)$ 。这些向量需要经过三个函数，分别是 g_o, g_p, g_s ，其中 g_o, g_s 的值需要作为候选变量，参与到调和函数 h 中，最终决定更新后的物体向量。

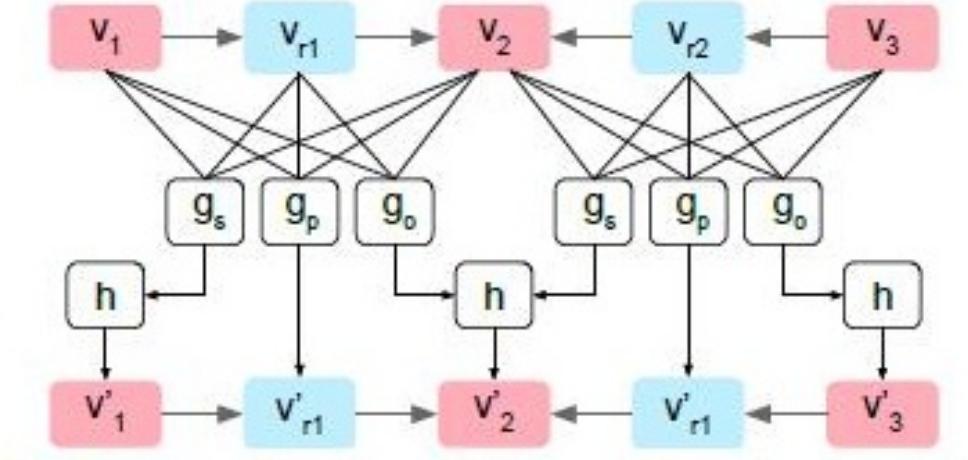


图 3.4 图卷积神经网络 (GCN) 单层结构图

关系向量的更新比较简单，它遵从式 (3.13)。

$$v'_r = g_p(v_i, v_r, v_j) \quad (3.13)$$

而图中的物体元素则更为复杂一些。要考虑每一个元素作为它参与的所有关系中的角色，而计算出它的更新向量。对于每一个物体向量 o_i ，它对应下列两个候选向量，供调和函数调用。其中 V_i^o 代表了所有 o_i 作为关系中被指向的点所在关系所生成的候选向量集合，反之 $V_i^s o_i$ 代表了所有 o_i 作为关系中指向起点所在关系所生成的候选向量集合。

$$V_i^s = \{g_s(o_i, r, o_j) \mid (o_i, r, o_j) \in E\}$$

$$V_i^o = \{g_o(o_j, r, o_i) \mid (o_j, r, o_i) \in E\}$$

随后，我们可以通过式 (3.14) 中的调和函数更新物体向量 v_i 。

$$v'_i = h(V_i^s, V_i^o) \quad (3.14)$$

生成的图像还需要两个额外的生成模型进行优化，以得到平滑效果更好的生成图像。图像的平滑程度是评价生成图像质量的重要标准之一。

第一部分中，我们得到了最终得到了每个物体的高级特征表达向量（关系的特征表达向量主要作为隐藏值影响物体的特征向量计算）。在第二部分中，则需要测算每一个物体的框架位置。

对于上一环节得到的每一个物体的特征表达向量 \hat{v}_i ，我们分别将它传入蒙板回归网络 (Mask Regression Network) 和框回归网络 (Box Regression Network)，分别

求出其在画布上的形状和位置。最终得到形状蒙板 \hat{m} 和 \hat{b} :

$$\hat{m} \sim M \times M$$

$$\hat{b} = (x_0, y_0, x_1, y_1)$$

将 \hat{m} 和物体特征向量 v_i 按照元素序逐一将值相乘，得到物体的形状与样式，再将其缩放，使其可以代入 \hat{b} 所规定的框架中，则可以得到这一物体的形状、位置与大小。

