(19)国家知识产权局

(12)发明专利申请

(10)申请公布号 CN 118413402 A
(43)申请公布日 2024.07.30

(21)申请号 202410875232.5

(22)申请日 2024.07.02

(71)申请人 合肥城市云数据中心股份有限公司
地址 230031 安徽省合肥市高新区玉兰大道767号机电产业园西二路科大国祯大厦4楼

(72)发明人 丁正 谢飞 龚见强 杨大伟 张家文

(74)专利代理机构 合肥国和专利代理事务所(普通合伙) 34131
专利代理师 张祥骞

(51)Int.Cl.
H04L 9/40(2022.01)
G06F 18/214(2023.01)
G06F 16/955(2019.01)
G06F 18/2431(2023.01)
G06F 18/25(2023.01)
G06N 3/0464(2023.01)
H04L 61/4511(2022.01)

权利要求书8页 说明书18页 附图3页

(54)发明名称

一种基于大语言模型的恶意域名检测方法

(57)摘要

本发明涉及一种基于大语言模型的恶意域名检测方法,与现有技术相比解决了难以针对恶意域名进行检测的缺陷。本发明包括以下步骤:预训练数据集和微调训练数据集的构建;设定URL-BERT模型;URL-BERT模型的预训练;URL-BERT模型的微调;待检测域名的获得;恶意域名检测结果的获得。本发明采用了大语言模型BERT来处理恶意域名,利用大语言模型强大的语义理解能力,可以更好地捕捉域名中的隐含信息和语境,提高恶意域名的识别准确性。

CN 118413402 A

1.一种基于大语言模型的恶意域名检测方法,其特征在于,包括以下步骤:

11)预训练数据集和微调训练数据集的构建:构建预训练数据集和微调训练数据集,并进行数据预处理;

12)设定URL-BERT模型;

13)URL-BERT模型的预训练:利用预训练数据集对URL-BERT模型进行预训练;

14)URL-BERT模型的微调:利用微调训练数据集对预训练后的URL-BERT模型进行微调;

15)待检测域名的获得:获得待检测的域名;

16)恶意域名检测结果的获得:将待检测的域名输入微调后的URL-BERT模型,获得恶意域名检测结果。

2.根据权利要求1所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述预训练数据集和微调训练数据集的构建包括以下步骤:

21)设定预训练数据集,预训练数据集为无标记域名数据,无标记的域名数据从公开的DNS查询日志、WHOIS数据库来源获取;

22)设定微调训练数据集,微调训练数据集为有标记域名数据,有标记域名数据从网络安全公司或公开的恶意域名数据库中获得,其中正样本和负样本数量均衡;

23)对预训练数据集和微调训练数据集进行去重、去除无关字符、转换为小写字母、标准化步骤的预处理;

24)对预处理后的预训练数据集和微调训练数据集进行分词处理。

3.根据权利要求1所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述设定URL-BERT模型包括以下步骤:

31)设定URL-BERT模型包括输入层、URL-BERT预训练层、字符级嵌入特征提取层、标记级别嵌入特征提取层、特征融合层和全连接分类层;

设定输入层:输入层接收分词处理后的文本作为模型的输入;

32)设定URL-BERT预训练层:预训练阶段通过设定MLM任务,对大量的无标记域名数据进行训练,学习捕获域名文本中的丰富语义信息和结构特征;

33)设定字符级嵌入特征提取层:字符级嵌入提取层提取域名文本的字符级别嵌入,利用BiGRU生成双向的向量表示,其中包含了域名内部字符间的复杂关联信息;对微调训练数据集,利用双向门控单元BiGRU对有标记域名数据进行字符级别的嵌入提取,BiGRU通过利用两个不同方向的GRU,在前向和后向方向上结合隐藏层状态,从而实现双向信息的整合,得到域名字符级别的嵌入向量$\mathbf{train}_{\mathrm{char\ features}}$;

34)设定标记级别嵌入特征提取层,进行标记级别嵌入;

35)设定特征融合层:

特征融合层将提取的字符级别嵌入特征向量和标记级别嵌入特征向量进行拼接,然后引入多个 Transformer 层来捕获特征之间的复杂关系;

在Transformer层之间的异构交互模块,用于将字符级嵌入和标记级嵌入在每个Transformer层之后进行组合和分离,组合操作丰富了不同表示之间的关联性,分离操作保留了字符级和标记级特征的独立性,促进了模型在双通道区分方面的表现;

36)设定全连接分类层:全连接分类层用于微调整个URL-BERT模型以适应特定的恶意域名检测任务,通过将特征融合层输出的特征向量输入到全连接层中,通过反向传播算法

最小化损失函数。

4.根据权利要求1所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述URL-BERT模型的预训练包括以下步骤:

41)将预训练数据集分词处理后的文本输入URL-BERT模型;

42)URL-BERT预训练层进行Masked Language Model预训练,即MLM预训练:

设$\hat{T}$是随机用[MASK]替换T中的一小部分token的分词序列,$\hat{T}=\{t'_1,t'_2,\ldots,t'_{len(T)}\}$中有15%的token被随机进行[MASK],其中t'_i表示第i个token被随机替换,MLM预训练过程的目标是根据上下文来预测被[MASK]的token的内容;

将$\hat{T}$输入到URL-BERT模型中,$\hat{T}$所对应的隐层向量H, H由$\hat{T}$经过BERT的编码器编码得到;

421)编码器处理过程:编码输入层的输入数据是嵌入向量$\hat{T}$;

422)自注意力机制:

输入$\hat{T}$通过线性变换得到查询矩阵Q、键矩阵K和值矩阵V,其中W^q, W^k, W^v是编码器的超参数:

$$Q=\hat{T}\cdot W^q,$$

$$K=\hat{T}\cdot W^k,$$

$$V=\hat{T}\cdot W^v,$$

然后计算注意力权重,使用缩放点积注意力,这一步产生的输出包含对输入序列内部的全局依赖关系的表示,

其中$\sqrt{d_k}$是键向量的维度,用于缩放点积,防止出现梯度消失问题,$\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})$是注意力权重矩阵,$\mathrm{Attention}(Q,K,V)$是一个注意力头的输出,

$$\mathrm{Attention}(Q,K,V)=\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})V,$$

多头注意力的输出:

将多个头的输出拼接起来,然后通过另一个线性变换,公式如下,

$$X=\mathrm{Mutihead}(Q,K,V)=$$

$$\mathrm{Concat}(\mathrm{head}_1,\mathrm{head}_2,\ldots,\mathrm{head}_n)W'$$

其中,$\mathrm{head}_h=\mathrm{Attention}(Q_h,K_h,V_h)$是第h个头的输出,W是一个超参数矩阵;

423)前馈神经网络:

对于多头注意力的输出用一个前馈神经网络进一步处理特征表示,前馈网络由两个线性层和一个激活函数组成,

$$\mathrm{FFN}(X)=\mathrm{ReLU}(XW_1+b_1)W_2+b_2,$$

其中W_1、b_1、W_2、b_2是可学习的参数;

接着进行残差连接和层归一化,用于增强模型的训练稳定性和收敛速度:

$$H=\mathrm{LayerNorm}(X+\mathrm{FFN}(X)),$$

这里LayerNorm表示层归一化操作;

编码器输出层:最终输出编码器的编码序列H;

接着,H经过Sigmoid操作得到对应的token分布;

MLM过程的公式如下,

$$P(t') = \prod_{i=1}^{lenT} P\left(t_i' | \hat{T}, \theta\right)^{m_i},$$

其中$P(t')$表示目标概率分布,即整个序列$\hat{T}$的概率分布,$P\left(t_i' | \hat{T}, \theta\right)^{m_i}$表示给定序列所有token的情况下,第i个token的预测概率,θ表示模型参数,m_i表示第i个token是否被[MASK],如果第i个token被[MASK],则$m_i = 1$, 否则$m_i = 0$,目的是对被[MASK]的token进行加权,被[MASK]的token对整个序列的概率产生影响;

在MLM预训练中,目标是最大化模型对整个序列$\hat{T}$的预测准确率,基于这一预测,计算交叉熵损失函数,并通过反向传播算法更新模型参数θ,最终得到预训练好的模型M,

$$\min L(\theta) = -\sum \hat{y}_i \log(P_i),$$

其中,θ是模型参数,$L(\cdot)$是交叉熵损失函数,$\hat{y}_i$是第i个token的真实标签,如果该token被[MASK]即该token的真实值,P_i是第i个token的预测概率,即$P_i = P\left(t_i' | \hat{T}, \theta\right)^{m_i}$,通过反向传播算法最小化损失函数$L(\cdot)$,更新模型参数$\theta$,最终得到预训练好的URL-BERT模型M。

5.根据权利要求1所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述URL-BERT模型的微调包括以下步骤:

51)字符级嵌入特征提取层和标记级别嵌入特征提取层的训练,进行字符级别嵌入;

微调数据集经过分词后形成了分词序列T,一方面,T经过字符级别嵌入特征提取层嵌入,得到了嵌入向量$train_{char\ features}$;另一方面T经过标记级别嵌入特征提取层嵌入,即T经过预训练好的模型的嵌入,再加上注意力掩码向量train_attention_mask,进行位置嵌入处理,得到带位置嵌入的标记级别嵌入向量$train_{toke\ n_{features}}$;

最后,将标记级别嵌入向量$train_{toke\ n_{features}}$与字符级别嵌入向量$train_{char\ features}$连接在一起,即在$train_{char\ features}$向量末尾连接$train_{toke\ n_{features}}$向量,形成一个更长的嵌入向量$T_{train}$,

$$T_{train} = Concat(train_{cha\ r_{features}}, train_{toke\ n_{features}});$$

52)设定特征融合层:设定特征融合层为模型M',模型M'包含一个被冻结参数的模型M层、多个Transformer层、多个异构交互层、一个全连接层和一个Sigmoid层;通过模块间的交互,特征融合层能够捕获url字符级别和标记级别间的复杂关系;

进行特征融合层的训练:

521)设定被冻结参数的模型M层为预训练好的URL-BERT模型M,

522)设定Transformer模块:

