**研究生实验报告**

|  |  |
| --- | --- |
| 实验名称： | Multi-modal Graph Fusion for Named Entity Recognition with Targeted Visual Guidance  复现报告 |

|  |  |  |  |
| --- | --- | --- | --- |
| 学 员： | 姜龚青剑 | 学 号： | 23026234 |
| 培养类型： | 非军籍 | 年 级： | 研一 |
| 专 业： | 计算机科学与技术 | 所属学院： | 计算机学院 |
| 指导教员： | 唐晋韬 | 职 称： | 教授 |
| 实 验 室： | PDL | 实验日期： | 2023年12月 |

国防科技大学

1. 论文简介

**1.1 研究动机**

命名实体识别( Named entity recognition, NER ) 旨在发现自由文本中的命名实体，并用图像将其分类为预定义的类型。近年来，多模态命名实体识别( Multi-modal named entity recognition, MNER )已成为NER的一个重要研究方向。

传统的NER的输入主要基于文本，而MNER通过将图像作为额外输入，对输入文本进行补充的方式，显著增强了传统的基于文本的NER，这背后的假设是额外的视觉语义信息有助于解决文本中多义词的歧义问题。显然，如何充分利用额外的视觉信息是MNER的核心问题之一，是否能充分利用视觉信息直接影响模型性能的好坏。

现有的工作大致可以分为如下两类： 1 ) 将整幅图像编码成一个全局特征向量，用于增强单词的表示，或指导单词学习一个视觉感知的表示(基于RNN); 2 ) 将整幅图像分割为多个区域，并基于Transformer架构与文本序列进行交互。现有的方法虽然取得了一定的效果，但主要是将图像进行编码后作为隐式输入，用作文本的补充，而没有充分挖掘句子和图像之间细粒度的对应关系(如在图中指定对应单词对应物体的范围)。为了充分利用额外的多模态语义信息，需要解决两个问题：1 )如何构建一个统一的表征来弥合两种模态之间的语义差异；2 )如何在这种统一表征的基础上实现语义交互。

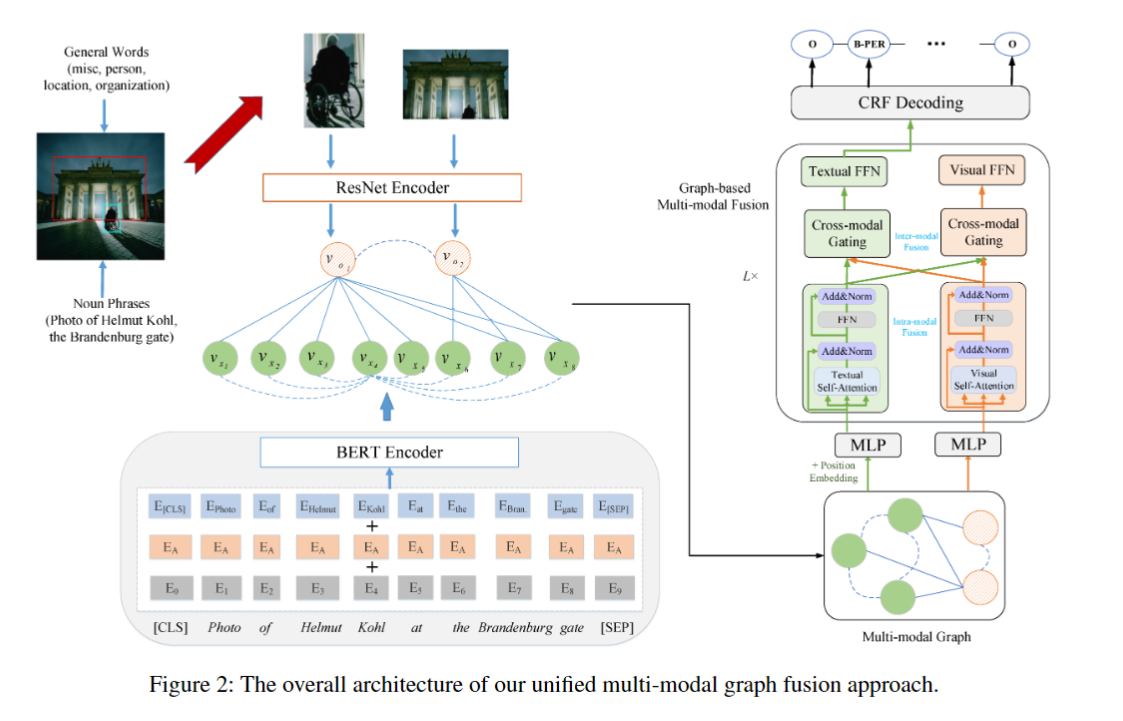
为解决上述问题，本实验中复现的论文提出了一种基于统一多模态图融合的MNER框架UMGF( Unified multi-modal graph fusion )，利用了图像和句子中词语的对应关系作为输入的补充，进一步增强MNER模型性能。