第三部分则是将在画布上的物体拼合，组成一幅新的“图像”。这个图像要求所有的元素都在其中对此，

3.5.2.2 鉴别模型

我们生成了两个鉴别网络模型，用作与前述生成模型进行对抗。

第一个判别器 D_{img} 是对整个 f 函数结果，即生成图片 \hat{I} 进行判别。这个判别器的损失函数是：

$$\mathcal{L}_{D_{img}} = \mathbb{E}_{x \sim p_{real}} [\log D(x)] + \mathbb{E}_{x \sim p_{fake}} [1 - D(x)] \quad (3.15)$$

其中， $x \sim p_{real}$ 指训练集中提供的真实图片 I ，而 $x \sim p_{fake}$ 则是指生成图片 $\hat{I} = f(g)$ 。生成模型函数 f 的训练目标是最小化判别器得出的 $\mathcal{L}_{D_{img}}$ 值。

$f, \mathcal{L}_{D_{img}}$ 函数组成的 GAN 网络模型，保证了图片整体的真实性和平滑性，与 [35] 中的判别器性质类似。

当然，在生成模型第二部分生成的物体图片要单独进行训练，这就需要一个使用另一个判别器 D_{obj} 来进行对每个物体的图像判别。但是物体的图像大小可能会与原图不相同，但这是不影响图片生成效果的，所以在判别之前需要用双线性插值^[36] 的方法将图片变换处理为相同的大小，以减小大小对判别带来的过拟合。

除此之外，判别器还要做一项特殊的工作。与以往的 GAN 网络不同，它是一道保险装置，希望再不输入物体类别，仅输入生成图像时，将物体判别为正确的分类。在这一方面，判别器和生成器都希望正确类别的置信度尽量高，以保证其在 [37] 中提到的判别标准（Auxiliary Classifier）中的可识别性。

经过以上恰当的双 GAN 模型训练与保险机制，可以得到可用的图片生成器 f 。

3.6 本章小结

本章具体地介绍了这一设计中所包含的设计细节及其理论依据，并确定了软件构架的结构，为完成代码构建和实验设计打下了基础。

本章中，确定了以 `python` 语言的 `tkinter` 库来写界面，保证界面简单简介易懂；用 CNN 和 LSTM 技术，基于注意力的方法，通过编码-解码的流程来完成图片标注功能；用基于位置场景的方法，由 GNN 和两个判别器构成的 GAN 模型来分步生成图片，最后生成合成图片的方法来实现由文本生成图片的功能。

在下一章里，我会继续记述实验的过程及其表现。

4 实验与分析

4.1 实验环境

4.1.1 模型训练环境

服务器环境，操作系统为 Ubuntu 16.04.6 LTS (GNU/Linux 4.4.0-142-generic x86_64)，使用命令行 ssh 工具登入。

内存 64GB，搭配显卡 2080 Ti 四块，由于资源分配原因本设计中实验使用两块。

使用 anaconda 创建虚拟环境，为算法各自安装需要的依赖包与训练环境。

4.1.2 软件样本运行环境

Macbook Pro 电脑 15.6 寸屏幕版，mid-15 release，A1398 型号；OS X 10.12(Serrina High) 系统；集成显卡，内存 16G，没有特别要求。

4.2 图片标注实验

4.2.1 模型代码

模型主题使用了开源代码，由第三方代码作者完成，在 Github 网站上开源；同时我在其中做了修缮维护，保证了复现在当前版本的依赖包上和现在可以运行的系统里仍然可以运行。

原代码开发环境是 python 2.7 和 tensorflow 0.14，由于目前科学计算使用的 python 以 python3 为主流，我将其修改维护至了 python 3.6 和 tensorflow 1.14 版本下可以运行的代码。