Transformer由编码器和解码器组成,编码器用于处理输入数据,解码器负责生成输出,

编码器包括编码输入层、自注意力机制、多头注意力、前馈神经网络、残差连接和层归

一化、编码器输出层；

解码器包括解码输入层、自注意力机制、编码器-解码器注意力机制、前馈神经网络、残差连接和层归一化、解码器输出层；

5221)编码器处理过程:编码输入层的输入数据是嵌入向量T_{train}；

自注意力机制：

对于每个头h,将输入T_{train}通过线性变换得到查询矩阵$Q_h^{encoder}$、键矩阵$K_h^{encoder}$和值矩阵$V_h^{encoder}$,其中$W_h^{Q_e}$, $W_h^{K_e}$, $W_h^{V_e}$,h是编码器的超参数：

$$Q_h^{encoder} = T_{train} W_h^{Q_e},$$

$$K_h^{encoder} = T_{train} W_h^{K_e},$$

$$V_h^{encoder} = T_{train} W_h^{V_e},$$

接下来,对每个头计算注意力权重,使用缩放点积注意力,这一步产生的输出包含对输入序列内部的全局依赖关系的表示,

其中$\sqrt{d_k}$是键向量的维度,用于缩放点积,防止出现梯度消失问题,$Softmax(\frac{Q_h^{encoder} \cdot K_h^{encoder^T}}{\sqrt{d_k}})$是注意力权重矩阵,$Attention(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是一个头的注意力机制的输出,

$$Attention(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = Softmax(\frac{Q_h^{encoder} \cdot K_h^{encoder^T}}{\sqrt{d_k}}) V_h^{encoder},$$

多头注意力的输出：

将多个头的输出拼接起来,然后通过另一个线性变换,公式如下,

$$X = Mutihead(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = Concat(head_1, head_2, \ldots, head_H) W^0,$$

其中,$head_h = Attention(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是每个注意力头的输出,$W^0$是一个超参数矩阵；

前馈神经网络：

对于多头注意力的输出用一个前馈神经网络进一步处理特征表示,前馈网络由两个线性层和一个激活函数组成,

$$FFN(X) = ReLU(XW_1 + b_1)W_2 + b_2,$$

其中W_1、b_1、W_2、b_2是可学习的参数；

残差连接和层归一化:用于增强模型的训练稳定性和收敛速度,

$$Output\_encoder = LayerNorm(X + FFN(X)),$$

这里LayerNorm表示层归一化操作；

编码器输出层:最终输出编码器的编码序列$Output_{encoder}$；

5222)解码器处理过程：

解码输入层的输入数据为嵌入向量T_{train};

自注意力机制:T_{train}通过线性变换得到查询矩阵$Q_h^{decoder}$、键矩阵$K_h^{decoder}$和值矩阵$V_h^{decoder}$,其中$W_h^{Q_d}$, $W_h^{K_d}$, W_h^{d},h是解码器的超参数,

$$Q_h^{decoder} = T_{train} W_h^{Q_d},$$

$$K_h^{decoder} = T_{train} W_h^{K_d},$$

$$V_h^{decoder} = T_{train} W_h^{V_d},$$

对每个头计算注意力权重,使用缩放点积注意力;

将多个头的输出拼接起来,经过残差连接和层归一化得到解码器的自注意力输出$A_{decoder}$;

编码器-解码器注意力机制:解码器对编码器的输出进行注意力计算,以捕捉输入序列和输出序列之间的关系,允许解码器根据编码器的信息调整自身的注意力以生成目标序列;

对于解码器的每个位置i,使用解码器的输出$Q_h^{decoder}$作为查询,编码器的输出$K_h^{encoder}$、$V_h^{encoder}$作为键和值,计算注意力权重:

$$Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}) = Softmax(\frac{Q_h^{decoder} \cdot K_h^{encoder^T}}{\sqrt{d_k}}) V_h^{encoder},$$

$$A_{encoder\_decoder} = Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}),$$

多头注意力的输出:将解码器的自注意力输出和编码器-解码器注意力输出进行结合,得到最终的注意力输出;

将它们按元素加权相加,其表达式如下:

$$A = \alpha \cdot A_{decoder} + (1 - \alpha) \cdot A_{encoder\_decoder},$$

其中,最终的注意力输出为A,α是一个可学习的权重参数:

前馈神经网络:对最终的注意力输出为A进行一次前馈神经网络处理,以进一步提取和转换特征表示:

$$FFN(A) = ReLU(AW_3 + b_3)W_4 + b_4;$$

残差连接和层归一化:在前馈神经网络的输出上应用残差连接和层归一化,以增强模型的训练稳定性和收敛速度:

$$Z = Output\_decoder = LayerNorm(A + FFN(A)),$$

解码器输出:解码器的输出序列为Z;

523)设定异构交互模块:异构交互层中将标记级和字符级表示视为两种不同类型的特征表示,在每个Transformer转换层后,将这两种表示进行组合和分离操作;

全连接网络负责将标记和字符表示映射到相同的特征空间中,以确保它们具有一致的维度和语义表示,然后通过CNN层进行连接和集成,最后经过残差连接和层归一化操作;

5231)分离与组合处理:

将解码器输出的向量z按长度分离成标记级表示$t_i(x)$和字符级表示$h_i(x)$，i表示第i个位置的标记/字符表示,分别对$t_i(x)$和$h_i(x)$应用专用的全连接网络进行转换：

$$t_i'(x) = W_t * t_i(x) + b_t,$$

$$h_i'(x) = W_h * h_i(x) + b_h,$$

接着将转换后的标记和字符表示连接起来形成融合后的表示：

$$w_i(x) = [h_i'(x); t_i'(x)];$$

5232)CNN卷积:对融合后的表示$w_i(x)$应用CNN操作,用于集成特征,

$$m_{j,t} = \tanh(W_j * w_{t:t+s_j-1} + b_j),$$

其中,$w_{t:t+s_j-1}$表示从w_t到w_{t+s_j-1}的嵌入的拼接,s_j表示第j个滤波器的窗口大小;

5233)分离:将经过卷积后的组合表示$m_{j,t}$分离为两个不同通道的特征表示,

$$m_i^t = \mathrm{GELU}(W_{m_t} * m_i(x) + b_{m_t}),$$

$$m_i^h = \mathrm{GELU}(W_{m_h} * m_i(x) + b_{m_h}),$$

其中,GELU是激活函数,用于分离融合后的特征表示;

5234)残差连接和层归一化:将分离后的特征表示与初始标记和字符表示进行相加,以重组两个通道的表示：

$$T_i(x) = t_i(x) + m_i^t,$$

$$H_i(x) = h_i(x) + m_i^h,$$

最后对重组后的表示进行层归一化操作,以确保稳定性：

$$T_i(x) = \mathrm{LayerNorm}(T_i(x)),$$

$$H_i(x) = \mathrm{LayerNorm}(H_i(x)),$$

分离后的特征将连接后将得到下一层Transformer的输入;

524)设定全连接和Sigmoid 分类模块：

进行全连接层的训练：

将最后一层Transformer的输出Output_decoder传递给具有N个隐藏单元的全连接层,其数学表达如下：

$$y = \mathrm{Activation}(xW + b),$$

其中,$x = \mathrm{Output_decoder}$是输入向量,W和b是层的权重和偏置,Activation(·)是激活函数;

Sigmoid 用于将全连接层的输出映射到类别概率分布,Sigmoid层将全连接层的输出结果y传递给Sigmoiod函数,Sigmoid函数将输入的实数值映射到0到1之间的概率值,全连接层的输出为y,经过Sigmoid函数后的输出表示为：

$$\mathrm{Output} = \mathrm{sigmoid}(y);$$

53)URL-BERT模型微调过程：

将微调数据集的标记级别嵌入向量$\mathrm{train}_{\mathrm{token}_{\mathrm{features}}}$与字符级别嵌入向量

$train_{char\ features}$进行处理后的嵌入向量T_{train},使用预训练好的URL-BERT模型M构建出的微调模型M′,通过将数据输入模型M′,将输出结果与真实标签进行比较,然后计算损失函数,通过反向传播算法更新模型参数,迭代训练模型直至收敛,使得微调后的URL-BERT模型进行恶意域名分类。

6.根据权利要求2所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述对预处理后的预训练数据集和微调训练数据集进行分词处理包括以下步骤:

61)针对预训练数据集和微调训练数据集中的域名数据,分词操作将整个域名字符串按照其结构特性进行拆分,去除域名中的特殊字符,以“/”、“.”、“-”为分隔符对域名进行分词:

输入一个经过预处理后的域名D,定义一个分词函数f来表示这一过程:

$$f(D) = \{分词列表\},$$

分词函数f()输入是一个经过预处理的字符串形式的域名,输出是一个由该域名分词后的片段组成的分词列表T,其中$T = \{t_1, t_2, \ldots, t_i, \ldots, t_{len(T)}\}$, len(T)为T中单词内容的数量;其表达式如下:

$$T = f(url) = [t_1, t_2, \ldots, t_i, \ldots, t_{len(T)}],$$

经过初步分词后,利用BERT分词器对T进行嵌入处理;

62)进行词嵌入处理:为了标识输入序列的开始和结束,在T开头添加[CLS]标记,在结尾添加[SEP]标记,并将token序列长度限制为128,对于长度未达到128个标记的输入序列,使用特定的填充标记[PAD]来扩展序列长度,直至其达到128个标记的限制,得[[CLS],T,[SEP],[PAD],…,[PAD]];

63)进行段嵌入处理:将段嵌入全部设置为0,即:

[[CLS],T,[SEP],[PAD],…,[PAD]]与相同长度的[0,0,0,…,0,0]相加;

64)进行位置嵌入处理:位置嵌入根据T中token的不同位置信息编码位置向量,向URL-BERT模型引入token的位置关系,位置嵌入全部设置为0,即:

经过段嵌入处理的序列,与相同长度全为0的序列相加;

65)生成注意力掩码向量:使用注意力掩码向量区分输入序列中的实际标记与填充标记,

对于序列中的每个真实标记,其对应的注意力设定为1;对于每个填充标记[PAD],其对应的注意力设定为0;生成注意力掩码向量 train_attention_mask,其中1的个数为$len(T) + 2$,即:

经过位置嵌入处理的序列,与序列[1,1,1,…,0,0]相加。

7.根据权利要求5所述的一种基于大语言模型的恶意域名检测方法,其特征在于,所述进行字符级别嵌入包括以下步骤:

71)输入序列表示:首先,对于输入序列$T = \{t_1, t_2, \ldots, t_i, \ldots t_{len(T)}\}$,其中每个$t_i$是子词分词的标记,len(T)是序列中子词的总数,每个标记t_i由字符$c_1^i, \ldots, c_{n_i}^i$组成,其中n_i表示子词的长度,

字符输入的总长度表示为$N = \sum_{i=1}^{len(T)} n_i$;

72)字符向量生成:对于每个字符c_j^i,都生成一个字符嵌入向量e_j^i,

这通过将字符c_j^i与字符嵌入矩阵W_c相乘来完成,即$e_j^i = W_c \cdot c_j^i$,该过程将每个字符映射到一个高维空间中的向量表示;

73)BiGRU处理:生成的字符嵌入向量e_j^i被送入BiGRU中进行处理,BiGRU操作的结果是生成每个字符的隐藏状态向量h_j^i,

$h_j^i = BiGRU(e_j^i)$,隐藏状态向量h_j^i包含了字符的上下文信息;

74)构建字符级嵌入:对于每个标记t_i,其字符级嵌入向量组成了一个序列$\{h_1^i, h_2^i, \ldots, h_{n_i}^i\}$,这个序列被处理成一个标记级别的嵌入向量$h_i(x)$;

