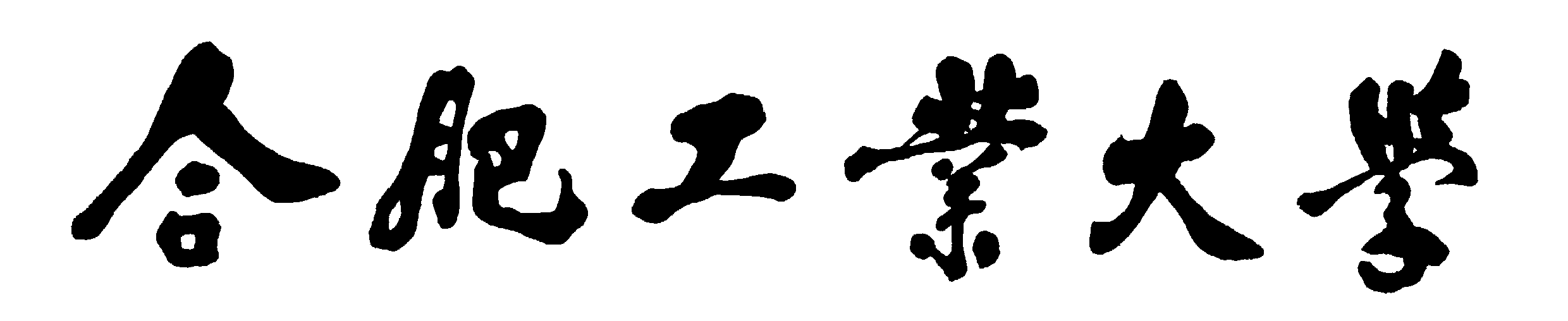
****

计算机与信息学院

人工智能原理期末报告

|  |  |
| --- | --- |
| 专 业 班 级 | 计算机科学与技术22-4班 |
| 学生姓名及学号 | 侯腾跃 2022217477 |
| 任 课 教 师 | 张 赞 |
| 2024 ~2025学年第 一 学期 | |

Exploring Chain-of-Thought for Multi-modal Metaphor Detection

探索多模态隐喻检测的思路

（Yanzhi Xu∗ ,Yueying Hua∗ ,Shichen Li and Zhongqing Wang† Natural Language Processing Lab, Soochow University, Suzhou, China

*{yzxuxyz,yyhua1224}@stu.suda.edu.cn；[scli\_21@outlook.com](mailto:scli_21@outlook.com)*

*[wangzq@suda.edu.cn](mailto:wangzq@suda.edu.cn）)*[）](mailto:wangzq@suda.edu.cn）)

侯腾跃、2022217477、计算机与信息系

计算机科学与技术、2022级4班

摘 要：隐喻在广告和互联网梗图中很常见。然而，互联网梗图的自由形式常常导致缺乏高质量的文本数据。隐喻检测需要对文本和视觉元素进行深入的解释，这需要大量的常识知识，这对语言模型来说是一个挑战。为了应对这些挑战，我们提出了一个紧凑的框架C4MMD，它采用了链式思考（Chain-of-Thought，CoT）方法来进行多模态隐喻检测。具体来说，我们的方法设计了一个受CoT启发的三步过程，从多模态大型语言模型（MLLMs）中提取并整合知识到较小的模型中。我们还开发了一个模态融合架构，将大模型中的知识转化为隐喻特征，并通过辅助任务提高模型性能。在MET-MEME数据集上的实验结果表明，我们的方法不仅有效增强了小模型的隐喻检测能力，而且超越了现有模型。据作者所知，这是第一次系统性地利用MLLMs进行隐喻检测任务。方法的代码已在GitHub上公开提供

关键词：多模态隐喻检测 (Multi-modal Metaphor Detection)；链式思考 (Chain-of-Thought, CoT)；大型语言模型 (Large Language Models, LLMs)；多模态大型语言模型 (Multi-modal Large Language Models, MLLMs)；C4MMD框架；隐喻特征 (Metaphor Features)

引言

隐喻在我们的日常表达和写作中非常普遍，对自然语言处理（NLP）的下游任务有很大的影响，如语义理解、情感分析等。随着社交媒体的兴起，多模态隐喻引起了人们的兴趣，因此出现了几个多模态隐喻的数据集。与单一模态检测相比，多模态隐喻检测不仅在句子中发现隐喻，还将其归类为图像主导、文本主导或互补。第二个主要挑战来自于文本内容的质量差，主要来源于社交媒体上的广告和梗图。文本赋予图像更多的隐喻特征。最近的努力使用OCR技术提取图像中的文本。然而，仅依赖OCR将它们转换为平行文本会导致文本位置信息的丢失。