代码结构分为几个部分：主函数、基本模型类、训练算法。

4.2.2 运行过程

通过在论坛上调研、向专家咨询，我认为较少的数据会导致图片标注模型的效果减弱，会限定构图、限定主题、限定语汇，且识别精确度会很差。为此，我选用了 MSCOCO 数据集作为训练和测试集。对于这次实验，我先选择了最早的 COCO2014 数据集，作为试验，后来发现训练时间成本较长，决定使用 COCO2014 数据集下训练生成的模型作为应用模型。

这一算法的虚拟实验环境 python 版本为 3.7, tensorflow 版本为 1.14.0, 其中安装的依赖包有: numpy 1.17.2, OpenCV 4.1.1.26, NLTK 3.4.5, Pandas 0.25.1, Matplotlib 3.1.1, tqdm 4.36.1。其中 NLTK 用作训练集中文字信息的分词处理, 仅仅执行了 `nltk.download("punkt")`, 作为数据支撑, 没有下载完整数据。

模型的训练在实验环境下使用 4 块 NVIDIA 2080 Ti 显卡, 训练了 233 小时, 执行了 100 个 epoch, 每一个 epoch 都遍历了训练集中的所有数据。

4.2.3 遇到的主要问题与解决方法

4.2.3.1 数据传输问题

服务器的物理地址在浙江大学, 属于教育网的浙江位置, 从我的工作地点到浙江大学校内当前跳转节点较多, 使用 RVPN 进行连接后, 数据传输速度上限是 2.00Mbits/sec, 所以传输一个总大小为 19GiB 以上的数据集, 需要的时间为 40000 秒, 相当于十余小时。另外, 由于 RVPN 的连接不稳定, 而 `scp` 命令传输文件不支持断点续传, 所以经常出现文件损坏的现象。

经过多次尝试和实践, 我采用分包的方式, 将数据集化为 1GiB 到 2GiB 大小的压缩包, 由 `scp` 命令逐一传输; 同时使用完善、成熟的数据集, 尽量最大化一次成功率。

4.2.3.2 依赖包函数不兼容问题

`tensorflow` 在目前的稳定版本中, 已经增加了很多新的借口, 而 1 版本和 0 版本的接口, 有一些已经取消了。即使我安装的是 1 版本的 `tensorflow` 环境, 仍然有一些接口无法调用。经过调研, 我使用了评价较高的 `tensorflow.compat.v1` 库, 来实现变易出错的接口, 解决了大部分问题。

4.3 自然语言生成图片实验

4.3.1 模型代码

模型使用了开源代码, 由论文原作者完成, 在 Github 网站上开源; 同时我在其中微调, 以适合我所需的数据集。

这一套代码的特点是, 它含有一套下载数据集的 bash 代码, 方便下载。这一套代码推荐在 googleapis.com 网站上直接使用 `wget` 命令下载数据, 这样下载的数据传输速度比较稳定, 可以正常下载使用数据。

代码主要包含一个模型运行的主函数、一个模型类和一个训练模型的方法。除此之外，以面向对象的思想对训练过程中的辨别器 D 等类、双线性插值、图片级联优化网络等方法都单独编码，然后加上工具方法的编码，构成了整个工程。

4.3.2 运行过程

运行中由于代码设计，只使用一块显卡进行训练。

因为显卡的设置问题，不可以同时使用同一块显卡即运行 tensorflow 模型，又训练 pytorch 模型，因为如果它们同时运行，不同的容器会导致驱动将所有显存都分配给 tensorflow 的任务，导致 pytorch 训练的模型无法正常运行。所以，需要等到上一模型训练完毕，再开始训练这一模型。或者，可以让它们在不同的显卡上运行。这在一定程度上影响了我的时间安排，导致我仅在 VG 数据集上运行了代码。

在 VG 数据集上，我使用了代码中编写的“强注意力生成”方法制作的生成模型，得到了一个可以从自然语言自动生成图片的模型。

4.3.3 遇到的主要问题与解决方法

这一模型更加的复杂，并且开源代码仅有 pytorch 版本的实现。按照要求，为了运行这段代码，需要安装 python 3.5 下的 pytorch 0.4.0 版本。但是，按照版本要求，在目前的 cuda 版本为 10.0 的电脑中不可以安装这一版本的 pytorch 了，所以可以替代性地使用 torch 1.0.0 来运行程序。