对于每个标记t_i,将其第一个和最后一个字符的隐藏状态向量进行连接,得到标记级嵌入向量$h_i(x) = [h_1^i(x); h_{n_i}^i(x)]$,通过对整个输入序列的字符进行处理,获得了每个标记的上下文字符嵌入向量,即域名字符级别的嵌入向量$train_{char\ features}$,其表达式如下:

$$train_{char\ features} = \{h_1(x), h_2(x), \ldots, h_i(x), \ldots h_{len(T)}(x)\}。$$

一种基于大语言模型的恶意域名检测方法

技术领域

[0001] 本发明涉及网络空间安全与人工智能技术领域,具体来说是一种基于大语言模型的恶意域名检测方法。

背景技术

[0002] 随着互联网的快速发展,网络安全遭受威胁,各类恶意网站和恶意软件泛滥成灾,许多用户和网站都容易受到安全威胁,其中,恶意域名是网络攻击的主要路径之一。每天涌现出数以万计的新域名,这些域名中除大量正常注册的良性域名(Benign Domain)外,还存在大量被恶意注册用于进行非法活动的恶意域名(Malicious Domain),恶意域名成为各种网络非法活动的主要攻击路径之一。

[0003] 域名作为桥梁,将用户与恶意代理(如勒索软件、恶意软件和木马)连接起来,恶意域名通常用于实施非法窃取和存储个人敏感信息、非法控制和管理受感染的主机等恶意行为。几乎所有网络攻击都依赖恶意域名的使用,域名系统受到攻击者的滥用。因此,如何有效地检测恶意域名是网络安全领域的热点和难点问题。

[0004] 现有的恶意域名检测方法主要分为基于规则、机器学习和深度学习三类。基于规则的方法通过黑白名单检测域名的合法性,这类方法简单直接;基于机器学习的方法从域名中提取特征(例如域名长度、不同IP地址的数量等),然后构建基于机器学习的分类器来区分良性域名和恶意域名;基于深度学习的方法通过深度神经网络对域名的文本进行编码建模。

[0005] 现有的恶意域名检测方法在一定程度上都取得了很好的效果。但是基于规则的方法面对爆炸性增长的恶意域名时效果有限;基于机器学习的方法严重依赖于专家知识和受限于特征表达能力,且攻击者可以通过添加最小特征扰动(改变域的长度或修改网络参数,如TTL)轻松绕过模型检测;基于深度学习的方法具有忽视了位置嵌入对域名向量的重要性以及缺乏可解释性等缺陷。

[0006] 域名通常由多个部分组成,如二级域名、顶级域名等,它们通过点(.)连接,如图2所示。去除域名中的“.”、“-”、等特殊字符,以“.”为分隔符对域名进行分词后得到域名的Token,深度学习方法通常将域名视为纯文本数据,忽略了域名中Token对于位置的敏感性,即相同的Token位于Domain/Top Level Domain(TLD)/Path时应具有不同的含义和表示方法。

[0007] 那么,如何基于域名的特殊文本信息,设计一种恶意域名检测方法已经成为急需解决的技术问题。

发明内容

[0008] 本发明的目的是为了解决现有技术中难以针对恶意域名进行检测的缺陷,提供一种基于大语言模型的恶意域名检测方法来解决上述问题。

[0009] 为了实现上述目的,本发明的技术方案如下:

[0010] 一种基于大语言模型的恶意域名检测方法,包括以下步骤:

[0011] 11)预训练数据集和微调训练数据集的构建:构建预训练数据集和微调训练数据集,并进行数据预处理;

[0012] 12)设定URL-BERT模型;

[0013] 13)URL-BERT模型的预训练:利用预训练数据集对URL-BERT模型进行预训练;

[0014] 14)URL-BERT模型的微调:利用微调训练数据集对预训练后的URL-BERT模型进行微调;

[0015] 15)待检测域名的获得:获得待检测的域名;

[0016] 16)恶意域名检测结果的获得:将待检测的域名输入微调后的URL-BERT模型,获得恶意域名检测结果。

[0017] 所述预训练数据集和微调训练数据集的构建包括以下步骤:

[0018] 21)设定预训练数据集,预训练数据集为无标记域名数据,无标记的域名数据从公开的DNS查询日志、WHOIS数据库来源获取;

[0019] 22)设定微调训练数据集,微调训练数据集为有标记域名数据,有标记域名数据从网络安全公司或公开的恶意域名数据库中获得,其中正样本和负样本数量均衡;

[0020] 23)对预训练数据集和微调训练数据集进行去重、去除无关字符、转换为小写字母、标准化步骤的预处理;

[0021] 24)对预处理后的预训练数据集和微调训练数据集进行分词处理。

[0022] 所述设定URL-BERT模型包括以下步骤:

[0023] 31)设定URL-BERT模型包括输入层、URL-BERT预训练层、字符级嵌入特征提取层、标记级别嵌入特征提取层、特征融合层和全连接分类层;

[0024] 设定输入层:输入层接收分词处理后的文本作为模型的输入;

[0025] 32)设定URL-BERT预训练层:预训练阶段通过设定MLM任务,对大量的无标记域名数据进行训练,学习捕获域名文本中的丰富语义信息和结构特征;

[0026] 33)设定字符级嵌入特征提取层:字符级嵌入提取层提取域名文本的字符级别嵌入,利用BiGRU生成双向的向量表示,其中包含了域名内部字符间的复杂关联信息;对微调训练数据集,利用双向门控单元BiGRU对有标记域名数据进行字符级别的嵌入提取,BiGRU通过利用两个不同方向的GRU,在前向和后向方向上结合隐藏层状态,从而实现双向信息的整合,得到域名字符级别的嵌入向量$\mathbf{train}_{\mathrm{char\ features}}$;

[0027] 34)设定标记级别嵌入特征提取层,进行标记级别嵌入;

[0028] 35)设定特征融合层:

[0029] 特征融合层将提取的字符级别嵌入特征向量和标记级别嵌入特征向量进行拼接,然后引入多个 Transformer 层来捕获特征之间的复杂关系;

[0030] 在Transformer层之间的异构交互模块,用于将字符级嵌入和标记级嵌入在每个Transformer层之后进行组合和分离,组合操作丰富了不同表示之间的关联性,分离操作保留了字符级和标记级特征的独立性,促进了模型在双通道区分方面的表现;

[0031] 36)设定全连接分类层:全连接分类层用于微调整个URL-BERT模型以适应特定的恶意域名检测任务,通过将特征融合层输出的特征向量输入到全连接层中,通过反向传播算法最小化损失函数。

[0032] 所述URL-BERT模型的预训练包括以下步骤：

[0033] 41)将预训练数据集分词处理后的文本输入URL-BERT模型；

[0034] 42)URL-BERT预训练层进行Masked Language Model预训练,即MLM预训练：

[0035] 设$\hat{T}$是随机用[MASK]替换T中的一小部分token的分词序列，$\hat{T}=\{t'_1,t'_2,\ldots,t'_{len(T)}\}$中有15%的token被随机进行[MASK],其中t'_i表示第i个token被随机替换,MLM预训练过程的目标是根据上下文来预测被[MASK]的token的内容；

[0036] 将$\hat{T}$输入到URL-BERT模型中,$\hat{T}$所对应的隐层向量H，H由$\hat{T}$经过BERT的编码器编码得到；

[0037] 421)编码器处理过程:编码输入层的输入数据是嵌入向量$\hat{T}$；

[0038] 422)自注意力机制：

[0039] 输入$\hat{T}$通过线性变换得到查询矩阵Q、键矩阵K和值矩阵V,其中W^q，W^k，W^v是编码器的超参数：

[0040] $Q=\hat{T}\cdot W^q$,

[0041] $K=\hat{T}\cdot W^k$,

[0042] $V=\hat{T}\cdot W^v$,

[0043] 然后计算注意力权重,使用缩放点积注意力,这一步产生的输出包含对输入序列内部的全局依赖关系的表示,

[0044] 其中$\sqrt{d_k}$是键向量的维度,用于缩放点积,防止出现梯度消失问题,$\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})$是注意力权重矩阵,$\mathrm{Attention}(Q,K,V)$是一个注意力头的输出,

[0045] $$\mathrm{Attention}(Q,K,V)=\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})V,$$

[0046] 多头注意力的输出：

[0047] 将多个头的输出拼接起来,然后通过另一个线性变换,公式如下,

[0048] $$X=\mathrm{Mutihead}(Q,K,V)=\mathrm{Concat}(head_1,head_2,\ldots,head_n)_W,$$

[0049] 其中,$head_h=\mathrm{Attention}(Q_h,K_h,V_h)$是第$h$个头的输出,$W$是一个超参数矩阵；

[0050] 423)前馈神经网络：

[0051] 对于多头注意力的输出用一个前馈神经网络进一步处理特征表示,前馈网络由两个线性层和一个激活函数组成,

[0052] $$\mathrm{FFN}(X)=\mathrm{ReLU}(XW_1+b_1)W_2+b_2,$$

[0053] 其中W_1、b_1、W_2、b_2是可学习的参数；

[0054] 接着进行残差连接和层归一化,用于增强模型的训练稳定性和收敛速度：

[0055] $$H=\mathrm{LayerNorm}(X+\mathrm{FFN}(X)),$$

[0056] 这里LayerNorm表示层归一化操作；

[0057] 编码器输出层:最终输出编码器的编码序列H；

[0058] 接着,H经过Sigmoid操作得到对应的token分布;

[0059] MLM过程的公式如下,

[0060] $$P(t') = \prod_{i=1}^{lenT} P(t_i'|\hat{T},\theta)^{m_i},$$

[0061] 其中$P(t')$表示目标概率分布,即整个序列$\hat{T}$的概率分布,$P(t_i'|\hat{T},\theta)^{m_i}$表示给定序列所有token的情况下,第i个token的预测概率,θ表示模型参数,m_i表示第i个token是否被[MASK],如果第i个token被[MASK],则$m_i = 1$, 否则$m_i = 0$,目的是对被[MASK]的token进行加权,被[MASK]的token对整个序列的概率产生影响;

[0062] 在MLM预训练中,目标是最大化模型对整个序列$\hat{T}$的预测准确率,基于这一预测,计算交叉熵损失函数,并通过反向传播算法更新模型参数θ,最终得到预训练好的模型M,

[0063] $$\min L(\theta) = -\sum \hat{y}_i \log(P_i),$$

[0064] 其中,θ是模型参数,$L(\cdot)$是交叉熵损失函数,$\hat{y}_i$是第i个token的真实标签,如果该token被[MASK]即该token的真实值,P_i是第i个token的预测概率,即$P_i = P(t_i'|\hat{T},\theta)^{m_i}$,通过反向传播算法最小化损失函数$L(\cdot)$,更新模型参数$\theta$,最终得到预训练好的URL-BERT模型$M$。