# 问题背景和相关工作介绍

## 研究内容

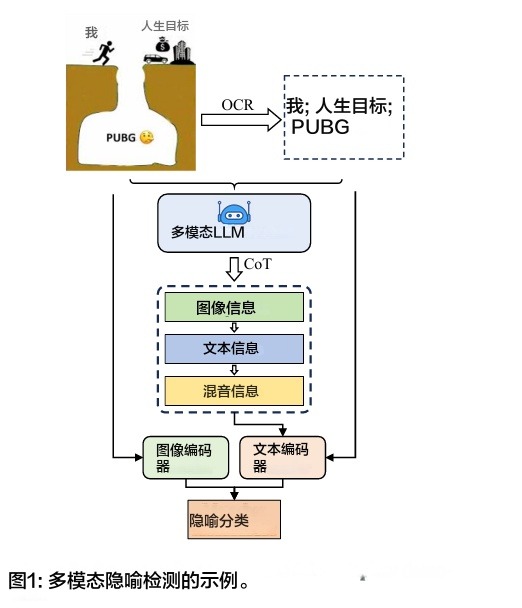


图1 多模态隐喻检测的示例

从形式上看，多模态隐喻检测任务属于典型的多模态分类问题。给定一组跨模态样本对，该任务旨在确定是否存在隐喻特征并提供分类结果。我们的工作重点是检测图像-文本对中的隐喻，因此该任务被表述为：

|  |  |
| --- | --- |
| *Y* = *F* (*xI , xT* ) | (1) |

其中 xI 和 xT 分别表示图像和文本模态的特征。我们的目标是采用一种更有效的方法 F，以确保分类结果 Ŷ 更紧密地与真实值 Y 对齐。为此，我们开发了一个细粒度分类器。具体来说，我们将补充的图像描述向量与视觉模态输入向量整合作为图像向量，将文本分析向量与文本输入向量整合作为文本向量，并合并这些向量形成跨模态向量。这三个向量然后用于分类目的。分类器使用跨模态向量来检测隐喻，使用图像向量来识别图像主导的内容，使用文本向量来识别文本主导的内容。这种方法增强了使用多模态特征进行精确隐喻检测的能力。

如图2所示，C4MMD的架构由两个主要组件组成：知识总结模块和下游多模型融合结构。在知识总结模块中，我们向MLLM提供一对图像-文本，并设计了一个带有CoT提示的三步模板。前两个模板指导MLLM专注于单一模态——无论是文本还是图像，忽略另一个来生成解释和见解。在第三步中，MLLM结合了两种模态的见解。基于之前的分析，模型实现了对两种模态的更深入理解和更全面的整合。

在从MLLM获得不同模态的额外文本信息后，我们将这些信息与原始文本合并形成文本输入。同样，输入图像被视为视觉模态输入。然后，模型通过模态特定的编码器处理这些输入以派生特征向量。

在多模型融合模块中，我们缩放和组合来自不同模态的向量，并定义向量，将文本分析向量与文本输入向量结合作为文本向量，并合并这些向量形成跨模态向量。这三个向量然后用于分类目的。分类器使用跨模态向量来检测隐喻，使用图像向量来识别图像主导的内容，使用文本向量来识别文本主导的内容。这种方法增强了使用多模态特征进行精确隐喻检测的能力。

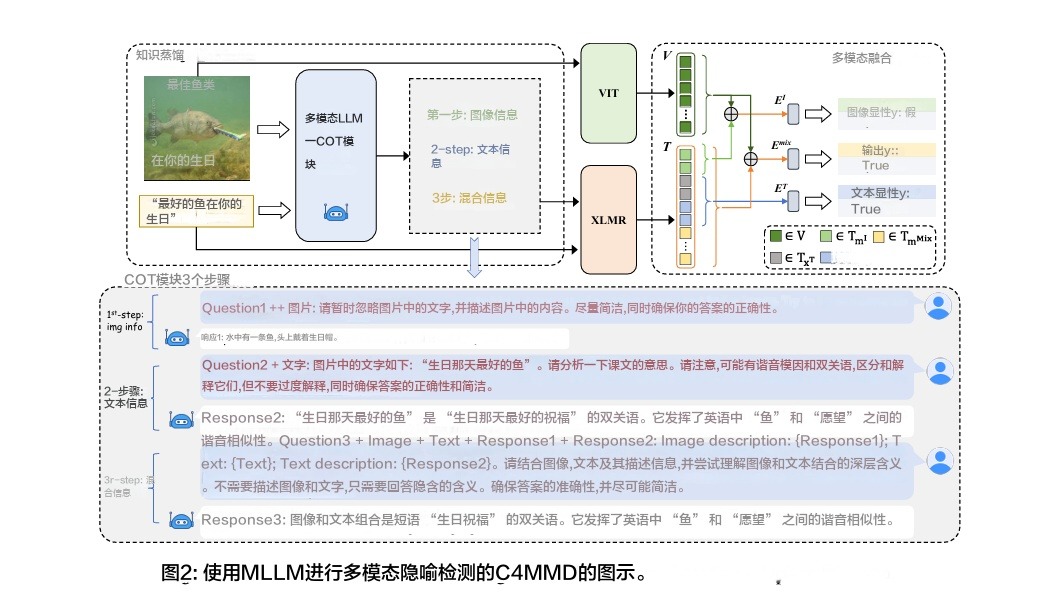


图2 使用MLLM进行多模态隐喻检测的C4MMD的图示

为了指导MLLM生成更高质量、更具信息量的特征，我们采用了CoT提示。这种方法指导MLLM分析跨模态的更深层次信息。然后我们利用这些补充信息来帮助较小的模型实现更好的语义理解和模态融合。总之，我们构建了如下的三步提示。