在安装虚拟环境的时候，需要注意无论是 anaconda 官网默认源还是清华大学镜像源都难以链接，并正常下载完毕 torch 安装包，因为其大小高达 700MB，在下载到一半的时候就常常会出现 HTTP: 200 错误代码，导致下载中断。可以打出下列 shell 命令，将在官网下载好的 whl 文件安装在环境里。这里的包一定要符合环境下的 python 版本要求，不知道要求的可以在 python shell 中运行“`wheel.pep425tags.get_supported()`”代码，查询 wheel 安装要求。

```
conda create venv env_name
conda activate env_name
pip install torch-x.x.x-cp3x-cp3xm-ostype.whl
pip install torchvision
```

4.4 实验分析与总结

4.4.1 图片标注表现

对图片标注的模型，我选取了三个模型作为对比，使用 BLEU 评分^[38] (Bilingual Evaluation Understudy) 的方式进行评价。

BLEU 评分是 IBM 出品的机器翻译评分标准，按照式 (4.1) 的算法得出评分。这个评分是对一条翻译打出，后续数据中取平均得分。

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^N w_n \log P_n\right)$$

其中, $BP = 1 \quad c > r$
 $e^{1-r/c} \quad c \leq r$

(4.1)

式中 BP 即简短惩罚，即在句子短的时候降低评分。在本次评分中，不加入 BP 机制。

其中，我希望测试二十个 epoch 量级的训练是否可以支撑一个可用的模型，事实上 19 个 epoch 的模型经过个例测试，不能很好地生成注意力转移的机制，导致句子不够通顺，经常出现连续无意义重复关键词的现象。后来，我使用这一断点“214999.npy”（后称“断点模型”）继续训练，得到了 100 个 epoch 后生成的“289999.npy”（后称“强注意力模型”）模型。

另外，我引入了预训练的“弱注意力模型”的测试数据^[30]，作为对照，观察注意力机制对标注结果的影响，以选取最合适的选择，嵌入软件。对于上述三种模型，我使用 nltk 包中所带算法，使用 1-4 精度修正的四种评分进行评价，得到的结果如表所示。其中若关注模型和第二组强关注模型是文献 [30] 中的结果。

表 4.1 三种模型的 BLEU 评分

模型选择	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
断点模型					—
弱关注模型	70.7%	49.2%	34.4%	24.3%	23.90%
强关注模型（我的训练）	70.3%	53.6%	39.8%	29.5%	—
强关注模型（文中数据）	71.8%	50.4%	35.7%	25.0%	23.04%

可以看到，弱关注模型表现稍好于强关注模型，但是在原文中结论有所不同。因为我自己训练的数据可能因为环境不同表现有少许的差异，但是在同一环境下，强关注模型的训练结果更有利翻译的准确性。

4.4.2 文本生成图像算法表现

理解文本生成的图像涉及到图像标注这一不成熟且复杂度高的算法（正如前文工作所说），所以这一算法的表现不能使用 BLEU 评分来评价。对于这一图片是否被人理解，采用两种方法评价。

第一种是采用检查图片平滑性的一种评分^[39](Inspection Scores)，作为量化的参考，检查图片质量。这一部分我摘取了 StackGAN^[23] 的相关数据，对比这一算法所生成的图片质量是否过关。从表 4.2 中可以看出，这一模型所生成的图像在数据上不比 StackGAN 更加逼真。

表 4.2 图片平滑性评分与 StackGAN 模型对比

模型类别	数据集类别	
	COCO	VG
文中模型（无 D_{img} 作用） ^[29]	5.6 ± 0.1	5.7 ± 0.3
文中模型	6.7 ± 0.1 ^[29]	5.5 ± 0.1
StackGAN ^[23]	8.4 ± 0.2	-