[0065] 所述URL-BERT模型的微调包括以下步骤:

[0066] 51)字符级嵌入特征提取层和标记级别嵌入特征提取层的训练,进行字符级别嵌入;

[0067] 微调数据集经过分词后形成了分词序列T,一方面,T经过字符级别嵌入特征提取层嵌入,得到了嵌入向量$train_{char\ features}$;另一方面T经过标记级别嵌入特征提取层嵌入,即T经过预训练好的模型的嵌入,再加上注意力掩码向量train_attention_mask,进行位置嵌入处理,得到带位置嵌入的标记级别嵌入向量$train_{toke\ n_{features}}$;

[0068] 最后,将标记级别嵌入向量$train_{toke\ n_{features}}$与字符级别嵌入向量$train_{char\ features}$连接在一起,即在$train_{char\ features}$向量末尾连接$train_{toke\ n_{features}}$向量,形成一个更长的嵌入向量$T_{train}$,

[0069] $$T_{train} = Concat(train_{char\ features}, train_{toke\ n_{features}});$$

[0070] 52)设定特征融合层:设定特征融合层为模型M',模型M'包含一个被冻结参数的模型M层、多个Transformer层、多个异构交互层、一个全连接层和一个Sigmoid层;通过模块间的交互,特征融合层能够捕获url字符级别和标记级别间的复杂关系;

[0071] 进行特征融合层的训练:

[0072] 521)设定被冻结参数的模型M层为预训练好的URL-BERT模型M,

[0073] 522)设定Transformer模块:

[0074] Transformer由编码器和解码器组成,编码器用于处理输入数据,解码器负责生成输出,

[0075] 编码器包括编码输入层、自注意力机制、多头注意力、前馈神经网络、残差连接和层归一化、编码器输出层;

[0076] 解码器包括解码输入层、自注意力机制、编码器-解码器注意力机制、前馈神经网络、残差连接和层归一化、解码器输出层；

[0077] 5221)编码器处理过程:编码输入层的输入数据是嵌入向量T_{train}；

[0078] 自注意力机制：

[0079] 对于每个头h,将输入T_{train}通过线性变换得到查询矩阵$Q_h^{encoder}$、键矩阵$K_h^{encoder}$和值矩阵$V_h^{encoder}$,其中$W_h^{Q_e}$, $W_h^{K_e}$, $W_h^{V_e}$,h是编码器的超参数：

[0080] $$Q_h^{encoder} = T_{train} W_h^{Q_e},$$

[0081] $$K_h^{encoder} = T_{train} W_h^{K_e},$$

[0082] $$V_h^{encoder} = T_{train} W_h^{V_e},$$

[0083] 接下来,对每个头计算注意力权重,使用缩放点积注意力,这一步产生的输出包含对输入序列内部的全局依赖关系的表示,

[0084] 其中$\sqrt{d_k}$是键向量的维度,用于缩放点积,防止出现梯度消失问题,$\mathrm{Softmax}(\frac{Q_h^{encoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}})$是注意力权重矩阵,$\mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是一个头的注意力机制的输出,

[0085] $$\mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = \mathrm{Softmax}(\frac{Q_h^{encoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}}) V_h^{encoder},$$

[0086] 多头注意力的输出：

[0087] 将多个头的输出拼接起来,然后通过另一个线性变换,公式如下,

[0088] $$X = \mathrm{Mutihead}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = \mathrm{Concat}(head_1, head_2, \ldots, head_H) W^O,$$

[0089] 其中,$head_h = \mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是每个注意力头的输出,$W^O$是一个超参数矩阵；

[0090] 前馈神经网络：

[0091] 对于多头注意力的输出用一个前馈神经网络进一步处理特征表示,前馈网络由两个线性层和一个激活函数组成,

[0092] $$\mathrm{FFN}(X) = \mathrm{ReLU}(XW_1 + b_1)W_2 + b_2,$$

[0093] 其中W_1、b_1、W_2、b_2是可学习的参数；

[0094] 残差连接和层归一化:用于增强模型的训练稳定性和收敛速度,

[0095] $$\mathrm{Output\_encoder} = \mathrm{LayerNorm}(X + \mathrm{FFN}(X)),$$

[0096] 这里LayerNorm表示层归一化操作；

[0097] 编码器输出层:最终输出编码器的编码序列$\mathrm{Output}_{encoder}$；

[0098] 5222)解码器处理过程：

[0099] 解码输入层的输入数据为嵌入向量T_{train}；

[0100] 自注意力机制:T_{train}通过线性变换得到查询矩阵$Q_h^{decoder}$、键矩阵$K_h^{decoder}$和值矩阵$V_h^{decoder}$,其中$W_h^{Q_d}$, $W_h^{K_d}$, W_h^{d},h是解码器的超参数,

[0101] $$Q_h^{decoder} = T_{train} W_h^{Q_d},$$

[0102] $$K_h^{decoder} = T_{train} W_h^{K_d},$$

[0103] $$V_h^{decoder} = T_{train} W_h^{V_d},$$

[0104] 对每个头计算注意力权重,使用缩放点积注意力;

[0105] 将多个头的输出拼接起来,经过残差连接和层归一化得到解码器的自注意力输出$A_{decoder}$;

[0106] 编码器-解码器注意力机制:解码器对编码器的输出进行注意力计算,以捕捉输入序列和输出序列之间的关系,允许解码器根据编码器的信息调整自身的注意力以生成目标序列;

[0107] 对于解码器的每个位置i,使用解码器的输出$Q_h^{decoder}$作为查询,编码器的输出$K_h^{encoder}$、$V_h^{encoder}$作为键和值,计算注意力权重:

[0108] $$Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}) = Softmax(\frac{Q_h^{decoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}}) V_h^{encoder},$$

[0109] $$A_{encode\ r\_decoder} = Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}),$$

[0110] 多头注意力的输出:将解码器的自注意力输出和编码器-解码器注意力输出进行结合,得到最终的注意力输出;

[0111] 将它们按元素加权相加,其表达式如下:

[0112] $$A = \alpha \cdot A_{decoder} + (1-\alpha) \cdot A_{encode\ r\_decoder},$$

[0113] 其中,最终的注意力输出为A,α是一个可学习的权重参数:

[0114] 前馈神经网络:对最终的注意力输出为A进行一次前馈神经网络处理,以进一步提取和转换特征表示:

[0115] $$FFN(A) = ReLU(AW_3 + b_3)W_4 + b_4;$$

[0116] 残差连接和层归一化:在前馈神经网络的输出上应用残差连接和层归一化,以增强模型的训练稳定性和收敛速度:

[0117] $$Z = Output\_decoder = LayerNorm(A + FFN(A)),$$

[0118] 解码器输出:解码器的输出序列为Z;

[0119] 523)设定异构交互模块:异构交互层中将标记级和字符级表示视为两种不同类型的特征表示,在每个Transformer转换层后,将这两种表示进行组合和分离操作;

[0120] 全连接网络负责将标记和字符表示映射到相同的特征空间中,以确保它们具有一致的维度和语义表示,然后通过CNN层进行连接和集成,最后经过残差连接和层归一化操作;

[0121] 5231)分离与组合处理:

[0122] 将解码器输出的向量Z按长度分离成标记级表示$t_i(x)$和字符级表示$h_i(x)$，i表示第i个位置的标记/字符表示,分别对$t_i(x)$和$h_i(x)$应用专用的全连接网络进行转换:

[0123] $$t_i'(x) = W_t * t_i(x) + b_t,$$

[0124] $$h_i'(x) = W_h * h_i(x) + b_h,$$

[0125] 接着将转换后的标记和字符表示连接起来形成融合后的表示:

[0126] $$w_i(x) = [h_i'(x); t_i'(x)];$$

[0127] 5232)CNN卷积:对融合后的表示$w_i(x)$应用CNN操作,用于集成特征,

[0128] $$m_{j,t} = \tanh(W_j * w_{t:t+s_j-1} + b_j),$$

[0129] 其中,$w_{t:t+s_j-1}$表示从w_t到w_{t+s_j-1}的嵌入的拼接,s_j表示第j个滤波器的窗口大小;

[0130] 5233)分离:将经过卷积后的组合表示$m_{j,t}$分离为两个不同通道的特征表示,

[0131] $$m_i^t = \mathrm{GELU}(W_{m_t} * m_i(x) + b_{m_t}),$$

[0132] $$m_i^h = \mathrm{GELU}(W_{m_h} * m_i(x) + b_{m_h}),$$

[0133] 其中,GELU是激活函数,用于分离融合后的特征表示;

[0134] 5234)残差连接和层归一化:将分离后的特征表示与初始标记和字符表示进行相加,以重组两个通道的表示:

[0135] $$T_i(x) = t_i(x) + m_i^t,$$

[0136] $$H_i(x) = h_i(x) + m_i^h,$$

[0137] 最后对重组后的表示进行层归一化操作,以确保稳定性:

[0138] $$T_i(x) = \mathrm{LayerNorm}(T_i(x)),$$

[0139] $$H_i(x) = \mathrm{LayerNorm}(H_i(x)),$$

[0140] 分离后的特征将连接后将得到下一层Transformer的输入;

[0141] 524)设定全连接和Sigmoid 分类模块:

[0142] 进行全连接层的训练:

[0143] 将最后一层Transformer的输出$\mathrm{Output_decoder}$传递给具有N个隐藏单元的全连接层,其数学表达如下:

[0144] $$y = \mathrm{Activation}(xW + b),$$

[0145] 其中,$x = \mathrm{Output_decoder}$是输入向量,$W$和$b$是层的权重和偏置,$\mathrm{Activation}(\cdot)$是激活函数;

[0146] Sigmoid 用于将全连接层的输出映射到类别概率分布,Sigmoid层将全连接层的输出结果y传递给Sigmoiod函数,Sigmoid函数将输入的实数值映射到0到1之间的概率值,全连接层的输出为y,经过Sigmoid函数后的输出表示为:

[0147] $$\mathrm{Output} = \mathrm{sigmoid}(y);$$

[0148] 53)URL-BERT模型微调过程：

[0149] 将微调数据集的标记级别嵌入向量$\mathrm{train}_{\mathrm{token\ features}}$与字符级别嵌入向量$\mathrm{train}_{\mathrm{char\ features}}$进行处理后的嵌入向量$\mathrm{T}_{\mathrm{train}}$，使用预训练好的URL-BERT模型$\mathrm{M}$构建出的微调模型$\mathrm{M}'$，通过将数据输入模型$\mathrm{M}'$，将输出结果与真实标签进行比较，然后计算损失函数，通过反向传播算法更新模型参数，迭代训练模型直至收敛，使得微调后的URL-BERT模型进行恶意域名分类。