**1.2 方法介绍**

整体而言，UMGF首先将输入的句子和对应的图像用一个统一的多模态图来表示。在该图中，每个节点表示一个语义单元(文本单词或视觉对象)，并引入两种类型的边分别建模同一模态内语义单元之间的语义关系(模态内边)和不同模态语义单元之间的语义对应关系(模态间边)。

在这个统一的多模态图的基础上，我们再堆叠多个基于图的多模态融合层，这融合层迭代地执行节点间的语义交互来进行图编码。在这个过程中，我们区分了两种模态的参数，并依次进行模态内和模态间的融合来学习多模态节点表示。

最后，我们使用CRF解码器利用这些表示来执行实体标注。与之前的模型相比，UMGF能够充分地利用多模态语义单元之间的语义交互来进行NER。



UMGF整体框架如上图所示。具体而言，对多模态融合图(上图右下角)，其中每个节点代表一个语义对象，如文本词或视觉对象。首先将句子分词，得到的每一个单词作为一个独立的文本节点加入多模态图；然后使用可视化工具包为每个单词在图中检测对应的视觉对象，并画出边界框。考虑到图中的视觉对象可能不存在对应的单词，为保证每个视觉对象都有对应的文本节点，UMGF定义了几个通用的文本节点，如人、地点、组织和其他。随后，将所有检测到的视觉对象作为独立的视觉节点加入多模态图。对于多模态图中的边，任意同一模态的两个节点通过模态内边( intra-modal edge )相连，每个表示任意名词短语的文本节点和对应的视觉节点通过模态间边( inter-modal edge )连接；除此之外每个单词和用通用单词检测到的视觉节点用一个模态间边连接。

在将多模态图输入到堆叠的融合层之前，UMGF引入了两个带ReLU激活函数的多层感知器( MLP )，将来自两个模态的不同特征投影到同一空间。具体来说，对于每个文本节点，将初始状态初始化为BERT编码器编码得到的嵌入和位置编码之和，然后通过文本MLP变换为维度d的向量。对于每个视觉节点，将初始状态用ResNet中的视觉特征进行初始化，然后通过视觉MLP变换为维度d的向量。

对多模态图进行初始化后，通过L层多模态融合层进行编码。每个融合层依次进行模态内和模态间融合来更新所有节点状态。这样，最终的节点状态同时包含有同一模态内的上下文的语义信息和跨模态的语义信息。由于视觉节点和文本节点是包含不同模态信息的两类语义单元，在更新过程中对两种不同模态的节点使用不同的参数。

具体来说，模态内融合利用了自注意力机制，通过收集来自同一模态的邻居节点的消息来生成每个节点的上下文表示；第l层的上下文有关的文本表征由MultiHeadAttention生成，如下式所示：



是多头自注意力函数；第l层的上下文有关的视觉表征也用类似的方法生成。



模态间融合利用了跨模态门控机制，通过元素级操作来收集每个节点的跨模态邻居的语义信息。具体而言，文本节点的表示由下式得到：



其中，为文本节点相邻的视觉节点，和为参数矩阵。类似的，视觉节点的表示由下式得到：



其中，为文本节点相邻的视觉节点，和为参数矩阵。这样做的好处是能够更好地根据各模态的语境表征来判断模态间的融合程度。

最后，采用基于位置的前馈网络FNN生成文本节点状态和视觉节点状态。



由于视觉信息已经通过多个基于图的多模态融合层融合到所有文本节点中，因此可以用CRF解码器进行条件序列标注，将文本节点分类。

1. 实验目的

根据论文的实验设置，复现UMGF在twitter2015和twitter2017两个MNER数据集上的实验结果。对人(PER)，地点(LOC)，组织(ORG)，其他(OTHER)四个单一类型和总体的精确率(Pre)、召回率( Rec)和F1分数。同时在自数据集和跨数据集的场景下进行实验，以验证模型的泛用型和有效性。

