GEM Gaussian Embedding Modeling for Out of Distribution Detection in GUI Agents

Yang

2025 年 7 月 7 日

1 Introduction

GUI Agent 遇到的 OOD 问题：

- Internalization OOD：在 ID 的环境中执行任务，但是错误的假设了存在不支持的功能 (操作的 APP 在 ID 中，但是具体功能不在)
- Extrapolation OOD：工作于动态的不断演变的环境中，之前没见过 (这个 APP 不在 ID 中)

GUI Agent 的 OOD 问题可能会引发安全问题，因为模型处理 OOD 问题的能力依赖于泛化能力，可能会出错造成损失，因此检测 OOD 问题很重要

GUI Agent 的 OOD detection 面临的问题：

- Complex Embedding Space：屏幕包含的 UI 元素复杂度远高于 MLLM 且用户指令的多样性也更高
- Envolving Environment：系统的更新和新 APP 的出现会导致 OOD 检测需要不断迭代

2 Challenges

2.1 Problem Formulation

一个 GUI Agent$F$ 首先经过 D_{ID} 的数据集的训练，然后给出 user instruction$x$ 和屏幕截图 s，之后根据这个输入，模型预测行为 a_t 并执行操作，直到完成任务或者超出最大步数限制。由此定义一个 OOD 检测函数：若 (s_t, x) 是 OOD，则 $f_{OOD}(s_t, x) = 1$

2.2 Popular OOD Detection Method

主要分类有两种：

- Embedding-based：模型当下的 ID 存在一个 embedding space(向量空间)，可以将输入放入模型的向量空间比较分布之间的差距，用欧式距离等进行衡量
- Uncertainty-based：通常用熵、置信度等描述不确定性的量来衡量

其中 Uncertainty-based 效果更差，模型很难说明其不确定性，原因在于 ID 和 OOD 数据的可分性很低。

上述两种方法依赖于 $YoudenIndex = argmax_t(TPR(t) - FPR(t))$，其中 TPR 与 FPR 分别代表 true positive rate 和 false positive rate。而 TP 代表输入是 ID，预测也是 ID，FN 代表输入是 ID，预测是 OOD，则 $TPR = \frac{TP}{TP+FN}$.

而真正训练的数据集具有复杂性和多样性，但是他们的 embedding space 在距离上表现出明显的 cluster(聚类) 现象，即同一距离出现大量样本，不同聚类之间具有明显距离差距

3 Algorithm

首先定义 encoder layerl_e，作用是将 (s_i, x_i) 映射到 embedding vector，即 $e_i = l_e(s_i, x_i)$，由此可以得到 $D_{embedding}$。然后计算该数据集的 centroidμ，即 $\mu = \frac{1}{k}\sum_{i=1}^{k} e_i$，然后计算 Euclid distance，即 $d_i = |e_i - \mu|_2$，得到 $D_{distance}$。显然距离分布不是一个典型的分布，所以我们假设分布是由 m 个不同的高斯分布混合而成，那么有 $p(d) = \sum_{j=1}^{m} \pi_j N(d|\mu_j, \sigma_j^2)$，且 $\sum_{j=1}^{m} \pi_j = 1$，其中 $N(d|\mu_j, \sigma_j^2)$ 代表该高斯分布概率密度函数代入 d 的取值。最后再计算 log-likelihood：$logL_m = \sum_{i=1}^{k} p(d_i)$。该公式的参数包括 $\{\pi_j, \mu_j, \sigma_j\}$，通过 EM 算法最大化 log-likelihood 得到。EM 算法首先计算 $\gamma_{ij} = \frac{\pi_j N(d_i|\mu_j, \sigma_j^2)}{p(d_i)}$，然后令 $\pi_j^{new} = \frac{1}{k}\gamma_{ij}$，$\mu_j^{new} = \frac{\sum_{i=1}^{k}\gamma_{ij}d_i}{\sum_{i=1}^{k}\gamma_{ij}}$，$\sigma_j^{2new} = \frac{\sum_{i=1}^{k}\gamma_{ij}(d_i - \mu_j^{new})^2}{\sum_{i=1}^{k}\gamma_{ij}}$，参数化 m 用 BIC：$BIC(m) = -2logL_m + mlogk$。最后使用一个区间 $[\mu_j - n\sigma_j, \mu_j + n\sigma_j]$，如果 d_i 落在某个区间里就认为不是 OOD