[0150] 所述对预处理后的预训练数据集和微调训练数据集进行分词处理包括以下步骤：

[0151] 61)针对预训练数据集和微调训练数据集中的域名数据，分词操作将整个域名字符串按照其结构特性进行拆分，去除域名中的特殊字符，以“/”、“.”、“-”为分隔符对域名进行分词：

[0152] 输入一个经过预处理后的域名D，定义一个分词函数f来表示这一过程：

[0153] $\mathrm{f(D)} = \{\text{分词列表}\}$，

[0154] 分词函数f()输入是一个经过预处理的字符串形式的域名，输出是一个由该域名分词后的片段组成的分词列表T，其中$\mathrm{T} = \{\mathrm{t_1}, \mathrm{t_2}, \ldots, \mathrm{t_i}, \ldots, \mathrm{t_{len\,(T)}}\}$，$\mathrm{len(T)}$为$\mathrm{T}$中单词内容的数量；其表达式如下：

[0155] $$\mathrm{T} = \mathrm{f(url)} = [\mathrm{t_1}, \mathrm{t_2}, \ldots, \mathrm{t_i}, \ldots, \mathrm{t_{len\,(T)}}],$$

[0156] 经过初步分词后，利用BERT分词器对T进行嵌入处理；

[0157] 62)进行词嵌入处理：为了标识输入序列的开始和结束，在T开头添加[CLS]标记，在结尾添加[SEP]标记，并将token序列长度限制为128，对于长度未达到128个标记的输入序列，使用特定的填充标记[PAD]来扩展序列长度，直至其达到128个标记的限制，得$[[CLS], \mathrm{T}, [\mathrm{SEP}], [\mathrm{PAD}], \ldots, [\mathrm{PAD}]]$；

[0158] 63)进行段嵌入处理：将段嵌入全部设置为0，即：

[0159] $[[CLS], \mathrm{T}, [\mathrm{SEP}], [\mathrm{PAD}], \ldots, [\mathrm{PAD}]]$与相同长度的[0,0,0,…,0,0]相加；

[0160] 64)进行位置嵌入处理：位置嵌入根据T中token的不同位置信息编码位置向量，向URL-BERT模型引入token的位置关系，位置嵌入全部设置为0，即：

[0161] 经过段嵌入处理的序列，与相同长度全为0的序列相加；

[0162] 65)生成注意力掩码向量：使用注意力掩码向量区分输入序列中的实际标记与填充标记，

[0163] 对于序列中的每个真实标记，其对应的注意力设定为1；对于每个填充标记[PAD]，其对应的注意力设定为0；生成注意力掩码向量$\mathrm{train_attention_mask}$，其中1的个数为$len(\mathrm{T}) + 2$，即：

[0164] 经过位置嵌入处理的序列，与序列[1,1,1,…,0,0]相加。

[0165] 所述进行字符级别嵌入包括以下步骤：

[0166] 71)输入序列表示：首先，对于输入序列$\mathrm{T} = \{\mathrm{t_1}, \mathrm{t_2}, \ldots, \mathrm{t_i}, \ldots \mathrm{t_{len\,(T)}}\}$，其中每个$\mathrm{t_i}$是子词分词的标记，$\mathrm{len(T)}$是序列中子词的总数，每个标记$\mathrm{t_i}$由字符$\mathrm{c_1^i}, \ldots, \mathrm{c_{n_i}^i}$组成，其中$\mathrm{n_i}$表示子词的长度，

[0167] 字符输入的总长度表示为$\mathrm{N} = \sum_{\mathrm{i=1}}^{\mathrm{len\,(T)}} \mathrm{n_i}$；

[0168] 72)字符向量生成:对于每个字符c_j^i,都生成一个字符嵌入向量e_j^i,

[0169] 这通过将字符c_j^i与字符嵌入矩阵W_c相乘来完成,即$e_j^i = W_c \cdot c_j^i$,该过程将每个字符映射到一个高维空间中的向量表示;

[0170] 73)BiGRU处理:生成的字符嵌入向量e_j^i被送入BiGRU中进行处理,BiGRU操作的结果是生成每个字符的隐藏状态向量h_j^i,

[0171] $h_j^i = \mathrm{BiGRU}(e_j^i)$,隐藏状态向量$h_j^i$包含了字符的上下文信息;

[0172] 74)构建字符级嵌入:对于每个标记t_i,其字符级嵌入向量组成了一个序列$\{h_1^i, h_2^i, \ldots, h_{n_i}^i\}$,这个序列被处理成一个标记级别的嵌入向量$h_i(x)$;

[0173] 对于每个标记t_i,将其第一个和最后一个字符的隐藏状态向量进行连接,得到标记级嵌入向量$h_i(x) = [h_1^i(x); h_{n_i}^i(x)]$,通过对整个输入序列的字符进行处理,获得了每个标记的上下文字符嵌入向量,即域名字符级别的嵌入向量$\mathrm{train}_{\mathrm{char\ features}}$,其表达式如下:

[0174] $\mathrm{train}_{\mathrm{char\ features}} = \{h_1(x), h_2(x), \ldots, h_i(x), \ldots h_{\mathrm{len}(T)}(x)\}$。

[0175] 有益效果

[0176] 本发明的一种基于大语言模型的恶意域名检测方法,与现有技术相比采用了大语言模型BERT来处理恶意域名,利用大语言模型强大的语义理解能力,可以更好地捕捉域名中的隐含信息和语境,提高恶意域名的识别准确性。同时,在预训练时引入了位置嵌入技术,充分考虑了域名中Token的位置信息对于表示的重要性。通过将位置信息与Token的语义向量相结合,能够更准确地表达不同位置上的Token含义,从而提高了模型对于恶意域名的检测性能。

[0177] 本发明还利用BiGRU提取字符级别嵌入,充分利用双向信息,更好地捕捉更细粒度的上下文信息,从而更全面地理解url数据,提高嵌入的准确性和表达能力。

[0178] 在与传统的基于机器学习的特征提取方法相比,本发明可以避免手动设计特征,并且可以自动学习更高级别的特征表示。这种端到端的训练方式可以更好地适应数据的变化和复杂性,从而提高模型的泛化性能和可迁移性,提高应对对抗攻击的能力。与深度学习方法相比,本发明引入了位置嵌入,改善模型对序列数据的处理能力。同时还通过预训练任务以最大限度地提高模型的泛化能力和迁移能力,即使面对新的恶意域名分类任务,模型也能够快速适应并取得良好的效果。

[0179] 综上所述,基于大语言模型的恶意域名检测方法在准确性和普适性方面具有明显优势。

附图说明

[0180] 图1为本发明的方法顺序图;

[0181] 图2为现有技术中的URL结构图;

[0182] 图3为BiGRU网络的结构图;

[0183] 图4为异构交互层的结构图;

[0184] 图5为模型总体架构图。

具体实施方式

[0185] 为使对本发明的结构特征及所达成的功效有更进一步的了解与认识,用以较佳的实施例及附图配合详细的说明,说明如下:

[0186] 传统的基于特征工程的检测方法在面对复杂网络变化的时候难以取得很好的效果,其次,域名中相同的Token位于Domain/TLD/Path时应具有不同的含义和表示方法,即域名的Token具有位置敏感性。现有研究将域名视为纯文本序列,并应用语言模型来表征域名。许多研究使用“词袋”模型对域名标记进行建模,并提取词汇特征,例如使用n-gram特征用于域名分类,完全忽略了Token的位置敏感性。

[0187] 本发明通过无标记域名数据预训练BERT,再有效结合域名的字符级嵌入特征和标记级嵌入特征对URL-BERT进行微调。利用BERT模型的上下文理解能力和位置敏感性,捕捉域名深层的语义信息,避免了单纯基于特征的方法中对手工定义规则或特征的严重依赖。

[0188] 如图1所示,本发明所述的一种基于大语言模型的恶意域名检测方法,包括以下步骤:

[0189] 第一步,预训练数据集和微调训练数据集的构建:构建预训练数据集和微调训练数据集,并进行数据预处理。

[0190] (1)设定预训练数据集,预训练数据集为无标记域名数据,无标记的域名数据从公开的DNS查询日志、WHOIS数据库来源获取。

[0191] (2)设定微调训练数据集,微调训练数据集为有标记域名数据,有标记域名数据从网络安全公司或公开的恶意域名数据库中获得,其中正样本和负样本数量均衡。

[0192] (3)对预训练数据集和微调训练数据集进行去重、去除无关字符、转换为小写字母、标准化步骤的预处理。

[0193] (4)对预处理后的预训练数据集和微调训练数据集进行分词处理。其包括以下步骤:

[0194] 针对预训练数据集和微调训练数据集中的域名数据,分词操作将整个域名字符串按照其结构特性进行拆分,去除域名中的特殊字符,以“/”、“.”、“-”为分隔符对域名进行分词:

[0195] 输入一个经过预处理后的域名D,定义一个分词函数f来表示这一过程:

[0196] $$f(D) = \{分词列表\},$$

[0197] 分词函数f()输入是一个经过预处理的字符串形式的域名,输出是一个由该域名分词后的片段组成的分词列表T,其中$T = \{t_1, t_2, \ldots, t_i, \ldots, t_{len(T)}\}$, $len(T)$为T中单词内容的数量;其表达式如下:

[0198] $$T = f(url) = [t_1, t_2, \ldots, t_i, \ldots, t_{len(T)}],$$

[0199] 经过初步分词后,利用BERT分词器对T进行嵌入处理;

[0200] 进行词嵌入处理:为了标识输入序列的开始和结束,在T开头添加[CLS]标记,在结尾添加[SEP]标记,并将token序列长度限制为128,对于长度未达到128个标记的输入序列,使用特定的填充标记[PAD]来扩展序列长度,直至其达到128个标记的限制,得[[CLS],T,[SEP],[PAD],…,[PAD]];

[0201] 进行段嵌入处理:将段嵌入全部设置为0,即:

[0202] [[CLS],T,[SEP],[PAD],…,[PAD]]与相同长度的[0,0,0,…,0,0]相加;

[0203] 进行位置嵌入处理:位置嵌入根据$\mathbf{T}$中token的不同位置信息编码位置向量,向URL-BERT模型引入token的位置关系,位置嵌入全部设置为0,即:

[0204] 经过段嵌入处理的序列,与相同长度全为0的序列相加;

[0205] 生成注意力掩码向量:使用注意力掩码向量区分输入序列中的实际标记与填充标记,

[0206] 对于序列中的每个真实标记,其对应的注意力设定为1;对于每个填充标记[PAD],其对应的注意力设定为0;生成注意力掩码向量$\mathbf{train_attention_mask}$,其中1的个数为$len(\mathrm{T})+2$,即:

[0207] 经过位置嵌入处理的序列,与序列[1,1,1,…,0,0]相加。

[0208] 第二步,设定URL-BERT模型。

[0209] 在此,设计了模型的整体架构,考虑到域名文本的特殊性,综合考虑了字符级和标记级的特征提取、特征融合和分类等多个方面,设计了一个综合性的模型结构。

[0210] 在预训练阶段,URL-BERT模型通过MLM任务对大规模无标记域名数据进行预训练,学习捕获域名文本中的丰富语义信息和结构特征;在特征提取和融合阶段,设计了合适的特征提取方法来整合字符级和标记级的特征,并利用Transformer层来捕获特征之间的复杂关系,以提高模型的表征能力和分类性能。

[0211] 这种设计的好处在于能够综合利用不同级别的信息,从而提高模型的表征能力和分类性能。字符级嵌入特征可以捕捉域名文本的细粒度特征,如单个字符的语义信息和顺序关系;而标记级嵌入特征则能够捕捉更高级别的语义信息。通过特征融合,模型可以同时考虑到这两个层面的特征,使得模型能够更全面地理解域名文本的含义。

[0212] URL-BERT架构设计中,特征融合能够综合利用字符级和标记级的嵌入特征,使模型能够更全面地理解域名文本;而Transformer能够处理域名文本中的长距离依赖关系,并更好地捕捉上下文信息。这些优势共同作用,提高了URL-BERT模型在恶意域名检测任务中的准确性和泛化能力。

[0213] 其具体步骤如下:

[0214] (1)如图5所示,设定URL-BERT模型包括输入层、URL-BERT预训练层、字符级嵌入特征提取层、标记级别嵌入特征提取层、特征融合层和全连接分类层;

[0215] 设定输入层:输入层接收分词处理后的文本作为模型的输入。

[0216] (2)设定URL-BERT预训练层:预训练阶段通过设定MLM任务,对大量的无标记域名数据进行训练,学习捕获域名文本中的丰富语义信息和结构特征。

[0217] (3)设定字符级嵌入特征提取层:字符级嵌入提取层提取域名文本的字符级别嵌入,利用BiGRU生成双向的向量表示,其中包含了域名内部字符间的复杂关联信息;对微调训练数据集,如图3所示,利用双向门控单元BiGRU对有标记域名数据进行字符级别的嵌入提取,BiGRU通过利用两个不同方向的GRU,在前向和后向方向上结合隐藏层状态,从而实现双向信息的整合,得到域名字符级别的嵌入向量$\mathbf{train}_{\mathrm{char\ features}}$。

[0218] (4)设定标记级别嵌入特征提取层,进行标记级别嵌入。

[0219] (5)设定特征融合层:

[0220] 特征融合层将提取的字符级别嵌入特征向量和标记级别嵌入特征向量进行拼接,

然后引入多个 Transformer 层来捕获特征之间的复杂关系;

[0221] 在Transformer层之间的异构交互模块,用于将字符级嵌入和标记级嵌入在每个Transformer层之后进行组合和分离,组合操作丰富了不同表示之间的关联性,分离操作保留了字符级和标记级特征的独立性,促进了模型在双通道区分方面的表现。

[0222] (6)设定全连接分类层:全连接分类层用于微调整个URL-BERT模型以适应特定的恶意域名检测任务,通过将特征融合层输出的特征向量输入到全连接层中,通过反向传播算法最小化损失函数。

[0223] 第三步,URL-BERT模型的预训练:利用预训练数据集对URL-BERT模型进行预训练。

[0224] 本发明利用大量无标记的域名数据对BERT(Bidirectional Encoder Representations from Transformers)进行预训练得到URL-BERT模型,URL-BERT模型将通过自监督学习的方式学习域名数据中的丰富语义信息和结构特征,理解域名文本中的语义关系、词汇规律和结构特征,为后续的恶意域名检测任务提供更为深入和全面的语言理解能力。

[0225] 该预训练过程涉及到自然语言处理、深度学习等多个领域的知识。该提高了自然语言处理的应用于url的效果和效率,可以大幅提升模型的对于url文本理解的准确性和模型的下游泛化能力。同时预训练降低了模型的训练成本,缩短训练时间,并可以将已有的url数据库有效利用,提高数据的利用率。

[0226] (1)将预训练数据集分词处理后的文本输入URL-BERT模型。

[0227] (2)URL-BERT预训练层进行Masked Language Model预训练,即MLM预训练:

[0228] 设$\hat{\mathrm{T}}$是随机用[MASK]替换$\mathbf{T}$中的一小部分token的分词序列,$\hat{\mathrm{T}}=\{t'_1,t'_2,\ldots,t'_{len(T)}\}$中有15%的token被随机进行[MASK],其中t'_i表示第i个token被随机替换,MLM预训练过程的目标是根据上下文来预测被[MASK]的token的内容;

[0229] 将$\hat{\mathrm{T}}$输入到URL-BERT模型中,$\hat{\mathrm{T}}$所对应的隐层向量$\mathbf{H}$,$\mathbf{H}$由$\hat{\mathrm{T}}$经过BERT的编码器编码得到;

[0230] A1)编码器处理过程:编码输入层的输入数据是嵌入向量$\hat{\mathrm{T}}$;

[0231] A2)自注意力机制:

[0232] 将输入$\hat{\mathrm{T}}$通过线性变换得到查询矩阵Q、键矩阵K和值矩阵V,其中W^q, W^k, W^v是编码器的超参数:

[0233] $Q=\hat{\mathrm{T}}\cdot W^q$,

[0234] $\mathrm{K}=\hat{\mathrm{T}}\cdot W^k$,

[0235] $V=\hat{\mathrm{T}}\cdot W^v$,

[0236] 然后计算注意力权重,使用缩放点积注意力,这一步产生的输出包含对输入序列内部的全局依赖关系的表示,

[0237] 其中$\sqrt{d_k}$是键向量的维度,用于缩放点积,防止出现梯度消失问题,$\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})$是注意力权重矩阵,$\mathrm{Attention}(Q,K,V)$是一个头的注意力机制的输出,

[0238] $\mathrm{Attention}(Q,K,V)=\mathrm{Softmax}(\frac{Q,K^T}{\sqrt{d_k}})V$,

[0239] 多头注意力的输出:

[0240] 将多个头的输出拼接起来,然后通过另一个线性变换,公式如下,

[0241] $$X = \mathrm{Mutihead}(Q,K,V) = \mathrm{Concat}(\mathrm{head}_1,\mathrm{head}_2,\ldots,\mathrm{head}_n)W'$$

[0242] 其中,$\mathrm{head}_h = \mathrm{Attention}(Q_h,K_h,V_h)$是第$h$个头的输出,$W$是一个超参数矩阵;

[0243] A3)前馈神经网络:

[0244] 对于多头注意力的输出用一个前馈神经网络进一步处理特征表示,前馈网络由两个线性层和一个激活函数组成,

[0245] $$\mathrm{FFN}(X) = \mathrm{ReLU}(XW_1 + b_1)W_2 + b_2,$$

[0246] 其中W_1、b_1、W_2、b_2是可学习的参数;

[0247] 残差连接和层归一化:用于增强模型的训练稳定性和收敛速度,

[0248] $$H = \mathrm{LayerNorm}(X + \mathrm{FFN}(X)),$$

[0249] 这里LayerNorm表示层归一化操作;

[0250] 编码器输出层:最终输出编码器的编码序列H;

[0251] 接着,H经过Sigmoid操作得到对应的token分布;

[0252] MLM过程的公式如下,

[0253] $$P(t') = \prod_{i=1}^{lenT} P(t_i'|\hat{T},\theta)^{m_i},$$

[0254] 其中$P(t')$表示目标概率分布,即整个序列$\hat{T}$的概率分布,$P(t_i'|\hat{T},\theta)^{m_i}$表示给定序列所有token的情况下,第i个token的预测概率,θ表示模型参数,m_i表示第i个token是否被[MASK],如果第i个token被[MASK],则$m_i = 1$, 否则$m_i = 0$,目的是对被[MASK]的token进行加权,被[MASK]的token对整个序列的概率产生影响;

[0255] 在MLM预训练中,目标是最大化模型对整个序列$\hat{T}$的预测准确率,基于这一预测,计算交叉熵损失函数,并通过反向传播算法更新模型参数θ,最终得到预训练好的模型M,

[0256] $$\min L(\theta) = -\sum \hat{y}_i \log(P_i),$$

[0257] 其中,θ是模型参数,$L(\cdot)$是交叉熵损失函数,$\hat{y}_i$是第i个token的真实标签,如果该token被[MASK]即该token的真实值,P_i是第i个token的预测概率,即$P_i = P(t_i'|\hat{T},\theta)^{m_i}$,通过反向传播算法最小化损失函数$L(\cdot)$,更新模型参数$\theta$,最终得到预训练好的URL-BERT模型$M$。

[0258] 第四步,URL-BERT模型的微调:利用微调训练数据集对预训练后的URL-BERT模型进行微调。

[0259] 微调步骤需要对模型进行优化和调整,主要是针对特征融合层中各个模块进行训练,选择合适的超参数以及调整模型的学习力,达到最佳的微调效果。模型的微调过程在预训练的基础上,进一步提高了恶意域名检测的分类准确性, 通过特征融合和Transformer的应用,模型具有更好的特征表达和文本建模能力,可以加速恶意域名检测的分类效率。同时微调过程可以适用于多个领域,如应用于url的恶意家族分类、url多分类、判断url对关系、生成对抗样本等问题,具有广泛的应用前景。

[0260] 本方法通过微调,有效结合域名的字符级特征和标记级特征征。通过字符特征捕

捉域名文本在单个字符级别的顺序关系信息,不依赖语义理解的同时直观地反应了域名文本的重要信息;利用BERT模型的上下文理解能力和位置敏感性,通过标记级特征捕捉域名深层的语义信息,避免了单纯基于特征的方法中对手工定义规则或特征的严重依赖。

[0261] (1)字符级嵌入特征提取层和标记级别嵌入特征提取层的训练,进行字符级别嵌入;

[0262] 进行字符级别嵌入包括以下步骤:

[0263] 输入序列表示:首先,对于输入序列$T = \{t_1, t_2, \ldots, t_i, \ldots t_{len(T)}\}$,其中每个$t_i$是子词分词的标记,$len(T)$是序列中子词的总数,每个标记$t_i$由字符$c_1^i, \ldots, c_{n_i}^i$组成,其中$n_i$表示子词的长度,

[0264] 字符输入的总长度表示为$N = \sum_{i=1}^{len(T)} n_i$;

[0265] 字符向量生成:对于每个字符c_j^i,都生成一个字符嵌入向量e_j^i,

[0266] 这通过将字符c_j^i与字符嵌入矩阵W_c相乘来完成,即$e_j^i = W_c \cdot c_j^i$,该过程将每个字符映射到一个高维空间中的向量表示;

[0267] BiGRU处理:生成的字符嵌入向量e_j^i被送入BiGRU中进行处理,BiGRU操作的结果是生成每个字符的隐藏状态向量h_j^i,

[0268] $h_j^i = BiGRU(e_j^i)$,隐藏状态向量h_j^i包含了字符的上下文信息;