同时为了对比UMGF对于UMT的改进程度，还在相同的实验环境和基准下复现了UMT的部分实验结果，用于对比。

1. 实验步骤

**3.1 代码**

UMGF代码从<https://github.com/TransformersWsz/UMGF>拉取

UMT代码从<https://github.com/jefferyYu/UMT>拉取

修改部分详情见<https://github.com/hydro-kun/2023winter-jgqj-NLPhomework>

**3.2 数据准备**

UMT：Twitter2015和twitter2017数据集的文本部分已提供，图片从<https://drive.google.com/file/d/1PpvvncnQkgDNeBMKVgG2zFYuRhbL873g/view>下载；需要修改run\_mtmner\_crf.py中552行和554行的图片路径。下载ResNet-152，<https://download.pytorch.org/models/resnet152-b121ed2d.pth>，并放到UMT/resnet目录下。

UMGF：Twitter2015和twitter2017数据集的文本部分已提供，图片部分下载同UMT，需要按照[One-Stage Visual Grounding](https://github.com/TransformersWsz/onestage_grounding/blob/master/README.md)的方法为每个单词在图中检测对应的视觉对象，并画出边界框，并将图片放在./data/twitter2015/image下。(或直接从[twitter2015\_img.tar.gz(password: l75t)](https://pan.baidu.com/s/1DCACHmDKYiW21Vnmn6YIvQ?pwd=l75t) 和[twitter2017\_img.tar.gz(password: 2017)](https://pan.baidu.com/s/173PbLBFWDEHjWH3zLtzZww?pwd=2017)下载)

**3.3 对比系统构建**

使用UMGF开发者在github提供的训练存档点model.pt，对存档点在相同数据集和不同数据集上进行测试。UMT仅提供了测试输出，因此使用论文中的超参数进行训练，并在相同数据集和不同数据集上进行测试。

具体代码和运行说明详见<https://github.com/hydro-kun/2023winter-jgqj-NLPhomework>

1. 实验环境

**4.1 实验软硬件平台**

硬件环境：GPU为GeForce RTX 2080，CPU为Intel(R) Xeon(R) Gold 6226R CPU @ 2.90GHz，内存128G。

软件环境：Ubuntu版本为22.04，Python版本为3.7，UMGF的Pytorch版本为1.7.1，UMT的Pytorch版本为1.0.0，CUDA版本为11.2。UMGF为2023.11.23从Github下载的版本，UMT为2023.12.11从Github下载的版本。两个框架使用相同的CUDA、Python版本。

**4.2 实验超参数设置**

UMGF与论文的超参数设置保持一致，文本嵌入用BERT-base初始化为768维的向量，视觉嵌入用ResNet152初始化为2048维的向量。经过MLP变换为512维的向量。多头注意力头数为8，学习率为0.0001，随机失活率为0.5，误差系数为0.5，多模态图融合层数为2，batch\_size设置为16，epoch数量设置为60。

**4.3 评估指标**

定义：

TP（True Positive）：被正确预测的正例。即该数据的真实值为正例，预测值也为正例的情况；

TN（True Negative）：被正确预测的反例。即该数据的真实值为反例，预测值也为反例的情况;