步骤1. 首先，为确保模型专注于理解图像中的对象、场景或其他视觉元素（由 xI 表示）而不受到文本特征的干扰，我们引导模型根据模板问题1来理解解释图像信息。

问题1：请暂时忽略图像中的文本，描述图像内容。尽量简洁，同时确保答案的正确性。

这一步可以表述如下：

|  |  |
| --- | --- |
| *mI* = *MLLM* (*xI* ,Question1) | (2) |

步骤2：接下来，为了更好地理解文本（由 xT 表示）中隐藏的含义，同时排除任何图像特征的干扰，我们引导模型根据模板问题2来理解解释文本信息。

问题2：请分析文本的含义。请注意可能存在同音异义的梗和双关语，区分并解释它们，但不要过度解释，同时确保答案的正确性和简洁性。这一步可以表述如下：

|  |  |
| --- | --- |
| *mI* = *MLLM* (*xI* ,Question2) | (3) |

步骤3：最终，我们希望模型综合前两步（由 mI 和 mT 表示）的结果，并进一步整合图像和文本特征（xI 和 xT），从而获得更深层次的跨模态交互信息。我们鼓励模型根据模板融合不同模态的特征。

问题3：请结合图像、文本及其描述信息，尝试理解图像和文本结合的深层含义。无需描述图像和文本，只回答隐含的含义。确保答案的准确性，并尽量简洁。这一步可以表述如下：

|  |  |
| --- | --- |
| *mMix* = *MLLM* (*xI , xT , mI , mT ,* Question3) | (4) |

在从MLLM获得额外模态信息后，我们设计了一个模态融合架构，以促进模态间整合，并有效利用MLLM产生的额外信息来增强隐喻检测能力。我们使用图像编码器和文本编码器来获取图像 xI 和文本 xT 的向量化编码，以便于后续的跨模态融合。考虑到MLLM生成的额外信息以文本形式呈现，我们将其视为额外的视觉 mI、文本 mT 和混合 mMix 信息。这些信息与原始文本连接，然后通过文本编码器进行处理。

|  |  |
| --- | --- |
| V =ViT-Encoder(*xI*), T =XLMR-Encoder(*xTmT*, *mI*,*mMix*) | (5) |

其中 V 是图像编码器的输出，T 是文本编码器的输出。

为了使文本编码器在计算过程中区分来自不同模态的文本，我们采用了一种类似于BERT的段落编码方法，为每个模态的文本添加额外的可学习参数向量。

进入文本编码器的第 i 个词 xi（xi ∈ {*xTmT*, *mI*,*mMix*}）的向量化编码 Embi 可以表示如下：

|  |  |
| --- | --- |
| *Emb* = *E* (*x* ) + *E* (*i*) + *E (segment*(*x ))* | (6) |

其中 ET、EP 和 ES 分别表示可学习的标记嵌入、位置编码和段落嵌入矩阵。术语 segment(xi) ∈ (0,1,2,3) 指的是词 xi 的段落编码。

我们使用图像编码器和文本编码器来获取图像 xI 和文本 xT的向量化编码，以便于后续的跨模态融合。考虑到MLLM生成的额外信息以文本形式呈现，我们将其视为额外的视觉 mI、文本 mT 和混合 mMix 信息。这些信息与原始文本连接，然后通过文本编码器进行处理。我们的多模态隐喻检测模型的训练目标涉及三个不同损失函数的整合，分别表示为LI、LT和LM根据模型的预测结果和真实值，LCE是交叉熵损失函数。为了优化整体性能，我们最终的损失函数是：

|  |  |
| --- | --- |
| Lsum=0.5⋅LI+0.5⋅LT+LM*Lsum*​=0.5⋅*LI*​+0.5⋅*LT*​+*LM*​ | (7) |

## 问题背景介绍

在自然语言处理（NLP）领域，隐喻是一种常见的语言现象，它通过将一个概念与另一个不同但有相似性的概念联系起来，以富有想象力的方式表达思想或情感。隐喻的识别和理解对于提高机器对人类语言的理解和处理能力至关重要。然而，隐喻的检测和解释面临挑战，尤其是当隐喻出现在多模态环境中，如结合文本和图像的互联网梗图时。随着社交媒体的兴起，多模态隐喻（结合文本和图像的隐喻）变得越来越普遍。这些隐喻通常包含丰富的视觉和文本信息，需要模型具备跨模态理解能力。此外，由于互联网梗图的自由形式，这些文本数据往往质量不高，给隐喻检测带来了额外的难度。