[0269] 构建字符级嵌入:对于每个标记t_i,其字符级嵌入向量组成了一个序列$\{h_1^i, h_2^i, \ldots, h_{n_i}^i\}$,这个序列被处理成一个标记级别的嵌入向量$h_i(x)$;

[0270] 对于每个标记t_i,将其第一个和最后一个字符的隐藏状态向量进行连接,得到标记级嵌入向量$h_i(x) = [h_1^i(x); h_{n_i}^i(x)]$,通过对整个输入序列的字符进行处理,获得了每个标记的上下文字符嵌入向量,即域名字符级别的嵌入向量$train_{char\ features}$,其表达式如下:

[0271] $train_{char\ features} = \{h_1(x), h_2(x), \ldots, h_i(x), \ldots h_{len(T)}(x)\}$。

[0272] 微调数据集经过分词后形成了分词序列T,一方面,T经过字符级别嵌入特征提取层嵌入,得到了嵌入向量$train_{char\ features}$;另一方面T经过标记级别嵌入特征提取层嵌入,即T经过预训练好的模型的嵌入,再加上注意力掩码向量train_attention_mask,进行位置嵌入处理,得到带位置嵌入的标记级别嵌入向量$train_{toke\ n\ features}$;

[0273] 最后,将标记级别嵌入向量$train_{toke\ n\ features}$与字符级别嵌入向量$train_{char\ features}$连接在一起,即在$train_{char\ features}$向量末尾连接$train_{toke\ n\ features}$向量,形成一个更长的嵌入向量$T_{train}$,

[0274] $T_{train} = Concat(train_{char\ features}, train_{toke\ n\ features})$。

[0275] (2)设定特征融合层:设定特征融合层为模型M',模型M'包含一个被冻结参数的模型M层、多个Transformer层、多个异构交互层、一个全连接层和一个Sigmoid层;通过模块间的交互,特征融合层能够捕获url字符级别和标记级别间的复杂关系;

[0276] 进行特征融合层的训练:

[0277] B1)设定被冻结参数的模型M层为预训练好的URL-BERT模型M，

[0278] B2)设定Transformer模块：

[0279] Transformer由编码器和解码器组成，编码器用于处理输入数据，解码器负责生成输出，

[0280] 编码器包括编码输入层、自注意力机制、多头注意力、前馈神经网络、残差连接和层归一化、编码器输出层；

[0281] 解码器包括解码输入层、自注意力机制、编码器-解码器注意力机制、前馈神经网络、残差连接和层归一化、解码器输出层；

[0282] B21)编码器处理过程：编码输入层的输入数据是嵌入向量T_{train}；

[0283] 自注意力机制：

[0284] 对于每个头h，将输入T_{train}通过线性变换得到查询矩阵$Q_h^{encoder}$、键矩阵$K_h^{encoder}$和值矩阵$V_h^{encoder}$，其中$W_h^{Q_e}$，$W_h^{K_e}$，$W_h^{V_e}$，h是编码器的超参数：

[0285] $$Q_h^{encoder} = T_{train} W_h^{Q_e},$$

[0286] $$K_h^{encoder} = T_{train} W_h^{K_e},$$

[0287] $$V_h^{encoder} = T_{train} W_h^{V_e},$$

[0288] 接下来，对每个头计算注意力权重，使用缩放点积注意力，这一步产生的输出包含对输入序列内部的全局依赖关系的表示，

[0289] 其中$\sqrt{d_k}$是键向量的维度，用于缩放点积，防止出现梯度消失问题。$\mathrm{Softmax}(\frac{Q_h^{encoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}})$是注意力权重矩阵，$\mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是一个头的注意力机制的输出，

[0290] $$\mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = \mathrm{Softmax}(\frac{Q_h^{encoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}}) V_h^{encoder}$$

[0291] 多头注意力的输出：

[0292] 将多个头的输出拼接起来，然后通过另一个线性变换，公式如下，

[0293] $$X = \mathrm{Mutihead}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder}) = \mathrm{Concat}(head_1, head_2, \ldots, head_H) W^O,$$

[0294] 其中，$head_h = \mathrm{Attention}(Q_h^{encoder}, K_h^{encoder}, V_h^{encoder})$是每个注意力头的输出，$W^O$是一个超参数矩阵；

[0295] 前馈神经网络：

[0296] 对于多头注意力的输出用一个前馈神经网络进一步处理特征表示，前馈网络由两个线性层和一个激活函数组成，

[0297] $$FFN(X) = \mathrm{ReLU}(XW_1 + b_1)W_2 + b_2,$$

[0298] 其中W_1、b_1、W_2、b_2是可学习的参数；

[0299] 残差连接和层归一化：用于增强模型的训练稳定性和收敛速度，

[0300] $Output\_encoder = LayerNorm(X + FFN(X))$,

[0301] 这里$LayerNorm$表示层归一化操作;

[0302] 编码器输出层:最终输出编码器的编码序列$Output_{encoder}$;

[0303] B22)解码器处理过程:

[0304] 解码输入层的输入数据为嵌入向量T_{train};

[0305] 自注意力机制:T_{train}通过线性变换得到查询矩阵$Q_h^{decoder}$、键矩阵$K_h^{decoder}$和值矩阵$V_h^{decoder}$,其中$W_h^{Q_d}$, $W_h^{K_d}$, W_h^d,h是解码器的超参数,

[0306] $$Q_h^{decoder} = T_{train} W_h^{Q_d},$$

[0307] $$K_h^{decoder} = T_{train} W_h^{K_d},$$

[0308] $$V_h^{decoder} = T_{train} W_h^{V_d},$$

[0309] 对每个头计算注意力权重,使用缩放点积注意力;

[0310] 将多个头的输出拼接起来,经过残差连接和层归一化得到解码器的自注意力输出$A_{decoder}$;

[0311] 编码器-解码器注意力机制:解码器对编码器的输出进行注意力计算,以捕捉输入序列和输出序列之间的关系,允许解码器根据编码器的信息调整自身的注意力以生成目标序列;

[0312] 对于解码器的每个位置i,使用解码器的输出$Q_h^{decoder}$作为查询,编码器的输出$K_h^{encoder}$、$V_h^{encoder}$作为键和值,计算注意力权重:

[0313] $$Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}) = Softmax(\frac{Q_h^{decoder} \cdot K_h^{encoder\ T}}{\sqrt{d_k}}) V_h^{encoder},$$

[0314] $$A_{encode\ r\_decoder} = Attention(Q_h^{decoder}, K_h^{encoder}, V_h^{encoder}),$$

[0315] 多头注意力的输出:将解码器的自注意力输出和编码器-解码器注意力输出进行结合,得到最终的注意力输出;

[0316] 将它们按元素加权相加,其表达式如下:

[0317] $$A = \alpha \cdot A_{decoder} + (1 - \alpha) \cdot A_{encode\ r\_decoder},$$

[0318] 其中,最终的注意力输出为A,α是一个可学习的权重参数:

[0319] 前馈神经网络:对最终的注意力输出为A进行一次前馈神经网络处理,以进一步提取和转换特征表示:

[0320] $FFN(A) = ReLU(AW_3 + b_3)W_4 + b_4$;

[0321] 残差连接和层归一化:在前馈神经网络的输出上应用残差连接和层归一化,以增强模型的训练稳定性和收敛速度:

[0322] $Z = Output\_decoder = LayerNorm(A + FFN(A))$,

[0323] 解码器输出:解码器的输出序列为Z;

[0324] B3)设定异构交互模块:如图4所示,异构交互层中将标记级和字符级表示视为两

种不同类型的特征表示,在每个Transformer转换层后,将这两种表示进行组合和分离操作;

[0325] 全连接网络负责将标记和字符表示映射到相同的特征空间中,以确保它们具有一致的维度和语义表示,然后通过CNN层进行连接和集成,最后经过残差连接和层归一化操作;

[0326] B31)分离与组合处理:

[0327] 将解码器输出的向量$\mathbf{Z}$按长度分离成标记级表示$\mathbf{t_i(x)}$和字符级表示$\mathbf{h_i(x)}$, $\mathbf{i}$表示第i个位置的标记/字符表示,分别对$\mathbf{t_i(x)}$和$\mathbf{h_i(x)}$应用专用的全连接层进行转换:

[0328] $$t_i'(x) = W_t * t_i(x) + b_t,$$

[0329] $$h_i'(x) = W_h * h_i(x) + b_h,$$

[0330] 接着将转换后的标记和字符表示连接起来形成融合后的表示:

[0331] $$w_i(x) = [h_i'(x); t_i'(x)];$$

[0332] B32)CNN卷积:对融合后的表示$\mathbf{w_i(x)}$应用CNN操作,用于集成特征,

[0333] $$m_{j,t} = \tanh(W_j * w_{t:t+s_j-1} + b_j),$$

[0334] 其中,$\mathbf{w_{t:t+s_j-1}}$表示从$\mathbf{w_t}$到$\mathbf{w_{t+s_j-1}}$的嵌入的拼接,$\mathbf{s_j}$表示第j个滤波器的窗口大小;

[0335] B33)分离:将经过卷积后的组合表示$\mathbf{m_{j,t}}$分离为两个不同通道的特征表示,

[0336] $$m_i^t = \mathrm{GELU}(W_{m_t} * m_i(x) + b_{m_t}),$$

[0337] $$m_i^h = \mathrm{GELU}(W_{m_h} * m_i(x) + b_{m_h}),$$

[0338] 其中,GELU是激活函数,用于分离融合后的特征表示;

[0339] B34)残差连接和层归一化:将分离后的特征表示与初始标记和字符表示进行相加,以重组两个通道的表示:

[0340] $$T_i(x) = t_i(x) + m_i^t,$$

[0341] $$H_i(x) = h_i(x) + m_i^h,$$

[0342] 最后对重组后的表示进行层归一化操作,以确保稳定性:

[0343] $$T_i(x) = \mathrm{LayerNorm}(T_i(x)),$$

[0344] $$H_i(x) = \mathrm{LayerNorm}(H_i(x)),$$

[0345] 分离后的特征将连接后将得到下一层Transformer的输入;

[0346] B4)设定全连接和Sigmoid 分类模块:

[0347] 进行全连接层的训练:

[0348] 将最后一层Transformer的输出Output_decoder传递给具有$\mathbf{N}$个隐藏单元的全连接层,其数学表达如下

[0349] $$y = \mathrm{Activation}(xW + b),$$

[0350] 其中,$x = \mathrm{Output_decoder}$是输入向量,$\mathbf{W}$和$\mathbf{b}$是层的权重和偏置,$\mathrm{Activation}(\cdot)$是激活函数;