第二种评价方法是主观评价，具体观看例句生成的图片，感受图片质量的差别。在 Github 开源代码中可以找到 StackGAN 模型的预训练模型^①，可以根据这一模型用测试语句生成图像，对比直观效果。

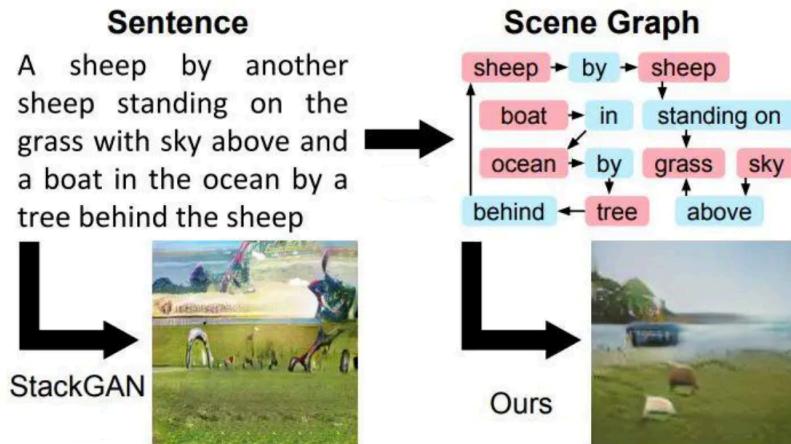


图 4.1 比较 StackGAN 和本文算法的生成图像差异

有一首小诗^②，因为其包含许多位置关系，非常适合这一算法。以此为例，可

①在 github 网址 <https://github.com/hanzhanggit/StackGAN-Pytorch> 可以找到预先训练完成的模型，并从原文找到了评测数据

②由文献 [29] 原作者提出，以解释模型在生成图像方面的优势

以表现此算法对人类主观理解上的优势。

对于如下诗句，StackGAN 模型和本文使用模型分别生成了如图 4.1 的两张截然不同的图像。显然，本文使用模型保留了大部分的位置关系，并且没有产生图像上的扭曲。从人类的视角来看，

A sheep by another sheep
standing on the grass
with sky above and a boat in the ocean
beyond the tree next to the sheep.

另外，在图 4.2 我们可以清晰地从场景关系图的树状发展流程中，看到构成生成图片的过程。

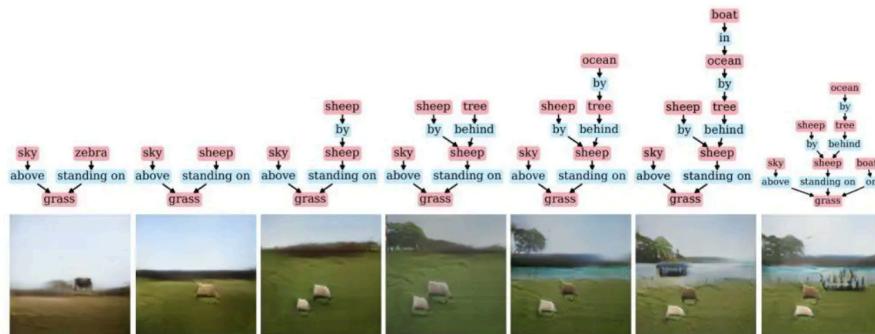


图 4.2 比较 StackGAN 和本文算法的生成图像差异

4.5 本章小结

这一章节分别介绍了软件界面编码方式、软件两个功能模型训练方式过程与软件部署调试的过程。在软件界面编码中，试用了 `tkinter` 进行编码，用比较简明的方式，实现了用例图中的效果；在软件功能模型训练中，记述了 LSTM 模型和 GAN 模型分别在环境安装、数据集选取、训练次数调整和呈现方式的过程