## 已有的相关工作

早期的隐喻检测任务被限制在单一模态，并采用了基于规则约束和隐喻词典的方法（Fass, 1991; Krishnakumaran and Zhu, 2007; Wilks et al., 2013）。随着自然语言处理（NLP）领域的蓬勃发展，基于机器学习的方法（Turney et al., 2011; Shutova et al., 2016）和基于神经网络的方法（Mao et al., 2019; Zayed et al., 2020）相继出现。随着Transformer（Vaswani et al., 2017）的引入，基于预训练模型的方法逐渐取代了以前的方法，成为当前的主流方法（Cabot et al., 2020; Li et al., 2021; Lin et al., 2021）。Ge et al. (2023)将当前的努力归类为四个主要方向，即额外数据和特征方法（Shutova et al., 2016; Gong et al., 2020; Kehat and Pustejovsky, 2021）、语义方法（Mao et al., 2019; Choi et al., 2021; Su et al., 2021; Zhang and Liu, 2022; Li et al., 2023b; Tian et al., 2023a）、基于上下文的方法（Su et al., 2020; Song et al., 2021）和多任务方法（Chen et al., 2020; Le et al., 2020; Mao et al., 2023; Badathala et al., 2023; Zhang and Liu, 2023; Tian et al., 2023b），其中语义方法和多任务方法已成为最近研究的主要焦点。

# 算法介绍

## 解题思路

1. 数据准备 (data\_divide.py)

数据加载：从MET-Meme数据集加载原始数据，该数据集包含中文和英文的图像及其对应的文本和标签。

数据分割：将数据集简单分割为训练集、验证集和测试集，比例为6:2:2。

数据增强：通过读取图像文件名和对应的文本数据，将文本信息添加到数据集中。

数据转换：将数据从列表格式转换为字典格式，并保存为JSON文件，以便于后续处理。

2. CoT模块 (CoT\_module.py)

加载模型：使用AutoConfig和AutoTokenizer从预训练模型路径加载模型配置和分词器。

数据处理：对于训练、验证和测试数据集中的每一条数据，执行以下步骤：

图像描述：忽略图像中的文本，描述图像内容。

文本分析：分析图像中的文本，注意同音异义和双关语。

图像和文本融合：结合图像描述和文本分析，理解图像和文本的深层含义。

知识提取：使用大型语言模型（如InternLM-XComposer）生成补充信息，这些信息用于辅助下游模型进行更准确的隐喻检测。

数据保存：将生成的补充信息保存到JSON文件中，供后续训练使用。

3. 模型训练 (C4MMD\_train.py)

模型初始化：初始化视觉模型（如ViT）和语言模型（如XLM-Roberta），并组合成一个多模态模型。

数据加载：使用自定义的DataLoader和Collator加载和处理数据。

模型训练：使用AdamW优化器和交叉熵损失函数进行训练。采用线性调度器和预热策略调整学习率。

模型评估：在验证集上评估模型性能，记录最佳验证F1分数。如果模型在验证集上的性能没有改进，则提前停止训练。

结果保存：保存最佳模型参数，并记录训练过程中的关键指标。

4. 模型评估

评估函数：定义评估函数evaluation，计算准确率、召回率、精确率和F1分数。

测试集评估：在测试集上运行评估函数，得到最终的模型性能指标。

结果记录：将测试集上的最佳F1分数和其他指标记录到日志文件和JSON文件中。

5.多次运行

多次训练：为了确保结果的稳定性，算法多次运行训练过程，记录每次运行的平均性能指标。

性能分析：分析多次运行的结果，计算平均F1分数、召回率、精确率和准确率。

通过这些步骤，算法能够有效地训练和评估一个多模态隐喻检测模型，利用链式思考方法和大型语言模型生成的补充信息来提高模型的理解和分类能力。

## 算法伪代码和细节介绍

# 伪代码：CoT模块处理流程

# 设置环境变量

设置 CUDA\_VISIBLE\_DEVICES 为 "0"

设置 CUDA\_LAUNCH\_BLOCKING 为 "1"

# 初始化日志记录器

创建 logger 实例

设置日志格式和处理器

# 解析命令行参数

使用 HfArgumentParser 解析 ModelArguments, DataTrainingArguments, Seq2SeqTrainingArguments

# 设置随机种子

根据 training\_args.seed 设置随机种子

# 加载模型配置

从 model\_args.model\_name\_or\_path 加载 AutoConfig 配置

设置 config.pre\_seq\_len 和 config.prefix\_projection

# 初始化模型和分词器

创建 InternLMXComposerQForCausalLM 模型实例

从 model\_args.model\_name\_or\_path 加载 AutoTokenizer 实例

设置模型的 tokenizer 和最大长度

# 定义保存 JSON 数据的函数

Function save\_json(file\_name, data)

将 data 转换为 JSON 格式并保存到 file\_name 指定的文件

# 定义加载 JSON 数据的函数

Function load\_json(file\_path)

从 file\_path 指定的文件加载 JSON 数据

# 定义 CoT 模块处理流程

Function process\_data(file\_list, origin\_data\_path, new\_data\_path, image\_file\_path)

For 每个文件名 in file\_list:

检查是否存在新数据路径下的 JSON 文件

如果存在，加载新数据

否则，加载原始数据路径下的 JSON 文件

For 每行数据 in 数据:

If 已经包含 'internlm\_mix\_info'，则跳过

构造图像描述问题 Question1 并获取模型回答 response1

构造文本分析问题 Question2 并获取模型回答 response2

构造图像和文本融合问题 Question3 并获取模型回答 response3

更新数据行

每处理 100 行数据，保存一次中间结果

保存处理后的数据到新数据路径下的 JSON 文件

Print "finish!"