FP（False Positive）：被错误预测的正例。即该数据的真实值为反例，但被错误预测成了正例的情况；

FN（False Negative）：被错误预测的反例。即该数据的真实值为正例，但被错误预测成了反例的情况。

则本实验中用到的度量为：准确率（Accuracy，ACC）、精确率（Precision，PRE）、召回率（Recall，REC），F1分数。具体定义如下式所示：



对于两种框架，在每种实体分类（PER、LOC、ORG、OTHER）的F1分数和总体的准确率和召回率和F1分数上进行比较。

1. 实验结果及分析

**5.1 自领域场景(Self-domain Scenario)**

对UMGF在twitter2015的训练集上训练，在twitter2015的测试集上测试；对UMT在twitter2015的训练集上训练，在twitter2015的测试集上测试，比较这两个框架在单一类型上的F1分数，和总体的准确率，召回率，F1分数。在twitter2015数据集上的实验结果如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Approaches | Twitter-2015 | | | | | | |
| Single Type(F1) | | | | Overall | | |
| PER | LOC | ORG | OTHER | Pre. | Rec. | F1 |
| UMT(论文) | 85.24 | 81.58 | 63.03 | 39.45 | 71.67 | 75.23 | 73.41 |
| UMT(复现) | 85.05 | 80.88 | 61.31 | 39.45 | 70.22 | 75.78 | 72.89 |
| UMGF(论文) | 84.26 | 83.17 | 62.45 | 42.42 | 74.49 | 75.21 | 74.85 |
| UMGF(最佳检查点) | 85.19 | 82.34 | 61.99 | 42.13 | 73.18 | 75.78 | 74.46 |
| UMGF(复现) | 84.76 | 81.30 | 59.54 | 40.34 | 72.39 | 74.26 | 73.31 |

Table 1：在twitter2015数据集上的自领域场景下的测试结果，其中UMT(论文)和UMGF(论文)为从论文中摘取的数据，UMT(复现)和UMGF(复现)为复现结果，UMGF(最佳检查点)为对作者提供的存档点测试的结果

同上，在twitter2017数据集上的实验结果如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Approaches | Twitter-2017 | | | | | | |
| Single Type(F1) | | | | Overall | | |
| PER | LOC | ORG | OTHER | Pre. | Rec. | F1 |
| UMT(论文) | 91.56 | 84.73 | 82.24 | 70.10 | 85.28 | 85.34 | 85.31 |
| UMT(复现) | 88.57 | 82.91 | 82.18 | 64.26 | 82.36 | 84.01 | 83.18 |
| UMGF(论文) | 91.92 | 85.22 | 83.13 | 69.83 | 86.54 | 84.50 | 85.51 |
| UMGF(最佳检查点) | 92.15 | 84.26 | 81.92 | 67.34 | 86.07 | 84.61 | 85.34 |
| UMGF(复现) | 89.92 | 83.85 | 82.22 | 68.45 | 84.25 | 84.68 | 84.47 |

Table 2：在twitter2017数据集上的自领域场景下的测试结果，其中UMT(论文)和UMGF(论文)为从论文中摘取的数据，UMT(复现)和UMGF(复现)为复现结果，UMGF(最佳检查点)为对作者提供的存档点测试的结果

可见，在自领域场景下，UMGF框架的复现结果比论文中的数据或最佳检查点的测试结果的F1整体分数要差1~2%，在PER, LOC, ROG, OTHER单一类型的F1分数上，最大相差达到了2%。在超参数设置一致的情况下，可能是因为硬件环境和软件环境的差异导致的，本次实验在RTX3090上运行，CUDA版本为11.2，而论文的硬件环境为TITAN RTX，CUDA版本为10.2。另一方面，本次实验的对比系统UMT也没有达到论文的实验效果，UMT框架的复现结果比论文中的数据的F1整体分数差2%，而在单一类型下的F1分数的差距则在2%~6%左右。在这种情况下，对比UMGF和对比模型UMT的复现结果。

UMGF的论文结果与UMT的论文结果相比，在单一类型的F1分数各有优劣，有0.5%~3%左右的浮动，总体的F1分数则比UMT好0.2%~1%左右。UMGF的复现与UMT的复现结果相比，单一类型的F1分数有0~1%的提升，总体的F1分数也有1%的提升，这是因为UMGF构建的统一的多模态图可以更好地捕捉文本和目标视觉节点间的语义交互，更好地利用额外的视觉信息，在不同数据集上都有更稳定的表现。