5 设计意义阐述与展望

5.1 我的设计意义

- 5.1.1 制作了一个简洁易用系统
- 5.1.2 活用知识、实践了深度学习技术

5.2 未来发展方向

参考文献

- [1] Cai W, Qiu G. Visual question answering algorithm based on image caption[A]. 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)[C], 2019 : 2076 – 2079.
- [2] FANG H, GUPTA S, IANDOLA F, et al. From captions to visual concepts and back[A]. Proceedings of the IEEE conference on computer vision and pattern recognition[C], 2015 : 1473 – 1482.
- [3] LEBRET R, PINHEIRO P O, COLLOBERT R. Phrase-based image captioning[J]. arXiv preprint arXiv:1502.03671, 2015.
- [4] ZHU X, LI L, LIU J, et al. Image captioning with triple-attention and stack parallel LSTM[J]. Neurocomputing, 2018, 319 : 55 – 65.
- [5] YOU Q, JIN H, WANG Z, et al. Image captioning with semantic attention[A]. Proceedings of the IEEE conference on computer vision and pattern recognition[C], 2016 : 4651 – 4659.
- [6] FANG F, LI Q, WANG H, et al. Refining attention: A sequential attention model for image captioning[A]. 2018 IEEE international conference on multimedia and expo (ICME)[C], 2018 : 1 – 6.
- [7] ZHU X, LI L, LIU J, et al. Captioning transformer with stacked attention modules[J]. Applied Sciences, 2018, 8(5) : 739.
- [8] VINYALS O, TOSHEV A, BENGIO S, et al. Show and tell: A neural image caption generator[A]. Proceedings of the IEEE conference on computer vision and pattern recognition[C], 2015 : 3156 – 3164.
- [9] KINGMA D P, WELLING M. Auto-encoding variational bayes[J]. arXiv preprint arXiv:1312.6114, 2013.
- [10] REZENDE D J, MOHAMED S, WIERSTRA D. Stochastic backpropagation and ap-

proximate inference in deep generative models[J]. arXiv preprint arXiv:1401.4082, 2014.

- [11] HINTON G E, SEJNOWSKI T J, ACKLEY D H. Boltzmann machines: Constraint satisfaction networks that learn[M]. [S.l.] : Carnegie-Mellon University, Department of Computer Science Pittsburgh, 1984.
- [12] ACKLEY D H, HINTON G E, SEJNOWSKI T J. A learning algorithm for Boltzmann machines[J]. Cognitive science, 1985, 9(1) : 147 – 169.
- [13] HINTON G E, SALAKHUTDINOV R R. Reducing the dimensionality of data with neural networks[J]. science, 2006, 313(5786) : 504 – 507.
- [14] GOODFELLOW I, POUGET-ABADIE J, MIRZA M, et al. Generative adversarial nets[A]. Advances in neural information processing systems[C], 2014 : 2672 – 2680.
- [15] PRADHAN S, HACIOGLU K, KRUGLER V, et al. Support vector learning for semantic argument classification[J]. Machine Learning, 2005, 60(1-3) : 11 – 39.
- [16] SURDEANU M, HARABAGIU S, WILLIAMS J, et al. Using predicate-argument structures for information extraction[A]. Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics[C], 2003 : 8 – 15.
- [17] GILDEA D, PALMER M. The necessity of parsing for predicate argument recognition[A]. Proceedings of the 40th annual meeting on association for computational linguistics[C], 2002 : 239 – 246.
- [18] YAO Y, HUANG Z. Bi-directional LSTM recurrent neural network for Chinese word segmentation[A]. International Conference on Neural Information Processing[C], 2016 : 345 – 353.
- [19] ZHOU J, XU W. End-to-end learning of semantic role labeling using recurrent neural networks[A]. Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)[C], 2015 : 1127 – 1137.
- [20] BIRD S, LOPER E, KLEIN E. Natural Language Processing with Python[M]. [S.l.] : O'Reilly Media Inc, 2009.