# 主程序

process\_file\_list = ['train\_data', 'val\_data', 'test\_data']

origin\_data\_path = "原始数据路径"

new\_data\_path = "处理后数据路径"

image\_file\_path = "图像文件路径"

调用 process\_data(process\_file\_list, origin\_data\_path, new\_data\_path, image\_file\_path)

# 伪代码：数据集分割和处理

# 初始化参数

load\_path = "从met-meme数据集下载的数据路径"

random.seed(42)

# 定义函数：读取文件

Function read\_file(file\_name, image\_file, chinese=False)

input: file\_name (str) - 要读取的文件

input: image\_file (str) - 图像文件的路径前缀

input: chinese (bool) - 是否处理中文数据

output: output\_list (list) - 处理后的数据列表

output: head (list) - 文件标题行

Open file and read data

For each line in the file:

If it's the header:

head = line

Else:

If chinese:

Modify line[0] by adding image\_file prefix and replacing underscores

Else:

Modify line[0] by adding image\_file prefix

Append modified line to output\_list

Shuffle output\_list

Return output\_list, head

# 定义函数：分割数据集

Function device\_dataset(data)

input: data (list) - 原始数据列表

output: train\_data (list) - 训练集数据

output: val\_data (list) - 验证集数据

output: test\_data (list) - 测试集数据

Calculate train\_num as 60% of data length

Calculate val\_num as 20% of data length

Split data into train, val, and test sets

Return train\_data, val\_data, test\_data

# 定义函数：写入文件

Function write\_file(file\_name, data, head)

input: file\_name (str) - 要写入的文件名

input: data (list) - 要写入的数据列表

input: head (list) - 文件标题行

Open file for writing

Write head to file

For each line in data:

Write line to file

# 定义函数：添加文本数据

Function add\_text(data, text\_data, head)

input: data (list) - 原始数据列表

input: text\_data (dict) - 文本数据字典

input: head (list) - 文件标题行

output: updated\_data (list) - 添加文本后的数据列表

output: updated\_head (list) - 更新后的标题行

For each line in data:

Get image name from line[0]

Get corresponding text from text\_data

Insert text into line at position 1

Insert 'text' into head at position 1

Return updated\_data, updated\_head

# 定义函数：转换列表到字典

Function convert\_list2dict(data, head)

input: data (list) - 数据列表

input: head (list) - 标题行

output: new\_data (list) - 转换后的字典列表

Create new\_data list

For each line in data:

Create a dictionary with keys from head and values from line

Append dictionary to new\_data

Return new\_data

# 定义函数：保存JSON文件

Function save\_json(file\_name, data)

input: file\_name (str) - 要保存的文件名

input: data (dict) - 要保存的数据字典

Open file for writing in JSON format

Write data to file

# 主程序

# 读取中文和英文文本数据

chinese\_text\_data, \_ = read\_file(f'{load\_path}/C\_text.csv', 'Chinese/', chinese=True)

english\_text\_data, \_ = read\_file(f'{load\_path}/E\_text.csv', 'English/')

# 将文本数据转换为字典

chinese\_text\_data = {line[0]: line[1] for line in chinese\_text\_data}

english\_text\_data = {line[0]: line[1] for line in english\_text\_data}

# 读取中文和英文标签数据，并添加文本数据

chinese\_data, c\_head = read\_file(f'{load\_path}/label\_C.csv', 'Chinese/', chinese=True)

chinese\_data, c\_head = add\_text(chinese\_data, chinese\_text\_data, c\_head)

english\_data, e\_head = read\_file(f'{load\_path}/label\_E.csv', 'English/')

english\_data, e\_head = add\_text(english\_data, english\_text\_data, e\_head)

# 分割中文和英文数据集

c\_train, c\_val, c\_test = device\_dataset(chinese\_data)

e\_train, e\_val, e\_test = device\_dataset(english\_data)

# 合并训练集、验证集和测试集

train\_data = c\_train + e\_train

val\_data = c\_val + e\_val

test\_data = c\_test + e\_test

# 随机打乱数据集

random.shuffle(train\_data)

random.shuffle(val\_data)

random.shuffle(test\_data)

# 保存数据集为JSON文件

save\_json('../data/train\_data.json', convert\_list2dict(train\_data, c\_head))

save\_json('../data/val\_data.json', convert\_list2dict(val\_data, c\_head))

save\_json('../data/test\_data.json', convert\_list2dict(test\_data, c\_head))

算法：C4MMD 训练过程

输入参数：

language\_model (str): 语言模型的路径

vision\_model (str): 视觉模型的路径

train\_data (str): 训练数据的路径

val\_data (str): 验证数据的路径

test\_data (str): 测试数据的路径

batch\_size (int): 每个批次的样本数量

epochs (int): 训练的轮数

learning\_rate (float): 学习率

输出参数：

best\_model (dict): 保存最佳模型的参数

f1\_log (dict): 记录训练过程中的关键指标

数据集分割和处理

read\_file：读取CSV文件并返回处理后的数据列表和标题行。

device\_dataset：将数据集分割为训练集、验证集和测试集。

write\_file：将数据写入CSV文件。

add\_text：将文本数据添加到数据列表中，并更新标题行。

convert\_list2dict：将数据列表转换为字典列表，便于后续处理。

save\_json：将数据保存为JSON文件。

CoT模块处理流程

输入参数:

file\_name (str): 要保存的文件名。

data (dict): 要保存到文件的数据。

file\_path (str): 要加载的 JSON 文件的路径。

file\_list (list): 要处理的文件列表。

origin\_data\_path (str): 原始数据路径。

new\_data\_path (str): 处理后数据路径。

image\_file\_path (str): 图像文件路径。

输出参数:

process\_data(file\_list, origin\_data\_path, new\_data\_path, image\_file\_path):返回从 JSON 文件中加载的数据。