**5.2 跨领域场景(Cross-domain Scenario)**

对UMGF在twitter2015的训练集上训练，在twitter2017的测试集上测试；对UMT在twitter2015的训练集上训练，在twitter2017的测试集上测试，比较这两个框架在单一类型上的F1分数，和总体的准确率，召回率，F1分数。在twitter2015上训练，twitter2017数据集上测试的实验结果如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Approaches | Twitter-2015 -> Twitter2017 | | | | | | |
| Single Type(F1) | | | | Overall | | |
| PER | LOC | ORG | OTHER | Pre. | Rec. | F1 |
| UMT(论文) | 81.24 | 67.89 | 39.52 | 31.87 | 67.80 | 55.23 | 60.87 |
| UMT(复现) | 81.33 | 70.71 | 42.39 | 31.43 | 44.89 | 58.54 | 50.81 |
| UMGF(论文) | 81.83 | 72.25 | 41.20 | 32.00 | 69.88 | 56.92 | 62.74 |
| UMGF(最佳检查点) | 80.24 | 67.01 | 40.00 | 29.95 | 65.32 | 56.58 | 60.64 |
| UMGF(复现) | 81.18 | 67.19 | 42.22 | 30.00 | 66.83 | 57.39 | 61.75 |

Table 3: 在跨领域场景下，在twitter2015上训练，twitter2017数据集上测试的实验结果，其中UMT(论文)和UMGF(论文)为从论文中摘取的数据，UMT(复现)和UMGF(复现)为复现结果，UMGF(最佳检查点)为对作者提供的存档点测试的结果

同上，在twitter2017上训练，twitter2015数据集上测试的实验结果如下：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Approaches | Twitter-2017 -> Twitter2015 | | | | | | |
| Single Type(F1) | | | | Overall | | |
| PER | LOC | ORG | OTHER | Pre. | Rec. | F1 |
| UMT(论文) | 80.34 | 71.30 | 47.97 | 20.13 | 64.67 | 63.59 | 64.13 |
| UMT(复现) | 78.15 | 71.47 | 48.03 | 22.92 | 62.11 | 63.74 | 62.91 |
| UMGF(论文) | 79.62 | 71.94 | 49.48 | 20.24 | 67.00 | 62.81 | 66.21 |
| UMGF(最佳检查点) | 80.10 | 71.72 | 47.25 | 18.75 | 69.66 | 62.01 | 65.61 |
| UMGF(复现) | 79.28 | 71.59 | 47.75 | 20.14 | 68.10 | 62.90 | 65.40 |

Table 4：在跨领域场景下，在twitter2017上训练，twitter2015数据集上测试的实验结果，其中UMT(论文)和UMGF(论文)为从论文中摘取的数据，UMT(复现)和UMGF(复现)为复现结果，UMGF(最佳检查点)为对作者提供的存档点测试的结果

在自领域场景下，UMT的复现结果和论文结果在twitter2017数据集上测试时出现了较大的差异，在单一类型的F1分数上，复现结果与论文的数据各有优劣，在PER, LOC, ORG三种类型上均有0.1%~3%的提升，在OTHER类型上则小了0.4%。但在整体的F1分数和准确率上出现了10%~20%的差异；而在twitter2015数据集上测试时则是在单一类型和整体的F1分数上各有优劣，存在0~2%左右的上下浮动。关于这点，这可能是因为twitter2017测试集相对较小，而且模型的泛化性较差的原因，因此在不同数据集上测试时效果下降明显。对于UMGF框架，复现结果则与论文结果吻合，在不同类型和整体的F1分数上，浮动在1%以内。

对于UMGF的论文结果和UMT的论文结果，UMGF有2%~5%的优势；对于UMGF的复现结果和UMT的复现结果，UMGF则有2%~10%左右的优势，这有可能是因为twitter2017数据集存在一些没有对应图片的文本，而且twitter2017的数据集较小，因此在缺失了一些模态的twitter2017上，利用多模态图来进行模态间交互和融合的UMGF的效果表现更好。

六、实验心得

在这次实验中，我通过对UMGF论文的复现和UMT论文的复现，熟悉了论文复现流程，学习了有关MNER相关的知识，了解了多模态和Transformer在NER上的应用、基于GNN的多模态融合技术、和如何将不同模态数据转化为相同维数的嵌入。同时，通过对论文的复现和实验，我的代码能力也得到了提升；通过分析复现结果与论文结果的异同，我进一步学会了如何分析实验结果，并思考造成复现结果和实验结果不同的原因。总之，我在这次实验中学到了很多东西，受益匪浅。