[0351] Sigmoid 用于将全连接层的输出映射到类别概率分布,Sigmoid层将全连接层的

输出结果y传递给Sigmoiod函数,Sigmoid函数将输入的实数值映射到0到1之间的概率值,全连接层的输出为y,经过Sigmoid函数后的输出表示为:

[0352] $Output = sigmoid(y)$。

[0353] (3)URL-BERT模型微调过程:

[0354] 将微调数据集的标记级别嵌入向量$train_{toke\ n_{features}}$与字符级别嵌入向量$train_{char\ _{features}}$进行处理后的嵌入向量$T_{train}$,使用预训练好的URL-BERT模型$M$构建出的微调模型$M'$,通过将数据输入模型$M'$,将输出结果与真实标签进行比较,然后计算损失函数,通过反向传播算法更新模型参数,迭代训练模型直至收敛,使得微调后的URL-BERT模型进行恶意域名分类。

[0355] 第五步,待检测域名的获得:获得待检测的域名。

[0356] 第六步,恶意域名检测结果的获得:将待检测的域名输入微调后的URL-BERT模型,获得恶意域名检测结果。

[0357] 本发明提出了一种基于BERT模型的恶意域名检测方法。与传统的基于规则和黑名单的检测方法相比,本发明能够更深层次地理解域名的语义信息和上下文关系,从而大幅提高了恶意域名的识别准确率。通过结合字符级嵌入和标记级嵌入,本方法引入了的位置信息,还捕捉了不同维度间的信息关系,进一步增强了模型对于域名结构的理解能力;其次,预训练BERT模型作为标记级别特征提取器,可以利用大规模预训练数据中学到的丰富语言知识。通过引入特征融合层,本发明能够灵活适应恶意域名分类任务的特定需求,提供了一种既高效又准确的恶意域名检测方案。总体而言,本发明的方法有高效、准确、易于实施等优点,对提升网络安全防护水平有积极效果。

[0358] 本发明所述方法以域名为研究对象,域名作为一种特殊的文本,可以借助大型语言模型的强大表示能力,捕获域名字符级特征和标记级特征之间的相互关系,从不同尺度挖掘域名的特征,为恶意域名检测提供了创新性解决方案。

[0359] 以上显示和描述了本发明的基本原理、主要特征和本发明的优点。本行业的技术人员应该了解,本发明不受上述实施例的限制,上述实施例和说明书中描述的只是本发明的原理,在不脱离本发明精神和范围的前提下本发明还会有各种变化和改进,这些变化和改进都落入要求保护的本发明的范围内。本发明要求的保护范围由所附的权利要求书及其等同物界定。


预训练数据集和微调训练数据集的构建
设定URL-BERT模型
URL-BERT模型的预训练
URL-BERT模型的微调
待检测域名的获得
恶意域名检测结果的获得

图 1

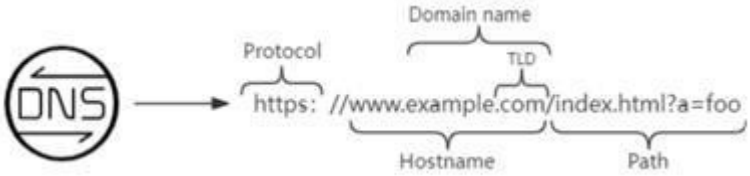
DNS
Protocol
Domain name
TLD
https://www.example.com/index.html?a=foo
Hostname
Path

图 2

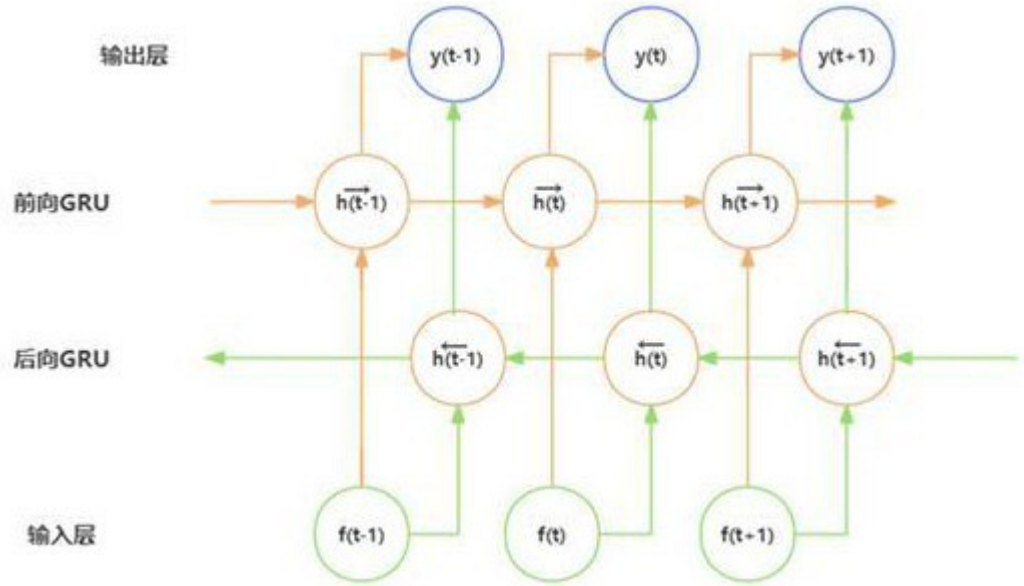
输出层
y(t-1)
y(t)
y(t+1)
前向GRU
h(t-1)
h(t)
h(t+1)
后向GRU
h(t-1)
h(t)
h(t+1)
输入层
f(t-1)
f(t)
f(t+1)

图 3

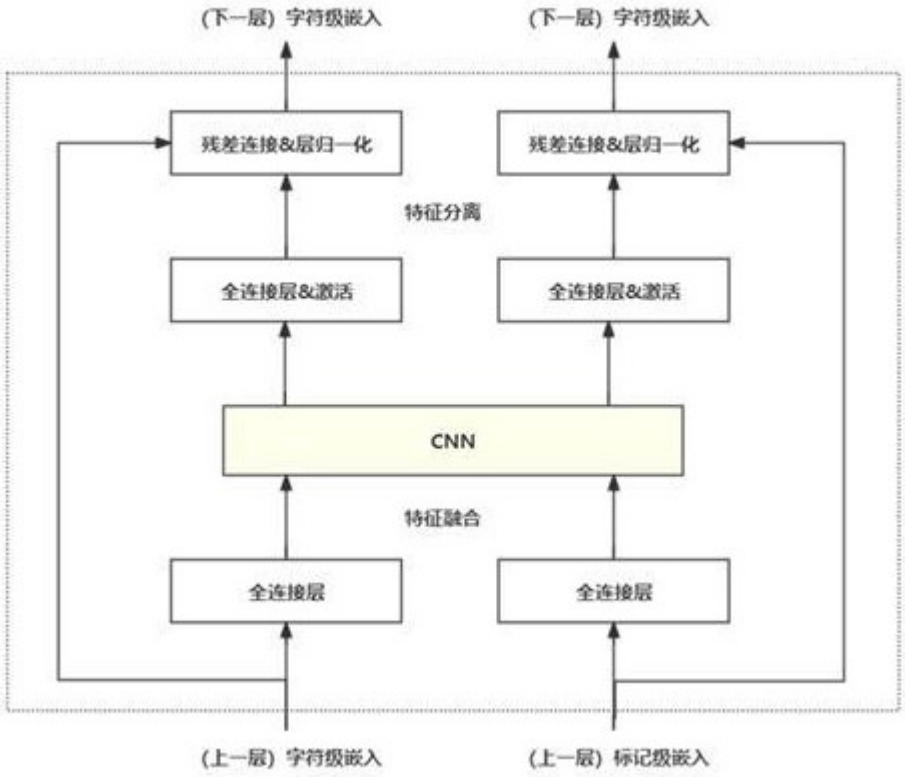
(下一层) 字符级嵌入
(下一层) 字符级嵌入
残差连接&层归一化
残差连接&层归一化
特征分离
全连接层&激活
全连接层&激活
CNN
特征融合
全连接层
全连接层
(上一层) 字符级嵌入
(上一层) 标记级嵌入

图 4

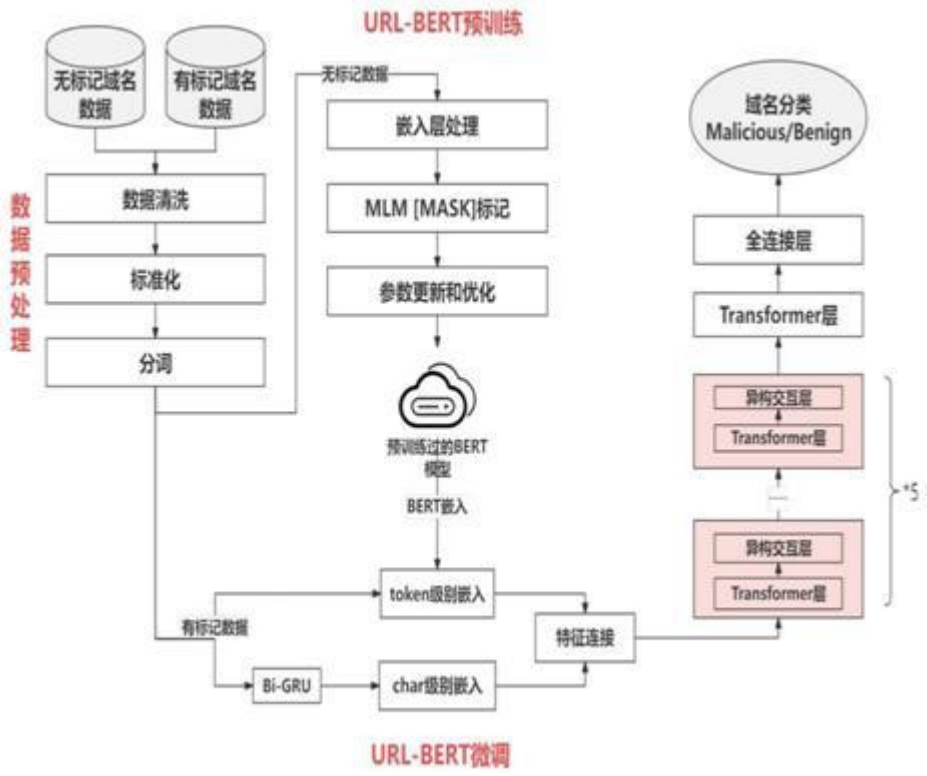
URL-BERT预训练
无标记域名数据
有标记域名数据
无标记数据
嵌入层处理
域名分类
Malicious/Benign
数据清洗
MLM [MASK]标记
全连接层
数据预处理
标准化
参数更新和优化
Transformer层
分词
异构交互层
Transformer层
预训练过的BERT模型
*5
BERT嵌入
异构交互层
Transformer层
token级别嵌入
特征连接
有标记数据
Bi-GRU
char级别嵌入
URL-BERT微调

图 5