# 实验

## 实验设置

选择了徐等人（Xu et al., 2022）提出的多模态隐喻数据集，该数据集包含从社交媒体收集的10,000个梗图图像。使用OCR方法从这些图像中提取文本信息，构建了多模态隐喻数据集，其中包括6,000个中文条目和4,000个英文条目。除了隐喻的分类标签外，它们还标注了隐喻的来源和相关的情感。

所有训练模型都设置了1e-5的学习率，批量大小为8，并在设有早停机制的情况下训练了100个epoch。数据集被随机打乱，并按照6:2:2的比例划分为训练集、验证集和测试集。

所有实验都在一个3090 24G GPU上进行。我们的方法的最终结果通过对五个不同的随机种子取平均值得到，平均单次运行时间在20-30分钟内。最后，模型的性能基于F1分数进行评估。

采用了低秩适应（LoRA Hu et al., 2021）微调方法对大型语言模型（LLMs）进行微调。所有设置都遵循了Alpaca-LoRA\*中使用的设置。

## 实验结果与分析

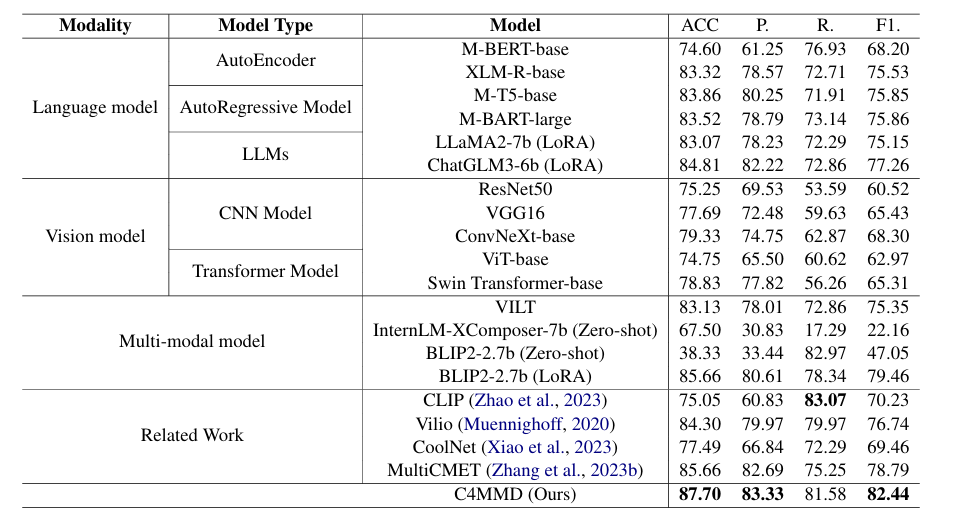
表1：不同方法对多模态隐喻检测任务的结果

表1显示了不同模型的功能多模态隐喻检测任务。在这里我们只对主要分类结果进行评价。我们没有评估两个子任务的结果yI和yT作为两个子任务，主要是设计服务于主要任务。我们的方法在中文和英文样本集中都取得了最好的结果。考虑到MLLM （InternLM-XComposer-7b）直接产生的结果，我们允许它进入直接为图像和文本生成额外的特征，有效地利用大型模型的功能。再加上下游的分类器，这方法产生了加性效应。多模态模型的性能差异很大，大多数模型都没有超越语言模型。这强调了文本的重要性识别多模态隐喻的情态。mllms在零射击场景中表现不佳，部分原因是我们设计的提示模板。

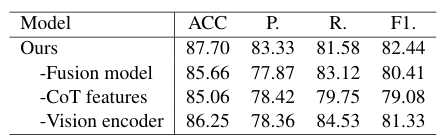


表2：模型中各部件的烧蚀研究论隐喻检测

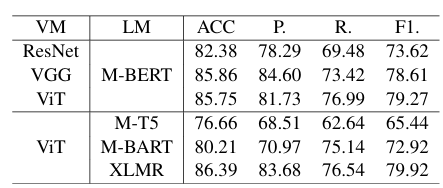


表3：不同语言和视觉的影响隐喻检测任务的模型组合VM为视觉模型，LM为语言模型。然后我们使用一个线性层来融合两者的特征模式

然而，主要原因是模型无法理解任务。鼓舞人心，过后好对BLIP2进行了优化，其性能超过了所有其他比较方法。这证明了图像和文本模式之间交互的好处任务和大模型如何能够有效地理解和处理这个任务后微调。在相关工作中，研究与我们的研究密切相关，如Zhang等人的研究（2023b）。Muennighoff（2020）等，都取得了有竞争力的成绩。然而，Xiao et al.（2023）的Twitter情绪分类与我们的任务有所不同，因此表现较弱。