- [21] NOWOZIN S, CSEKE B, TOMIOKA R. f-gan: Training generative neural samplers using variational divergence minimization[A]. Advances in neural information processing systems[C], 2016 : 271–279.
- [22] MIRZA M, OSINDERO S. Conditional generative adversarial nets[J]. arXiv preprint arXiv:1411.1784, 2014.
- [23] ZHANG H, XU T, LI H, et al. Stackgan: Text to photo-realistic image synthesis with stacked generative adversarial networks[A]. Proceedings of the IEEE international conference on computer vision[C], 2017 : 5907–5915.
- [24] DONAHUE J, KRÄHENBÜHL P, DARRELL T. Adversarial feature learning[J]. arXiv preprint arXiv:1605.09782, 2016.
- [25] LI Y, SWERSKY K, ZEMEL R. Generative moment matching networks[A]. International Conference on Machine Learning[C], 2015 : 1718–1727.
- [26] PERARNAU G, VAN DE WEIJER J, RADUCANU B, et al. Invertible conditional gans for image editing[J]. arXiv preprint arXiv:1611.06355, 2016.
- [27] JIN Y, ZHANG J, LI M, et al. Create Anime Characters with A.I. ![J]. コミックマーケット 92, 2017, ウ 0 5 a.
- [28] ZHANG H, XU T, LI H, et al. Stackgan++: Realistic image synthesis with stacked generative adversarial networks[J]. IEEE transactions on pattern analysis and machine intelligence, 2018, 41(8) : 1947–1962.
- [29] JOHNSON J, GUPTA A, FEI-FEI L. Image Generation from Scene Graphs[J/OL]. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2018. <http://dx.doi.org/10.1109/CVPR.2018.00133>.
- [30] XU K, BA J, KIROS R, et al. Show, attend and tell: Neural image caption generation with visual attention[A]. International conference on machine learning[C], 2015 : 2048–2057.
- [31] PRESSMAN R S. Software engineering: a practitioner's approach[M]. [S.l.] : Palgrave macmillan, 2005.

- [32] BAHDANAU D, CHO K, BENGIO Y. Neural machine translation by jointly learning to align and translate[J]. arXiv preprint arXiv:1409.0473, 2014.
- [33] WEAVER L, TAO N. The Optimal Reward Baseline for Gradient-Based Reinforcement Learning[A]. UAI'01 : Proceedings of the Seventeenth Conference on Uncertainty in Artificial Intelligence[C], San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 2001 : 538–545.
- [34] CHEN Q, KOLTUN V. Photographic image synthesis with cascaded refinement networks[A]. Proceedings of the IEEE international conference on computer vision[C], 2017 : 1511 – 1520.
- [35] ISOLA P, ZHU J-Y, ZHOU T, et al. Image-to-image translation with conditional adversarial networks[A]. Proceedings of the IEEE conference on computer vision and pattern recognition[C], 2017 : 1125 – 1134.
- [36] JADERBERG M, SIMONYAN K, ZISSERMAN A, et al. Spatial Transformer Networks[A]. NIPS[C], 2015 : 1 – 14.
- [37] ODENA A, OLAH C, SHLENS J. Conditional image synthesis with auxiliary classifier gans[A]. Proceedings of the 34th International Conference on Machine Learning-Volume 70[C], 2017 : 2642 – 2651.
- [38] PAPINENI K, ROUKOS S, WARD T, et al. BLEU: a method for automatic evaluation of machine translation[A]. Proceedings of the 40th annual meeting on association for computational linguistics[C], 2002 : 311 – 318.
- [39] SALIMANS T, GOODFELLOW I, ZAREMBA W, et al. Improved techniques for training gans[A]. Advances in neural information processing systems[C], 2016 : 2234 – 2242.

致谢

以简短的文字表达作者对完成论文和学业提供帮助的老师、同学、领导、同事及亲属的感激之情。

附录 A 部分代码

A.1 界面代码

A.2 图片标注主函数代码

A.3 图像生成主函数代码