Algorithm 1: GEM Algorithm

Require : GUI agent $\mathcal{F}$, encoder layer l_e, ID dataset $\mathcal{D}_{\text{ID}} = \{(s_i, x_i)\}_{i=1}^{k}$
Ensure : OOD detection function f_{OOD}

$\mathcal{D}_{\text{embedding}} \leftarrow \emptyset$
for $i \leftarrow 1$ **to** k **do**
  $e_i \leftarrow l_e(s_i, x_i)$
  $\mathcal{D}_{\text{embedding}} \leftarrow \mathcal{D}_{\text{embedding}} \cup \{e_i\}$
end
$\mu \leftarrow \frac{1}{k} \sum_{i=1}^{k} e_i$
$\mathcal{D}_{\text{distance}} \leftarrow \emptyset$
for $i \leftarrow 1$ **to** k **do**
  $d_i \leftarrow \|e_i - \mu\|_2$
  $\mathcal{D}_{\text{distance}} \leftarrow \mathcal{D}_{\text{distance}} \cup \{d_i\}$
end
for $m \in \{1, \ldots, M\}$ **do**
  Initialize GMM parameters $\{\pi_j, \mu_j, \sigma_j^2\}_{j=1}^{m}$
  repeat
    for $i \leftarrow 1$ **to** k **do**
      for $j \leftarrow 1$ **to** m **do**
        $\gamma_{ij} \leftarrow \frac{\pi_j \mathcal{N}(d_i|\mu_j, \sigma_j^2)}{\sum_{l=1}^{m} \pi_l \mathcal{N}(d_i|\mu_l, \sigma_l^2)}$
      end
    end
    for $j \leftarrow 1$ **to** m **do**
      $\pi_j \leftarrow \frac{1}{k} \sum_{i=1}^{k} \gamma_{ij}$
      $\mu_j \leftarrow \frac{\sum_{i=1}^{k} \gamma_{ij} d_i}{\sum_{i=1}^{k} \gamma_{ij}}$
      $\sigma_j^2 \leftarrow \frac{\sum_{i=1}^{k} \gamma_{ij} (d_i - \mu_j)^2}{\sum_{i=1}^{k} \gamma_{ij}}$
    end
  until *convergence*
  $\log \mathcal{L}_m \leftarrow 0$
  for $i \leftarrow 1$ **to** k **do**
    $\ell_i \leftarrow \sum_{j=1}^{m} \pi_j \cdot \mathcal{N}(d_i \mid \mu_j, \sigma_j^2)$
    $\log \mathcal{L}_m \leftarrow \log \mathcal{L}_m + \log(\ell_i)$
  end
  $\text{BIC}(m) \leftarrow -2 \log \mathcal{L}_m + m \log k$
end
$m^* \leftarrow \arg\min_m \text{BIC}(m)$
Fit final GMM with m^* components: $\{(\pi_j, \mu_j, \sigma_j)\}_{j=1}^{m^*}$
for $j \leftarrow 1$ **to** m^* **do**
  Define ID interval $I_j = [\mu_j - 3\sigma_j,\ \mu_j + 3\sigma_j]$
end
Define $f_{\text{OOD}}(s, x)$ as:

$$f_{\text{OOD}}(s, x) = \begin{cases} 0, & \text{if } \|l_e(s, x) - \mu\|_2 \in \bigcup_{j=1}^{m^*} I_j \\ 1, & \text{otherwise} \end{cases}$$

return f_{OOD}

4 Result

该方法很好的区分了 ID 与 OOD，大部分准确率在 95% 以上。除此之外研究神经网络检测层数和检测 OOD 准确率的情况，结果是开始随着层数加深而上升，之后开始下降，且有趣的是集中在第 9 层达到峰值 (50% 的结果)，因素有两个，一个是特殊任务相关的特征，另一个是视觉或文本的特征，在层数上升时，前者重要性增加，后者降低，在 9 层附近达到平衡

5 Summary

本篇论文讲述了一种应用于区分 GUI Agent 测试数据 ID 和 OOD 的方法，首先根据训练的数据绘制分布，然后在 embedding 空间里对测试数据进行向量化，应用混合高斯分布，利用算法求参数最优化结果，然后利用区间分布判断测试数据属于 ID 还是 OOD，结果是显著提高了分辨的成功率