表2显示了我们的消融实验后的模型。将模型中的融合结构替换为线性层导致显著减少表演。这表明了加法的必要性帮助模型下的传统融合结构支持MLLM生成的额外特征。此外，消除了CoT生成方法的传销，并完全依赖于一步生成方法，引起了更大的注意性能下降。这也表明cotmethod可以生成更好的附加功能，从而协助下游模型制作更准确的判断。有趣的是，模型的性能当我们移除图像时，它只略微倾斜处理模块。这表明MLLM能提供一定程度的视觉信息吗更小的型号，但更全面的形成仍然需要远见的贡献模型。

我们测试了多重视觉和模态融合过程中的文本模型。的语言模型在测试时统一设置为MBERT视觉模型和ViT的使用一致在测试语言模型时。从表3和表1中的数据来看，尽管在单模态环境下，视觉模型VGG和文本模型M-T5达到了最好的per性能，ViT和XLM-R的结合在模态融合上超越所有其他人。ResNet + MBERT和VGG+ mbertare也提出了基线模型Met-Meme （Xu et al., 2022）。根据结果，我们报告的结果和他们一样。

图 3 展示了不同规模的模型在我们的架构下的能力。考虑到改进率通常介于 0 和 1 之间，而模型大小通常在数亿之间，我们将所有模型大小除以4亿，使其比例介于 0 和 1 之间，这样就可 以在同一张图上同时显示模型大小和改进率。很明显，随着模型大小的增加，特别是当模型最初较小时，性能会逐渐得到明显改善。当模型太小时，额外的文本信息并不会产生积极的效果，反而有可能对模型的性能产生负面影响。只有当模型规模增大时，模型才有能力理解更长的上下文信息。

图4：案例研究的例子

进一步探讨我们建议的有效性模型中，我们从测试中选择两个例子数据集，如图4所示。第一个例子演示了一个以形象为导向的隐喻。通过直接比较印章和土豆，它描绘了看的后果太多可爱的海豹了。传销，通过它的un对图像的理解，准确识别海豹和土豆的相似之处，从而帮助下游模式正确的判断。在第二个示例中，MLLM确定了fea从图像和文本中提取图像，然后com把这些结合起来才能正确理解幽默在表情包中表达的意思。下游模型准确地识别出它不包含隐喻的功能。相比之下，方法缺乏从大模型中提取附加信息判断它是隐喻仅仅基于“像个淑女”这句话导致了错误判断。

# 结语

结论：这项研究旨在通过利用先进的MLLMs来解决多模态隐喻解释的挑战。作者设计了一个三步CoT提示方法，从图像和文本中提取更丰富的信息。MLLMs的增强知识对于帮助较小的模型理解每个模态内的隐喻特征以及模态融合至关重要。这项工作不仅推进了多模态隐喻检测，也为未来探索MLLMs在解决复杂语言和视觉挑战方面的潜力铺平了道路。

局限性：作者认为他们工作的主要局限性在于只在多语言梗图数据集上测试了他们的隐喻检测能力，并没有扩展到梗图数据集的其他子任务，如有害性检测，也没有扩展到其他多模态数据集的隐喻检测。然而，尽管缺乏实验数据，作者对他们工作在这些方向上的适用性有信心，这将是他们未来研究的一个焦点。

参考文献

[1]Khalid Alnajjar, Mika Hämäläinen, and Shuo Zhang.2022. Ring that bell: A corpus and method for multi-modal metaphor detection in videos. arXiv preprintarXiv:2301.01134.

[2]Kaiming He, Xiangyu Zhang, Shaoqing Ren, and JianSun. 2016. Deep residual learning for image recog-nition. In Proceedings of the IEEE conference oncomputer vision and pattern recognition, pages 770–778

[3]Naveen Badathala, Abisek Rajakumar Kalarani, Tejpals-ingh Siledar, and Pushpak Bhattacharyya. 2023. Amatch made in heaven: A multi-task framework forhyperbole and metaphor detection. arXiv preprintarXiv:2305.17480.

[4]Dan Hendrycks and Kevin Gimpel. 2016. Gaus-sian error linear units (gelus). arXiv preprintarXiv:1606.08415.

[5]Pere-Lluís Huguet Cabot, Verna Dankers, David Abadi,Agneta Fischer, and Ekaterina Shutova. 2020. Thepragmatics behind politics: Modelling metaphor,framing and emotion in political discourse. In Find-ings of the association for computational linguistics:emnlp 2020, pages 4479–4488.

[6]Edward J Hu, Yelong Shen, Phillip Wallis, ZeyuanAllen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang,

and Weizhu Chen. 2021. Lora: Low-rank adap-tation of large language models. arXiv preprintarXiv:2106.09685.

[7]Xianyang Chen, Chee Wee Leong, Michael Flor, andBeata Beigman Klebanov. 2020. Go ﬁgure! multi-task transformer-based architecture for metaphor de-tection using idioms: Ets team in 2020 metaphorshared task. In Proceedings of the second workshopon ﬁgurative language processing, pages 235–243.

[8]Gitit Kehat and James Pustejovsky. 2021. Neuralmetaphor detection with visibility embeddings. InProceedings of\* SEM 2021: The Tenth Joint Confer-ence on Lexical and Computational Semantics, pages222–228.

[9]Minjin Choi, Sunkyung Lee, Eunseong Choi, HeesooPark, Junhyuk Lee, Dongwon Lee, and Jongwuk Lee.Wonjae Kim, Bokyung Son, and Ildoo Kim. 2021. Vilt:Vision-and-language transformer without convolu-tion or region supervision. In International Con-ference on Machine Learning, pages 5583–5594.PMLR.

[10]Saisuresh Krishnakumaran and Xiaojin Zhu. 2007.Hunting elusive metaphors using lexical resources.In Proceedings of the Workshop on Computationalapproaches to Figurative Language, pages 13–20.

[11]Duong Le, My Thai, and Thien Nguyen. 2020. Multi-task learning for metaphor detection with graph con-volutional neural networks and word sense disam-biguation. In Proceedings of the AAAI conference on artiﬁcial intelligence, volume 34, pages 8139–8146.

[12]Shuqun Li, Liang Yang, Weidong He, Shiqi Zhang,Jingjie Zeng, and Hongfei Lin. 2021. Label-enhanced hierarchical contextualized representationfor sequential metaphor identiﬁcation. In Proceed-ings of the 2021 Conference on Empirical Methodsin Natural Language Processing, pages 3533–3543.

[13]Yucheng Li, Shun Wang, Chenghua Lin, andGuerin Frank. 2023b. Metaphor detection via ex-plicit basic meanings modelling. arXiv preprintarXiv:2305.17268.

[14]Yair Neuman, Dan Assaf, Yohai Cohen, Mark Last,Shlomo Argamon, Newton Howard, and OphirFrieder. 2013. Metaphor identiﬁcation in large texts corpora. PloS one, 8(4):e62343.

[15]Yair Neuman, Dan Assaf, Yohai Cohen, Mark Last,Shlomo Argamon, Newton Howard, and Ophir Frieder. 2013. Metaphor identiﬁcation in large textscorpora. PloS one, 8(4):e62343.

[16]Chang Su, Kechun Wu, and Yijiang Chen. 2021. En-hanced metaphor detection via incorporation of ex-ternal knowledge based on linguistic theories. InFindings of the Association for Computational Lin-guistics: ACL-IJCNLP 2021, pages 1280–1287.

[17]Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei,Zheng Zhang, Stephen Lin, and Baining Guo. 2021.Swin transformer: Hierarchical vision transformerusing shifted windows. In Proceedings of the IEEE/CVF international conference on computer vi-sion, pages 10012–10022.

[18]Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Fe-ichtenhofer, Trevor Darrell, and Saining Xie. 2022.A convnet for the 2020s. In Proceedings of the IEEE/CVF conference on computer vision and pat-tern recognition, pages 11976–11986.

[19]Yuan Tian, Nan Xu, Wenji Mao, and Daniel Zeng.2023a. Modeling conceptual attribute likeness anddomain inconsistency for metaphor detection. In Pro-ceedings of the 2023 Conference on Empirical Meth-ods in Natural Language Processing, pages 7736–7752.

[20]Rui Mao, Xiao Li, Kai He, Mengshi Ge, and ErikCambria. 2023. Metapro online: A computational metaphor processing online system. In Proceed-ings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 3: SystemDemonstrations), pages 127–135.

[21]Hugo Touvron, Louis Martin, Kevin Stone, Peter Al-bert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open founda- tion and ﬁne-tuned chat models. arXiv preprint arXiv:2307.09288.

[22]Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. Advances in neural information processing systems, 30.

[23]Jason Wei, Xuezhi Wang, Dale Schuurmans, MaartenBosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou,et al. 2022. Chain-of-thought prompting elicits rea-soning in large language models. Advances in neural information processing systems, 35:24824–24837.

[24]Linting Xue, Noah Constant, Adam Roberts, Mihir Kale,Rami Al-Rfou, Aditya Siddhant, Aditya Barua, andColin Raffel. 2020. mt5: A massively multilingualpre-trained text-to-text transformer. arXiv preprint arXiv:2010.11934.

附录A：阅读和学习过程的感想、体会、建议



结论：论文指出，多模态隐喻检测面临着文本数据质量差和对常识知识要求高两大挑战。这些挑战对传统的语言模型提出了新的要求。提出的C4MMD框架通过结合链式思考(CoT)方法和多模态大型语言模型(MLLMs)，有效地提高了小模型的隐喻检测能力，并且在MET-MEME数据集上超越了现有模型。

体会：论文展示了如何将先进的MLLMs和CoT方法应用于多模态隐喻检测，这是一种创新的应用方式，为其他NLP任务提供了新的思路。论文是第一次系统性地利用MLLMs进行隐喻检测任务，这表明在NLP领域，系统性的研究对于推动技术进步至关重要。

建议：考虑将C4MMD框架应用于更广泛的多模态数据集和其他NLP任务，如情感分析、文本摘要等，以验证其泛化能力。尝试不同的模态融合结构和辅助任务设计，以进一步提高模型性能和鲁棒性。在使用社交媒体数据集时，应注意数据的潜在偏见和有害内容，确保模型的输出不会传播负面信息。

附录B：

源码：

